EdgeX Foundry System Management Roadmap

Update: 12/6/2018 (v3)
Author: Jim White, Sys Mgmt WG Chair

Contents

Over the course of the last year, in addition to providing the first elements of system management functionality in the Delhi release of EdgeX, the community has also come together to better define what is considered in scope for EdgeX system management long term.

This document is meant to address the purpose and features of EdgeX system management functionality going forward. While there is room for change and re-evaluation in the future, the community has additionally taken some steps to clearly define management functionality that it does not consider in EdgeX purview and explicitly out of scope for system management for the foreseeable future. Note, some of these features may have been considered in scope during earlier scoping efforts of the project.

Context

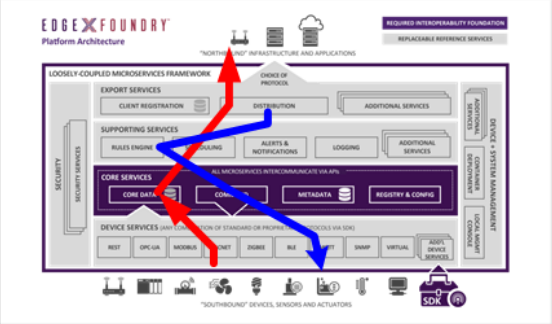
Control Plane versus Data Plane

The term “system management” can be misleading and lend itself to confusion when describing edge software capability. It may help to understand two planes of data and control that a software platform like EdgeX addresses.

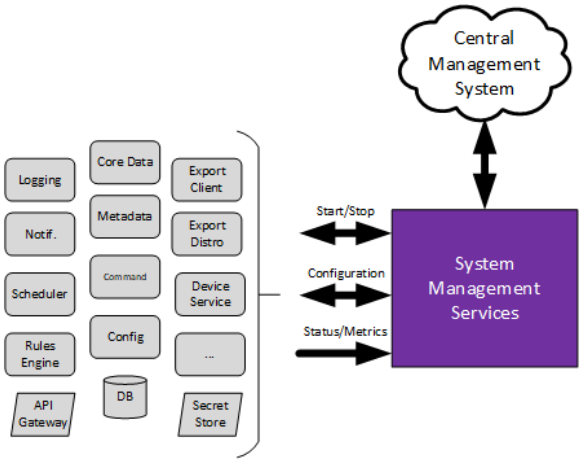
An IoT platform like EdgeX is used to collect and process the data from “things” – that is the platform ingests data that is physically sensed from IoT sensors and devices. The platform may manage and care for the data for short periods of time (such as store and protect the data in a database for later transport), perform operations on the data (such as transform it to a format that can be used by other applications or cloud systems), analyze the data for certain criteria that warrant taking an action such as actuating another device or sensor (ex: turning off a motor if a sensor detects abnormal vibration in the machinery), or alerting another system to a potential issue (ex: sending an SNMP alert message to a monitoring station if a temperature sensor detects excessive heat in a room). Work associated with collecting, managing and disbursing sensed data is work associated to the “data plane”.

The work associated to operating and managing the IoT platform software and infrastructure is “control plane” operations. This includes getting the IoT platform and infrastructure running (or shutdown), configuring the platform software for the particular use case, and understanding the health and status of the software platform (is it running and what type of resources is the IoT software platform using?). Analysis of any control plane data may be used to take action as well, but this action revolves around the IoT platform itself – not the sensed or controlled world. For example, in the control plane, it may be determined that a service needs to be restarted because it is consuming too much memory.

System management functionality, as determined by the EdgeX community, is generally associated with control plane data and operations. In short, the control plane (and system management) is about managing the IoT platform and infrastructure. The data plane is about managing and understanding the physical world that the IoT platform is there to observe and control.



