加速しよう、未来を。
VERISERVE

ベリサーブナビゲーション

Veriserve Navigation

Veriserve Navigation

【巻頭特集】

ソフトウェアが自動車の価値を決定付ける「SDV」の現在・過去・未来を識者が語る

電気通信大学　名誉教授
株式会社アイシン　元取締役
新 誠一氏

【特別寄稿】

Open SDV Initiativeの目指すもの

名古屋大学
未来社会創造機構 モビリティ社会研究所 特任教授
二宮 芳樹氏

2025
September
Vol.26

加速しよう、未来を。
VERISERVE
https://www.veriserve.co.jp

本　　社　東京都千代田区神田三崎町 3-1-16 神保町北東急ビル 9 階
西日本支社　大阪府大阪市西区阿波座 1-4-4 野村不動産四ツ橋ビル 3 階
中 部 支 社　愛知県名古屋市東区泉 2-27-14 関電不動産高岳ビル 13 階

Veriserve Navigation

ベリサーブナビゲーション
2025 September **Vol.26**

ベリサーブは、日本のメーカーやソフトウェアベンダーが作るプロダクトやサービスを対象に、ソフトウェア品質の向上を支援する会社です。自動車のソフトウェア化と共に当社のモビリティ事業は拡大し、現在では全事業の過半を占めるに至っています。
当社はソフトウェア品質に長年向き合っています。加えてモビリティ領域での豊富なソリューション実績を武器とし、引き続きサービスの向上に進めてまいります。

自動車メーカー各社は、近年の自動車のソフトウェア化を最大限に生かすため、より市場へのリリース間隔を短縮すべく、ソフトウェア開発の短スパン化を目指されていると思います。そのような開発スタイルにおける品質ソリューションを、当社が扱っているツール群を活用いただくことで応えていけないかと考えています。

常務執行役員 モビリティ事業本部 本部長
東 弘之

短スパン開発に応えるため、現在、TERUS※とConTrackという二つのツールを軸とした品質ソリューションを提案しています。TERUSは市場の声を短時間でより詳細に分析するツール、ConTrackは使いやすく外部システムとの連携も行いやすいトレーサビリティ管理ツールです。これらを活用いただくことで、スピードを持って開発サイクルを回す一助となればと思っています。

今後とも、お客様の求めていることを迅速に実現するような品質ソリューションを提供してまいります。引き続きご愛顧のほど、よろしくお願い申し上げます。

※TERUS:インターネット上に分散しているさまざまな顧客体験データを自動収集し、独自の顧客体験分析AIを活用してお客様の声を抽出・可視化することで、お客様の総評や感情の起伏などを具体的かつ高精度に分析するツール。

Contents

今回の表紙には、特集テーマ「モビリティ最前線」に関連し、自動車や技術の進歩を象徴する「進」の文字を選びました。
「進」には「先進」や「前進」といった意味が込められ、新しい技術への挑戦とその活用を表現しています。
「ベリサーブナビゲーション」は、高度化するソフトウェアの品質の確保や改善につながる情報を発信することで、関係業界の活性化と日本のソフトウェア開発競争力向上に貢献してまいります。

ソフトウェアが自動車の価値を決定付ける「SDV」の現在・過去・未来を識者が語る

自動車業界は現在、「CASE（Connected、Autonomous、Shared＆Service、Electric）*1」を旗頭にした、メカからエレキという「100年に1度の大変革期」に突入しています。これまでの自動車は、もっぱらメカの機能や性能が製品価値の大きな要素でした。しかし、現在は、エレキを一歩進めてソフトウェアによって短期間に新たな価値をユーザーに届ける「SDV（Software Defined Vehicle）」が注目されています。そして、SDVの実現のためには、車載ソフトウェアの開発や品質保証におけるパラダイムシフトが欠かせません。

＊1：自動車の未来を示す概念で、「Connected（接続）」「Autonomous（自動運転）」「Shared & Services（共有・サービス）」「Electric（電動化）」の頭文字を取ったもの

こうした自動車業界における新たな潮流が意味するところやもたらされる価値、そしてソフトウェア開発・品質保証の世界に与えるインパクトなどについて、電気通信大学 名誉教授で、株式会社アイシン 元取締役の新 誠一氏に聞きました。

電気通信大学　名誉教授／株式会社アイシン　元取締役／
キャノンメディカルシステムズ株式会社　先端研究所 元所長／
一般社団法人水道情報活用システム標準仕様研究会　会長／
技術研究組合制御システムセキュリティセンター　元理事長

新 誠一氏
Shin Seiichi

プロフィール

東京大学大学院修了。電気通信大学名誉教授。専門は制御工学、システム工学で、計測自動制御学会の元会長を務め、経済産業大臣賞など受賞多数。株式会社アイシンの取締役、キヤノンメディカルシステムズ株式会社 先端研究所所長として、産業界の技術革新と発展に貢献。また、水道情報活用システム標準仕様研究会会長として社会インフラの DX も牽引するなど、学術と産業の両面で幅広く活躍。

OTAを実現するための三つのチャレンジ

――新先生は既に10年以上前から、現在のSDV時代の到来を予期してさまざまな提言を行っていました。あらためてSDVの現状について見解をお聞かせください。

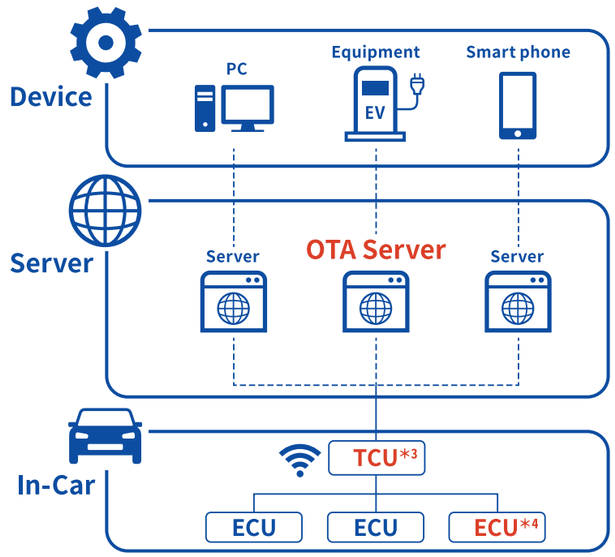
SDVを理解する上で極めて重要なキーワードの一つに、「OTA（Over The Air）*2」があります。これは無線を介して製品のソフトウェアをリモートアップデートする技術を指す用語ですが、PCやスマートフォンの世界では既にOTAでOSやアプリケーションをリモートアップデートすることは当たり前になっています。ここに来てようやく自動車の世界でも、OTAによる車載ソフトウェアのアップデートが実用化されつつあります。

＊2：無線通信で車両のソフトウェアを更新する技術

――なぜ自動車の世界でOTAが求められるようになってきたのでしょうか？

最大の要因は「品質問題」です。今や自動車の大半の機能は数千万行規模のソフトウェアで制御されています。何らかの問題が生じた際には、修正したソフトウェアを自動車にインストールしなくてはなりません。この作業を現在はディーラーで整備士が手作業で行っています。ただでさえ整備士が足りないといわれている今日、もし大量のリコールが発生すればあっという間に工場がパンクしてしまいます。そこでOTAで人手を介さずにソフトウェアをアップデートすることで、ソフトウェア起因の品質問題に少ない人手でも迅速に対処できるようになります。

以上は販売側の事情ですが、顧客側の意識も変わってきました。ソフトウェアはハードウェアに比べて寿命が短く、新しい機能を出してもすぐに陳腐化してしまいます。そこで新しいソフトウェア機能をいち早く顧客に提供して、自社製品の価値を保つため



図表1　OTAのシステム構成

＊3：TCU（Telematics Control Unit）車外とネットワーク経由にて通信するための装置
＊4：ECU（Electronic Control Unit）車両を制御する小型コンピューターの名称

にも、OTAによるソフトウェアアップデートが極めて重要になってきます。

――このOTAを実現する上で、車載ソフトウェアの開発や品質保証にはどのようなことが求められているのでしょうか？

大きく分けて三つの技術分野で、新たなチャレンジが求められています。

まず一つ目は、構造化プログラミングからオブジェクト指向プログラミングへの転換です。これまでの車載コンピューターは、自動車の機能ごとに専用のECU（Electronic Control Unit）が取り付けられており、それぞれを制御するソフトウェアが個別に実装されています。これらのソフトウェアは主にC言語を使った構造化プログラミングで実装されていましたが、ECUの数が増え過ぎてしまい、OTAには向きません。100個規模の多数のECUで個別に制御するのではなく、少数のチップセットによってまとめて制御する方向に進んできています。上位のソフトウェア開発もライブラリーからの機能の切り貼りに変化してきましたので、オブジェクト指向プログラミング、具体的にはC＋＋言語での実装スキルおよび開発環境が必須になってきました。

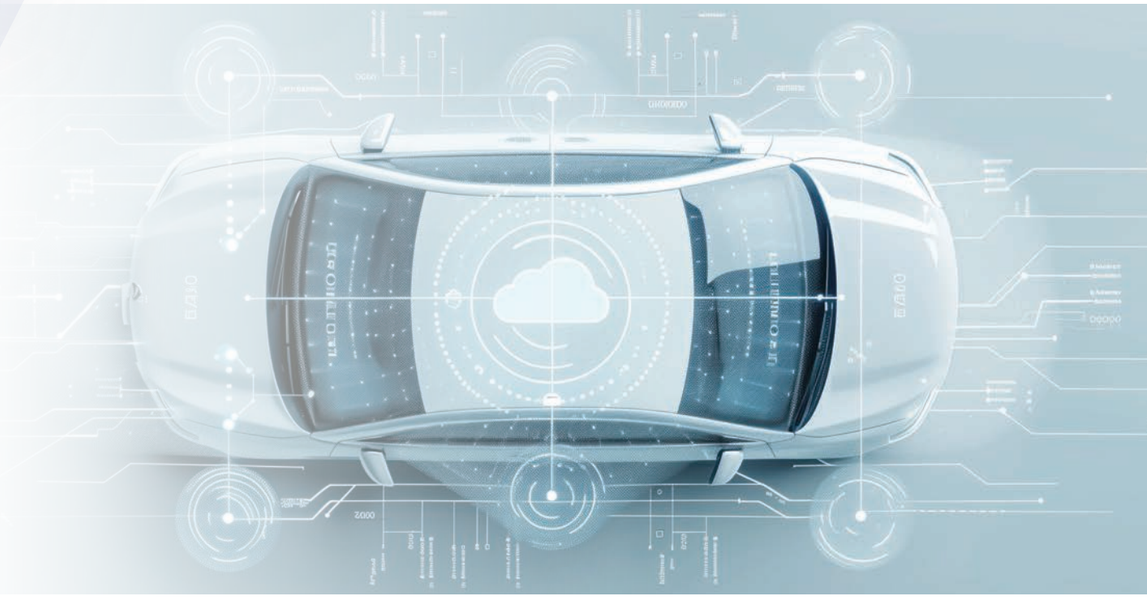
現在の車載ソフトウェアはCADツールで開発し、実装コードは自動生成するスタイルが主流なので、開発エンジニアが新たな言語を習得する必要性はあまりないかもしれません。しかし、開発ツールベンダーやテストベンダーにとっては大きなチャレンジになると思います。

二つ目は、車載コンピューターの通信プロトコルの変化です。かつて主流だったCAN（Controller Area Network）[*5]ではOTAによる大量のソフトウェアアップデートには対応できません。徐々にイーサネットへの転換が進んでいきます。それは、新しい可能性を拓くとともに、新たな脅威の扉を開くことになります。認証、暗号化は当たり前、侵入検知、車載ソフトウェアの開発者やテストエンジニアはこのトレンドにも対応していかなくてはなりません。

＊5：車両内のECU同士が通信するためのネットワーク規格

そして三つ目が、「AUTOSAR Adaptive Platform[*6]」への対応です。これまで車載ソフトウェアの標準インターフェース規格として「AUTOSAR Classic Platform」がデファクトスタンダードとして長らく用いられてきましたが、現在ではOTAにも対応した新規格のAUTOSAR Adaptive Platformが急速に普及しつつあります。従って車載ソフトウェアの開発やテストの現場でも、この新規格に対応した手法が求められるようになってきています。

＊6：次世代の自動車向けソフトウェアアーキテクチャ。特に高度な運転支援システム（ADAS）や自動運転機能のような動的で複雑なアプリケーションに対応するために設計されている



今日のSDVの新潮流へと連なるオブジェクト指向開発の歴史

――OTAをはじめとするSDVの新潮流に対応するためには、車載ソフトウェアエンジニアにもパラダイムシフトが求められるわけですね。

その通りです。そして、ソフトウェア開発の過去の歴史をきちんと把握することによって、この新潮流の本質をより深く理解できるようになります。車載ソフトウェアに先立って、一般のソフトウェア開発の世界では、既に1990年代から「OMG（Object Manage-

