FedCSIS
2015

What has gone wrong in the cloud?

Monitoring the cloud and things that run in it

Wojtek Kozaczynski

Principal Software Development Engineer
Microsoft – Developer Division – Application Insights

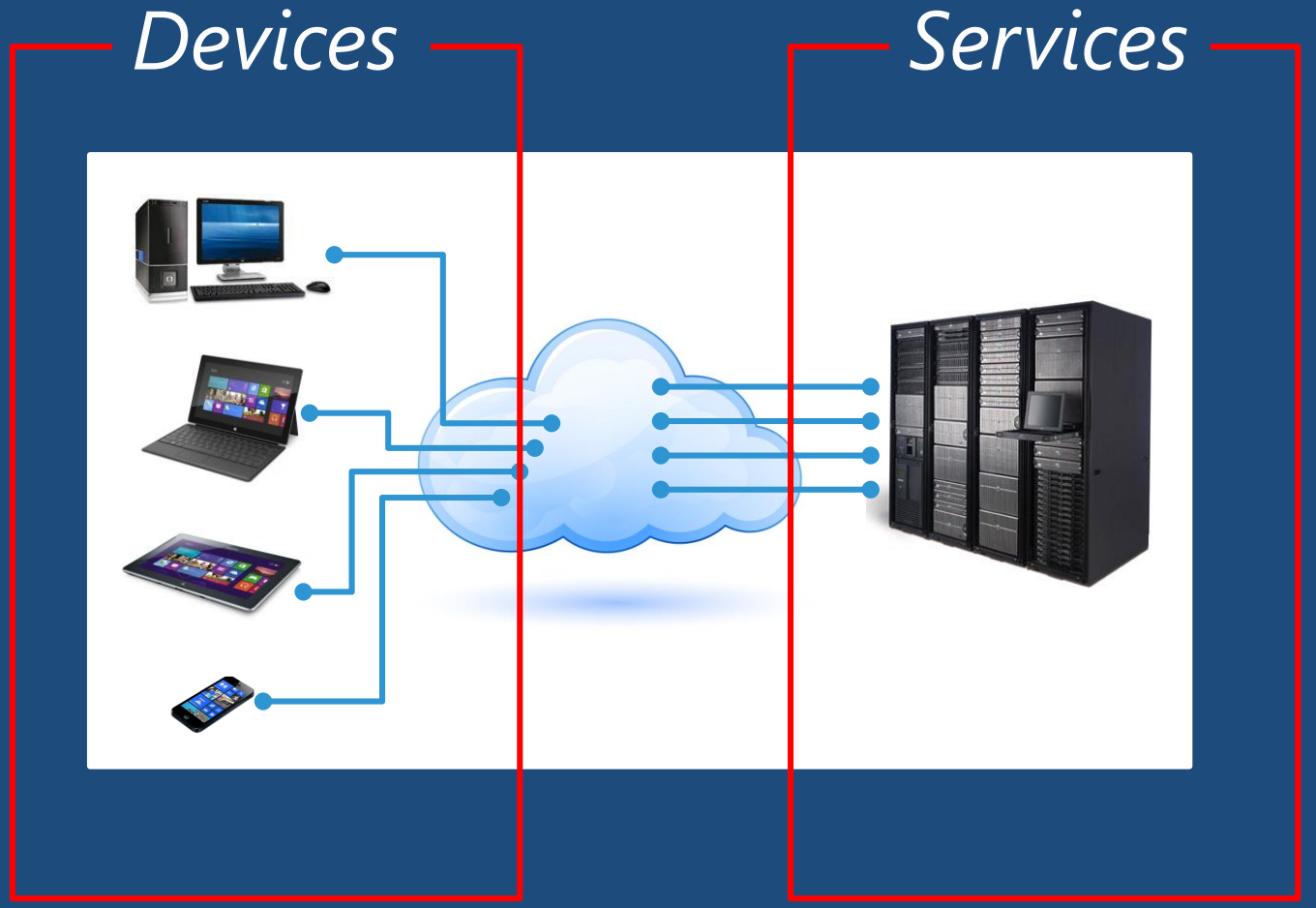
Who is Wojtek?

