2023-05-22 Architecture WG Meeting Notes

Date

22 May 2023

ZOOM Meeting Information:

Monday, May 22, 2023, at 11:30am PT/2:30pm ET.

Join Zoom Meeting

https://zoom.us/j/7904999331

Meeting ID: 790 499 9331

> ## Antitrust Policy Notice
>
> Linux Foundation meetings involve participation by industry competitors, and it is the intention of the Linux Foundation to conduct all of its activities in accordance with applicable antitrust and competition laws. It is therefore extremely important that attendees adhere to meeting agendas, and be aware of, and not participate in, any activities that are prohibited under applicable US state, federal or foreign antitrust and competition laws.
>
> Examples of types of actions that are prohibited at Linux Foundation meetings and in connection with Linux Foundation activities are described in the Linux Foundation Antitrust Policy available at http://www.linuxfoundation.org/antitrust-policy. If you have questions about these matters, please contact your company counsel, or if you are a member of the Linux Foundation, feel free to contact Andrew Updegrove of the firm of Gesmer Updegrove LLP, which provides legal counsel to the Linux Foundation.
>
> openIDL

Attendees:

- Sean Bohan (openIDL)
- Mason Wagoner (AAIS)
- Ken Sayers (AAIS)
- Faheem Zakaria (Hanover)
- Dale Harris (Travelers)
- Peter Antley (AAIS)
- Brian Hoffman (Travelers)
- Joseph Nibert (AAIS)
- Brian Mills (AAIS)
- David Reale (Travelers)
- Tsvetan Georgiev (Senofi)
- Ash Naik (AAIS)
- Satish Kasala (Hartford)

Agenda:

- NO AWG NEXT WEEK (Cancelled Mon 5/29)
- Update on openIDL Testnet/Workshops (SeanB)
    - openIDL NodeBuilder Workshop 2023
- IWG update (YankoZ)
    - This Thurs: Maintainer Access

- Update on RRDMWG and internal Stat Reporting with openIDL (Peter Antley)
    - IaC and UI framework.
- Architecture Decisions for OLGA
- AOB:

Notes:

- RRDMW and Stat reporting
    - OLGA
    - Stat Reporting
    - ingest homeowners, stat records from residential property lines
    - Weds - order of operations, what lines next?
    - confirm what they are doing for Jumne
- Architectural Decisions for OLGA
    - HDS
    - native AWS
    - react web page, single page app, REST API
    - node based lambda functions
    - simple queue service
    - S3 buckets
    - at this time - utilizing TOAST UI (grid component to display table data)
    - elected to choose TOAST as toast makes most sense, other options for GRID components?
    - chose toast - grid component that didn't have any weird business tiers
    - most grid vendors give away for free, adv things charge for
    - TOAST UI - POS for restaurants
    - open source grid component - filtering and sorting
    - KS - gen discussion
        - know this is a ref implementation
        - any choices made could be changed
        - focusing on AWS, not azure
        - AWS is licensed thing, paid for services
        - thought important - open source code we publish to github as much as possible should not be dependent on purch or licensed software
        - code shouldn't have dependencies for things we need to license
        - bigger carriers - check everything as a dependency, bring in manually into our environ, trying to figure out how to decide whats ok and no
        - license/not licensed? pre-approval?
        - TOAST first place this came up
        - TOast, AG Grid, MuyTable
        - keep it from getting complicated, putting on license/non license
        - red herring? wrong place?
        - some decisions may run afoul
        - open source vs licensed but meet the need
        - should we always choose open source one
    - DR
        - policy vs practice
        - usually even open source or permissive licenses, enterprise adopts when some party bundles into a SaaS offering (agreeement, click thru, rest handled by someone else)
        - on internal projects, ok to do license wrangling, usually more common one agreement with primary partner
    - KS
        - from carrier perspective, not sure if it applies to ref implementation, not a SaaS
        - not running it for you, you are running in your enviroment
    - DR
        - most likely, 3rd party provider, abstract complexity
        - structure, buying a service from provider, agreements transparent, well defined, shift licensing burden to someone else
        - see it a lot with F/OSS projects, enterprise versions, spring to handle tech implementation as a service and abstract licensing
        - something considered
        - one more thing a carrier has to think about - not trivial
    - KS
        - we have 2 companies, TRV, Hartford - diff opinions
        - Hartford - run it ourselves, TRV - SaaS it
        - stay with less licensed software and less controversial security stuff
    - DR - diff between open sourced and licensed?
    - KS - pay money for, pay the vendor
        - when doing ref implementation, "dependency you have to pay for"
    - DR - few licenses are the gold standard, dont want to play in that game unless have to
        - network effects? get as many on as possible, out of the box doc - explains packages, licences, make avail more explicitly than you would do manually
        - might want to make it very explicit - here is the list of every license in this stack, their permissions and conditions
    - KS - licensing and purchasing are diff issues
    - DR
        - non trivial, not insurmountable, lay out the licenses and what they mean and what to do - go a long way in helping
    - KS
    - SK
        - open to both models (build vs SaaS)
        - open to subscription model as well (AAIS runs infra and services that go with that)
    - KS - other grid components? what do all use? React or Angular grid components (needs to work on React for sure)
        - TOAST.ui - leaning towards, functionally complete, less of a drag, checking license, possibly MIT license

        - AG Grid (pro license)
        - MUI Table (pro license)
        - "Our VPC or Vendor's VPC"
    - DR - if model requires anyone to have multiple licenses paid for and managed? problems
        - all permissive and open source one thing
        - a lot of vendor relationships to manage is another thing
        - 13 licenses not ideal, harder if some subscript based
        - contracts and licenses and vendor agreements - less attractive
    - PA
        - paid will be unattractive
        - acceptable features
        - MUI Table - could be a fit
        - does grid component work well with ORM + REST
        - done a lot of doodling, ready to start doing POCs
        - getting working code pulled together
        - this iteration, goal is to load stat data to HDS w/o validations
        - JN - working on serverless implementation of rules engine and validations, modernized rules engine, getting it to run serverless, after he gets rules engine working work on stubbing out react pages and throw up grids THEN peter will do API stuff, end of June test UI to display and show
        - Serverless implementation -do IAC using SAM template, and use cloud formation for one persisitent server
        - objections?
    - KS
        - IAC (infrastructure as code easier as AWS native vs Terraform)
        - dont have much exp with Terraform inside of AAIS
        - cloud formation is simple in comparison, dont want to overburden with extra work
        - Serverless Architectural Model - declarative vs descriptive
        - SAM for Olga, not in general
        - another way to handle this - carrier participate - more complicated - participate in detail, maybe terraform brought in easier
        - right now, terraform affect timelines
    - PA - some prebuilt tools in AWS to implement SAM
    - KS - bad idea? think about?

Time	Item	Who	Notes

Documentation:

Notes: (Notes taken live in Requirements document)

Recording: