VMware Smart Assurance K4M 1.0.0

User Guide

Abstract

This user guide describes use of the K4M KPI engine service in Smart Assurance. This service provides facilities for computing KPIs over a set of metrics.

REV 190422

Published in the USA April 2019 VMware Smart Assurance K4M 1.0.0 User Guide

Table of Contents

Table of Figures

Chapter 1 Introduction

About this document

This document is a user guide for the VMware K4M product. It describes the product's features, technical details and usage.

Organization

This guide consists of the following main topics:

- An overview of VMware K4M.
- Important concepts.
- A description of the overall architecture including all the components and technologies.
- A reference guide to installation and administration, including operation and troubleshooting details.
- Details on how to create Key Performance Indicators (KPIs) and compute them using the KPI Stream entity.
- REST API used to perform various operations with VMware K4M is described.

Additionally, the user should refer to the following documents for important topics:

- **VMware Smart Assurance K4M 1.0.0 Installation and Administration Guide** - Installation and administration including troubleshooting details are covered in this document.
- **VMware Smart Assurance K4M 1.0.0 KPI Designer User Guide** – This document describes how to create and deploy KPI definitions and KPI streams via a Graphical User Interface.
- Installation and operation details of the KPI Designer are available in the **VMware Smart Assurance UI documentation.** Though part of the VMware K4M product, the KPI Designer is installed as part of the VMware Smart Assurance UI.

About VMware K4M

K4M is a general-purpose Key Performance Indicator (KPI) computation engine.

It provides a model for defining KPIs and a runtime stream processing engine to compute them. It employs the notion of KPI Stream which is the entity that connects sources of input data (such as metrics) to a KPI calculation and the output of that calculation to a destination. Message brokers and databases are examples of sources and destinations. Input is expected to arrive as an unbounded stream of events with values. The KPI is computed over all the values arriving within a specific interval of time or window. At the heart of VMware K4M lies the KPI Engine that provides the runtime platform for processing streaming input, computing KPIs and distributing KPIs to destinations. VMware K4M accepts input data in different formats. Computed KPIs can be output in different formats as well.

NOTE

1. K4M does NOT collect metrics. It performs computation using metrics from an input source. It is assumed separate systems collect metrics and make them available to the input source.
2. K4M does NOT report on KPIs and metrics. It distributes KPIs to a destination. Separate systems are responsible for reporting.

Usage in VMware Smart Assurance

VMware K4M is provided as an optional add-on to the Smart Assurance suite. Features and usage:

- K4M should be deployed if any KPI computation is needed.
- Can be used to calculate KPIs over system metrics, such as load averages, memory or bandwidth utilization, or Quality of Service heuristic values.
- Communicates with other Smart Assurance components via common Kafka broker.
- Contains out-of-the-box capability to compute vIMS KPIs using metrics collected by DCF SNMP Clearwater Collector (part of VMware Smart Assurance).
    - DCF publishes collected metrics onto Kafka.
    - K4M subscribes to above metrics, computes KPIs and publishes KPIs back to Kafka.
    - KPIs are reported by VROps using the Smart Assurance Management Pack.
      NOTE: For details, please refer to:

    1. **VMware vIMS SNMP collector** chapter from **VMware Smart Assurance UI Platform User and Configuration Guide**.
    2. **vRealize Operations Management Pack for Smart Assurance Adapter Guide** from VMware support centre https://www.vmware.com/support.html

- K4M has high performance and scalability.

Chapter 2 Concepts

KPI Value or KPI

Also referred to as *KPI*, this is the entity computed by the KPI Engine. It is defined by a *KPI Definition* (described below) that consists of its *calculation* expression and other relevant properties. A *KPI* is computed over a stream of input values or *metrics*, arriving within a time *window*.

Metric

A named measurement, collected from a source. Usually (but not necessarily) a numeric value. *Metrics* are collected by the VMware Smart Assurance Data Collection Framework (DCF).

Metric Catalog

A structured list of *metrics* associated with some managed entity. The *metric catalog* is a JSON file that users can add to VMware K4M so that these metrics are available in the KPI Designer when creating KPI definitions.