- formally educated in Poland
- almost entire professional career in the US
  - UICC, Andersen Consulting, System S/W Associates, Rational, IBM, Microsoft
- Principal S/W Development Engineer
- IC (Individual Contributor)
- for the last 5+ years working on OS and application monitoring and diagnostics
  - WMI, WinRM, PowerShell, ETW, Azure, …
- currently an architect on the ApplicatinInsights team, which is developing a public Azure/VSO telemetry service

The shortest version of the talk

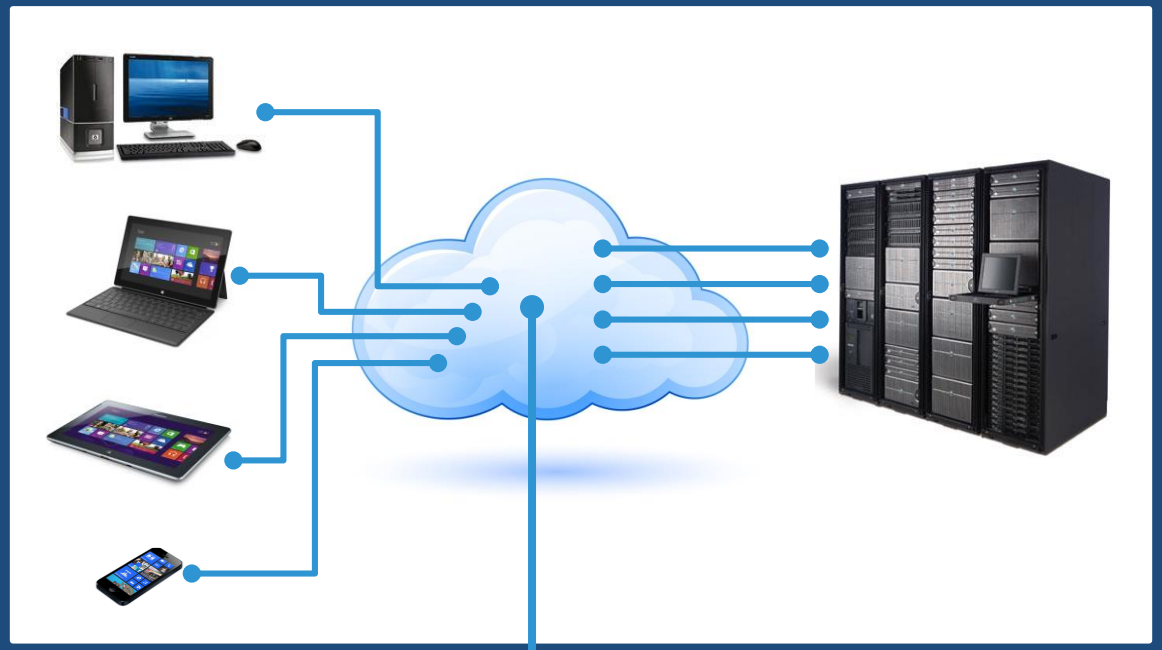
1. Real cloud applications get complex quickly
2. The only way to understand what is happening in, or to, a cloud application is to be very deliberate what telemetry data it sends
3. In order to analyze telemetry from a complex cloud application you have to build another cloud application, which may be equally or more complex that the one it monitors

Cloud application



	Devices	Cloud Services
execution environment	• you have no access to the devices • application is self-contained	• you have access to, manage, and pay for the runtime • your application takes dependency on cloud services
application and users	• number of instances each owned by a different user • you know very little about the users	• one instance owned by you
telemetry focus	• use • usability • failure detection and analysis	• performance • throughput • availability • dependencies on cloud services • resource utilization • failure detection and analysis
telemetry data	• interrupted flow • small per instance volume • possibly large cumulative volume	• continues flow • large volumes • about application, execution environment, dependencies

Telemetry service



Telemetry service == cloud application that

- collects
- processes, and
- presents

data about other cloud applications

Telemetry Service

About the telemetry data

- What should it look like?
  - what should it represent?
  - how should it represent it?
- How can we make all devices and servers send consistent data so we can
  - collect
  - store, and
  - process

  it consistently?

