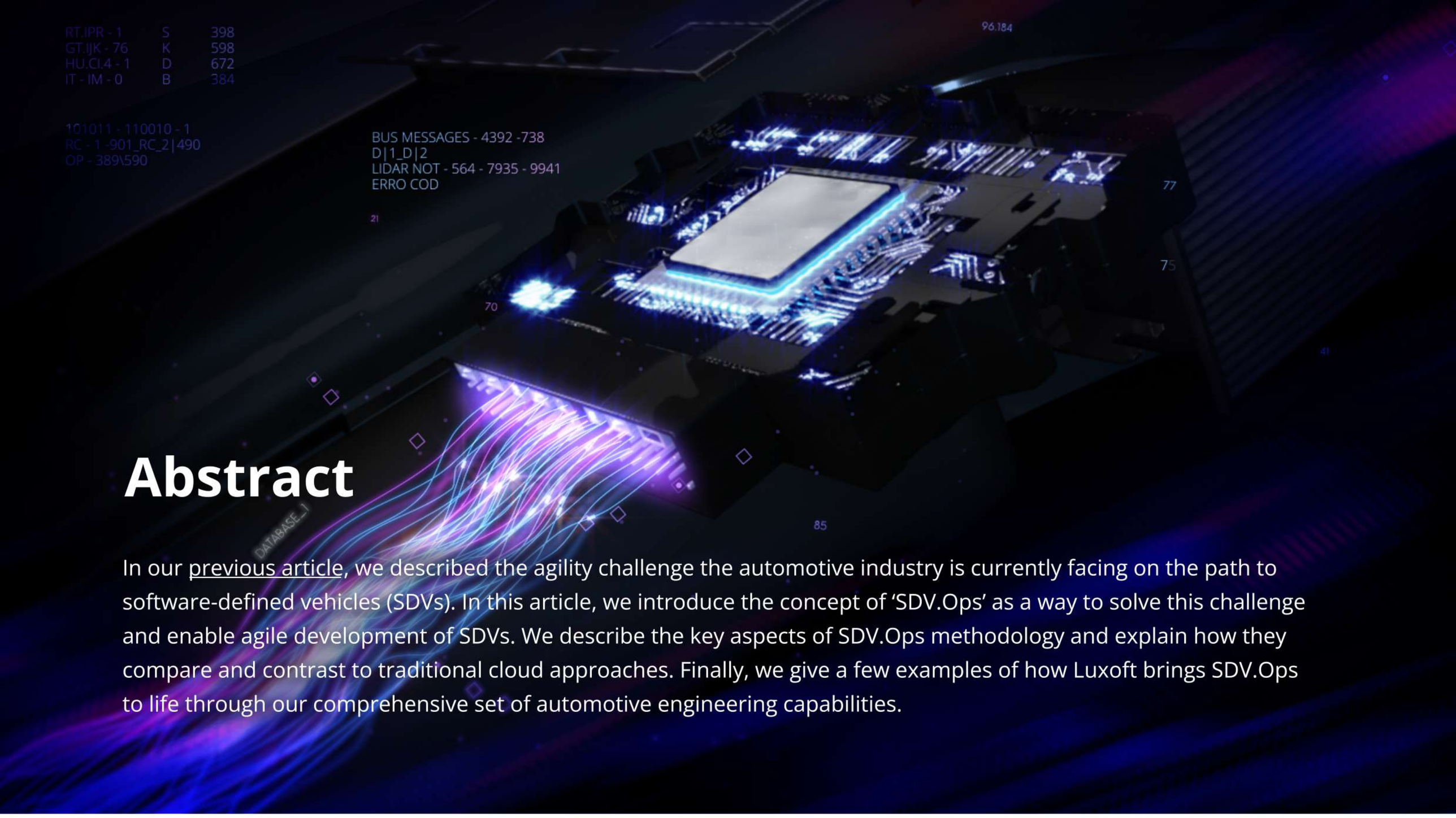
Luxoft
A DXC Technology Company

SDV.Ops: Solving the agility challenge for software-defined vehicles

by **Andre Podnozov**, Chief Architect
Andreas Lindenthal, Advisory Director, Automotive



Abstract

In our previous article, we described the agility challenge the automotive industry is currently facing on the path to software-defined vehicles (SDVs). In this article, we introduce the concept of 'SDV.Ops' as a way to solve this challenge and enable agile development of SDVs. We describe the key aspects of SDV.Ops methodology and explain how they compare and contrast to traditional cloud approaches. Finally, we give a few examples of how Luxoft brings SDV.Ops to life through our comprehensive set of automotive engineering capabilities.

