Guidehouse
INSIGHTS

White Paper

Automotive Over-The-Air Updates

A Cost Consideration Guide

Published April 2021

Commissioned by Aurora Labs

Sam Abuelsamid
Principal Research Analyst
Guidehouse Insights

Guidehouse
INSIGHTS

Executive Summary

New technologies including electrification, 5G, computer vision and software solutions based on advanced AI and machine learning are converging. As a result, automotive manufacturers are seeing an increase in security (/hacks) and safety (/bugs) challenges while also having an opportunity to transform business models to include new, aftermarket services and features-on-demand, delivered using over-the-air (OTA) solutions.

The United Nations Economic Commission for Europe's (UNECE) World Forum for Harmonization of Vehicle Regulations has adopted new regulations to manage cyber risks and provide safe and secure software updates. The adopted regulation (WP.29) outlines the requirements and processes needed to assure that the software update management system (SUMS) guarantees safe, secure and reliable OTA updates. Given the widespread use of UN regulations in the automotive sector around the world, the broad adoption of these regulations is expected for all new vehicle introductions starting in 2022.

The realization of using OTA for updates, bumper-to-bumper, has tremendous upsides for the automotive manufacturer and the consumer alike. In the same vein, using OTA update solutions for the entire car comes with additional cost considerations – some obvious, some not so obvious.

Until now, vehicle software updates applied over-the-cable (OTC) have had a complete binary image file downloaded to overwrite the existing image in the ECU storage (Full Image Update). OTA updates have generally followed a similar model to that used for mobile OTA updates where binary images are compared to create an update file that is downloaded to a separate memory bank (Legacy Binary Update). These approaches create several technical and cost challenges when applied at scale. A Line-of-Code (LOC) approach (Line-of-Code Update) that identifies changes in the software at a line-of-code resolution and only applies differential updates with the potential to roll back can be more reliable and cost-effective.

This guide will examine the four major cost drivers for OTA updates and how they can be mitigated, as well as how to optimize the customer experience. The below table summarizes the four cost considerations across the three update technologies for a sample use case that assumes quarterly updates for a fleet of 10 million vehicles, with each vehicle containing 10 micro-processor ECUs and 90 micro-controller ECUs.

	Full Image Update	Legacy Binary Update	Line-of-Code Update
Annual Data Transmission Costs	$ 1,007,560,000	$ 103,250,800	$ 30,982,800
Annual Cloud Costs	$ 19,949,691	$ 2,044,366	$ 613,460
Endpoint Integration Costs	$ -	$ 1,442,308	$ -
Incremental Cost of Dual Bank Storage	$ 1,700,000,000	$ 1,700,000,000	$ -
Total	**$ 2,727,509,691**	**$ 1,806,737,474**	**$ 31,596,260**

These technologies and cost drivers will be explored and explained in the following pages of the report.

Table of Contents

Introduction

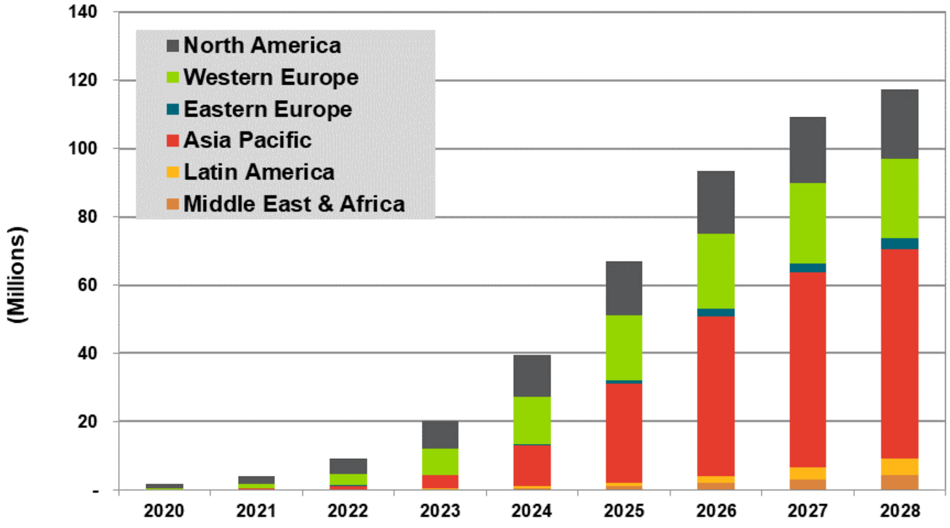
Moving to the Smartphone on Wheels

Since the birth of the modern smartphone with Apple's introduction of the iPhone in 2007, consumers have become accustomed to regular updates appearing on devices that fix problems and add new functionality. In 2012, Tesla brought this to the automotive space with the debut of the Model S and the rest of the industry has been working to catch up ever since.

While the basic premise of over-the-air (OTA) updates carries over from consumer electronics to the transportation sector, the implementation details are far more complex. A smartphone that fails to properly update or takes time to reboot isn't typically a life-or-death scenario. The electronics of a phone are also highly consolidated, typically based around a single system-on-a-chip (SoC). Modern vehicles can range from a handful of domain compute units to upwards of 100 electronic control units (micro-processor and micro-controller ECUs) networked together. The update process needs to be carefully managed from development to validation to distribution.

Guidehouse Insights projects that by 2028, more than 117 million vehicles sold annually will be capable of supporting OTA updates of most systems inside the vehicle. Given the low margin nature of the automotive industry, distributing frequent OTA updates in the most cost-effective and reliable manner is crucial to success.