Common telemetry representation

- Commonly telemetry items contain two parts
  - data about context in which the item was generated
  - data item itself
- In practice there are only four basic telemetry types
  - measurements
    - single values or aggregations
  - events
    - named collections of properties representing events
  - trace messages
  - crash data
    - exception trees, call stacks, memory dumps

Open source SDKs == a way to consistency

- .Net
- Java
- JavaSctipt
- node.js
- iOS
- Android
- C++
- PHP
- Ruby
- Python
- OSX
- Server ASP .Net
- Unity
- Xamarin
- Angular JS

https://github.com/Microsoft/ApplicationInsights-Home

README.md

ApplicationInsights-Home

Welcome to the central repository for Application Insights projects. We intend to include any content that applies to all our repositories here, while specific content to each project will be found with that project.

More information about Application Insights can be found here.

Our Repositories

Android

Asp.Net v5

C++ Universal apps

DotNet Core

iOS

JavaScript

Java

Node.js

OSX

PHP

Python

Ruby

WordPress on GitHub and Release Version on Wordpress.org

Xamarin

Python

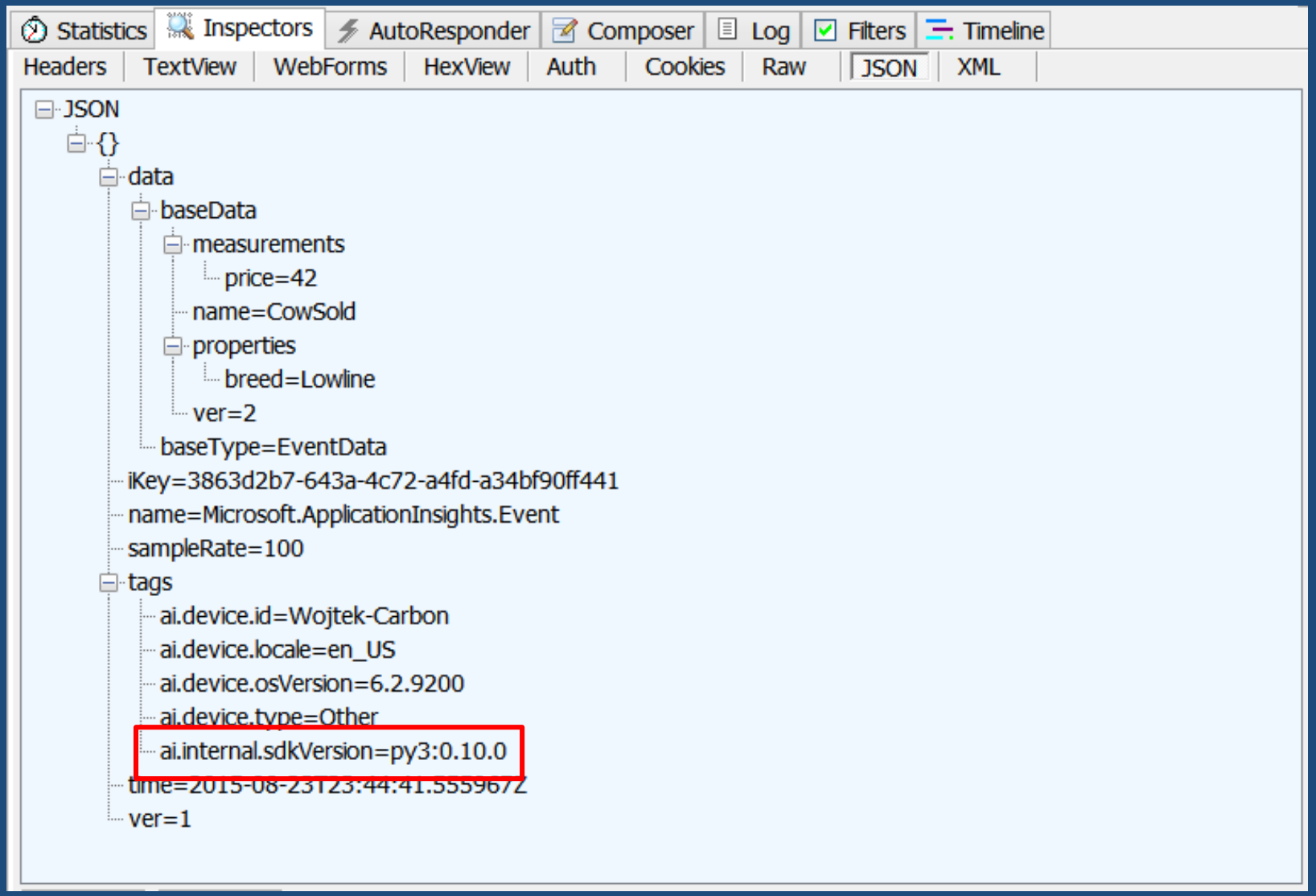
```
from applicationinsights import TelemetryClient
tc = TelemetryClient('3863d2b7-643a-4c72-a4fd-a34bf90ff441')
tc.track_event('CowSold', { 'breed': 'Lowline' }, { 'price': 42 })
tc.flush()
```