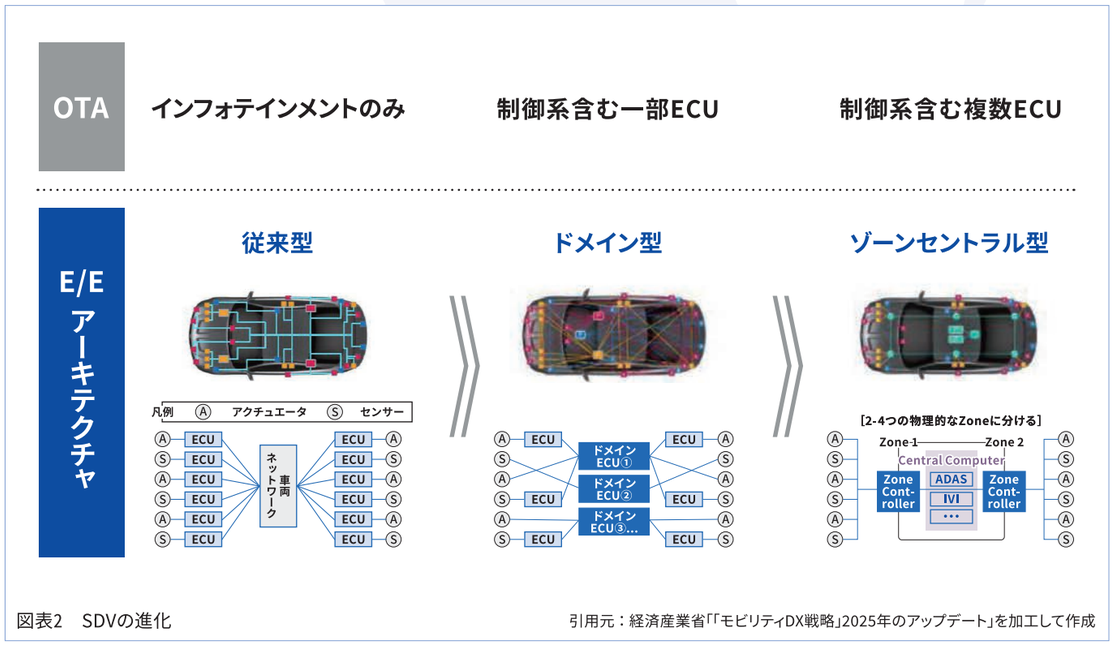
ment Group）*7」という標準化団体を中心に、オブジェクト指向ソフトウェアの標準化作業が進められてきました。

＊7：標準的なソフトウェア仕様を策定する国際的な団体

それまでのコンピュータソフトウェアは、全ての機能モジュールが一枚岩で実装された「モノシリック」なアーキテクチャが主流でした。しかし、それではソフトウェアの開発効率やメンテナンス性に制約が生じるため、モジュール同士がオープンなインターフェース仕様を介して自由に連携できるアーキテクチャが提唱されるようになりました。こうしたニーズに応える形でOMGが策定した規格が「CORBA（Common Object Request Broker Architecture）*8」であり、米マイクロソフトが開発した「COM（Component Object Model）*9」「.NETアーキテクチャ*10」でした。

＊8：異なる環境のソフト間で連携するための通信仕様
＊9：ソフトの部品を再利用・連携させるための技術仕様
＊10：Windows上で動作するアプリ開発の統合基盤

その後、これらのアーキテクチャが広く普及した結果、現在サーバーやPC、スマートフォンなどの分野では、ソフトウェアのモジュールを個別に入れ替えたりアップデートすることで、機能を進化させたり不具合を修正することがすっかり当たり前になりました。そしてこの流れが、今日ようやくSDVという形で自動車の世界にも取り入れられようとしているのです。



図表2　SDVの進化

引用元：経済産業省「「モビリティDX戦略」2025年のアップデート」を加工して作成

――こうした流れは、ソフトウェア品質保証の世界にどのような変化をもたらすとお考えですか？

頻繁にソフトウェアが更新されるようになれば、当然ソフトウェアのテストも頻繁に行わなくてはなりません。そうなればベリサーブのようなテストプロセス全般をサポートする企業の負担も急増するため、テストの自動化・省力化の取り組みは急務だと思います。しかも車載ソフトウェアの場合、「特定の運転シーンでのみ発生する問題」も数多く発生しますから、それらをいかに再現・特定するかという課題も新たに持ち上がってきます。

さらには、セキュリティの問題もこれからますます深刻化してきます。昨年から自動車のセキュリティに関する規制が新たに設けられ、車載ソフトウェアの脆弱性を狙ったサイバー攻撃への対処は今後必須になってきます。自動車のセーフティを守るという観点では、ソフトウェアの品質管理だけでなくセキュリティ対策もより重要性を増してくるはずです。

未来のSDVは「ハードウェアのアップデート」も視野に

――ここまでSDVの過去と現在について伺ってきましたが、これからSDVはどのような姿に進化していくとお考えでしょうか？

SDV時代になると、自動車の価値のかなりの部分がソフトウェアによってもたらされるようになりますから、自動車のハードウェアを保有するインセンティブが薄れてきます。そのため、CASEでいうところの「Shared＆Service*11」のビジネスモデルが将来的には主流になると考えられます。既にこれを見越して多くの自動車メーカーが、自動車を販売する従来のビジネスモデルから、自動車をリースしたり、月額課金や従量課金のサブスクリプションサービスで自動車を提供したりするビジネスモデルへとシフトし始めています。

＊11：モビリティを所有から共有へ、製品からサービスへと転換することで、より効率的で持続可能な移動社会を実現するための概念

――そうしたビジネスモデルの具体例としては、どのようなものがありますか？

国内の自動車メーカーの中では、やはりトヨタ自動車の取り組みが群を抜いています。トヨタは既に2019年から、「KINTO」と呼ばれる新車のサブスクリプションサービスの提供を始めています。これは月額のリース料を支払うことで、トヨタの新車を安価に乗り続けられるというものです。サブスクリプションモデルでより多くのユーザーにトヨタ車に乗り続けてもらうことで、最終的には維持・メンテナンスサービスで収益を確保しようというのが同社の狙いです。

これは、プリンター機器メーカーが実践しているビジネスモデルと類似しています。プリンターのハードウェア製品を赤字覚悟の低価格で販売して、その後のトナーや紙の売り上げで利益を確保する。これと同様のビジネスモデルを自動車メーカーが志向しており、KINTOはその先頭を走っています。

――ソフトウェアのアップデートによる維持・メンテナンスや機能強化などが、このようなビジネスモデルの鍵を握るわけですね。

その通りです。加えてKINTOでは、ソフトウェアだけでなく、ハードウェアもアップデートすることで自動車の価値を維持・向上させることができます。2023年から提供が始まった「KINTO Unlimited」というサービスメニューを利用すると、メーカーオプション装備などもアップデートできるようになっています。

このように、将来的には、ソフトウェア、ハードウェア双方の製品価値を継続的に訴求していくビジネスモデルが、自動車業界だけでなくあらゆる分野で主流になるはずです。そうなるとテストベンダーは、新しいソフトウェアだけでなく、アップデートされたハード

ウェアプラットフォームにも都度対応していく必要が出てくるでしょう。

ソフトウェア品質保証が担う役割と責任はより重く

――このようにソフトウェアの役割がどんどん広がり、アップデートも頻繁に行われるようになっていくと、その開発や品質保証に関わる人々にかかる負担は重くなっていく一方では？

現在あらゆる製品でソフトウェアの役割が拡大し、その規模が肥大化した結果、ソフトウェアの全体像を把握するのが極めて困難になりました。例えば、PCのOSに関して、かつてWindows XPのソースコードが4,000万行といわれており、一人の人間が全てを把握するのは不可能でした。今日の車載ソフトウェアのソースコードは約1億行といわれており、しかも先ほど申し上げたようにチップセットで機能を集約する方向に進んでいます。そのためソフトウェア全体として予期せぬ挙動や不具合を発生させてしまう事態を、完璧に予知することは事実上あり得ません。

――そこでソフトウェア品質保証の取り組みの重要性が増してくるわけですね。

はい。エンジニアがソフトウェアの全体像を把握しきれなくなってしまったために、開発時にやむなく取りこぼしてしまう品質上の問題を、ベリサーブのような第三者検証企業が実施するテストによってあらかじめつぶすことで、初めてSDVの品質を担保できるようになります。

ただし、どれだけ入念にテストを行ったとしても、ユーザーが製品を利用するシーンを100パーセント漏れなくテストケースで網羅することはできません。そこで重要になってくるのが、万が一不具合が発生してしまった際に迅速に問題を修正して製品をアップデートするための仕組みであり、OTAはまさにそのための技術なのです。

既に車載ソフトウェアの開発作業は、ツールの進化によってかなり省力化されています。同様に、テストに関しても近年は自動化・省力化が徐々に進んできましたが、今後は「ツールで自動的に開発されたソフトウェアの品質をいかに担保するか？」という新たな課題にも直面することになります。ベリサーブをはじめとするQA・テスト専門企業には、ぜひこうした課題の解決にも取り組まれることを期待しています。

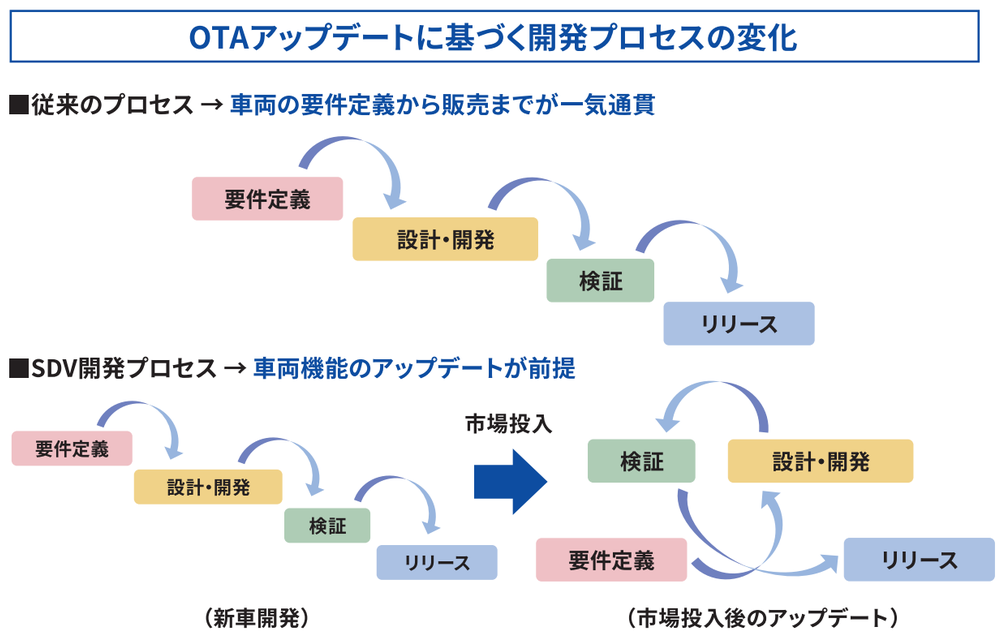
アジャイル開発を行う際は「機能安全」の確保が大前提

――このように継続的にソフトウェアをアップデートし、短い間隔でリリースしていく開発スタイルには、「アジャイル開発手法」が適しているといわれています。しかしその一方で、アジャイル開発における品質管理にはまだまだ課題が多いという指摘もあります。

アジャイル開発は従来のウォーターフォールと比べて自由度が高く、頻繁にソフトウェアをアップデート・リリースしていくSDVのビジネスモデルとは親和性が高いと言えます。ただし、開発の自由度やスピードを高める一方で、品質や安全性も同時に担保しなくてはなりません。

製造業の世界では、これを「機能安全」という概念で表します。もともと製造業では、機械が安全に作動することを担保する「機械安全」の取り組みを重視してきたのですが、製品の中にコンピューターがどんどん入り込んでいったことで、コンピューターで機械の作動を監視して安全を担保する機能安全の考え方が徐々に主流になっていきました。例えば、自動運転の分野では、AIが決して危険な運転操作を行わないよう、「絶対にこのような操作を行ってはならない」という約束事項を「ガーディアン」として定

OTAアップデートに基づく開発プロセスの変化



図表3　OTAアップデートに基づく開発プロセスの変化

引用元：経済産業省「モビリティDX戦略」2025年のアップデート」を加工して作成

義することで安全を担保しています。

ソフトウェアを開発する際も同様に、「このような動作をしてはならない」というガーディアンをあらかじめ定義して、かつそれらが開発ツールによって確実に実装される仕組みを用意しておくことが重要です。そのような環境をしっかり整えることによって、アジャイルによる高速で柔軟なソフトウェア開発と、ソフトウェアの安全性の担保を両立できるようになるでしょう。

――これはセキュリティ対策についても当てはまる話ですね。

そうですね。今やあらゆるソフトウェアにとってセキュリティ対策は必須ですから、ソフトウェアを開発している最中にうっかり脆弱性を作り込んでしまわないよう、ツールやプロセスでセキュリティを担保できる環境を整備することが重要です。これからの車載ソフトウェアは、OTAを介してネットワークとどんどんつながっていきますから、外部からサイバー攻撃を受けるリスクが格段に高まります。そのため、外部からのアクセスを無条件に信頼せず、全てのアクセスに対して必ず厳格な認証やアクセス制御を行う「ゼロトラスト」の考え方に立ち、セキュリティ対策を実装していかなくてはなりません。