Chart 01 Annual Light Duty Vehicles with OTA Update Capability, World Markets: 2019-2028



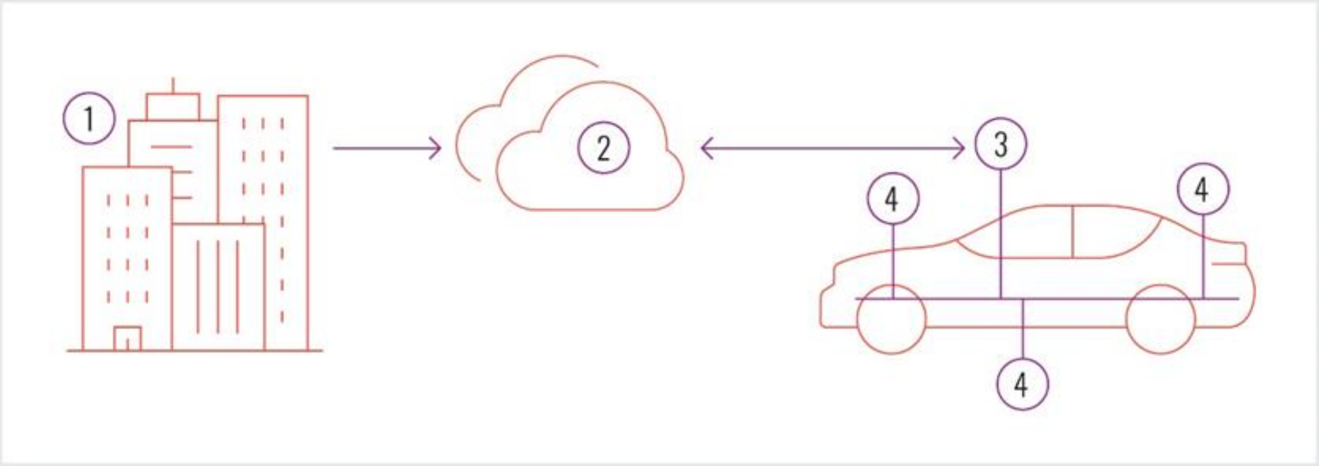
(Source: Guidehouse Insights)

Four Elements of a Software Update Management System (SUMS)

An OTA update solution includes four components (Figure #1). The four components include the enablement of the update file to be 1) created 2) distributed 3) secured and 4) applied on all endpoints.

While all four components are required for a complete solution, not all components need to be from the same vendor. The update file creating technology (1) and the endpoint update technology (4) need to be fully compatible to ensure that the updates produced can be properly installed on the target device. The distribution and security components of the chain can be from various vendors or even developed in house by the auto manufacturer.

Figure 1 OTA Solution Components



(Source: Aurora Labs)

1. Update file creation

The first stage of an OTA update solution is the technology that identifies changes between versions of software and creates the update file. It is technically possible to simply replace the software of an ECU with a complete binary image of the new software as technicians commonly do now when they have physical access at a service facility using an over the cable (OTC) solution. The binary image file replaces the entire contents of the storage memory with a complete updated version of software. This is not feasible or desirable for an OTA update because of high transmission costs and the lack of security sending a full image over-the-air. All OTA update technology vendors compete to provide a technology that creates the smallest possible differential update file while requiring the least disruption to the development and software management processes. This includes development, testing, documentation, certification, configuration management and homologation. This software update file is often called the delta file or the additive update file.

2. Update service management

Once the update file is ready to be distributed to target vehicles it enters the second stage of the OTA update solution. The files are stored in databases together with a database of all the connected vehicles. The vehicle database must have an up-to-date registry of the installed and managed software and the versions of each in the managed vehicles. This is required to ensure that the correct update package is sent to the correct vehicle. This vehicle database is often used for additional vehicle management services such as remote maintenance, warrantee management, fleet management, feature licensing and more. The update service is often a module within a full-service deployment platform (SDP) that also manages security aspects such as secure file transport and encryption key management.

3. Vehicle connectivity, security and update management

To ensure that access to the vehicle is secure it is best practice to enable only one highly secured application in the vehicle to have access to the network (Wi-Fi and cellular) for all data connectivity needs. This application is responsible for a secure connection with the SDP in the cloud, to authenticate the SDP, to authenticate the update file that is received from the update service module and guarantee the integrity of the data sent to the SDP. Further, the application is connected to the vehicle gateway router to enable the update files, once received, authenticated, and decrypted, to reach the intended ECU for the update to commence. This application is also responsible to report the update success or failure back to the update service module running on the SDP.

4. Endpoint update technology

The fourth stage of an OTA update solution is the technology that applies the update file onto the target ECU. The available resources of the target ECU must be taken into consideration when assessing an OTA solution. Unlike the head-unit, telematics control unit (TCU) or domain controllers that have powerful processors, large flash drives and an abundance of RAM, the ECUs used on the power train, vehicle interior, communication and safety ECUs are often resource constrained limiting their ability to integrate and execute legacy update technologies. Another point to consider is the sensitivity of the target ECU to downtime. Can the target ECU be taken offline when it is being updated or is zero-downtime a requirement? How long can the car battery sustain an offline update process? And finally, should there be a problem with the new software version, does the endpoint update technology have the ability to revert to a previously safe, secure and certified version, even if network connectivity has become unavailable?

Four Cost Considerations

There are a range of factors that play into the costs of deploying OTA updates to fleets of in-service vehicles including data transmission, software integration and hardware requirements.

