EDGEX FOUNDRY™

EdgeX Face to Face Action Items and Decisions

Fuji Release

Seoul Korea, April 29 – May 2, 2019

Architect's Day Decisions

All persons attending the Architect's day, make sure the decisions captured are accurate to your best recollection.

- The Fuji release is targeted for end of October 2019 and it is likely to be a minor release
- The focus of the Fuji release is to "harden" the services, improve testing, and work to understand more of its performance characteristics
- While high availability and being able to run services in more of a distributed/load balanced way remains a future goal, it was determined that EdgeX will not focus on this for the Fuji release.
- On the issue of performance…
  - We eventually need to be able to answer 3 basic questions:
  - 1) Will EdgeX fit on my system? - size of EdgeX services, infrastructure, etc. and hardware/platform requirements
  - 2) What is the speed of data through the system? - from device service sensor data ingestion to the rules engine and back down through command to another device service to trigger a put command, how long does this take?
  - 3) How many "things" can be processed at a time? – with caveats on the type of thing, type of data, etc.
  - These questions need to be answered on real hardware (both Intel and ARM)
- On the issue of plugins…
  - The future of Go plugins is undetermined.
  - Andre is checking with members of the Go community/leadership to see if some definitive guidance on plugins can be determined.
  - Dell is taking the lead to explore Hashicorp's plugin mechanism – is there something we can leverage/borrow
  - Until more information on plugins is determined, the use of Go modules & interfacing will be used to isolate / decouple "providers" or alternate implementations where possible.
  - Until the plugin issue is resolved, EdgeX will not adopt other providers for infrastructure or other needs.
  - For example, EdgeX will not adopt other database providers such as ObjectBox
  - It is the goal of EdgeX to have a single implementation of all services and infrastructure elements (database, messaging, etc.), but until the current plugin/module issue is better resolved, EdgeX will carry on with the current implementations and work to try not to adopt new ones.

Architect’s Day Decisions (part 2)

- On Store & Forward…
  - EdgeX has the “store” part of store and forward already.
  - It needs the forward part for:
    - Disconnected modes and the ability to send data to an endpoint when a connection is re-established. There should be two modes to disconnected mode operations. One – forward all data since the disconnection occurred. Two – forward a select slide of the data using some form of query filter (such as forward non-redundant readings).
    - Bach modes – rather than send all the data all the time, collect the data and send the data in a batch at designated intervals.
  - Forward operations will be accomplished by application services in the Fuji release.
- Regarding the Rules Engine
  - There is still a strong desire to replace the Java/Drools rules engine with something lighter weight, yet easy to operate.
  - There are a number of poorly used/unmaintained Go options (outlined in a Github issue)
  - Andre suggested looking into Json-Logic (see Andre GIthub)
  - IoTech has been using Node Red in place of Drools
  - Andy Foster from IoTech demonstrated IoTech XRT and it’s use of a Lua scripting engine
  - During the Fuji release we will do some exploration of options.
  - Until a suitable replacement is found, the Java/Drools option will remain in place for Fuji.
  - We will explore the JVM used with the rules engine service and upgrade it if the current JVM is out of support.
- On command information being made available to the north-side
  - While external applications may desire more command information and some conveniences/syntax sugar could be supplied, at this time it is felt we don’t have enough use case guidance or needs to make changes.
  - In the future, the application services may supply more command information to the north.
  - In the future, the command API, data structures representing commands, caching of commands for quicker/easier access and other refinements could be made.

Architect’s Day Decisions (part 3)

- Regarding Application services and export services
  - With the Fuji release, the application services should be 100% functional replacements for export services (client and distro)
  - An earlier decision not to support cloud IoT platforms directly was revisited. The feeling was that some examples at least needed to be provided to make connecting to cloud providers easier.
  - Application services will now provide example services that get data to “the big 3” of Amazon IoT, Azure IoT and Google IoT Core – starting with Amazon and Azure for the Fuji release. Adding support for Google or Chinese cloud providers (Tencent, Alibab, etc.) is a stretch goal.
- EdgeX will move from RAML to Swagger for API documentation during the Fuji Release
  - Swagger provides better tooling
  - Swagger is an open standard now – OpenAPI
  - Swagger is more popular in development circles
  - RAML documents will be archived after Fuji
- Regarding Device Service changes…
  - Black box tests (and test plan) are a necessity and priority work
  - There is a need to deregister devices/device services – to be done for Fuji
  - While there is a need for dynamic discovery scaffolding in the SDKs, more requirements documentation and design are needed first and will be done for Fuji
  - Readings need to be cached in order to:
    - 1) avoid hitting the device unnecessarily on get requests
    - 2) avoid sending data to core data superfluously
  - Change device services (and core services) so that value descriptors can only be created from metadata/core (protect responsibilities)
  - Use nanoseconds for all Event/Reading timestamps (a change from milliseconds and to be altered in Core/Metadata too).

