

Hypervisors in the vehicle: A standards-based approach to virtualization

A report based on the webinar by Automotive World and GENIVI Alliance.



Hypervisors in the vehicle: A standards-based approach to virtualization

Tero Salminen, Senior Director Product, Virtualization at OpenSynergy
Gunnar Andersson, Technical Lead at GENIVI Alliance

Automotive World Webinar | 13 February 2020



The webinar which was broadcast in February of 2020 is still available for download – see [1]. It however had some poor audio quality in some parts. A better recording is now available at [1] but some may still prefer to get this information in written form.

The document is not an exact transcript of the webinar, but it is close. The content and message is the same, and the information is presented here together with the corresponding presentation slides from the webinar, with the following adjustments:

- Common spoken expressions have been exchanged for more typical written language to reach more typical written quality.
- Minor clarifications and some lost details have been added.
- A few chapters are extended and elaborate on details that did not fit within the time of the webinar
- For the Q&A session, more thoughtful and complete answers to given questions can be given, now that there is time to formulate the answers.

Also, the section presented by Tero Salminen from OpenSynergy has a good quality audio in the recording, and is not included in text form in this document.



The first part of the presentation was done by Gunnar Andersson (GENIVI Technical Lead), who is also the author of this text.

We promised to address a question in this webinar : “Is there a need for a more standards-based approach for deploying hypervisors in the vehicle?”

- to ensure that automotive requirements are met,
- to reduce risk and concerns among adopters of virtualization,
- to promote portability,
- and to minimize system integration effort.

This report includes what we have already done toward this end, and leave the question a open for you to think about. After hearing/reading this and the results that we published in a first version, if at the end of this you believe the answer is Yes, then we are really looking forward to your support and participation. If anyone thinks no, then we look forward to your thoughts on this as well.

Today's Outline



- **GENIVI Alliance** projects, then and now
- **OpenSynergy** introduction and reasons for the standards approach
- **The Hypervisor Project**
- **Why Virtualization?**
- **Challenges → Solutions**
 - Can we do virtualization better, with the help of standards and shared requirements?
- The Spec --> **Areas of standardization**
- Virtual **Platform?**
- What about **VIRTIO?**
- **The Approach:** Avoiding standards proliferation
- **Areas of Standardization** – content
- **Future Outlook**
- **Q&A**

Today's Outline

First, to explain how and why GENIVI is addressing these questions I will give a little overview of GENIVI, how we think and an update on what GENIVI is today, because some of you might still need that.

Then we will hear from OpenSynergy to hear their reflections on virtualization and reasons for companies to support this standards effort .

The specification we are highlighting today was announced in its first release in combination with the CES 2020 conference. I'd like to explain how we work in the associated project that produces the specification.

Then a brief discussion of why virtualization is needed in the first place. Often presentation will go into a now relatively tired ramble about how the industry is changing, the need for consolidation of ECUs, and mixed criticality. This is a message we assume the audience has heard many times over. The idea of "why virtualization" is rather to note that it is a question that could be analyzed in detail, and mention some separate work trying to answer that question in a serious way.

Then moving on to Challenges and solutions. This is where we intend to make the core argument for how standards might make our situation better. What are the issues today, and how could we address them together?

Then, if you haven't yet looked at the specification itself, there is a basic explanation of content here.

Why is it called a platform and what is a platform? This might be obvious, but definition of terms is a core activity in all our projects, and since there is likely a mixed audience, if you first need to understand what virtualization is, some of this presentation of the different parts of a virtualized system might help.

What about VIRTIO? We have a clear direction and statement to make there. Primarily, the rationale for an automotive specification, that complements VIRTIO.

Further discussion of the approach, i.e. why and how to write a new specification. Why is a new specification not unnecessary or overlapping work, and how do we want to structure this work?

Further deep dive into the content, assuming we have time left, and a Future outlook.
At the end, time for some Q&A.

Slide: Comic

Since this is a heavy technical subject, the presentation was started on a lighter note with a comic showing two individuals, seemingly from the stone-age, each working on half of a wheel . The idea is proposed by one of them that that **collaboration** might be a good idea?

The comic strip was used in the webinar presentation with permission from its copyright holder, and it can be viewed in the recording.

Honestly, there have been times when I looked at the automotive industry and thought about something like those confused ancient wheel-building people. I imagine them being oblivious to each other's similar activities before they take interest in the commonality of the needs and have the epiphany that collaboration might help! Our industry sometimes seem to fail to understand that the really interesting work begins when everyone already has the basic fundamental building blocks in place, like the wheel in the analogy.

Let's therefore put the virtualization work in context with a short review of the GENIVI Alliance of today.

GENIVI now has over 10 years of experience in supporting standardization and shared software solutions in our automotive industry, and this is the key: collaboration.

We still believe that collaboration, on multiple different levels has the potential to make the automotive industry much more effective, and it is frankly likely to be a necessity, to adequately address the future complexity challenges.

You could therefore say that collaboration was a survival skill back in the stone age, and is equally in today's modern business environment.



10 Years

Is there still work to do?



Over 10 years now, GENIVI has worked with this idea of creating shared specification and software, getting rid of waste, avoiding reinventing of the same software over and over, avoiding the wasted effort that come from integration problems between software, and... just general *uncertainty*. This uncertainty goes beyond technical matters – it can show itself as distrust and constant friction in the business and contract relationships. This is why the industry conversations we seek to facilitate can have profound effects.

Inefficiency prevents the industry from having the time and resources to solve the really challenging problems. Just picking on one, does *anyone* here believe that our industry is current successfully solving the future cybersecurity challenge?

If like me you think the answer is no, then there are two ways to work on that:

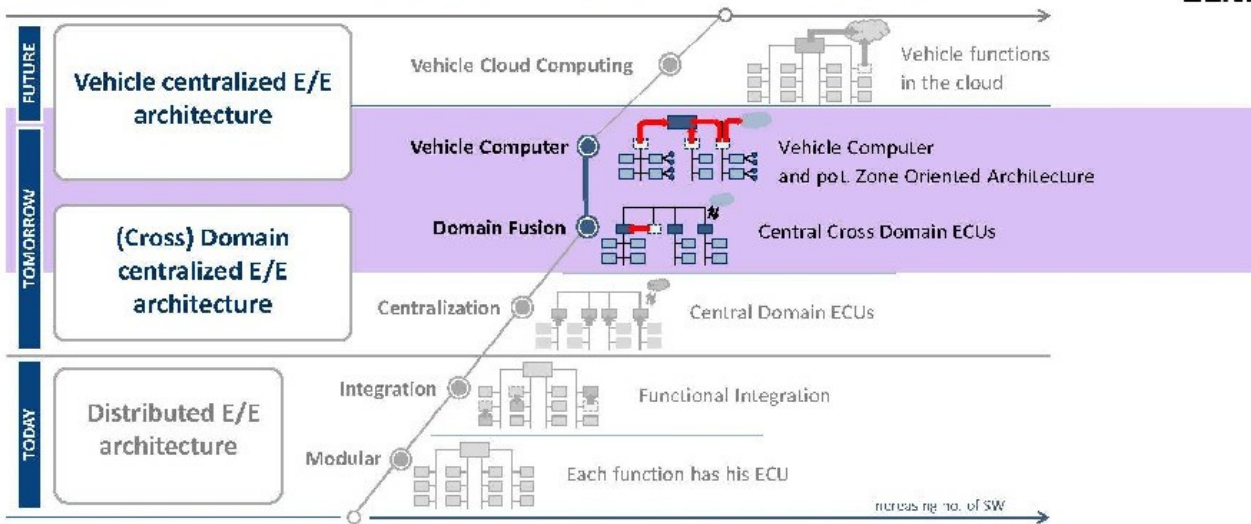
We could address the security question directly and become more effective in that work, and GENIVI works on that too. But you can also realize that there is a lot of effort spent on unnecessary basic development problems in other areas. If the industry fixes that through effective collaboration, it will free up resources to *instead* work on e.g. Quality. Safety. Innovation.

Will every one of you, after this webinar and report put in small part of your resources towards the industry goal to achieve a virtualization standard?

Many of our projects use this 5-level model about the history and future of in-car electrical architectures.



Trends in Vehicle EE and Software Architecture



© Robert Bosch GmbH 2016. All rights reserved. Also regarding any deposit, exhibition, reproduction, editing, distribution, as well as in the event of applications for industrial property rights.

13 February 2020 | Copyright © GENIVI Alliance 2020

7

The picture is used with permission from Bosch who is the copyright holder.

NOTE It is likely that you could get the right to use this picture through other means, but in this document the open-source license of the document does not include this picture.

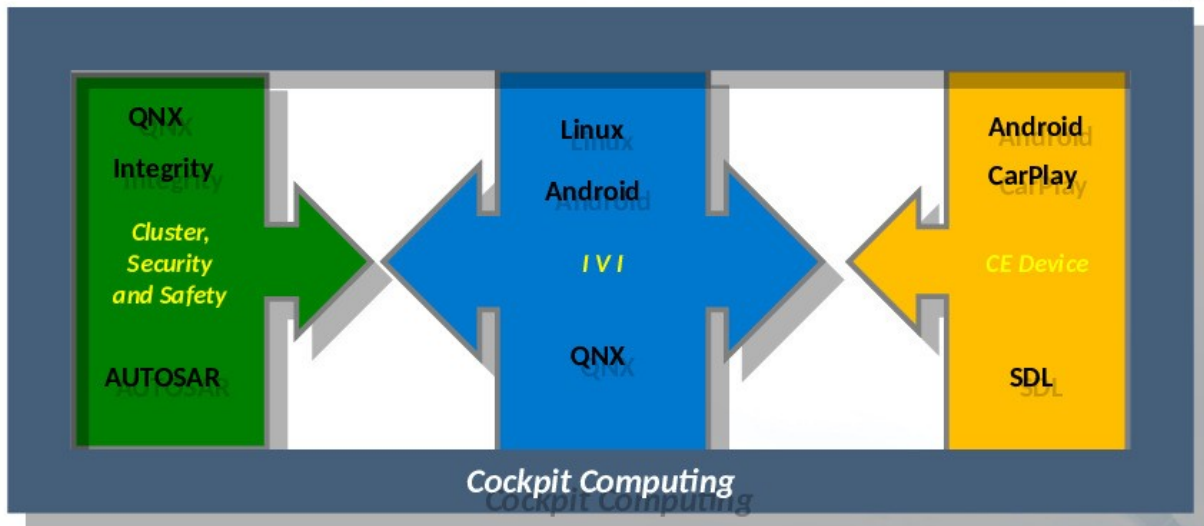
The industry is currently speaking a lot about consolidation and domain fusion. This includes creating large central high performance computer(s) to process many different tasks, and that is one place where virtualization is likely to be expected.

I believe that very quickly following on the domain fusion came a kind of push-back and the realization that the physical reality of the car is still there. Physical laws of the universe can't be broken... such as it matters where you locate an antenna if you want wireless communication and it matters how long the cable to that antenna must be. For some other functions, initial processing might be best done NEAR the sensor that measured the data. All of those physical realities of the car balance against the consolidation desires, and so the Zone-oriented architecture approach came up. This seeks to optimize the in-car electrical architecture to the maximum. It can strike the right balance between distributed systems and consolidated systems.

The industry is at this 4th level today and systems are increasingly connected, but it is nothing compared to the vision of the 5th level of Vehicle Cloud Computing. Since it's a future this is a vague idea at best, but in a cloud-driven architecture we are not only considering cars to be more heavily connected, to each other and to infrastructure but certain parts of the car may become a flexible computing unit on wheels. Just like servers in the cloud come and go and just like the software running on those servers is constantly deployed and redeployed, we might see similar flexibility in the vehicles themselves.

It's always a step-wise process but looking far ahead, vehicles are destined to become part of a "computing cloud" of sorts, in strong combination with supporting compute power in the fixed infrastructure. Virtualization is only likely to continue to grow in that scenario.

The current reality is Multi-OS Domain Integration



13 February 2020 | Copyright © GENIVI Alliance 2020

8

The in-vehicle reality is Multi-OS domain integration.

Domain integration carries different requirements, different levels of safety and security, and in the current situation in automotive, also **different operating systems**.

This picture shows us that GENIVI today works with the previously shown vision of architecture, functional consolidation, connectedness, and so on. In doing that, we recognize that there is today and for a foreseeable future a diversity that must be handled that includes different operating systems and frameworks. This is not a given, but rather a reaction to the automotive industry today. We need therefore to find common standards that include bridging between so far incompatible and diverse solutions that are being used today.

It is in this Multi-OS environment and with these widely applied integration technologies that GENIVI participants now spend their time. The GENIVI Alliance is taking the responsibility to bridge between companies, and between industry trends, other consortia and ecosystems.

GENIVI has Moved “beyond IVI” and Into the Centralized and Connected Vehicle Cockpit



The GENIVI Alliance develops standard approaches for integrating operating systems and middleware present in the centralized and connected vehicle cockpit.

The alliance links adopters of Android™ Automotive, AUTOSAR, Linux, and other in-vehicle software with solution suppliers resulting in a productive and collaborative community of 80+ members worldwide.

13 February 2020 | Copyright © GENIVI Alliance 2020

GENIVI has moved "beyond IVI"

If you take away one thing from this introduction it is therefore that GENIVI has moved beyond IVI and into the centralized and connected vehicle cockpit.

Now consider the key words here, and how the necessary connections extend beyond the car cockpit:

Connected - is reflected in almost all of our projects but there are a few in particular that deal with designing shared data models and reducing the fragmentation of the vehicle data-driven cloud-connected architecture previously mentioned.

Centralized – this leads to the discussion on consolidation and thus virtualization.

It is here worth pointing out that I see nothing that limits the virtualization standard we are talking about today to the car cockpit... as we shall see it is widely applicable. We *should* be aiming for a standard for all in-vehicle computing systems. This is particularly important when functions are re-distributed between ECUs, because the electrical architectures are likely to continue to be redesigned in fundamental ways, so that traditional cockpit functions and other functions may no longer be separate entities.

GENIVI addresses this Multi-OS, multi-domain reality by linking together adopters of different Operating systems

- agreeing on standards for interoperability
- seeking out and agreeing on shared software for fundamental technologies. Ever since the start of the alliance, we believe open-source licensing is a very effective way to realize that.
- Finally, GENIVI of course also creates a network of business opportunities among its members

But GENIVI's impact goes far beyond its core membership by running publicly available projects that you can join to start participating in while you consider being more involved in the alliance through membership.

We also have a very wide industry collaboration with other groups and consortia. To highlight only one, our conversation with AUTOSAR including activities on standards, tools and software implementations to bridge between AUTOSAR based systems and non-AUTOSAR systems.

Related GENIVI Projects



- Android™ Automotive Special Interest Group (SIG)
- Cloud and Connected Services Project
- Automotive Cybersecurity
- Interoperability with Adaptive AUTOSAR (FARACON Project)
- [Hypervisor Project](#)

Related GENIVI Projects

There are more projects that are either ramping up or ramping down.

We wanted to just quickly highlight these. We have a saying that keeps coming back in the GENIVI projects today and that is "Big picture", because with the complex interaction of today nothing in the vehicle stands on its own!