NOTE: Currently the *metric catalog* must be submitted to VMware K4M via its REST API. It is not possible to add it via the KPI Designer in this release.

Calculation

A computation over a *metric* or set of *metrics*, for example, computing a *KPI* viz. their average. A *calculation* specifies a list of *metric* names, the language used to define the expression (Spring Expression Language (SpEL)), and the computation expression containing arithmetic operators and/or functions applied over the *metrics*.

Window

The time window over which a *calculation* is performed. *Windows* can be sliding (i.e., overlapping) or tumbling (i.e., disjoint). The *calculation* is performed over all the *metric* values arriving within the *window* to generate a *KPI Value*. The user can choose to use the event's own timestamp generated at its source or the system time when processing the event. This specification also includes the periodicity or frequency with which a *KPI Value* is to be calculated. The configuration of *window* attributes is described in greater detail as part of the *KPI Definition* creation.

Group

Key for grouping input *metrics* - for example, by the device to which the metric applies. Computed *KPI Values* will be grouped by the same key.

KPI Definition

A uniquely named group of properties that describes one or more *KPIs.* This includes each *KPI*'s *calculation* and settings common to all *KPI*s viz. a *window* definition, *group* and *metric* filters.

Formatter

Converts between the external and internal VMware K4M representations of a *metric* or *KPI Value*. *Input Formatters* parse incoming *metrics* and *Output Formatters* format the computed KPI Values. Out-of-box formatters are provided for JSON and CSV formats. The *Formatter* is specified by its name.

Input

Set of attributes of the source of *metrics* which will be ingested into the KPI Engine. Attributes can include URL or IP address and port number of the source service. In VMware Smart Assurance 10.0, this will be one or more topics on one or more Kafka brokers. Also specifies a *formatter* used to convert from the external representation of *metric*s to their internal format.

Output

Set of attributes of the destination of computed *KPI Value*s output by the KPI Engine. Destination can be a service such as a message broker that publishes *KPI Value*s or a database that stores them for future use. In VMware Smart Assurance 10.0, this will be a topic on a Kafka broker. Also specifies a *formatter* used to convert from the internal representation of values to their external format.

KPI Stream

A uniquely named association that connects one or more *KPI Definition*s with one or more *input*s and one or more *output*s. When the *KPI Stream* is deployed, the KPI Engine processes the incoming metric streams as defined by the *input*s, computes the *KPI Value*s as specified in the *KPI Definition*s and outputs the *KPI Value*s as specified in the *outputs*.

KPI Job

A running instance of a *KPI Stream*. Created in the KPI Engine when the *KPI Stream* is deployed.

Service

A component that provides specific function (stream processing, storage, messaging, configuration, etc.) and enables interoperability and integration options. All VMware K4M components are considered to be *services* with properties that include identifiers, access points and status. These details are available via its REST API.

NOTE: Currently the ability to create/update/delete *Service*(s) is available only through the REST API and not via the KPI Designer.

Figure 1 below depicts some of the entities and their relationships, described above.

