Cloud Pak for Watson AIOps Sample Demo Script for the live demo environment.

This script is intended as a guide to demonstrate Cloud Pak for Watson AIOps using the live demo environment, running the Cloud Pak itself and the demo application. The script is presented in a few sections. You can utilize some or all sections depending upon your client’s needs.

The script is intended to be used with live Cloud Pak for Watson AIOps 3.x demo environment that you can reserve via [TechZone](https://techzone.ibm.com/collection/cloud-pak-for-watson-aiops-shared-demo-environments) or [install by yourself](https://github.com/niklaushirt/cp4waiops-deployer) very easily

In the demo script,

* “**Action**” denotes a setup step for the presenter.
* “**Narration**” denotes what the presenter will say.
* “**Note**” denotes where the presenter may need to deviate from this demo script or add supplemental comments.

## Key Terminology

You should be familiar with the following terminology when discussing Cloud Pak for Watson AIOps:

* **Application**: IBM Cloud Pak for Watson AIOps brings together the capability to group resources from different data types into applications. Clients can flexibly define an application to meet their business needs. With applications, you can obtain an integrated view of resources to understand inter-dependencies.
* E**vent**: A point-in-time statement in Cloud Pak for Watson AIOps that tells us that something happened somewhere in a client’s environment. It tells us what happened, where it happened, and when it happened. An event does not have to be exceptional or actionable, it can simply tell us something has happened.
* **Alert**: An alert in Cloud Pak for Watson AIOps represents an abnormal condition somewhere in an environment that requires resolution. It tells us what is happening, where it is happening, and when it started to happen. It may be informed by one or more events. It has a start time and end time.
* **Story**: A story in Cloud Pak for Watson AIOps represents an outage or reduction in service which is currently impacting customers and requires rapid remediation. It is created based on one or more trigger alerts that indicate the outage or reduction in service. Any alert of severity Major or Critical will act as a trigger alert. Other alerts that share the same cause may add context to the story.
* **Incident**: An incident in ServiceNow is an event of interruption disruption or degradation in normal service operation. An open incident in ServiceNow implies that the customer is impacted, or it represents the business risk.
* **Topology**: A topology is a representation of how constituent parts are interrelated. In Cloud Pak for Watson AIOps, an algorithm analyzes how the event nodes are proximate to each other and groups them into a topology-based correlation.

## Demonstration scenario

Overview

This use case shows clients how IBM Cloud Pak for Watson AIOps proactively helps avoid application downtimes and incidents impacting end-users. You play the role of an SRE/Operations person who has received a Slack message indicating that the RobotShop application is not displaying customer ratings. This is an important feature of the RobotShop application since RobotShop is the main platform from which the fictional company sells its robots.

The use case demonstrates how Cloud Pak for Watson AIOps can assist the SRE/Operations team as they identify, verify, and ultimately correct the issue. The demonstration shows integration with Instana, Turbonomic, ServiceNow, and Slack. Slack is the ChatOps environment used for working on this incident.

You will demonstrate the following major selling points around Cloud Pak for Watson AIOps:

1. Pulls data from various IT platforms: IBM Cloud Pak for Watson AIOps monitors incoming data feeds including logs, metrics, alerts, topologies, and tickets, highlighting potential problems across incoming data, based on trained machine learning models.
2. Utilizes AI and natural language processing: An insight layer connects the dots between structured and unstructured data, using AI and natural language processing technologies. This allows you to quickly understand the nature of the incident.
3. Provides trust and transparency: Using accurate and trustworthy recommendations, you can move forward with the diagnosis of IT system problems and the identification and prioritization of the best resolution path.
4. Resolves rapidly: Time and money are saved from out-of-the-box productivity that enables automation and utilizes pre-trained models. A “similar issue feature” from past incidents allows you to get services back online for customers and end-users.

