20200513-Virtual-Technical-Summit-AASIG-VehicleAPI-VHAL-Workshop-Minutes

1	**## Introduction**
2	Philippe Robin welcomed everyone and introduces the agenda and his expectations 74 attendees Alexander Domin as a moderator describes the agenda (slide #2) slide deck for this workshop is here
3	**## Project overview & proof-of-concept demo**
4	Architecture on slide #3 shows Android Head Unit + other ECUs that can use other operating systems. On Android: App represents the UI and provides a representation of data gathered from Data Server Alexander explains the head unit layers. App with Manifest containing permissions (concept). App is implementing UI and business logic. Rely on data from boardnet. We introduce a data server (GraphQL server) that will connect to vehicle network (e.g. SOME IP, LIN, etc.). Data server will translate OEM data from network to VSS specification compliant data - this is specifically VSS Feeder component. Presented components are logical components - not necessarily separate entities. Authentication service protects data from unwanted access - generate tokens (JSON Web | Tokens) for application - it needs to file requests to access data. Package Manager component is an AOSP component - it serves information about application permissions granted during app installation. Based on those permissions data server will be able to determine whether to grant access to data or not.
5	Alexander presents the work split (slide #4) based on work packages identified during last February f2f meeting.Feeder is about connecting data server to network. Runtime configuration. Deployment describing mapping between CAN, SomeIP, signal domain to VSS standard. Plan is to work on helping developers to make this glue logic. Being a current contributor does not mean he/she owns the work item completely- contribution on all topics are welcome ! Alex requests for contributors in particular for the following work items • VSS database concept creation • app creation using GraphQL to access vehicle data
6	Q: Does VSS server provide Authorization to vehicle data access, similar to standard Android permission model? A: Yes, similar to standard Android model, certain permissions (added for VSS access) can be granted to app, only difference is that the enforcement of permissions are on graphql server side. Q: Did you consider passing the VSS information to the Vehicle HAL from Google? A: We will answer this later during presentation Q: how would it be possible to have VSS database running on basic ECUs? are you considering some kind of powerful ECU like a central unit communicating with small ECUs? A: Not decided yet, good question, no plans related to HW, rather investigating APIs and needed operations to achieve this, depends on translation effort from vendor-specific standards to VSS. Nothing specific in mind when comes to ECU power. • VSS Database: storage and caching - need low latency, we need to investigate what kind of storage suits here. • VSS data server - more complex than key - value. More complex queries will be allowed with this API. Obeys authentication and authorization mechanisms. • Authentication Service - who and which kind of data can access. • typesafe API framework for interacting with data server. Note: other useful link - Vehicle Data Model Overview & Gap Analysis including VSS is there

7	Stefan explains the demo and then shows it: • `vss-feeder` module means `VSS Feeder` component on the diagram (slide 3) • `vss-graphql` means `Apollo GrapQL` component on the diagram (slide 3) • `db` which is json database (for now) and means `VSS Database` component on the diagram (slide 3) • Project setup on Docker • OpenDS simulator used for providing Vehicle data • Vss-Feeder connects to simulator, retrieves the properties and save it in database • Apollo GraphQL Server enables to play around with GraphQL query • Able to query data from car by GraphQL query based on schema generated from VSS standard Stefan details the relationship between OpenDS and VSS feeder - it's our input. Later on we are translating data from OpenDS to VSS standard. Apollo GraphQL is responsible for resolving queries from apps. It asks requested values from database server. Data-feeder and data-server are separate binary running.Apollo brings some playground that can act as test app. Can be accessed as localhost:4000 and allows to write queries directly. VSS is structuring values in tree structure. • Example: current gear: vehicle->drivetrain->transmission->gear. Apollo playground gives you some hints about available leafs and types provided. Structuring is an advantage over standard Android solution. Queries can contain requests for many values from different branches. One can see that values are read in runtime from simulator (on playground). You can subscribe for changes. Values can be monitored without polling - advantage over Android model. Alexander jumps in and explains VSS is a standard specified in YAML files available on GitHub. It is a JSON like specification - this file can be translated to GraphQL schema (showing schema for GraphQL). Explains how VSS schema translates to GraphQL schema.

