Application Performance Management

State of the Art and Challenges for the Future

André van Hoorn
Mario Mann
Dušan Okanović
Christoph Heger

Tutorial @ 8th ACM/SPEC International Conference on Performance Engineering, April 22, 2017, L’Aquila, Italy
Who Are These Guys?

André van Hoorn  
University of Stuttgart  
Inst. of Softw. Techn. Reliable SW Systems

Mario Mann  
NovaTec Consulting  
Application Performance Management

Dušan Okanović  
University of Stuttgart  
Inst. of Softw. Techn. Reliable SW Systems

Christoph Heger  
NovaTec Consulting  
Application Performance Management
Performance Problems are Omnipresent

An unexpected error occurred
Try again later
Temporarily not available
We are experiencing heavy demand
Please visit us again later
... more capacity is on the way

Application Performance Management: State of the Art and Challenges for the Future
4/22/2017
Influence of Poor Performance on the Success of Businesses

- 50% of the online customers leave the website after load times > 2 seconds.
- Only 40% of the customers return after having experienced a performance problem.

Immediate response:
- 0.1 s: Users notice the delay, sense of flow gets lost

Users’ attention gets lost:
- 1.0 s: Users need to reorient themselves before each interaction
- 10.0 s: Response time [log 10]
“Application performance management (APM), as a core IT operations discipline, aims to achieve an adequate level of performance during operations. To achieve this,

APM comprises methods, techniques, and tools for

• **continuously monitoring** the state of an application system and its usage, as well as for
• **detecting, diagnosing, and resolving performance-related problems** using the monitored data.”

André van Hoorn und Stefan Siegl.
Application Performance Management (APM). Continuous Monitoring of Application Performance.
(Poster in ObjektSpektrum magazine; in German)
Order for free: http://www.sigs-datacom.de/wissen/fachposter.html
Einfluss von Performance auf Erfolg

Anwendungsperformance hat direkte Auswirkungen auf den Unternehmenserfolg:

- 2 Sekunden Ladezeit = 60% Umsatz
- 0,5 Sekunden Ladezeit = 95% Umsatz

Konsequenzen schlechter Performance

- Benutzerabbruch
- Anwenderzufriedenheit

Application Performance Management

Durch APM-Werkzeuge und -Prozesse kann Anwendungsperformance kontinuierlich überwacht und optimiert werden.

1. Collect
2. Process
3. Reason & use
4. Present

Zusammenarbeit aller Abteilungen

- Business
- Entwicklung
- Testen
- Betrieb
Collecting Data from All System Levels

- Agents collect data from all system levels
- On application level the agents are often technology-dependent

<table>
<thead>
<tr>
<th>Where?</th>
<th>What?</th>
<th>How?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Business</td>
<td>Sales data, conversion and bounce rate</td>
<td>Active</td>
</tr>
<tr>
<td>User</td>
<td>User interactions: length of stay, load times, errors; number of resources on HTML pages</td>
<td>Stimulation of the system by periodic requests.</td>
</tr>
<tr>
<td>Application</td>
<td>Component interactions, method response times, trace data</td>
<td>E.g., synthetic user transactions</td>
</tr>
<tr>
<td>Middleware</td>
<td>Queuing statistics, pooling, garbage collection</td>
<td></td>
</tr>
<tr>
<td>Operating System</td>
<td>File handling statistics, virtualization, thread statistics</td>
<td></td>
</tr>
<tr>
<td>Hardware</td>
<td>CPU load, memory consumption, I/O statistics</td>
<td></td>
</tr>
</tbody>
</table>
Trace-based Metrics (Selection)

<table>
<thead>
<tr>
<th>Metric</th>
<th>What?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Response Time</td>
<td>Component interactions, method response times, trace data</td>
</tr>
<tr>
<td>CPU Time</td>
<td></td>
</tr>
<tr>
<td>Method Name</td>
<td></td>
</tr>
<tr>
<td>Return Type</td>
<td></td>
</tr>
<tr>
<td>Logging Level</td>
<td></td>
</tr>
<tr>
<td>SQL Statement</td>
<td></td>
</tr>
<tr>
<td>Error Message</td>
<td></td>
</tr>
</tbody>
</table>

Monitoring (Measurement-based Performance Evaluation)

Application Performance Management: State of the Art and Challenges for the Future
Measurement Approaches and Techniques (Examples)