## Demonstration flow

1. Scenario introduction
2. Trigger problem situation [In the background]
3. Verify the status of the Robot Shop application.
4. [Optional] Work with the ChatOps story. Explain the AI model for detecting log anomalies:
   1. View events
   2. View log anomalies
   3. Add findings to the story
   4. Update status by notifying users and adding notes
5. In Cloud Pak for Watson AIOps:
   1. View Application resource topology
   2. View Events
   3. Explain correlation methods
6. [Optional] In ServiceNow:
   1. View similar incidents
   2. Create a new incident
   3. View the Watson AIOps Events associated with the Story
   4. Explore similar incidents from the past
7. [Optional] In Cloud Pak for Watson AIOps:
   1. Look at the integrations with other products:
      1. Github
      2. Metric Anomaly (Metric Manager in CP4AIOps)
      3. Turbonomic
      4. Instana
8. In Cloud Pak for Watson AIOps:
   1. Correct the issue with Runbook
   2. Update issue through Slack interface
   3. Verify Runbook execution and functionality of Robot Shop
9. Summary

# Introduction

**Narration**

Welcome to this demonstration of the Cloud Pak for Watson AIOps platform. In this demo, I am going to show you how Watson AIOps can help your operations team proactively identify, diagnose, and resolve incidents across mission-critical workloads.

You’ll see how:

* Watson AIOps intelligently correlates multiple disparate sources of information such as logs, metrics, events, tickets and topology
* All of this information is condensed and presented in actionable alerts instead of large quantities of unrelated alerts
* You can resolve a problem within seconds to minutes of being notified using Watson AIOps’ automation capabilities

During demonstration, we will be using the sample application called RobotShop, which serves as a proxy for any type of app. The application is built on a microservices architecture, and the services are running on Kubernetes cluster.

**Action**

## Use demo introductory PowerPoint presentation, to illustrate the narration

**Narration**

Let’ look at the environment that we have set up. Our sample application: “RobotShop” is running as a set of microservices in a Kubernetes cluster (slide 2). Typically, the Operations team maintaining such application has a collection of tools through which they collect various data types. Here we have Instana for metrics and tracing (slide 3), Prometheus to gather metrics and fire alerts and ElasticSearch to gather logs (slide 5). These can be replaced by any tools depending on the user environment.

All these data types like: traces, metrics, logs, events are brought into Watson AIOps. This industry-leading AIOps platform can detect anomalies in metrics and logs (slides 6-8) to proactively spot issues and is doing topology-based entity-linking (slides 9) to create a story (slides 10), to provide operators a detailed context to a problem.

Watson AIOps presents all of that to the user through ChatOps interface (Slack in this demo) or through a dedicated Web user interface (slides 11).

**Note**

We're using Slack in this demo, but Watson AIOps also integrates with Microsoft Teams, so depending on which tools is used in the organization, notifications will get delivered that way.

**Narration**

Finally, the Watson AIOps platform integrates Runbook automation to advise users on available fix procedures to the problem and let them automate execution (slides 12-13).

**Note**

The following step does not have to be shown to the client – you may perform the action in the background if possible.

**Action**

## Open AIManager **Demo** UI, and trigger error situation

Point your browser to the AIManager Demo UI, login with the token “P4ssw0rd!” and trigger the error situation you would like to use in your demo. This action injects the stream of simulated events into the system, which replicates what could happen in a real life situation.

**Graphical user interface, website

Description automatically generated**

**Action**

## Open the RobotShop application user interface (link can be found under Support on the Welcome screen). Play with the application UI.

**Narration**

In this demo I am the application SRE (Site Reliability Engineer) responsible for an e-commerce website called RobotShop, an online store operated by my company. In the middle of the day (when clients make most of the purchases) I received a slack message on my mobile, alerting me that there is some problem with the site.

Let’s verify what’s going on with the RobotShop site. The application is up but displays an error that it cannot get any ratings

**Action**

## Open any robot details to demonstrate there is no ratings displayed.

**Narration**

I know that there are many ratings for each of the products that we sell, so when none are displayed, it means that there is a likely problem with Ratings service withing application that may heavily impact client’s purchasing decisions, as well as may be a sign of a wider outage.