Data Plane



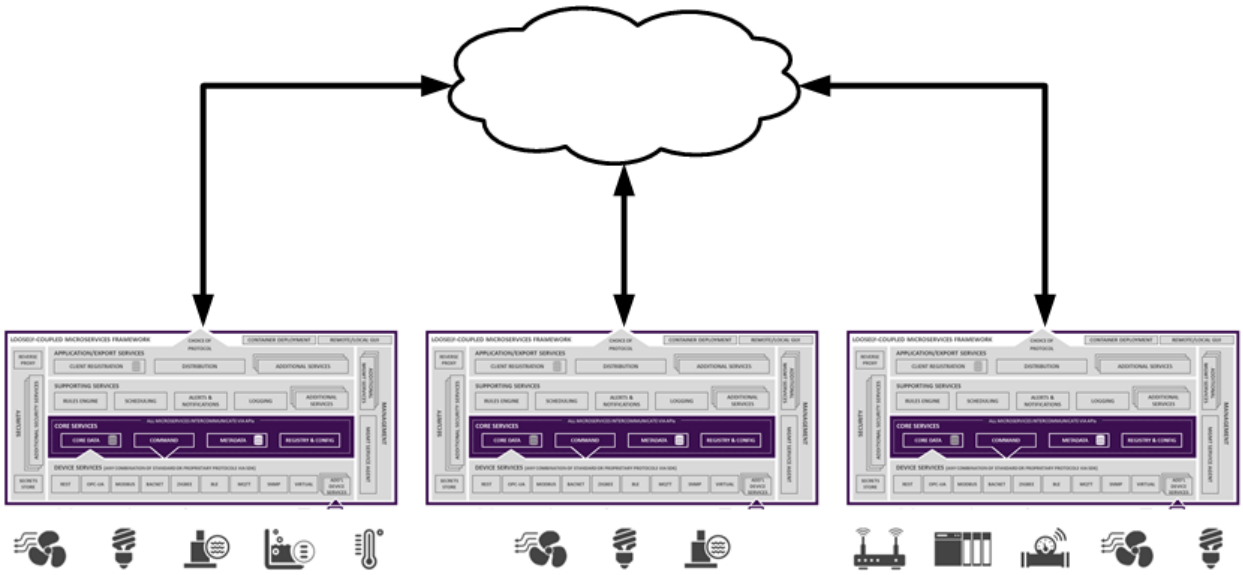
Control Plane

There are some use cases where the data plane and control plane overlap. For example, the functionality within the control plane may determine that a component of the IoT platform is using more memory than it should and could lead to overall system failure. This functionality does not yet exist in EdgeX, but eventually the EdgeX control plane could inform EdgeX sensor data ingestion services of the data plane to take fewer readings from the sensors in order to conserve system resources. These are advanced (and future) scenarios that do require thinking loosely about the concepts of data and control planes. As this is a roadmap document, these features are within the realm of future releases but not in the next few releases.

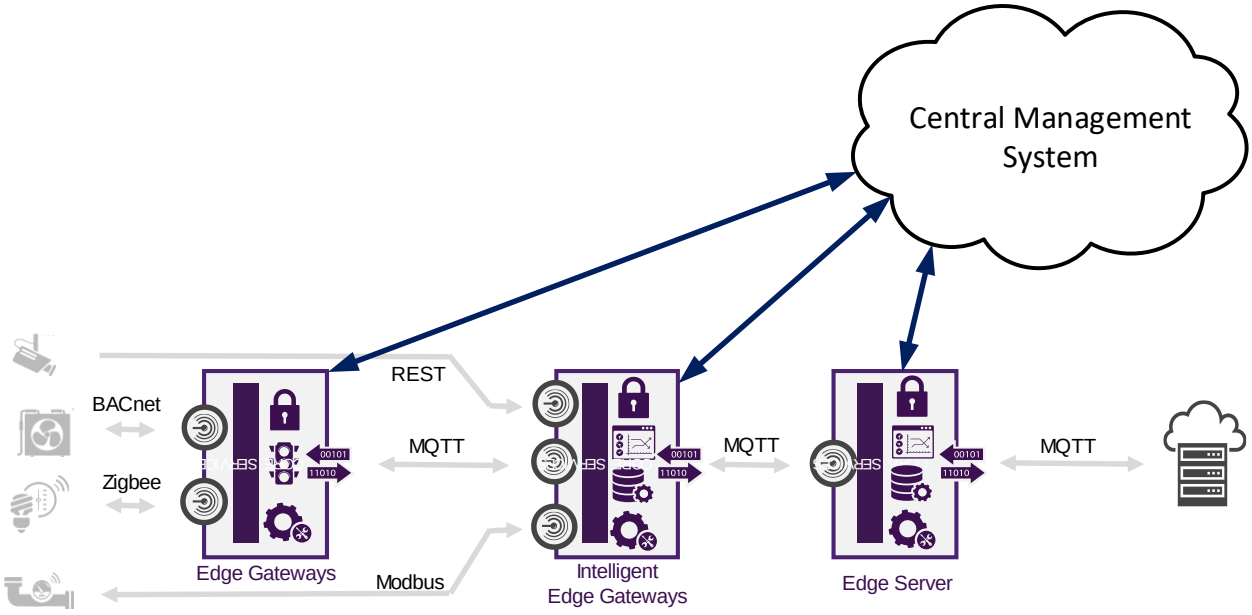
Importantly, the logical and complete separation of monitoring/managing the sensed world (the "things") versus monitoring/managing the IoT platform (EdgeX micro services) serves as a useful guide to help determine the boundaries of system management in EdgeX. EdgeX system management today is about facilitating some central management systems monitor and manage the EdgeX micro services.

Facilitating Multiple Management Systems

EdgeX is an edge platform. It typically runs as close to the physical sensor/device world as it can in order to provide the fastest and most efficient collection and reaction to the data that it can. In a larger "fog" deployment, there could be several instances of EdgeX each managing and controlling a subset of the "things" in the overall deployment.



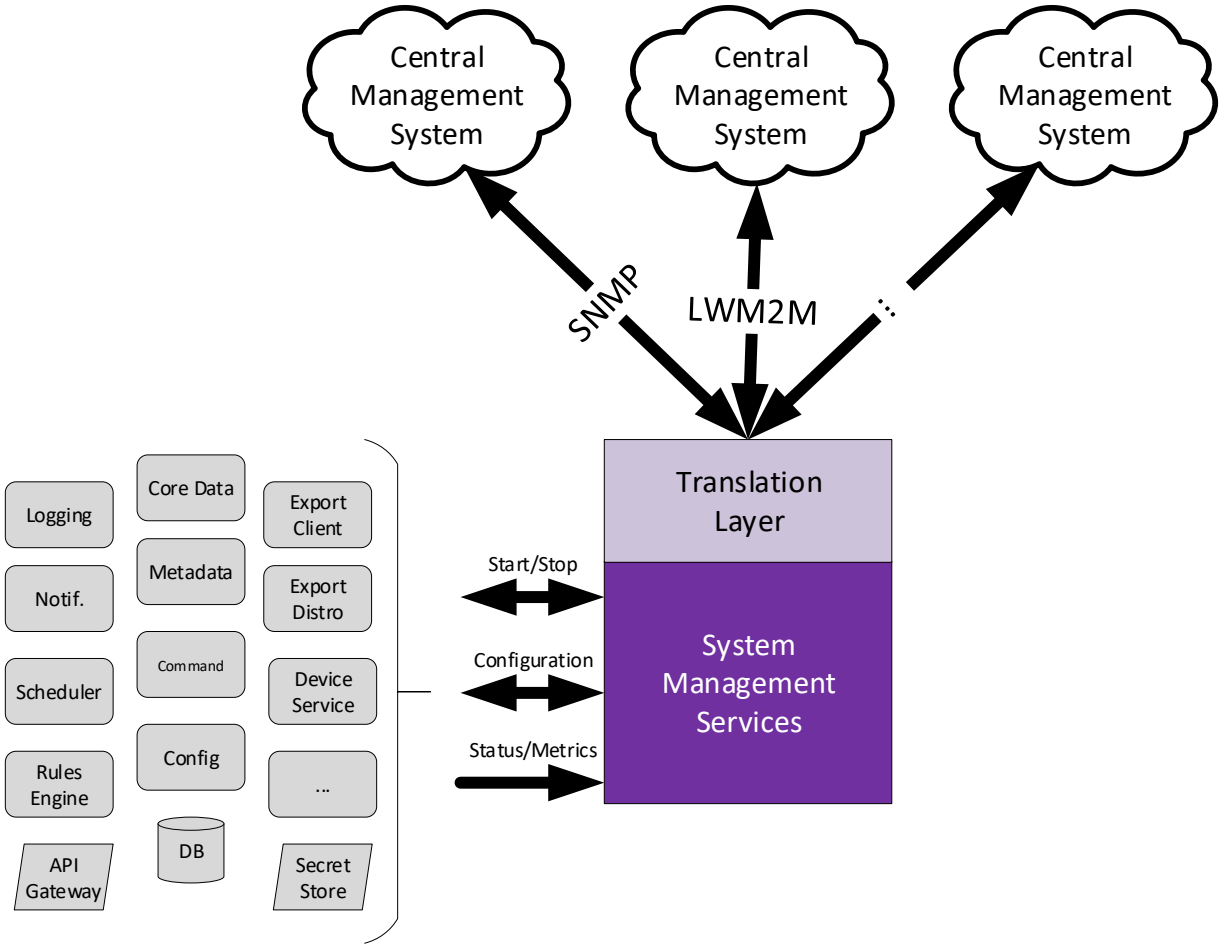
In a typical fog deployment, a larger management system will want to manage the control plane of the edge systems as well as all the intermediate and upper level nodes and resources of the overall deployment. Just as there is a management system to control all the nodes and infrastructure within a cloud data center, and across cloud data centers, so too there will likely be management systems that will manage and control all the nodes (from edge to cloud) and infrastructure of a complete fog or IoT deployment.



EdgeX system management is not the larger control management system. Instead, EdgeX system management capability is meant to facilitate the larger control management systems. When a management system wants to start or stop the entire fog deployment, EdgeX system management capability is there to receive the command and start or stop the EdgeX platform and associated infrastructure of the EdgeX instance that it is aware of.

Unfortunately, there are many control management systems today. Each of these systems operates differently. Many use different protocols and operate with different APIs. Just as EdgeX serves to provide the interoperability at the data plane level (speaking multiple sensor protocols and formats), EdgeX system management must provide interoperability at the control plane level (speaking multiple management system protocols and formats) – sometimes supporting multiple control plane level translations as the use case and deployment will require.

Because EdgeX is typically at the edge or farthest ends of an IoT deployment, it will not be central and in charge of a fog deployment. Therefore, EdgeX system management must be able to interoperate with any central management systems speaking a variety of control plane protocols (like SNMP, LWM2M, OMA DM, etc.). It must facilitate control plane operations at the edge, it cannot dictate the form or shape of control plane communications at the edge.