1. Cloud-to-Vehicle Communication and Data Transmission Costs

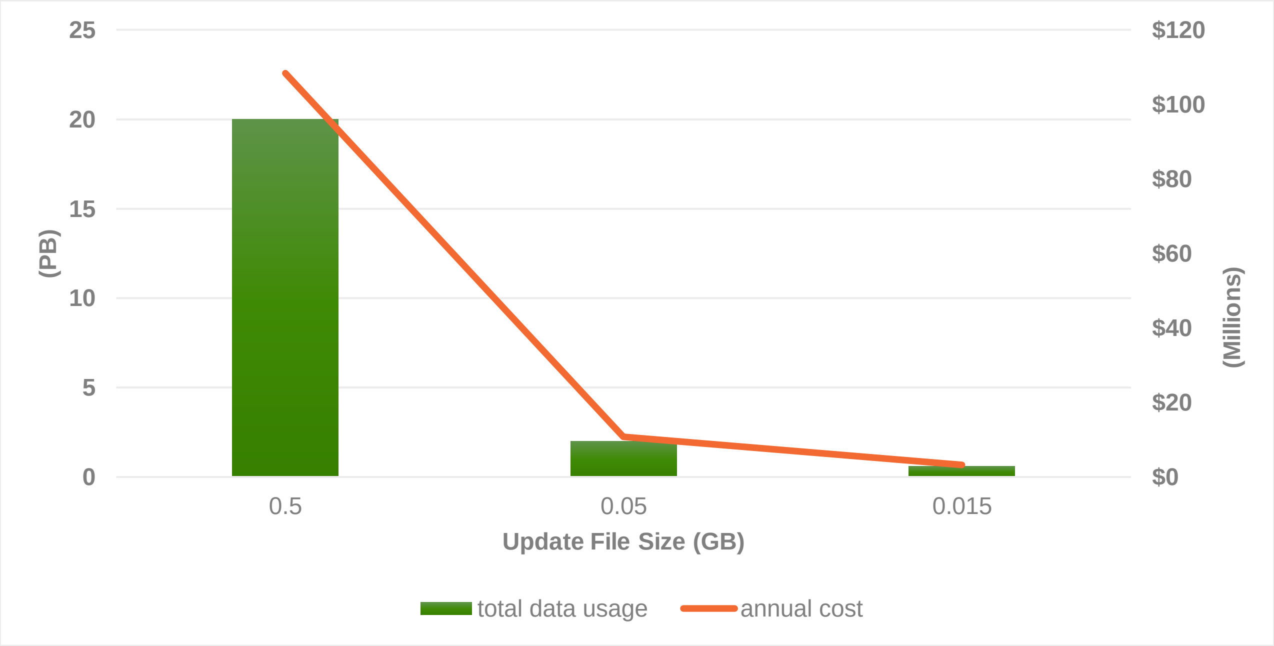
Data transmission costs grow rapidly with the size of the fleet, something that Tesla has already experienced. Prior to 2018, Tesla transmitted OTA updates over the cellular data connection built into its vehicles at no additional cost to customers. However, following the launch of the Model 3, it changed its policies so that only customers subscribing to a premium connectivity plan costing $100 per year would receive OTA updates via cellular connection while other customers had to connect their vehicle to a wifi hotspot.

General Motors includes three years of its basic OnStar connectivity services to all buyers of new vehicles. In mid-2020, it was announced that a premium subscription at $25 per month would be required to continue receiving map updates to enable its Super Cruise hands-free driving system. Other OEMs are expected to start passing on more costs for updates to consumers in the coming years which is likely to reduce adoption of this capability.

For example, if a manufacturer had a fleet of 10 million vehicles in service that required OTA updates, the annual data transmission costs can easily reach into the hundreds of millions of dollars if complete image files are transmitted. In this example of a security update, if a 500 MB binary image file for a single in-vehicle infotainment system (IVI) were transmitted to the fleet once a quarter, more than 21.5 PB would be used annually at a cost of more than $108.3 million. Reducing the update file size by 90% to 50 MB with a binary differential update drops the annual cost to $10.83 million while Line-of-Code Updates that could be as small as 15 MB drop the annual cost to just $3.25 million, only 3% of a full image update. It should be noted that both binary and LOC updates are optimized for software updates and would yield different results for map updates and updates with many new graphics.

With 5G cellular networks now being rolled out, it is possible that the cost of cellular data transmissions will come down however this will likely be counter-balanced by the number of OTA updates performed per vehicle each year as vehicle development becomes more agile and CI/CD methodologies are adopted.