.Net

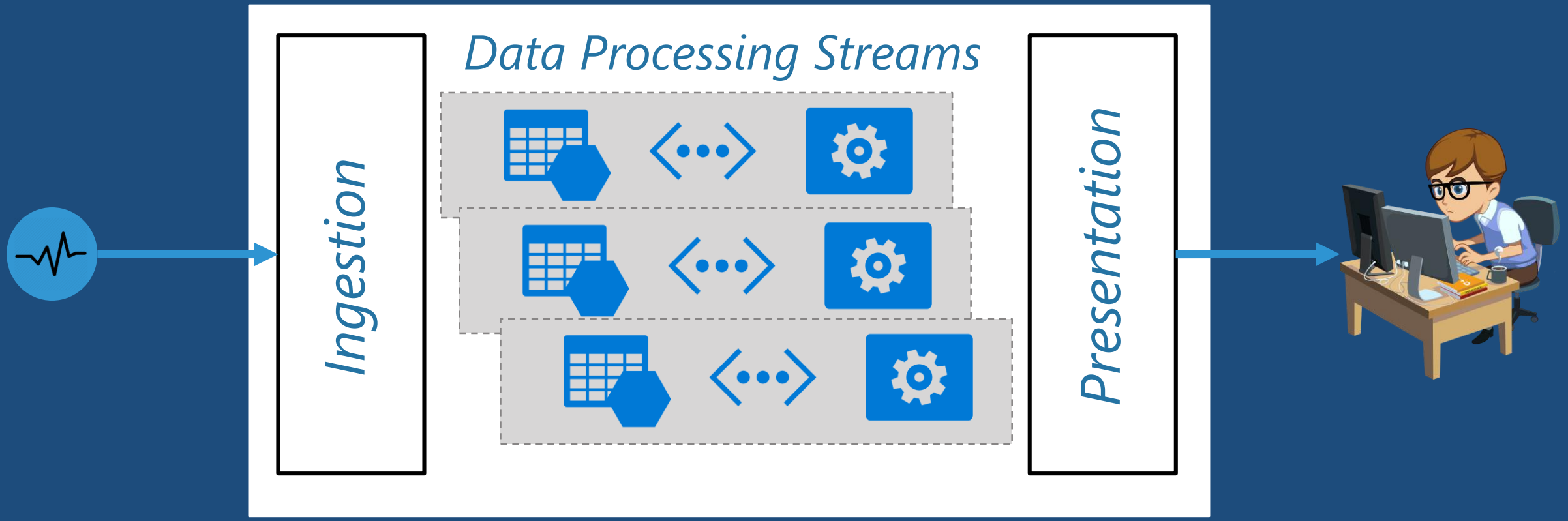
```
TelemetryClient tc = new TelemetryClient();
tc.Context.InstrumentationKey = "3863d2b7-643a-4c72-a4fd-a34bf90ff441";

EventTelemetry et = new EventTelemetry();
et.Properties.Add("breed", " 'Braford");
et.Metrics.Add("price", 77);
tc.TrackEvent(et);
tc.Flush();
```

The same data on-the-wire



Anatomy of Telemetry Service



About ingestion volumes

- How much data an application sends?