- **Event-driven measurement**
  - **Event**: a change in system state
    - e.g., incoming requests, access to HDD, operation execution, throwing exceptions
  - Calculate performance metrics whenever an event occurs
  - Simplest metric: counter

- **Tracing** (also event-driven)
  - Recording a certain part of the system state when an event occurs

- **Example: Code adjustments**
  - `printf` – easy, but with questionable maintainability
  - Bytecode engineering
  - Aspect-oriented Programming (AOP)

- **Example: Sampling**
  - Measurement (state or counter) is performed in specified time intervals

- Through overhead and increased resource consumption, measurements influence the system!
Data is collected from the system…

represented as **time series**…
Reconstructing Information from Data

- Data is collected from the system...
- represented as **time series**...
- ... and as **detailed execution traces**, and used to support problem analysis

<table>
<thead>
<tr>
<th>Trace</th>
<th>Response time</th>
</tr>
</thead>
<tbody>
<tr>
<td>▼ • doFilter(...)</td>
<td>[1793 ms]</td>
</tr>
<tr>
<td>▼ • searchTitleAndDescription(...)</td>
<td>[1730 ms]</td>
</tr>
<tr>
<td>▼ • searchTitleAndDescriptionWithOneWord(...)</td>
<td>[1632 ms]</td>
</tr>
<tr>
<td>▼ • list(...)</td>
<td>[352 ms]</td>
</tr>
<tr>
<td>▼ • executeQuery() {SQL: Select PROD_ID ...}</td>
<td>[143 ms]</td>
</tr>
<tr>
<td>▼ • executeQuery() {SQL: Select INV_ID ...}</td>
<td>[12 ms]</td>
</tr>
<tr>
<td>▼ • ...</td>
<td>...</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>
Traces in an APM Tool
• High quantity of information has to be pre-processed
• It has proven useful to use different views to show the data
• Views are navigable and can be categorized by both scope and detail level
Example: Application Topology Discovery and Visualization

Application Performance Management: State of the Art and Challenges for the Future
Manual or automated conclusions and actions can be derived from the information, e.g.,

- **Problem detection and alerting**
  - E.g., increased response times and resource utilization
  - Detection, for instance, based on thresholds and baselines

- **Problem diagnosis and root cause isolation**
  - E.g., N+1 problem, too many remote calls, poor DB queries
  - Detection based on monitoring information

- **System refactoring and adaptation**
  - E.g., auto-scaling in cloud-based architectures
Dynamic Software Analysis and Application Performance Management

- Monitoring Records
- Analysis Configuration (via API and WebGUI)
- Measurement
- Monitoring log/stream
- Pipes and Filters
- Software System with Monitoring Instrumentation
- Online and Offline Visualization

- [http://kieker-monitoring.net](http://kieker-monitoring.net)

*Application Performance Management: State of the Art and Challenges for the Future*
Institute of Software Technology
Reliable Software Systems

Application Performance Management
Part 2 – APM Tools

1. Collect
2. Process
3. Present
4. Reason & use

André van Hoorn
Mario Mann
Dušan Okanović
Christoph Heger

Tutorial @ 8th ACM/SPEC International Conference on Performance Engineering, April 22, 2017, L’Aquila, Italy
Contents

• Commercial APM tools
  • Magic quadrant
  • Timeline of APM tools
  • Goals of APM tools
  • Architecture
  • EUM, Server, Database Monitoring
  • How APM tools can help you to identify problems
• Open Source APM tools
Commercial APM Tools
Timeline of Tools

- **1998**: Wily – CA APM
- **2005**: DynaTrace
- **2006**: Kieker
- **2008**: AppDynamics
- **2017**: Cisco

### Commercial tools
- Wily – CA APM
- DynaTrace
- AppDynamics
- Cisco

### Open source tools
- inspectIT
- Kieker
Goals of APM Tools

End-User Perspective
End to End Monitoring
Smart Analytics
Lifecycle by Design

@Dynatrace

Application Performance Management: State of the Art and Challenges for the Future

4/22/2017
Sensors – How Do They Work?