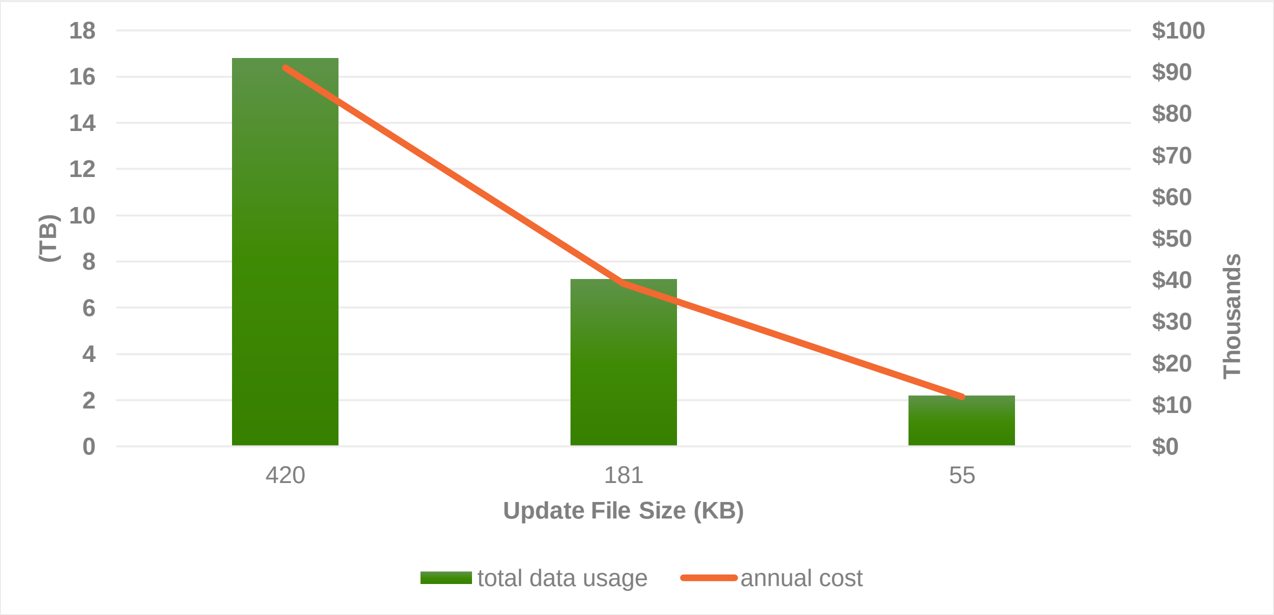
Chart 2 Annual data transmission cost for large OTA update file sizes for fleet of 10 million vehicles



(Source: Guidehouse Insights)

In a car there can be up to 10 large micro-processor ECUs and another 90 smaller micro-controller ECUs. These smaller ECUs would typically run AUTOSAR with much smaller files. For example, annual costs for a 10 million vehicle fleet on a similar update schedule, for a single body-control-unit with a file size of 420 KB, would be $84,000 per ECU in the vehicles being updated. Reducing that file size down to 55 KB with Line-Of-Code Update technology drops that cost to $10,920 per year for each ECU that gets updated, an 87% reduction.

Chart 3 ***Annual data transmission cost for AUTOSAR OTA update file for fleet of 10 million vehicles***



(Source: Guidehouse Insights)

Table 1. ***Summary of Cost Comparisons for Infotainment (IVI) Update Data Transmission***

Avg. Update Size	0.5 GB	0.05 GB	0.015 GB
Annual IVI Update File Data Transmission Cost ($)	$100,000,000	$10,000,000	$3,000,000

(Source: Guidehouse Insights)

Table 2. ***Summary of Cost Comparisons for AUTOSAR Data Transmission***

Avg. Update Size	429 KB	180 KB	56 KB
Annual AUTOSAR Update File Data Transmission Cost ($)	$84,000	$36,120	$10,920

(Source: Guidehouse Insights)

2. Endpoint Update Technology Integration Costs

One of the most challenging aspects of remote updating is the integration of the update technology into the endpoint ECU. Legacy binary update solutions that create proprietary update file formats are required to have an update installer software agent installed on each target ECU. Integration of the update agent requires changes to the bootloader to initiate the update process and integration of APIs to enable the update agent to read/write/erase flash memory blocks. In reality, not every ECU in the vehicle has the necessary resources (RAM, Flash, CPU) to integrate and run a binary update. This integration requires many weeks of development and quality assurance resources that add to the overall cost of the project and may even cause delays in meeting the project delivery schedule.

Assuming this integration engineering and validation takes approximately 6 weeks for each ECU that will require updates, the OEM/Tier-1 engineering cost is likely to be $14,423 for each ECU. With many contemporary vehicles having as many as 100 ECUs or more, the total cost to add update capability could cost nearly $1.5 million dollars. This does not include vendor integration costs.

Table 3. Summary of Cost Comparisons for Engineering

	1 ECU	25 ECUs	100 ECUs
Endpoint integration	$14,423	$360,577	$1,442,308

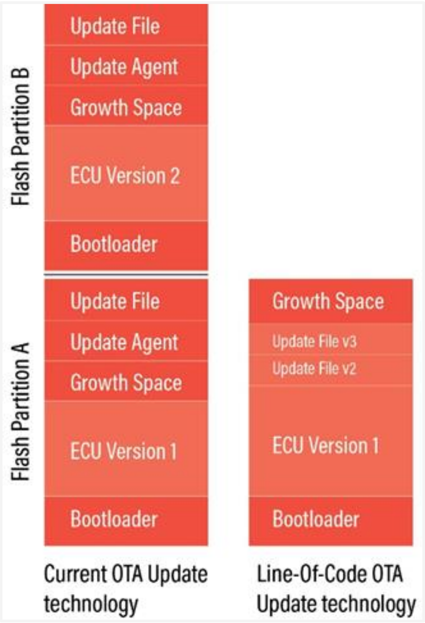
(Source: Guidehouse Insights)

An AI-based Line-Of-Code Update technology is integrated into the development toolchain, automatically creating update files by the toolchain in the same file format as the original embedded ECU file (ELF, S-Rec, Intel HEX). The update logic is pre-integrated into the ECU code by the toolchain enabling the update on the target ECU to be achieved without the need for a proprietary update agent. While this method requires an intimate understanding of the software, designated hardware and development toolchain, it removes the need for costly integration of a proprietary software client on every ECU in the vehicle and guarantees that all ECUs can be updated. In addition, using standard file formats removes the burden of adapting the testing systems, production systems and dealership maintenance systems.

