OPEN SOURCE APPLICATION PERFORMANCE MONITORING FOR THE CROWD: INSPECTIT

JUNE 17TH, 2016,

IVAN SENIĆ, CHRISTOPH HEGER, MARIO MANN

WORKSHOP KARLSRUHER ENTWICKLERTAG

NOVATEC

AGENDA

1. Introduction
2. Setup of inspectIT
3. Configuration of the Instrumentation
4. -- Break --
5. Analysis of Performance Problems
6. Collaboration of Dev and Ops
7. Questions and Answers

Who are we?

Ivan Senić

ivansenic

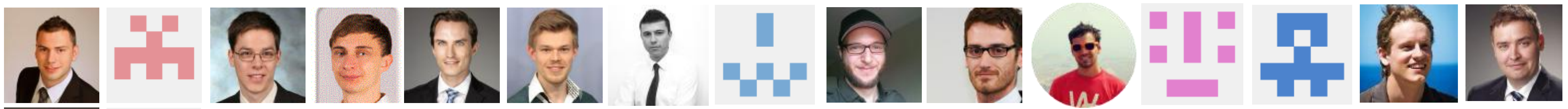
Mario Mann

mariomann

Christoph Heger

hegerchr

… from the inspectIT APM Open Source Project



18 contributors and growing

Why is performance important?

User Experience

Response Time Limit	Perception
0.1 second	Users feel their actions are directly causing something to happen
1.0 second	Users feel they are navigating freely and stay focused on their current train of thought
10.0 seconds	Users feel a break of the flow, get impatient while waiting and leave the site

Jacob Nielsen, Power of 10: Time Scales in User Experience, 2009

Reports

- 49% of online shoppers expect load times of <=2 seconds
- 50% of online shoppers abandon page if load time is > 2 seconds
- 42% of men and 35% of women don‘t give a company a second chance after experiencing slow load times

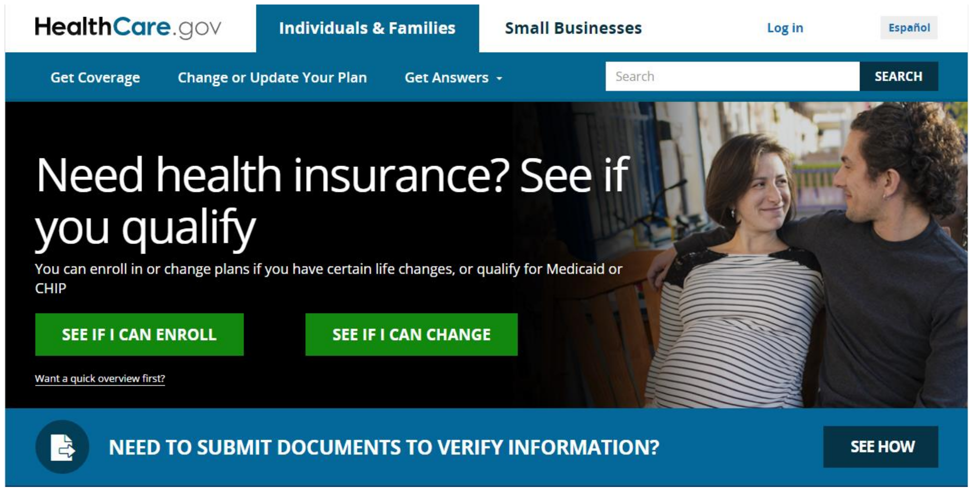
Business Impact

- Brand perception, conversions, revenue, shopping card abandonment, page views, and search engine rankings

The HealthCare.gov Fiasco

Summary
[Kevin Surace, fastcompany.com; Byron Wolf, CNN]

- Among the worst software launches in history
- Hundreds of millions of dollars were spent
- Outages and slow response times were the norm
- U.S. government had to apologize
- Companies like Google and Red Hat rushed to the rescue


HealthCare.gov Individuals & Families Small Businesses Log in Español
Get Coverage Change or Update Your Plan Get Answers Search SEARCH

Need health insurance? See if you qualify

You can enroll in or change plans if you have certain life changes, or qualify for Medicaid or CHIP

SEE IF I CAN ENROLL SEE IF I CAN CHANGE

Want a quick overview first?

NEED TO SUBMIT DOCUMENTS TO VERIFY INFORMATION? SEE HOW

[Image: healthcare.gov]

- 125k plus concurrent users peak
- Load times between 6 and 22 seconds depending on business transaction

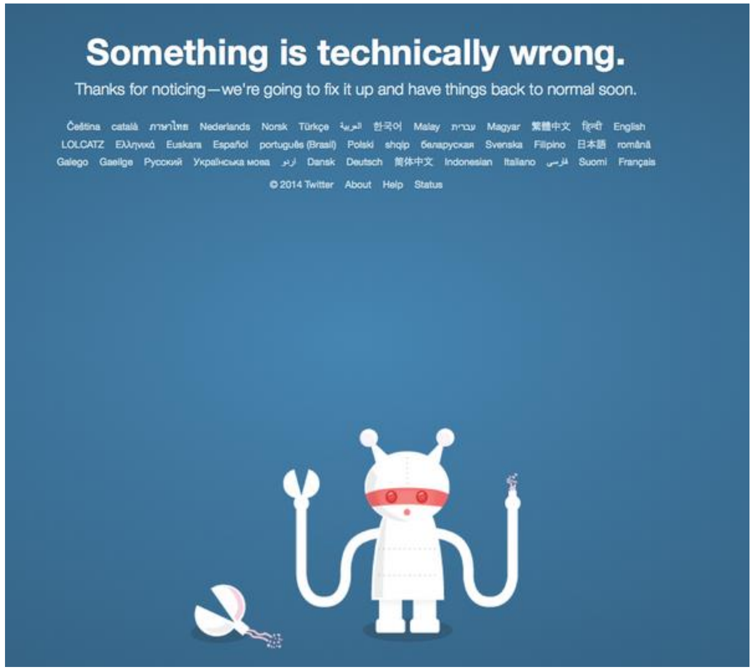
Never actually tested for scalability and performance before its launch

Ellen DeGeneres‘ Oscars selfie crashes Twitter

Summary

- 2.6 million retweets in about 2 hours

- 20 minutes service disruption



Something is technically wrong.

Thanks for noticing—we're going to fix it up and have things back to normal soon.

Čeština català ภาษาไทย Nederlands Norsk Türkçe العربية 한국어 Malay עברית Magyar 繁體中文 हिन्दी English
LOLCATZ Ελληνικά Euskara Español português (Brasil) Polski shqip беларуская Svenska Filipino 日本語 română
Galego Gaeilge Русский Українська мова اردو Dansk Deutsch 简体中文 Indonesian Italiano فارسی Suomi Français

© 2014 Twitter About Help Status

[Image: Twitter.com]

[Image: Ellen DeGeneres]

Be aware of the unexpected

Metrics

Good Metrics

- Latency (response time of requests, methods, third-party services, etc.)
- Utilization (CPU, memory, etc.)
- Platforms (devices, browsers, apps, resolutions, geolocation, etc.)

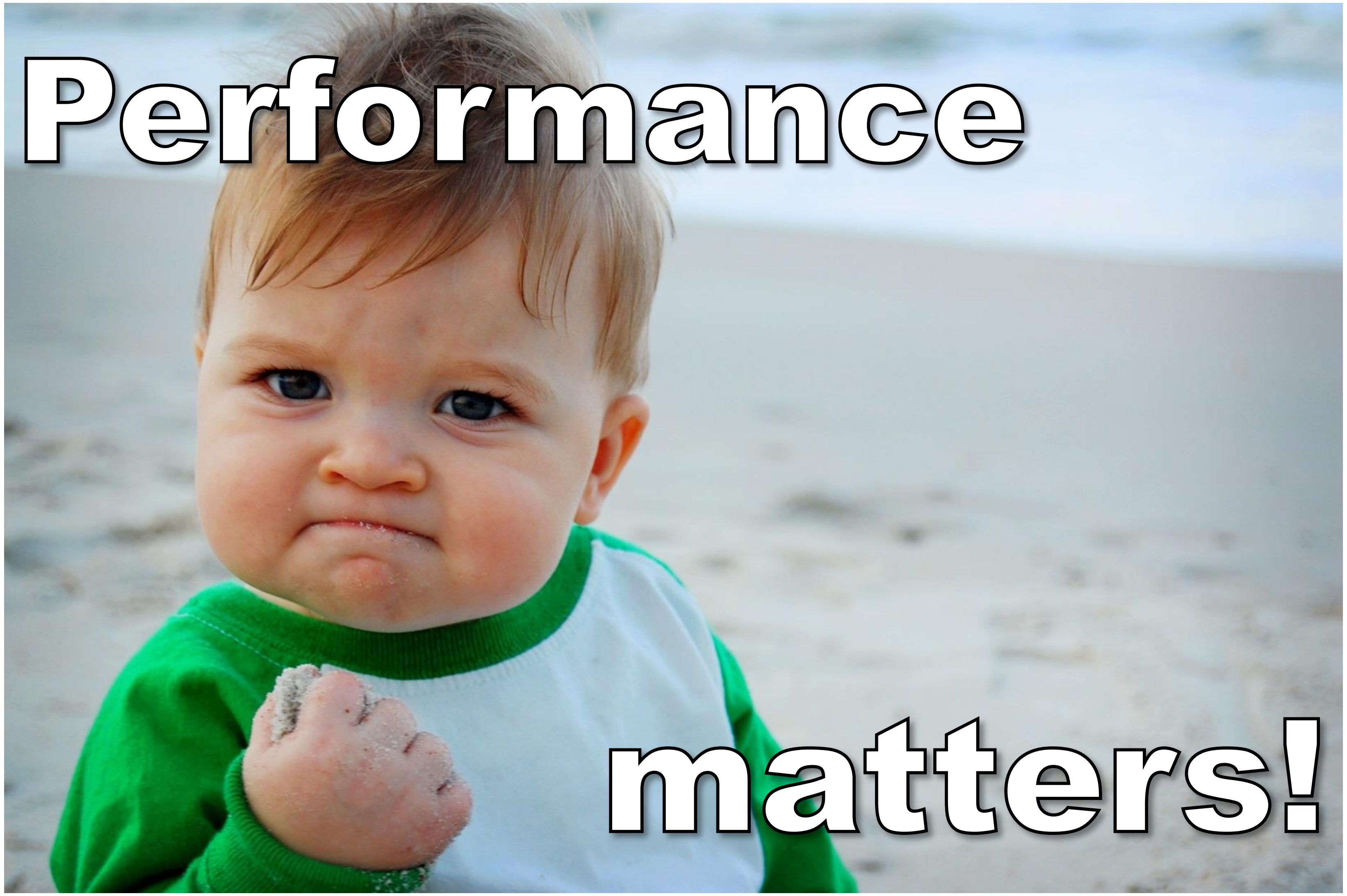
„[…] what you measure you improve […]“ – Larry Dragich

