Activate Microsoft Azure with Azure Application Insights

Conditions and Terms of Use

Microsoft Confidential

This training package is proprietary and confidential, and is intended only for uses described in the training materials. Content and software is provided to you under a Non-Disclosure Agreement and cannot be distributed. Copying or disclosing all or any portion of the content and/or software included in such packages is strictly prohibited.

The contents of this package are for informational and training purposes only and are provided "as is" without warranty of any kind, whether express or implied, including but not limited to the implied warranties of merchantability, fitness for a particular purpose, and non-infringement.

Training package content, including URLs and other Internet website references, is subject to change without notice. Because Microsoft must respond to changing market conditions, the content should not be interpreted to be a commitment on the part of Microsoft, and Microsoft cannot guarantee the accuracy of any information presented after the date of publication. Unless otherwise noted, the companies, organizations, products, domain names, e-mail addresses, logos, people, places, and events depicted herein are fictitious, and no association with any real company, organization, product, domain name, e-mail address, logo, person, place, or event is intended or should be inferred.

Copyright and Trademarks

Microsoft may have patents, patent applications, trademarks, copyrights, or other intellectual property rights covering subject matter in this document. Except as expressly provided in written license agreement from Microsoft, the furnishing of this document does not give you any license to these patents, trademarks, copyrights, or other intellectual property.

Complying with all applicable copyright laws is the responsibility of the user. Without limiting the rights under copyright, no part of this document may be reproduced, stored in or introduced into a retrieval system, or transmitted in any form or by any means (electronic, mechanical, photocopying, recording, or otherwise), or for any purpose, without the express written permission of Microsoft Corporation.

For more information, see **Use of Microsoft Copyrighted Content** at
http://www.microsoft.com/en-us/legal/intellectualproperty/Permissions/default.aspx

Microsoft, Internet Explorer, Outlook, OneDrive, Windows Vista, Zune, Xbox 360, DirectX, Windows Server and Windows are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries. Other Microsoft products mentioned herein may be either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries. All other trademarks are property of their respective owners.

Module Overview: Application Insights

Agenda

- Understanding Application Insights
- Diving Deeper into Application Insights
- Custom Telemetry/Logging with Application Insights API
- Monitor Logs

Module Overview

Section : Understanding

Lesson: Understanding Application Insights

Facing these questions?

What is Application Insights?

1

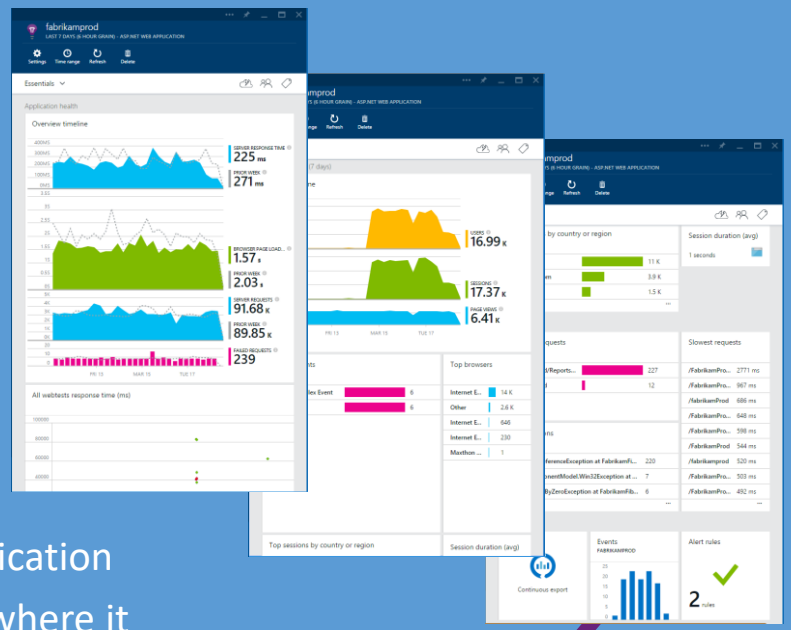
Telemetry is collected at each tier: mobile applications, server applications and browser

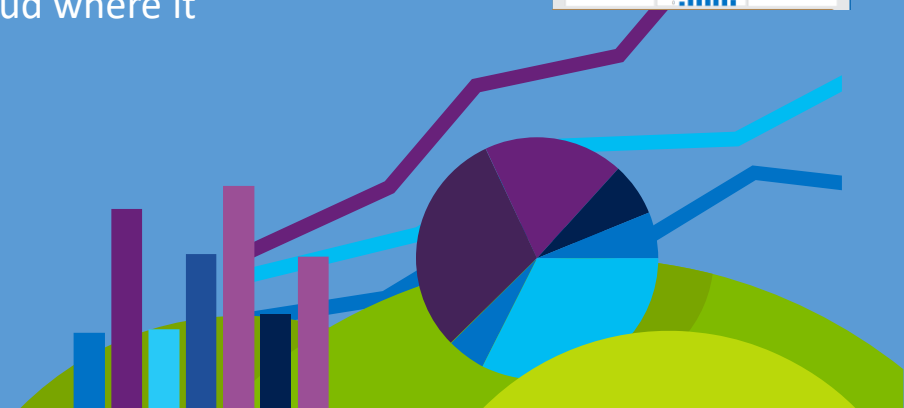
2

Telemetry arrives in the Application Insights service in the cloud where it is processed & stored

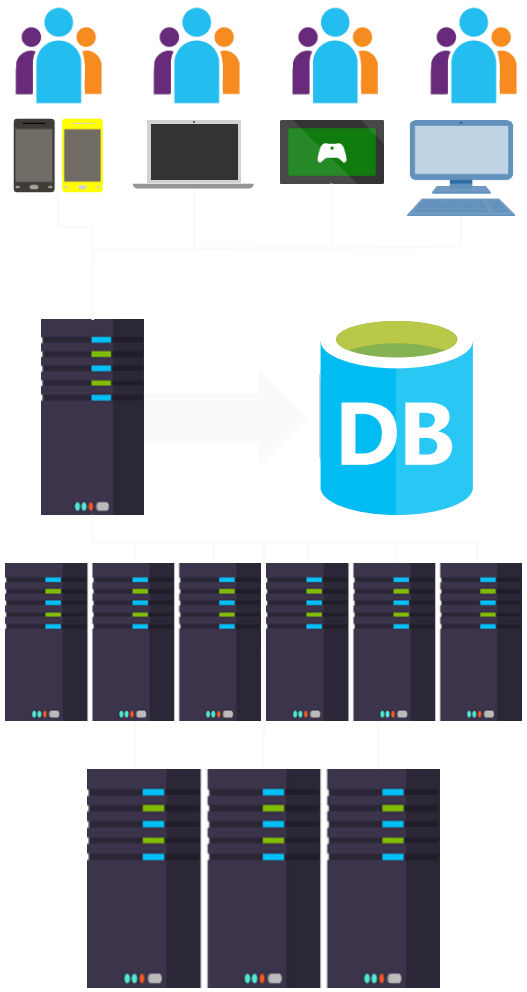
3

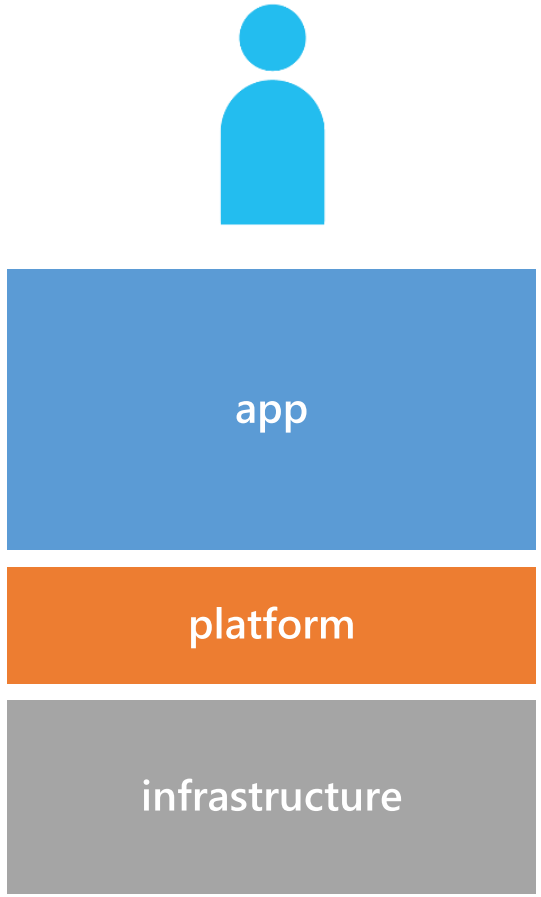
Get a 360° view of the application including availability, performance and usage patterns





Sources of Telemetry





1

Outside-in monitoring

URL pings and web tests from 16 global points of presence

2

Observed user behavior

How is the application being used?