Architect Day Decision Summary

- Fuji will likely be minor release and targeted for end of October 2019
- The focus of the Fuji release is to “harden” the services, improve testing, and work to understand more of its performance characteristics
- We will work to capture service performance metrics in Fuji
- No movement on plugins this release, but we will research the state of plugins and possible options
- We will implement “forward” capability in Fuji through application services
- The Java/Drools engine will remain, but we’ll look hard for alternatives
- We’ll collect more use cases/needs before facilitating commands to the north-side applications
- Application services will be export service equivalents in Fuji & we’ll provide at least 2 cloud connector examples (Azure IoT and Amazon IoT)
- We’ll move from RAML to Swagger
- There will be a number of changes to the DS SDK (black box tests as a priority, clean up and additional support)

Business Meeting Outcomes

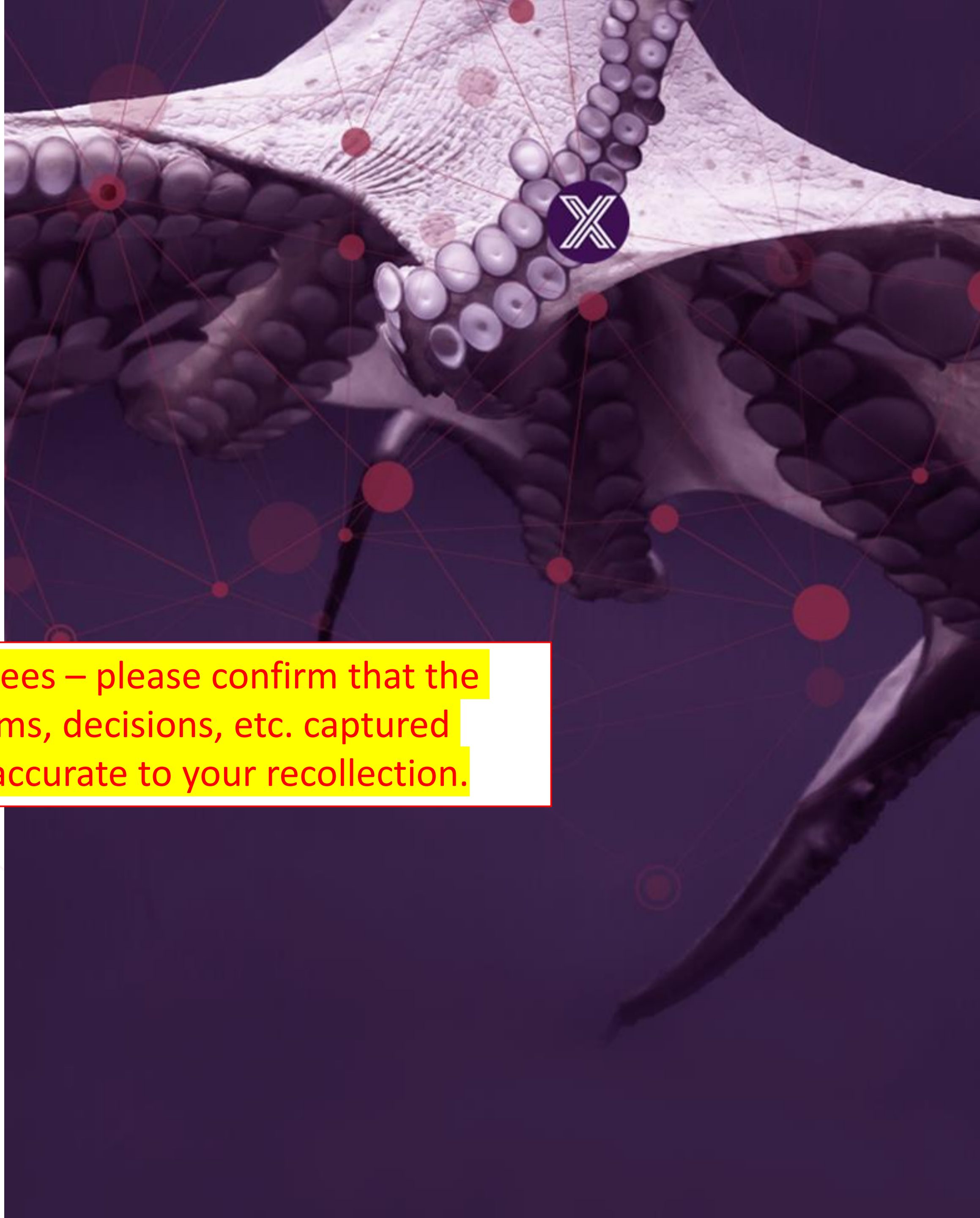
Keith, Randy and Brad – please confirm summary or provide your comments/content

