Device Services WG Meeting Minutes 2019-04-01

C SDK

Adding AutoEvents

Go SDK

Go updates from Cloud are pending. Have a PR open for AutoEvents.

Device Profile changes

Section names

Proposed renaming in the device profile:

- Resources DeviceCommands
- Commands CoreCommands

Voted on the name change, vote passed with +4

- [ ] Tony to file issues for the name change (against go-mod-core-contracts, edgex-go, device-sdk-go and other repositories) to resolve interlocks.

Resource/DeviceCommand operations

In Java code you can mix reads/writes in a DeviceCommand. The C and Go SDKs do not support these mixed operations.

Proposal to remove mixing "get" and "set" operations in a single resource/command.

- [ ] Tony to update device service requirements to simplify resource operations (remove mixing), and to incorporate comments from https://github.com/edgexfoundry/device-sdk-go/issues/213

Core Command proxies Device Commands

We anticipate scenarios still find this quite relevant. Such as those related to security, policy restrictions, caching, metrics. For example, denying third party calls initiated from outside EdgeX to not awake devices without system awareness.

Some performance critical scenarios may benefit from facilitating direct DS commands.

Service startup & registry

Not currently using core-metadata or registry (consul) to prevent multiple instances of the same device service.

DS Names

Regarding the presence of Device Service in Core Metadata. We are not using this as a semaphore to control how many instances start up.

Device Service names are passed to SDK. In Core Services these are supplied in constants.

DS Registration

We use registration with Consul, so DS will un-register on clean shutdown. However, there is no such unregister command currently.

We could add a flag in the SDK to reflect expectations (e.g., force a uniqueness check, add reference count that allows multiple instances, etc.)

- How to handle restarts of a crashed service?

Tony: When the registry is in-use, the original idea was that the [service] registry would provide the functionality to prevent multiple instances of a service from being started via it's registration mechanism. This is one of the reasons services should unregister themselves on graceful shutdown. That said, I'm not sure how we handle this if a service crashes.

The deployment method seems to own resiliency policy and determine which or how many DS instances gets started/not (SNAP, docker compose, kubernetes, SMA).. Or however you deploy it..

If running second instance, second will fail if it uses different host/port. Who is responsible to assign the host/port for second instance? Perhaps DS asks registry if already present - if so exits out. Higher level SMA/other is responsible to clear out stale registration. This could occur in response to # failed health checks for a registered service. The requirements and related policy is yet to be defined.

SMA may provide the glue to handle the needs across deployment methods.

Jim: The issue impacts all services, not just device services. For SMA to conduct the start/stop.. how does it know which are available?

Q. Can DS come up dynamically? After all things have come up? **A.** Yes.

Okay so how does SMA know of the new dynamic existence of DS? Requires Registry to be up – which introduces a new rule.

Enumerating services that are supposed to be present. How to assure you don't start two?

Follow Up: This is a topic of the next WG meeting with Trevor guiding team through a responsibility-driven approach.

Bottom Line: Trying to support high availability is not in scope for Edinburgh. By FUJI we will have an outline of who owns responsibility for what. Have a system in place that could run in cloud environment to bring things up/down and it survives. So will defer the discussion.

Remaining Device Profile Simplification

When a device is deleted from CoreMetadata it is the creator's responsibility to clean up adjacent/related data. This issue was started and punted for Edinburgh..

Relates to the Blacklist/Whitelist issue discussed at Edinburgh F2F.. if somebody (UI - IOTech/VMWare) creates a device, don't leave references to that device in CoreMetadata..

Opens: Such elements like events/addressables; flag to validate for ingestion. If I delete a device, should I delete all events? Do we convert from hard reference to soft reference; event.device could become a name instead of a lookup?

The renaming we discussed earlier is helpful, but there are also Device structs and some PR219 still on deck. Diana to work on this and tease out how CoreData works in device profiles (Tony ETA back by Thurs).

We split issue #828 (https://github.com/edgexfoundry/edgex-go/issues/828) to separately tackle near term profile simplification vs Fuji blacklist.

Action: Jim Moving 828 to Fuji declaration

Value Descriptor Endpoint

Tony is evaluating whether Value Descriptor endpoint will look at device (and device profile; commands and expected values). Concern about it possibly being incomplete.. Value Descriptors are created from Device Resources. If those being reported are only from CoreCommand API we could be missing some.
Diana: We enumerate through Commands which appear to be logically the same consolidated view in EdgeX system.
When Device profile is imported, SDK adds value descriptors but these are indirectly referenced when needed later. We have event/reading which has ref to device. Could we have legitimate Value Descriptor without values/readings?

Action: Tony to add command to Issue #150 (https://github.com/edgexfoundry/device-sdk-go/issues/150) by EOD today. Resource Operations. Jim may have resources if help is needed (with guidance provided).