Introducing SDV.Ops

Before we talk about solving the agility challenge for SDVs, let's briefly recap the problem statement: For SDVs that have already been sold, we want to be able to add new features and improve existing functionality in an agile manner; although it's likely that the new approaches introduced here will influence *all stages* of vehicle development. To be clear, we are talking about next-generation vehicles, not the ones that are out on the streets today.

To address our problem statement, we want to take inspiration from the cloud domain, where similar agility challenges have been successfully solved with Microservices and DevOps. Pure cloud approaches won't work in the automotive domain, so instead, let's combine the best of both worlds and try to re-examine cloud DevOps through the lens of automotive realities. The result will be a set of approaches, methodologies and best practices for continuous improvement of software-defined vehicles — SDV.Ops.

As the name suggests, SDV.Ops can be viewed as 'an SDV that Ops itself'. In other words, SDV.Ops allows vehicles to maintain normal operation automatically, requiring minimal human supervision. This is a limited interpretation, however – the context of SDV.Ops is wider. SDV.Ops offers benefits to both software **and** hardware components inside the vehicle; to large fleets of connected vehicles; to ecosystems the SDVs participate in; and ultimately to automotive customers every step of the way.

In short, we can formulate the SDV.Ops mission statement like this:

> "The goal of SDV.Ops is to ensure reliable and secure operation of intelligent capabilities that are added to software-defined vehicles in an agile manner."

Evolution from DevOps to SDV.Ops

SDV.Ops can be viewed as an evolutionary progression of DevOps concepts, applied to large fleets of vehicles instead of large cloud service deployments. The following diagram illustrates the stages of this progression, from traditional DevOps to advanced DevOps, to SDV.Ops:

Let’s walk through the stages illustrated above:

- In the traditional DevOps approach, a cloud service is operated by the same engineers who develop the code of the service. This contrasts with the pre-DevOps approach where operations were performed by a dedicated team which was separate from the Development team
- Today, more advanced forms of DevOps (including DevSecOps) rely heavily on intelligent automation to keep cloud services operating smoothly.
  This intelligent Ops automation is enabled by AI/ML which can help diagnose common operational issues and automatically apply mitigations for well-understood cases. The role of Ops engineers in the advanced DevOps stage is to ensure that the intelligent automation stays within the boundaries of the operational parameters it was designed for. Indeed, operational characteristics of services in a cloud environment are unbounded in principle due to the potentially limitless cloud scale. So, if an especially bad incident goes into uncharted territory (think massive DDoS attack, for example), Ops engineers will take manual control and apply the appropriate recovery steps based on their (human) judgement
- In SDV.Ops, we don’t have an Ops engineer for every vehicle — even if we had enough engineers to operate fleets with hundreds of millions of vehicles, the human Ops might still be insufficient for ensuring the tight reaction times that are required in certain automotive scenarios. That’s why the in-vehicle segment of the SDV.Ops operational model can only rely on Ops automation for the majority of issues. Fortunately, the in-vehicle environment may have a more bounded operational domain compared to cloud environments. This would allow us to develop exhaustive in-vehicle Ops automation that could account for all possible vehicle behaviors, both normal and abnormal

Differences from DevOps

Although SDV.Ops is inspired by DevOps, the parallels between them do not always hold true. Here are a few examples where they diverge:

- SDV.Ops is not a role, unlike a ‘DevOps engineer’ role. This is directly related to the same challenges that SDV.Ops addresses: It’s not possible to have human Ops for vehicle systems (due to intermittent connectivity, large scale of fleets, reaction time

constraints, etc. as discussed previously)
- SDV.Ops proposes a different philosophy of operations compared to DevOps: Operating large fleets of SDVs becomes a feature of the system, rather than an administrative activity
- Whereas DevOps has established a solid base of cloud-centric practices for agile development (such as pipeline orchestration), SDV.Ops is more focused on the vehicle side and uses that base as a jumping-off point for bringing the benefits of Ops automation into the vehicle

We will further explore the topic of 'DevOps vs. SDV.Ops' in a future article, so now let's take a closer look at the foundational principles of SDV.Ops.

Principles of SDV.Ops

1. In-vehicle Ops automation

Of all SDV.Ops principles, this is the most important one — everything else serves to support it. The SDV.Ops mission statement above can only be fulfilled by Ops automation that continuously monitors and adjusts the operational parameters of various components to achieve reliable vehicle operation. The Ops automation should be intelligent enough to:

a. Ideally, avoid any malfunctions by anticipating the required adjustments
b. Automatically identify the root cause and apply appropriate mitigations to resolve the issue if a malfunction does occur
c. Provide graceful degradation of functionality to ensure safe vehicle operation if it's not possible to completely resolve the issue

The automotive industry is already making good progress in automatic identification — and even prediction — of hardware component failures using advanced analytics based on AI and ML techniques. Case in point — Luxoft's recent work on self-healing vehicles.

Luxoft's remote repair of SDVs

Automatic resolution, on the other hand, is still a frontier territory for the automotive industry. The video above showcased a human engineer's involvement in fixing the detected issue, albeit with AI's help, yet even the software industry — with all its advanced methodologies — still approaches the automated issue resolution with caution. The reason is intuitive to understand: Massive cloud systems have unbounded operational domain, so trying to automate their operations is like trying to boil the ocean. There can be an unlimited number of possible states and their transitions in a large system, so there's no guarantee that an issue won't go from bad to worse after applying automated mitigation steps.

Fortunately for the automotive industry, the in-vehicle environment has a bounded operational domain. Therefore, it can be more comprehensively modeled using the advanced system toolsets from the software industry: This way we're not boiling the ocean, only a very large lake.

2. Comprehensive vehicle modeling and simulation

In order for in-vehicle Ops automation to make the right decisions, it needs a good understanding of behavior for the components we're applying the Ops automation to. It should be able to answer questions like: What will be the result of some parameter changes for a given component? How do you get a desired behavior from a given component in certain operating conditions? That 'understanding of behavior' is based on the model of that component. Traditional modeling techniques based on mechanical and physical simulations are being combined with AI-based techniques today for the highest quality of intelligent insight. Of course, if we want Ops automation of the whole vehicle, we need to have a comprehensive model of the whole vehicle — this is known as a digital twin of the vehicle.

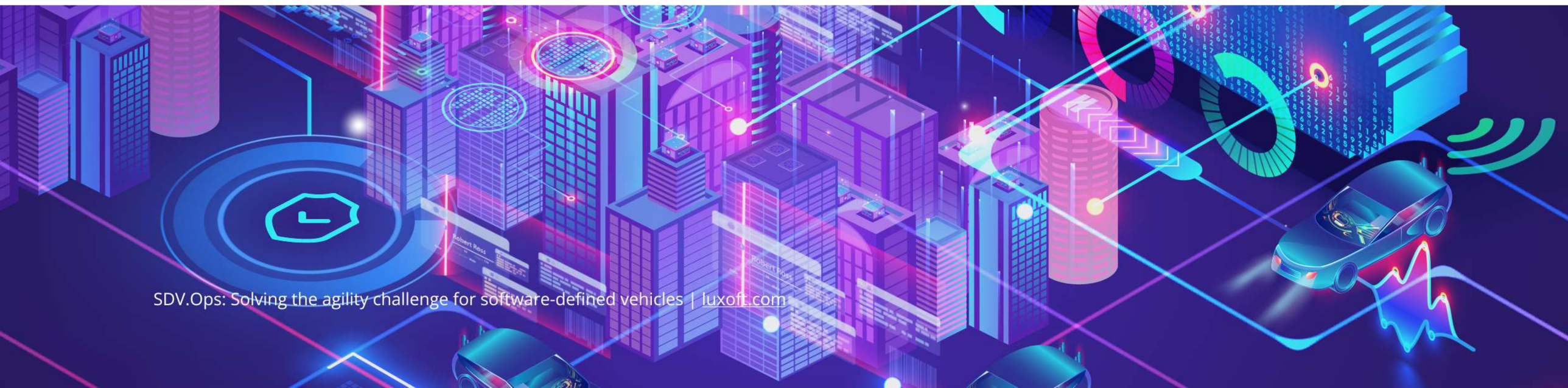
Luxoft is making great strides in applying model-based approaches to development and testing of SDVs, as well as in the area of digital twins.