It is by really considering how each technology extends into the rest of the vehicle and out to the connected cloud that we can have the best impact. Otherwise you tend to overlook all the unnecessary integration / translation / adaption / boiler-plate code, and so on, that needs to be written if technologies continue to be developed in isolation.

As one example, we provide a Special Interest Group for Android Automotive adopters (AA-SIG). OEMs and involves suppliers come together and share experience and concerns. and here we consider this big-picture thinking that might not be met in each individual production project or in the conversations OEMs have directly with Google's Android engineers. There is a risk of looking at the Android "IVI" system in isolation.

The AA-SIG is not only about concerns and experiences however – we do concrete design work in this group, based on the direct needs of OEMs and detail "problem areas" as reported by implementing

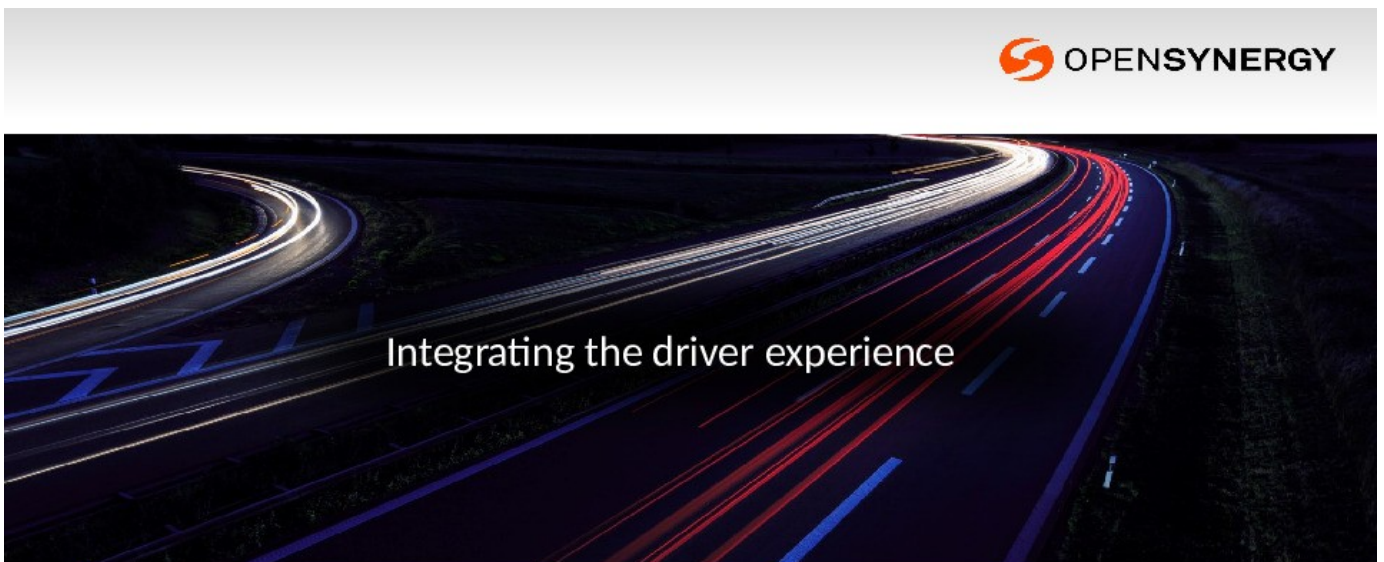
suppliers. This is design work (and software) that the working group envisions to affect future Android systems in the car.

Cloud and Connected Services sounds like a generic name but is similarly analyzing and designing with a big picture approach to seek consensus in the industry around the design of the FULLY vehicle-data driven connected architecture.

Security, of course. We could discuss that for a long time, but it is not directly the subject of today.

The collaboration with AUTOSAR has been already mentioned, and the FARACON tool specifically translates interface descriptions between AUTOSAR-XML and Franca IDL used for non-Autosar systems.

With that introduction finished we welcomed Tero Salminen from OpenSynergy to provide a view from a hypervisor expert company on the potential of virtualization standards.



Hypervisors in the vehicle: A standards-based approach to virtualization

public

- OpenSynergy have been working with hypervisors and virtualization for automotive domain for more than 10 years
- We always have three problems
 1. Get the hypervisor running on a particular architecture
 2. Virtualize guest operating systems
 3. Virtualize the devices
 - 1) and 2) rather easy to do especially now when both ARM and Intel support virtualization
 - 3) is huge amount of work that needs to be done repeatedly
- From the customer point of view the device virtualization is the most expensive part and it also locks them to a particular vendor as switching the hypervisor means re-implementing the device virtualization part.

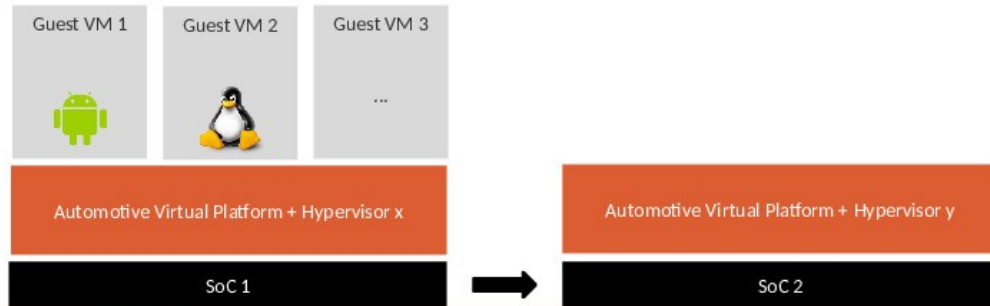
□ We decided on look on to ways to make device virtualization easier and cheaper for the ecosystem

- OpenSynergy has been a member of GENIVI for several years
- 2017 GENIVI started to look beyond their initial scope in IVI systems and hypervisor was one potential topic for a future project
- GENIVI Hypervisor project was official started early 2018 and OpenSynergy became one of the main contributors to the project
- Device virtualization quickly became the focus area of the Hypervisor project and VIRTIO was selected as the technology that will be used for the standard interfaces
 - Use of VIRTIO allows us to use existing technology currently mainly used in cloud
 - Every hypervisor can support VIRTIO
 - Automotive specific domains that are defined in the GENIVI **Automotive Virtual Platform Specification** (e.g. audio, codecs, 3D GPU, power management) need to be added to the VIRTIO Specification
- Working both on the context of GENIVI (Automotive Virtual Platform Specification) and OASIS (VIRTIO Specification) gives the maximal benefits and connects the two worlds (Automotive and Cloud).

What is it? A standardized platform being specified by GENIVI, implemented by OpenSynergy and other HV vendors

Standard Based on open standards (OASIS VIRTIO) to allow multiple interoperable implementations.

It means: Hypervisor and hardware can be swapped without the need to adapt the software



Automotive Virtual Platform



VIRTIO 1.1

VIRTIO 1.2 or later



Headquarters

Further Locations

Berlin

OpenSynergy GmbH
Rotherstraße 20
D-10245 Berlin
Germany
Phone: +49 30 / 6098 5400

Munich

OpenSynergy GmbH
Stamberger Str. 22
D-82131 Gauting / Munich
Germany
Phone: +49 89 / 2153 9073

Utah

OpenSynergy, Inc. (USA)
765 East 340 South
Suite 106
American Fork, Utah 84003
USA

California

OpenSynergy, Inc. (USA)
501 W. Broadway, Suite 832
San Diego, California 92101
USA
Phone: +1 619 962 1725

Shanghai

OpenSynergy GmbH
2608, Enterprise Square
228 Mei Yuan Road
Shanghai
P.R. of China
Phone: +86 132 6 277 7738

E-Mail info@opensynergy.com
Web www.opensynergy.com

Hypervisor Project

- A collaborative group addressing challenges in Virtualization
- A key activity since the start of GENIVI's Domain Interaction and Multi-OS integration strategy
- Bootstrapped with a number of workshops, All-Member Meetings, Tech-Summits
- Weekly call every Monday at 10am CET
 - US-friendly call time is possible (let us know your interest)
- All documented on GENIVI Public Wiki:
 - [Minutes](#)
 - [Agenda announcement](#)



To put the writing of a virtualization specification in context, just a short introduction to our project organization.

The "Hypervisor Project" is perhaps an unimaginative name, but it is not the name but the **work that matters**, so we have continued with that. We have weekly phone calls which are currently run with primarily European and Asian participants due to the time slot.

GENIVI is an international organization and a big reason to have the webinar and report, is to connect with all of our supporters in North America and start the conversation. We're happy to schedule meeting times as needed for other time zones and generally just make it work, so please send an email over to get the conversation started!

We publish agenda, minutes and results on our public Wiki and you are welcome to join the project also if you have not yet sent in that membership application. This is run as a public project.

Let's summarize the goal of the entire Hypervisor Project as follows:

Hypervisor Project: Goal



Make virtualization easier to deploy in automotive environments

13 February 2020 | Copyright © GENIVI Alliance 2020



"Make Virtualization easier to deploy in automotive environments!"

Project Activities

- ***Make Virtualization easier to deploy in automotive environments***
- **1) Standardization**
 - Hypervisor APIs including Virtual device interfaces
 - -> → Today's main topic!
 - Architecture considerations
- **2) Outreach and education**
 - Designing with virtualization – Whitepaper
 - “Why (when) is a hypervisor necessary?”

Project Activities

“Make Virtualization easier to deploy in automotive environments!”

How might we address that?

First, Standardization, the key premise of today's conversation. Remember the question: “Is there a need for a more standards-based approach?”

Second, Outreach and education

- Every time we put together any project group we tend to educate each other. It is a prerequisite to bring everyone to roughly the same level of competency in a subject matter for the group to then be effective.

But we also strive to make publications. In addition to the specification, which is actually written to be quite instructive as well, we are discussing a planned Whitepaper talking about why or when a hypervisor is necessary.

Participants Since Project Start

- ADIT
- Alpine
- ARM
- Bosch
- Elektrobit
- Kernkonzept
- EPAM + other Xen-Project representatives
- Green Hills Software
- OpenSynergy
- Perseus
- Sysgo / PikeOS
- Virtual Open Systems
- XVisor founder and developer

Participating companies

The companies that have been involved at some point since the project start. I hope we didn't forget anyone of you.

Some participants have come and gone but it's great to have such a wide variety of producers of virtualization technologies as well as users and buyers influencing this work.

If you personally are not on the list yet you should join us at least for a while to influence the next step now that we have shown that the collaboration and standardization *is* possible also in this area. No matter where and how you are related to this subject matter, there are things that everyone can provide, including just quality improvement on the delivered texts. No matter what your role is, you can provide **something** to this work.

Why Virtualization?

"Just use separate CPU cores"

"Just use Linux containers"

Look out for the technical arguments in upcoming whitepaper

Why Virtualization?

In a typical conversation we are likely to have these possible questions and objections popping up.

All of those are great and are not really objections once you understand the details. These technologies are part of designing a complete system.

For example, containers will be used regardless, when you are running Linux. There is basically not a single Linux system in servers and desktop that does not make use of namespace isolation somewhere, and increasingly in the embedded space as well.

We also see a changing landscape. Hardware manufacturers are adding more hardware features to support virtualization.

That in itself is nothing new since the basic virtualization support has been in server/desktop processors for decades, but is continuing in the embedded space today, to further cover the many shared peripheral hardware on embedded System on Chip designs.

The increased hardware support somewhat reduces the complexity in the software layers that implement device sharing but there is still a need for some software layer to set up the hardware features correctly.

Also as we mentioned, today we have a Multi-OS situation to deal with.

In the end there will always be **some** shared resources remaining and we have domain experts in the group working on explaining why virtualization is needed to FULLY solve this. They're the experts but this is my take on it:

Often there's just a kind of hand waving going on so that when the spatial isolation is achieved and basic scheduling then there are often simple claims made that "Task A can never affect Task B"

This first assumes that you fully trust the software layer doing that isolation, whether that is the Linux kernel or something else (that is an ongoing discussion but not something I am covering today).

Spatial isolation is primarily the idea that memory separation has been achieved, so task B cannot read or modify the memory of task A. (Using the word “task” in a generic sense here – it translates to processes running on a kernel and/or systems running on virtual machines).

My understanding from speaking to our team is that the really intricate details of temporal isolation are possibly overlooked. The way that one task can very subtly influence another due to shared timing on buses, on hardware peripherals, on the complex analysis of cache coherency and cache misses.

Consider for example that some caches are dedicated to each core so if you achieve core separation of tasks that is one thing, but then there are shared caches as well. Together with the memory bandwidth, the need for

In addition, there is often other peripheral hardware that is shared in various ways. Consider the desire to use DMA functionality to quickly offload the execution of data movement to hardware that is independent from the CPU core. At the same time, those DMA transfers may be associated with different virtual machines executing functions with different levels of criticality. In many implementations, ensuring full spatial/memory isolation is difficult enough and implementation of “Virtual IOMMU” can be complicated, and also depend on the exact capabilities of the hardware. The timing aspects must then be added on to that.

All of this has solutions of course. To some extent advanced kernels like Linux may implement the possibility to configure things in ways that handle some of the aspects. Overall however, the solutions that implement this control are likely to be in a piece of software that ought to be called a hypervisor, and the system design that fully handles this ought to be called virtualization.



Can we do better, with the help of standards and shared requirements?

The working group has an idea is to collect an understanding of problems and solutions in a separate document. We call this a whitepaper for now, but it may rather be more appropriate as something else later on. The idea is anyway to defend this premise that virtualization is needed to meet fully meet security, safety, goals and so on. We would much welcome your input on that work.



Automotive Virtual Platform Specification

13 February 2020 | Copyright © GENIVI Alliance 2020

25

The specification we have created and released in draft is called **Automotive Virtual Platform Specification**.

AVPS – Approach

a.k.a. Avoiding XKCD standards proliferation

a.k.a. How to add value with a new specification



- Common requirement specification for the automotive industry

- “Stand on the shoulders of giants” (build on previous work!)
- Open License (allow anyone to participate, build on top of this)
- Holistic approach - Take whole industry into account –
 - Current trends and needs in industry for software
 - The evolution of future hardware (virtualization support)

AVPS - approach a.k.a. Avoiding XKCD standards proliferation

How should we approach this?

How to avoid the XKCD effect? *(If by chance you have missed this widely common reference, just type "xkcd standards" into any web search.)*

The question is how do we add value with yet another specification?

No doubt all significant software work is built on the results of previous work. But we can take the approach of explicitly seeking that, recognizing what has come before and base on existing standards where possible.

Open source document licensing is used to make sure the result is as widely accessible as possible

The holistic approach, or the big picture again, is about adding knowledge about the current trends and needs in automotive and talking to designers of automotive specific hardware to ensure that influences the result as well.

“Why not just use VIRTIO?”

13 February 2020 | Copyright © GENIVI Alliance 2020

27

A conversation about this work might sound something like this

- Hey we're writing a standard specification for virtualization.
- What do you mean by that?
- Well, I would say, to put it in technical terms it is essentially a defined API between the operating system kernel and the hypervisor or the virtual hardware.
- Oh, but why not just use VIRTIO?

At times it can be frustrating to meet many people that seem to presuppose that the work is unnecessary, overlapping or conflicting with things that already exist. But it took me time to realize that this is not necessarily the reason. Sometimes the question should rather be understood simply as "Oh, that's interesting, can you tell me more?"

