State of the Art of Visualization in APM Tools

André van Hoorn
Dušan Okanović

Tutorial @ Visualizing Systems and Software Performance (VSSP)
July 10, 2018. Dagstuhl, Germany
State of the Art of Visualization in APM Tools

Part 1/2: Introduction to APM

André van Hoorn
Dušan Okanović

Tutorial @ Visualizing Systems and Software Performance (VSSP)
July 10, 2018. Dagstuhl, Germany
Vorhang auf: OTTO bei eventim.de


Ticketalarm – kein Event mehr verpassen!

Registrieren Sie sich für den eventim.de-Ticketalarm, und Sie werden per E-Mail informiert, sobald an neue Termine Ihrer Lieblingstars und -events gibt.


Über diesen Künstler

Künstler-Biografie


Weiterlesen
MOMENT, BITTE!
SIE BEFINDEN SICH IM WARTERAUM VON EVENTIM, BITTE VERSUCHEN SIE ES SPÄTER WIEDER.
“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.”

Application Performance Management (APM) 
Continuous Monitoring of Application Performance. (in German)
Order for free: [http://www.sigs-datacom.de/wissen/fachposter.html](http://www.sigs-datacom.de/wissen/fachposter.html)
Einfluss von Performance auf Erfolg

Anwendungsperformance hat direkte Auswirkungen auf den Unternehmenserfolg:

1. 2 Sek ten, Wichtigkeit: Wo genau und Umsatz

60% der Online-Kunden verlassen ohne eine Seite, die Laufzeiten > 2 Sekunden

Nur 30% der Online-Kunden wechseln nach einem Performance-Problem zurück.

Konsequenz schlechter Performance

Umsetzung von Anwendungen

Zulässigkeit des Anwendungsprojekts

Userseitiges Erleben des Projektes

Application Performance Management

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

wichtig im gesamten Software-Lebenszyklus

...relevant für alle Systemebenen

...erfordert Zusammenarbeit aller Abteilungen

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

Application Performance Management

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

wichtig im gesamten Software-Lebenszyklus

...relevant für alle Systemebenen

...erfordert Zusammenarbeit aller Abteilungen

1. Collect
2. Process
3. Present
4. Reason & use
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>Passive</td>
</tr>
<tr>
<td>Operating System</td>
<td>File handling statistics, virtualization, thread statistics</td>
<td>Collection of runtime data from real system usage.</td>
</tr>
<tr>
<td>Hardware</td>
<td>CPU load, memory consumption, I/O statistics</td>
<td>E.g., injection of code, analysis of network traffic, resource utilization, or log files</td>
</tr>
</tbody>
</table>

Some technologies on lower levels provide standard interfaces for data collection, e.g., Nagios, JMX.
## Trace-based Metrics (Selection)

**What?**

- Metric
- Response Time
- CPU Time
- Method Name
- Return Type
- Logging Level
- SQL Statement
- Error Message
- ...

**Application**

Component interactions, method response times, trace data

---

Monitoring (Measurement-based Performance Evaluation)

Client Browser

0.3s

0.2s
Request

Response

0.4s

0.3s
Application Server

0.5s

0.1s
Query

Result

0.1s

3.0s
Database Server

4.9s
Data Recorder

Visualization

Data Analysis

0.3s

3.0s

4.9s
Reconstructing Information from Data

- 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 PR0D_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>
<tr>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>
- 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

© AppDynamics
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
State of the Art of Visualization in APM Tools

Part 2/2: Examples Visualizations in APM Tools

Tutorial @ Visualizing Systems and Software Performance (VSSP)
July 10, 2018. Dagstuhl, Germany
Commercial APM Tools
Magic Quadrant

Source: Gartner (March 2018)
Application Overview – Dynatrace

175.71ms 0.01%

55.74 per minute
163.84ms (5.2%)

40.09 per minute
104.33ms (3.31%)

10.12%

138 Active Visits 14.5% mobile

End User Experience
Analyze performance by geolocation, client types and operating system

Mobile Applications
Inspect mobile visits, and crashes

4 Applications

Focus on Passing Transactions Show Transaction Details (PurePaths)
Response Time Hotspots

Process: dtwsagent[Apache 2.2]
Show Web Server Health

Host: win-cv9q3ipjetb
Show Host Health

CPU MEM NET DISK

Failure Rate
0 (0%)

Execution Time per Transaction: 319.19ms
Transaction Response Time Contribution: 10.12%
Failed Transactions: 0 (0%)
Application Overview – Instana
End User Monitoring – Instana

```
Views | Load Time (mean) | Load Time (90th) | Load Time (95th)
1,011 | 1.18s            | 1.81s            | 1.81s
13% Server | Browser 87%       |
```