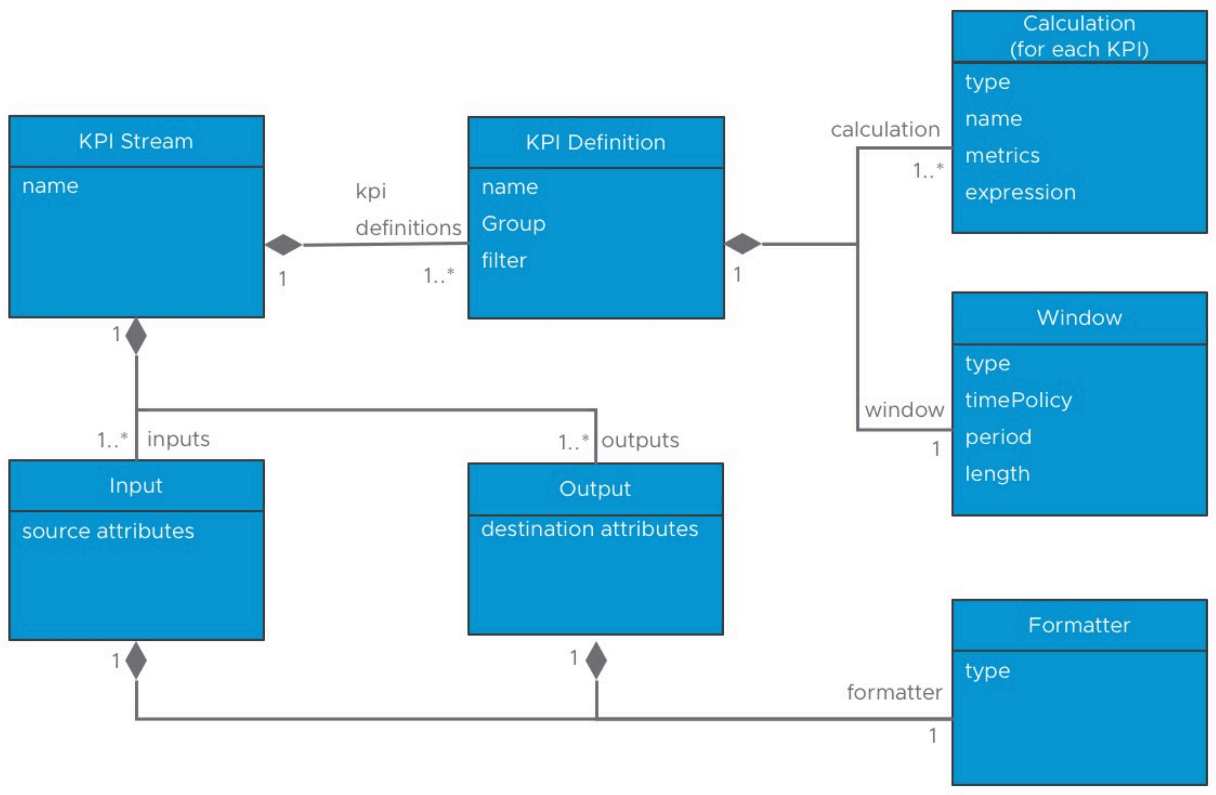


Figure 1 VMware K4M Concepts

Chapter 3 VMware K4M Architecture

Architecture, including components and technologies are described in this section. Figure 2 below illustrates the internal architecture of VMware K4M as well as external dependencies it integrates with as part of an end-to-end solution such as VMware Smart Assurance. Note that KPI Designer is hosted on a separate UI Node and the rest of the VMware K4M components are hosted on the Compute Node (K4M Host).

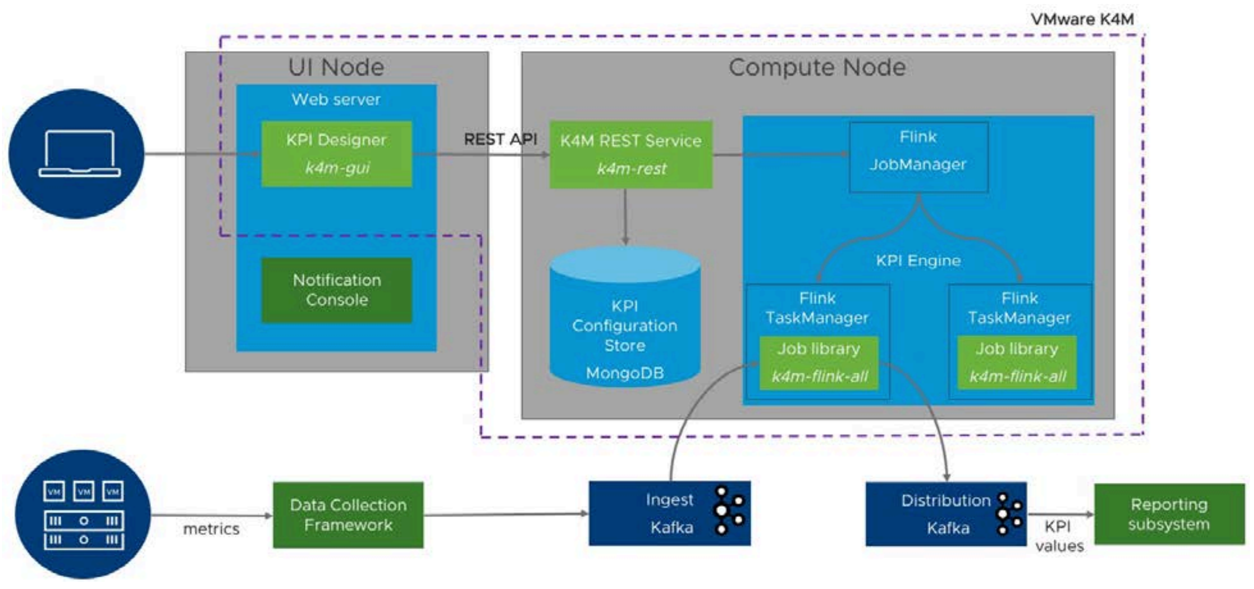


Figure 2 VMware K4M Architecture

VMware K4M Components

KPI Designer

Users interact with VMware K4M via its KPI Designer Graphical User Interface (GUI). They can,

1. Create, configure and save KPI Definitions and KPI Streams.
2. KPI Streams can be deployed to initiate computation of KPIs in the backend KPI Engine.
3. Existing KPI Definitions and KPI Streams can be viewed and further edited.

KPI Designer is a Clarity-conformant Angular web-based application.

This component is available as part of the VMware Smart Assurance UI and is hosted along with the Notification Console. Please refer to the **VMware Smart Assurance UI documentation** for installation details.

Please refer to the **VMware Smart Assurance K4M 1.0.0 KPI Designer User Guide** for details on how to use the KPI Designer.

K4M REST Service

The K4M REST Service, responds to user requests via the KPI Designer GUI and performs all required backend tasks in VMware K4M. It provides a REST interface (API) to VMware K4M. The main tasks include:

1. Create/Update/Delete all KPI Definitions, KPI Streams and metric catalogs by connecting to the KPI Configuration Store.
2. Deploy streams by sending KPI Streams(s) to the KPI Engine.
3. Monitoring the health of VMware K4M components such as the KPI Engine, KPI Configuration Store, its own health as well as any Kafka broker that VMware K4M integrates with.

The REST API details are described in a subsequent section.

NOTE: It is recommended to use the KPI Designer to create and deploy KPI Definitions and KPI Stream, and not directly use the REST API (unless mentioned explicitly otherwise), as the KPI Designer provides capabilities for a more user-friendly experience.

The K4M REST Service is a Spring Boot [2] application and uses Spring Framework [3].

KPI Engine

The KPI Engine is responsible for running one or more KPI Streams that are deployed on it by creating "jobs". The execution of KPI Streams involves three major steps:

1. Connect to the input source and process the incoming stream of metrics. This involves parsing them into an internal representation which can be used for computation.
2. As per the KPI Definition, KPIs are computed from the metrics. This includes filtering unwanted metrics, grouping them by the group-by key and applying the computation expression over all the metrics in the specified window.
3. The computed KPIs are converted into the external representation as specified in the output formatter and dispatched to the destination.

The KPI Engine is based on Apache Flink [4]. Flink is a distributed data stream processing engine. It uses the following processes to run KPI streams:

- *Flink JobManager* – Flink master controller; launches and manages execution of tasks.
- *Flink TaskManager* – Flink worker; executes tasks.
- *Job library* – Execution environment for K4M jobs; provides common library services such as aggregation functions.

Users do not have to directly deal with low-level details of distributed stream processing, such as the individual processes described above. They interact via the KPI Designer GUI as described above.

KPI Configuration Store

This is a configuration database that stores JSON representation of user-created artifacts required for computing KPIs viz.

1. KPI definitions (each represented as a KPI Definition JSON document)
2. KPI streams (each represented as a KPI Stream JSON document)
3. List of metrics available for KPI computation (represented as a Catalog JSON document).