3

Developer traces and events

Whatever the developer would like to send to Application Insights

4

Observed application behavior

No coding required – service dependencies, queries, response time, exceptions, logs, etc.

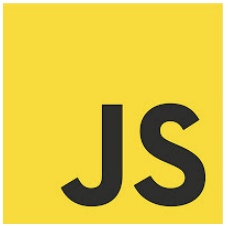
5

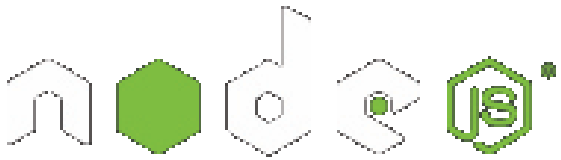
Infrastructure performance

System performance counters

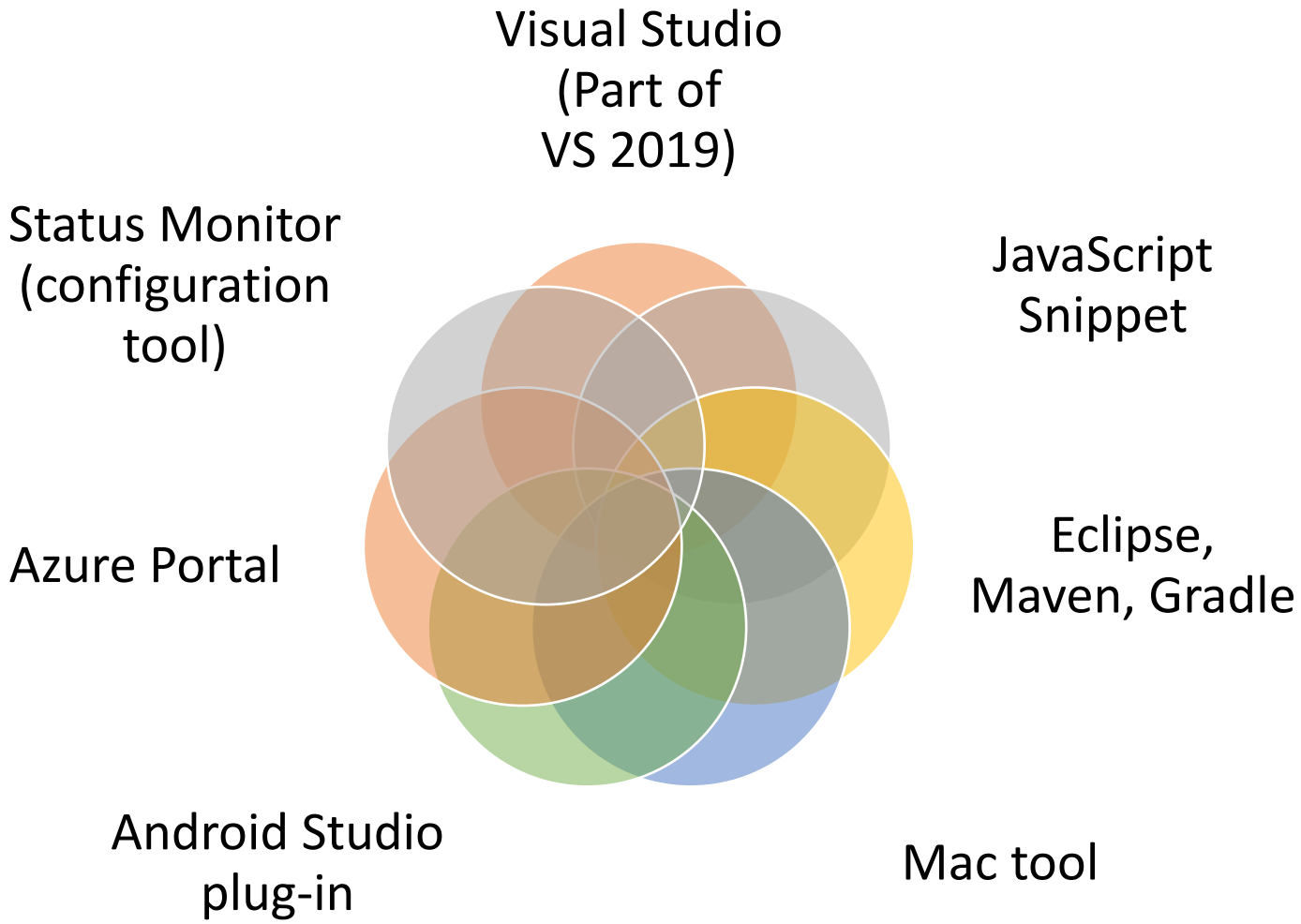
Supported Platforms & Languages

- Any Platform - Any App
  - On-Premises, Azure, AWS, Google Cloud, App Stores…
- Open Source SDKs
- Logging Frameworks
  - Log4Net, nLog, System.Diagnostics, Log4J, Logback