Good Practices

- Complete coverage of the entire stack for business transactions (no siloed approaches)
  - Transaction-level detail
  - Code level
  - Correlated across all component tiers
  - Any device (browser, smartphone, tablet, etc.)
- Selective transaction-level business context capture
  - e.g., user, shopping card details like product, number of items, revenue
- Synthetic probing for availability and general health

Recommendation: Gil Tene – How NOT to Measure Latency, https://youtu.be/lJ8ydIuPFeU

Take Away

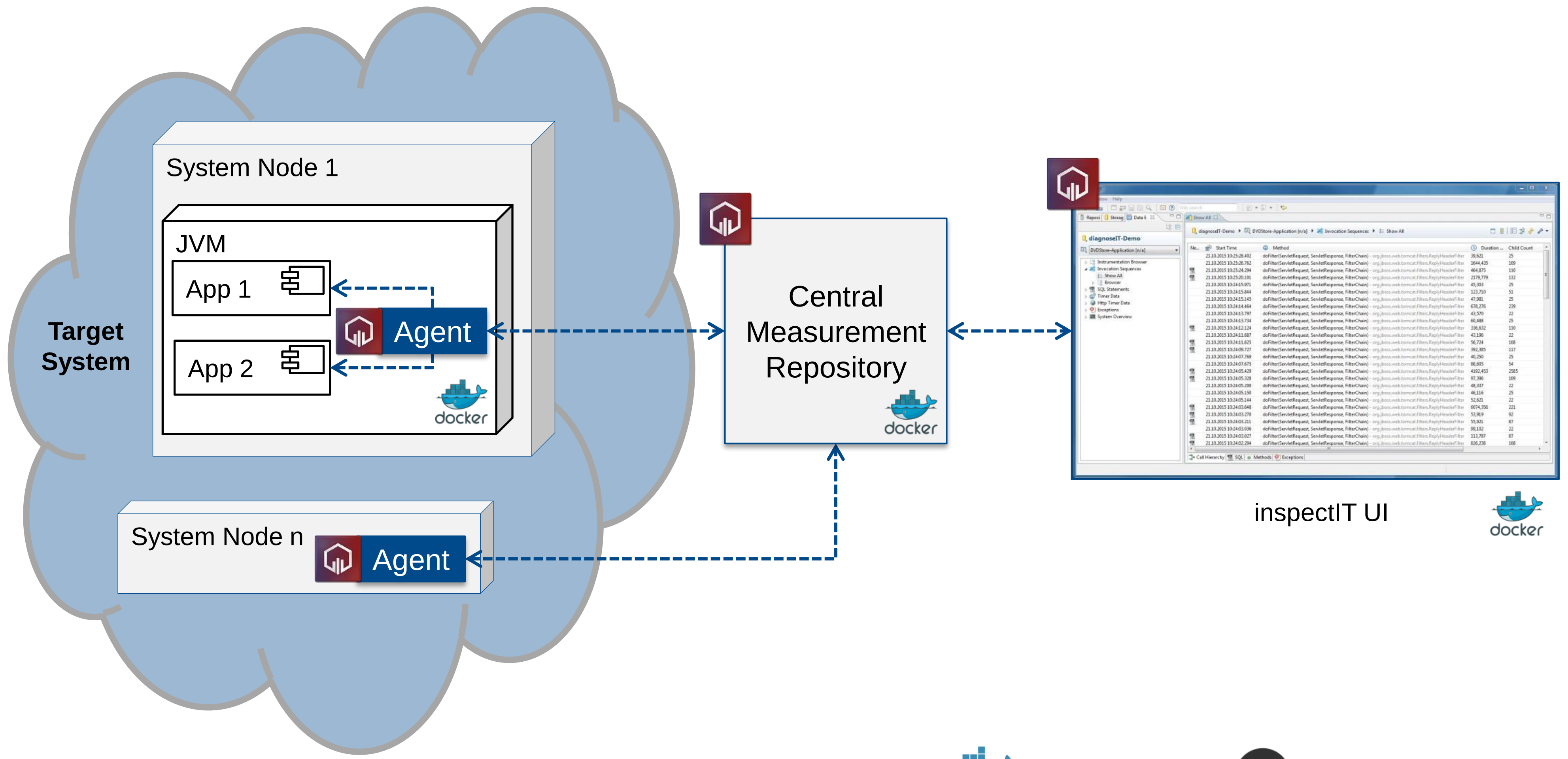


[Image source: http://content.hollywire.com/sites/default/files/Success-Kid.jpg]

INSPECTIT
BEHIND THE SCENES

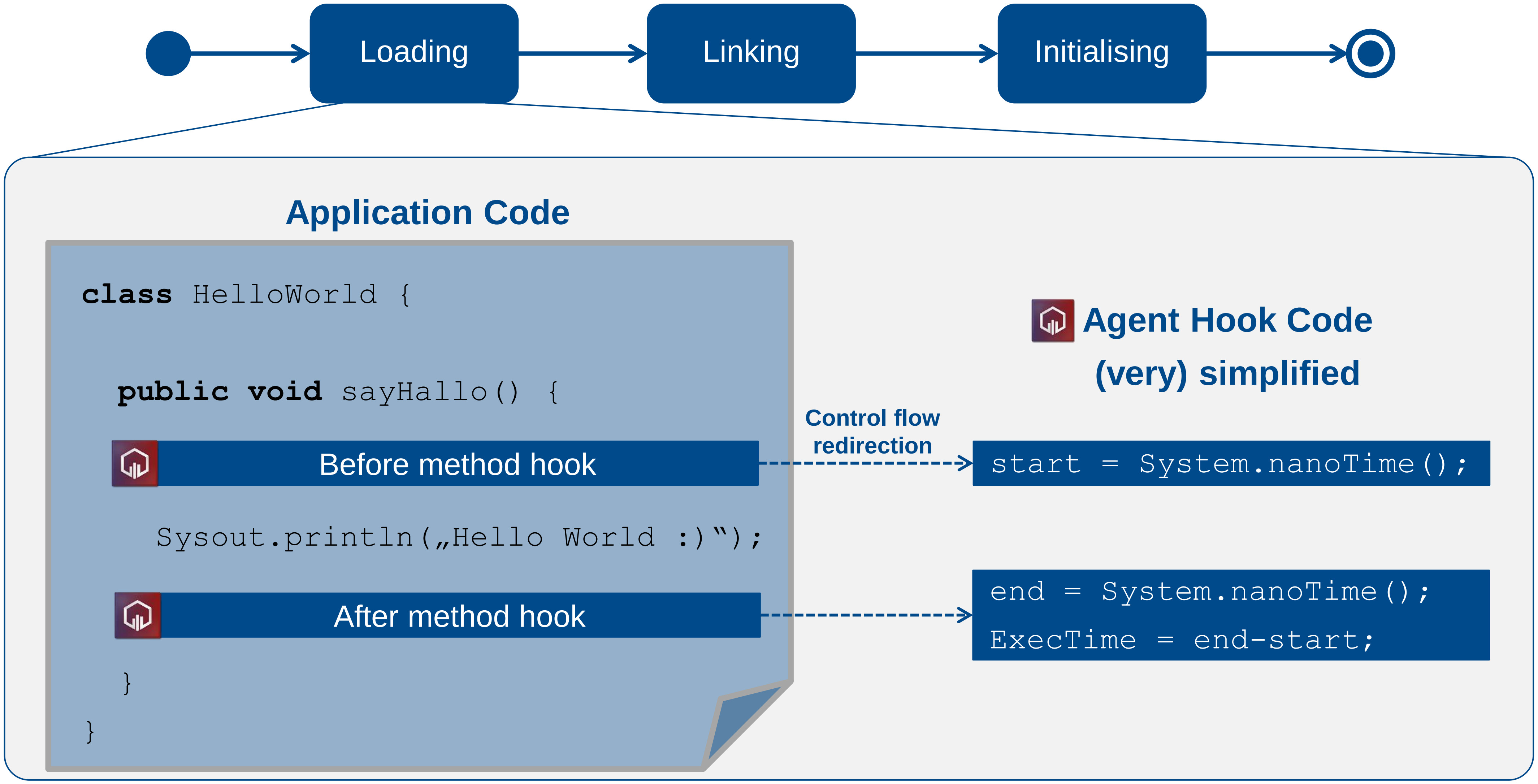
Overview

Agent-based Approach



inspectit   inspectIT-docker

Hooking into class loading



```
class HelloWorld {

  public void sayHallo() {
    Before method hook
    Sysout.println(„Hello World :)“);
    After method hook
  }
}
```

```
start = System.nanoTime();
```

```
end = System.nanoTime();
ExecTime = end-start;
```

Sensors

Sensor Type	Information
Timer	Method execution time
Invocation Sequence	Method execution tree (trace)
SQL	SQL query, parameter bindings
Logging	Log message
Exception	Java exception
HTTP Data	Servlet methods, HTTP parameters
System data	e.g., CPU utilization, memory utilization, number of loaded classes
JMX	e.g., thread pool size