- Every sensor (custom and OOTB) instruments Java/.NET methods
- Code gets added to each method that matches the sensor rule to
  - measure execution time
  - capture method arguments and return values
  - count method invocations
  - capture exceptions

Uninstrumented Application

Instrumented Application

@Dynatrace

Application Performance Management: State of the Art and Challenges for the Future

4/22/2017
Architecture – AppDynamics

Source:
https://docs.appdynamics.com/download/attachments/34271888/dbmondplussingleapp_appd_architecture.png?version=4&modificationDate=1487053783658&api=v2
Sensors – How Do They Work?
Application Overview – Dynatrace

Show the total Tier time with Process, Host, and Transaction

Deeper analysis

138 Active Visits
14.5% mobile
4 Applications

End User Experience
Analyze performance by geolocation, client types

Mobile Application
Inspect mobile visits, crashers

Application Performance Management: State of the Art and Challenges for the Future
4/22/2017
Application Overview – AppDynamics

Online_Retail

Dashboard | Events | Top Business Transactions | Transaction Snapshots | Transaction Score
--- | --- | --- | --- | ---

Application Flow Map

Tiers with technology

Remote Services

Databases

@AppDynamics

Application Performance Management: State of the Art and Challenges for the Future

4/22/2017
Real User Monitoring – AppDynamics

End User Monitoring

Application Performance Management

Infrastructure Visibility

iOS  Android  <HTML>  Python  C++  Java  node.js  .NET  PHP  SQL  NoSQL

@AppDynamics

Application Performance Management: State of the Art and Challenges for the Future  4/22/2017
Real User Monitoring – Dynatrace

Source: https://qph.ec.quoracdn.net/main-qimg-cd5a4b6e8cc29465d2cea98e055f606e.webp
End User Monitoring – AppDynamics

- Requests Time
- Crashes
- Geolocation

Application Performance Management: State of the Art and Challenges for the Future
4/22/2017
Server Monitoring – AppDynamics

- Availability
- CPU Usage
- Memory Usage
- Network
- Processes
- Volumes
Database Monitoring – AppDynamics
Dashboarding

easyTravel Triage shows easyTravel’s data from last 30 minutes

easyTravel Triage and System Activity

easyTravel Response Time

Database Health

I/O

Memory

CPU

Customer Web - Layer Activity Per Minute

B2B/Payment - Layer Activity Per Minute

Memory Consumption

CPU Activity

Database Activity Per Minute

WebRequest Network I/O Per Minute

@Dynatrace

Application Performance Management: State of the Art and Challenges for the Future  4/22/2017
How APM tools can help to identify problems?
How APM tools can help to identify problems?

Filter

Top findings to optimize performance
Challenges

- Share configurations between stages
- Identify business transactions and map this to functional level
Open Source APM tools
Weaknesses of Commercial APM Tools

- Licensing costs
  - Typical model: $x per agent → scaling?

Microservices

- Flexibility
  - Adaption to own needs
  - Bug fixing

Mobile Revolution

- Interoperability
  - Sources
  - Tools for analyse
  - Other APM tools

Internet of Things

- Sustainability
  - Stopp development
  - Change of strategy of APM vendor

Vendor Lock-In

Open Source

[Abbildung: http://blog.wso2.com]

Open Source APM tools

Monitoring & Application Deep Dive
- Scouter
- Kieker
- PINPOINT
- ZIPKIN
- Glowroot
- INSPECTIT

Performance Modeling
- OME
- MNet++

Load Testing
- Bucky
- PIWIK

Real User Monitoring

System & Resources Monitoring
- iCINGA
- influxdata
- sensu
- Prometheus
- MUNIN
- ZABBIX

Low-Level Performance Profiling
- JRat
- JMemProf
- Profiler4j

Web Performance Analysis
- YSlow
- Google Chrome Developer Tools
What is inspectIT?

Open Source

- Development since 2005
- Open Source since 2015

Platform

- Platform-principle
  - Integration with tools
  - Extensibility

Pareto-principle

- Focus on main functionality
- Experience of APM projects

Application Performance Management: State of the Art and Challenges for the Future
Gathering and Visualizing Timeseries Data

Data Collectors

- Kieker
- VisualVM
- Custom Code
- ... INSPECTIT

Graphing Tools

- Graphana
- Kibana
- graphite

Persistence

Time Series Databases

- InfluxDB
- graphite
- Prometheus

Query & Visualize

Application Performance Management: State of the Art and Challenges for the Future

4/22/2017
Dashboards

Application Performance Management: State of the Art and Challenges for the Future