Getting Started

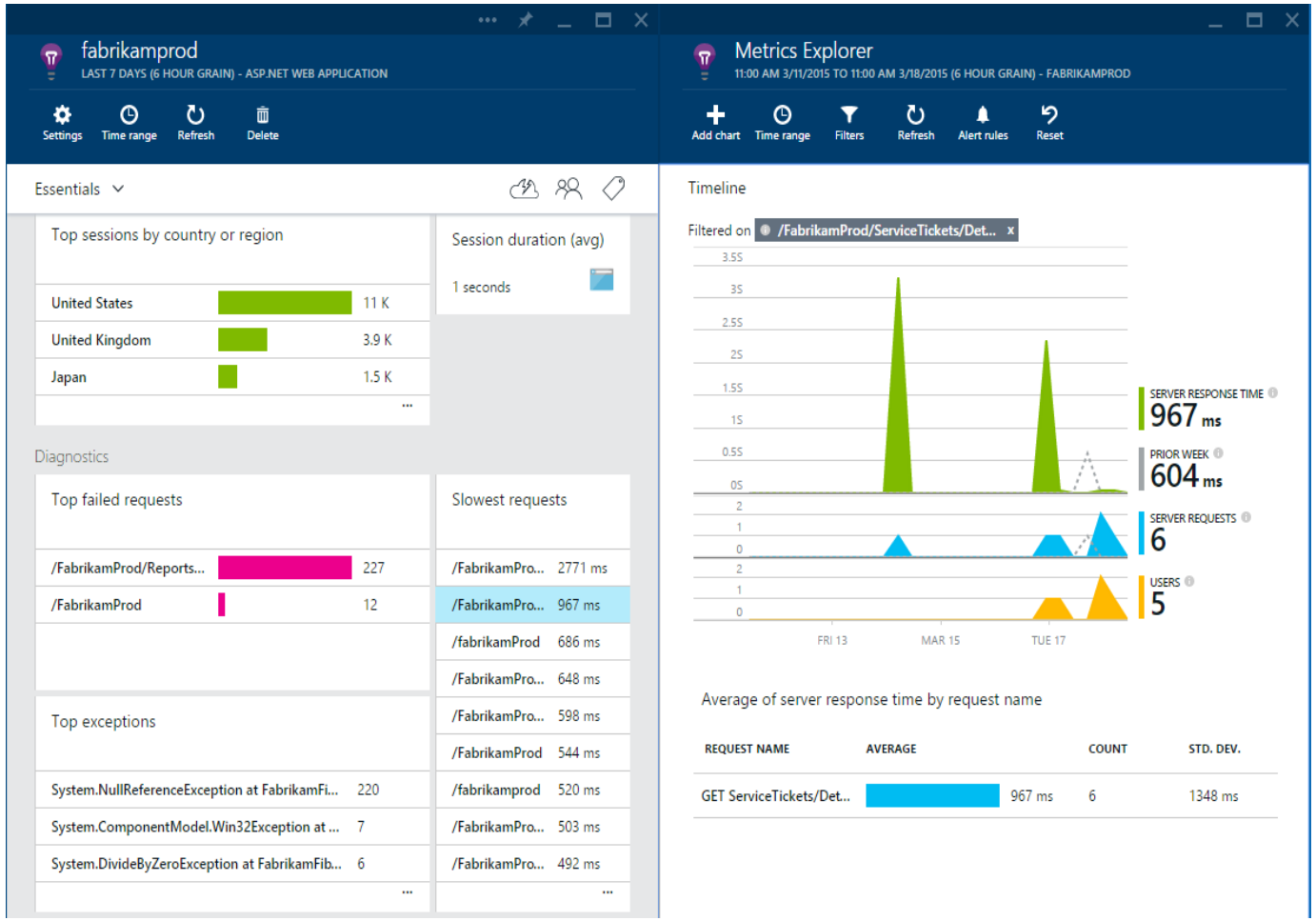


Detailed insights with click of a button

Correlate performance & usage in a single view

Pinpoint problems

Click on specific metric and drill into detailed information



Real World Problems

Service Level Impact

Operational Level Impact

Availability

Responsiveness

Reliability

Latency

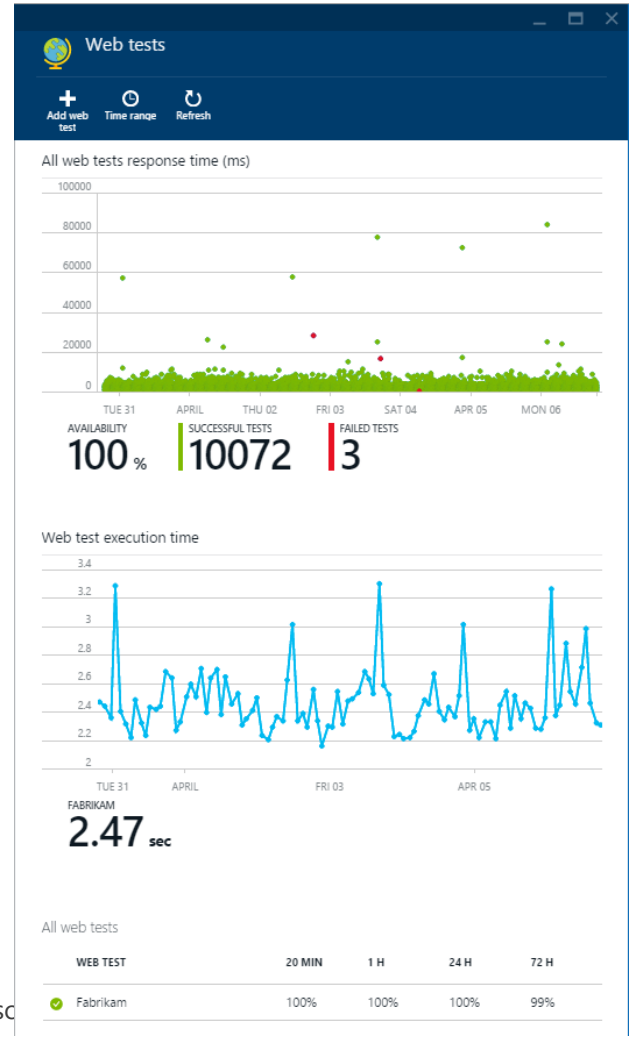
Identify & Triage issues: Availability

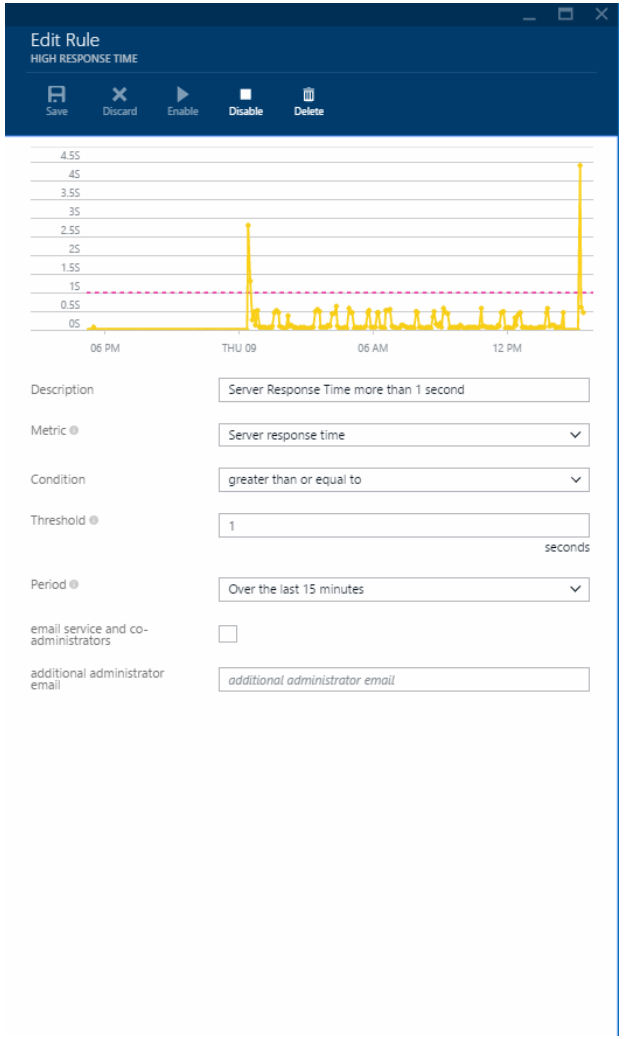
Challenges

- Visibility to application health?
- Visibility to application outages to minimize customer impact?
- Hard to get information to determine impact of issue?

Solutions

- Reduce ‘Mean Time to Detect’
- Ensure service availability with 16 global points of presence
- URL ping tests and rich multi-step web tests
- Threshold based alerts on metrics and perf. counters
- Real time alerts via email





