The Differing Ways to Monitor and Instrument

Southern Computer Measurement Group

Jonah Kowall, VP Market Development and Insights
Twitter: @jkowall
Jonah Kowall’s Background

- 23 years in IT
- Infrastructure and Operations enterprises and startups (17 yrs)
  - Security - CISSP, CISA, PCI
  - Started one of the first content filtering companies
- Head of global monitoring at Thomson Reuters
- Head of IT Operations at MFG.com – Bezos Expeditions
- Gartner Research VP 3.5 years
- Strategy AppDynamics 1.5 years
Agenda

• Introduction to Instrumentation
• Instrumentation done by people
• Challenges in instrumentation of the right processes
• Technical instrumentation of Browsers and Mobile
• Technical instrumentation of Server (php, Java, .NET…)
• How to implement the right logging
• How to correlate across technologies
Definitions

Instrumentation

“The design, construction, and provision of instruments for measurement, control, etc; the state of being equipped with or controlled by such instruments collectively.”

Telemetry

“Automated communications process by which measurements are made and other data collected at remote or inaccessible points and transmitted to receiving equipment for monitoring.”
Software Instrumentation Data Types

- **Metrics**
  - Key value pairs
  - Numeric values
  - Time series

- **Events**
  - Informational
  - Errors
  - Critical Events
Use Cases for Data Types

• Metrics
  ○ Average, Peak
  ○ Percentage
  ○ Correlation to Metrics and Events

• Events
  ○ Search
  ○ Parse
  ○ Correlate to Metrics
Software Instrumentation

• Logging
  ○ Supplied by vendor
  ○ Created by developers
  ○ Not easily controlled

• Push Collection
  ○ Attach and extract
  ○ Software agent or network tap

• Pull Collection
  ○ Polling APIs - HTTP, SNMP, WMI
Priorities for Instrumentation

- Business
- Application
- Services
- Infrastructure

Events and Metrics
Business Instrumentation
Uplevel the Conversation

• Understand the customer  
  - Internally and externally
• Requirements should be gathered across business and IT teams
• Responsibility for definition of monitoring should be shared
Business Metrics and KPIs

- **Customer Metrics**
  - Conversion between products
  - Loyalty and retention (churn)
  - Usage metrics (feature and product)

- **Sales / Marketing Metrics**
  - Revenue
  - Cost of customer acquisition
  - User flows through applications
Technical Metrics and KPIs

- End to end performance
  - User through transaction hops
  - Error isolation
- End user experience
  - Client side errors
  - Latency per element (page or app) + 3rd party
  - Client side DNS
- Application component performance
  - Metrics from app server
  - Metrics from code
  - Queries
  - Errors
- Intra application component performance
Use Cases for Business and Technical Data

- Usage
- Problem identification - MTTI
- Problem resolution - MTTR
- User satisfaction
- Usability
- Performance
- Change analysis
  - A/B testing
  - data center moves
  - technology changes
Instrumenting Browsers

High-Resolution Time
w3c.github.io/hr-time
- Defines DOMHighResTimestamp
- Defines Performance interface
- Defines time origin for Window/Worker

Performance Timeline
w3c.github.io/performance-timeline
- Extends Performance interface
- Defines PerformanceEntry interface
- Defines PerformanceObserver interface

Resource Timing
w3c.github.io/resource-timing
- Extends Performance interface
- Defines PerformanceResourceTiming interface

Navigation Timing
w3c.github.io/navigation-timing
- Extends PerformanceResourceTiming interface
- Defines PerformanceNavigationTiming interface
- Extends PerformanceResourceTiming interface

User Timing
w3c.github.io/user-timing
- Extends Performance interface
- Defines PerformanceMark & PerformanceMeasure interfaces

Server Timing
w3c.github.io/server-timing
- Extends Performance interface
- Defines PerformanceServerTiming interface

Frame Timing
w3c.github.io/frame-timing
- Extends Performance interface
- Defines PerformanceFrameTiming
Instrumenting Mobile
Simulating Users

Synthetic transactions are good for

- SLAs
- Availability
- Baseline performance
- DNS
- SSL

But if you try to use it as a barometer you will fail
Backend Instrumentation of Java and .NET
Java Instrumentation