The store is implemented using the open source MongoDB NoSQL DBMS [5].

External Dependencies

VMware K4M can integrate with different metric sources and destination services. As part of the VMware Smart Assurance solution it integrates with the following components:

Data Collection Framework (DCF)

Collects metrics from systems being monitored, and publishes them in a standard format to a Kafka broker for consumption by VMware K4M.

Kafka

Message broker, provides communication channel between DCF, VMware K4M and reporting subsystem.

Reporting subsystem

Applications such as VROps can consume KPIs generated by VMware K4M to create dashboards and reports.

Notification Console

Clarity-conformant Angular UI component, co-located with KPI Designer.

Chapter 4 Installation and Administration

Please refer to the **VMware Smart Assurance K4M 1.0.0 Administration and Installation Guide** for details related to installation and administration including troubleshooting.

Out-of-the-box vIMS KPI definitions

VMware K4M comes with a set of KPIs along with the corresponding stream and catalog definitions for the vIMS use case. Instructions to deploy the KPIs are part of the installation guide. These definition files can be used as examples of the JSON DSL. Please refer to the **VMware Smart Assurance K4M 1.0.0 Administration and Installation Guide** for more details.

Creating and Computing KPIs

The steps to create and compute KPIs are:

1. Upload the metric catalog. The catalog should contain all the metrics you require to compute KPIs from. You may have multiple catalogs.

2. Create a KPI Definition that contains the KPIs you wish to compute.

3. Create a KPI Stream. Configure it such that, the input metric source is connected to the KPI Definition you created and the KPI Definition is connected to the output destination. You may have multiple input sources and output destinations.

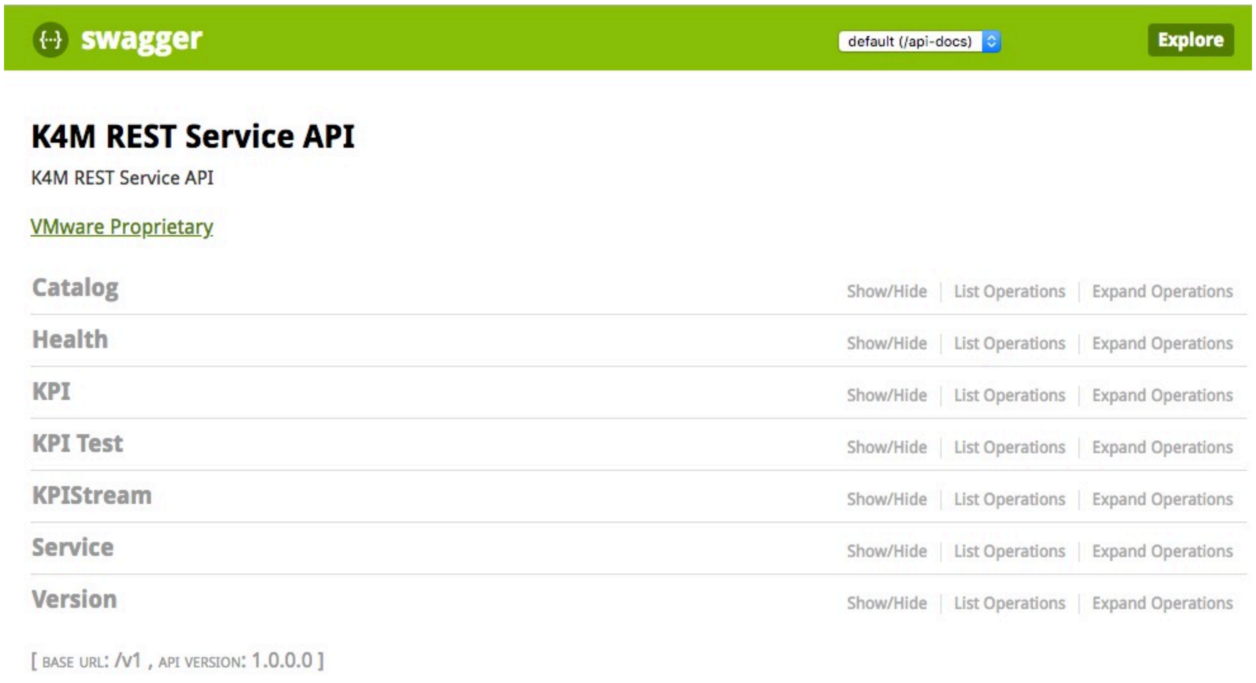
4. Deploy the KPI Stream.