SETUP

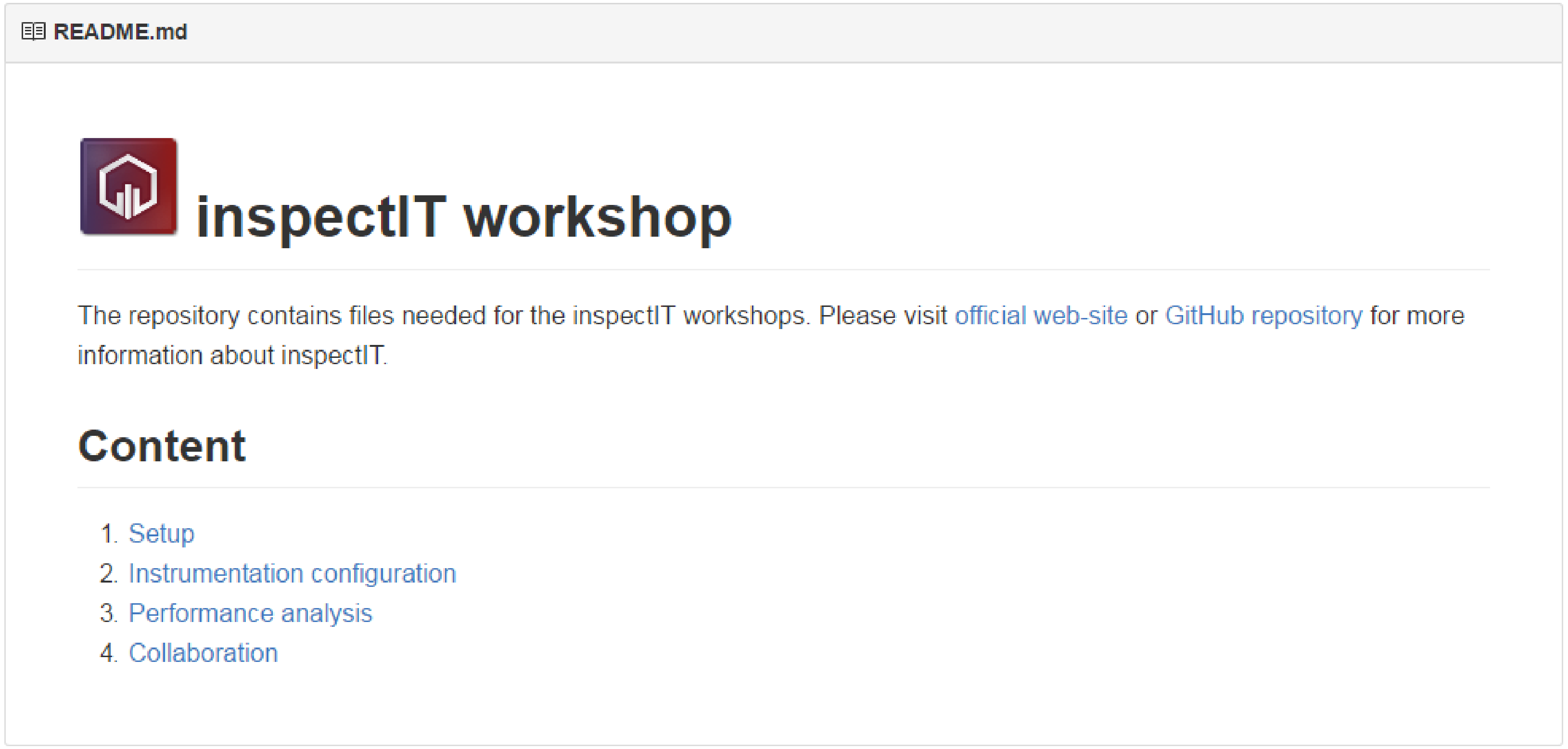
USB Sticks

Content of USB Stick

- inspectIT
  - Agent
  - Central Measurement Repository
  - UI
- DVDStore Sample Application
- Workshop Tutorial

Workshop Description

https://github.com/inspectit-labs/workshop



README.md

inspectIT workshop

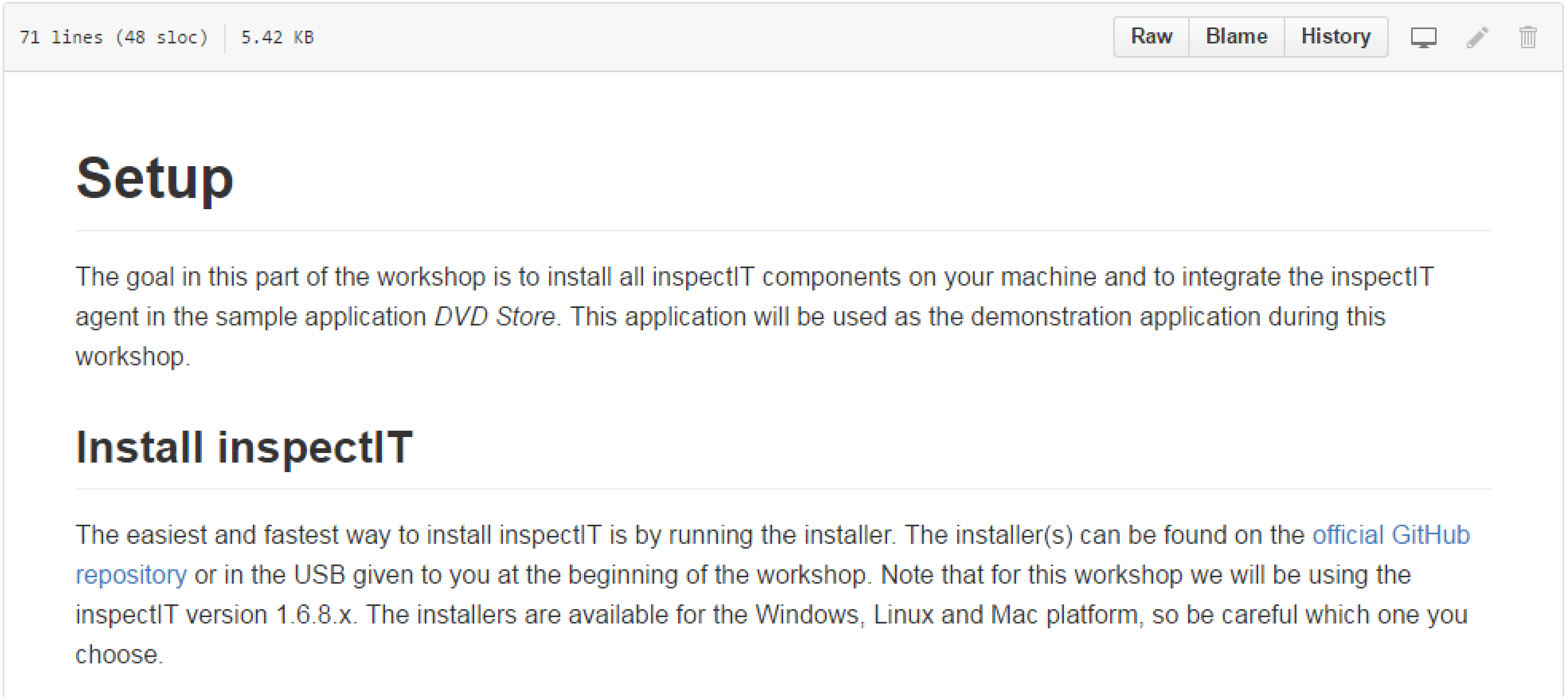
The repository contains files needed for the inspectIT workshops. Please visit official web-site or GitHub repository for more information about inspectIT.

Content

1. Setup
2. Instrumentation configuration
3. Performance analysis
4. Collaboration

Practice Time

https://github.com/inspectit-labs/workshop/blob/master/SETUP.md



Setup

The goal in this part of the workshop is to install all inspectIT components on your machine and to integrate the inspectIT agent in the sample application *DVD Store*. This application will be used as the demonstration application during this workshop.

Install inspectIT

The easiest and fastest way to install inspectIT is by running the installer. The installer(s) can be found on the official GitHub repository or in the USB given to you at the beginning of the workshop. Note that for this workshop we will be using the inspectIT version 1.6.8.x. The installers are available for the Windows, Linux and Mac platform, so be careful which one you choose.

INSTRUMENTATION CONFIGURATION

Instrumentation Configuration

Why is instrumentation important?

- Defines kind/amount of data you will see in inspectIT
- Defines how much overhead you will add to the monitored application

Instrumentation Configuration

What is meaningful instrumentation?