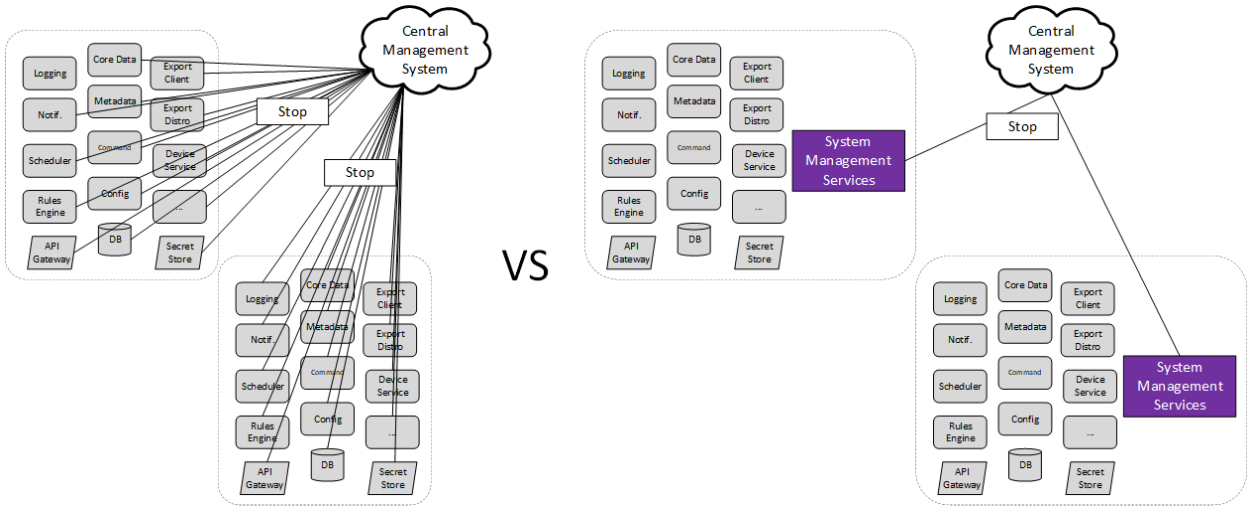
EdgeX System Management Functionality

EdgeX will offer the following types of control plane functionality:

Micro service operational control

As EdgeX is comprised of many loosely coupled, autonomous micro services, “starting” or “stopping” EdgeX is not defined by a single operation as in a monolithic application, but really a collection of starts or stops of all of the micro services and their infrastructure (like databases, registries, etc.). Micro services do not lend themselves to easy management by central management systems. A central management system would have to make several “stop” calls to stop a micro service platform, versus one stop in a monolithic application. The multi-component environment of a micro service solution adds to the scale and flexibility of the platform, but makes it harder for control management systems to manage.

Depending on the size of the IoT deployment, a central management system may have to manage hundreds or thousands of IoT edge nodes and therefore thousands or tens-of-thousands of EdgeX micro services. To better facilitate central management of many EdgeX instances, common operations such as start, stop and restart must adhere to a common API. Further, an EdgeX management service should be able to lateral the operation request to each micro service instance so as to reduce the central management system’s coupling and knowledge of each individual micro service.



Standardizing operational APIs and making use of an EdgeX control plane service to locally invoke the operational controls potentially reduces remote calls, improves latency, and allows for the operations to cascade from a central management point through a hierarchy of control plane nodes.

Today, operational control of EdgeX includes starting services (and infrastructure), stopping services (and infrastructure) and restarting services (and infrastructure). Restart may be a combination of stop and then start.

Micro service configuration management

Each EdgeX micro service will offer its configuration information through a control plane management service (the EdgeX system management service) to a central management system. While the configuration information for any service is available through either the configuration service (i.e. the Consul micro service today) or via local configuration (such as by configuration file deployed with the service), EdgeX will use the management service to pull the appropriate configuration from each micro service and make it available to central management and 3rd party systems.

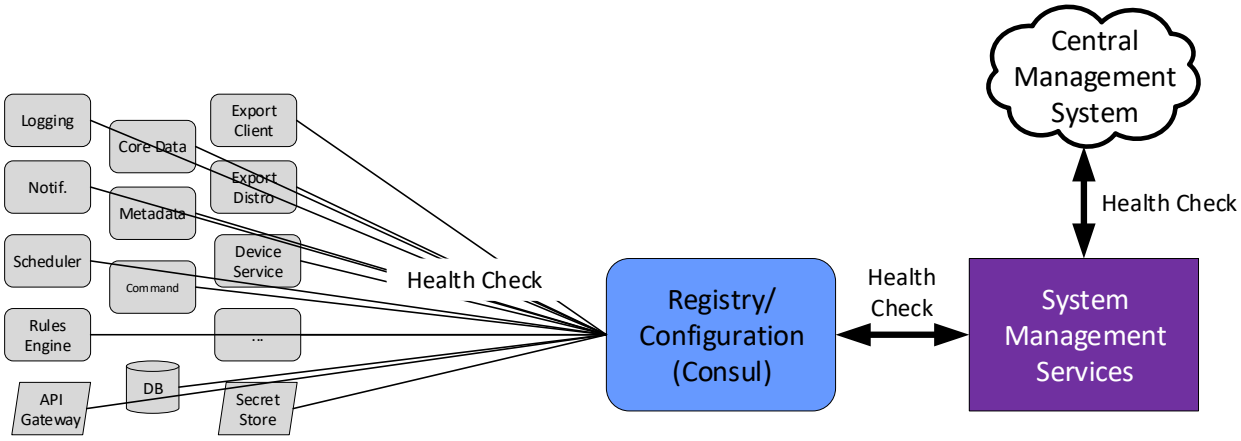
Some micro service configuration will be allowed to be updated, dynamically or between micro service restarts (example, changing a micro service port would require a restart of the service). EdgeX configuration organization and naming standards will be used to dictate which configuration properties are allowed to be updated (i.e. are writable versus read only). As with operational controls (like start, stop and restart), an EdgeX management service will provide central management systems access to update and change the configurations via common API.