3. Endpoint Update Memory Requirements

There is a challenge to find a balance between update safety, update user experience and cost. On the one hand, there is a need to assure that the update completes successfully and that the update can revert to a previous version in case of a failure. On the other hand, there is a desire to reduce the amount of time the vehicle needs to be in a safe-state and offline to customer use during the update process. Additionally, the amount of time a vehicle is able to be in a safe-state, running on the car battery, is limited and causes degradation of the battery over time. The only way to solve this challenge with legacy update solutions (full image or binary updates) is very costly and requires creating dual partitions in the endpoint memory and doubling the available memory (also known as A/B memory). In this manner, the updated software version image is written to the second partition and if it fails, the system can revert to the previous version that is located on the first partition (Figure #2).

Figure 2 Endpoint recovery memory requirements



(Source: Aurora Labs)

While flash memory prices have declined significantly in recent years, the number of devices that may need to be updatable over the entire vehicle fleet adds up to significant cost. Current prices for 512 MB NAND flash chips that might be used in domain controller architectures cost about $8 each but a manufacturer may produce several million vehicles annually with multiple (3-4) domain controllers, TCU, Head-Units and gateways. Even in an architecture with smaller discrete micro-controller ECUs that use 2, 4, 16 or 32 MB flash memory chips, they typically still cost $1 or more each and a vehicle may have 100 ECUs that require updating. If a manufacturer builds 10 million vehicles a year that require double banks of flash, that could easily surpass $1.5 billion in additional costs.

Table 4. Summary of Cost Comparisons for Storage

	10 High-End ECUs	90 Small ECUs
Incremental cost of dual bank storage for 10M vehicles annually	$800,000,000	$900,000,000

(Source: Guidehouse Insights)

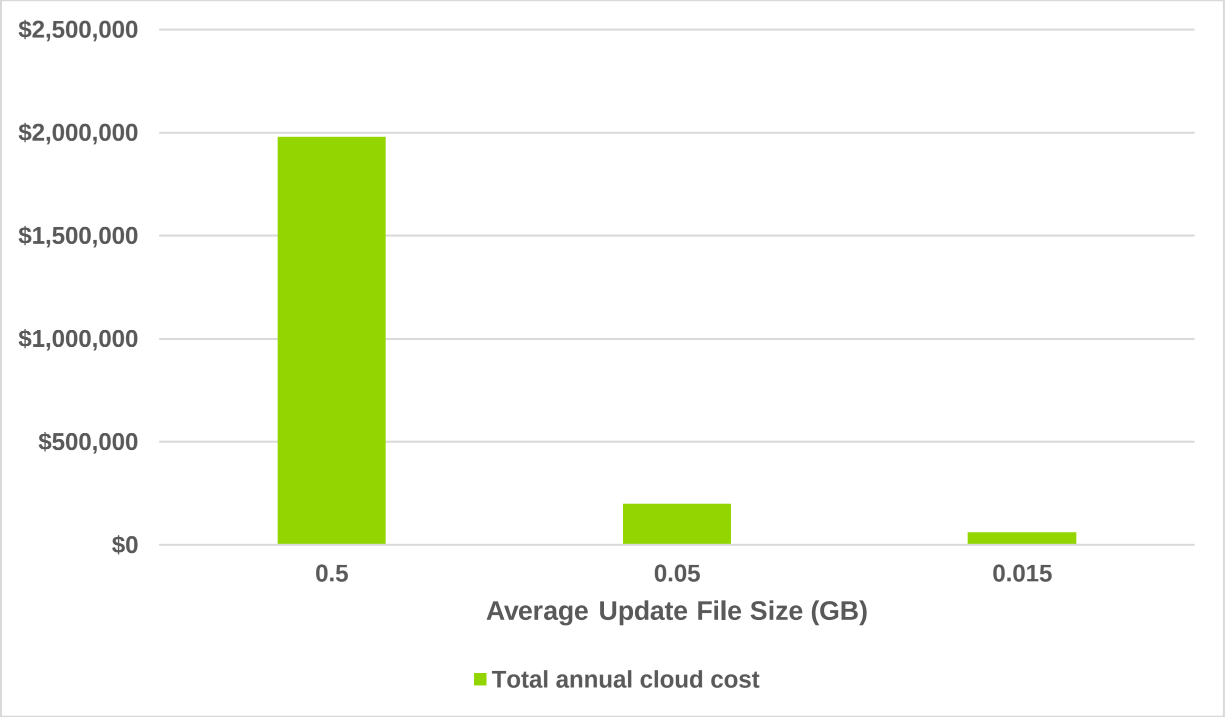
In addition to the vast waste of expensive flash memory resources, this redundancy method only allows for a system to rollback one version. In complex software systems with interdependencies between the multiple endpoints, a failure in ECU A may require ECU B to revert back two versions and ECU C to revert back three versions in order to maintain the integrity and compatibility of the entire system. This is not possible with the dual partition redundancy method.

The Line-Of-Code Update technology balances the need for safety and user experience with cost requirements by not overwriting the existing software that is written on the flash when performing the update (zero-downtime) and not requiring additional flash memory on the endpoint to enable reverting to previous versions. Instead, Line-Of-Code Update technology writes the fully executable update file to the next free space on the flash memory. This is repeated every time a new update file is available, never deleting the previous version. In this manner, should a software version fail, the ECU will be able to revert to any previous version, maintaining both ECU uptime and system integrity.

4. Cloud Storage Costs

Another major expense related to OTA updates is the cost of storing the actual update files in the cloud. Using the same example set from the previous section, these costs are not trivial. Cloud storage providers typically have two distinct charges associated with the service provided. The first is just the cost of static storage for the binary files. Three of the largest cloud providers average about $0.02 per GB per month for storage. Even if the maximum binary image update file size is 500 MB, once a quarter, this cost is minimal at less than $1 per year. Even in vehicles with 100 or more ECUs, most ECUs have much smaller image files at 1 MB or less each.