Views vs Page Load Time

Page Load Breakdown

- DNS: 0ms
- TCP: 0ms
- SSL: 0ms
- Request: 15ms
- Response: 33ms
- DOM: 516ms
- Children: 592ms

Uncaught Errors

```
Uncaught errors 76
```
Database Monitoring – AppDynamics

Dashboard for ECommerce-MySQL:
- Server Health: Normal
- Type: MySQL
- Total Time in Database: 00:03:42

Load and Time Spent in Database:
- Time Spent: 00:03:42
- Load: 97.33k

Top 10 SQL Wait States:
- 84.4% User sleep
- 6.5% Init
- 3.9% Update
- 1.7% Sending data
- 1.3% Updating
- 0.9% Writing to net
- 0.4% Removing tmp table
- 0.4% Opening tables
- 0.4% Freeing items

Average number of active connections:
- 0.3

Saturday, Jun 13, 2015 00:26:00
Number of connections: 0

Saturday, Jun 13, 2015 00:26:00
Time spent: 00:03:24
Load: 602
Throughput: 10.03 exe/sec
Server Monitoring – AppDynamics

Tutorial: State of the Art of Visualization in APM Tools
Server Monitoring – Instana

Tutorial: State of the Art of Visualization in APM Tools
Problem Identification – Dynatrace
Problem Identification - Instana

- **CPU Steal Time Exceeds 5% Too Often**
  - The virtual machine is undersized or the hypervisor host is overloaded. Consider allocating more than 1 CPU for this machine.
  - Started: 2017-11-16 08:05:42

- **System Load Too High**
  - Machine is more than 50% of its time at a load higher than 2
  - Started: 2017-11-16 06:23:12
Open Source APM tools
Open Source APM tools
Dashboards – Grafana

![Grafana Dashboard](image)

**Response Times**

- **SELECT mean("duration") FROM "stream" WHERE "type" = 'normal' AND "businessTransaction" =~ '/^$BusinessTransaction$/ AND $timeFilter GROUP BY time($interval) fill(null)'

**Graph**

- **A** SELECT mean("duration") FROM "stream" WHERE "type" = 'normal' AND "businessTransaction" =~ '/^$BusinessTransaction$/ AND $timeFilter GROUP BY time($interval) fill(null)

- **B** SELECT mean("duration") FROM "stream" WHERE "type" = 'problem' AND "businessTransaction" =~ '/^$BusinessTransaction$/ AND $timeFilter GROUP BY time($interval) fill(0)

- **C** SELECT mean("mean") FROM "statistics" WHERE "businessTransaction" =~ '/^$BusinessTransaction$/ AND $timeFilter GROUP BY time($interval) fill(null)

- **D** SELECT mean("lowerConfidenceLevel") FROM "statistics" WHERE "businessTransaction" =~ '/^$BusinessTransaction$/ AND $timeFilter GROUP BY time($interval) fill(previous)

- **E** SELECT mean("upperConfidenceLevel") FROM "statistics" WHERE "businessTransaction" =~ '/^$BusinessTransaction$/ AND $timeFilter GROUP BY time($interval) fill(previous)
Dashboards – Grafana

![Dashboard Image]

- **Request Count**: 304
- **Normal Requests Count**: 154
- **Anomal Request Count**: 150

**Response Times**
- Normal requests
- Slow requests
- Mean
- Confidence band lower
- Confidence band upper

**Throughput**
- Avg. Throughput: 0.08 reqs

**Slow Request Rate**
- Duration (normal): 154, 95.15 ms, 1.96 s, 3.55 s, 5.04 min
- Duration (problem): 150, 59.99 ms, 4.96 s, 28.08 s, 11.41 min

**Response Time Distribution**
- Average, Median, 95 percentile, 75 percentile, Max
Dashboards – Kibana
Dashboards – Kibana

Apache - Total Visitors

4,931,584

Apache - Unique Visitors

City | Unique Visitors
---|---
Beijing | 562
Redmond | 445
Ashburn | 400
Chicago | 373
Los Angeles | 245
Seattle | 233
San Jose | 232
Singapore | 208

Apache - Country and Status

Apache - Bytes and Count

Count

@timestamp per 3 hours

Apache - Country traffic by hour

Apache - Visitor Map (geocentroid)