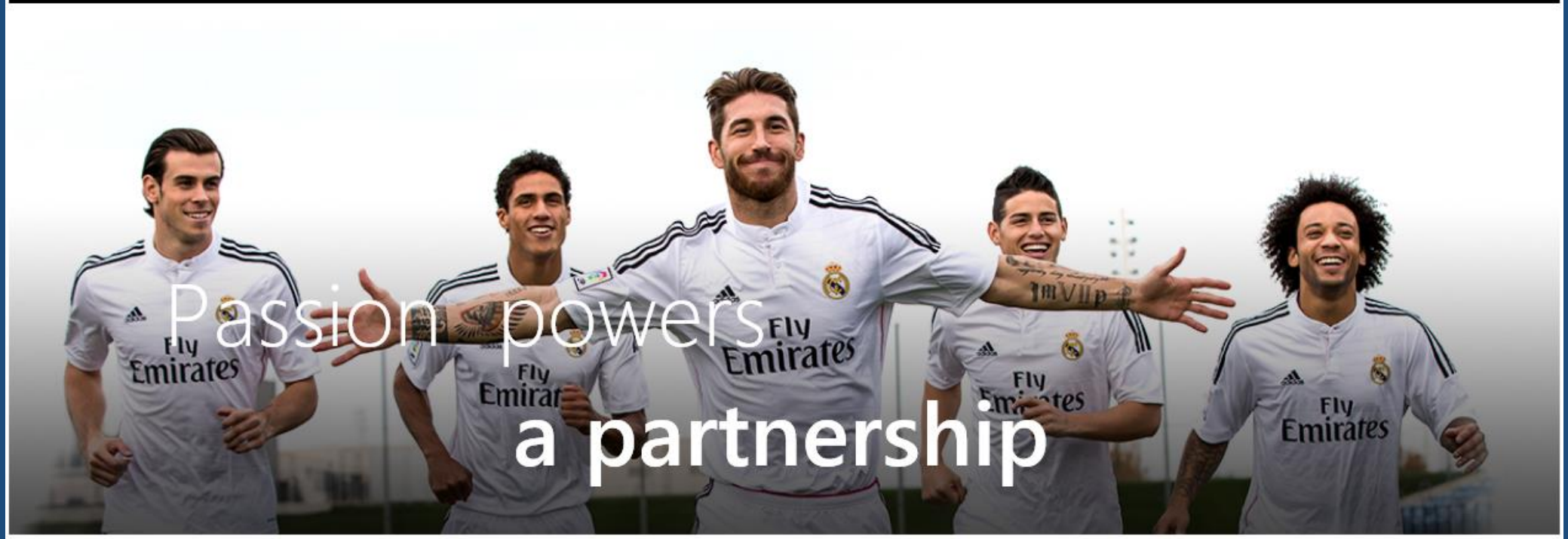
App Size	Documents/sec	Documents/month
Small	< 200	< 518,400,000
Medium	200 - 500	~ 0.5 – 1.3 Bil
Large	500 – 1,000	~ 1.3 – 2.3 Bil
Very large	1,000+	2.3+ Bil
Super large	n*1K	n * 2.5 Bil

Ingesting at super large scale

- A very popular internet phone application ☺
  - n*100 Mil users/devices and >10 K servers
  - 1 Mil [docs/sec] at peaks
    - 5-10 [TB/hour] => 150 [TB/day]
  - 5 geo-distributed ingestion regions
    - n*1 K machines (Azure services) ingesting data
  - > 500 Azure storage accounts
    - ~50 – 100 K storage TPS

Microsoft

Official Technology Partner



Microsoft and Real Madrid are joining together to transform the way the best fútbol club in the world connects with its 450 million international fans. Through Microsoft Cloud technology and devices, we will change the way Real Madrid fútbol is played, coached, watched and ultimately experienced.

Hala Madrid!

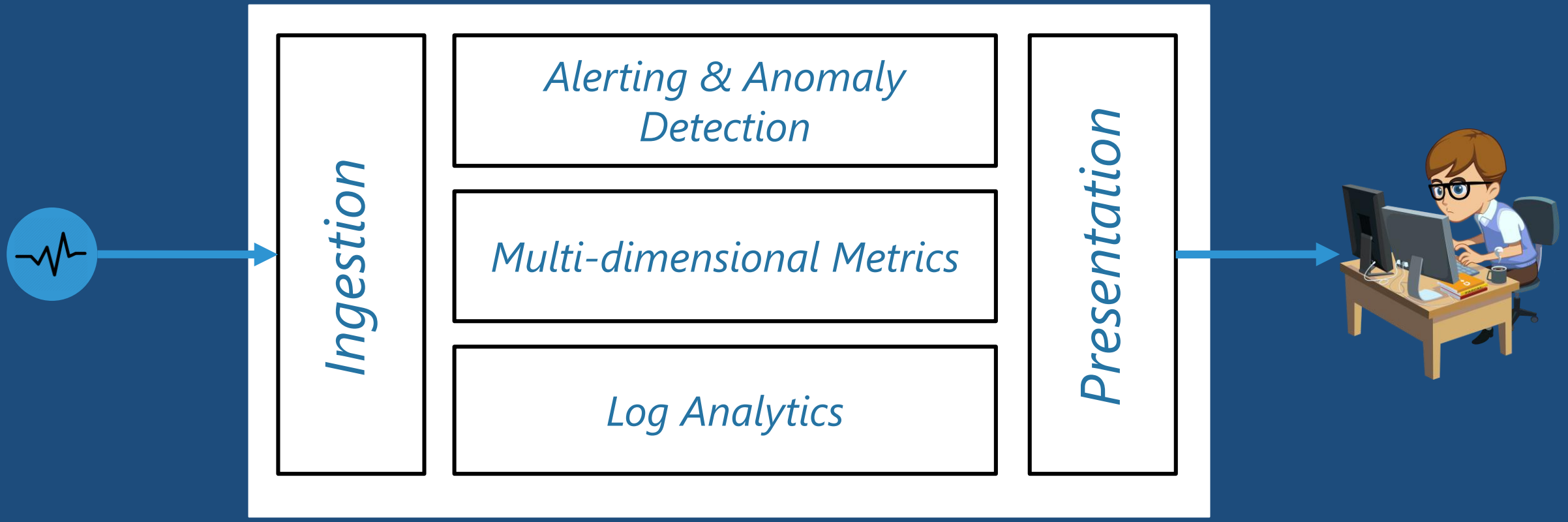
Ingesting at large scale

- Real Madrid CF
  - 8 geo-distributed ingestion regions
  - Ingested to 6 data clusters, each with ~100 nodes/machines
  - 1.5 – 2 K [docs/sec] sustained ingestion rate from devices
  - > 2 K [docs/sec] sustained ingestion rate from servers

Common use of telemetry data

- APM (Application Performance Management)
  - What is the volume of data transactions?
  - What is the average response time? What is the 90th-percentile response time?
  - What is the load on my servers (CPU + Memory)?
  - What is the size of request queue? . . .
- Issue detection
  - If you detect average CPU usage across cluster higher than 80%, let me know ASAP.
  - If you detect abnormal memory usage in any cluster node, let me know ASAP.
- Diagnostics
  - What failed and when?
  - What happened before the failure?
  - What was the root cause? . . .
- Usage
  - Who uses my application, where and when?
  - What are the most used feature?
  - What are the least used features? . . .