1. `java -javaagent:path/to/javaagent.jar ApplicationClass`

2. `AgentEntryPoint.premain(String agentArgs, Instrumentation inst)`

3. `ApplicationClass.main(String args[])`

JSR-163 (Java™ Platform Profiling Architecture) added in Java 1.5

Overloads the default behavior of Java to allow hooks into code for many use cases

Since JDK 1.6, for the Oracle HotSpot JVM, a javaagent may be dynamically attached to a running JVM by specifying the process-id (pid).
- Profiling API loaded into the same process as the application process that is being profiled.
- Callback interface (ICorProfilerCallback in the .NET Framework version 1.0 and 1.1, ICorProfilerCallback2 in version 2.0 and later)
- CLR calls the methods in that interface to notify the .NET agent of events in the profiled process
- Profiler can also call into the CLR by using the methods in the ICorProfilerInfo and ICorProfilerInfo2 interfaces to obtain information about the state of the profiled application
- Callbacks are used to inject MSIL (Microsoft Intermediate Language) bytecode into existing application code for instrumentation.
Backend Instrumentation of Interpreted Languages
Monkey Patching

Wikipedia relevant definition:
In Ruby,[3] Python,[4] and many other dynamic programming languages... dynamic modifications of a class or module at runtime, motivated by the intent to patch existing third-party code as a workaround to a bug or feature which does not act as desired

- Replace methods / attributes / functions at runtime
- Apply a patch at runtime to the objects in memory, instead of the source code on disk;

Disclaimer: Can be very dangerous, hard to maintain
Zend callback methods `zend_execute(…)`*, `zend_execute_internal(…)`* and `zend_compile_file(…)`* so that it can wrap the original implementations with instrumentation code.

Handles state changes and new web server initialization (which are PHP instances)
Node.js Instrumentation

- Wrap methods using before, after and around aspect interceptors.
- Callback along with after, before and around aspect interceptor.
- Notifications when asynchronous calls are complete.
Python Instrumentation

1. PYTHONPATH=appdynamics/autoinject: python ...
2. Transient bootstrap built-in module
3. Instantiate agent
   Register interceptors
4. Run Python Application code

Python interpreter

Python agent
Logging Best Practices

- Easily parsed (JSON)
- Time (long), Source
- Log errors and exceptions
- Logs are not transaction records, they are not good metric stores
- Write your own identifiers for each statement logged (or instrument and inject)
- Think about security implications (plain text, on disk, syslog insecure)
- Keep small (thanks Java, .NET…)
- Don’t overdo it (performance implications)
Transaction Correlation
Correlation in end to end APM
Correlation in asynchronous calls (headache)
Correlation in Open Source

Zipkin is the most advanced, many new forks and instrumentations (Java, JavaScript, Python, Ruby, C#, Go)
No async support :(  

[Diagram showing a network of services with Zipkin at the center]
Future of Correlation in Open Source

• OpenTracing an open standard API for instrumentation
  o Doesn’t manage overhead, can hang yourself
• PivotTracing runs distributed traces on demand
  o After issues detected

• Spigo visualization and simulation lots of evolution
Correlation in Logs

- Log every transaction segment
- Persist a GUID or transaction ID
- This is very difficult in large teams
- Inefficient to analyze and pull metrics from logs
- Doesn’t work unless you own the code
Transaction Correlation and Logs!

- Many integrations across APM and Log vendors
- Can add correlation in code and use any log tool
  - ex: [%X{AD.requestGUID}]  
- We auto inject and correlate (one platform)
Intro to AppDynamics
Principles of Digital Transformation

1. The App is the Business
2. Velocity is Critical
3. Agile Architectures
4. Public Cloud
Application Complexity is Exploding

Move Fast and Don’t Break Things

**Trends**

1. Multi-device & IoT
2. Microservices & Containers
3. Micro-databases & NoSQL
4. Hybrid Cloud
5. Continuous Deployment
How AppDynamics works
How AppDynamics works
How AppDynamics works
How AppDynamics works
How AppDynamics works

TRANSACTIONS
- Login
- Search Flight
- Status
- Purchase
How AppDynamics works

TRANSACTIONS
- ✔ Login
- Search Flight
- Status
- ✔ Purchase
How AppDynamics works

Operations

TRANSACTIONS
- Login
- Search Flight
- Status
- Purchase
How AppDynamics works

Developer

- Transactions
  - Login
  - Search Flight
  - Status
  - Purchase
How AppDynamics works