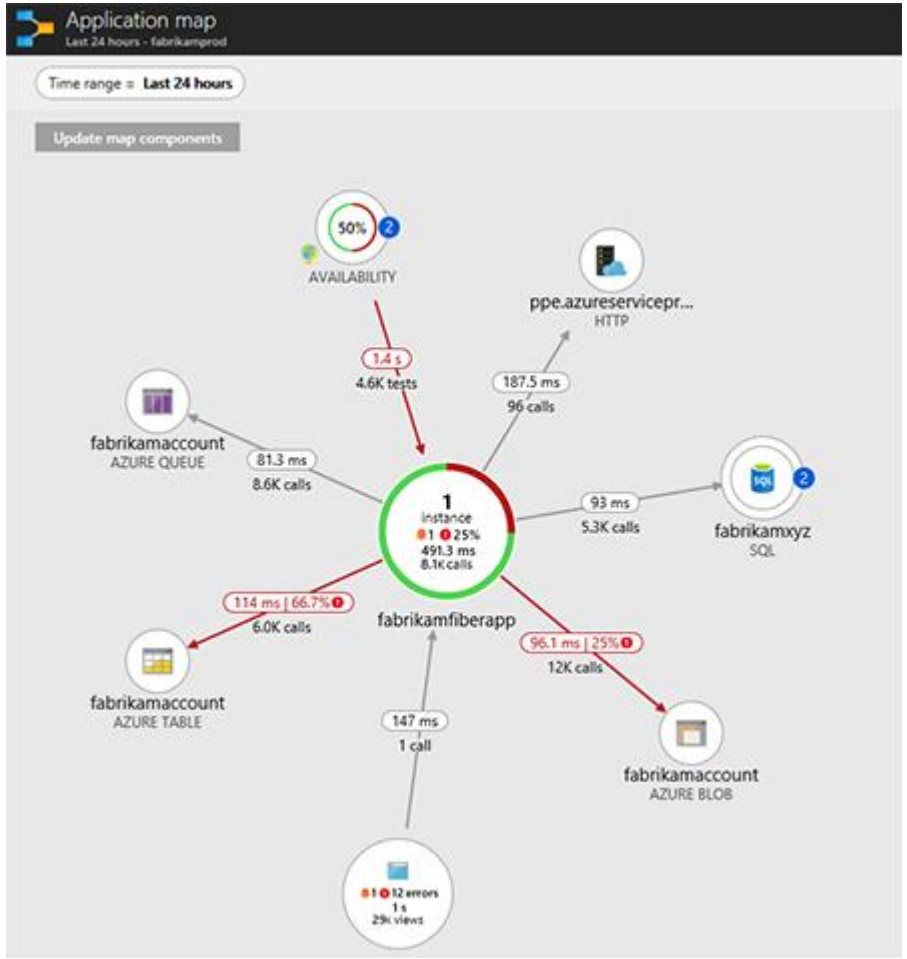
Application Map

Challenges

- Monitor dependencies responsiveness

Solutions

- One Graphical Overview
- Allowing DrillThrough to Performance Analysis



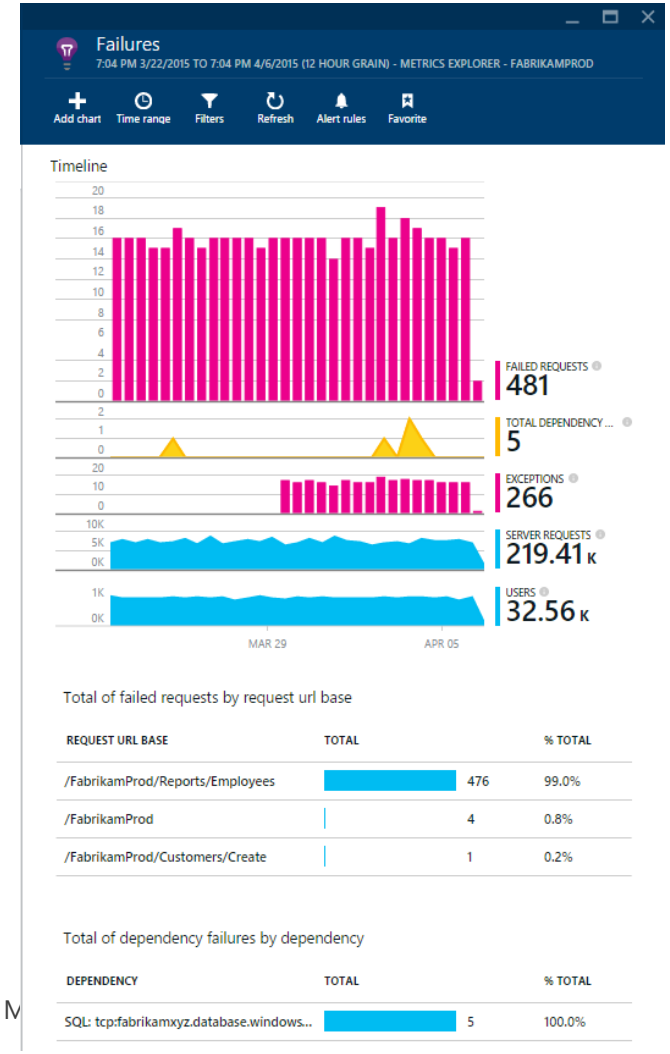
Diagnose & Solve problems: Performance

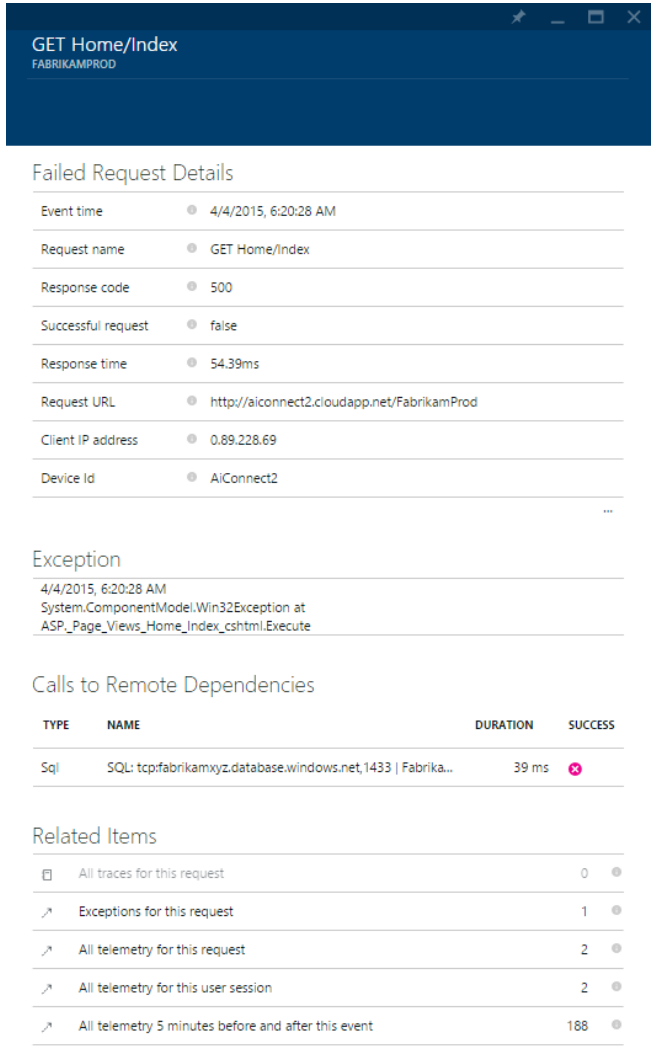
Challenges

- Uncover root cause of issues in production?
- Quickly resolve app crashes?
- Monitor app responsiveness