- EdgeX will reconstitute it's marketing working group
  - Serve to meet EdgeX marketing needs (event planning, promotional material, etc.) – Keith to lead
  - Serve to provide LF Edge Marketing Group with EdgeX feedback and insure EdgeX marketing needs are satisfied
- Commerce project group
  - Will continue to drive requirements around a board set of use cases (broader than just retail)
  - Working with retail and other ecosystem customers to understand needs and how to bring EdgeX into solutions (without trying to sell a framework)
  - Will work with TSC to drive requirements, gap analysis, and architectural input
- Certification Working Group
  - Decision to work toward self-assessment availability by EOD 2019
    - Working toward device service self-assessment first
    - Requires device service black box tests
  - Re-brand this "EdgeX Ready" when used externally
  - Decision to work toward certification in future releases (required too much infrastructure and other work to make feasible for this release)

EdgeX Training

- EdgeX Introductory Tutorial (Hall)
  - Slides: PDF slides
  - Recording: zoom recording
- Getting started as an EdgeX developer (Conn)
  - No slides
  - Recording: zoom recording
- Building Application Services using the SDK (Johanson & Goodell)
  - Slides: PDF slides
  - Recording: zoom recording
- Building Device Services using the SDK (Tony/Toby/Cloud)
  - Slides: coming soon
  - Recording: zoom recording

EDGE X FOUNDRY™



General Meetings

All attendees – please confirm that the action items, decisions, etc. captured here are accurate to your recollection.

Fuji Scoping Decisions

Strike Team Reports

- CBOR/binary support - Toby on point with Trevor and Janko to assist with core, supporting and export services changes
  - Will the C SDK be updated to support CBOR – need Steve to provide feedback
- Redis implementations of DB services - Andre & Trevor
  - Docker and documentation work remaining
- Initial performance (and load) testing infrastructure and results – Andy
  - Move repo to main org – James handling
- LF support for TIG stack
  - Now done
- Certification Program - Randy and Jim
- SDK Unit testing (to the point short of any major refactoring) – Steve
  - Not a show stopper but not as much coverage as hoped
- SIGL Static Analysis - James (with LF support)
- Kong Upgrade - Tingyu
- Device services (Edinburgh public availability of Modbus, MQTT, BACNet, BLE, Virtual Device, SNMP) - Steve
  - Dell to provide SNMP DS
  - BLE will be open sourced and Jim working on getting it into holding repo
  - Also include device-opcua
- Security storing service secrets in Vault and protecting the master token - Jim and Tingyu with Intel resource assistance
- Documentation clean up - Michael with WG leads to nominate areas of need first and complete doc updates by deadline
  - Need more on DS and SDK docs – Toby & IoTech to help
  - Device Profile – being worked by Michael and Tony
- UI updates
  - IoTech UI - Keith
  - VMWare UI - Jim with Huaqiao
  - UIs not able to be updated will be archived to the Delhi release