5. Verify that KPIs have reached the destination.

For the steps 1 to 4 above Please refer to the **VMware Smart Assurance K4M 1.0.0 KPI Designer User Guide** which describes them in detail.

Chapter 5 REST API

The K4M REST Service exposes a REST API that can be accessed at http://K4M_HOST_IP:8083/v1 as shown in Figure 3 below.



K4M REST Service API

K4M REST Service API

VMware Proprietary

Catalog	Show/Hide | List Operations | Expand Operations
Health	Show/Hide | List Operations | Expand Operations
KPI	Show/Hide | List Operations | Expand Operations
KPI Test	Show/Hide | List Operations | Expand Operations
KPIStream	Show/Hide | List Operations | Expand Operations
Service	Show/Hide | List Operations | Expand Operations
Version	Show/Hide | List Operations | Expand Operations

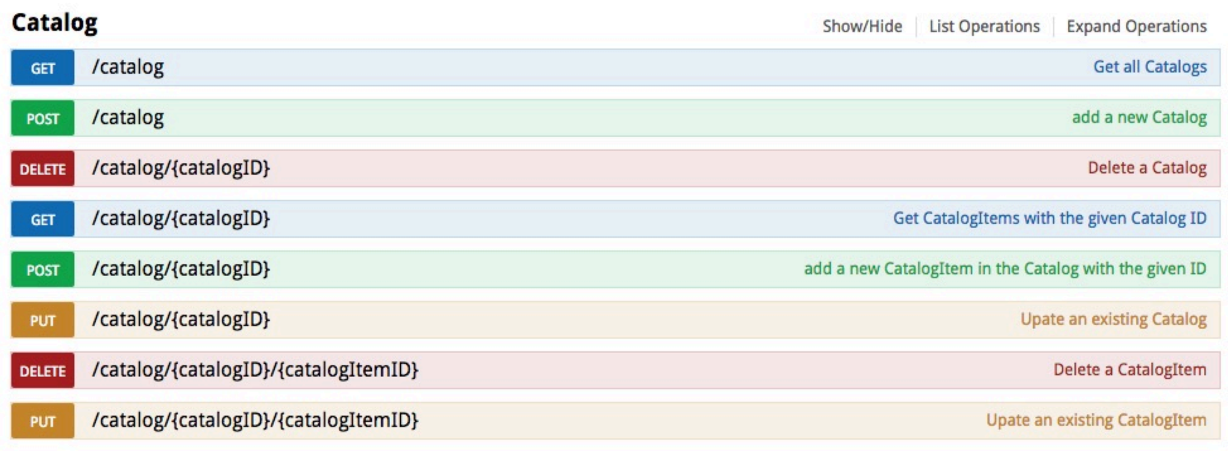
[BASE URL: /v1 , API VERSION: 1.0.0.0]

Figure 3 REST API endpoints

This section provides a summary of all the endpoints. For further detail please refer to the above URL in your VMware K4M instance.

For details about using the Swagger UI in general please refer to Swagger documentation [6].