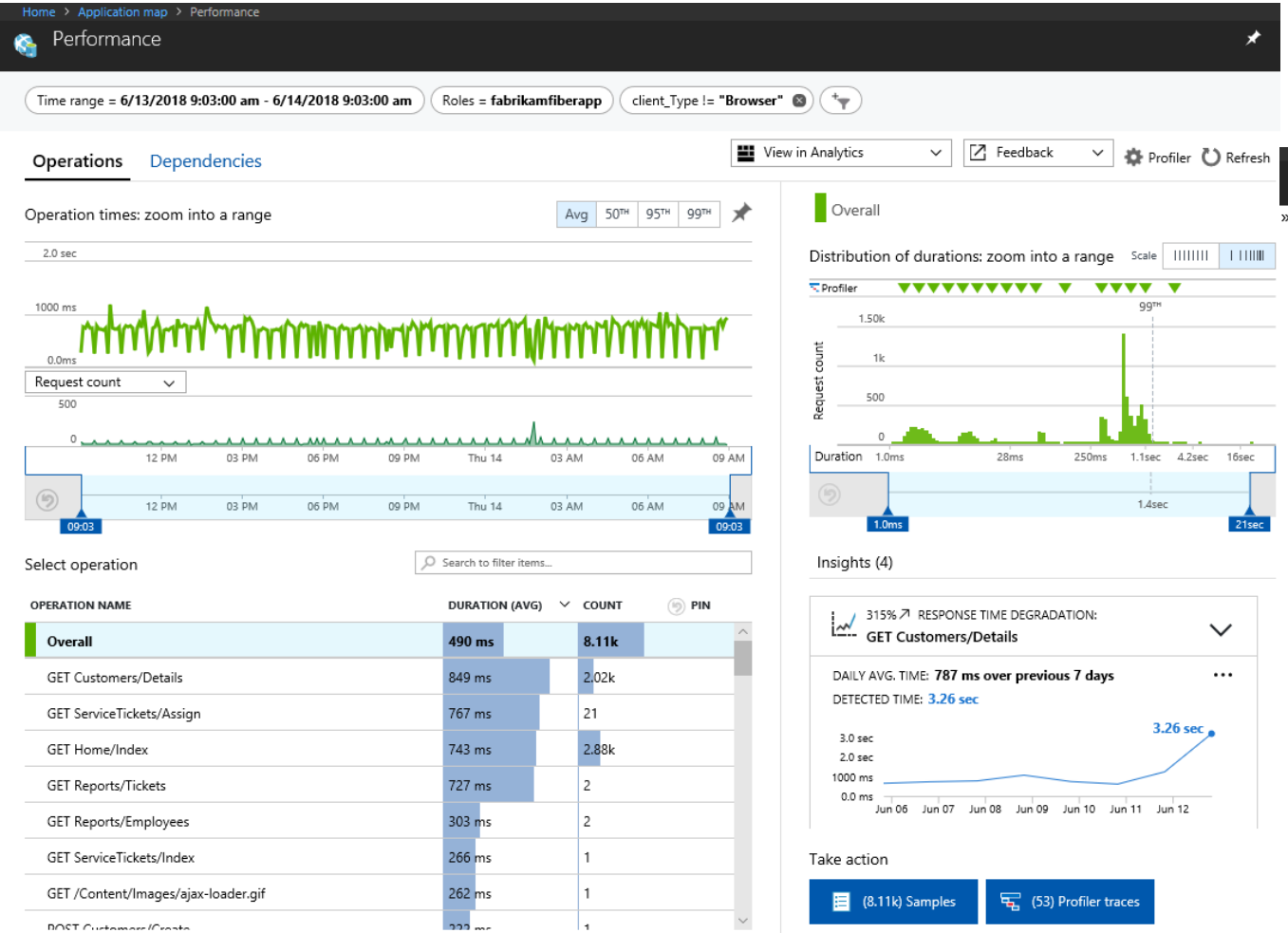
Solutions

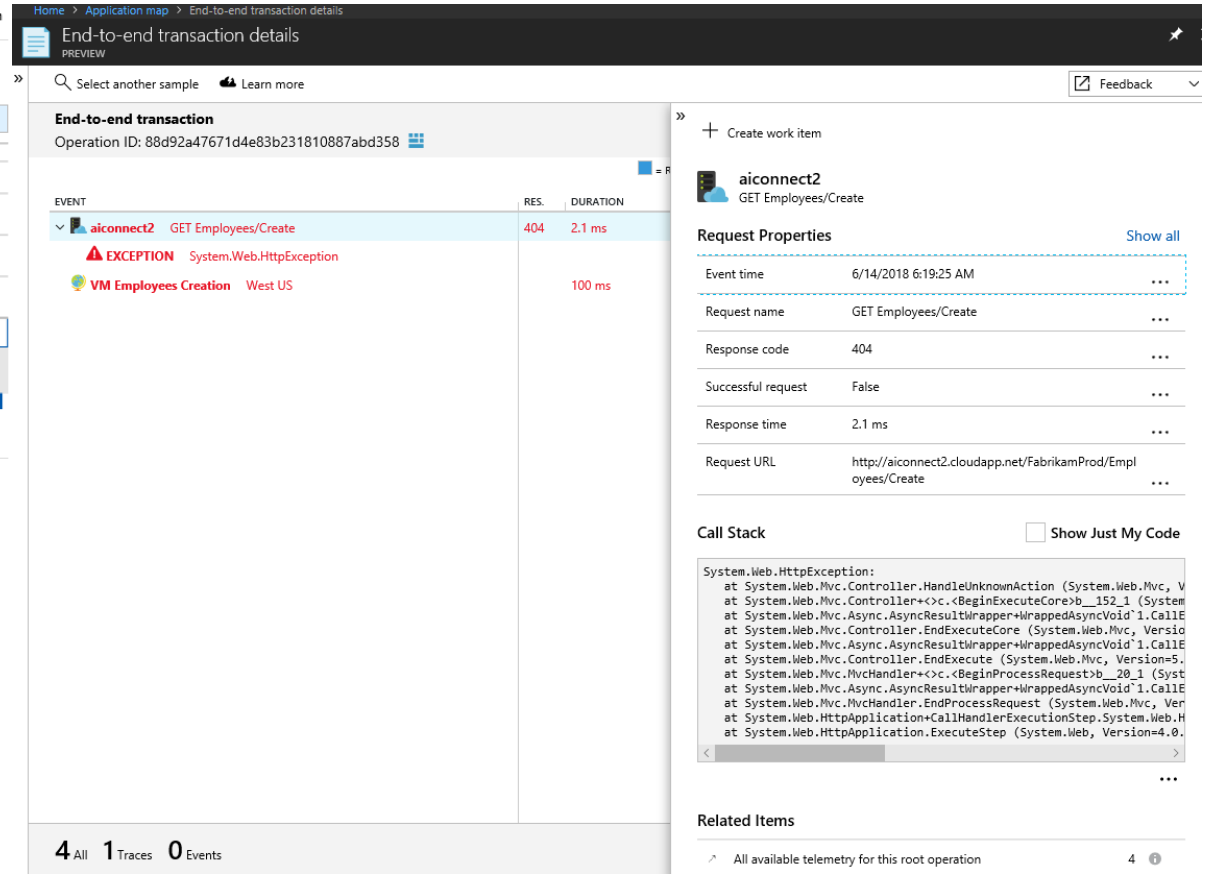
- Reduce ‘Mean Time to Resolve’
- Identify issues with request rate & dependency response times
- Multi-dimensional analyses over custom