Micro service operational and performance metric information

Central management systems will need to know if an EdgeX micro service or infrastructure element is operational and if it is performing within expected resource parameters (such as memory or CPU). The EdgeX management service must be able to collect and provide micro service (and associated infrastructure) performance metrics to central management and 3rd party systems. In the future, it will need to monitor that these parameters stay within configured parameters (example: any micro service cannot exceed 5% of CPU) and raise an alert if the metric does not fall within the expected parameters.

The operational status (it is either up or down and not responding) of an EdgeX micro service and its infrastructure, like performance metrics, is information that an EdgeX management service must be able to collect and provide to central management and 3rd party systems.

The EdgeX registry service (Consul today) performs route health checks (ping tests) of the EdgeX micro services and infrastructure. Rather than re-creating these same checks, the EdgeX management service will collect and relay the registry's information about the operational status of the service(s) to interested systems.

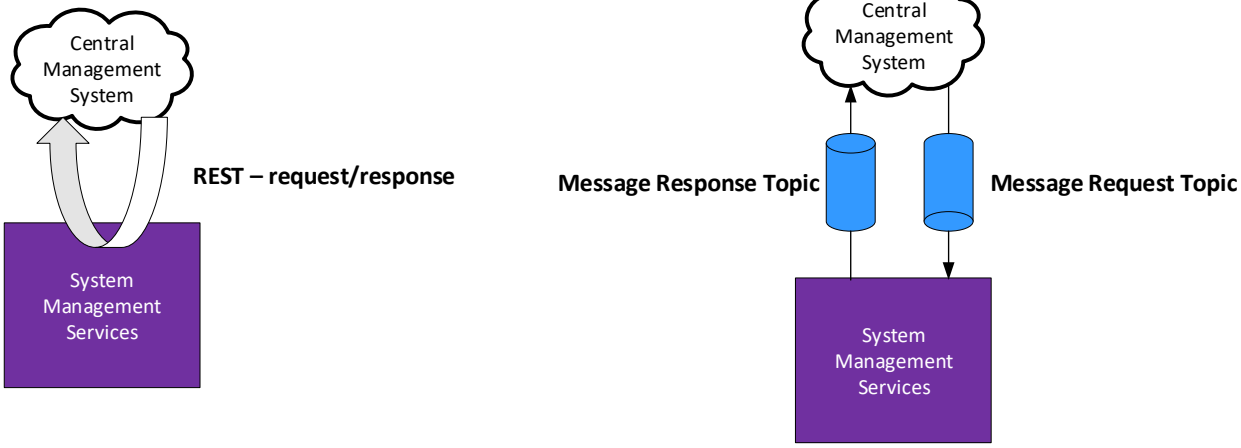


As with data plane information, EdgeX and its system management capability should someday offer the ability to persist or cache any control plane metric or status data so that it can be stored and forwarded to interested control systems during periods of connectivity (which can be short in some use cases).

Additionally, the control plane metrics are just data and, like sensor data (from the data plane), could be exported to third party systems like Cloud systems or enterprise applications. In future versions of EdgeX, the Export Services (soon to be the Application Services) could be used to transport data from EdgeX system management to any number of backend applications, cloud systems, etc.

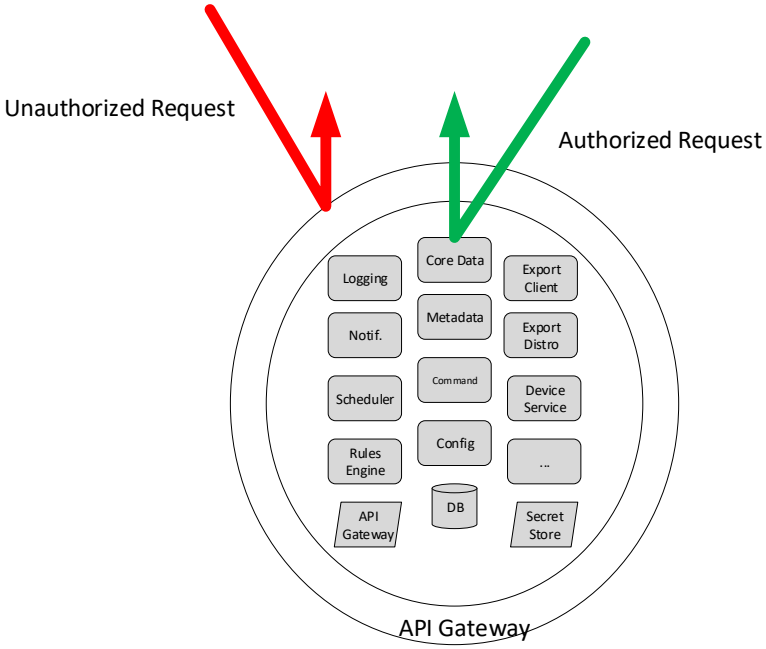
Currently, the system management operational and performance metric information is collected on request – meaning it requires the central management system to request it (via REST request) of the EdgeX system management service. In the future, it is envisioned that the collection occurs on some sort of schedule and it is pushed to the central management systems on a configurable schedule.