# [Optional] ChatOps and AI model explanation

**Note:** Use this part of the demo script if your client is interested in ChatOps, or you want to explain how the log anomaly detection works. If not, you can skip to the next section on page 14.

**Action**

## Open Slack interface (Login with the credentials displayed in the AIManager Demo UI app)

Graphical user interface, website

Description automatically generated

**Note:** The **“Slack”** button in the Demo UI should open the Slack interface in the browser. To use Slack in the browser, select **“open this link in the browser”** option as shown below.

Graphical user interface, text, application

Description automatically generated

To login with the provided password, please select **“sign in with the password”** link.

Graphical user interface, application

Description automatically generated

If you have troubles accessing Slack with the provided credentials, please contact [dymaczewski@pl.ibm.com](mailto:dymaczewski@pl.ibm.com) to have your own Slack user added to the proper workspace ([IBM Cloud Pak for Watson AIOps Demos](https://ibmcloudpakfo-thf7559.slack.com/))

**Narration**

The Slack message displays the story generated by Cloud Pak for Watson AIOps. The content includes related log anomalies, topology and blast radius, similar incidents, recommend actions based on past trouble tickets, relevant events, runbooks, and more.

**Action**

## At the bottom of the message, click **Show more**. *Result: More information is shown*

Graphical user interface, text, application

Description automatically generated

**Narration**

You can see probable cause of the incident, all the alerts that have been grouped under this story, recommended actions and runbooks that have been collected from the similar incidents in the past. ChatOps interface is a virtual “war room” where different SREs, different teammates can see the information and work on the resolution in a collaborative way. Examining all the presented data, I can see that Watson AIOps suggests that the incident was started with a Git commit that changed the memory limits for MySQL database.

**Action**

## Click **View alerts**. *Result: The view alerts window is shown.*

Graphical user interface, text, application, Teams

Description automatically generated

**Narration**

Many SRE teams use Slack or Microsoft Teams as their main communication tool. Integration with Slack (in this case) helps me to collaborate with other teammates on quickly resolving the issue and share findings without switching between tools and screens. This is an implementation of the principle called ChatOps.

One thing that I can do now, is to look at Alerts that were correlated into the Story by Cloud Pak for Watson AIOps.

**Action**

## Explore available **alerts**. At the end click **Filter** field at the top and select ‘**Log anomalies**’. *Result: The view shows just detected log anomalies.*

Graphical user interface, application

Description automatically generated

**Note**

You must adjust narration depending on the alerts created in the environment you are using (Be aware that demo content may change over time)

**Narration**

As you can see, there are many alerts coming from the different sources. You can see events coming from Git, Instana, Turbonomic, etc. Cloud Pak for Watson AIOps provides our-of-the-box large number of integrations with popular tools and almost any tool can be integrated using generic webhooks. We will explore those alerts a bit later, let’s first look if there are any detected log anomalies.

**Action**

## Check, if there are any Log anomaly alerts shown. If there are, click **‘Show more’**. *Result: The view shows just detected log anomaly alerts.*

Graphical user interface, text, application

Description automatically generated

**Note**

You must adjust narration depending on the alerts created in the environment you are using (Be aware that demo content may change over time). If there are any **Log anomaly alerts**, follow the script below. If there are no, skip to the next section.

**Narration**

As you can see, the system has detected log anomaly for the ‘ratings’ component of RobotShop application. Very often log anomalies can be detected prior to actual outage, that’s why they are so valuable in proactive avoiding of the incidents. Let’s look at the findings of the AI model to understand what the log anomaly looks like. Let’s explore the alerts shown.

**Action**

## Scroll down again until you see Pattern 1, Pattern 2, and Pattern 3. *Result: Patterns are displayed.*

Graphical user interface, text, application, email

Description automatically generated Graphical user interface, application

Description automatically generated

**Narration**

Let’s look at the detected log anomaly. As SRE, I don’t have to manually parse the application logs or define any error patterns. The AI model is trained to learn what the regular logs look like and is then able to detect in the flowing log stream any significant deviations from the normal behavior.