But let's answer the question, nonetheless. The first answer is, as Tero explained in his presentation that we ARE using VIRTIO!

- A large part of the specification is referring to VIRTIO. And I highlight this word **referring**, which will be explained later.

It's however about digging a bit deeper. First of all we know from experience that not all hypervisor implementations implement VIRTIO all the time. They may also have found in *some areas* some criticism of the solution. The shared conversations can help to analyze if those areas are a problem, and perhaps find a middle ground, or propose a different solution if warranted for a particular area.

Next, there are constant proposals not yet accepted into the VIRTIO spec. If the agreement was “just use VIRTIO”, are those also required? Or not? When do we expect such things to be introduced and is it OK to have a conflicting implementation while the final details are put in place, or not?

Then you might go and browse GitHub. There are many independent third-party projects that use the VIRTIO way to implement a particular device sharing and these tend to be named VIRTIO-something. In other words, they explicitly use the mechanisms such as virtqueues, as defined by VIRTIO. Are those projects then included in our agreed requirement?

Are even **all** the features in the official released VIRTIO included? What about those that only make sense for desktops or servers?

And when and how are alternative "optimized" non-VIRTIO solutions acceptable?

So if you were planning a new project including negotiating a business buyer/supplier setup, estimating costs, and defining contractual obligations, would it be acceptable to have this sentence: "Just use VIRTIO" as the basis for that agreement?

Of course it would not. When we **really** need to understand each other we are seeking a much clearer, much more specific documented standard on what the automotive industry requires.

- While this rigidity might seem heavy, such clarity is not a burden. Rather it frees everyone up from uncertainty and gives us all a shared basis for the conversation.

- But it is a lot of conversational work to do to achieve. Avoiding the standards work you are not likely to be able to avoid that conversation effort. In fact you will have it over and over. Sometimes people complain that agreeing in a many-to-many group-consensus takes a long time and a lot of work. I think those people are heavily underestimating the efforts spent on repeated one-on-one conversation between every combination of seller and buyer, both in the planning of projects and most definitely in the execution in the meetings that negotiate how to solve "problems" that pop up.

Finally, the AVPS specification is bound to include several additional requirements that are, and will remain, out of scope for VIRTIO. The first release already includes such areas. Therein is the rationale for creating the AVPS in parallel with VIRTIO.

Standards proliferation? How to add value with a new specification



As stated, reuse already existing good work!

- Clarify & specify scope and applicability
- A reference/delta specification =
 - Specify which parts of referenced spec apply, which are mandatory
 - Specify differences or add/delete requirements
- Work with "upstream"
- Augment spec with other types of requirements

Standards proliferation - again

So to summarize, how do we add value here:

- Reuse heavily. Discuss. Do we have consensus on this approach? Will everyone implement it as written, or do we need to change things to make it more accessible. (Every implementer needs to be part of that conversation if it is going to work well).
- Reuse by reference. We don't copy the requirements and details from VIRTIO into AVPS - we specify when we can, which parts of which chapters in the referenced specification are to be applied.
- The delta here is that AVPS can describe only the difference compared to our "upstream" as it is known in the open-source world. It could be as simple as some subject is already well described, but optional, in a referenced specification, and for automotive we might consider it to be mandatory. Such clarifications are made in a reference/delta specification. (Our Wiki has more details on this approach).
- Of course work with upstream, meaning we hope to be able to feed back into VIRTIO and other related specifications.
- And as noted, there will be new types of requirements that need to be agreed and published.

Automotive Virtual Platform Specification



- Primarily a Interface description document, plus additional requirements
- Vendor/Hypervisor neutral
- Operating System neutral
- Hardware neutral
- Encourages many interoperable independent implementations
- Based on existing standards and references those
 - VIRTIO
 - Arm® platform standards
 - Selected small specifications that might be implemented on other hardware
- **AVPS selects, clarifies, and augments requirements to match Automotive needs**

The primary reason to present this today is to say that the document exists. It's no longer just some vision or idea. It's a real thing. It still needs (your) input to make it comprehensive across all areas.

As written on the slide the work has these intentions.

- Define standard interfaces + additional requirements
- Aim to be vendor neutral
- Aim to be OS neutral
- Aim to be hardware neutral

A special thanks here to **Arm**[®] who have been very supportive of our work. Just two things to point out here on hardware neutrality. We are as always looking to make a standard that works on multiple/all hardware. We invite established (e.g. Intel) and new (e.g. RISC V) CPU architectures to join to bring us further in that quest, as well as silicon vendors that produce automotive SoCs.

What we have found are few interface specifications, that seem to be applicable as interfaces to virtual hardware (i.e. part of the virtual platform standard API), despite that the specification is published by Arm. Thus we expect that some of those interface standards can in fact be applied on any CPU type.

We welcome more support and help in achieving hardware independence for the AVPS.

AVPS selects, clarifies and adds requirements to match Automotive needs

Why “Virtual Platform”?

13 February 2020 | Copyright © GENIVI Alliance 2020

30

Why "Virtual Platform"

The name virtual platform was proposed for at least 2 reasons:

- 1) Highlight the standardization aspect of a “platform” on which portable software can be written. We have a few visual slides on this later on.
- 2) Recognize that the actual design of the hypervisor and related architecture differ, even if they would appear to do the same thing. Therefore it is not the specification of a hypervisor per se.

To elaborate on the second point: Sometimes we may be a bit imprecise in our definition and speak about features that the "Hypervisor" must implement. In reality it might be more correct to describe it differently. Terms may mean slightly different things in different teams, but the fundamental design might also be different in reality.

For example, you can imagine a large hypervisor which implements the entire specification while executing in a higher CPU privilege mode than the operating system kernel. In other words as a hypervisor is expected it acts as a kind of underlying kernel for the OSes. This large HV is akin to a monolithic kernel like Linux, whereas other virtualization platform designs mimic a kind of microkernel approach. In those, the hypervisor name is considered a minimal code that only deals with scheduling and separating VMs, and the kind of virtual devices or virtual hardware we are describing in AVPS is perhaps actually implemented in some bespoke VMs on this architecture.

There are different ways to design hypervisors and a virtualization platform. Some systems delegate the device handling to special dedicated virtual machines and some put emphasis on the hypervisor being a **minimal** part, working only as a kind of separation kernel.

Going further, many are familiar with the Xen architecture with its naming of Dom zero, DomU and other such definitions. Then there is KVM, which is realized by a kernel module loaded into the Linux kernel so

that the Linux kernel itself can act as a hypervisor. It can then run potentially other instances of the Linux kernel, or other operating systems of course.

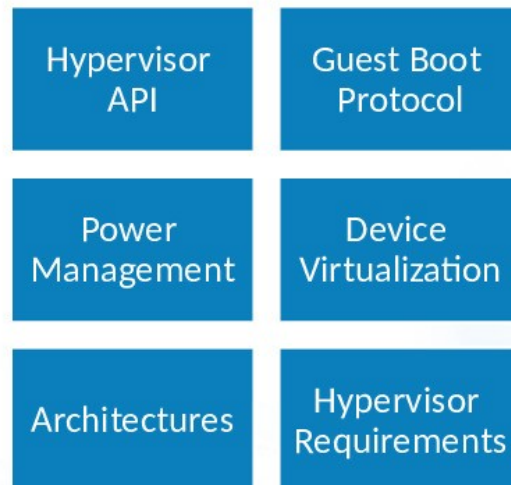
There is hosted virtualization on other systems – you can run Linux appearing inside of a window in Microsoft Windows, or MacOS, and so on. We are not getting a bit too far away from embedded.


In summary, recognizing that all of these are valid and different approaches that can still fulfill a standard specification, we should be speaking of something more generic, like a platform, as opposed to only calling it a “Hypervisor API”.