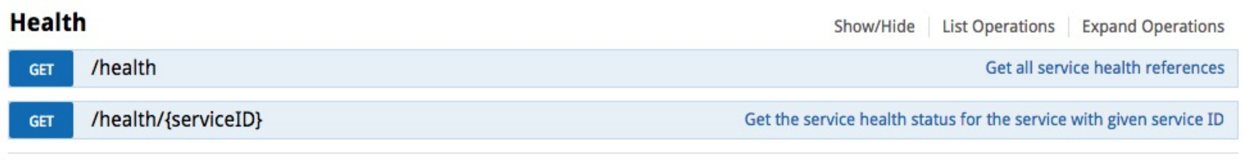
Catalog


Catalog Show/Hide | List Operations | Expand Operations

GET	/catalog	Get all Catalogs
POST	/catalog	add a new Catalog
DELETE	/catalog/{catalogID}	Delete a Catalog
GET	/catalog/{catalogID}	Get CatalogItems with the given Catalog ID
POST	/catalog/{catalogID}	add a new CatalogItem in the Catalog with the given ID
PUT	/catalog/{catalogID}	Upate an existing Catalog
DELETE	/catalog/{catalogID}/{catalogItemID}	Delete a CatalogItem
PUT	/catalog/{catalogID}/{catalogItemID}	Upate an existing CatalogItem

Figure 4 Catalog REST endpoint

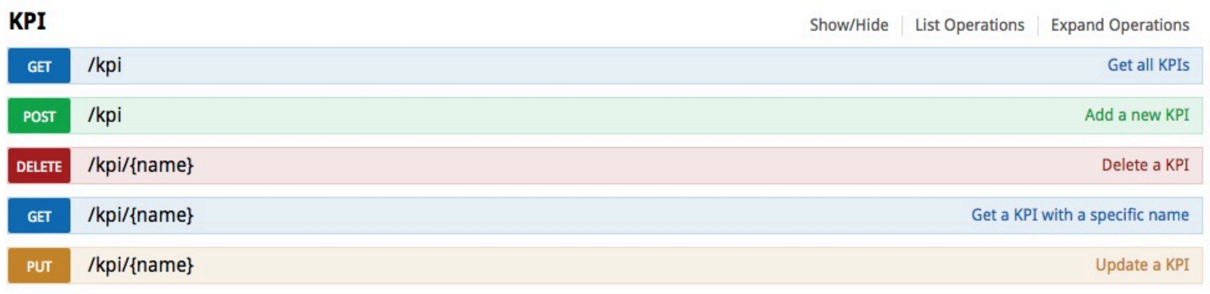
Health


Health Show/Hide | List Operations | Expand Operations

GET	/health	Get all service health references
GET	/health/{serviceID}	Get the service health status for the service with given service ID

Figure 5 Health REST endpoint

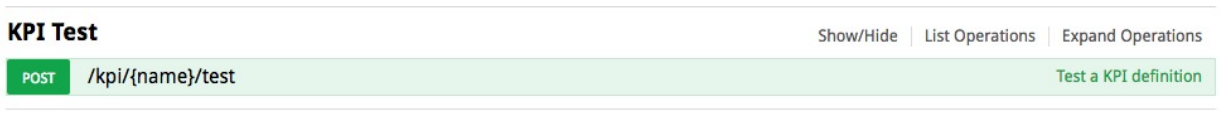
KPI


KPI Show/Hide | List Operations | Expand Operations

GET	/kpi	Get all KPIs
POST	/kpi	Add a new KPI
DELETE	/kpi/{name}	Delete a KPI
GET	/kpi/{name}	Get a KPI with a specific name
PUT	/kpi/{name}	Update a KPI

