Included as a given:

** Refactor of API requests/responses to avoid "data leakage"

1.) Major version rev
   a. Previous goal was to build major version longevity into V2
   b. Re-assess and determine if still applicable.
      i. If so, we have to be clear about what we include for V2
         1. Big ticket items -- transport and security
         2. Positioning for the future, backward compatibility
            a. removal of Support-logging
            b. repurpose Core-data --> Support-data
      ii. If not, we have more flexibility
   c. I do have a concern that whatever our level of planning at this time, since we are not considering HA as a goal, HA support will force us to rev major version.

2.) Transport agnosticity
a. Included as part of Dell’s V2 proposal due to community input RE: alternative architectural footprints (pub/sub from DS to AppSvcs)
   i. DTOs used by REST endpoints should be the same as those used via pub/sub (just like today’s ZMQ bus from Core-Data to AppSvcs)
b. Dell extended this proposal to include all operations supported by the EdgeX platform, not just sensor event ingestion
c. If we limit our scope for V2 to HTTP/REST, could limited pub/sub be added without rev’ing major version?

3.) Integrated security solution from the beginning of V2
a. Such as identity mgmt, auth/auth, encryption
   i. Bryon's full V2 proposal --> https://docs.google.com/presentation/d/1S56M5-kSFDOdNOT0IELHDdhuy-JbNOdgoDPwfx-Mi8c/bm
b. If transport agnosticity (#2 above) is optional, what effect does limiting V2 transport to HTTP/REST have on this proposal?
   i. Elimination of /batch endpoint for instance, only support use-case endpoints
   ii. Would later introduction of limited pub/sub cause a major version rev?
   iii. Should we still consider all of the following as requirements applied to HTTP/REST which can then be carried over to pub/sub?
      1. https://docs.google.com/presentation/d/1S56M5-kSFDOdNOT0IELHDdhuy-JbNOdgoDPwfx-Mi8c/edit#slide=id.g7d50bde1ce_0_45

Referring to the bullet above see this slide

Question from Tony referred to whether GET REST API endpoints would be secured by Option #2 on that slide. Bryon’s prelim evaluation is “YES”

**

Proposal: There is some flexibility w/r/t V2 longevity – could leave V2 as “experimental” for some time.

Implement OpenAPI V2 REST/HTTP endpoints using defined DTOs

** Request DTOs for GET/DELETE do not exist for REST, would these be needed for incremental pub/sub use cases?

V2 Implementation will be beta until further notice, must co-exist and not impede V1

** Awareness of separate persistence, previous effort had decided we would have separate V1/V2 persistence. This needs to still carry forward b/c state models will most likely be different.

Add incremental pub/sub going forward as use cases require

Utilize Option 2 from security slide for both.

Deck to be updated for Security WG review (4-Mar-2020)

Ability to enable/disable security

**
4.) Unified base implementation across all platform sub-domains (Device Service, Core, AppServices)
   a. Sounds like this is desirable to the community
      i. It sounds to me like the depth of this common implement needs to be further articulated
      ii. Two choices
         1. Common Impl, everybody use it
         2. Impl for Core only, folks review and decide if they want to use it.
      iii. Jim wants devs who come to the platform to see similarity in API and service wireup.
   b. Does this necessitate DS / AppServices to define their own V2 API spec?
      i. Should they do this before we start implementation so that we have a full sense of the interactions in the system?
      ii. Device Service API review toward principles in V2 ADR
   c. Is there any benefit to this from a certification / testing standpoint?