Cadence Check

- April & Oct remain target release months – keep release for Oct but in future look at putting F2F about 1 month in advance of release to allow for some community coding opportunities
  - Fuji – Oct 2019
  - Geneva – April 2020
  - Hanoi release – Oct 2020
  - Ireland release – April 2020
- F2F planning around time of completion of each release
  - Chandler –Nov 2019
  - ??? – April 2020 (venue to be selected – nominations??)
- Conferences – pass to new marketing committee for consideration
  - At least 2 x large marketing/promotional events (Hannover Messe, IoT SWC)
  - At least 1 x developer focused event
    - Internet of Things World – May
  - Feeling is that perhaps to IIOT focused and presenting to the same group in both
  - May need a 2nd dev focused event – but which one

Fuji General Scope

- Only 3-4 month cycle
  - Targeting minor release for late October
- Major themes
  - ~~Support High Availability/Better Distribution~~ Move this to Geneva
  - ~~Understand~~ Take initial steps toward gaining a better understanding of performance characteristics
    - By Geneva, be able to answer the 3 performance questions:
      - 1) Will EdgeX fit on my system? - size of EdgeX services, infrastructure, etc. and hardware/platform requirements
      - 2) What is the speed of data through the system? - from device service sensor data ingestion to the rules engine and back down through command to another device service to trigger a put command, how long does this take?
      - 3) How many "things" can be processed at a time? – with caveats on the type of thing, type of data, etc.
    - These questions need to be answered on real hardware (both Intel and ARM)
  - Improved Security
  - ~~First Certifications~~ Offer first steps in the certification process
    - Provide self assessment by end of 2019
  - Application Services replace Export Services

EDGE X FOUNDRY™

Fuji – General (cross area)

In

- Move to Go ~~2.0~~ 1.12 (evaluate 1.13)
- ~~Upgrade to Mongo 4.0~~
  - ~~If Mongo is still a reference implementation~~
- ~~Apply “12 Factor Apps” to all services~~
- ~~Services can live on any platform~~
- ~~Services can have multiple instances~~
- Provide watchers/callbacks for config or data changes (stretch)
- Swagger ~~(in addition to RAML)~~ API documentation (replace RAML)
- Sponsor some hackathons (Beau F to investigate and lead efforts)

Out

- ARM 32 support
- Windows development support
  - 0MQ issue
- Apply “12 Factor Apps” to all services
- Services can live on any platform
- Services can have multiple instances
- Upgrade Mongo

Fuji – Core (and Supporting)

In

- Consul 1.4
- Blacklist/whitelist of devices
  - Or whatever we need to support auto device provisioning
- Store and Forward ability (Move to appl services)
  - When north side connectivity is lost
  - Design/document vs implementation?
- ~~High Availability~~
  - ~~Application of 12 factor principles~~
  - ~~Service orchestration~~
  - ~~For non-HA scenarios~~
  - ~~Instrumentation~~
  - ~~Design/document vs implementation?~~
- ~~Database persistence~~
  - ~~Reference Implementation Database Selection~~ (defer for now)
  - ~~Decoupling from specific providers~~
  - ~~Long v short term storage (e.g. core-metadata v. core-data)~~
  - ~~Migration when models change~~
- Refactoring needs (as we can)
  - Command refactoring (device profile v command service)

Out

- Data Filter between DS and Core Data
- Add SMS to notifications
- Logging service rework
  - Intel has developed a Logging/Telemetry as a Service – Replace/Augmenting service
- Support for alternate logging format
  - XML & CSV in addition to JSON
- Command improvements
  - Min/max limits
  - Security authorization option(s)
- High Availability
  - Application of 12 factor principles
  - Service orchestration
  - For non-HA scenarios
  - Instrumentation
  - Design/document vs implementation?

Fuji – Application WG

In

- Store and Forward ability
  - When north side connectivity is lost
  - To support batch mode and sending data on a schedule
  - Design/document vs implementation?
- Add Black box tests
- ~~Rules Engine Replacement~~ (stick with Drools for now and research options)
  - Explore OpenJDK support based on current JVM usage
- Next iteration of Application Services and App Functions SDK (“Export Services equivalence”)
  - App Service to send data to Rules Engine
  - ~~Message envelop changes~~
  - Added functions
    - ~~Filters: for value ranges, ???~~
    - Compression
    - Encryption
    - ~~Signing~~
    - Additional endpoints (HTTPS/MQTTS)
    - Cloud connector examples for Azure/AWS (others like Google IoT, Alibaba, Tencent, etc. stretch)
    - Add Vault integration so that secrets for above can be secured
  - ~~More dynamic topic configuration / addressing~~