Chart 4 Annual cloud storage cost for various OTA update file sizes for a large ECU



(Source: Guidehouse Insights)

However, cloud providers also charge for egress of files from their platform. For the same three providers, the average egress cost is $0.099 per GB. This is in addition to the data transmission costs over the cellular network. In this case, the egress charges for 500 MB files would amount to $1.98 million annually. When the update size is reduced to 50 MB, the cost declines to $198,000 and just $59,400 if the file size is reduced to 15 MB, a 97% reduction. Cloud provider egress charges can be mitigated somewhat by using a content delivery network, but this approach adds its own cost and complexity to the process.

Table 5. Summary of Cost Comparisons for Cloud

Avg. Micro-processor Update Size	0.5 GB	0.05 GB	0.01 GB
Annual Cloud Cost ($)	**$1,980,000**	**$198,000**	**$59,400**
Avg. Micro-controller Update Size	420 KB	181 KB	55 KB
Annual Cloud Cost ($)	**$1,663**	**$715**	**$216**
Total for a full vehicle	**$19,949,691**	**$2,044,366**	**$613,460**

(Source: Guidehouse Insights)

Other Factors to Consider

Update File Creation Complexity and Costs

Today, software is increasingly developed using agile methodologies, replacing older waterfall methods. This has led to the use of Continuous Integration/Continuous Deployment (CI/CD) development toolchains such as Jenkins and a steady stream of new software versions. In the past, a new version of software would be released once every six to twelve months. The intent now is to have software updates released quarterly, if not monthly.

Legacy OTA update solutions compare binary files at the end of the development cycle to identify differences between source and target software versions. This is a cumbersome and costly method that is best suited for waterfall development. When issues are discovered, tracing them back to the source and ensuring that any dependent software modules haven't been impacted can be time consuming and costly in terms of engineering resources.

The CI/CD environment requires a solution that seamlessly integrates into the toolchain, reducing the integration complexity and cost. Integration into the development toolchain will automatically and continuously create an update file every time a new software version is created, further simplifying the update file creation process. In addition to the benefit of a seamless integration, the update technology will be comparing differences in the lines-of-code leading to much smaller update files than a binary image diff comparison to reduce the cost of storing and transmitting the update files.

Documentation and Homologation

Before any software is deployed to a vehicle it must go through precise and demanding quality verification procedures that include clear documentation of the changes in the software code, code functionality and the influence the code has on other parts of the vehicle systems. This is a time-consuming process but one that is necessary for certification, regulation, warranty and audits.

Line-Of-Code Update technology creates the update file by analyzing changes to the software functionality connections and behavior. In this manner, in addition to creating an update file, the LOC update technology also identifies and documents what has and has not changed between software versions. This information greatly streamlines documentation and homologation procedures. Validation of the effect of the software changes on the entire automotive E/E system and evidence of such is a requirement in the newly adopted UNECE WP.29 regulations.

Additionally, as the LOC update files are in the same standard file format as the rest of the automotive software, they will seamlessly fit with the existing documentation processes thereby reducing additional unwanted procedural costs.

Recommendations for an OTA Update Solution

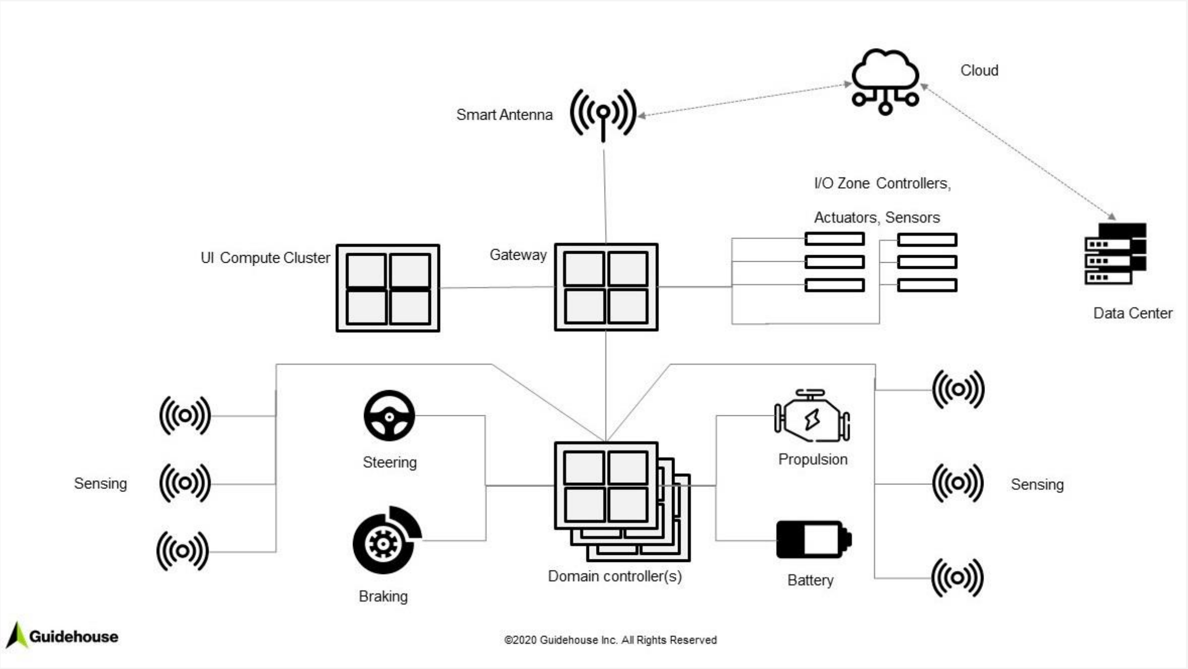
This section outlines specific approaches that can be taken to mitigate the costs associated with implementing OTA update capabilities for vehicles.