- Rule: get as much as possible information with as less as possible instrumentation points
- Note: the configuration applies to the complete JVM

Not a good idea:

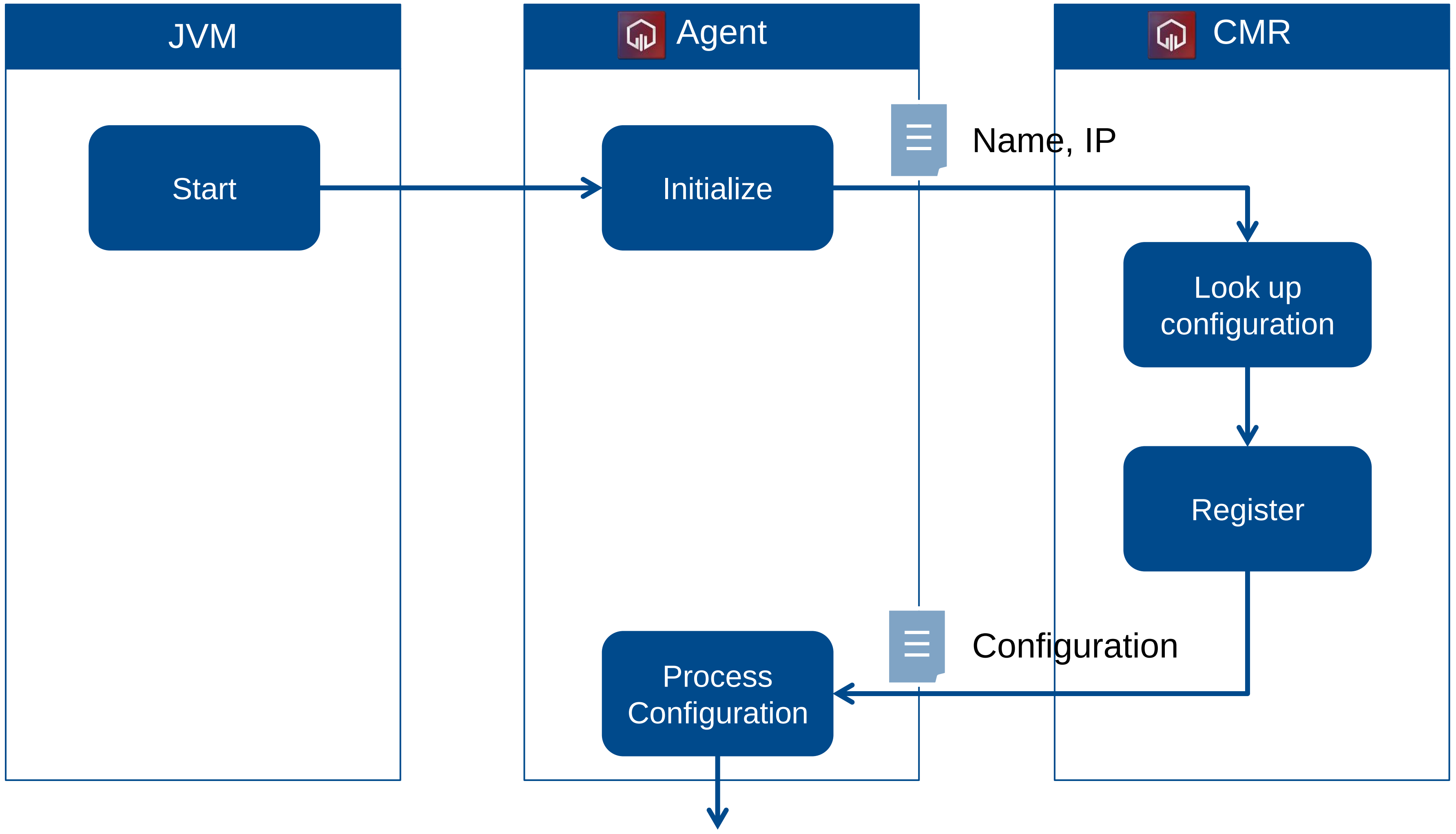
- Methods like toString(), equals() & hashCode()
- Getters and setters
- Any method that is executed often and it’s expected to be executed fast

Better:

- Services, DAOs, database calls, remote calls, etc.
- From experience: method that perform some work (execute, perform, calculate, transform, load, etc..) ☺

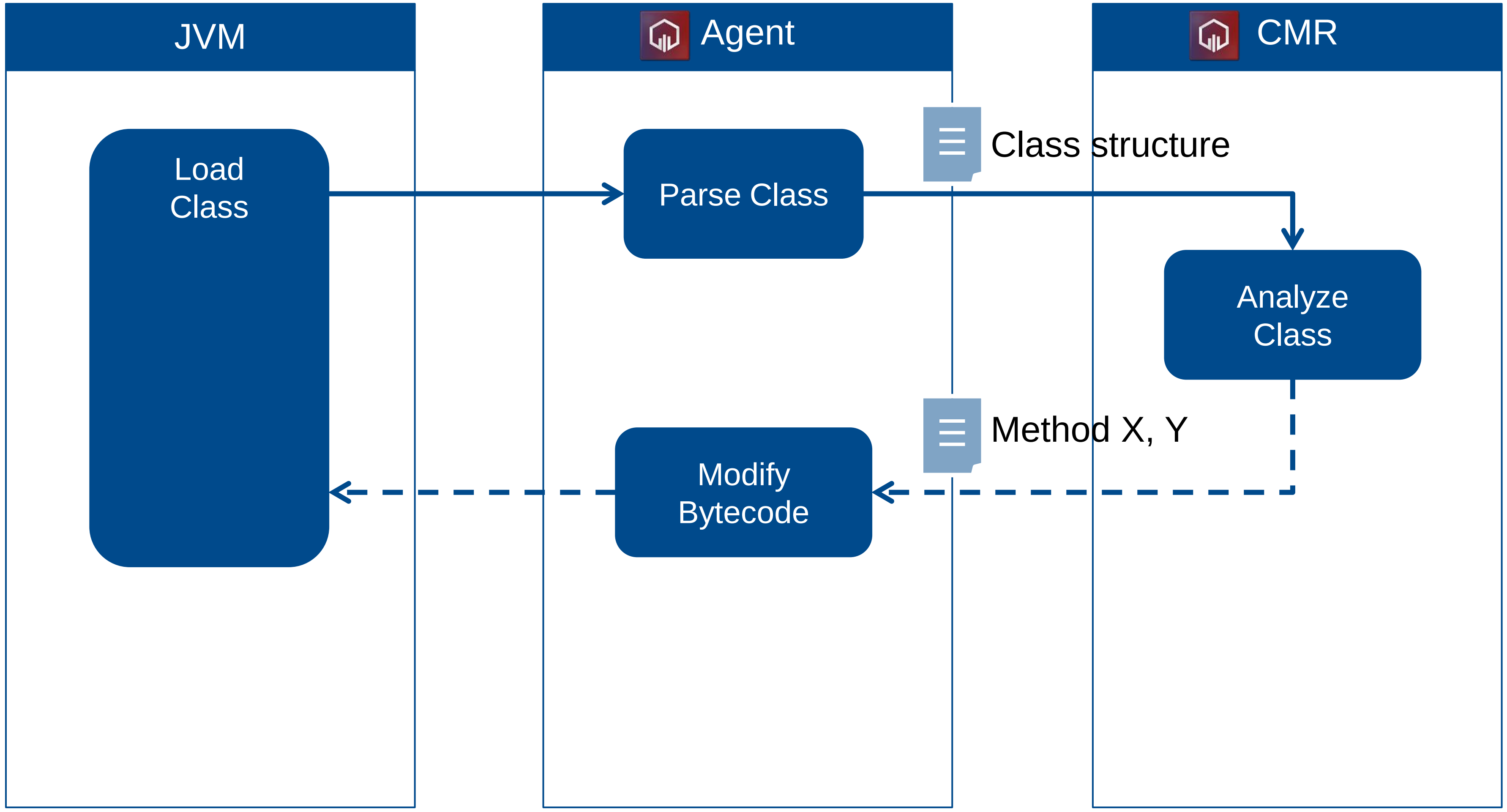
Agent Registration

How instrumentation works in inspectIT?



Method Instrumentation

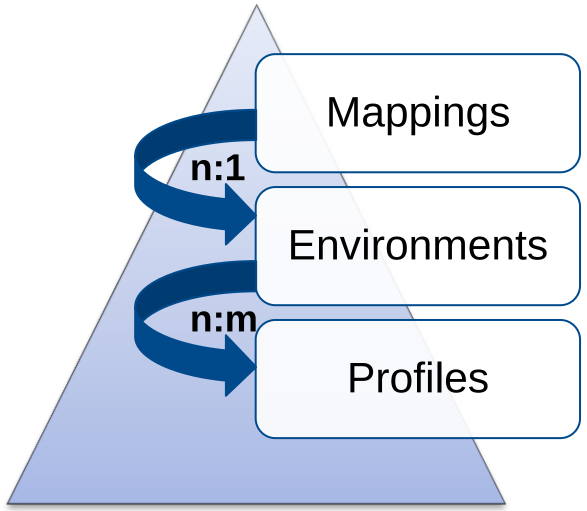
How instrumentation works in inspectIT?