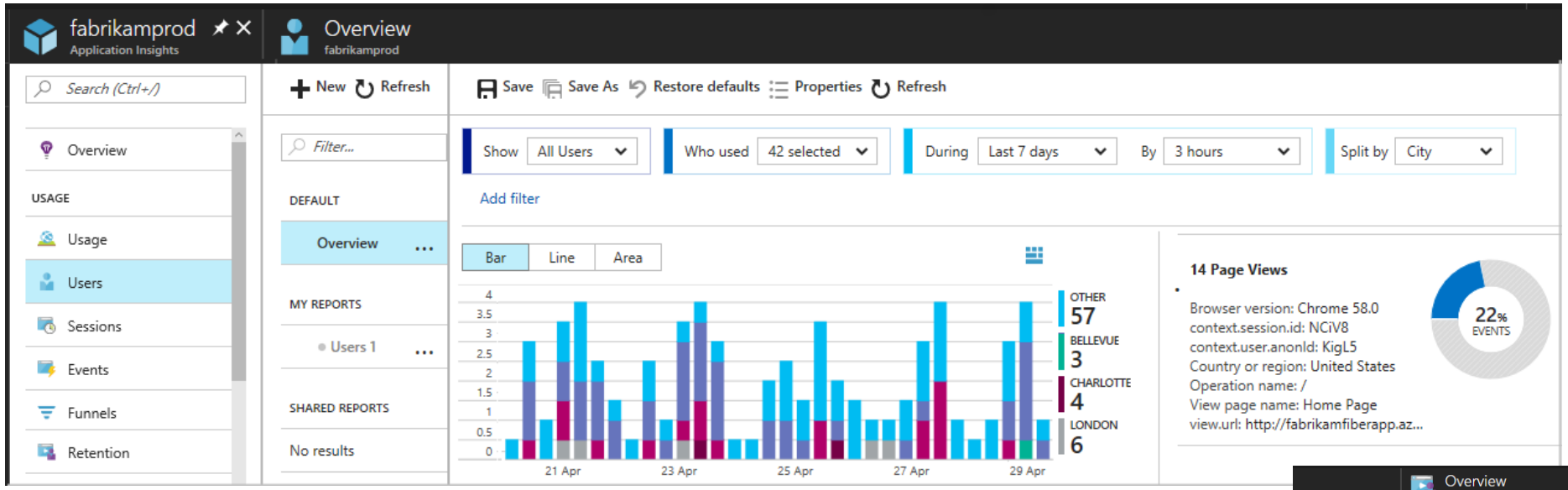


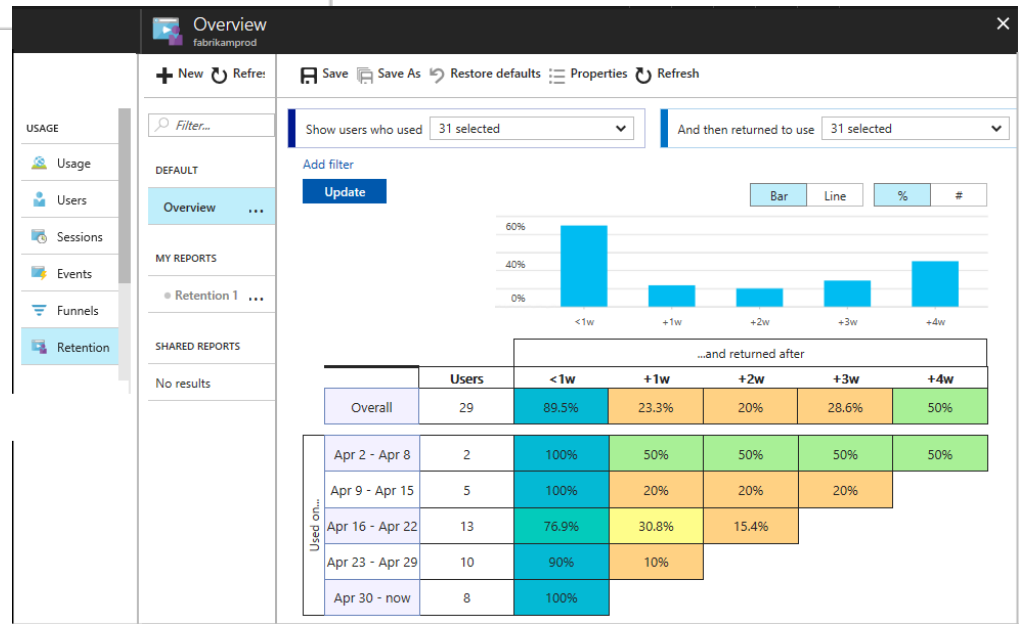
Performance





Usage Analysis with Application Insights





Adding Application Insights

There are 3 ways to apply AI to a web application :

@ Runtime

- Instrument your web app on the server
- Monitor web apps that are already running – No code changes

@ Build time

- Adding Application Insights SDK

@ Both (Build/Runtime)

- Compile SDK into your app code and run apply the run-time extensions

Adding Application Insights

Instrument web apps at runtime with Application Insights Status Monitor

- Here's a summary of what you get by each route:

Feature	Build time	Run time
Requests & exceptions	Yes	Yes
More detailed exceptions		Yes
Dependency diagnostics	On .NET 4.6+, but less detail	Yes, full detail: result codes, SQL command text, HTTP verb
System performance counters	Yes	Yes
API for custom telemetry	Yes	No
Trace log integration ILogger, NLog, log4Net, or System.Diagnostics.Trace	Yes	No
Page view & user data	Yes	No
Need to rebuild code	Yes	No

Adding Application Insights in VS Project

- Adds these files to your project:
  - ApplicationInsights.config
  - Ai*.js
- Installs these NuGet packages:
  - Application Insights (AI) API - core API
  - AI API for Web Applications - server
  - AI API for JavaScript Applications - client
- Inserts items into:
  - Web.config
  - packages.config

Execution mode

Different execution mode when running Application Insights in :

Debug : Development Mode

- Instantaneous send

Release : Production Mode

- Batch send of data

Flexibility and extensibility

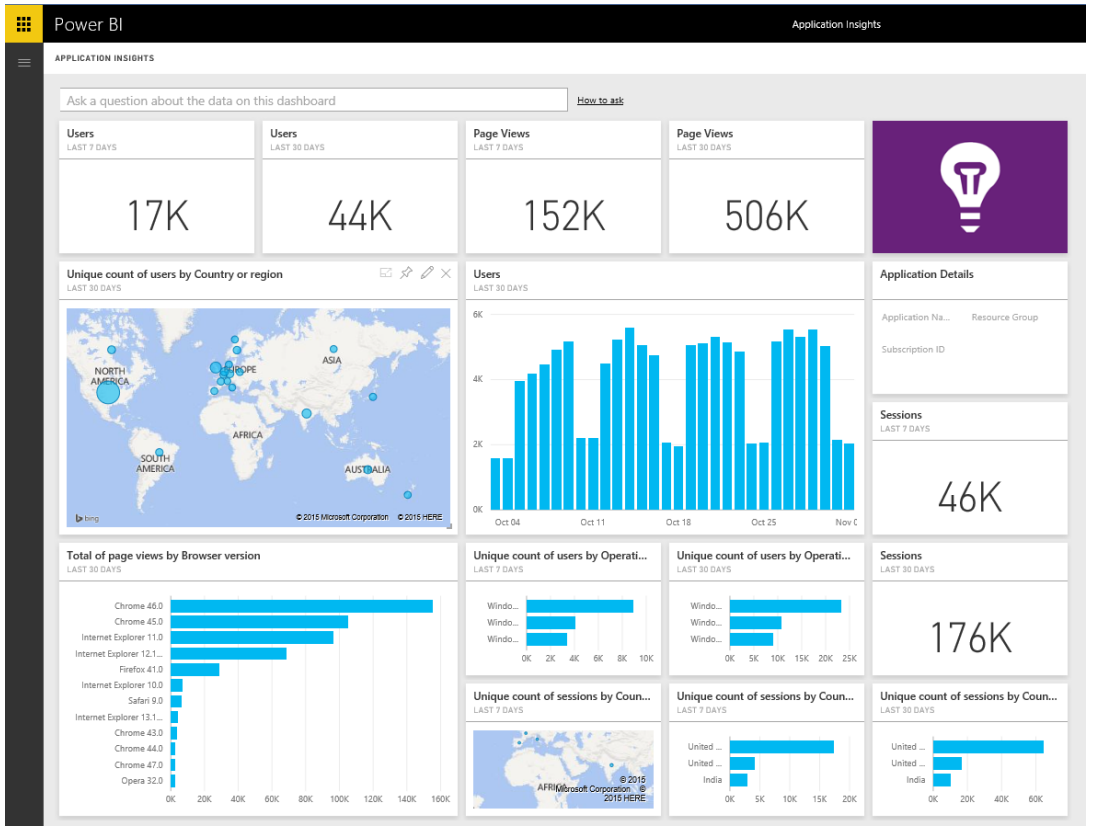
Open Source SDKs to power insights for any web app

Continuously export data to Azure Blob Storage or SQL

Get immediate alerts via emails & web-hooks integration

Visualize data with Power BI Content Pack

Query data via API



Diagnostics Trace

Support

- log4J
- Log4net
- Nlog
- System.Diagnostics.Trace

Enable easy correlation

- Logs traces
- User actions
- Exceptions
- Other events

Track Method

TrackTrace

- Custom trace for dynamic/punctual to fine-tuned query, investigations
- Ex: any logs, warning, etc.

TrackMetric

- Track anything that is purely numeric
- ex : e-commerce selling t-shirt would be tracking number of sold items…

TrackRequest

- Collect information about a request outside of the usual application workflow.

Custom Events

Need to understand how your application is being used.

Can define custom event to track:

- features
- scenarios
- any actions you wish to monitor

```
var tc = new Microsoft.ApplicationInsights.TelemetryClient();
tc.TrackEvent("GameEnd");
```

Using Telemetry Initializer

- Create a custom Initializer to override selected behavior of standard telemetry:
  - Using ITelemetryInitializer

```
// Telemetry initializer class
public class MyTelemetryInitializer : IContextInitializer
{
    public void Initialize (TelemetryContext context)
    {
        context.Properties["AppVersion"] = "v2.1";
    }
}
```

  - Update Global.asax

```
protected void Application_Start()
{
    // ...
    TelemetryConfiguration.Active.ContextInitializers
    .Add(new MyTelemetryInitializer());
}
```

Adding performance counters

- Edit ApplicationInsights.config

```
<ApplicationInsights xmlns="http://schemas.microsoft.com/ApplicationInsights/2013/Settings">
  <TelemetryModules>
    <Add Type="Microsoft.ApplicationInsights.Extensibility.DependencyCollector.DependencyTrackingTelemetryModule,
Microsoft.ApplicationInsights.Extensibility.DependencyCollector" />
    <Add Type="Microsoft.ApplicationInsights.Extensibility.PerfCounterCollector.PerformanceCollectorModule,
Microsoft.ApplicationInsights.Extensibility.PerfCounterCollector">
      <Counters>
        <Add PerformanceCounter="\ASP.NET\Requests Rejected"/>
        <Add PerformanceCounter="\ASP.NET\Application Restarts"/>
      </Counters>
    </Add>
    <Add Type="Microsoft.ApplicationInsights.Extensibility.Implementation.Tracing.DiagnosticsTelemetryModule,
Microsoft.ApplicationInsights" />
    <Add Type="Microsoft.ApplicationInsights.Extensibility.Web.RequestTrackingTelemetryModule,
Microsoft.ApplicationInsights.Extensibility.Web" />
    <Add Type="Microsoft.ApplicationInsights.Extensibility.Web.ExceptionTrackingTelemetryModule,
Microsoft.ApplicationInsights.Extensibility.Web" />
    <Add Type="Microsoft.ApplicationInsights.Extensibility.Web.DeveloperModeWithDebuggerAttachedTelemetryModule,
Microsoft.ApplicationInsights.Extensibility.Web" />
  </TelemetryModules>
```

Demo: Application Insights (Portal)

Typical adhoc and iterative diagnostic process

How are we making this easier?