4/22/2017
Dashboards
The Flaw of Averages

Gil Tene – https://www.youtube.com/watch?v=lJ8ydIuPFeU

The Flaw of Averages: Why We Underestimate Risk in the Face of Uncertainty
Sam L. Savage, with illustrations by Jeff Danziger – http://flawofaverages.com
Dashboards

Request Stats for Business Transaction: Login

- Request Count: 304
- Normal Requests Count: 154
- Anomal Request Count: 150
- Avg. Throughput: 0.08 req/s

Response Times

Slow Request Rate

Metric | Count | Min | Avg | Max | Total
--- | --- | --- | --- | --- | ---
duration (normal) | 154 | 95.15 ms | 1.96 s | 3.55 s | 5.04 min
duration (problem) | 150 | 59.99 ms | 4.26 s | 28.01 s | 11.41 min

Application Performance Management: State of the Art and Challenges for the Future 4/22/2017
Alerting

Alert Threshold

Define the threshold and check interval for the new alert definition.

Alert threshold: 600
Check interval [min]: 1
Send alerts to the following e-mail addresses:

marius.oehler@novatec-gmbh.de

New Channel

Name
Type
- Email
- PagerDuty
- Pushover
- Slack
- Threema Gateway
- VictorOps
- webhook

Email addresses

You can enter multiple email addresses using a ";" separator

Save
Send Test
Anomaly Detection & Alerting

Application Performance Management: State of the Art and Challenges for the Future
What is Web Performance Analysis about?

End User Experience?
- Satisfied
- Tolerating
- Frustrated

Performance
- Performance issues
- Functional Errors
- Network (Local ISPs, Mobile network carriers)
- Third party content providers

User Actions
- First user action
- Last user action
- Did the visit convert?
- Did the visit bounce?

Users
- Device
- Resolution
- Browser Versions
- Geolocation
Measuring End-User Experience

JAVA LAND 2017
28. bis 30. März 2017 im Phantasialand in Brühl
Testing and Monitoring Web Performance

Real User Testing
- External testing with all major
  - browsers,
  - operating systems,
  - mobile devices
  and real world data

Synthetic Monitoring
- Continuous external testing
- Known testing nodes
- Different Locations (ISP, Networks)
- No baseline traffic required
- Competitor Benchmark
- Availability testing (incl. 3rd Party)
  e.g., every 30 mins from different nodes

Real User Monitoring
- Real End Users
- Real Interaction
- Real traffic

QoS Validation

QoS Optimization

QoS Expectation

Anomaly
Server Monitoring

Source: https://www.icinga.com/wp-content/uploads/2016/08/Screen-Shot-2016-08-29-at-16.11.23.png
Benefits of Open Source Tools

- No licensing costs
- Growing community
- Pick the tools which work for your needs and combine them
Challenges

- From perspectives of industry and research…
Institute of Software Technology
Reliable Software Systems

Application Performance Management

Part 3 – APM Advanced

André van Hoorn
Mario Mann
Dušan Okanović
Christoph Heger

Tutorial @ 8th ACM/SPEC International Conference on Performance Engineering,
April 22, 2017, L’Aquila, Italy
Problem Diagnosis

Detect problem
Interpretation of measurements
Component Deep-Dive
Problem: Too Many Traces to Analyze
diagnoseIT Overview
http://diagnoseit.github.io/

*Expert-guided automatic diagnosis of performance problems in enterprise applications.*
Agenda

- APM interoperability
- Anti-patterns and trace based detection
- Mobile
- Overhead control
- Results aggregation
- Business integration
APM Interoperability

Convert .dat to .itds

X

Convert .itds to .map

Kieker

InspectIT

DiagnoseIt

PMX

iObserve2
APM Interoperability

Solution
# Data Export in APM Tools

<table>
<thead>
<tr>
<th>Open Source</th>
<th>Export possible</th>
</tr>
</thead>
<tbody>
<tr>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>✘</td>
<td>✔</td>
</tr>
<tr>
<td>✘</td>
<td>✔</td>
</tr>
<tr>
<td>✘</td>
<td>✘</td>
</tr>
<tr>
<td>✘</td>
<td>✘</td>
</tr>
<tr>
<td>✔</td>
<td>✘</td>
</tr>
<tr>
<td>✔</td>
<td>✘</td>
</tr>
<tr>
<td>✔</td>
<td>✘</td>
</tr>
<tr>
<td>✔</td>
<td>✘</td>
</tr>
<tr>
<td>✔</td>
<td>✘</td>
</tr>
<tr>
<td>✔</td>
<td>✘</td>
</tr>
<tr>
<td>✔</td>
<td>✘</td>
</tr>
</tbody>
</table>