Management is Critical

The automotive environment is quite distinct from most of the other use cases where OTA updates have been deployed over the past 20 years. Even before Tesla began to deploy full vehicle OTA updates with the introduction of the Model S in 2012, other OEMs were performing OTA updates although the updates were limited to the TCU and later the infotainment head unit. Only recently have most OEMs started delivering updates to safety critical domains.

The solutions developed for updating the TCU or head unit are not typically adequate for updating other systems in the vehicle. Aside from displaying some vehicle information, the head unit is typically more isolated from many other systems, or at least it should be.

Figure 3 Block diagram of modern E/E architecture

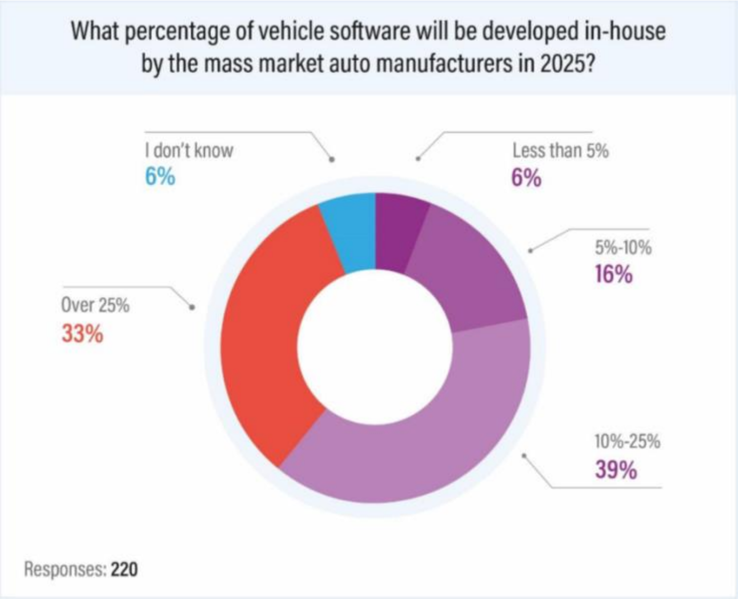


(Source: Guidehouse Insights)

The complexity of traditional automotive electrical/electronic (E/E) architectures and the safety critical nature of the target drives a need for reliable and robust update management solutions that are also cost effective. That complexity is a result of the piece-meal nature of automotive electronics development since the 1970s.

The dozens of ECUs coming from across the supply base often run a range of different compute architectures and software. Each new feature introduced often comes with its own ECU that must be managed. This is further amplified by the sheer volume of vehicles that may require updates as a result of deploying more than 90 million new vehicles annually. OEMs, led by new entrants such as Tesla and Lucid are now moving to consolidate the E/E architecture to relatively small number of higher performance domain controllers that power multiple systems with software from multiple vendors. However, even these consolidated domain controller architectures still require similar solutions in order to manage cost and reliability of updates.

Chart 5 Automotive Software Survey



(Source: Aurora Labs)

Even with the movement toward modern domain controller E/E architecture, most of the software running on those devices is still expected to come from suppliers. A recent survey of auto industry stakeholders revealed that two-thirds of respondents expect less than one-quarter of in-vehicle software to be developed in-house by 2025.

What to Do

Ideally, any solution should be fully integrated into the standard software development and validation tool chain of all contributors including in-house development teams and suppliers. It should be able to track all changes being made and generate the documentation required for complete traceability and verification. With new homologation rules coming into force in UNECE member countries and elsewhere, tracking of changes will be critical. Even in markets where there are no specific homologation requirements for software, such as the U.S., product liability is also a major concern and traceability can aid in investigating and understanding the root cause of issues that will inevitably occur.

A comprehensive update solution requires the following features and characteristics:

- Integration into existing software development and build workflows
- Automated documentation and evidence of changes
- The ability to generate the smallest practical distribution files by creating delta update files that won't require reprogramming the entire flash in an ECU
- Support for standard file formats including Intel Hex, S19 Records
- No requirement for on-vehicle client software support for updates
- Minimized vehicle downtime during updates

Since human developers are naturally prone to error, the ideal tool set will automatically analyze changes as are they are implemented in order to provide full traceability back to the source. In markets such as UNECE member states that require vehicle type approvals, full documentation of running changes is required in order get and maintain approval. A tool chain that helps to automate this documentation process can save both time and valuable engineering resources.

Automated analysis tools should understand all the pathways through the code and what has changed between versions. Integration of such tools into the build system should enable the ability to automatically generate update files that are focused only on the incremental changes. Automation of this process should provide more reliable and seamless generation of delta update files without missing connections to existing code.

Whatever tools are deployed should be capable of generating fully compliant update files in the usual formats such as delta.bin files that are standardized Intel Hex, S Records that bind uncoupled, non-related changes between binary files automatically. Software changes that are bound into one standard ELF file, can be quickly ready for distribution as an S19 record.

Optimizing the Customer Experience

Another of the many challenges in automotive OTA is downtime, especially for consumer vehicles. Consumers generally have the expectation that they can get in their vehicle and go at any arbitrary time. Finding that they might have to wait 30 minutes or more for an update to complete would not be considered acceptable by most vehicle owners. Commercial fleets often have scheduled downtime and updates can be coordinated with that downtime. However, even for those customers, extended periods where the vehicle may be disabled are generally unacceptable.