これは安全性についても同様で、「誰かが安全を担保してくれるだろう」と無条件に信頼するのではなく、自分たちが開発するソフトウェアが危険なラインを絶対に越えないよう、厳格に監視・管理する仕組みを作り込んでいく必要があります。

あらゆるモノが自律的に移動する「モビリティ」の時代に

――昨今の自動車業界では、「モビリティ」という言葉も頻繁に使われるようになりました。

モビリティという概念は極めて広範な対象を含んでいて、最終的には自動車に限らず「あらゆるもの」を移動することで人々の生活を便利にしようという考え方です。最も分かりやすい例は、キッチンカーです。これはもともと「“お店”を動かしたい」というアイデアから考案されたものですが、社会インフラの分野でも同じようなアイデアが既に具現化されています。

例えば、給水車です。これによって、災害で水道インフラが破損してしまった地域や、水道インフラが老朽化してしまった過疎地などに、速やかに水を届けられるようになります。同様に、電気インフラを移動させることができる電源車や、臨時の携帯基地局を設置するための移動基地局なども、モビリティによってインフラを移動させるための仕組みです。


キッチンカー

給水車


電源車


キャンピングカー

図表4　モビリティの拡張例

民間の分野では、例えば近年キャンピングカーがはやっていますが、これはまさに居住や生活に必要なインフラを移動させるためのモビリティの仕組みに他なりません。また自動車レースの世界では、各チームが「トランスポーター」という大型トラックの中にレーシングカーや機材など必要な設備を詰め込んで、全国のサーキットを移動しながらレースを戦っています。これもまた、モビリティの典型的な例だと言えます。

――私たちの身近なところで、既にモビリティは着々と浸透しつつあるのですね。

今挙げたのは小規模で分かりやすい例ですが、今後はさらに大きなスケールでモビリティが進展していくでしょう。例えば、工場や製油所、浄水場などを移動させるような仕組みが実現したり、ビルを丸ごと移動させるようなアイデアも出てきたりするでしょう。大事な点は、これらのものが単に指示した通りに移動するだけでなく、自律的に判断して自動的に移動するようになるということです。車の自動運転はその典型例ですし、倉庫内の運搬やピッキングを自動化する「自動倉庫」のような仕組みは既に広く実用化されています。

モビリティと聞くと、得てして「エンジン車がEVに置き換わる」「車がインターネットとつながる」といったミクロな視点にとどまりがちですが、今申し上げたように、モビリティによる革新は社会を大きく変えていく可能性を秘めています。ソフトウェアの開発や品質保証に関わる方々は、このような大変革に今後いかに対応していくか、今まさに問われているのではないでしょうか。

特別寄稿

Open SDV Initiativeの目指すもの

名古屋大学
未来社会創造機構 モビリティ社会研究所 特任教授
二宮 芳樹氏
Yoshiki Ninomiya

プロフィール

1983年名古屋大学大学院工学研究科修士課程終了。工学博士。1983年株式会社豊田中央研究所　技師を経て、2004年同社走行支援センシング研究室室長、2011年同社安全・情報システム研究部部長、自動運転や運転支援システムの開発に従事。2014年名古屋大学未来社会創造機構 特任教授（～現在）、同大学COIモビリティ部門長、2015年株式会社ティアフォー取締役（兼業、現在フェロー）。2024年Open SDV Initiative AD-ADAS WGリーダー（～現在）。自動運転の社会実装とそのための課題解決に従事。自動車技術会会員。

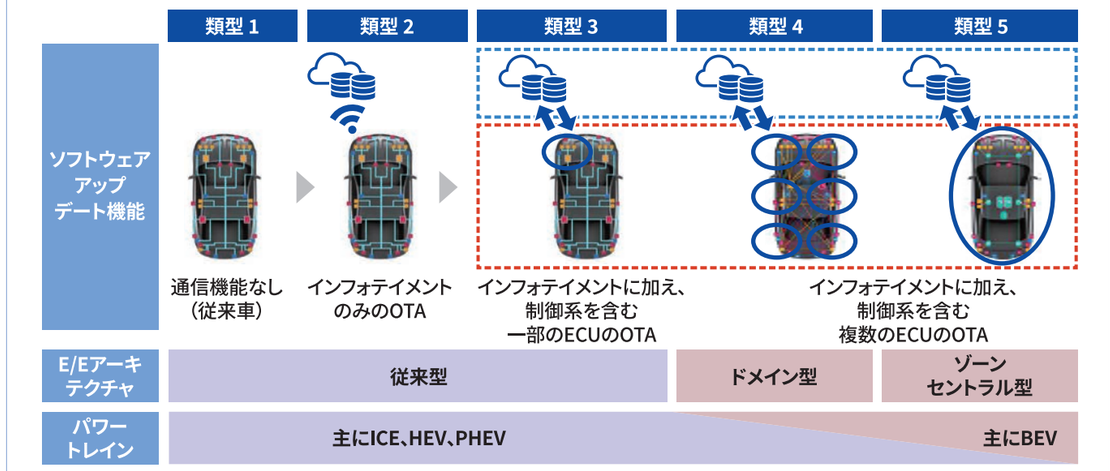
1. はじめに

2010 年代から自動車業界を取り巻く状況変化を表すキーワードは、「100 年に一度の大変革」といわれた CASE であった。すなわち Connected（車両とインターネットとの接続）、Automated（自動運転）、Shared（カーシェアリング）、Electric（電動化）によって、車の在り方そのものの変革といわれてきた。そして最近、それに加わり自動車業界をにぎわしているキーワードが SDV である。筆者は、これまで自動車会社の研究所で 30 年間自動運転（AD：Automated Driving）や運転支援（ADAS：Advanced Driver Assistance Systems）の研究開発を行い、その後大学で 10 年間自動運転の社会実装の研究を行い、同時に大学発の自動運転のスタートアップを立ち上げてきた。そして、この 1 年は SDV（Software Defined Vehicle）の標準化に向けたコンソーシアム Open SDV Initiative に参加し、AD/ADAS 領域における SDV について研究している。本稿では、AD/ADAS 領域の長年の経験から SDV を論じてみたい。

2. SDVとは

2.1. SDVの定義

SDVは 直訳すると「ソフトウェアで定義された車」となる。車は2000年以前からエンジンも含めてECUでソフト制御されているので、SDVを「主にソフトウェアで動作させる車」と解釈するのは正しくない。答えはSDVの代表例といわれているTeslaが教えてくれる。Teslaはビジネスモデル、開発・生産方式、設計を全て0から新発想で作られたEV（Electric Vehicle）である。TeslaがSDVといわれる理由は大きなディスプレイなどのコックピットデザインが先進的だからではなく、OTA（Over-the-Air）によってディーラーに行かなくても遠隔で性能向上が即座に行える点といわれる。具体例としては、制動距離が長いという雑誌レポート後の即座の制動距離改善、オプションとして充電1回の走行距離が拡大、自動運転性能の大幅な向上などがある。以上からSDVの定



図表1　SDVの定義と類型

出展：経済産業省第4回モビリティDX検討会資料を加工して作成

図表2　SDVで提供されるサービス例


自動運転が機能アップ（Tesla）


OTAによる最新状態維持（BYD）


サードパーティーアプリケーションの魅力（Tesla）


機能やサービスをカスタマイズ（NIO）

義は、「SDVとは、ソフトウェアのOTAによって販売後に機能を拡張・変更できる車」ということになる。ちなみに日本の車の国策を議論する経済産業省のモビリティDX検討会でのSDVの定義は、「制御系ソフトウェアをアップデート可能なOTA機能を搭載した車両」となっている（図表1の赤枠部分）。

SDVにはどのようなうれしさがあり、どのように自動車世界を変えるのだろうか。ユーザーにとっては「ソフトウェア更新による性能向上により、価値が購入後にも向上する車」であり、「ユーザーの欲しい機能をソフトウェアで実現できるカスタマイズが可能な車」となる。メーカーにとっては「車の価値を上げるソフトウェアで収益が上がる」、「サードパーティーなどの別業界からの参入を促し、車の価値がさらに上がる」ということになる。SDVは「車のスマートフォン化」ともいわれる。通信機器だった携帯がスマートフォンになってビジネスの世界が大きく変わったことが、車でも起き始めている。

2.2. SDVの現状

SDVは米国Teslaや中国の新興NEV（New Energy Vehicle）ですでに実用化され、世界で最も生産されている車はSDV車となった。中国市場では新型車はSDVであることが当たり前になった。スマートフォンのようにアプリケーションを選択、追加できるコックピットが作られ、ソフト更新で機能アップする自動運転機能が提供されている（図表2）。

しかし、OTAが可能なこれら車のEE（電気・電子）アーキテクチャーを踏襲するだけではSDVはできない。SDVのもう一つの要点は、開発速度が企画から市場投入まで約１年少しと従来とは一線を画している点である。従来の車両の開発は、仕様を決めて、垂直統合で時間をかけて緻密に擦り合わせすることで達成された。このプロセスは、開発期間がかかり、機能変更や向上は容易でない。さらにこの開発スタイルでは、多くの車種を提供している自動車会社では、車種の数だけの膨大な開発リソースがかかることになる。SDV化はソフトウェア開発を車両のハードウェアの開発サイクルや車種から切り離すことにより実現する。以上からSDVは、車開発スタイルの抜本的な変革と表裏一体であることが分かる。SDVは、車をスマートフォン化するビジネスの変革であり、同時に開発・運用スタイルの変革である。この潮流に乗り遅れることは、SDVだけでなく、車作りからも遅れを取ることになる。

SDV化の遅れは、日本の重要産業である自動車の敗北となる可能性があるので、経済産業省の主催するモビリティDX検討会では、SDVを国の重要目標と位置付け、国家目標として「2030年のSDVのグローバル販売台数における日系シェア3割を実現」としている。日本は自動車会社の数が多く、裾野も広い産業なので、会社ごとにこの改革を独立に行うことはハードルが高い。携帯や家電で起きたことが自動車で起きないようにするには、連携し、協調領域を適切に設定することが肝要となる。

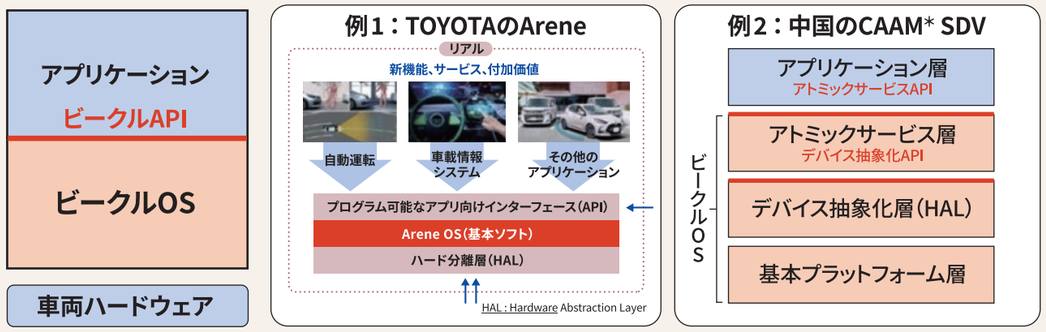
2.3. SDVの課題

SDV化は、車のスマートフォン化であり、車開発スタイルの抜本的な変革である。特に従来からの車産業にとっては開発スタイルの革新が大きな課題となる。

開発スタイルの革新を実現するためのポイントは以下のように整理できる。

1）ソフトウェアとハードウェアの分離
2）クラウドベースの仮想開発環境の導入
3）DevOpsの考え方の導入
4）AI技術の活用

この中で本解説において注目しているのが、ソフトウェアとハードウェアの分離である。従来の車開発はハードウェアを基軸に機能開発が行われ、車両ごとに専用のソフトウェアが統合されるため、開発サイクルはハードウェアの開発と一致して長期的なものになっていた。ソフトウェアとハードウェアの分離は、車に分散していたECUを集約・再編してその上に車全体を制御できるビークルOS（ソフトウェアプラットフォーム）を構築するこ