Areas of Standardization - preview



- Key areas for standardizing interfaces
- Booting, for example, is not VIRTIO
- Power Management:
 Typical embedded needs are greater and different from desktop/server
- Additional requirements...





Challenges to Solve

“*Make virtualization easier to deploy in automotive*”



Challenges in Implementing Virtualization



- Porting existing full systems from “real hardware” to virtual machine
 - Kernel to (virtual) hardware boundary is now different
 - This boundary is different on different HVs
 - → Varying feature support
 - → Integration problems, uncertainties
 - Behavioral changes?
 - Performance?
 - Scheduling of VMs & processes (e.g., Real-time or otherwise critical)
 - Isolation guarantees, popularly referred to as handling *mixed criticality*
 - Safety critical functions and Cybersecurity concerns
- Vendor Lock-in

Actual product development is very diverse, but they share at minimum the need for reducing uncertainty and avoiding integration problems. Some projects might use virtualization to port existing legacy systems into a virtual environment. Avoiding, or at least anticipating, the difference between virtual and bare-metal hardware is one thing. Another is to have a predefined agreement on the expected feature support, no matter what hypervisor you choose.

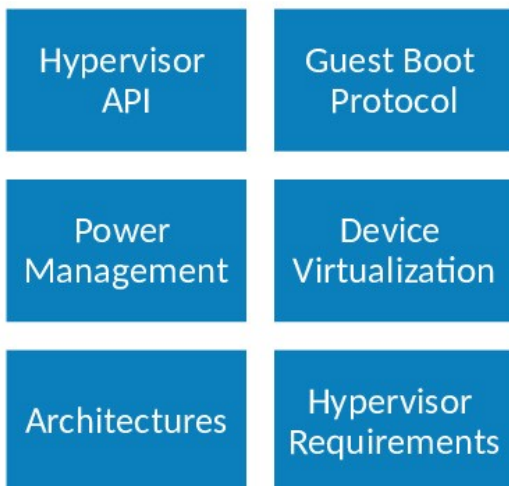
Some things, like concerns around performance, might not be solved directly by standards. We can recognize such areas exist that standards do not solve directly, but that means we need to get other concerns out of the way more efficiently, so that there is time and resources in projects, to work on what is remaining.

The final big point is Vendor Lock In. Here there is an obvious argument to be made from the OEM or buyer perspective but honestly a lot to be said for the advantages on the supplier side to have a specification to lean on when you are making the argument to your potential buyer that “we're not in the business of locking in, but on getting business based on the value, the merits of our solution.” Agreed standards simplify conversation and can therefore get business relationships and successful product development going .

In summary we urge every virtual system to adhere to the specification, and if you can't, you should let the details be known so that the work on AVPS can potentially adjust for it. Whether you are a end-product development organization (OEM, Tier-1, etc.) or an implementor of virtualization technology, now is the time to add **your** input into this, so that the result is useful, and usable, also for you.



Effects of Standardization



- Virtual machines can be ported across hypervisors !
- Common code (drivers, etc.) !
- Shared validation tests !
- Easier multi sourcing !
- Standards can be referenced in OEM specifications !
- Shortened development cycle !
- Software/system re-use !

So we would like to think that the process of creating the specification will help to reduce these problems and have these effects:

- Portability across hypervisors, obviously. At face value this seems to be the primary thing. We spoke about avoiding lock-in but that's more of a future insurance thing. And you reasonably might ask "how often will I actually change hypervisors?"

I want to therefore stress that in my opinion all standards work like this has tremendous secondary synergy effects that may be even more important. The conversations we are having yield a common understanding and a common basis to stand on between all partners, when you get to a product development or when you are in a negotiation phase to figure out responsibilities.

Then moving on to the explicit things listed on the slide:

Common code potential appears once agreements have been met on standards. For open-source projects this is fairly obvious. There are many details to delve into there but there are potentials even in a world that mixes open and closed licensing.

- At minimum we can highlight that checking multiple implementations against common specification can be done with common test methods, test protocols and of course hopefully programmed automated tests.

- Common testing methods. This is huge. No matter if your implementation is open source or closed sourced, validation testing could be the same, and shared.

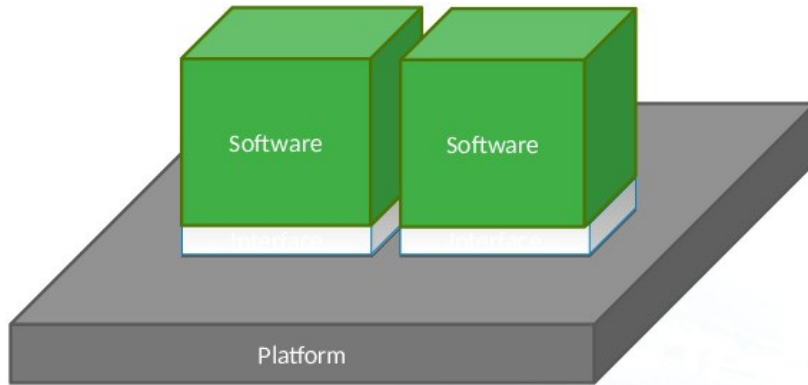
- For OEMs or other buyers in the value chain, sourcing different components of a system from different vendors becomes easier if the integration risks are being managed. Sourcing parts from different vendors require that you are managing risks around integration

- Of course virtualization in itself, once you can trust that integration uncertainties have been reduced, enables the potential to buy the different virtual machines essentially from different sources, with, not *eliminated*, but *reduced* concerns.

To summarize, The outlook is that the AVPS can be used in the requirements between partners, and that future system flexibility is in fact improved due to parts being interoperable.

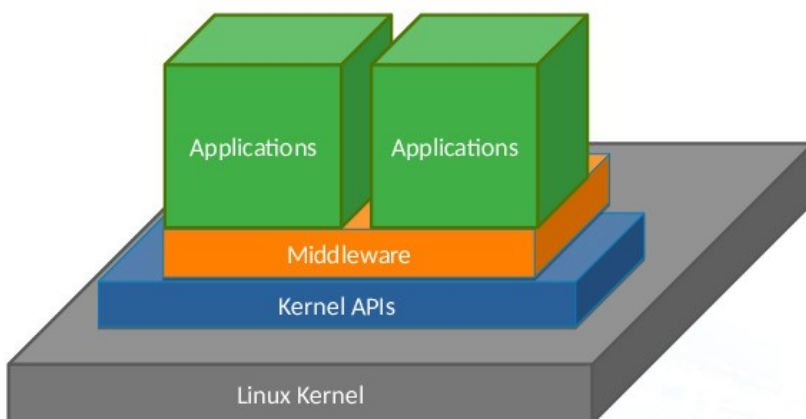
What is a “Platform”?