8	Code repos
	
	Authentication Service implementation
	
	• Sources: https://github.com/stefanwysocki/aasig_dev_platform/tree/develop/vendor/genivi/modules/VssAuthenticationService
	
	External Data Server implementation
	
	• **Data server:** Publishing of GraphQL comprehensive example done
	  • Sources: https://github.com/GENIVI/vss-graphql
	• **VSS feeder:** Basic functionality is working
	  • Sources: https://github.com/GENIVI/vss-feeder

9	Q: are you working with low level APIs such as VSOMEIP in order to get the data ? A: this might be out of scope of this standardization group, however that's the plan. This simulator is for non production. Explains about various possible feeders that can be developed for many solutions even OEM custom ones. A: Alexander: wonders about creating guidelines and mapping rules between data mediums (CAN, LIN...) and VSS. Q: I think Franca is kind of Model tool that we can use for translate those interfaces A: Alexander - what we identified during f2f and in Detroit - additional architecture approach - showing additional VHAL communicating with Apollo GraphQL. In genivi we can provide component (VHAL) basing on AOSP VHAL HIDL and GraphQL interface basing on VSS. A: it is valid to have both (franca as describing the interfaces: methods+data, and VSS as structure of data) Q: can the Google boxes (green) be replaced by another OS, e.g. Apple ... ? the question can be: is the concept only valid for Google? A: Alexander - I guess could be similar. Left side could be reusable across platforms (Virtual ECU containing data server). A: Gunnar - Right side green boxes is something that Google requires and is not valid for other systems. However overall approach (data server with VSS standard with GraphQL interface) is not bounded to Android only. followed by short discussion Q. the question could be - why not considering 100% open source not linked to Android, which is not 100% open A. Android is 100& OSS, Google addons are not (and BSP addons can be closed source)
10	## **Topics discussion**
11	### **Google Vehicle Properties Implementation based on GraphQL Service.**

12	Alexanders shows slide #6 to extend the view of current project status and get a common understanding of the GENIVI Google VHAL architectural concept. • the idea is to provide support for applications using already established CarAPI from Google • and to write a generic component (Vehicle HAL) to be reused since GraphQL is standardized and HIDL is standardized Q: Whether 3rd party applications will be allowed to access vehicle data from VSS? A: Depends on OEM if it will grant permissions to 3rd party apps A: Alexander: it's up to OEMs. We can see app evolution on mobile phones. If we provide APIsto these guys, we can expect maybe not as much but still a good amount of innovation. We need to create permission groups for internal devs (more attributes) and ext devs (less). Permissions model is the key and afterward it's up to OEMs what to do next. A: Gunnar: This project is about helping automotive industry to be successful with Android that is actually demanded and we are focusing our current solution on AOSP

13	Alexander: Getting back to architecture (VHAL included), presents the current challenges:
	
	• mapping between VSS and AOSP VHAL spec - where to store it?
	• inconsistencies between standards (unit conversion like in tank capacity: VSS in l or %, AOSP in ml)
	  • gaps to be identified and proceed with actions
	  • maybe we should convince Google to change something and look at the experience bundled in VSS contributions (OEMs already discussed the VSS format)
	• authentication - reminds about the concept of querying Package Manager about app permissions by Authentication Service. How to create tokens for apps using Car API - investigation to be conducted ?
	• create VHAL component - PoC and further on, should there be a reference implementation or what ?