Instrumentation Configuration

Configuration concepts

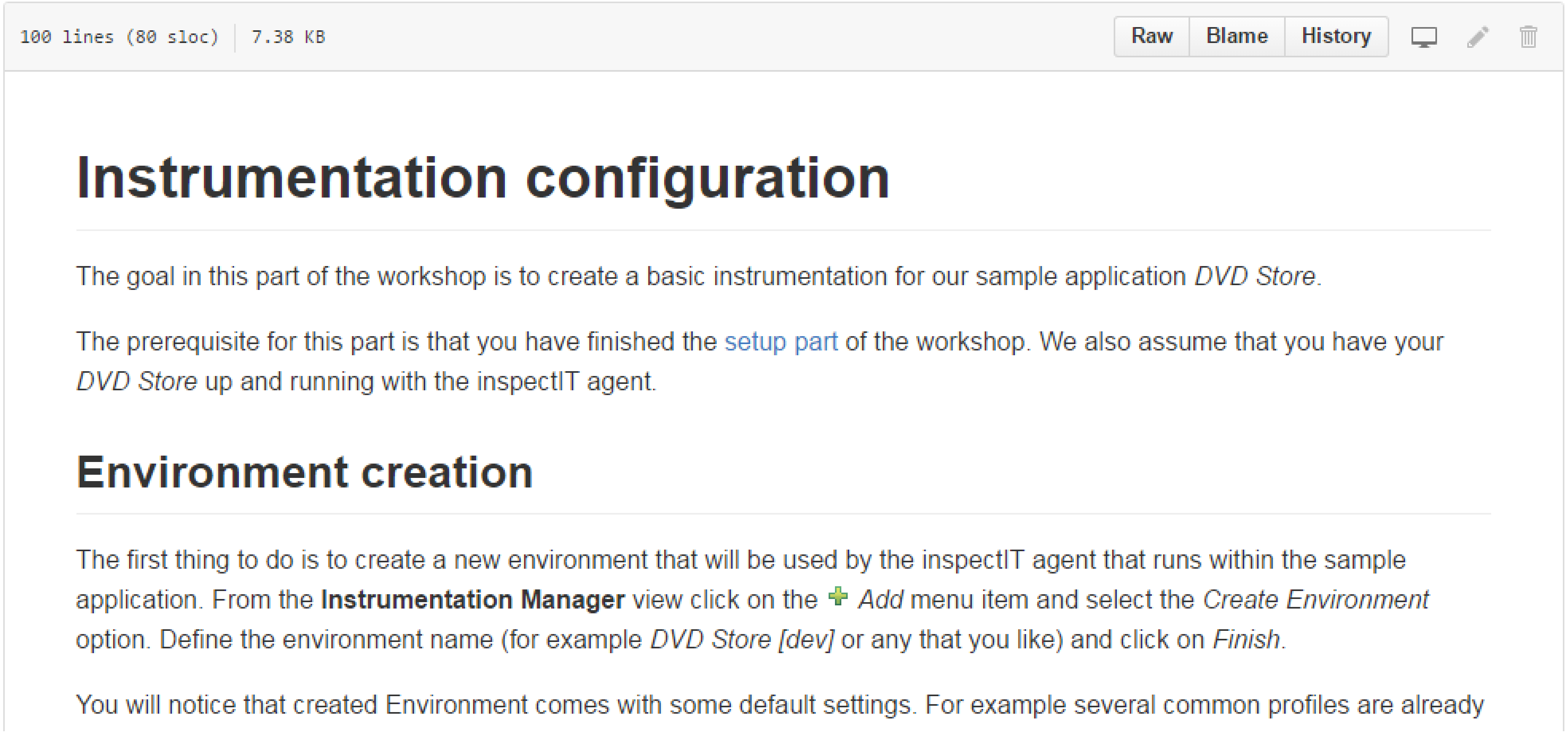
- Complete instrumentation configuration is located on the CMR and managed by the UI
- 3 basic structures:
  - **Profile**
    - defines on which classes/methods should specific sensors be applied
    - defines exclude rules
  - **Environment**
    - holds basic settings for the agent (sending strategy, sensor options, etc.)
    - defines which profiles will be used in the given environment
  - **Mappings**
    - maps single (or multiple) agents to one environment



Tool Time

Practice Time

https://github.com/inspectit-labs/workshop/blob/master/INSTRUMENTATION.md



Instrumentation configuration

The goal in this part of the workshop is to create a basic instrumentation for our sample application *DVD Store*.

The prerequisite for this part is that you have finished the setup part of the workshop. We also assume that you have your *DVD Store* up and running with the inspectIT agent.

Environment creation

The first thing to do is to create a new environment that will be used by the inspectIT agent that runs within the sample application. From the **Instrumentation Manager** view click on the ✚ *Add* menu item and select the *Create Environment* option. Define the environment name (for example *DVD Store [dev]* or any that you like) and click on *Finish*.

You will notice that created Environment comes with some default settings. For example several common profiles are already

PERFORMANCE ANALYSIS

Performance Analysis

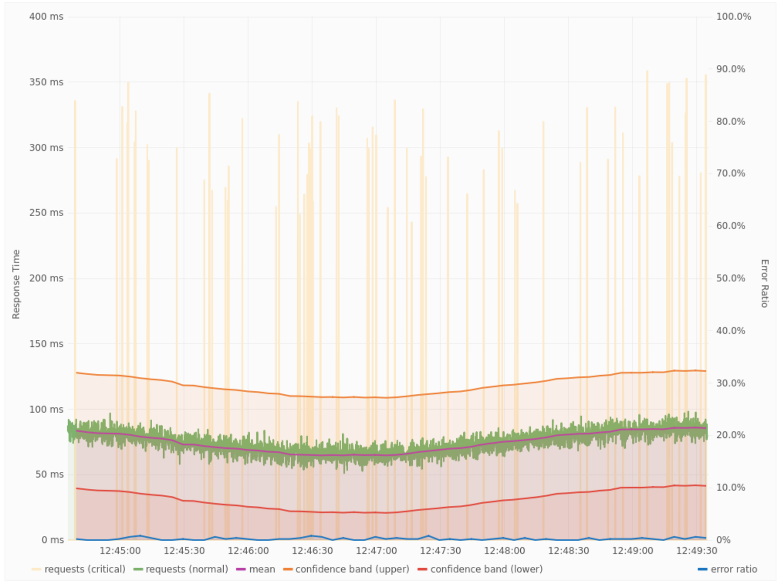
What are the goals of the performance analysis?

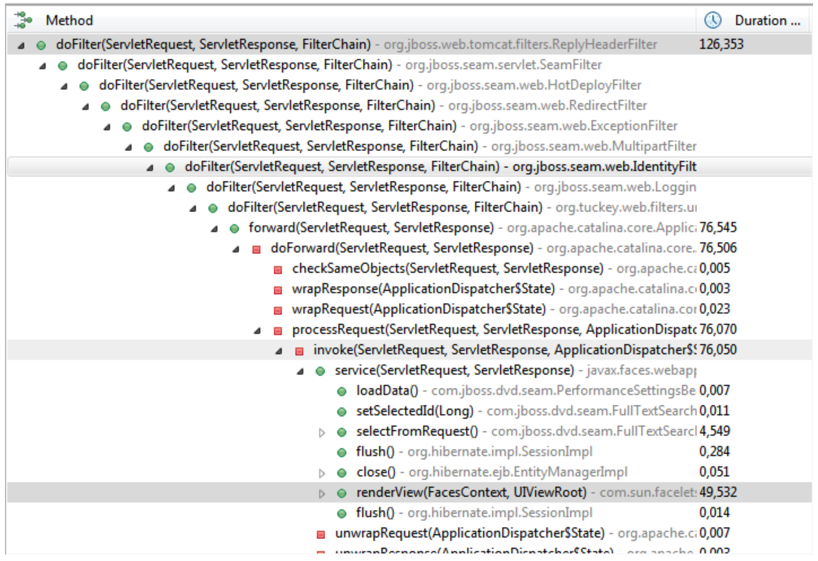
Monitoring is about getting concrete Numbers continuously

- Understand application behavior
  (where and when is our application slow)
- Detect anomalies deviating from the baseline

Diagnosing is about finding the root cause when it is slow

- Find the root cause(s) of the performance problem(s)
- Propose changes (optional)