- **Kieker**
- **INSPECTIT**
- **AppDynamics**
- **CA Technologies**
- **Dynatrace**
- **New Relic**
- **Riverbed**
- **IBM**
## Comparison table (excerpt)

<table>
<thead>
<tr>
<th>Metric</th>
<th>OPEN.xtrace</th>
<th>Kieker</th>
<th>inspectIT</th>
<th>AppDynamics</th>
<th>Dynatrace</th>
<th>New Relic</th>
</tr>
</thead>
<tbody>
<tr>
<td>Response Time</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>CPU Time</td>
<td>✔</td>
<td></td>
<td></td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>Method Name</td>
<td>✔</td>
<td>✔</td>
<td></td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>Return Type</td>
<td>✔</td>
<td>✔</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Logging Level</td>
<td>✔</td>
<td></td>
<td></td>
<td>✔</td>
<td></td>
<td>✔</td>
</tr>
<tr>
<td>SQL Statement</td>
<td>✔</td>
<td></td>
<td></td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>Error Message</td>
<td>✔</td>
<td></td>
<td></td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
</tbody>
</table>

*Towards performance tooling interoperability: An open format for representing execution traces.*
OPEN.xtrace Traces

- Execution trace is a data structure that captures control flow of method execution for a request served by the system (Ammons et al., 1997)

- Callables
  - Method executions
  - DB calls
  - Remote calls
  - Logging
  - Errors

Diagram:

- Trace
  - SubTrace
    - Location
      - Callable
        - Callable
          - Callable
            - SubTrace
              - ...

OPEN.xtrace

- What is available:
  - Trace model based on the available data in these tools
  - Adapters to convert the data between OPEN.xtrace and tools
  - Serialization

- Planned: OPEN.timeseries

- More on OPEN.xtrace
  - https://research.spec.org/apm-interoperability/
  - https://github.com/spec-rgdevops
OpenTracing

- Track what is going on in a complex, heterogeneous systems
- Capture important events during processing of client request
- Avoid vendor lock-in
- Wide industry support ([http://opentracing.io](http://opentracing.io))
diagnoseIT Overview

**Anti-patterns** are conceptually similar to patterns, however describe recurrent solutions to design problems which, however, may have a negative effect on different software quality attributes.

Example 1: One-Lane Bridge
[Smith and Williams, 2001]

• Problem

„[...] a point in the execution where one, or only a few, processes may continue to execute concurrently. All other processes must wait. […]”

• Cause

• „It frequently occurs in applications that access a database. Here, a lock ensures that only one process may update the associated portion of the database at a time.

• It may also occur when a set of processes make a synchronous call to another process that is not multi-threaded; all of the processes making synchronous calls must take turns “crossing the bridge.”“

• Solution

“Shared Resources principle[:] responsiveness improves when we minimize the scheduling time plus the holding time. Holding time is reduced by reducing the service time for the One-Lane Bridge, and by rerouting the work.”
Example 1: One-Lane Bridge – Causes

• Synchronization in source-code:

```java
public synchronized void buyItems (Collection<Item> items) {
    ...
}
```

• Thread- and connection- pools in application servers

```html
<Connector ... maxThreads="300" acceptCount="150" ...
```
Example 1: One-Lane Bridge – Symptoms

![Graph showing response time and CPU utilization vs. number of users.]

Application Performance Management: State of the Art and Challenges for the Future

4/22/2017
Example 2: N+1-Problem

Response time: 175,377.7 ms
response time > p% of T

Response time: 149,154.9 ms

Response time: 1,290.8 ms
Query: select * from Person where ...

Response time: 0.244 ms
Query: select * from Order where ...

Response time: 0.329 ms
Query: select * from Order where ...