3. Flexible and efficient in-vehicle software environment

The main ingredient of agility in the cloud — Microservices methodology — was enabled in part by a new execution environment based on containers. For the cloud environment, containers had a clear advantage: They were much smaller than full-blown virtual machines, and therefore were easier and quicker to work with, be it developing, testing, building, deployment, etc. But for the in-vehicle environment, with its constrained computing resources, containers are too heavyweight. Luxoft is actively researching various options for lightweight SDV execution environment, such as SDV Actors which we will discuss in our next post.

The SDV environment will have to be just as capable as containers were in the cloud, but still fit in the constrained in-vehicle compute resources:

- ML will have to be a first-class citizen. Creation of new functionalities in SDVs revolves around intelligent insights enabled by AI/ML, so it's only natural that the SDV environment has native support for the deployment of ML models
- Independently deployable components that are small enough to fit on in-vehicle hardware in the thousands. "Independent" here means that deploying one component shouldn't require downloading gigabytes of images from the cloud, since connectivity is also a constrained resource. And we are talking about many thousands of components because that's what's required when building the level of functionality that consumers of tomorrow will expect from vehicles
- The SDV execution environment should, in true



SDV.Ops: Solving the agility challenge for software-defined vehicles | luxoft.com

microservice spirit, offer developers the freedom to choose a dev toolchain according to their preference

4. Low-friction OTA

Cloud DevOps relies on CI/CD and automated pipelines to speed up the development and delivery of new functionality to users. To cover the complete end-to-end lifecycle of automotive software development, these pipelines would need to extend from the cloud all the way into the vehicle. In the future, we should be able to accommodate multiple small updates per day in a manner that is completely transparent to the vehicle user. Requirements for intelligent SDV OTA include:

- Accommodation of ML components as first-class citizens (like in the previous section)
- Sub-container update. Just like we discussed regarding the execution environment, OTA should also support fine-granularity updates of individual components, without the need to download full container images
- No-downtime update. While some types of updates will continue to require the vehicle to be safely stopped for lengthy periods of time, most updates shouldn't even require a system restart and will be completely transparent to the vehicle users

SDV OTA is another area of active research and development at Luxoft — we will share more details about it in a future post.