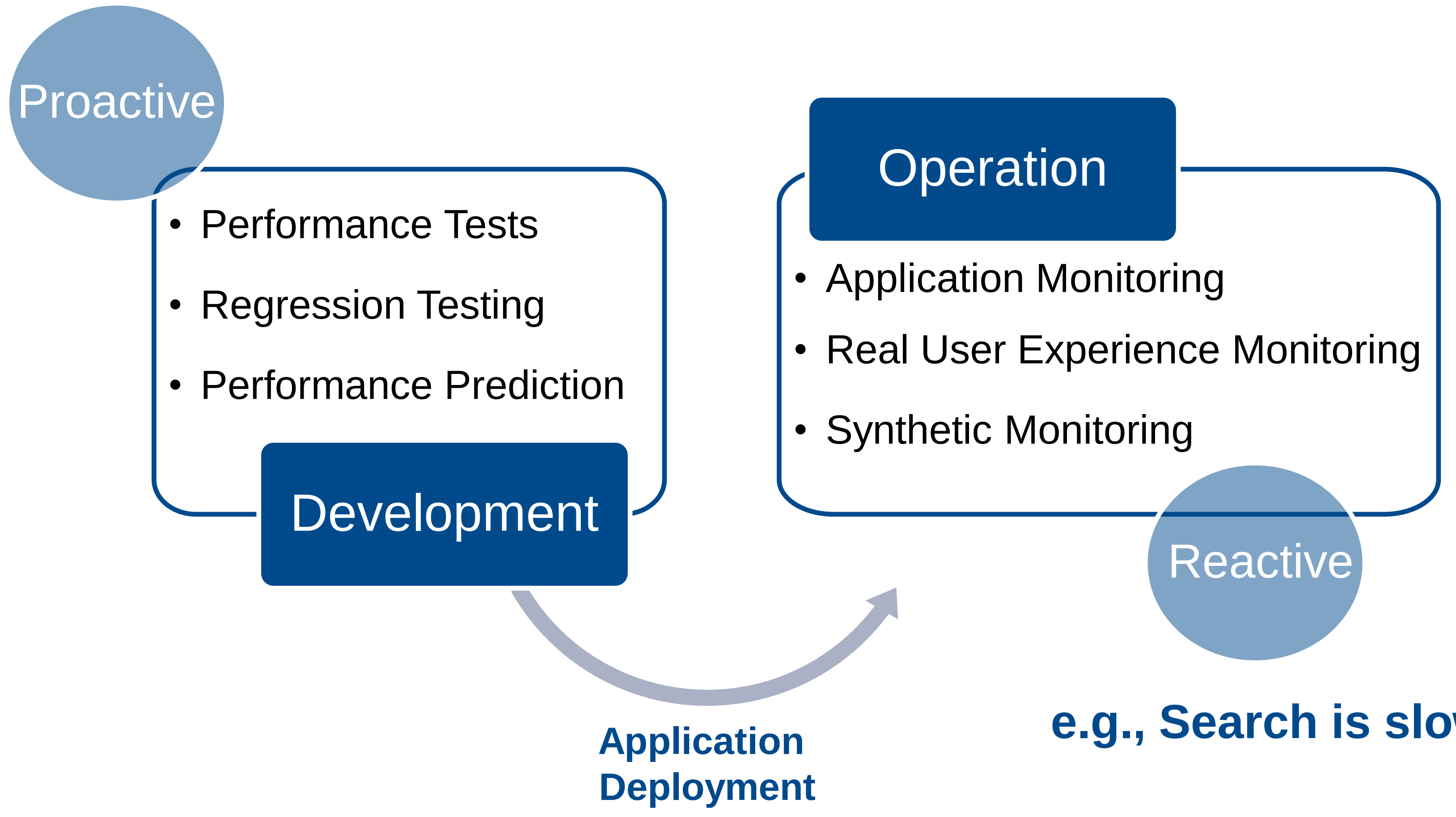


Method	Duration ...
◢ ● doFilter(ServletRequest, ServletResponse, FilterChain) - org.jboss.web.tomcat.filters.ReplyHeaderFilter	126,353
◢ ● doFilter(ServletRequest, ServletResponse, FilterChain) - org.jboss.seam.servlet.SeamFilter	
◢ ● doFilter(ServletRequest, ServletResponse, FilterChain) - org.jboss.seam.web.HotDeployFilter	
◢ ● doFilter(ServletRequest, ServletResponse, FilterChain) - org.jboss.seam.web.RedirectFilter	
◢ ● doFilter(ServletRequest, ServletResponse, FilterChain) - org.jboss.seam.web.ExceptionFilter	
◢ ● doFilter(ServletRequest, ServletResponse, FilterChain) - org.jboss.seam.web.MultipartFilter	
◢ ● doFilter(ServletRequest, ServletResponse, FilterChain) - org.jboss.seam.web.IdentityFilt	
◢ ● doFilter(ServletRequest, ServletResponse, FilterChain) - org.jboss.seam.web.Loggin	
◢ ● doFilter(ServletRequest, ServletResponse, FilterChain) - org.tuckey.web.filters.ur	
◢ ● forward(ServletRequest, ServletResponse) - org.apache.catalina.core.Applic	76,545
◢ 🟥 doForward(ServletRequest, ServletResponse) - org.apache.catalina.core.	76,506
🟥 checkSameObjects(ServletRequest, ServletResponse) - org.apache.c	0,005
🟥 wrapResponse(ApplicationDispatcher$State) - org.apache.catalina.c	0,003
🟥 wrapRequest(ApplicationDispatcher$State) - org.apache.catalina.co	0,023
◢ 🟥 processRequest(ServletRequest, ServletResponse, ApplicationDispatc	76,070
◢ 🟥 invoke(ServletRequest, ServletResponse, ApplicationDispatcher$	76,050
◢ ● service(ServletRequest, ServletResponse) - javax.faces.webap	
● loadData() - com.jboss.dvd.seam.PerformanceSettingsBe	0,007
● setSelectedId(Long) - com.jboss.dvd.seam.FullTextSearch	0,011
▷ ● selectFromRequest() - com.jboss.dvd.seam.FullTextSearc	4,549
● flush() - org.hibernate.impl.SessionImpl	0,284
▷ ● close() - org.hibernate.ejb.EntityManagerImpl	0,051
▷ ● renderView(FacesContext, UIViewRoot) - com.sun.facelet	49,532
● flush() - org.hibernate.impl.SessionImpl	0,014
🟥 unwrapRequest(ApplicationDispatcher$State) - org.apache.c	0,007
🟥 unwrapResponse(ApplicationDispatcher$State) - org.apache	0,003

NOVATEC

Reactive vs Proactive Performance Problem Analysis

e.g., Is search slow?



- Performance Tests
- Regression Testing
- Performance Prediction

- Application Monitoring
- Real User Experience Monitoring
- Synthetic Monitoring

e.g., Search is slow!

Performance Data

Proactive

- Load test
  - Provides most accurate timing data
    - simulate numerous virtual users running numerous use cases
    - can be executed on the production-like environment
  - Sometimes time consuming to create and maintain
- Alternatively: one-user testing
  - Does not provide accurate timing as there is only one user using the application
  - Can show some obvious performance bottlenecks
  - Can help in reaching the optimal instrumentation configuration
  - No time needed for setup

Reactive

- Production monitoring data
  - Provides real data of the application
  - Real load scenario
  - Real user interactions
  - Real user experience

Tool Time

INSPECTIT

Timer Data

Duration vs Exclusive Duration

Case A

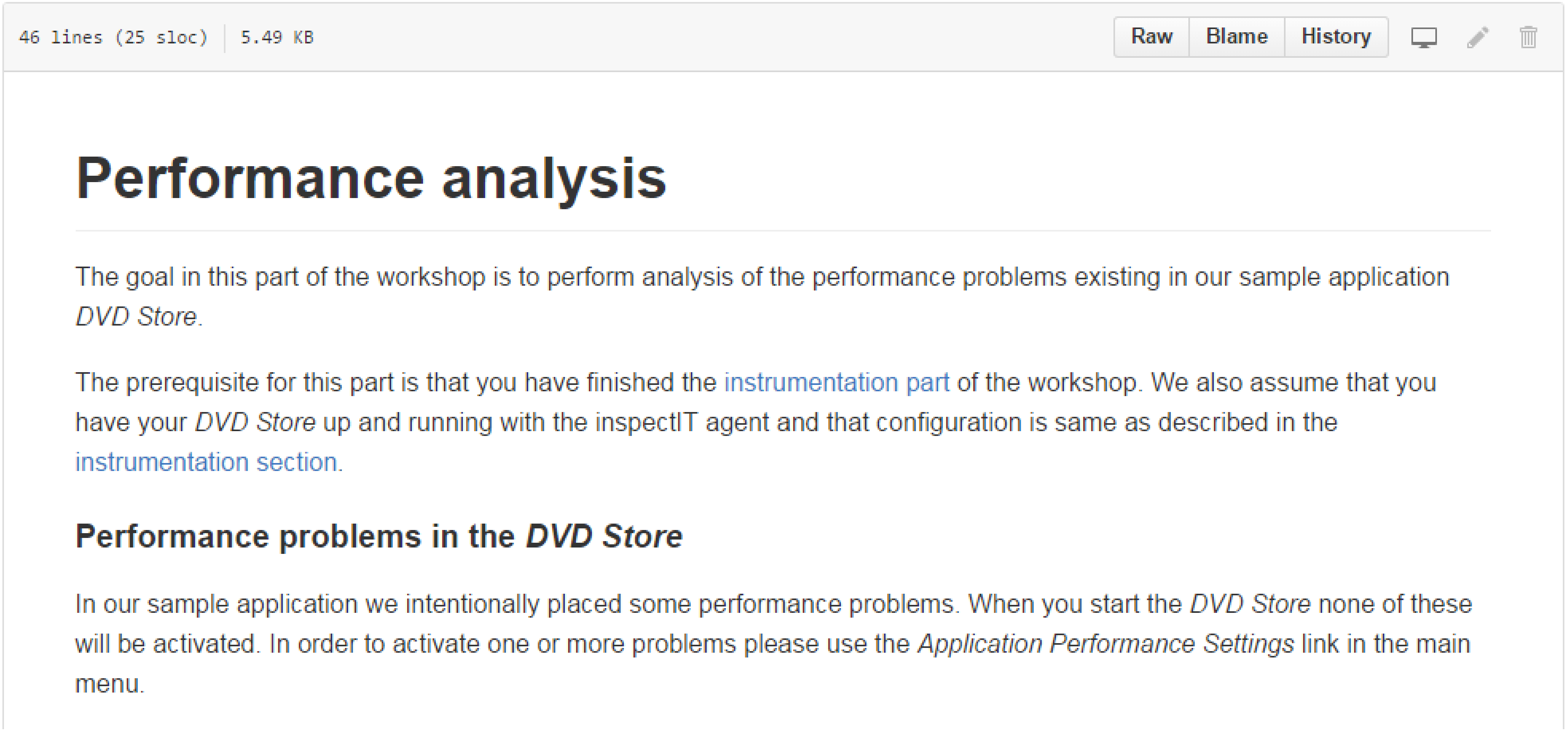
Method	Instrumented	Duration	Exclusive duration
foo()	Yes	**100 ms**	**70ms**
bar()	No	30 ms	30 ms
baz()	Yes	30 ms	30 ms

Case B

Method	Instrumented	Duration	Exclusive duration
foo()	Yes	**100 ms**	**40ms**
bar()	Yes	30 ms	30 ms
baz()	Yes	30 ms	30 ms

Practice Time

https://github.com/inspectit-labs/workshop/blob/master/ANALYSIS.md



Performance analysis

The goal in this part of the workshop is to perform analysis of the performance problems existing in our sample application *DVD Store*.

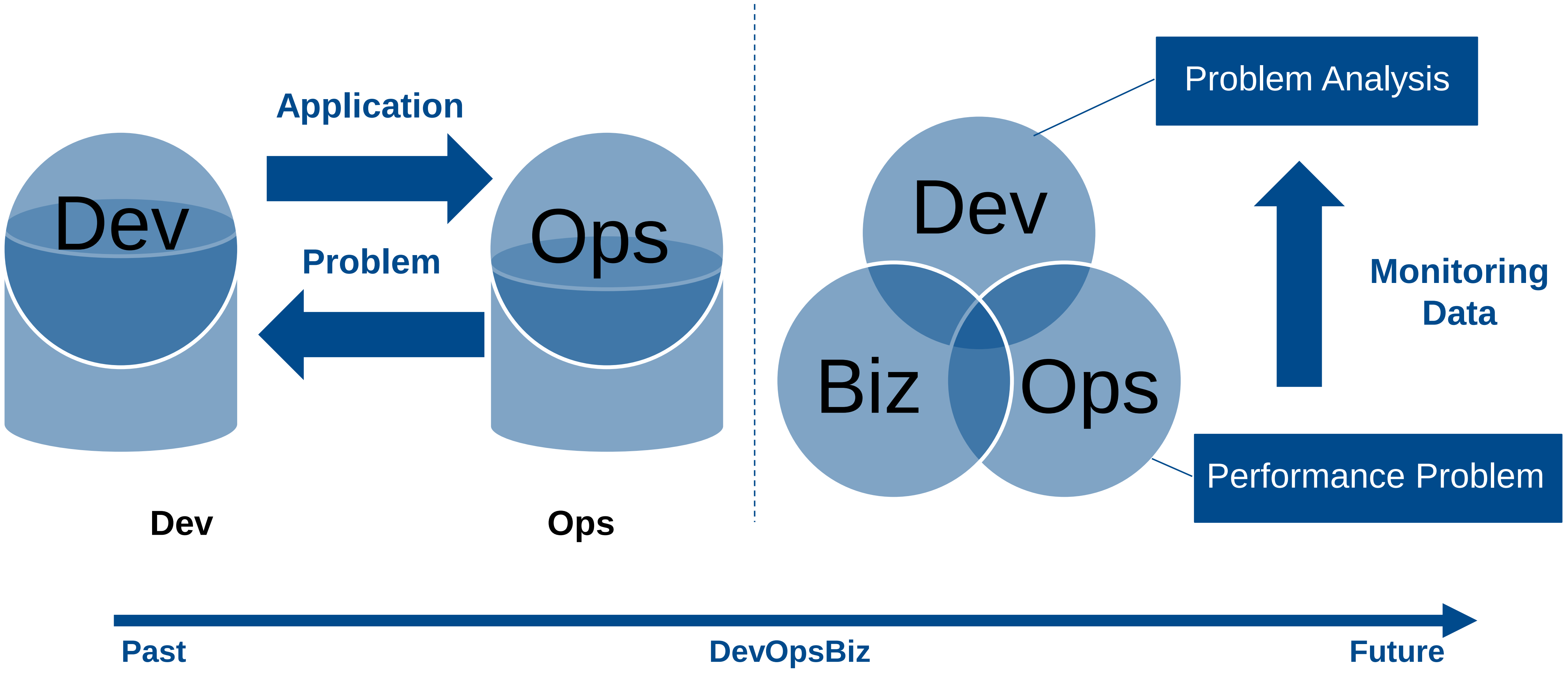
The prerequisite for this part is that you have finished the instrumentation part of the workshop. We also assume that you have your *DVD Store* up and running with the inspectIT agent and that configuration is same as described in the instrumentation section.