Out

- Export commands
- Alternate message bus support
- Other functions (transforms, enriching, encryption, …???)
- Blockchain/digital ledger attribution
- Additional language SDKs
- Rules engine to trigger notifications

EDGE X FOUNDRY™

Fuji – DS & SDK

In

- Test plan and black box tests
  - Blackbox test for Device Services - Virtual, Modbus, MQTT, BACNet, OPC UA, etc.
  - Initially develop common set of tests that can be used against all Device Services
- SDK clean up and improvements as needed
- Improvements to the SDK (stretch)
  - Cache of readings
  - Device dynamic discovery
- Intel Camera device integration
  - Camera Discovery Components (SAF Bus)
- ~~Intel RFID device services~~

Out

- Support for mesh network protocols
- Support device hierarchy
- Tooling for SDK (CLI, JetBrains, or Eclipse plugins, etc.)
- Tooling for Device Profile creation
- Downsampling – DS throttle back on readings if nothing is changing
- Additional language SDKs

Fuji – System Management

In

- ~~Implementation of SMA as lightweight registry provider (when Consul not there)~~
  - ~~Know what services are available when registry not there~~
- Start/stop/restart all done by the executor to include stop/restart of SMA
  - Executor tracks completion and returns results
- Executor for metrics collection
- Setting configuration
- SMA Translation layer (stretch)
  - Pick one protocol to start (LWM2M)

Out

- Storing metrics collected locally
- Callbacks (alert on changes to config/metric) –
- SMA translations to other protocols
  - Redfish
  - OMADM
- Actuation based on metric change
  - “rules engine” for control plane data
- Consider use of QoS and blockchain to prioritize resource usage by certain services.
- SMA to store configuration – lightweight configuration store
  - In replacement of Consul or whatever provider we have for reg/config).
  - Discussed at recent sys mgmt. WG meeting. Bigger question is do we still want Consul.

EDGE X FOUNDRY™

Fuji - Security

In

- Generation of PKI
- Distribution of per service Vault secrets
- HW secure storage abstraction layer (design only)
    - How to protect the Vault Master Key
- ~~Secure Service to Service requirements and design~~
- Ensuring the services running are those expected (and authorized)
- Renew/refresh threat assessment
- Need document defining what security is/does and can/will do

Out

- Service to service security
- Implementation of service-to-service communications
- Securely providing service updates
- How to securely provision new devices/sensor
    - Device identification and authentication/authorization
- Hyperledger/blockchain/digital ledger integration
- Protect data at rest
    - In DB like Mongo
    - In log files
- Privacy concerns (HIP-A, GDRP, …)

Fuji – Test/QA/Documentation

In

- ~~Improved unit tests across services (improvements should be driven by specific WG dev teams)~~
- Improve blackbox test structure including reorganization of the tests and better test case documentation
- New test framework (e.g. Robot or Cucumber) to support additional types of functional/blackbox and system integration tests e.g. Device Service or system level latency tests
- ~~Blackbox tests for new Application Services microservices~~ Moved to Appl Services WG
- ~~Blackbox test for Device Services - Virtual, Modbus, MQTT, BACNet, OPC UA. Initially develop common set of tests that can be used against all Device Services.~~ Moved to DS WG
- Automated performance testing
  - API Load testing (measure response time) and metrics (CPU, memory) collection for all EdgeX microservices (this work was started during the Edinburgh iteration)
  - EdgeX microservice startup times
- ~~Test coverage analysis e.g. using tools such as Codecov.io~~ – moved to Dev Ops

Out

- Automated Performance Testing
  - Automated system level latency and throughout testing (e.g. device read to export or device read to analytics to device actuation) - STRETCH GOAL
  - The ability to create summary reports/dashboards of key EdgeX performance indicators with alerts if thresholds have been exceeded
  - Blackbox and performance test runs against other container technologies supported (e.g. snaps
  - Automated performance testing - baseline performance of service binaries no container
- Configuration testing
  - Existing testing uses a single static configuration
  - Need to identify and add additional testing configurations to automated blackbox testing
- Static code analysis
  - Using tools such as SonarQube or Coverity to identify badly written code, memory leaks and security vulnerabilities – STRETCH GOAL
- Documentation
  - Replacement of RAML API documentation with Swagger – STRETCH GOAL
- Tracing
  - During testing, configuration
  - Candidate tools/technology based on OpenTracing standard: Zipkin, Jaeger