While the EdgeX system management service is REST based today, in the future, the communication with the system management service may be via message bus – allowing for more asynchronous and quality of service leveled communications.



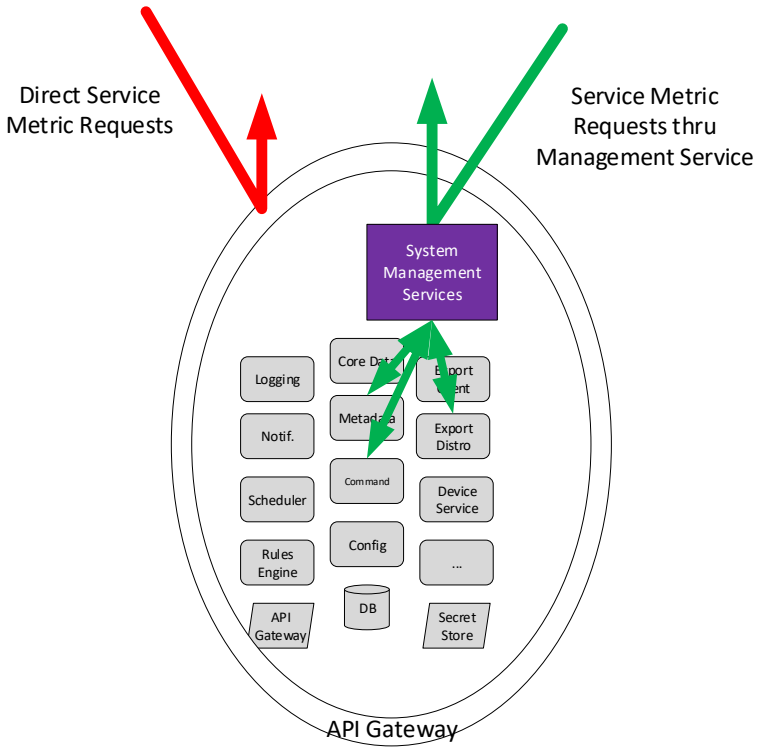
System Management Service (aka system management agent)

All the EdgeX micro service APIs will often be protected from outside requests.



The system management service will assist in protecting EdgeX and reducing the surface area of an API attack. Rather than opening up access to all services to the central management system, the system management service serves as a single point proxy to the control plane for all of EdgeX services for the central management system. The system management system thereby reduces the number of access points to EdgeX and reduces potential security vulnerabilities.

It also allows the central management system to be loosely coupled to all of EdgeX – requiring the central management system to again have just one access address (the address of the system management service) that it needs to know about for any EdgeX deployment.



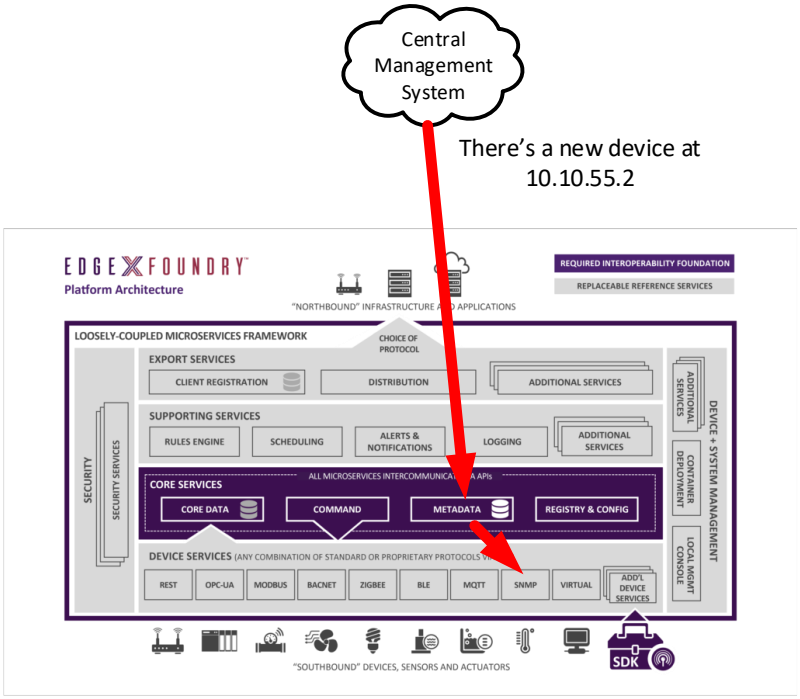
Finally, an EdgeX system management service allows all control plane requests/responses to be made in protocols and data format exchanges that very across central management systems. The system management service can transform central management requests into EdgeX understand control plane requests of each service and then transform the response back to the central management system's protocol/data format. In this way, the system management service acts as the interoperability engine driving the control plane (like device services and export services provide the same on the data plane).