Business

Transactions

- Login
- Search Flight
- Status
- Purchase
Application and Business Monitoring Products
App iQ Platform Summary

- Discover a business processing unit – BT
- Self learn its performance metrics – report constantly, all traffic
- Self learn its distributed activity – report constantly, all traffic
- Self learn its performance thresholds – evaluate constantly
- Self learn code level activity – report only when needed
- Automatic detection of SLA failures
- Self collect diagnostics – for anomalies and problem patterns
App iQ is Extensible

AppDynamics Apps

End-User Monitoring
Application Performance Monitoring
Infrastructure Visibility

App iQ Platform

Map iQ
Baseline iQ
Diagnostic iQ
Business iQ

Signal iQ

Over 100 3rd Party Extensions

AppD Cloud
Private Cloud
Key AppDynamics Differentiators

Map iQ
- End-to-end business transaction tracing
- Full business context

Baseline iQ
- Machine learning
- Alerting on deviations from baseline

Diagnostic iQ
- Deep diagnostic data to code-level
- Low overhead, in high-scale, production environments

Business iQ
- Real-time business monitoring and alerting
- Correlate app performance and business impact

Deployment Choice
Gartner Rates AppDynamics #1 Across All Critical Capabilities

AppDynamics #1 – Application Support

AppDynamics
Dynatrace
Dell
New Relic

AppDynamics #1 – Application Development

AppDynamics
Dynatrace
New Relic
Dell

AppDynamics #1 – IT Operations

AppDynamics
Dynatrace
HP
New Relic

AppDynamics #1 – Line of Business

AppDynamics
New Relic
Dynatrace
CA Technologies

AppDynamics #1 – DevOps Release

AppDynamics
Dynatrace
New Relic
HP

Source: Gartner Critical Capabilities for APM, 2016
The World’s Leading Enterprises Use AppDynamics

<table>
<thead>
<tr>
<th>Financial Services</th>
<th>Industrial</th>
<th>Media and Entertainment</th>
<th>Tech/Telecom</th>
<th>Healthcare</th>
<th>Consumer</th>
<th>Government</th>
<th>Partners</th>
</tr>
</thead>
<tbody>
<tr>
<td>UBS</td>
<td>Audi</td>
<td>comcast</td>
<td>salesforce</td>
<td>BlueCross Bluelight</td>
<td>POLO RALPH LAUREN</td>
<td>UNITED STATES POSTAL SERVICE</td>
<td>IBM</td>
</tr>
<tr>
<td>Citizens Financial Group, Inc</td>
<td>FedEx</td>
<td>Bloomberg</td>
<td>CISCO</td>
<td>MAYO CLINIC</td>
<td>TESCO</td>
<td></td>
<td>accenture</td>
</tr>
<tr>
<td>Paypal</td>
<td>United Airlines</td>
<td>Ernst &amp; Young</td>
<td>vodafone</td>
<td>McKesson</td>
<td>Expedia</td>
<td></td>
<td>Capgemini</td>
</tr>
<tr>
<td>Nasdaq</td>
<td>Choice Hotels</td>
<td>skyimedia</td>
<td>eHarmony</td>
<td></td>
<td></td>
<td></td>
<td>Cognizant</td>
</tr>
<tr>
<td>Quicken Loans</td>
<td>Siemens</td>
<td>EA</td>
<td>okta</td>
<td>Pfizer</td>
<td>Shutterfly</td>
<td></td>
<td>rockspace</td>
</tr>
</tbody>
</table>
“In their own words”

“Without AppDynamics, the transformation from being reactive to proactive would not have been possible!”
- The Container Store

“Business transactions gave the ING team an entirely new perspective on how to view requests across their distributed system.”
- ING

“AppDynamics provides historical data, so we can get automated baseline for normal performance and then trigger diagnostics if there’s a major deviation.”
- Priceline.com

“...it used to take several people looking at several different monitoring solutions, comparing numbers to find any problem with our system.”
- eHarmony

“AppDynamics is a game changer. It has transformed our applications into a living, breathing entity.”
- Pearson Education

“AppDynamics was rolled out globally across 23 data centers – deploying a total of 15,000 agents in just one week – all to a single AppDynamics report server.”
- Cisco

“It was simple to deploy. It took minutes, literally. In our proof of concept, we saw results within an hour.”
- NASDAQ

Source: AppDynamics Website
Thank You