A computing platform is an **environment** where **software** can be run
...with a well known interface



Standard APIs

GENIVI has a long history of driving standard APIs

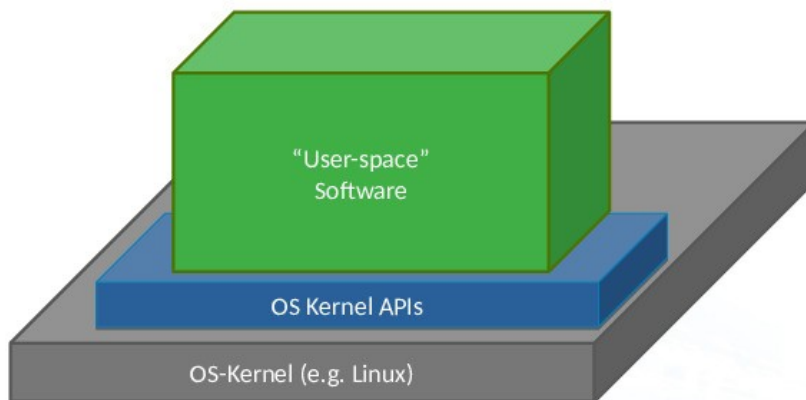


With these slides there is an explanation of platforms, GENIVI's experience in creating standard APIs



Single Kernel Systems

Application code depends on a stable well-defined kernel API

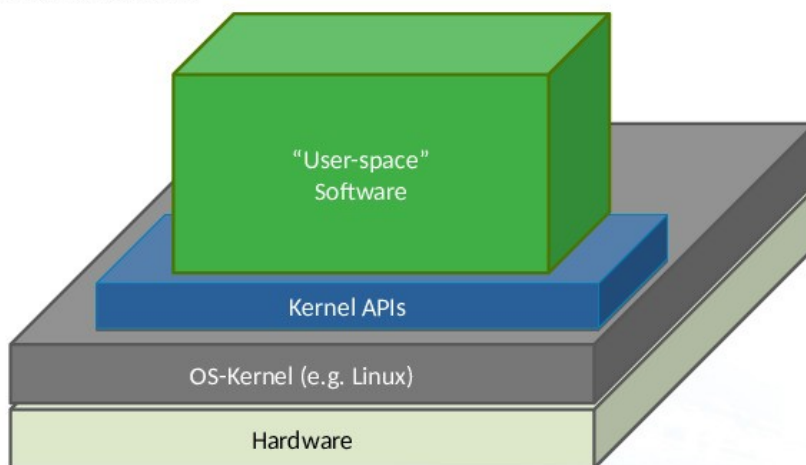


Tuesday, March 03, 2020 | Copyright © GENIVI Alliance 2019

39

Single Kernel Systems

Application code depends on a stable well-defined kernel API
OS Kernel controls hardware



Tuesday, March 03, 2020 | Copyright © GENIVI Alliance 2019

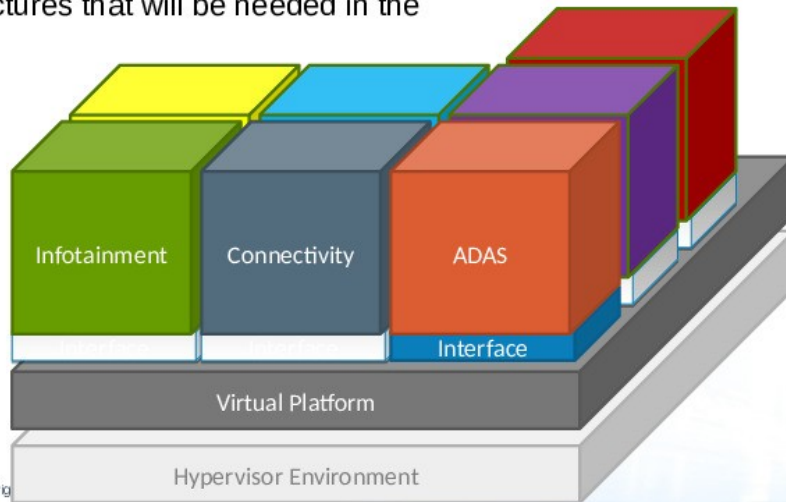
40

The execution model of processes on a kernel and kernel controlling hardware access, versus the execution of kernels in virtual machines and hypervisor controlling actual hardware access.

Automotive Virtual Platform



A shared automotive virtual platform will allow building the **software defined** vehicle system architectures that will be needed in the next decade.



Tuesday, March 03, 2020 | Copyright

43

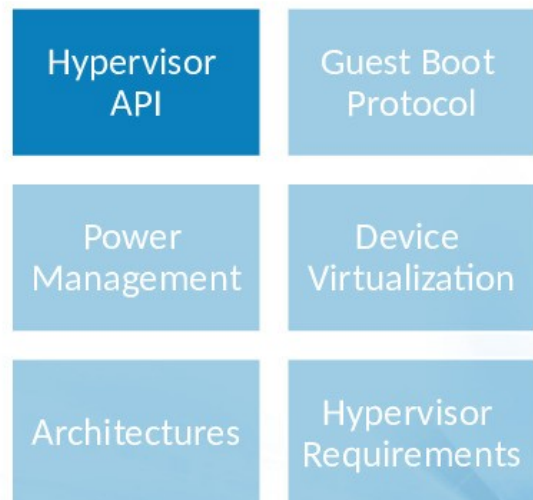
The outlook of a standard virtual platform may enable the necessary future flexibility and achievability of complex automotive systems.

Areas of Standardization



- Standard hypercall interfaces
 - Architecture calling conventions
- Scheduler control
- Guest state management

Find commonality between hypervisors, avoid interface overlap



Tuesday, March 03, 2020 | Copyright © GENIVI Alliance 2019

44

Areas of Standardization

I think we have talked enough about the rationale for this work and I hope to have further engagement in the Q&A. Let's then look deeper into the content of the specification

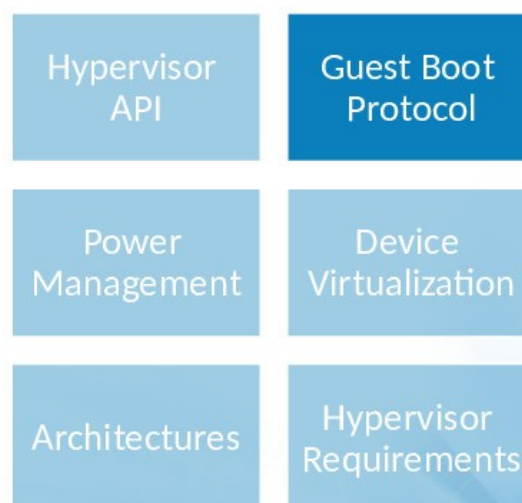
This section is relatively short because actually the released specification itself should give a clear overview. In particular we have not listed every device type here - just look at the table of contents in the specification.



Areas of Standardization

- Boot protocols allow hypervisors to load and boot virtual machines
- Often project or hypervisor specific
- Many protocols exist
 - Classic "Master Boot Record"
 - Linux aarch64 boot protocol
 - Android aboot
 - Standard UEFI
 - ARM Embedded Base Boot Requirements (EBBR)

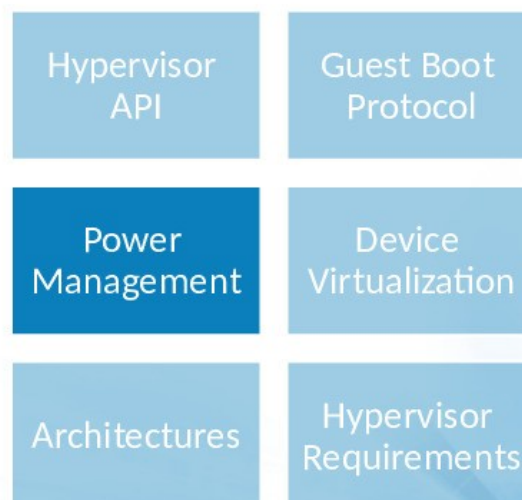
Allow booting any guest on any hypervisor



Areas of Standardization

- Power Management is critical for automotive
 - Thermal constraints
 - Standby endurance
- Many existing protocols
 - ACPI (Advanced Configuration and Power Interface)
 - ARM PSCI (Power State Control Interface)
 - ARM SCMI (System Controller Management Interface)

Find common and suitable interfaces for automotive & define minimum requirements

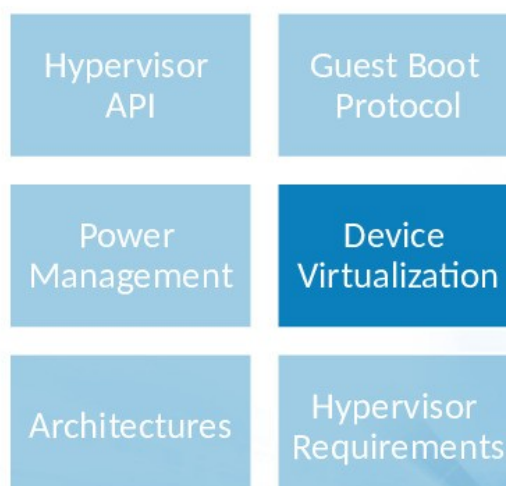


It's worth noticing the areas such as Booting are not VIRTIO-related, and complex areas such as power management that are very necessary for automotive and therefore need investigation and seeking agreement on how "it is supposed to work", when introducing virtualization.