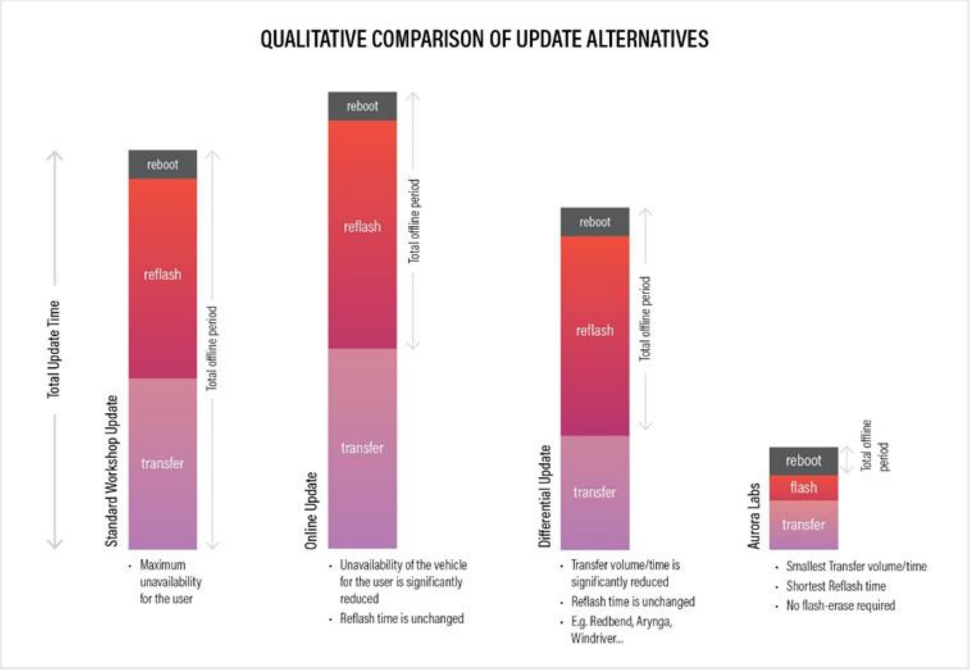
Solutions that create smaller incremental update files are a key factor in reducing both the amount of data to be downloaded but also reducing the time it takes to reprogram the ECU. Eliminating the need to read the entire original program into RAM, patch it in RAM, and then rewrite the patched version can be a tremendous time saver. A preferred solution should write the update file to the next segment of available storage and then link to it without ever taking the original offline.

Eliminating full rewrites of flash memory with smaller incremental updates that simply add rather than replace also improves the durability of the flash and reduces the risk of failure that would require costly hardware replacement.

Proof-of-Concept

A proof-of-concept (POC) with a leading European OEM compared a solution based on this approach with a full reflash in a garage/workshop, an online full reflash and other existing differential update technologies. The only offline time required for this approach was the reboot that occurred at the next key cycle.

Figure 4 Qualitative comparison of update methods as reported by a leading European OEM



(Source: Aurora Labs)

This particular evaluation with updates being loaded into the next available memory space demonstrated the ability to patch 20 version updates in a production body control ECU using the existing memory with no hardware updates required. In many cases this capability would be sufficient to support the entire lifespan of the vehicle without having to do a complete reflash or increase the amount of flash used in the ECU.

Since even highly automated systems may miss some errors and absolute security can never be guaranteed, a system as described retains each of the incremental updates stored on the ECU has another advantage. If anomalous behavior is detected either as a result of a logic error or exploitation of a security vulnerability, the potential exists to roll back to any prior version in real-time. This can be executed without a reboot or switching to an alternate memory partition.

Summary

Since the launch of the Tesla Model S in 2012, OTA updates have remained relatively uncommon. This, however, is starting to change and from 2022 onwards, OTA update capability will be regulated for all new vehicle introductions. This will significantly reduce the cost of software recalls, decrease recall completion time, increase recall completion rates, solve software security vulnerabilities, and enable the introduction of new software features.

Companies producing software development tools must take into consideration all the limitations and challenges of the embedded automotive architectures and the need to reduce OTA update costs and improve the user experience for both OEMs and customers. Line-of-Code Behavior update technology provides the opportunity to implement many or all of the recommended practices in this guide to address these challenges. Solutions that can provide these capabilities without the need for additional memory resources or a dedicated client integration are being demonstrated to be consistently the most cost effective and best suited for the current and future automotive software architectures with potential annual savings in the hundreds of millions of dollars.

Acronym and Abbreviation List

CD	Continuous Deployment
CI	Continuous Integration
ECU	Electronic control unit
E/E	Electrical/Electronic
LOC	Lines of code
OTA	Over-the-air
OTC	Over-the-cable
POC	Proof of concept
SoC	System on a chip
SDP	Full-Service Deployment Platform
TCU	Telematics Control Unit
UNECE	United Nations Economic Commission for Europe

Published April 2021

©2021 Guidehouse Inc.
1800 Tysons Blvd., 7th Floor
McLean, VA 22102 USA
www.guidehouseinsights.com

Guidehouse Inc. (Guidehouse) has provided the information in this publication for informational purposes only. The information has been obtained from sources believed to be reliable; however, Guidehouse does not make any express or implied warranty or representation concerning such information. Any market forecasts or predictions contained in the publication reflect Guidehouse's current expectations based on market data and trend analysis. Market predictions and expectations are inherently uncertain and actual results may differ materially from those contained in the publication. Guidehouse and its subsidiaries and affiliates hereby disclaim liability for any loss or damage caused by errors or omissions in this publication.
Any reference to a specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not constitute or imply an endorsement, recommendation, or favoring by Guidehouse.