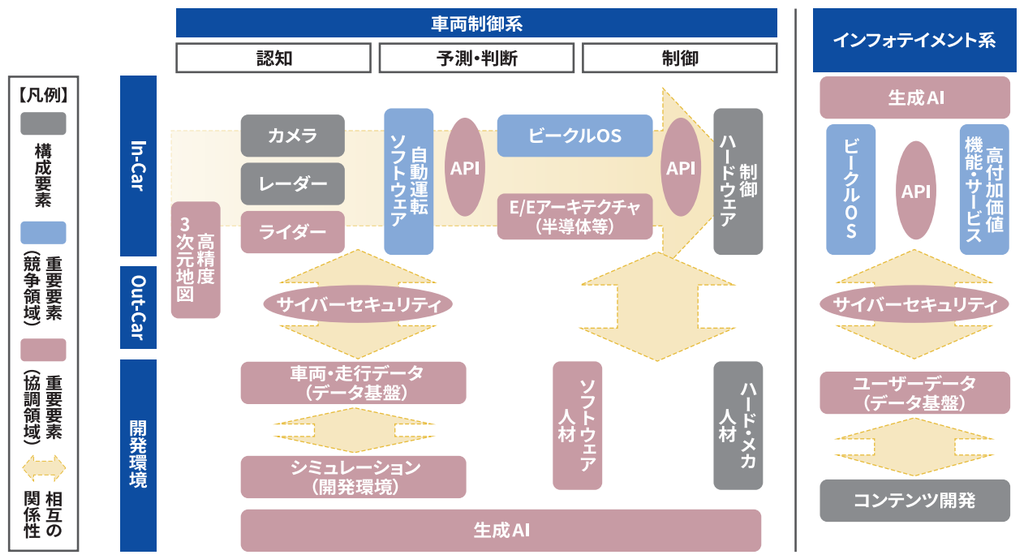
図表3　ビークルOSとAPI

とで実現された。ビークルOSは車両ハードウェアを隠蔽し、アプリケーション、ユーザーが効率的に、安全に利用、開発するためのソフトウェアである（図表3）。APIはビークルOSをアプリケーション、ユーザーが利用するためのアプリケーションプログラミングインターフェースである。この構造により、アプリケーションは車両のハードウェアと分離され、同じソフトウェアが別のハードウェアでも動作できるようになる。

図表3の具体例の一つは自動車会社がビークルOSを開発してAPIを設定している例（トヨタのArene OS）であり、もう一つは、中国自動車工業協会（CAAM）の提案するSDVのアーキテクチャーである。このAPIを設定することで開発の効率化や水平分業化が進み、標準化すれば業界全体でさらに加速される。

2.4. SDVの標準化の動き

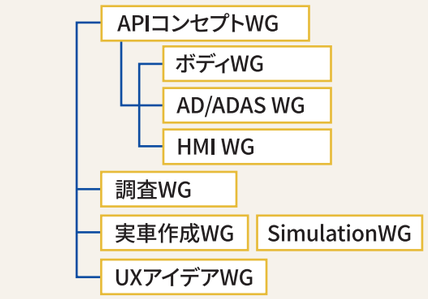
Teslaや中国の新興自動車会社に比べると、従来からの自動車会社は既存のサプライチェーンや開発法からの刷新は容易でない。また日本には多くの自動車会社があり、裾野の広い産業なので、会社ごとにこの改革を独立に行うことは効率が悪い。携帯や家電で起きたことが自動車で起きないようにするには、自動車会社間が連携し、協調領域を適切に設定することが肝要となる。経産省モビリティDX検討会では日本のSDV戦略として、協調領域を幅広く設定すると共に、ビークルOSのAPIの標準化を目標に掲げている（図表4）。



図表4　SDVにおける協調領域

出展：経済産業省第4回モビリティDX検討会資料を加工して作成

図表5　Open SDV Initiativeの組織体制



ビークルOS/APIの**全体アーキテクチャー策定**
ボディ ドメインのAPIの策定
AD/ADAS ドメインのAPIの策定
HMI ドメインのAPIの策定
世界の**APIの調査と連携**
API効果実証のための実車、 Simulation環境構築
サードパーティーアプリケーションの**アイデア創出**

そしてそのAPIの標準化を行う日本の組織として、経産省から名前が挙がったのが、JASPAR（Japan Automotive Software Platform and Architecture）とOpen SDV Initiativeの二つとなる。SDVに関するグローバルな標準化を推進する団体にはCOVESA（The Connected Vehicle Systems Alliance）、Eclipse、SOAFEE（Scalable Open Architecture For Embedded Edge）、AUTOSAR（AUTomotive Open System Architecture）、CAAMが存在するが、APIの標準化については、COVESA/AUTOSARとCAAMで行われており、まだ完成度の高いものにはなっていない。それらを下敷きに、よりオープンかつ技術軸での提案を狙う。

3. Open SDV Initiative

3.1. Open SDV Initiativeとは

Open SDV Initiative（OSDVI）は、2024年10月に名古屋大学の高田 広章教授の主導の下で発足したコンソーシアム型の研究組織である。参加機関は2025年6月末には60社となっている。JASPARは自動車会社や部品会社を中核メンバーとする団体であるのに対して、大学中心に幅広い関連業界で構成されるOSDVIは、サードパーティーのSDV分野の参入や、AD&ADAS領域のようにAPI策定に時間がかかる部分の先行的な役割を期待されている。

3.2. OSDVIの組織構成と目標

OSDVIの組織体制を図表5に示す。調査WGが世界のAPIの調査を行いながら、APIコンセプトWGが全体アーキテクチャーを策定し、さらに三つのドメインごとのWG、ボディ、AD/ADAS、HMI WGがそれぞれのAPIを構築する体制になっている。

またAPIの有効性を検証するための実車/Simulationのチームも組織化している。さらにOSDVIでターゲットとするアプリケーションのアイデアを検討するUXアイデアWGも設置した。OSDVIで想定しているサードパーティーの参入も視野に入れている

図表6　Open SDV Initiativeで想定するアプリケーション

1	ボディ系	セントリーモード、洗車モード
2	AD/ADAS 系	自動運転、自動駐車、バレーパーキング
3	HMI 系	メーターのパーソナライズ
4	情報提供系	ナビゲーション 、POI情報提供、旅行情報提供
5	エンタメ系	ドライブ記録、車両活用ゲーム、自動運転中のエンタメ
6	プローブ系	ドライブレコーダ、テレマ保険、安全運転診断／教育
		ストリートビュー、地図作成と活用（インフラメンテ）
7	その他	シェアリングカー管理、V2H、V2G

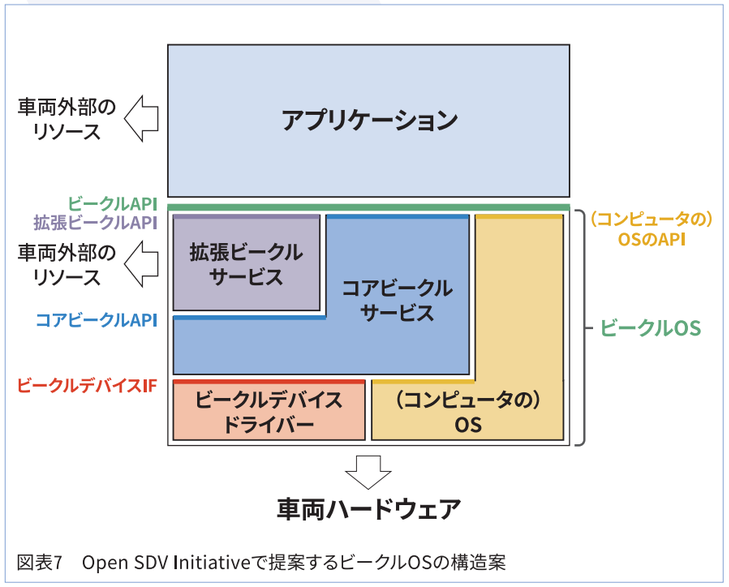
POI : Point of Interest
V2H : Vehicle to Home,　V2G : Vehicle to Grid

アプリケーション例を示す（図表6）。

OSDVIの目標は、標準に資するSDVのAPI仕様案を提案することである。マイルストーンとして2025年3月31日に、構築した暫定仕様を提案している（https://www.nces.i.nagoya-u.ac.jp/osdvi/index.html）。

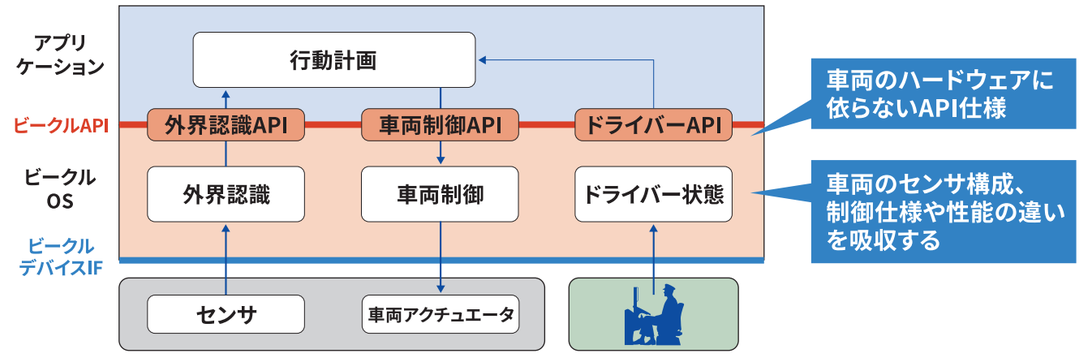
3.3. 提案したビークルOSとAPIの概要

OSDVIのAPI仕様提案の中核となるビークルOSの構造案（概念図）を図表7に示す。ビークルOSの構成要素をコンピュータのOS、ビークルデバイスドライバー、コアビークルサービス、拡張ビークルサービスとした。ビークルOSは、車両のハードウェアの



図表7　Open SDV Initiativeで提案するビークルOSの構造案

図表8　AD/ADAS領域での機能配置例



隠蔽と、アプリケーションの開発の容易性の実現を担う。コアビークルサービスは、車両のハードウェアを管理し、**1）車両のハードウェアの違いの吸収、2）車両のハードウェアのアクセスの調停、3）（可能な範囲での）安全性の確保の役割**を持つ。コアビークルAPIはコアビークルサービスを利用するためのAPIになる。拡張ビークルサービスはアプリケーションの開発を容易化するためのソフトウェアサービス群である。例えば、地図やV2Xなどの車両のハードウェア以外のリソース管理もここが担当する。アプリケーションが他のアプリケーションからの要求を受ける場合などにはそのアプリケーションは拡張ビークルサービスの位置付けになる。コアビークルサービスと拡張ビークルサービスを総称してビークルサービスと呼ぶ。ビークルデバイスドライバーは、車両のハードウェアデバイスをそれぞれ単独で抽象化するための階層である。