Apache - Browser to Country (vega)
Execution Traces – inspectIT

OPEN.xtrace

https://github.com/spec-rgdevops/OPEN.xtrace
## Tracing – Zipkin

<table>
<thead>
<tr>
<th>Services</th>
<th>Elapsed Time</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>c*.Test Cluster.local</code></td>
<td>1.585s: QUERY</td>
</tr>
<tr>
<td><code>c*.Test Cluster.local</code></td>
<td>1.000ms: Parsing select * from vortex_powervomit_test.envelopes limit 100; [SharedPool-Worker-1]</td>
</tr>
<tr>
<td><code>c*.Test Cluster.local</code></td>
<td>1.000ms: Preparing statement [SharedPool-Worker-1]</td>
</tr>
<tr>
<td><code>c*.Test Cluster.local</code></td>
<td>2.000ms: Computing ranges to query [SharedPool-Worker-1]</td>
</tr>
<tr>
<td><code>c*.Test Cluster.local</code></td>
<td>15.000ms: Submitting range requests on 257 ranges with a concurrency of 1 (24892.65 rows per range executed)</td>
</tr>
<tr>
<td><code>c*.Test Cluster.local</code></td>
<td>69.000ms: Executing seq scan across 7 sstables for (min:-9223372036854775808), (min:-9223372036854775808)</td>
</tr>
<tr>
<td><code>c*.Test Cluster.local</code></td>
<td>531.000ms: Submitted 1 concurrent range requests covering 257 ranges [SharedPool-Worker-1]</td>
</tr>
<tr>
<td><code>c*.Test Cluster.local</code></td>
<td>2.000ms: Seeking to partition beginning in data file [SharedPool-Worker-2]</td>
</tr>
<tr>
<td><code>c*.Test Cluster.local</code></td>
<td>1.000ms: Seeking to partition beginning in data file [SharedPool-Worker-2]</td>
</tr>
<tr>
<td><code>c*.Test Cluster.local</code></td>
<td>Seeking to partition beginning in data file [SharedPool-Worker-2]</td>
</tr>
<tr>
<td><code>c*.Test Cluster.local</code></td>
<td>Seeking to partition beginning in data file [SharedPool-Worker-2]</td>
</tr>
<tr>
<td><code>c*.Test Cluster.local</code></td>
<td>Seeking to partition beginning in data file [SharedPool-Worker-2]</td>
</tr>
<tr>
<td><code>c*.Test Cluster.local</code></td>
<td>1.000ms: Seeking to partition beginning in data file [SharedPool-Worker-2]</td>
</tr>
<tr>
<td><code>c*.Test Cluster.local</code></td>
<td>Reading 12 live and 0 tombstone cells [SharedPool-Worker-2]</td>
</tr>
<tr>
<td><code>c*.Test Cluster.local</code></td>
<td>Seeking to partition beginning in data file [SharedPool-Worker-2]</td>
</tr>
<tr>
<td><code>c*.Test Cluster.local</code></td>
<td>Seeking to partition beginning in data file [SharedPool-Worker-2]</td>
</tr>
<tr>
<td><code>c*.Test Cluster.local</code></td>
<td>Seeking to partition beginning in data file [SharedPool-Worker-2]</td>
</tr>
<tr>
<td><code>c*.Test Cluster.local</code></td>
<td>Seeking to partition beginning in data file [SharedPool-Worker-2]</td>
</tr>
<tr>
<td><code>c*.Test Cluster.local</code></td>
<td>8.000ms: Seeking to partition beginning in data file [SharedPool-Worker-2]</td>
</tr>
<tr>
<td><code>c*.Test Cluster.local</code></td>
<td>1.000ms: Read 12 live and 0 tombstone cells [SharedPool-Worker-2]</td>
</tr>
</tbody>
</table>
Build Your Own Landscape

https://openapm.io
Dynamic Software Analysis and Application Performance Management

Monitoring Records → Analysis Configuration (via API and WebGUI) → Analysis → Pipes and Filters → Online and Offline Visualization

Measurement → Monitoring log/stream → Monitoring Probe

Software System with Monitoring Instrumentation

• http://kieker-monitoring.net

Tutorial: State of the Art of Visualization in APM Tools

Kieker is distributed as part of SPEC® RG’s repository of peer-reviewed tools for quantitative system evaluation and analysis.

http://research.spec.org/projects/tools.html
Application (and Visualization) Examples

Distributed Monitoring (Java EE/SOAP)

Legacy System Analysis (Visual Basic 6)

Legacy System Analysis (COBOL)

3D Visualization of Concurrency

Tutorial: State of the Art of Visualization in APM Tools
Conclusion

- Commercial tools have most fancy visualizations
  ... how useful are they really?

- Open-source tools are flexible and can be adapted as needed
  ... but require a lot of work to setup

- Thanks to a more widespread adoption of (open-source) APM, it is easier to get access to rich APM data (e.g., distributed traces) and to integrate visualization approaches (e.g., via APIs)
  ... if you know what and how to visualize