The system management service may facilitate a few translations in the reference implementation of EdgeX but provides a future value add point for central management system and 3rd party providers wanting to provide additional translations.

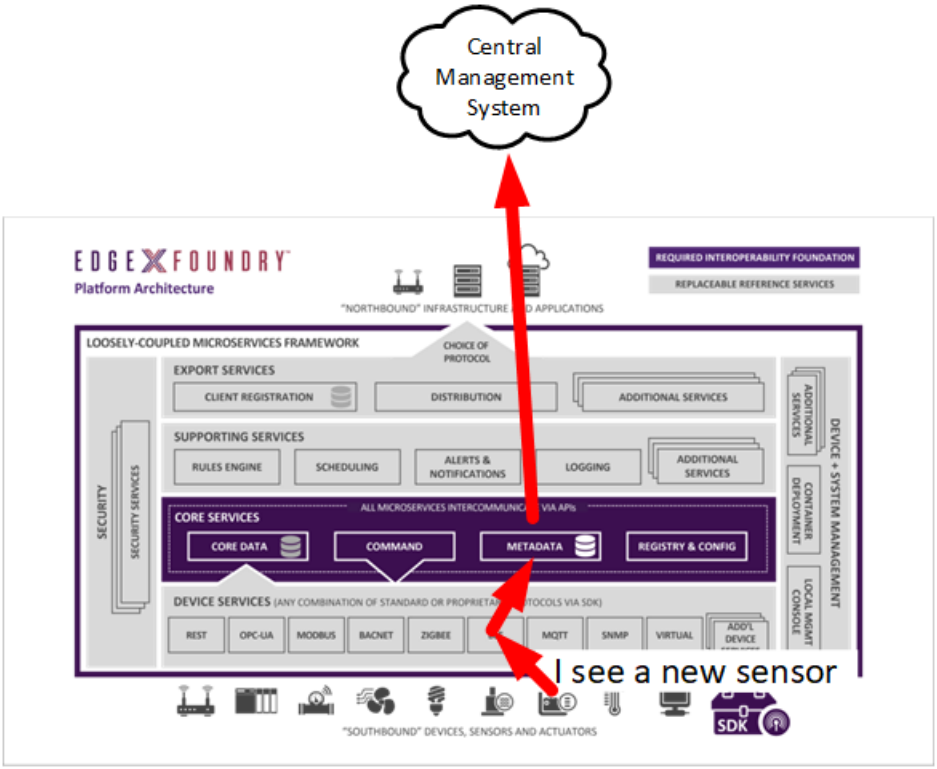
Note: a deployment of EdgeX could span multiple physical compute nodes (i.e. be distributed) and may include multiple instances of an EdgeX service (i.e. allow for load balancing of work across multiple copies of a micro service).

Device Onboarding

The ability to onboard or provision a new device with an IoT platform varies per device/sensor, protocol, security concerns, use case and a host of other factors. In some cases, a central management system will provide EdgeX with information about a new device it must manage. This is referred to as top-down provisioning. For example, a central management system may tell and instance of EdgeX to connect to (and manage) a new device at a specific IP address.



In other cases, automatic discovery of sensors and devices from within EdgeX (typically by a device service) will cause EdgeX to provision a new device. This is referred to as bottom-up provisioning. For example, a BLE device service could be searching for new BLE devices that enter platforms BLE range. If a new sensor is detected in its spectrum, the device service can start to automatically provision that device.



In the future, the EdgeX system management service must assist in provisioning a new sensor/device by keeping the EdgeX instance and central management systems informed – in both directions – of new device onboarding. It will need to provide APIs to be called from bottom up or top down that will help inform the other side of the new sensor/device addition and help to perform ancillary tasks (sometimes including service restarts) in order to successfully on board (or disconnect) a new sensor/device.

System Management Service is Optional

We have to accept that the system management service (and some of the API in the sys management API of each service) may be provided by other capability available in the fog deployment. For example, Kubernetes can start, stop and restart containers if EdgeX is deployed in a containerized environment managed by Kubernetes. Kubernetes could monitor some health aspects of an EdgeX container and restart the micro services if the services stopped. Other management facilities can monitor resource usage of EdgeX micro services and infrastructure and take action if it finds EdgeX micro services operating outside of prescribed parameters.

Therefore, EdgeX system management services may be disabled (and turned off) for some deployments. EdgeX services, which provide the system management API, must have a configuration setting that disables the API. This ensures that rouge processes cannot use the SM APIs to bypass alternate system management frameworks.