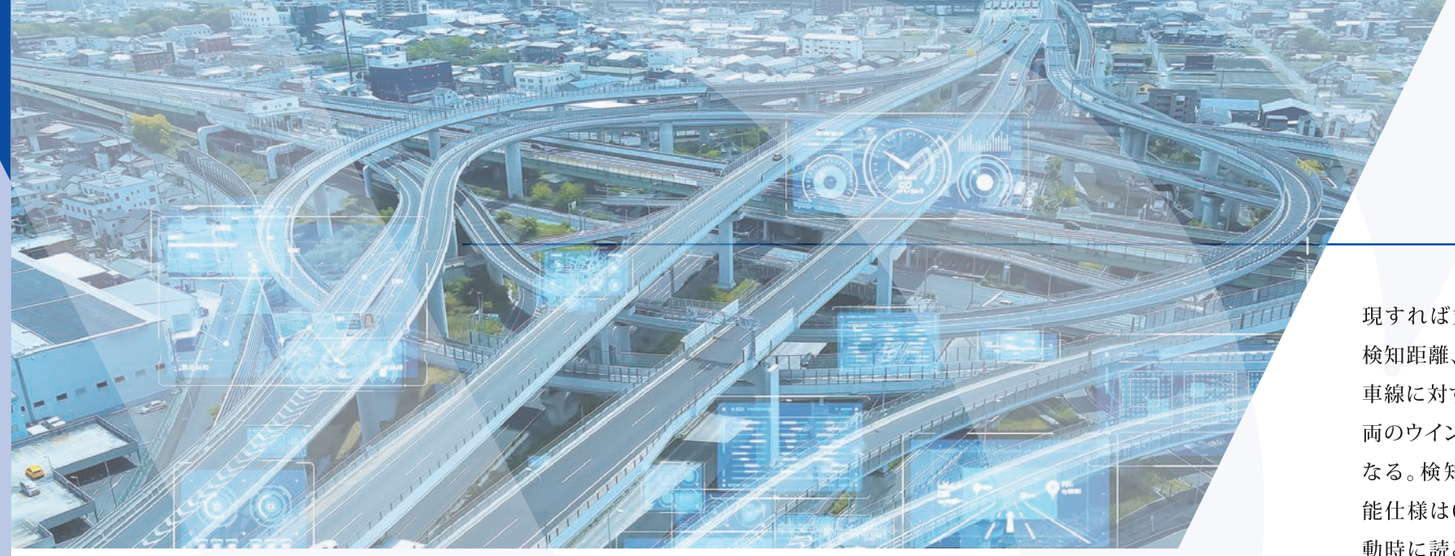
4. AD/ADAS領域のAPI

4.1. AD/ADAS領域のAPI概要

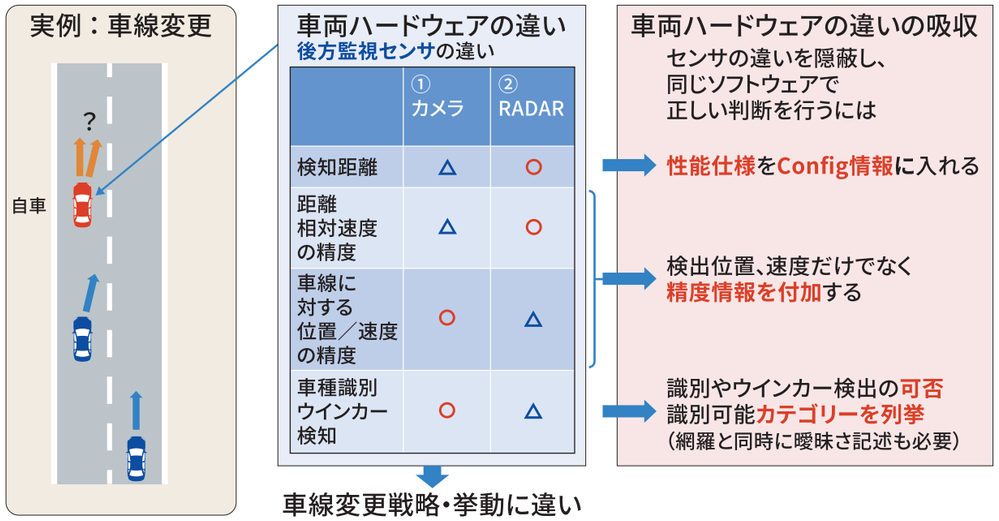
AD/ADAS、すなわち自動運転/運転支援領域におけるAPIを具体例として、SDVのAPI仕様の要件を明らかにする。AD/ADASは運転の自動化であり、人の運転でいうところの認知、判断、操作に対応して、外界認識、行動計画、車両制御という情報処理で構成される。外界認識には車両に搭載されるセンサが接続され、車両制御には車両のパワートレイン、ブレーキ、ステアリングを動かすアクチュエータに接続される（図表8）。車両のハードウェアに関係する部分とそうでない部分に整理すると、車両のハードウェアに依存する外界認識と車両認識はビークルOSに配置でき、周囲の走行状況から加速、減速、車線変更などを判断する行動計画がアプリケーションに配置できる。こうすれば、行動計画の部分はハードウェアと切り離すことができ、同一ソフトウェアを別の車両にも使えるようになる。標準のAPIに合わせて外界認識、行動計画、車両制御などのモジュールを作れば、さまざまなモジュールを切り替えて利用することも可能になる。

4.2. APIの仕様の要件

AD/ADAS領域の代表的なアプリケーションであるALKS（Automated Lane Keeping System：自動車線維持システム）を実例として、その時に必要になるAPIやビークルOSの要件を考察する。ハードウェアの違いとしてまずセンサの違いを考えてみる。センサの違いをビークルOSで吸収し、センサによらないAPI仕様にするには、APIはどのセンサでも共通となる認識結果を表現すればよい。



図表9に示すALKSの自車が車線変更の場合には、後続車の存在やその位置や速度が認識結果になる。ここでセンサがカメラの場合とRADARの場合を考えてみる。センサの違いによる認識結果の一般的な違いは、表に示すように、カメラは横方向の精度（特に道路に対する精度）が高く、車両の予測に使えるウインカーなどの情報も得られる。一方、RADARは検知距離が長く、距離や相対速度の精度が高いという特長がある。これらの特性の違いにより、センサが異なれば車線変更の戦略や挙動も異なることになる。APIに必要なのは、どちらの場合も行動計画で最良な判断ができるように、APIにセンサの特性の違いを表現することである。これはモデルベース開発の考え方に従い、認識結果を汎用的な周囲環境モデルとして表現し、その周囲環境モデルのパラメーターでセンサの特性の違いを表現すれば対応できる。この例では、検知距離、距離や相対速度の精度、車線に対する位置精度、検出した車両のウインカーなどの属性がそれになる。検知距離のような静的な性能仕様はConfig APIでシステム起動時に読み込まれる。



図表9　センサの違いを吸収するAPIの要件

これは、従来の擦り合わせでの開発過程で必要とした情報をあらかじめモデル表現し、ハードウェアと切り離すモデルベース開発と見なせる。開発は効率的になり、性能低下も抑えられる。ハードウェアの違いとしてセンサを例に説明したが、ECU性能であれば処理時間の違いになり、処理時間をConfig情報とし、遅れ時間保証のためのタイムスタンプ情報が必要になる。また、車両性能やサイズの違いであれば、性能や諸元をConfig情報とし、APIをハードウェアから切り離すためには、制御指令値に、目標加速度や目標曲率などの車両ハードウェアに依存しないものを選択する必要がある。

4.3. APIの構築状況

AD/ADAS領域のAPIは前節で示した考え方の基で、すでに世界で提案されているCOVESA、CAAMやそれらが参照しているISO23150、さらに日本のV2Xの実験仕様であるRC-019、JASPARの制御仕様などを参考にし、現在の標準は変更不要であればできるだけ準拠する形で以下の四つの仕様の策定を行い、公開している。

- **周辺環境モデル（Surround Model）**
- **車両制御（Motion Control）**
- **自車位置・方位（Current Location）**
- **ドライバー状態（Driver Model）**

5. まとめ

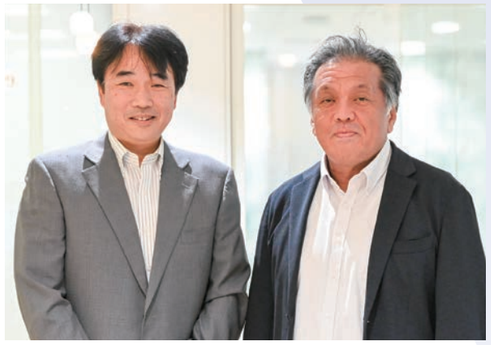
2024年10月に発足したOpen SDV Initiativeでは、日本のSDVのAPIの標準に資する仕様の策定を目標とし、2025年3月末に最初の暫定仕様案を策定・公開している。今後は、半年ごとをめどにその改定を行うと共に、実車やSimulatorを用いた有効性評価・検証を行う予定である。OSDVIの活動の前提は、日本がSDV化で勝ち抜くには、SDVにおける協調領域を拡大し、開発スタイルの革新をいち早く実現するための戦略への賛同と合意であり、この啓発・普及活動と連携して、このAPI標準化活動が有意義なものになると信じて進めていきたい。

SDV時代の車載ソフトウェア品質を支える

CI/CT×AIによる自動テスト環境構築

近年、自動車業界では「SDV（Software Defined Vehicle）*1」という新しいパラダイムが急速に注目を集めています。当社は、SDV時代に求められる「迅速」「高品質」「多様」な開発を下支えするパートナーとして、最先端のテスト自動化やAI技術の導入を進めています。本稿では、SDV時代の現場課題を整理し、当社が取り組む自動テスト環境およびAI活用の事例をご紹介します。

＊1：クラウドとの通信により、自動車の機能を継続的にアップデートすることで、運転機能の高度化など従来車にない新たな価値が実現可能な次世代の自動車



左：荒井 秀夫　右：植木 雄一

荒井 秀夫 Hideo Arai

株式会社ベリサーブ　モビリティサービス開発部

自動車部品メーカーで、ヒューマン・マシーン・インターフェースなど電子電装部品のシステム開発、新技術導入に従事。2023年、株式会社ベリサーブ入社。自動車メーカーの先行開発業務でプロジェクトマネジメント、AIを活用したデータ分析、要件定義に携わる。2025年よりモビリティ向け新サービス開発を推進中。

植木 雄一 Yuichi Ueki

株式会社ベリサーブ　モビリティサービス開発部 技術部長

自動車部品メーカーで、車載ソフトウェア開発領域における新技術導入、開発効率化・品質向上、グローバル人財育成などに従事。2023年より株式会社ベリサーブに入社し、SDV時代に対応したAIを活用した自動テスト環境構築など、モビリティ向け新サービス開発を推進中。

1 近年の車載ソフトウェアを取り巻く環境変化

1.1 SDV・CASE時代の到来と車載ソフトウェアの新潮流

自動車業界ではCASE*2やSDVといったキーワードが急速に浸透しています。コネクテッド化や自動運転技術の高度化により、「車両の価値や機能がソフトウェアで決まる」という本格的なソフトウェア・ファースト時代が到来しつつあります。

このようなトレンドの中、自動車業界では、

- **従来にない速度での仕様変更や機能追加**
- **プラットフォーム統合やFOTA（Firmware　Over-the-Air）*3導入によるリリースサイクルの短縮**
- **より高度かつ多様化した車載アーキテクチャの採用**

など、従来と比べてはるかに高度な情報管理や品質保証が求められるようになっています。

こうした環境変化の中、当社は長年にわたり車載ソフトウェア品質領域で培った知見とノウハウを生かし、「柔軟性」「スピード」「品質」を兼ね備えた継続的なテスト手法の開発に取り組んでいます。

＊2：自動車の未来を示す概念で、「Connected（接続）」「Autonomous（自動運転）」「Shared ＆ Services（共有・サービス）」「Electric（電動化）」の頭文字を取ったもの

＊3：無線通信で車両のソフトウェアを更新する技術

1.2 ソフトウェア開発現場が直面する品質リスク

車載ソフトウェアの開発現場は、前述のSDV・CASE化の流れを受けて大きな品質リスクと、それに伴う新たな課題に直面しています。

開発規模の急増と複雑化

車載システムが担う機能は年々拡大し、従来よりも多くのソフトウェア資産を、より短期間で高い品質で開発する事が求められています。

また、機能の連携範囲が拡大し、ソフトウェアの階層構造やモジュール分割がより細分化・高密度化するとともに、機能間のインターフェースや依存関係が複雑化しています。

ソフトウェア構造の多様化

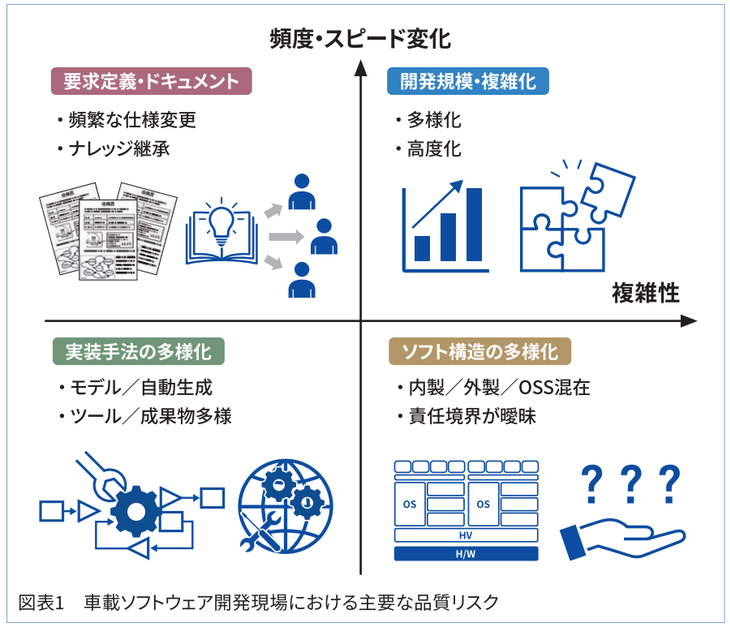
これまでは内製での開発が中心だったものが、外部サプライヤーやOSSを活用した外製ソフトの組み込みなど複数ベンダー混在型へと進化した結果、構造の複雑化や責任境界の曖昧化といった新たな管理課題が浮上しています。

実装手法の多様化

モデルベース開発や自動コード生成、パラメータ調整（コンフィグ）を反映したソフト自動生成など、新たな開発アプローチの導入が進んでいます。これにより品質管理はより高難度なものとなり、ツールや成果物のトレーサビリティ確保が重要な論点となっています。

要件定義の複雑化やドキュメント内容の曖昧化

多様で変化の激しい市場要求に応じるため、頻繁な要求仕様変更への



図表1　車載ソフトウェア開発現場における主要な品質リスク

迅速な対応が求められています。これに伴い、短期間での技術習得や設計業務に対するプレッシャー、そして人財育成・ナレッジ継承も大きな課題です。また、こうした急激な変更の中でドキュメントの内容が十分に整理・更新されないまま運用されるケースが増え、設計意図の曖昧化や伝達ミスといった新たなリスクも顕在化しています。これによって、後工程での解釈違いや手戻りの発生、品質低下などのリスクが高まっています（図表1）。

2 品質保証のために必要となるソフトウェア開発基盤の変革

前章で述べた車載ソフトウェア開発を取り巻く環境変化とそれに伴うリスクに対し、単なる部分最適や従来型の改善活動では対応が困難になっています。持続的に高品質・高効率なものづくりを追求するためには、開発体制・ツール・プロセスそのものを抜本的に見直す“基盤の変革”が不可欠であると考えています。

従来は、開発フェーズごとに個別最適化されたプロセスや、各社のノウハウ、やり方・手順を品質保証として実行する傾向にありました。一方、SDV時代の到来、すなわち開発規模の急拡大、システムの複雑化、頻繁な仕様変更、多様なステークホルダーとの協業といった新たな環境に適応するには、品質保証の考え方も「全体最適」「継続的なテスト」「自動化・見える化」を念頭に置いた設計・運用がより求められるようになっています。

このため、近年は

- **CI/CT（継続的インテグレーション／継続的テスト*4）**
- **トレーサビリティ管理（成果物や変更履歴、プロセスの一貫追跡）および自動化を活用した品質管理**
- **ナレッジや情報基盤のデジタル化・一元化**