**Pattern 1** indicates that the AI model detected 3 entries that does not match any known log pattern, but these entries were not errors. Log entry that was not qualified as an error but was not expected may suggest an issue as this is outside of normal behavior.

The same is true for **Pattern 2** which indicates the system received some unexpected error message. This adds more pieces of evidence to my case.

And finally, **Pattern 3** indicates that some usual log messages were expected but received none.

**Action**

## Click **Log preview** below Pattern 2 section. *Result: Actual log entry that triggered alert is shown.*

**Graphical user interface, application

Description automatically generated Graphical user interface, text, application, email

Description automatically generated**

**Narration**

As you can see, out of hundreds and thousands of log entries, AI engine has automatically picked the relevant one, showing the database connection error. This provides the final evidence to indicate that database connectivity is an issue.

**Action**

## Close **Log preview**. Scroll up and click **Attach log message patterns**. *Result: Message displays that logs were successfully attached.*

Graphical user interface, text, application, email

Description automatically generated

**Narration**

To streamline work on the issue I can immediately share my findings with other team members, attaching the relevant information to the conversation in the Slack channel. This way, other team members won’t waste time if they would decide to also work on this incident.

**Action**

## Close the message window. Scroll down to the bottom of the Slack message. *Result: The message scrolls and a new set of buttons are available.*

Graphical user interface, text, application

Description automatically generated

## Click **Set to in-progress**. *Result: New window is displayed*.

Graphical user interface, text, application

Description automatically generated

## Fill in the form. Click in the box below **Notes**. Provide a sample message to co-workers.

Graphical user interface, text, application

Description automatically generated

## Click **Submit**.

**Narration:**

As an SRE, I should document my work on the issue. I can share any findings with teammates, by attaching them to the story. To let others know I work on the problem, I can set the story status to “in progress”. I can also add some notes.

# Analyzing other information to confirm the root cause

**Action**

## In the story, click **RobotShop**. *Result: The Watson AIOps web interface opens showing the application topology.*

Graphical user interface, text, application, email

Description automatically generated A picture containing graphical user interface

Description automatically generated

**Narration**

Let’s take a closer look at the timeline and resources for the story by viewing the resource topology.

The interface shows the topology of the application. IBM Cloud Pak for Watson AIOps’ topology service delivers a working understanding of what you have in your environment, how the resources relate to each other, and how the environment has changed over time.

You can see that there are some statuses attached to the different resources, marked with colorful dots. Let’s view the details and status of the item with red status. You can see it’s the web component (front end) for our application.

**Action**

## Hover cursor over icons with the red status icon. Find the one which displays resource name “web”. Then, right-click the resource with the red icon again. Select **Resource details.** *Result: Detailed view displays.*

Chart, scatter chart

Description automatically generated Chart, scatter chart

Description automatically generated

**Narration**

The topology service provides operations teams with complete up-to-date visibility over dynamic infrastructure, resources, and services. The topology service lets you query a specific resource for details, and other relevant information.

**Action**

## Click the twisty on the story summary (upper left). Click **View all alerts.** *Result: Story details are displayed, then Alerts are displayed.*

Graphical user interface, text, application

Description automatically generated Graphical user interface, text, application

Description automatically generated

**Narration**

On the left, you can see a widget showing the story. This is the same story we have been viewing in Slack. In the story, you can see all the signals from different connected systems that were correlated.

**Action**

## In the First occurrence column, click the first entry. *Result: Alert details view displays.* Scroll down or close top sections to see all relevant information.

Graphical user interface, application

Description automatically generated

Application

Description automatically generated with low confidence

**Narration**

Notice, that alerts are not sorted by severity, but the AI engine ranked them by relevance. The ones that are likely related to the root cause are at the top. Let’s look at the first row for some more details. In the **Alert details,** you can see different types of groupings explaining why the specific alert was added to the story.