5. Security of intelligent vehicle capabilities

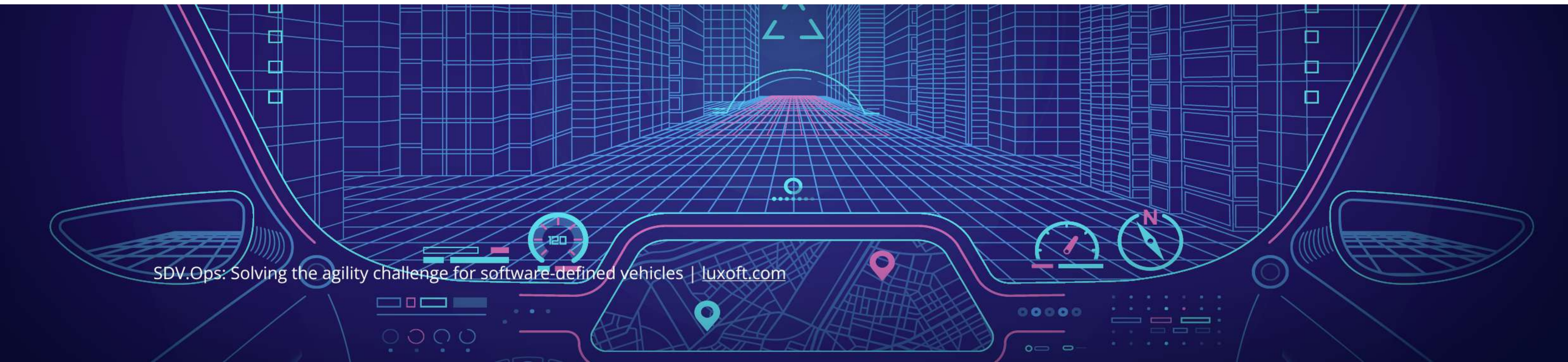
Security is among the most important concern in SDV.Ops, just like in DevSecOps. Traditional DevOps makes certain assumptions about the service operating environment, such as: Physical security of the deployment location (guarded data center). SDV.Ops invites us to rethink these assumptions when we take our code from the cozy confines of the data center into public environments where vehicles can be face-to-face with bad actors at any moment.

As AI continues to find more and more ways into modern vehicles, SDV.Ops must focus on AI security. SDV.Ops must protect the vehicles against adversarial attacks on the ML models that run both in-vehicle and in the cloud. There are several ways ML models can be attacked, and any such attack would pose a potential threat to the underlying business model of a company. Here are a few examples of adversarial attack vectors:

- Tampering with sensor readings that feed into the model inputs will temporarily reduce quality of intelligent insight
- Permanent model poisoning when tampered data is persisted in the training dataset, and the model is retrained on the compromised data
- IP theft through model reverse engineering if attackers can feed the model with a wide range of manipulated values and observe the outputs
- Autonomous driving interference can endanger human life and cause property damage