といった「新しい開発基盤」へ進化することが業界共通で求められています。

本章では、こうした現場ニーズを踏まえ、いかに品質保証を“基盤レベル”から変革し、安定的で俊敏な開発を実現するか、そのポイントと、当社の知見に基づく実践的な要素をご紹介します。

＊4：ソフト開発で自動的に統合・テストを繰り返す手法

2.1 CI/CTの意義と現場にもたらす価値

CIやCTは、コードやテスト資産を持続的に統合・テストすることで、品質リスクを“早期”かつ“継続的”に捕捉することのできる仕組みです。とはいえ、実際の車載・組み込みソフトウェアの開発現場では、新機能追加や大規模仕様変更のたびに、「まずは個別に手早く機能テストを進めたい」というニーズは根強く、全てのテスト資産を即座にCI/CT環境へ移行することは必ずしも現実的とは言い切れません。

特に新規機能や改修が発生した初期段階では、その機能単体を現場で直接テストした方が、開発スピードやテストリードタイムの観点からも合理的とも考えられます。

しかし、ここで得られたテスト資産やノウハウを、「蓄積・標準化」して徐々にCI/CT環境へ組み込んでいくことで、次の開発サイクル以降は同一機能や類似機能のテストが自動的かつ効率的に実行できるようになります。

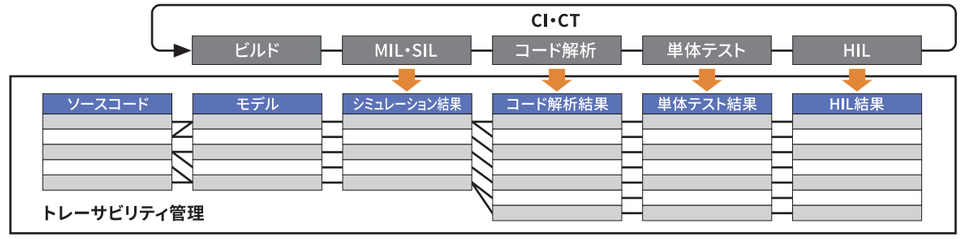
こうした「段階的導入・資産化サイクル」こそが、車載・組み込み領域で現実的かつ効果的なCI/CT活用スタイルであると考えます。

結果として、“単発・個別のテスト”の手間を、徐々に“全体の継続的な品質保証プロセス”へ昇華し、現場負担や属人化リスクを減らしつつ、資産の共有・再利用（再テスト性）も高められる。これらがCI/CTがもたらす最大の価値となります。

2.2 トレーサビリティと自動化がもたらす効果

複雑化と多様化が進む車載ソフトウェアの開発現場において、トレーサビリティ管理は、品質保証の基盤としてますます重要な役割を担っています。機能や仕様、テストケース、設計・実装などの各種成果物が複雑に絡み合う中、「どの要件がどの設計・テストやソースに結び付いているのか」を“見える化”し、容易に追跡できる仕組みが不可欠です。また、頻繁な仕様変更や大量の成果物管理、ドキュメントの改訂といった現場作業の負担も無視できないものとなっています。こうした中、トレーサビリティ管理の効率化・自動化は、現場の生産性維持と品質確保の両立に欠かせないと考えます。

図表2　CI/CTプロセス×トレーサビリティ管理の全体像



とりわけ、CI/CT環境下では、ビルドやテストが日々高速で繰り返され、大量の成果物が次々と生成されます。このような状況下で、全ての情報を手作業で管理し続けることは現実的ではありません。そのため、トレーサビリティ管理ツールと自動連携し、各種プロセス結果や関連成果物を自動的に集約・履歴一元化する仕組みが、効率的かつ確実な品質保証には必須となっています（図表2）。

3 当社の最新の取り組み：自動テスト環境構築事例

当社では日々高まるCI/CT導入ニーズに応えるべく、自動テスト基盤サービスを開発・進化させてきました。本章ではその全体像と、有効性・独自性について紹介します。

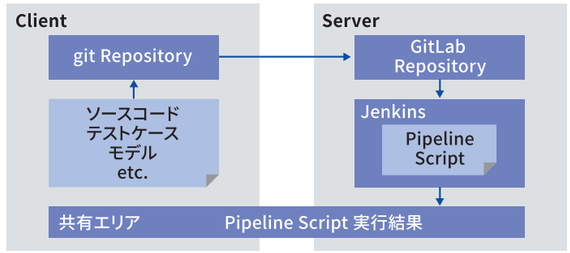
3.1 サービス全体像（アーキテクチャ・特徴）

当社による自動テスト環境は、クライアント側で作成・管理されるソースコードや各種テスト資産を、中央リポジトリ（GitLab）に一元的に集約することから始まります（図表3）。これにより、複数拠点や関係部署間での資産共有・バージョン管理が効率化され、品質保証の基盤強化と作業分担の明確化が実現できます。

サーバー側ではCI/CTパイプラインツール（Jenkins）と連携し、ビルド・各種テストの自動化と、実行結果の迅速な集約・活用までを一貫して実現。全ての成果情報は「共通エリア」に格納され、プロジェクト関係者がアクセス可能となります。これにより管理負荷や属人化リスクを減らし、迅速なフィードバックを可能としています。

主な構成要素とそれぞれの役割

図表4に示す通り、本開発基盤はバージョン管理、CI/CTパイプライン、テスト自動化スクリプト、結果共有機能、AI自動判定（詳細は4. AI画像認識による表示コンポーネント自動テスト事例　参照）などの構成要素から成り立っており、それぞれが現場の効率化と高品質化を支えます。



図表3　自動テスト基盤のアーキテクチャ概要

図表4　主な構成要素と役割

構成要素	主な技術／ツール	役割・特徴
バージョン管理	Git, GitLab	ソースコード・テスト資産の一元管理
CI/CTパイプライン	Jenkins, Pipeline	継続的インテグレーションと自動テスト
テスト自動化スクリプト	Shell, Python等	繰り返し実行、標準化
結果共有・可視化エリア	専用共有サーバー／DB	テスト結果の迅速な集約と活用
AI自動判定	OCR, 機械学習等	HMI／画像認識等、従来困難領域の効率化

3.2　取り組みの特徴

当社の取り組みにおいては、実運用で広く使われている外部ツール（Jenkins, GitLabなど）と、AI活用を組み合わせることで、現場適応性と将来拡張性を両立しています。

さらに、要件定義からテスト結果まで「トレーサビリティ」を確保。作業負荷や人依存を減らし、各個人のノウハウや、過去のテスト・検証など、現場に蓄積されるナレッジをシステムで一元化。これらをプロジェクト横断で標準化した手順を共有し活用する仕組みを構築することで、属人化の防止や工数削減、品質の平準化・向上を実現しています。

本取り組みでは、HILS（Hardware In the Loop Simulation）*5実行Scriptによる「テスト実行」「画像解析（AI）」「自動評価」がシームレスにサイクル化され、テスト中にエラーや異常が発生した場合でも即座に状況を検知・把握でき、迅速なフィードバックすることを可能としています（図表5）。

＊5：実機とシミュレーションを組み合わせて制御を検証する手法

図表5　HIL自動テストにおける実行・画像解析・自動評価サイクル

4　AI画像認識による表示コンポーネント自動テスト事例

近年の車載コックピット内のHMI（ヒューマン・マシン・インターフェース）*6の進化は目覚ましく、特にADAS（Advanced Driver-Assistance Systems）*7や燃費性能などディスプレーを活用した情報表示の重要性が増しています。一方でソフトウェアの短期開発サイクルや、従来の人の目視に頼ったHMI評価では習熟不足によるヒューマンエラーなどによるリスクが増加してしまう課題があります。

本項では、AIの物体検出技術に着目しHMIの表示テストへの導入することで、効率的でかつ品質の高いテストを実現することを目的として取り組んでいる事例を紹介します。

＊6：人と機械が情報をやりとりする操作・表示の仕組み

＊7：運転支援機能で安全性や快適性を高める技術

AIの物体検出技術は、自動運転向けに障害物の検出や交通量・人流の測定、生産現場では品質検査など、多くの分野で活用されています（図表6）。

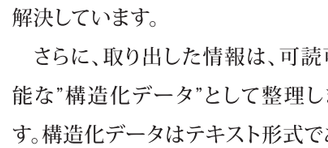
具体的に物体検出とは画像の中から”特定”のものを見つけ出し“それが何か?”、“どこにあるか?”を特定する技術です。この技術を用いてHMIのテストに当てはめると以下のようなテスト実行時の判定を自動化することができ、HMIには非常に適した技術であると言えます。

- **今、画面に何を表示しているのか?**
- **表示している場所は正しいのか?**

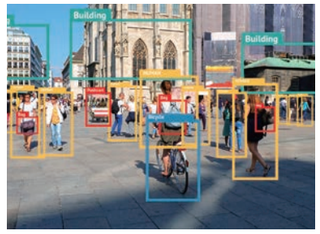
さらに、光学文字認識OCR（Optical Character Recognition）*8技術を組み合わせて使用することで、画面に表示している文字、数字を構造化データとして扱えるため、精度の定量的評価や、「期待値」や「設計仕様」との比較・判定を行うことができます。

- **数字の精度は正しいか?**
- **表示している文章は正しいか?**
- **文字化けやスペルミスがないか?**

＊8：画像内の文字を認識してデジタル化する技術



図表6　AIの物体検出技術の例



障害物の検出／識別

人物の検出／識別

HMI表示テストの自動化の課題

従来の自動化の取り組みでは、カメラでHMIに表示した画像を取得し期待している画像が表示されているか否かを画像のRGB値を基に期待値と比較し判別しています。しかし、この手法だと外光や反射などの外乱や、製品に採用されるグラフィックICの性能や表示デバイス（Display）の画素数・諧調特性の違いといった制限や制約により、画像を比較し判定するために多くのチューニングに時間を費やすといった課題があります。また、図表7のように車両からの信号を模擬しその時の表示の挙動をカメラで自動的に記録することはできても、その時に表示している内容が何かを、カメラ画像を人が目視で確認する必要があり、判定の自動化に課題が残ります。

AI画像解析とK-means法などによるハイブリッド構成での課題解決

当社の取り組みでは、物体検出技術およびOCR技術を用いて、表示しているものが何か、どこにあるか、文字・数字は何を示しているかといった情報を取り出し、その後、色を識別するための後処理をハイブリッド化することで目視に依存していた課題を解決しています。

さらに、取り出した情報は、可読可能な”構造化データ”として整理します。構造化データはテキスト形式であり、期待するものと比較が容易、かつ人による判断が介在しないため自動化を可能としヒューマンエラーを防止することができます（図表8）。

例　Air Bag警告灯、スピードメータの場合の構造化データ

物体検出とOCRの構造化データは、従来目視で取得していた「それが何か?」、「どこに表示しているか?」、「数値・文字は何か?」それぞれの情報が“Object”、“Left/Top/Width/Height/CenterX/CenterY”、“OCR_text”としてテキスト形式

図表7　人の目視に依存するため自動化が困難な事例

図表8　物体検出および、OCRの情報を構造化データに整理

構造化データ

物体検出								OCR_	色解析						
Object	Left	Top	Width	Height	CenterX	CenterY	Confidence	OCR_text	R2	G2	B2	Hue	Satulation	Value	Color
Air Bag	43.56	262.44	89.78	77.09	88.44	300.99	0.9995223	ー	243	67	61	1	0.71	0.95	Red
Speed	898.72	381.81	267.81	226.18	1032.62	494.90	0.9979836	138	255	255	255		0	1	White

で整理され期待値との照合を実現しています。

テスト結果の集約

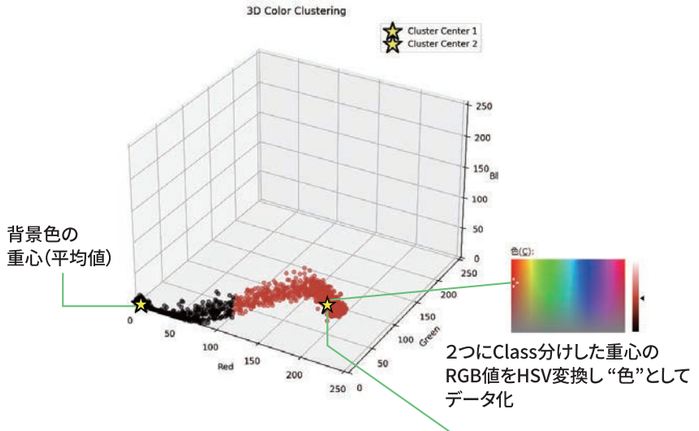
表示色情報の抽出

表示色の情報抽出では、前項の課題で示したように表示デバイスの分解能などの影響で、画像の境界線などが中間色となり、検出した物体の色を特定することが困難なケースがあります。本事例では、色の解析にk-means法*9を用いて色をClass分けすることで、分布の集約結果を基に物体の表示色を判定する手法を採用しました(図表9)。