Response time: 0.108 ms
Query: select * from Order where ...
Performance-Antipatterns – Classifications


Analysis automation

Trace

Instrumentation Refinement Request

Domain- and technology-specific analysis

Generic analysis

slowest method

many remote calls

many SQL calls

N+1 problem

Result

JDBC

Hibernate
List of Currently Detectable Antipatterns

- N+1 Query Problem
- Circuitous Treasure Hunt
- The Stifle
- The Ramp
- Traffic Jam
- More is Less
- Application Hiccups
- Garbage Collection Hiccups
Detecting Antipatterns in Mobile Environment

Java Agent

iOS Agent

OpenTracing spans

Buffer

OpenTracing spans

OPEN.xtrace

GUI

Application Performance Management: State of the Art and Challenges for the Future 4/22/2017
Antipatterns in Mobile Environment

- **Anti-patterns**
  - Too many remote calls
  - Too many remote calls to the same URL
  - Too many remote calls to the same server
  - Hard-disk utilization too high
  - Hard-disk usage increases too fast (ramp)
  - RAM utilization too high
  - RAM usage increases too fast (ramp)
  - Response is available after timeout (timeout occurred too early)
- **Glitches**
  - No connection available during a request
  - Latency between back end and mobile device is too high
diagnoseIT Overview

- Node 1
  - JVM
    - App 1
    - App 2

- Node n

- Monitoring Tool
  - Dynamic Instrumentation
  - API Labeling
  - Location Identification

- Traces
- Instrumentation Refinement Request
- Result Query

- Result

Application Performance Management: State of the Art and Challenges for the Future 4/22/2017
Overhead Control

- diagnoseIT can request more data from the monitoring tool
- Tool has to support changing of monitoring at runtime
- Instrumentation Quality Monitor can deny this request
- How to assess?
  - Modeling?
  - Machine learning?
diagnoseIT Overview

- Monitoring Tool
  - Dynamic Instrumentation
  - API Labeling
  - Location Identification
  - Instrumentation Quality Manager

- Traces
- Instrumentation Refinement Request
- Result Query
- Result

SUT

Node 1
- JVM
  - App 1
  - App 2

Node n

Application Performance Management: State of the Art and Challenges for the Future
4/22/2017
Application Performance Management: State of the Art and Challenges for the Future
Aggregating the Results

Categorization Process

T. Angerstein, D. Okanović, C. Heger, A. van Hoorn, A. Kovačević, T. Kluge: 
*Many flies in one swat: Automated categorization of performance problem diagnosis results.*
In: Proc. ACM/SPEC ICPE ’17
Aggregation in diagnoseIT

<table>
<thead>
<tr>
<th>Problem Overview</th>
<th>Severity</th>
<th># Occurrences</th>
</tr>
</thead>
<tbody>
<tr>
<td>Business Transaction: DVDstore02_search</td>
<td>high</td>
<td>1389</td>
</tr>
<tr>
<td>Node Type: DVDStore-Application</td>
<td>high</td>
<td>1389</td>
</tr>
<tr>
<td>Entry Point: HTTP POST (/dvdstore/browse)</td>
<td>high</td>
<td>1389</td>
</tr>
</tbody>
</table>

- Problem Instance: FullTextSearchAction.searchTitleAndDescription(...) | high | 14 |
- Problem Instance: SQL (select product0.PROD_ID as PROD1_5_0, actor2.ID as ID7_1,...) | medium | 71 |
- Problem Instance: PerformanceSettingsBean.getTime(...) | low | 300 |
- Problem Instance: HotDeployFilter.doFilter(...) | low | 3 |
- Problem Instance: FacesServlet.service(...) | low | 30 |
- Problem Instance: ApplicationDispatcher.invoke(...) | low | 1 |
- Problem Instance: NullInputValidator.doValidate(...) | low | 2 |

The cause consumes in average 23.7% (10795ms) of the response time (47549ms) of the corresponding business transaction.

In average 100% (10795ms) of the cause's execution time (10795ms) is spent on CPU. Hence, this performance problem is CPU bound!
Missing Business Context

How critical is this?

CPU utilization over time

Asking the right questions:

- How are the users affected?
- Which business transactions are affected?
- Are any time-critical batch jobs affected?
Defining Business Context
Conclusion

Node 1 (JVM)

App 1

App 2

Node n

Monitoring Tool
Dynamic Instrumentation
API Labeling
Location Identification

Instrumentation Quality Manager

Traces
Instrumentation Refinement Request
Result Query

Result

Challengers

Leaders

CA Technologies
AppDynamics
Dynatrace
New Relic

HPE
IBM

RNC

Nest Technologies

Complemane
Oracle
Microsoft

As of December 2016

Completeness of Vision

Ability to Digest