Areas of Standardization

- I/O Virtualization is key when hardware resources must be shared
- Approaches differ but cross-platform and cross-architecture standards exist
 - Device emulation
 - SoC vendor specific solutions
 - VIRTIO

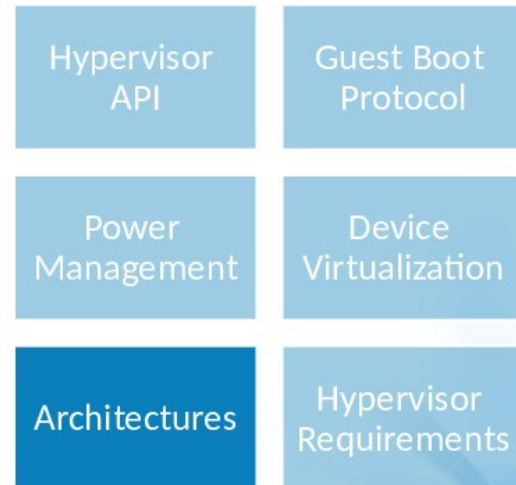
Define vendor neutral interfaces, leverage from enterprise and cloud infrastructure



Areas of Standardization

- Philosophies differ between hypervisors but general concepts often the same
 - Find a common language when talking about virtualization
 - Avoid naming confusion
 - Inform hypervisor users about their options
- Evaluate software architectures for virtualization

Make sure we can talk about the same thing and understand each other

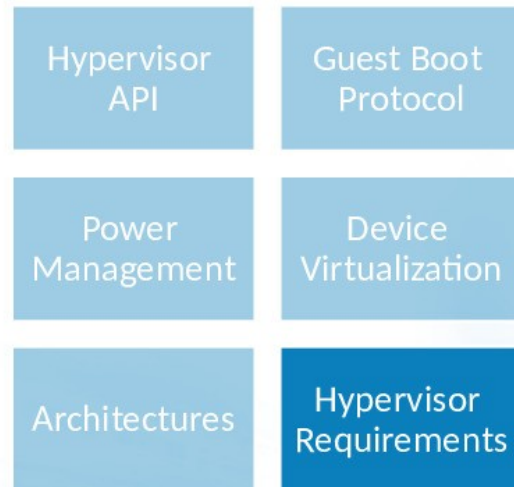


We already talked about the differences in the architecture of the virtualization / hypervisor layer. In addition it is possible to work on a common understanding of terms. Finally, architecture work can also extend to talk about the overall software/hardware architectures. In other words, how does the introduction of virtualization change the preferred solutions for typical use cases. As an example, how do solutions change for rear-view cameras. This, as most people know, include a lot of concerns such as the electrical/network transfer of the video data, latency in the software/hardware stack, reliability and functional safety requirements, and the challenge of early-video which must appear very quickly after power-on.

Areas of Standardization

- OEMs and Tier 1 have broad requirement spaces
 - OEM standards
 - Different system architectures need different requirements
 - Commonality is huge but emphasis differ

Find minimum requirements that allow an even playing field for all hypervisor vendors



The Automotive Virtual Platform Specification

- (Currently) no other collaboration group is doing this for the whole automotive scope. This is the initiative.
- *What do we mean by that?*
 - 1) The current project is open to members and non-members.
 - 2) In all projects, outreach to GENIVI's industry collaboration partners & other consortia

"Maybe we should collaborate..."



- 3) We recognize previous work in this area, build on top of it
- 4) We prepare for others to contribute, extend, build on top of our results. (Open document license)
- 5) We need only *one* automotive virtual platform standard.

This slide and highlights the fact that this is not intended to be an isolated standard for a subset of the industry. Through cross-industry including cross-consortia collaboration we can achieve a usable. The open project setup and the licensing of the result is one of many things driving towards that end.

The HV Project group led by GENIVI is now under way to produce the next version of the AVPS in a fully public project, and we invite your direct participation. At the same time we are expressing our desire to interact with any related groups so that we come together around the common goal.

The Automotive Virtual Platform Specification



Linux systems, Android systems, AUTOSAR, Real-time and safety-oriented systems
Vendor neutral.

Join the cause, help us merge and collaborate.
(There should be only one *automotive* virtual platform standard...)

If you agree – participate, review, write, express support, be public about that support, take part in outreach and dissemination. Join!

A specification like this is never really finished. And in version 1, we are definitely not complete on all chapters. There are areas where we need *your* expertise to strengthen the team, to do the analysis of the current state of the art in that area, and eventually write the actual chapter and requirements. But any other input to improve the result is equally welcomed, such as for example proof-reading, copy-editing, or helping out with illustrations.

We are looking forward to **your participation**. Honestly it would not be unreasonable that every viewer and reader of the webinar/report is here because of interest in the topic and, you therefore ought to bring at least one piece of input to it. I'm further expecting *some* of you to become a regular group participants. Do not just consume, but produce. I promise you it is a lot more fun.

Also, if you were not comfortable speaking on the Q&A, just write down your thoughts now, and send an email over to me.

Q&A

A subset of the questions asked during Q&A, and revised answers

Q: What is the license of the GENIVI Hypervisor?

A: This work aims to produce a specification and not a GENIVI Hypervisor. Multiple Hypervisors may follow the specification. There is not one selected hypervisor and not a particular promoted reference implementation. The working group includes several different competing vendors and technology promoters that work together. To promote that collaboration, this is one area where it would not make sense for the group or the GENIVI Alliance to select a “favorite”. With that said, in all similar work, those who do the work get influence, and all organizations are very welcome to produce and promote their own “reference implementation” which fulfills the specification.

The license of *the document* is a typical open-source document license defined by Creative Commons. Please refer to the document itself.

If there will be any code produced by the GENIVI Alliance in this project, it will be open-source licensed, like all other software we create. The default license in GENIVI is Mozilla Public License v2.0.

Q: How to get involved?

A: Any of these methods should help you get started:

- 1) Seek out the project information on the GENIVI Public Wiki: <https://at.projects.genivi.org/wiki>
- 2) Subscribe to genivi-projects list at <https://lists.genivi.org> – invites are sent to the list.
- 3) **Email GENIVI Technical Lead** at gandersson@genivi.org If you wish for an “on-boarding” separate call, just let us know and we could likely set something up.

Contacts



- **Join project:** <https://at.projects.genivi.org/wiki/display/DIRO/Hypervisor+Project>
or: <https://bit.ly/2vssj3o>
- **Questions/Join project (US time zone):**
Email **PMO** philippe.robin@technoveo.com & **Tech Lead** gandersson@genivi.org
- **Join GENIVI:**
- Email **Executive Director, Steve Crumb** scrumb@genivi.org

References:

- [1] <https://www.automotiveworld.com/webinars/hypervisors-in-the-vehicle-a-standards-based-approach-to-virtualization/>
- [2] <https://at.projects.genivi.org/wiki/display/DIRO/Hypervisor+Project>