Fuji Planning – DevOps

In

- Static code analysis tool identified and integrated into the EdgeX Jenkins Pipeline
    - Analysis of Docker /runtime artifacts, not the source code
- Code and artifact signing with semantic versioning
- Fix Documentation – edgex-go
- Build Performance Optimizations
    - Pipelines for EdgeX Foundry base build images
    - Basebuild images managed locally within Nexus
    - Leverage PyPi Proxy for local pip dependencies
    - ~~ARM builds – optimization leveraging different high CPU build nodes / OS (ARM team) –~~ Jim to handle
- Explore static code analysis
    - Like Codecov.io

Out

- Alternate deployment/orchestration
    - Beyond Docker/Snaps
    - Kubernetes
    - Kata Containers
    - …
- SonarQube – SonarCloud is already in play in the LF Decision: wait to see what codecov.io offers
- Suggestion to rename all of the Jenkins "arm" jobs so as to differentiate 32bit / 64bit architectures
- Full Pipeline transformation for EdgeX services

Fuji – Vertical Solutions

In

- ~~Additional project groups~~
  - ~~Oil/Gas~~
  - ~~Smart Factory~~
- Gap analysis by each project group
  - What is needed by the vertical that is not provided by EdgeX today?
- Proposed EdgeX roadmap additions
  - High level architectural needs/changes
  - High level designs
  - Technological suggestions/input
- Perform sync with Home Edge project
  - Determine how best to integrate

Out

Fuji - Certification

In

- Self Assessment process (Dec 2019)
  - Start with Device services, then application and core/supporting services
  - Have some sort of web page indicating those that have self certified

Out

- Service Certification

Developer Advocate Report & Action Items

- See Michael’s deck and summaries here: Dev Advocate Deck
- Action items (based on meeting between Michael and Jim later)
  - Update and improve the “offerings” page on the Web site (work with TSC and marketing groups to gather new info)
  - Improve the documentation – starting with these areas for Fuji
    - How to get started with Windows (VSCode, 0MQ install, etc.)
    - Common troubleshooting guide – how to pull the logs, how to decipher what’s in the log, how to check issues in docker, etc.
  - Get help from marketing on:
    - Decisions on events (which events to attend for developers and for community)
    - Help drive the vendor offering page
    - Help on document re-organization (how to improve to bounce rate)
    - Get travel budget for event support
    - Drive hackathon/training events

Architectural - Tech Debt Issues

- More unit & integration tests
  - Agreed that no coverage number or metric should be established, nor put any blocker to PR approval. However we should get the current coverage stats now and get them again at next F2F to see how things are progressing
- Release manager – we have a process, now what?
  - Agreed to find a release manager with upcoming elections or else lesson release manager rules on those stuck with the task
- Mongo & Redis – what should be our reference implementation database(s)
  - Keep for now; not accepting others like ObjectBox until plugin issue resolved
  - In general, we want to maintain just one reference implementation of everything.
- Traceability – we have a correlation id. What's next?
  - In the future (not Fuji), look at logging type (new log structure for structed logging)
  - Provide documentation on how to use the correlation ID and how to do-it-yourself traceability (perhaps show how to use with a common tool)
- Rules Engine replacement (the last of the Java)
  - Per architect's day – no change for Fuji but explore alternatives
- Environment support
  - Windows development & 0MQ (no change)
  - ARM 32 support (no change)
- Metadata needs to implement registration/callback mechanism for any metadata changes (ex: call me anytime the device changes)
- From Architect's day discussions/decision
  - Value descriptors creation should be the responsibility of core not device services
  - Events should be timestamped in nanoseconds