Performance problems in the *DVD Store*

In our sample application we intentionally placed some performance problems. When you start the *DVD Store* none of these will be activated. In order to activate one or more problems please use the *Application Performance Settings* link in the main menu.

COLLABORATION

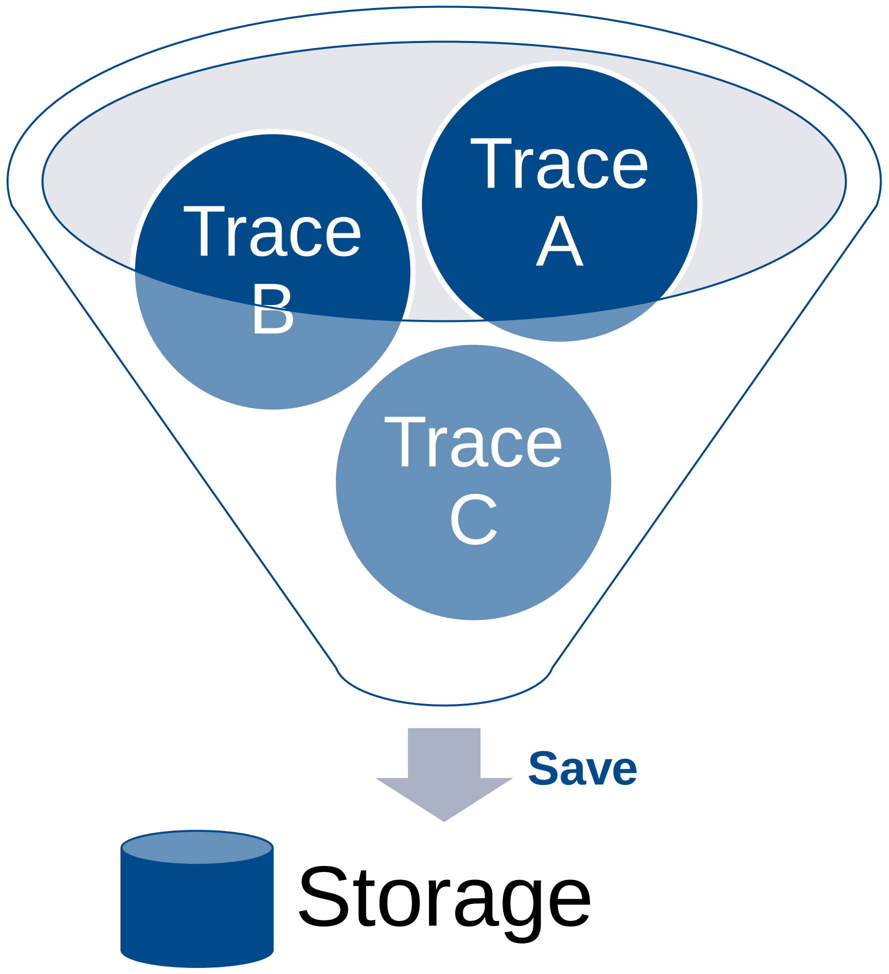
Why is Collaboration important?



Storages

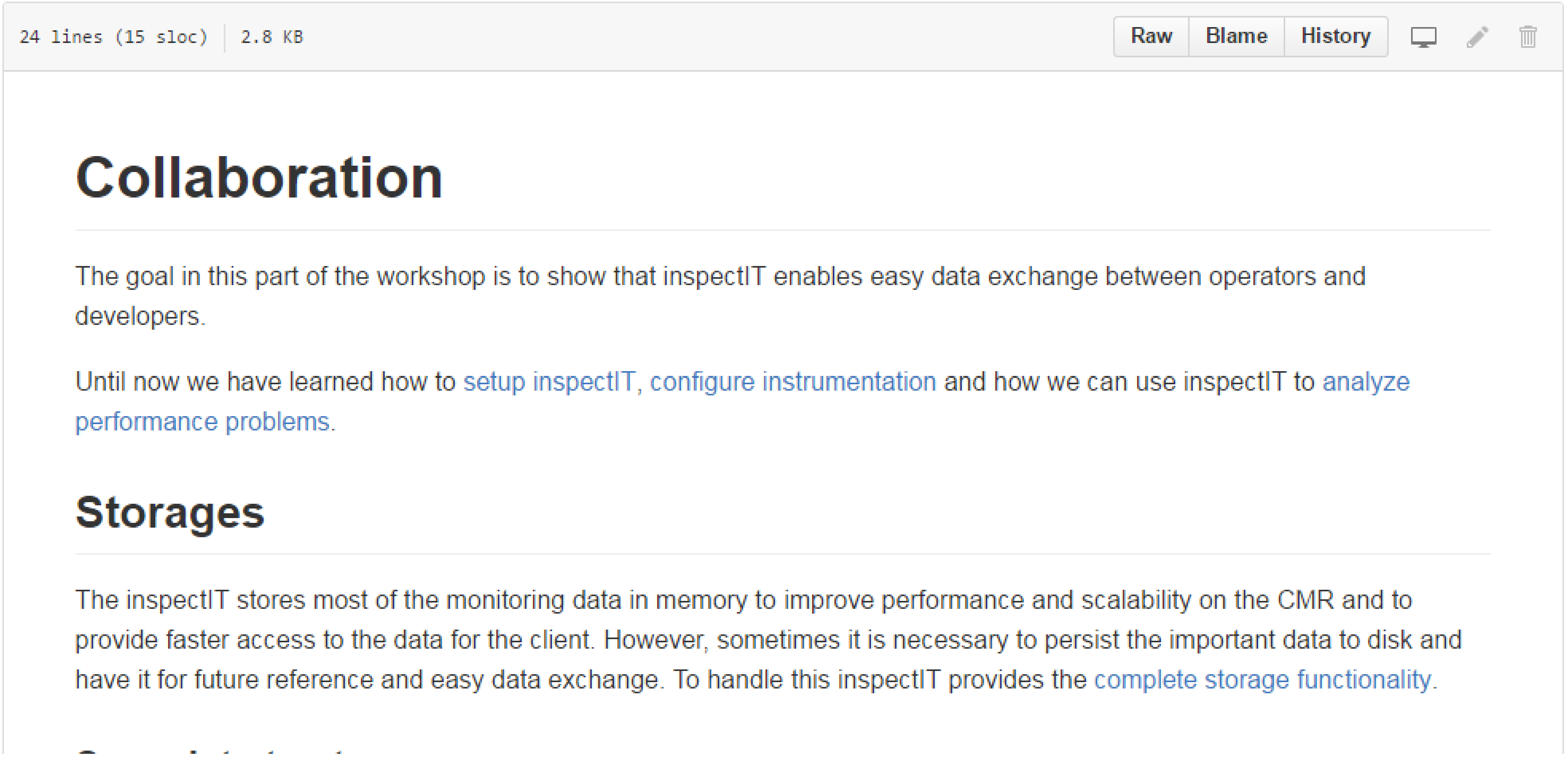
Export Data from the CMR to local Storage

- Any collected data
  - Traces
  - Timer
  - Exceptions
  - etc.
- Storages can be stored locally
- Offline analysis of data
- Export/Import functionality in for data exchange