表示色解析の結果についても解析結果を構造化データとして、“色”(例:赤)として抽出することができ、期待値として設定した色と照合できます。

*9:データを特徴に基づき複数のグループに分類する手法

これら自動化されたHMIテストの結果を、全て一元的に構造化データとして集約することで、経験値や人に依存した記録漏れや誤記入による情報の抜け落ちを防ぐことができています。本取り組みでは下図のようにまとめており、テスト結果を形式化し蓄積することで他のモデルとの比較も容易にでき、後から横断的な比較分析や、不具合傾向の調査など、継続的な品質改善サイクルの構築にも役立てています(図表10)。



	物体検出							OCR_	色解析						
Object	Left	Top	Width	Height	CenterX	CenterY	Confidence	OCR_text	R2	G2	B2	Hue	Satulation	Value	Color
Air Bag	43.56	262.44	89.78	77.09	88.44	300.99	0.9995223	ー	243	67	61	1	0.71	0.95	Red
Speed	898.72	381.81	267.81	226.18	1032.62	494.90	0.9979836	138	255	255	255		0	1	White

図表9　K-means法でClass分けの事例

5 今後の展望と価値創造

本稿でご紹介した自動テスト基盤は、SDV時代のニーズに即し、継続的な品質向上と効率化を実現する環境として進化してきました。現状はまだ“完成形”とは言えず、今後さらに価値を高めるために解決すべき課題に取り組んでいます。

5.1 現状の課題と進化の方向性

リポジトリ・CIツールの多様性への対応

現在、ソース管理・テスト資産の一元管理には主にGitLabを採用していますが、実際のユーザー環境では必ずしもGitLabが使われているとは限りません。また、CI・テスト実行エンジンも当社標準のツール構成が前提であり、これもまたユーザーごとに異なる(例:GitHub Enterprise, Bitbucket, Azure DevOps, 他CIツールなど)ケースが増加しています。

こういった、ユーザーごとの環境の違いに対応するため、次のような取り組みを行っています。リポジトリやCI/テストツールの抽象化・プラグイン化により、顧客環境固有の構成へ容易にカスタマイズ可能な仕組みを実装することにより、初期導入・拡張適応コストを大幅に低減し、多様な開発現場にシームレスにフィットできるようにしています。

パイプライン制御とユーザビリティの最適化

現状、CIパイプラインの各処理フロー(Stage選択、エラー時のフロー制御など)は主にスクリプトベースですが、システムとしての柔軟性・ユーザビリティには改善余地があります。

今後の展望:

直感的なWeb UIやテンプレート化により、開発者でなくとも容易にパイプライン編集・運用ができる柔軟な仕組みを実現し、属人化防止と現場導入の容易化を目指します。

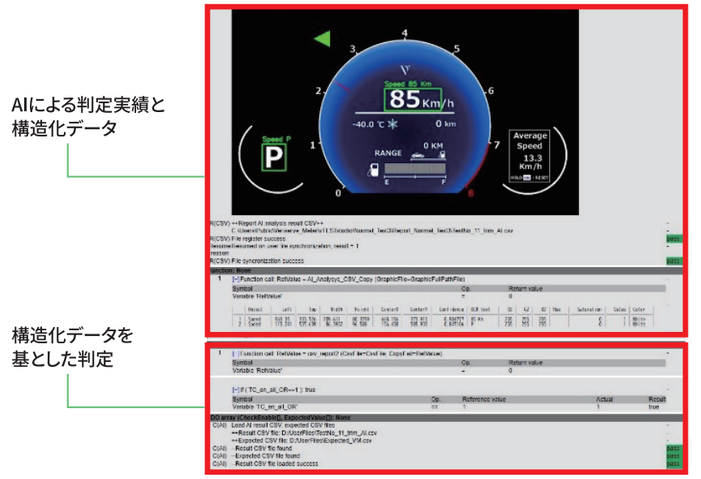
AI画像解析の実用性

HILSによるテスト等での画像解析による自動判定は、変化の激しい車載環境における自動品質保証として魅力的ですが、現時点では精度・速度・カバレッジ等で十分な実用レベルには達していないこともあります。

今後の展望:

AIアルゴリズムの精度向上や汎用性拡大に取り組むとともに、現場の実運用に即した評価指標やデータ整備も進め、機械学習モデルの継続的改善サイクルを強化していきます。

図表10　テスト結果の一例



5.2 中長期的な価値創造

前述の課題改善テーマに取り組むことで、

- **多様なユーザー環境に適応する「柔軟で拡張性の高い品質保証基盤」**
- **「現場ユーザーファースト」な運用性・ユーザーインターフェース／体験**
- **最先端のAI技術を活用できる自動化の現場実装**

といった事項の実現を目指します。また、今後も「ユーザーの要望に素早く応え、持続的な信頼性・効率性を担保できるソフトウェア開発・テスト基盤」としてSDV時代の技術革新をリードし、ユーザーによるプロダクト開発における真の価値向上に貢献します。

DocOpsで実現するQCDの向上

～ドキュメント管理の新しいアプローチ

近年、モビリティ分野において開発のQCD（品質・コスト・納期）向上は重要性を増しています。その中で、複雑化・グローバル化する開発プロセスを支える「ドキュメント管理」は、従来型の手法では限界が見え始めています。特にISO 26262などの機能安全や多様な法規対応が必須となる今、ヒューマンエラーの抑止、証跡性の確保、自動化による生産性向上が大きな課題となっています。こうした背景を踏まえ、本稿では、ドキュメント管理の新しいアプローチ「DocOps」に焦点を当て、その理念と実現の鍵を握るベリサーブの「ConTrack」について解説します。効率と品質を両立し、新たな価値を生みだすDocOpsの最新動向をお届けします。

横田 浩行 Hiroyuki Yokota

株式会社ベリサーブ　ConTrack事業部 開発課 課長

2009年、ベリサーブ入社。大手電機メーカーのプロジェクトなどでQAエンジニアを務めた後、新規事業の企画・推進に従事。その後はトレーサビリティ管理ツール「ConTrack」の企画・開発を主導するなど、システム開発全体のQCD向上のためのソリューション開発に注力。情報処理推進機構（IPA）のソフトウェア品質監査制度部会にも参画。

モビリティ領域におけるドキュメント管理の現状

最近のモビリティ分野におけるドキュメント管理の現状から本稿を始めます。もともとは、ISO26262機能安全規格がきっかけでトレーサビリティを入れる動きが十数年前に始まりました。まずはサプライヤーの状況を振り返ると、部品メーカーでは先行してトレーサビリティ管理を含む管理ツールの導入が進んでいきました。ただ、5～10年前くらいの段階では、ツールを使う分野は機能安全に関わったプロジェクトのみでした。また、QCD向上ではなく規格対応が主目的のツール導入が多かったと聞いています。

機能安全に関連しないプロジェクトでは、IATF16949（自動車産業において製品・サービスの不具合を予防し、ばらつき・ムダを低減させるためのマネジメントシステム規格）などQM（クオリティマネジメント）で十分対応できると考えられ、トレーサビリティ管理ツールの導入が進んだのは一部のプロジェクトに限られていたのが実態でした。

一方で、近年のトレンドとしてはECUの統合化やSDV化などによって、安全に関わるシステムとそうでないシステムを完全に切り離すことが徐々に難しくなりつつあります。また、ソフトウェアの規模や複雑性も増し、ツールを利用せずにQCDをコントロールすることも難しくなってきています。そのため、従来はトレーサビリティをツールで管理しなかったようなプロジェクトであっても「ConTrack」のようなツールの導入に関心を示すプロジェクト・組織が増えてきています。特に、最近のキーワードとしては、QCD向上を含む開発効率の向上に加え、ヒューマンエラーの抑止による安全性の担保が挙げられます。この2点は、自動車メーカーやサプライヤーが強く意識しており、こういった要求に応えられる製品を探しているという声を聞くことが多くなりました。

SOTIF*もキーワードとして挙げられます。SOTIFとは、性能限界を超えた状況やドライバーの誤操作など、システムに故障がない状態であっても、安全に自動車が動作することを目的とした国際規格です。従来はISO 26262への対応のみで十分であったプロジェクトでも、SOTIFのような新しい規格への対応が求められるようになっていきます。

このように対応すべき規格や法規が増えていく中、規格要求や法規要求、システム要求をどのような考え方で、どう落とし込んでいくのかという観点で、当たり前にドキュメント管理・トレーサビリティ管理を行わなければならない状況になりつつあると認識しています。

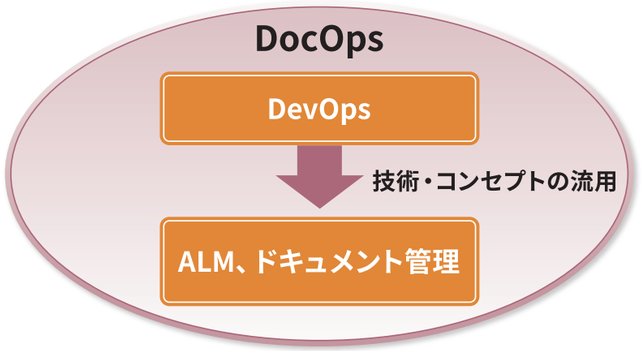
*SOTIF（ISO 21448）：ヒューマンエラーなどで生じるリスクを低減する自動車領域で扱う規格。

DocOpsとは？

昨今、モビリティの分野では、自動化、ヒューマンエラーの抑止、効率化が強く意識されています。例えば、特にヒューマンエラーについて、人間が実施する作業は何らかのミス・誤りが混入してしまうことを前提に開発プロセスを構築していくことが肝要であると考えられます。仕様変更を例とすると、OEMから新しい要求仕様書が提示された際に、サプライヤーでその変化点をチェックした上で対応を検討するのですが、人間が作業すると差分のチェック漏れが起きてしまい、結果的に要求変更の伝達漏れにつながってしまう可能性があります。このようにヒューマンエラーが品質低下の原因となることが考えられるため、機械化・自動化することによって、ヒューマンエラーを抑止したいといわれるようになっています。そこで、ベリサーブとして世の中に訴え始めたのが、DocOpsというアプローチです（図表1）。

DocOpsとは何なのかから説明します。DevOpsは広く普及していますが、これは、プロダクトやサービスの開発から運用までのライフサイクル全体を、さまざまな技術を使って自動化することでよりスピードアップすることと品質を高めることを両立する活

図表1　DocOpsのイメージ



動と言えます。

DevOpsでは、ソースコードが出来上がったらテストコードを実行し、テストを全部パスすれば、そのままビルドして本番環境へデプロイするといった一連の作業を自動化しています。これにより、コーディングからデプロイまでの流れについて、ヒューマンエラーの排除や品質向上について、これまでと比較にならないレベルでの効率性で実現可能となります。

ただし、システムが読める情報でないと自動化は難しいため、最近はAI技術を利用できるとはいえ、システムが判読できるような言語で実装されている工程（主にコーディング以降の工程）以外では自動化することが難しいのです。

より上流工程、要件定義や基本設計などの工程において、日本国内では、要件や仕様は、WordやExcelなどのレガシーなドキュメントに自然言語で書かれることが非常に多く、自動化しようと思ってもシステムに読み込ませることが難しい状況であることがほとんどです。

本来であれば、レビューまで自動化してしまえば人間だと気付けないようなところも、AI技術により解決できる可能性はありますが、レガシーなドキュメントがこれの実現を妨げてしまっていると考えています。

「ConTrack」は、WordやExcelなどのファイルから文書構造に従って項目に分解、文字情報を取り出せるので、レガシーなドキュメントからシステムがハンドリング可能なデータを抽出するための橋渡し役となることで、上流工程における自動化・効率化に寄与できるのではないかと考えています。このレガシーなドキュメントを対象にした自動化技術によって、システム開発の飛躍的なQCD向上を図っていこうという考え方がDocOpsなのです。

DocOpsによって、さまざまな業務・作業の自動化を実現することでQCD向上も達成できると考えています（図表2）。トレーサビリティ作業の自動化のみにとどまらず、デザインレビューを自動的に実施したり、要件や設計情報をデータとして集めたりすることによって、今までできなかった範囲まで自動化できるようになると考えています。

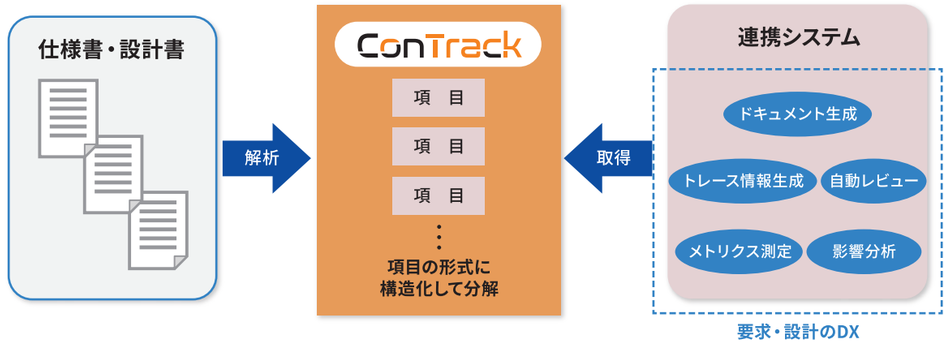
特に、繰り返し実施するような作業については、QCD向上の効果が大きくなっていくものと考えられます。