6. Collaborative ecosystem

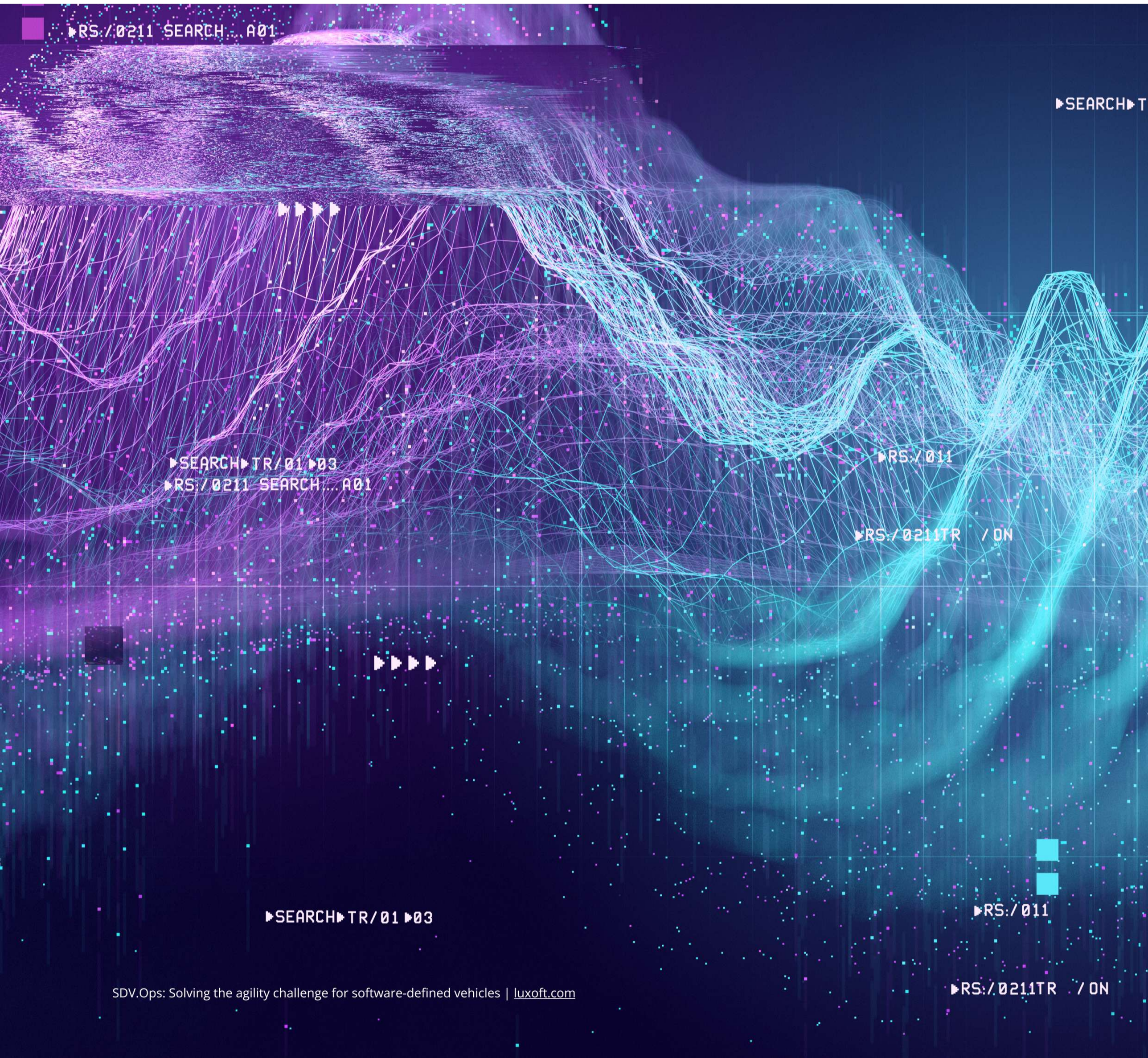
A focus on intelligent features based on AI/ML requires a shift from silo-thinking to wider thinking (cross-disciplinary, cross-domain, beyond-vehicle scenarios). SDV.Ops will enable wide-reaching benefits at the intersections between the Automotive domain and many adjacent domains, such as Smart City and Retail, Travel and Transportation, Construction and Housing, Public Sector and Future of Work — the list is endless, because the Automotive domain connects to so many other domains.



Solving the agility challenges of the future

New intelligent approaches to vehicle operations based on AI/ML are required if we are to realize the full promise of software-defined vehicles. The SDV.Ops concept that we introduced in this article is a multi-faceted topic that deserves much deeper exploration. We'll be continuing this series of publications covering various aspects of SDV.Ops, and in our next articles we'll embark on a quest for a better microservices environment for software-defined vehicles.

As a recognized innovator in the field of software-defined vehicles, Luxoft is uniquely positioned to help clients solve the agility challenges during this transformational time for the automotive industry. You can stay informed about our advancements in SDV.Ops and other topics related to SDVs on luxoft.com/industries/automotive or by following us on LinkedIn.



About **the authors**

Andre Podnozov
Chief Architect
apodnozov@dxc.com

As a seasoned technology leader at Luxoft, Andre has a wealth of expertise regarding the future of mobility. In his role as Chief Architect of Connected Mobility, he's shaping the landscape of software-defined vehicles by leveraging the latest advancements in IoT, AI/ML, Digital Twins and other exponential technologies. By bridging the gap between cloud, edge and in-vehicle systems, Andre is revolutionizing the way we think about transportation.

Andreas Lindenthal
Advisory Director, Automotive
andreas.lindenthal@dxc.com

Andreas is an Advisory Director at Luxoft Automotive. His key areas of interest are processes, methods and tools with focus on digital transformation. His background ranges from automotive hardware architectures, real time embedded operating systems, automotive sensors and infotainment systems development.

About Luxoft

Luxoft, a DXC Technology Company delivers digital advantage for software-defined organizations, leveraging domain knowledge and software engineering capabilities. We use our industry-specific expertise and extensive partnership network to engineer innovative products and services that generate value and shape the future of industries.

For more information, please visit **luxoft.com**