14	Q: Is it viable to provide a kind of VssAuthentication service library/.jar to Application layer and reduce changes at framework layer ? A: Stefan: we discussed similar. Bundling authentication library with apps can be challenging. A: Alexander: we need to protect secret, thus we would like to put it into framework layer. We need this kind of discussions in future. Q: as a corollary to the previous question, would it be feasible to encapsulate the communication from the app to both the authentication service and the graphql server within a contentprovider ? A: Already discussed, we are also considering this kind of approach, it is covered in the "Internal Data Server Architecture" concept which is an option for the software architectural design
15	### **Permission groups specification.**
16	Alexander: Deeper catch on security (slide #8). We need to protect our data. Stefan was looking in concept of permission model of Android. We discussed with Gunnar and Stefan that we would like to have similar perm model in Manifest with different prefix: (e. g. *vss.permission.AMBIENTLIGHTING_READ*). This what Package Manager is working with and storing. We don't have group of perms for VSS leafs. We need to find out syntax to specify this, where to store spec and how to handle. What I was trying to do: Apollo GraphQL directives - processed when data is served. Alexander: explains the example for ambient light in GraphQL schema and how directives can be used to check perms from token. vspec file is YAML file with VSS specification. Here comes VSS layers (depl files ?). Combining vspec file and depl file we can get GraphQL schema with perms directives. Gunnar: You find easier to apply perm for signal instead of perms to signal. But your depl file is a perfect example of VSS layers.Alexander: as Stefan pointed all the groups below the leaf. That's probably not best approach. Gunnar: this can be solved with tooling but we should not base on that and think about just text editor as first point of act.

17	Q: What is the advantage of having both VSS & Android VHAL solutions in parallel ? If we have Android VHAL anyway Apps can access it. A: Both ideas has been already considered and discussed (see "internal data server" and "custom HAL") A: Gunnar: VSS is something which aims at being as wide as it is possible (other ECUs, cloud connected services outside of the car). In VSS there are thousands of definition, Android has only a minimal subset of those values. Google could extend its list but we doubt to be full VSS spec. VSS could be better suited to industry demands, seems to be better alternative or a useful complement to Android. Q: Rather than building an adapter for a VSS feed into the AA VHAL (which hides VSS), have you considered building an alternative VHAL for AA (that does not have to be compatible with the existing AA VHAL) that uses VSS for the native data model? A: Gunnar: please look at slide #17(Google VHAL + OEM Extensions inside) - We went forward with the assumption that companies would like to run Google Services on Android and would like to certify their solutions. Others maybe won't be compliant but we would like to be generic and compliant with Google's rules. This is why we need to implement AA VHAL. Q: you need to expose VSS and not just use it as an intermediate language. A: Alexander shows slide #17 - Access via Customized HAL - Google VHAL + OEM Extensions - which is an early architectural proposal that contains a service exposing VSS. Alexander: back to permission groups, Alex shows on slide #11 the seq diagram -app -> getToken() -> Authentication Server -> getPackageInfo -> Package Manager -> read manifest -> Manifest file. App gets token basing on their perm and needs to pass this token to data server (Apollo GraphQL) with a query. Data server will then validate the query based on perm model from VSS layer (.depl file). Slide #13 shows a different approach : instead of .depl file, permission groups file (json) contains the list of signals below perms. JSON file will be parsed by Authentication service.
18	### **Translation of permission groups.**
19	Alex: there are two options: 1. permission groups are resolved inside the server 2. permission groups are translated into allowed attributes in the Authentication Service on Android Discussion about token handling Gunnar: change in groups description shouldn't imply change of behavior Alexander: challenge with writing those spec files. I prefers permission groups in json files. Gunnar: why not YAML? Alexander: there is no library for YAML on AOSP, there is for JSON. Do we force YAML implementation on platform? Gunnar: or do we create a compilation step for translating permission groups. Alexander: we can switch. Gunnar: about change of behavior on sequence chart - we can take a look at it other time.
20	### **JWT Token - what will be included and how it will be done? And what about the generation process ?**
21	JWT might use asymmetric encryption to resolve the problem with sharing the secret between both parties Discussion about shared secret, asymetric signing with private key, token time out, one time usage.
22	## **Technical readiness level assessment and discussion on how and when to reach out to Google**
23	Alexander describes the following 2 points that could be discussed with Google • VSS as an alternative or complement to Google Vehicle Properties • Access to vehicle data via GraphQL instead of Key Value pairs
24	## **Wrap up**
25	Meeting ends with feedback gathering + request for contribution Stephen: the GENIVI board farm could be used to run the AASIG Vehicle Data / VHAL proof-of-concept Reminder about weekly calls