Common data streams

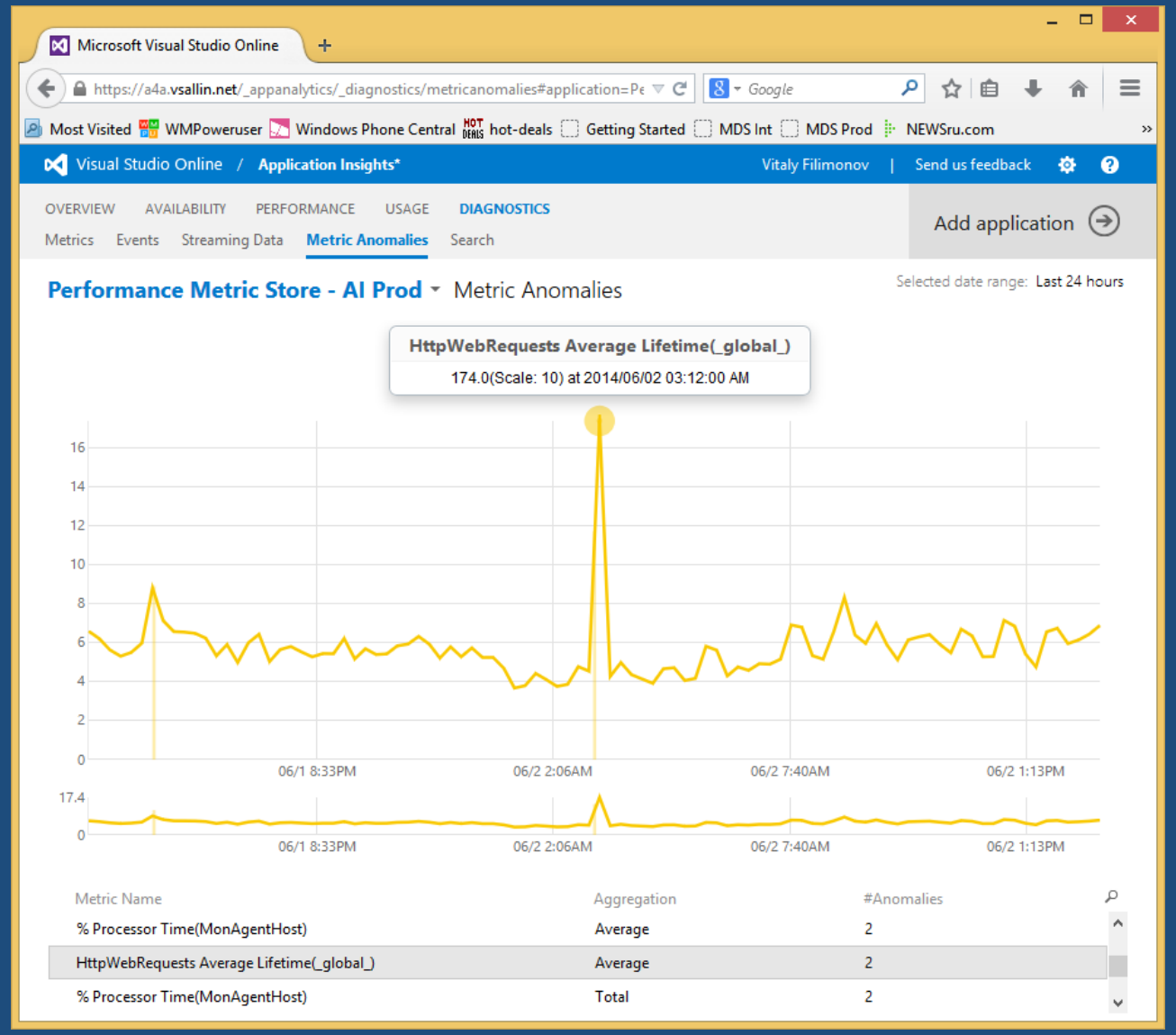


Alerting & anomaly detection

- Low latency (aka near-real-time) data stream with no data retentions
- Alerts/notifications based on two kind of rules
  - Aggregated metric below or above a threshold
  - Anomaly detection

…

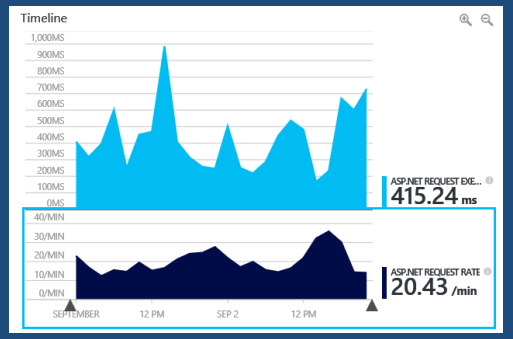
- Metric value
  - selected metrics
  - for a selected combinations of dimension values
  - in a tumbling time window
  - aggregation value (`count`, `min`, `max`, `average`, `stddiv`) is above/below a threshold
- Anomaly detection
  - watches a stream of aggregated metrics
  - learns what “normal” looks like and self adjusts
  - detects abnormal sequences
    - sharp rises or drops
    - unexpected values
    - spikes