**Action**

## Click **Scope-based grouping**. *Result: An explanation is displayed.*

Graphical user interface, text, application, email

Description automatically generated

**Narration**

Some alerts were added to the story because they occurred on the same resource within a short period (default is 15 minutes)

**Action**

## Click **Topological grouping**. *Result: The topological grouping is displayed*.

Graphical user interface, chart, scatter chart

Description automatically generated

**Narration**

Other alerts were grouped because they occurred on the logically or physically related resources. This correlation is using the application topology service that stitches topology information from different sources.

**Action**

## Click **Temporal correlation**. *Result: Temporal correlation is displayed*.

Graphical user interface, text, application, email

Description automatically generated

## **Close** the Alert details window.

**Narration**

Finally, the temporal correlation adds to the story events that previously, in history, are known to occur close to each other in the short time window. What is most important here is the fact that all these correlations happen automatically – there is no need to define any rules or program anything. In highly dynamic and distributed cloud-native applications this is a huge advantage that saves a lot of time and effort.

# [Optional] Integration with ITSM tool (ServiceNow)

**Note**

**IMPORTANT:** In the Robot Shop demo scenario, the integration with ServiceNow is simulated with the static screenshot. If the specific demo environment you are using does not have ServiceNow connection configured, you won’t be able to show anything else than provided sample views in 5.4. Skip this section if the environment you are using for the demo does not have ServiceNow.

**Narration**

Most large organizations use IT Service Management tools to govern processes around IT. Our organization is using ServiceNow for that purpose. When an incident is detected, the Incident record is automatically opened there. It can also be helpful to review details from similar incidents in the past, to help determine a likely resolution.

**Action**

## Go back to Slack interface*.*

**Narration**

In the Slack interface, you can see that a new incident was created in ServiceNow. Let’s explore how this integration looks like.

**Action**

## Under Incident, click **Incident number**. *Result: Displays the new incident in ServiceNow.*

Graphical user interface, text

Description automatically generated

**Narration**

When the story is created in Watson AIOps upon detecting suspicious alerts, the new incident is opened in ServiceNow, so other teams (for example HelpDesk operators) can be aware of the issue.

**Action**

## Click the **Watson AIOps Events** tab. *Result: Displays seven events.*

Graphical user interface, application

Description automatically generated

**Narration**

As you can see, the ServiceNow record is synchronized with Watson AIOps, so other teams which do not have access to the SRE tools can be up-to-date with the incident resolution status.

Integration with ServiceNow is not limited to opening Incidents. The IBM Cloud Pak for Watson AIOps aggregates information about similar messages, anomalies, and events for a component or application. It can also extract the steps used to fix previous incidents (if documented) and recommend resolutions using natural language processing. This AI model helps you discover historical incidents to aid in the remediation of current problems.

For a given problem, your team is presented with the top-ranked similar incidents from the past. These relevant similar incidents lead to fast incident resolution. Without these features, your team must manually search for past incidents and resolutions, which is time-consuming.

**Note**

The following action leads to the ServiceNow interface static screenshot.

**Action**

## Click any of therecommended actions. *Result: ServiceNow view opens with a similar incident from the past.*

Graphical user interface, text, application, email

Description automatically generated

**Narration**

It seems that there were a similar incident in the past. Let’s open the ServiceNow view with the closed incident.

**Action**

## Explore the information of the past incident in the ServiceNow UI.

Graphical user interface, application

Description automatically generated

**Narration**

It seems that previously the MySQL deployment run out-of-memory and the situation was fixed with the runbook.

**Action**

## Return to AI Manager browser tab.

**Narration**

I can see that the previous incident was related to change. Perhaps the developers again reduced the footprint too much and this is causing the issue.

# [Optional] Exploring other integrations

**Action**

## Navigate to Alerts list.

**Narration**

Let’s look at other alerts that were added to the story. The Sender column shows us the source for the Alerts, which could be Instana, Turbonomic, Humio, and so forth. We will start by exploring from the bottom of the list and explore the genesis of the issue. We can see that this story started gathering alerts at 11:37.