Architectural - Tech Debt Issues part 2

- Device Service improvements (from Architect’s day)
  - Clean up of device profile
  - Adding automatic discovery and provisioning (or at least a place in the SDK for DS creators to use to do automatic discovery and provisioning using blacklist/whitelist configuration). First – need requirements/design work to help define
  - Need device service/device deregistration
  - Cache of readings: to avoid hitting the device in some cases and to avoid sending data to core under some requests
  - All of these are behind priority of black box testing (and test plan) for Fuji
- Gain an appreciation of EdgeX performance and size characteristics (from Architect’s day)
  - What are the resource requirements for EdgeX
    - Answering the question: will it fit on my system?
  - What is the speed of an EdgeX transaction? I.e. how long does it take for a sensor reading to be ingested, perform analytic logic, and trigger actuation on a device (and be exported to the north side).
  - How many things can be attached to EdgeX at one time? How much data can be ingested and how often can it be sampled?
  - These questions must be answered on real platforms (both Intel and Arm) – not just on a virtual environment.
    - The results must also include information about the use case, data type, circumstances of execution
    - The performance tests/harness/tool description must be made available for others to be able to replicate and/or extend
- How do we deal with the rate of code change going forward?
  - Edinburgh is V1.0. We must beware of non-backward compatible changes
  - We must find a cadence that allows consumers to keep up – example, Redis developing core services with Redis that replace those of the Mongo implementation.
  - Another release manger issue.
  - Let’s try milestones like “no new functionality” freeze dates
- Go plugins – don’t appear to be on the Go roadmap (per Architect’s day)
  - We are checking with the Go community and groups like Hashicorp that is using Plugins – what is their vision and options thought
  - Because of plugins issue, we don’t really know how to effectively bring on other implementations (ex: ObjectBox) without many issues
- We will look to move from RAML to Swagger (aka OpenAPI)

New features, tools, processes, etc. Issues

- Need to handle the "forward" part of store and forward (from Architect's day)
  - "Disconnected or situational" Store and Forward - need to support (via application services) use case where the application services are not able to connect to and pass data to the desired endpoint (where EdgeX has lost connectivity to the north)
    - In this requirement, there are scenarios where all or a subset of unsent data is filtered and then sent when reconnected
  - "Batch mode or deliberate" Store and Forward – need to support (via application services) use where the application services is always connected to the endpoint, but the use case requires not sending events all the time, but batching the events up periodically and sending them in bulk to the north
- Documentation versioning
  - ~~Semantic versioning~~
  - Version with tool (Sphynix?)
  - Moving to separate repo
- Kubernetes (K8s and K3s) support
  - How do we facilitate under scale of solutions?
  - Not supporting any additions today, but welcome how-to-guides/blog posts, etc.

New features, tools, processes, etc. part 2

- Naming of Snaps and other deployment artifacts going forward
  - All artifacts (Docker or Snap) will be “edgex-<service name>”
  - Arm artifacts will contain the name “arm” to distinguish them from Intel artifacts (default)
- LF Edge
  - What are the projects and what is the potential impact
  - Jim provided overview – requests community stay aware and ready to provide assistance
- Application services and cloud providers (per architect’s day)
  - Change in philosophy -> provide example code for the big 3 (Amazon, Azure, Google) and then big Chinese providers.
  - In this release, add examples for Azure and Amazon
  - We also need to support MQTTS/HTTPS
  - Will require application services (and SDK) to use be able to use Vault for certs, tokens, etc.

Deployment Artifact discussion – time permitting

- Today
  - We produce many services with ***a*** reference implementation deployment artifact for each (offering two different types: Docker and Snaps)
- Tomorrow
  - Might we want to create deployment artifacts that contain multiple services?
    - All of core services running in a single container/snap – simplifying deployment under circumstances/use cases
  - Might we want to make/build some type of combined service?
    - All core services in a single executable – for example – in order to improve performance, lower footprint, etc.
  - Community was ok with exploration of options by Dell and report at next F2F.

Meeting Lessons Learned

- Architect's day worked better as invite only
  - Provide early warning that architect's day is by invite only to avoid extra travel day for some
  - Open architect's day to "contributors" and define contributor in advance
- Ideally, we want to offer a hackathon/plugfest day for EdgeX developers to address high priority issues / needs in advance of a release
  - For Geneva – look at adding special day for this
  - For Geneva – look at pushing release until 3-4 weeks after the F2F to assist with this
- Training was a good thing
  - Offer more hands-on exercises to the training
  - Provide the trainers more time to prepare
  - Schedule so that the "newbie" training can be attended by business folks
- Business day was valuable – just needs to be aligned with advanced training day
- We need to keep the meeting to 4 days if possible (5th day optional for plugfest)
- Use a Google poll to get more feedback