Microsoft Visual Studio Online
https://a4a.vsallin.net/_appanalytics/_diagnostics/metricanomalies#application=Pe
Google
Most Visited
WMPoweruser
Windows Phone Central
hot-deals
Getting Started
MDS Int
MDS Prod
NEWSru.com
Visual Studio Online / Application Insights*
Vitaly Filimonov | Send us feedback
OVERVIEW AVAILABILITY PERFORMANCE USAGE DIAGNOSTICS
Metrics Events Streaming Data Metric Anomalies Search
Add application
Performance Metric Store - AI Prod
Metric Anomalies
Selected date range: Last 24 hours
HttpWebRequests Average Lifetime(_global_)
174.0(Scale: 10) at 2014/06/02 03:12:00 AM
16
14
12
10
8
6
4
2
0
06/1 8:33PM
06/2 2:06AM
06/2 7:40AM
06/2 1:13PM
17.4
0
06/1 8:33PM
06/2 2:06AM
06/2 7:40AM
06/2 1:13PM
Metric Name
Aggregation
#Anomalies
% Processor Time(MonAgentHost)
Average
2
HttpWebRequests Average Lifetime(_global_)
Average
2
% Processor Time(MonAgentHost)
Total
2

MDM (multi-dimensional metrics)

- Medium-to-high-latency data stream with fixed data retention (days or weeks)
- Collects all metrics
  - commonly pre-aggregates them in small time windows
  - often receives pre-aggregated metrics from the application
- Constructs time series of metric values (`count`, `min`, `max`, `average`, `stddiv`) for
  - a given metric
  - a set of dimensions and their values

…



- Example request may look like this
  - "Show me distribution of
    - Max values of *CPU* metric
    - for the last two hours divided into 10-minut intervals
    - for `(cpu.clusterName == "C0T0")` and `(cpu.serverName == "A45")`"
- Comes down to indexing potentially **very** large set of metric documents

  `(metricName, timeInterval, dimName, dimValue) → {metric documents}`
- Dimension capping problem

  $$\text{indexSize} <= \left(\sum_{d_1}^{d_n}(\textbf{valueSet}^k).\textbf{Size}()\right) * \#\text{metrics} * \#\text{intervals}$$
  - capping is limiting number of dimensions and their values

Log analytics

- Data stream that
  - stores all telemetry data for (practically) unlimited time
  - supports user queries against that data
- What makes is possible?
  - The data is never updated
    - it can be indexed and compressed during ingestion
  - Most of the data is never looked at and when it is, it is looked at in a time interval
    - the data is organized in time-bound blocks
    - only relevant blocks are touched to compute a query
  - Data is not updated => query results can be cached
  - Older data is aggregated into progressively larger time windows (days, weeks, months)

	Alerting & Anomaly Detection	Multi-Dimensional Metrics	Log Analytics
APM		What is the average response time? What is the load on my servers? What is the size of the request queue?	What is the volume of date transactions?
Issue Detection	If you detect abnormal memory usage in any cluster node, let me know ASAP. If you detect average CPU usage across cluster higher than 80%, let me know APSP.	Show me average CPU usage across cluster for the last 2 hours.	
Diagnostics	What failed and when?	What filed and when? What happened before the failure?	What was the root cause?
Usage			Who uses my application, where and when? What are the most used features? What are the least used features?

Making $$ analyzing other peoples' telemetry

loggly

New Relic®

sumologic
Big Data for Real-time IT™

FLURRY

dynatrace

AppDynamic

Twitter Analytics

splunk>

Before you do it, think about …

- Multi-tenancy
- SLAs
  - Reliability
  - Availability
  - Security
  - …
- Compliance
- Scalability
- COGS
- Billing
- …

Back to my initial points

1. Real cloud applications can get very complex
2. You have to make your application send really good telemetry to have any chance of knowing what is happening in and to it
3. Building a good telemetry service may be a very sizable investment

wojtek@microsoft.com