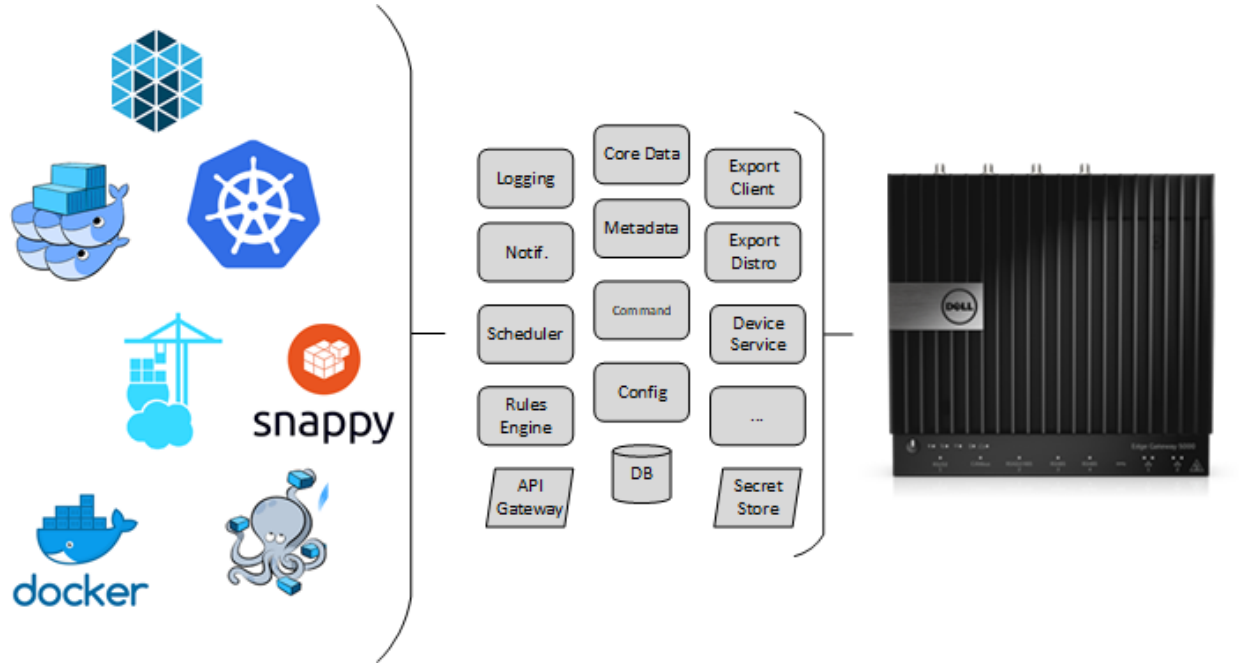
Out of scope

While often discussed in EdgeX architectural meetings, it has been decided that the following functionality will not reside in EdgeX and therefore will not be part of its future roadmap.

Orchestration & Deployment

As a loosely coupled, platform independent, distributable collection of micro services, EdgeX says nothing about how the micro services should be deployed, installed and run across compute nodes. Many of the cloud strategies (such as containerization and container orchestration in Kubernetes, Swarm, Mesos, Pivotal Container Services, and more) provide appropriate means and tools to deploy EdgeX for some use cases and deployment environments. Due to the resource constrained nature of many IoT environments, these cloud solutions may not always support the deployment and orchestration needs. Some organizations already have home grown and highly specialized means to deploy software into their constrained IoT infrastructure. Particularly small IoT deployments may even be managed with manual installation processes. Some operating systems now have orchestration capability (ex: Ubuntu Core with its snap technology).

Trying to support all the needs is outside the resources of the project. Additionally, other open source efforts and 3rd parties (some commercial) are attempting to solve some of these issues; therefore it is again important to try to facilitate all but not build to one (or a few), and to allow for creative solutions as the market determines which technologies will best support orchestration and deployment at the edge. The EdgeX community believes that it should be agnostic with regard to deployment and orchestration of the platform just as it is agnostic with regard to hardware and operating system.



EdgeX will continue to demonstrate and even provide example deployment and orchestration capability in some sample technologies (as it does for Docker Compose and Ubuntu Snaps today). EdgeX will also provide binary or other deployable artifacts (like Docker containers) that help to facilitate industry leading deployment and orchestration options as the community sees fit. EdgeX will also encourage and aim to facilitate additional deployment and orchestration options via outside projects and 3rd party hosted systems.

EdgeX will never dictate a deployment or orchestration strategy.

Updating non-EdgeX software or firmware (out-of-band updates or installations)

The deployment, installation, update, or uninstall of any non-EdgeX micro service or EdgeX required infrastructure software will not be performed by EdgeX or EdgeX system management services. This includes, but is not limited to

- Operating system patches or updates
- Container or orchestration tool
- Platform firmware or BIOS
- Device drivers

Many of these operations are conducted by out-of-band management systems. EdgeX considers these to be the realm of OOB management, central management or other 3rd party tools. EdgeX's system management service can and should help facilitate these systems perform their functions so far as EdgeX software is concerned (example: offering a stop API to stop all of EdgeX so that a new version can be installed), but not be directly responsible for update (or installation) of non-EdgeX software/firmware.

Distributed Management

When there exist several instances of EdgeX deployed over a vast array of compute nodes, EdgeX will not directly manage the failover, load balancing, scaling and other functionality as handled by cloud

cluster environments today. EdgeX will facilitate some base functionality to allow distribute management to occur (such as offering micro service metrics that can be used to alert a distributed management system to know when a micro service should be scaled out and load balanced), but it will not provide this functionality innately.