**Action**

## Sort list by **First occurrence**. Click the Github entry. *Result: Alert details view opens showing alert properties.*

Graphical user interface, application

Description automatically generated

**Narration**

Let’s look at Github alert first. It looks as if the development team has reduced the available memory for the SQL database.

**Action**

## Close the dialog. *Result: Return to Alerts window.*

## Explore other alerts as appropriate and relevant to your client environment.

**Narration**

Other events are confirming the hypothesis. We can see that the Security tool in the DevOps toolchain has detected the memory size change. Instana has also detected the change (see the message). The Selenium test has failed.

**Note**

Adjust the narrations to the list of displayed alerts.

You can also see the alert from detected Metric Anomaly related to MySQL peak memory usage.

**Action**

## If available, click the **line for Metric Anomaly**, then open Metric anomaly details.*Result: Metric anomaly view opens.*

Graphical user interface, application

Description automatically generated

**Narration**

Cloud Pak for Watson AIOps is capable of collecting raw metrics from multiple sources and detecting metric anomalies. It was trained on hundreds or thousands of metrics from the environment and constructs a dynamic baseline (shown in green). The graphic suddenly turns red which relates to detected anomaly when the database is consuming a higher amount of memory than usual.

**Action**

## Close the Metric anomaly details view. Highlight the line for Turbonomic.

Graphical user interface, application, table

Description automatically generated

**Narration**

You can also see the event coming from Turbonomic. Based on the observed usage there is a resize-up action generated by the system for the MySQL container.

Alerts coming from Instana indicate there is a problem with the RobotShop homepage. We can dive into the incidents related to the problem. Look at the triggering event and related events. We can look at the call stack from the Products microservice down to the database. You can see the error with MySQL.

This is yet another confirmation of the source of the problem.

# Fixing the problem with runbook automation

**Action**

## Navigate to **Stories** view. Click the name of the available story to open the details

Graphical user interface, application

Description automatically generated

**Action**

## Run the **Mitigate RobotShop Problem** runbook.

Graphical user interface, text, application, Teams

Description automatically generated

Background pattern

Description automatically generated

## Click **Start Runbook**. Click **Run** in Step 1. When finished, click **Complete**.

**Note**

The execution of the runbook can take few minutes. In case it takes too long click Complete and just follow the steps in section 7.5 to resolve the story.

**Narration**

Now that we know what the problem is, let’s correct what has happened. A runbook has been automatically identified but have not been executed. Runbooks are guided steps that IT operations teams use to troubleshoot and resolve problems. Some organizations might call these standard operating procedures or playbooks. When an incident occurs, IBM Cloud Pak for Watson AIOps matches an appropriate runbook to the problem. The runbook can be set to run automatically when it is matched to an incident, or it can run with user approval and participation.

Let’s take a look at the runbooks identified by Cloud Pak for Watson AIOps. You can see that the runbook has three steps. Watch the execution as it connects to the cluster and then scales up memory for the MySQL deployment.

**Action**

## [Optional] Open the RobotShop application. Verify that ratings are correctly shown

**Narration**

Before confirming that the runbook worked as expected, I should check the RobotShop application to see if it is working as expected.

**Action**

## Go back to story in the Slack. Click **Mark as resolved**.

Graphical user interface, text, application, email

Description automatically generated

## Click in the **Notes** section. Provide some sample note. *Result: Notes are added to the story*.

## Click **Submit**. *Result: Status changes to Resolved*.

Graphical user interface, text, application, email

Description automatically generated

# Demonstration summary

Today, I have shown you how Cloud Pak for Watson AIOps can assist the SRE/Operations team to identify, verify, and ultimately correct an issue with a modern, distributed application running in a cloud-native environment. The presented solution provides automatic application topology discovery, anomaly detection both with metrics and logs, and sophisticated methods of correlation of events coming from different sources. You have also seen how integration with other tools allow the effective implementation of a ChatOps environment.