Figure 6 KPI REST endpoint

KPITest

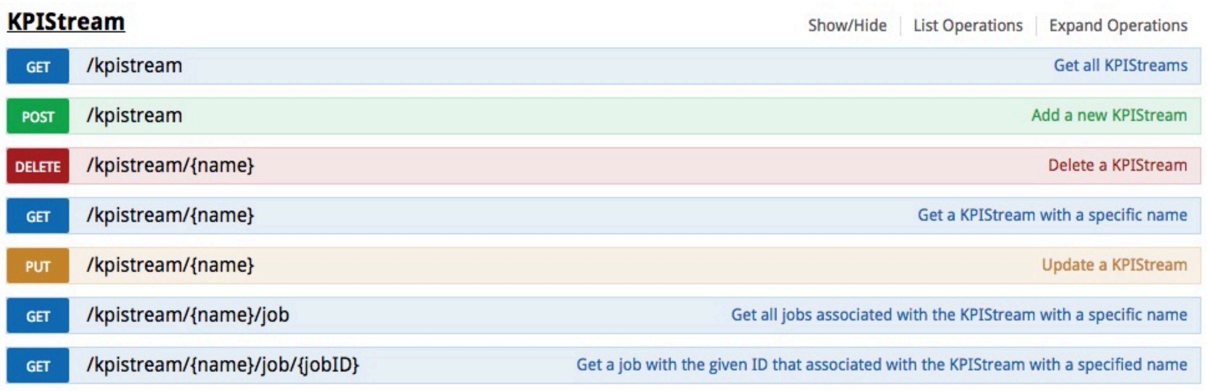

KPI Test

Show/Hide | List Operations | Expand Operations

POST	/kpi/{name}/test	Test a KPI definition

Figure 7 KPITest REST endpoint

KPIStream

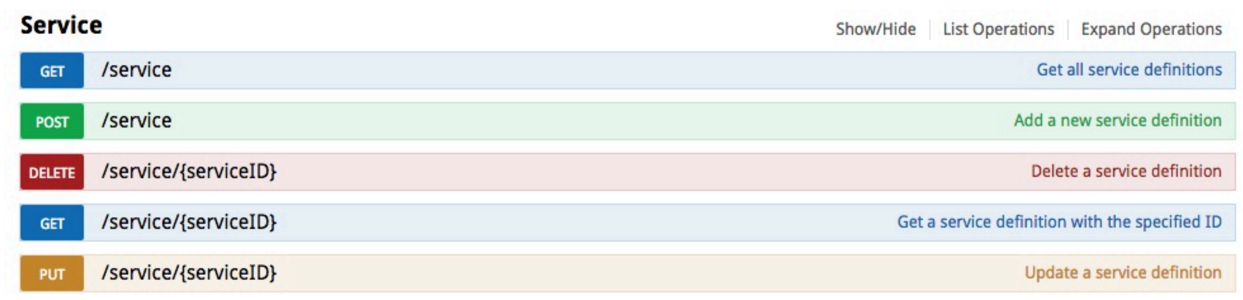

KPIStream

Show/Hide | List Operations | Expand Operations

GET	/kpistream	Get all KPIStreams
POST	/kpistream	Add a new KPIStream
DELETE	/kpistream/{name}	Delete a KPIStream
GET	/kpistream/{name}	Get a KPIStream with a specific name
PUT	/kpistream/{name}	Update a KPIStream
GET	/kpistream/{name}/job	Get all jobs associated with the KPIStream with a specific name
GET	/kpistream/{name}/job/{jobID}	Get a job with the given ID that associated with the KPIStream with a specified name

Figure 8 KPIStream REST endpoint

Service

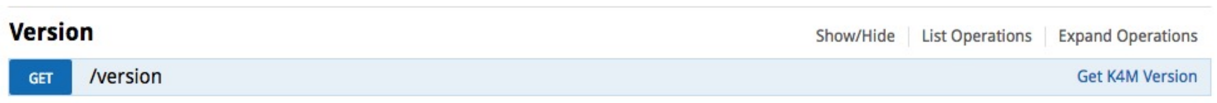

Service

Show/Hide | List Operations | Expand Operations

GET	/service	Get all service definitions
POST	/service	Add a new service definition
DELETE	/service/{serviceID}	Delete a service definition
GET	/service/{serviceID}	Get a service definition with the specified ID
PUT	/service/{serviceID}	Update a service definition

Figure 9 Service REST endpoint

Version


Version

Show/Hide | List Operations | Expand Operations

GET	/version	Get K4M Version

Figure 10 Version REST endpoint

Chapter 6 References

1. Apache Kafka - https://kafka.apache.org/documentation/

2. Spring Boot - https://spring.io/projects/spring-boot

3. Spring Framework - https://spring.io/projects

4. Apache Flink - https://flink.apache.org/

5. MongoDB - https://docs.mongodb.com/

6. Swagger UI - https://swagger.io/tools/swagger-ui/