Make the data available for queries

Simple language for adhoc and iterative queries

Fast enough to be highly interactive

Perform Inline visualizations

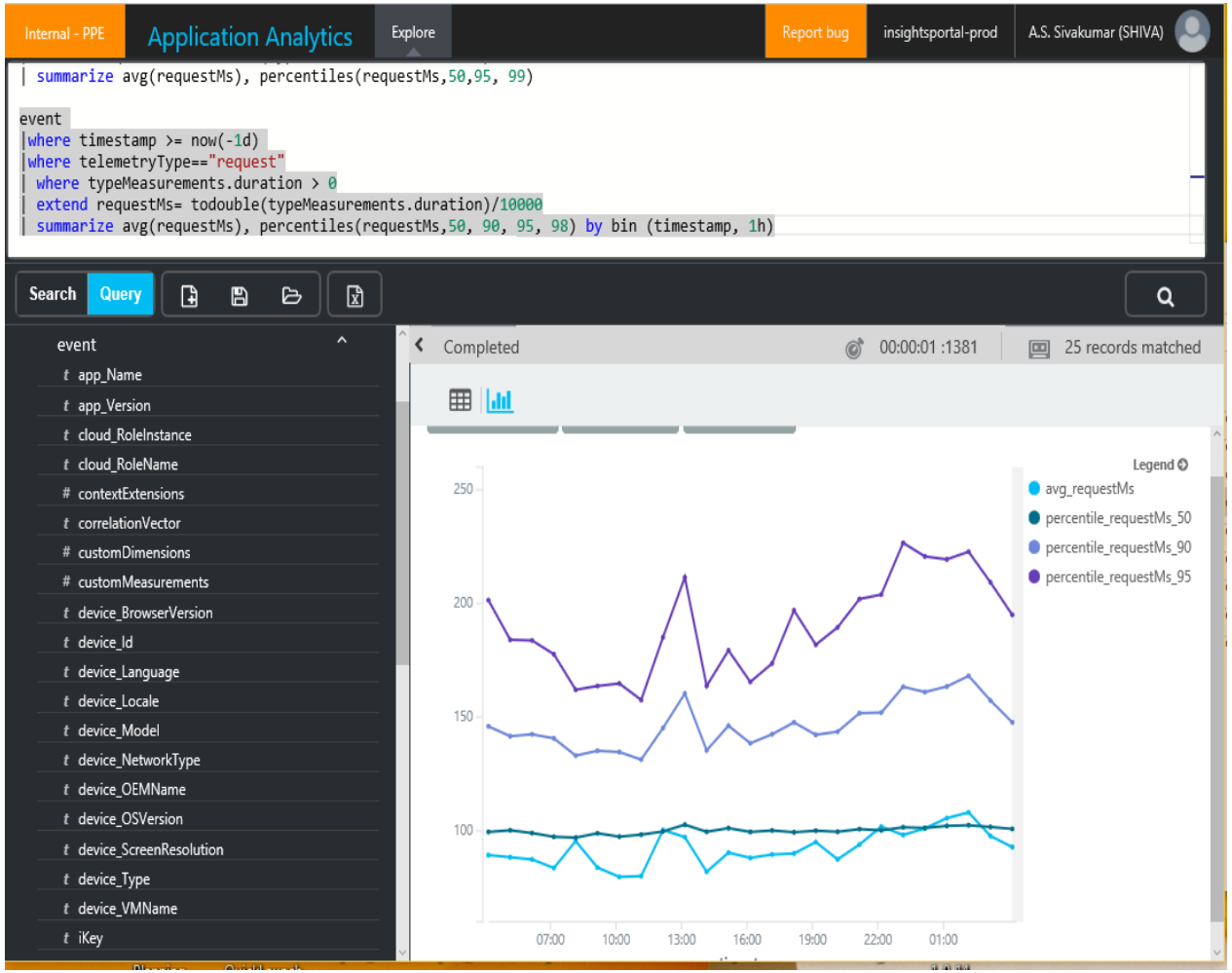
Operationalize the findings

Ad-hoc queries and interactive analytics

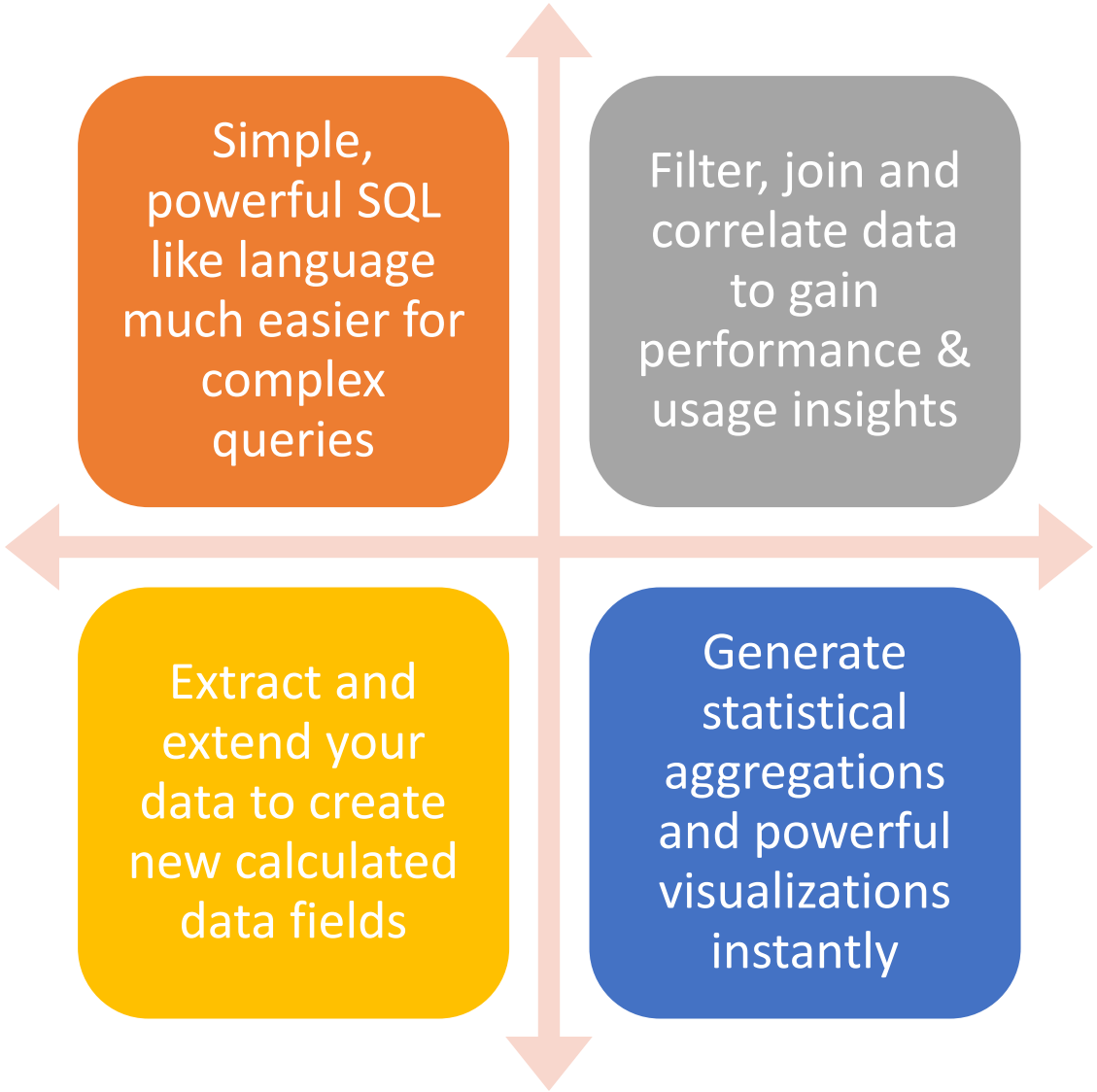
Diagnosing across app stack is hard unless various perspectives connected