Practice Time

https://github.com/inspectit-labs/workshop/blob/master/COLLABORATION.md



Collaboration

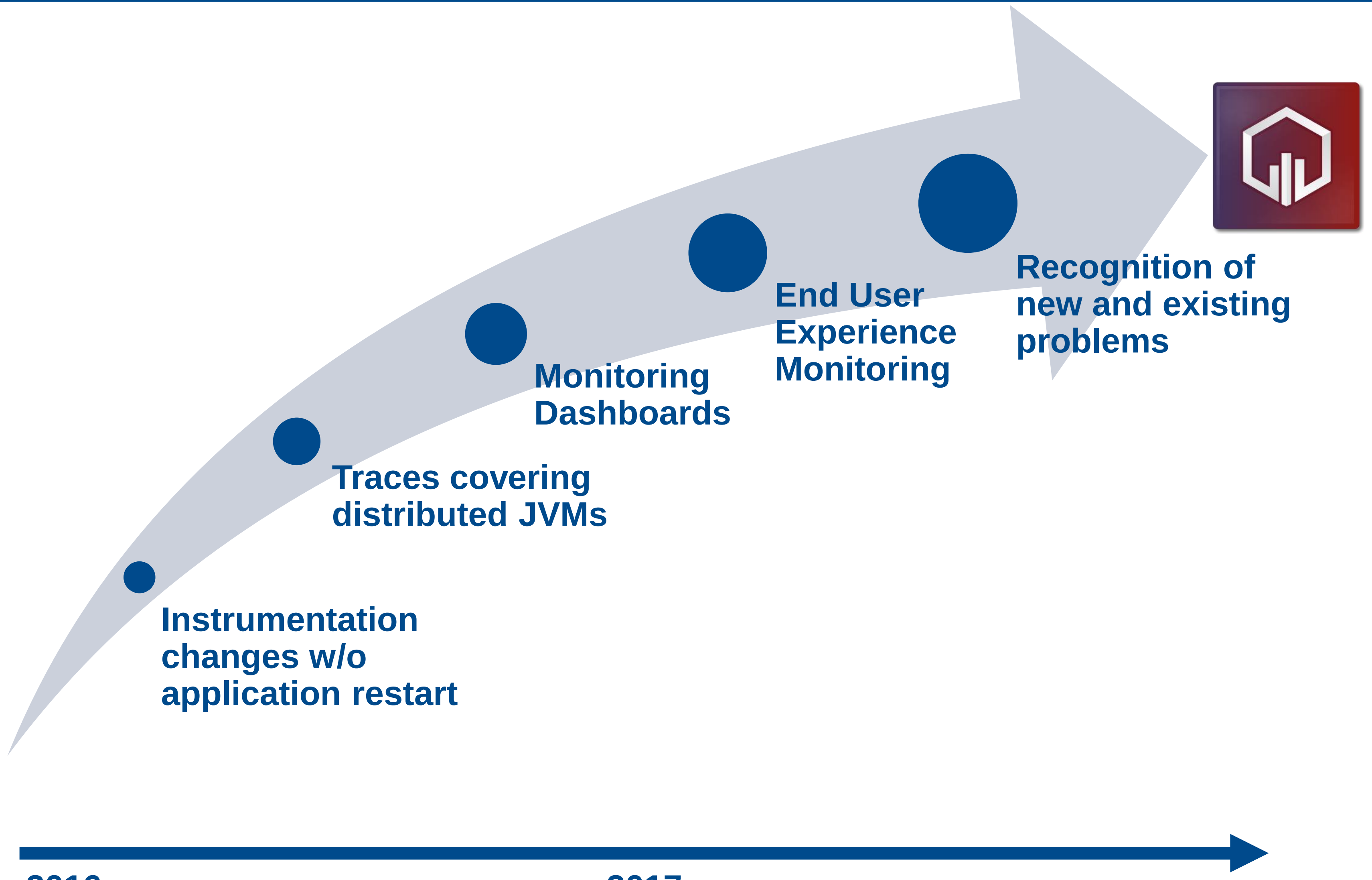
The goal in this part of the workshop is to show that inspectIT enables easy data exchange between operators and developers.

Until now we have learned how to setup inspectIT, configure instrumentation and how we can use inspectIT to analyze performance problems.

Storages

The inspectIT stores most of the monitoring data in memory to improve performance and scalability on the CMR and to provide faster access to the data for the client. However, sometimes it is necessary to persist the important data to disk and have it for future reference and easy data exchange. To handle this inspectIT provides the complete storage functionality.

What‘s coming next?


Instrumentation changes w/o application restart
Traces covering distributed JVMs
Monitoring Dashboards
End User Experience Monitoring
Recognition of new and existing problems
2016
2017

Contact

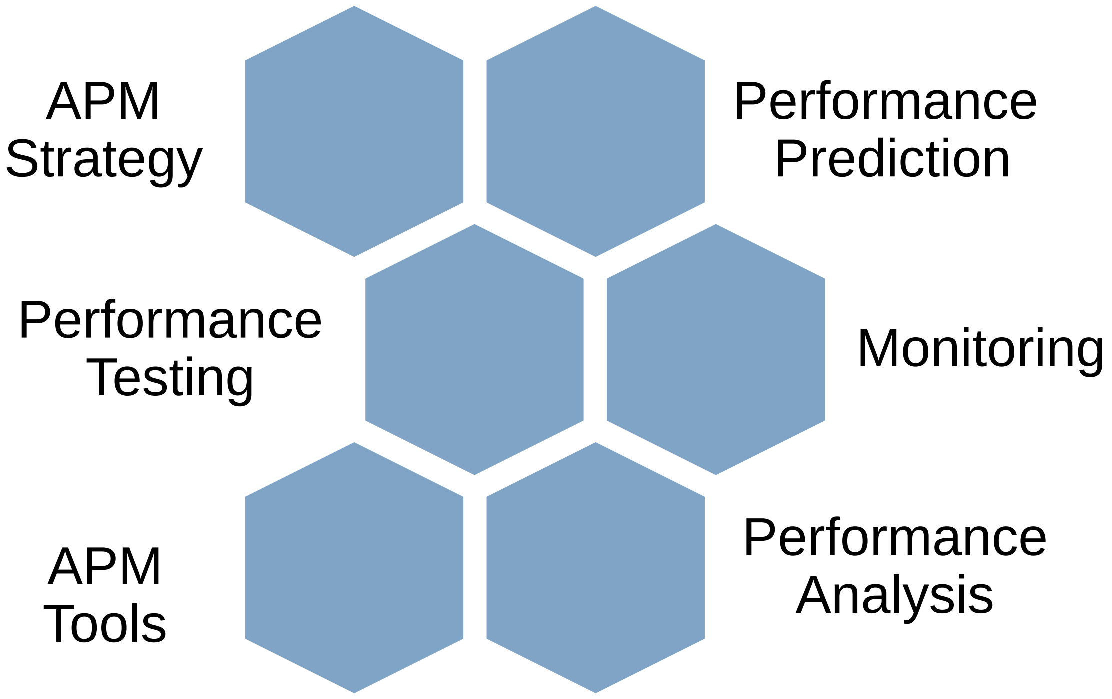
Web
http://www.inspectit.rocks

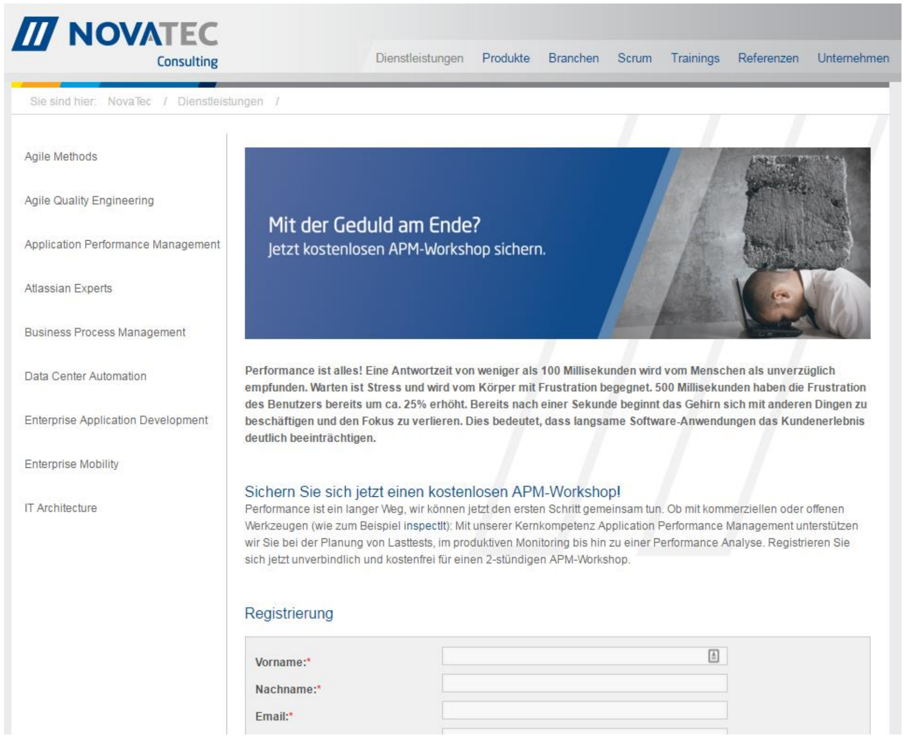
Gitter
https://gitter.im/inspectIT/chat

E-Mail
info.inspectit@novatec-gmbh.de

FREE Private APM Workshop

http://www.novatec-gmbh.de/dienstleistungen/apm-geduld/





Mit der Geduld am Ende?
Jetzt kostenlosen APM-Workshop sichern.

Performance ist alles! Eine Antwortzeit von weniger als 100 Millisekunden wird vom Menschen als unverzüglich empfunden. Warten ist Stress und wird vom Körper mit Frustration begegnet. 500 Millisekunden haben die Frustration des Benutzers bereits um ca. 25% erhöht. Bereits nach einer Sekunde beginnt das Gehirn sich mit anderen Dingen zu beschäftigen und den Fokus zu verlieren. Dies bedeutet, dass langsame Software-Anwendungen das Kundenerlebnis deutlich beeinträchtigen.

Sichern Sie sich jetzt einen kostenlosen APM-Workshop!
Performance ist ein langer Weg, wir können jetzt den ersten Schritt gemeinsam tun. Ob mit kommerziellen oder offenen Werkzeugen (wie zum Beispiel inspectIt): Mit unserer Kernkompetenz Application Performance Management unterstützen wir Sie bei der Planung von Lasttests, im produktiven Monitoring bis hin zu einer Performance Analyse. Registrieren Sie sich jetzt unverbindlich und kostenfrei für einen 2-stündigen APM-Workshop.

Registrierung

Vorname:*	
Nachname:*	
Email:*	

Take Away

[Image source: http://content.hollywire.com/sites/default/files/Success-Kid.jpg]