New and powerful big data query engine for all your app telemetry and root-cause analysis

Ad-hoc queries and full-text search helps answer tough questions instantly



Powerful query language



Powerful query language

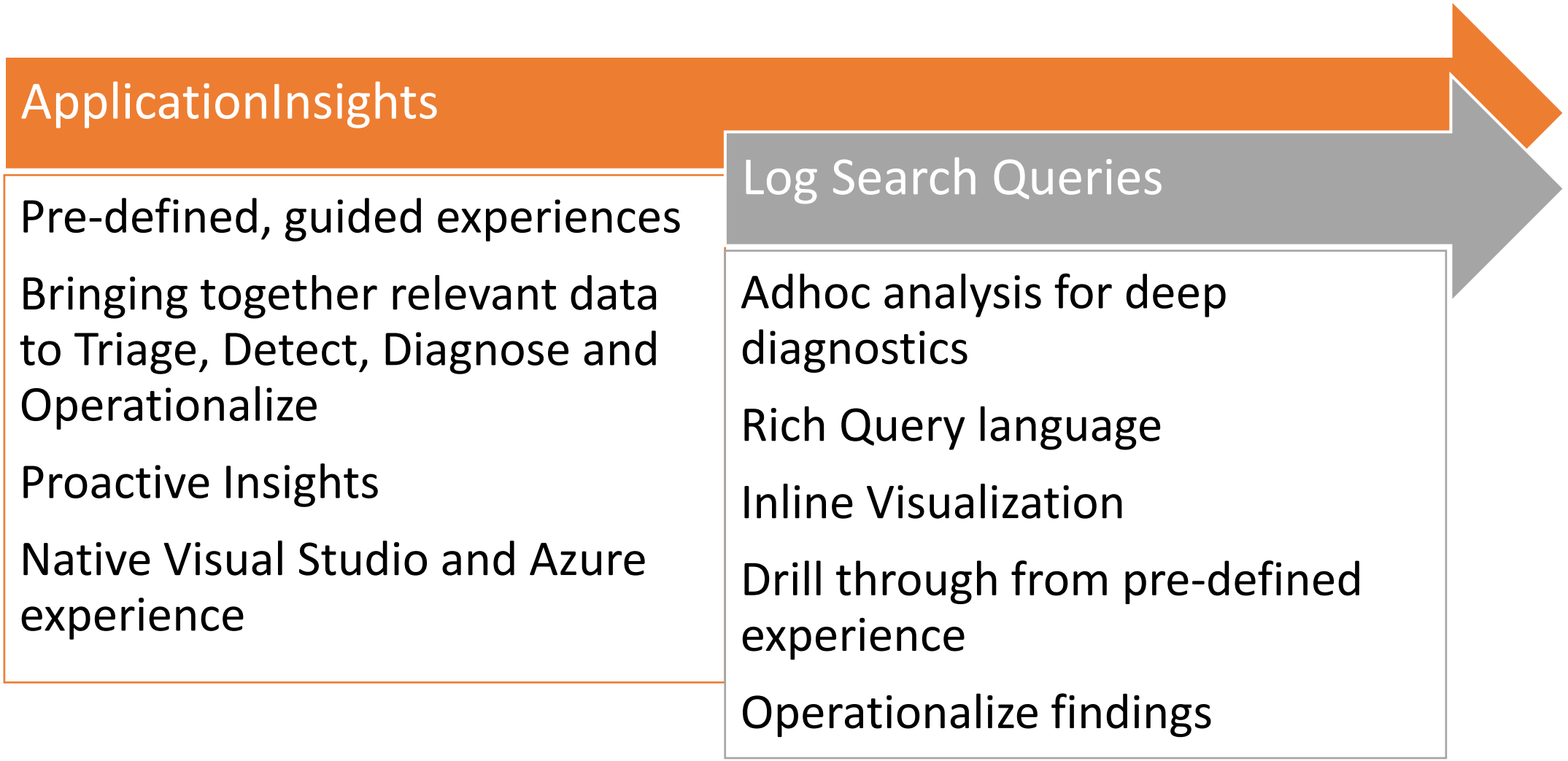
Service Performance	• Transaction response time • DB response time • Exceptions • CPU and memory utilization • Data center location
App & Business Metrics	• Custom events • Custom dimensions • App KPIs • Conversion rates
Customer Experience	• Response time • Errors • Geo location • Browser type and version • Session

Examples for interactive queries

- What are the 5th, 50th, and 99th percentiles of request duration for my app in the last 24 hours
- How many requests in the last 7 days met my SLA
- What are top exceptions today and how prevalent is a particular exception across my customers
- What are the typical page flows for my application in the last 7 days

and many more…

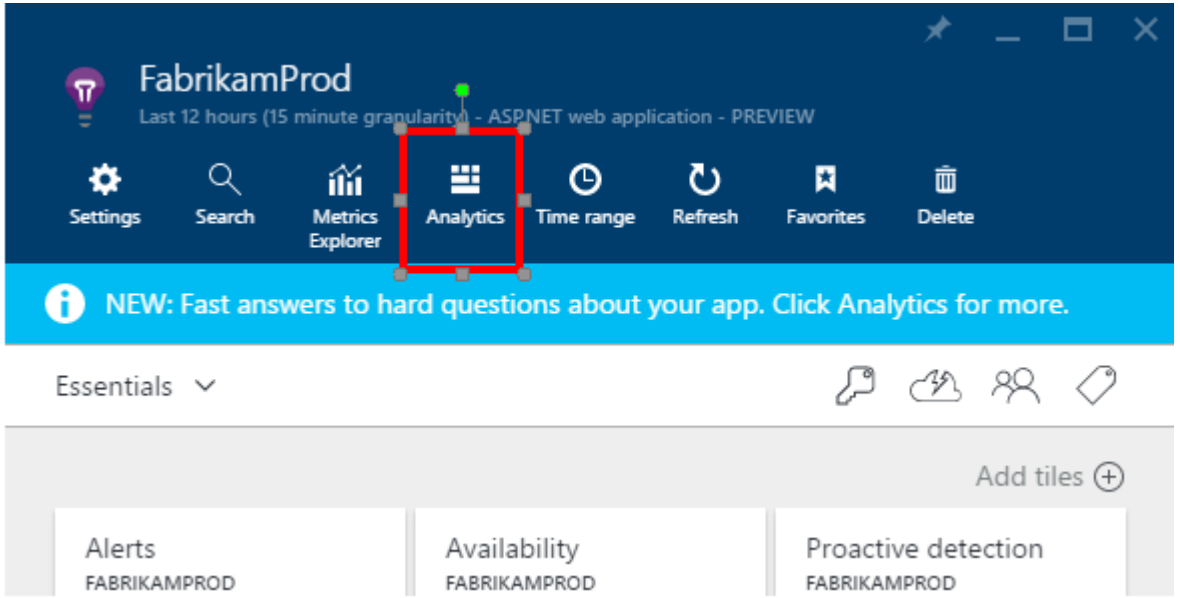
Log Search Queries & Application Insights



What can you do today

Start querying your telemetry by logging onto Application Insights

We'll look at it in detail later on



Provide your feedback and help us improve


Microsoft