例えば、大量の数を持つインターフェース仕様書の改訂に伴うデザインレビューなどは、人間による作業で全数チェックをすると非常に時間がかかる上にミスをしてしまうことは避けられません。また、頻繁に改訂が入る場合は、都度レビューする必要があり、工数増加やチェック漏れリスクの増大などにつながります。これをシステムに置き換え完全自動化できれば、今までサンプルチェックしかできなかったのが、全数チェックできるようになり、飛躍的なQCD向上につながるものと確信しています。

新たなドキュメント管理に向けて〜「ConTrack」とは

「ConTrack」開発の経緯

図表2　DocOpsによるシステム連携イメージ



ここからは、DocOpsを実現するための手段として紹介した「ConTrack」について、なぜベリサーブが開発するに至ったのか、その経緯を紹介します。

2000年代後半、複数の自動車メーカーで大規模リコールが相次ぎ、自動車業界では品質管理や危機管理、サプライヤー管理を含めて抜本的な見直しを迫られるようになりました。システム開発においては、高い品質・信頼性の開発を進めると共に、こういった取り組みがきちんとなされていることを説明できること（品質説明力の向上）によって、品質向上や危機管理対応を進めていくことになります。これを受ける形で2011年に自動車分野の機能安全規格であるISO26262が発行され、この規格の中で、要求・設計などのトレーサビリティを取ることが求められるようになりました。

今まで、Excelなどで管理できてはいたものの、専用ツールで管理する文化が根付いていなかったので、当事者となった開発現場では、慌ててツールを導入したという背景があります。

当時グローバルでよく使われていたツールは、日本特有の擦り合わせ開発のプロセスになじまず、運用にはかなりの苦労があったといわれています。ソフトウェアを開発するサプライヤーからすると、規格対応のために半ば無理やりツールを導入せざるを得ない状況だったと考えられます。結局ツール導入の効果を発揮できるような使い方をできず、規格対応の証跡を残すため、エビデンスの最終保管庫のような形で使われるケースが多かったと聞いています。ツール導入に高い費用を支払い、運用面に懸命に取り組んだにもかかわらず、結局、開発のQCD向上という本質的な効果をあまり得られないまま、取りあえず規格対応はクリアした、という現実が垣間見えます。

自動車開発の現場における品質確保、向上のための活動に深く携わってきた当社としては、これまで培ってきたノウハウを生かして、こういった現実における課題を克服するための道具を開発しよう、という考えに至ったのは自然な流れでもありました。

ALMとしての「ConTrack」の概要

「ConTrack」は、トレーサビリティ管理の機能だけではなく、アプリケーションライフサイクルマネジメントツール（以下、ALMツール）として開発しています。

これはシステム開発におけるドキュメント管理について、構成管理、変更管理も含めて統合的に実現することによって、QCD向上に寄与したいと考えているためです。

その中でも「ConTrack」のメインであり、競争力を持つものはトレーサビリティ管理機能であると考えています。

強力な文書解析エンジンを搭載することで、WordやExcel、PDFなどさまざまなフォーマットで表現されたドキュメントを「章」「節」「項目」という形で構造分解し、その粒度（単位）で

要求仕様が、基本設計、詳細設計といった各工程のどこで実現できているのか／できていないのかを可視化します（図表3）。

さまざまな開発環境や構成管理ツール、課題管理ツールなどと連携することによって、ユーザーが使っているツールチェーンに「ConTrack」を組み入れ、統合的な管理環境を実現し、品質と生産性の向上を目指しています。

「ConTrack」による DocOps事例

具体的にDocOps実現に向けた「ConTrack」活用の事例を紹介します（図表4）。

自動車分野では、ISO/SAE 21434というサイバーセキュリティのマネジメントシステム規格があります。各自動車メーカーやサプライヤーでは、この規格を自社の内規に落とし運用されています。

全社レベルで定められた内規は、部門ごとの開発プロセスにブレークダウンされ、それをベースに各チームが設計ドキュメントを起こしていく活動が行われています。

見るべき法規はさまざまな国や地域ごとに存在し、似たような条文であっても、実施すべき活動が違うこともあります。このため、この内規ではなぜこういう活動になっているのかということまで、説明性を担保することが求められています。

例えば、図表4にあるように、欧州、米国、日本、中国の各国における法規の条項が同じであり、対応すべき活動が同じであることから、全社内規は項目Bとして一つの規定になっていたとします。この時、中国の条項Bが法改訂により求められる活動が変わってしまった場合、全社内規の項目Bは適用できなくなります。こうしたケースでは、対応すべき活動の見極めが簡単にはいきませんが、「ConTrack」を活用することで、抜け漏れなく管理できる仕組みを用意しています。

法規文書は基本的にPDFファイルで公開されていることが多く、内規はWordなど別のファイル形式で作られていることも珍しくありません。ファイル形式が異なる場合でも、「ConTrack」は社内規定がどの法規に基づいて作られているのかを、線で結んで可視化できます。さらに、グローバル共通とか中国向けのみといった形で、「ConTrack」の中で属性情報を持つことができます。グローバル共通の項目だから、もし変更があった場合には一国を個別で切り出さないといけないとか、中国向けのみの対応なので、欧州の要求が変わったとしてもここは見なくていい、などといった判断が可能になります。

さらに「ConTrack」は履歴情報を保持するため、例えば2024年時点の情報、2025年時点の情報、という形（スナップショット）で逐次残します。後になって何かしらサイバーセキュリティに関連する問題が見つかった場合には、2024年に開発した時点の情報はどうなっているのかをすぐに出せる形になっています。

自動車分野での DocOps実現に向けた取り組み

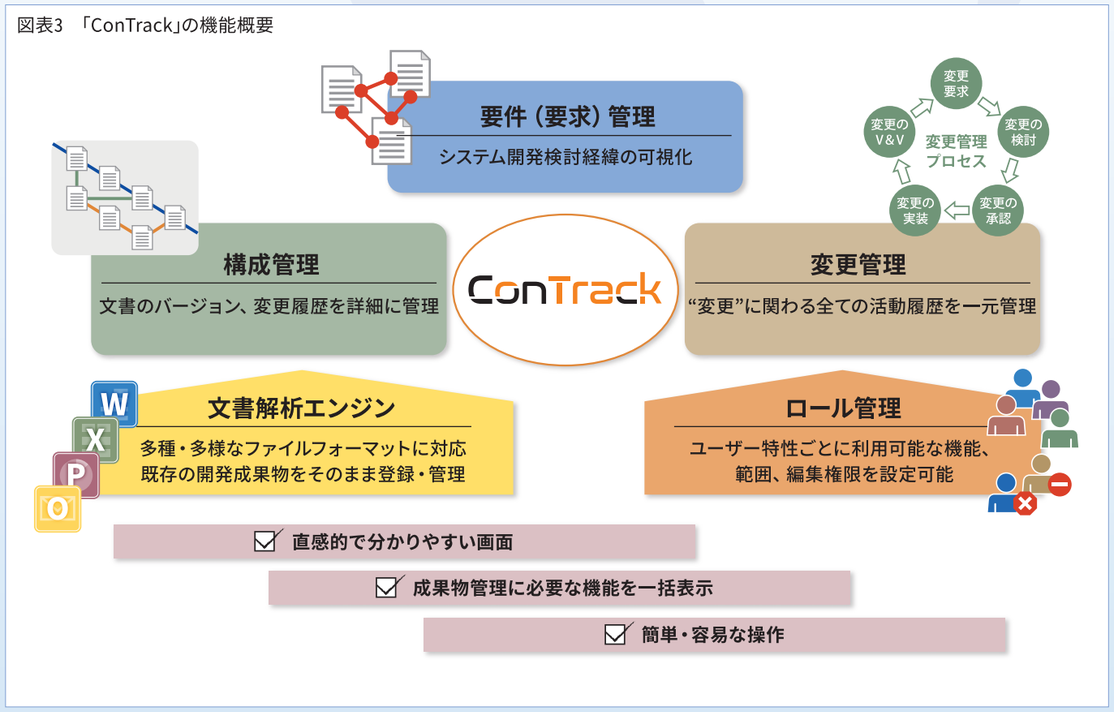
自動車の制御ソフトウェアの開発では、MIL（Model In the Loop）と呼ばれるシミュレーションベースの仮想環境上で仕様に対するテストを行わないと立ち行かなくなっています。

代表的なシミュレーションシステムであるMATLAB/Simulinkは、モデルベースド開発（MBD）、MBSE（Model-Based Systems Engineering）といった開発手法が用いられている現場で、広く活用されています。MATLAB/Simulinkは、ブロック線図でどういった情報をインプットにどのような計算を行うのか、計算ロジックとパラメータを設定し、どういった出力をしていくのか、ということをブロック（箱）と線でつなげ、その計算の流れをモデリングします。（図表5）。

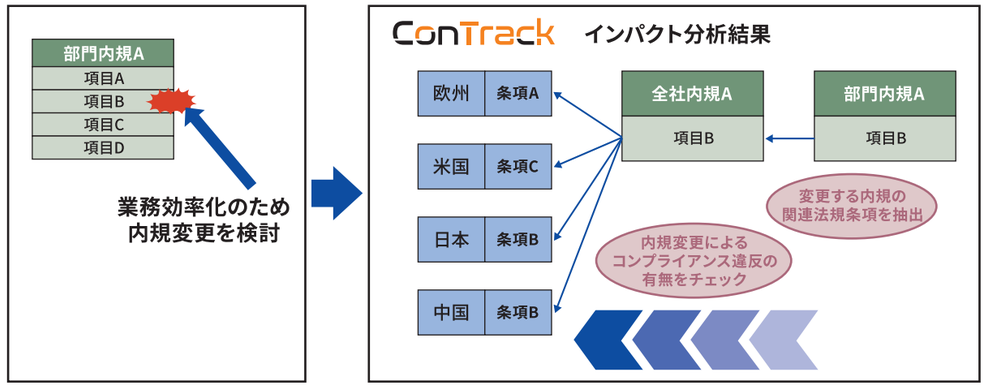
ブロック線図によるモデリングの課題として、設計担当者本人がどこをどう変えたのかを覚えていないケースが多いことが挙げられます。設計者自身が覚えていないためにその影響範囲も分からなくなってしまいます。その結果、変更による影響があらゆる箇所に出るリスクがあるため、下手をすれば、テストやデザインレビューを全部やり直すことになりかねません。また、変化点が分からないため、デザインレビューで見る人もどこが変わったのかが分からなくなってしまいます。

こういった課題に対して、「ConTrack」の差分の比較機能では、まず配置しているブロックが一緒であるかどうかをチェックします。さらに、制御の流れでブロックの順番や

図表3　「ConTrack」の機能概要



図表4　法規対応時のインパクト分析イメージ



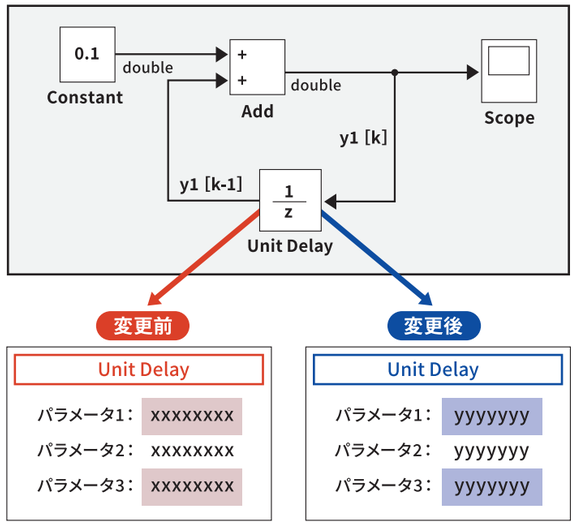
計算の順番が変わってしまうと、計算結果も変わってくるため、どういったブロックに情報を出して、どのように流れて計算していくのかという線の変更についても、差分が取れるようにしています。

もう一つはパラメータです。求めたい計算結果が得られるように、パラメータチューニングを行った前後の変化点がどこにあるのか示せるような形で出力していきます。どのブロックを、どのパラメータを、誰がどういう風に変更したのかを全て記録として残すことで実現しています。一方、MATLAB/Simulinkによる分析で得た、全ての変化点情報の出力を用いることでテストアナリティクス範囲を指定し、シミュレーションをHILS（Hardware In the Loop Simulation）で回すことはできるでしょう。しかしながら、HILSで全部のパターンを回そうとすると非常に時間がかかり一晩で終わらないほどのボリュームとなることさえ珍しくはありません。

DocOps普及・進展に向けた「ConTrack」のこれから

ここまで記した通り「ConTrack」は、トレーサビリティを取ることによって、影響範囲の分析や、変更が入った時にどこが変わったのか、その影響をどこが受けるのかを示せます。

図表5　制御モデルによる仕様の表現例



それにより、品質の劣化を抑えていく点が肝となります。さらに、自動車メーカーやサプライヤーでは、「自動化」「ヒューマンエラーの抑止」がキーワードになっています。

私たちの使命として、よりDocOpsを実施しやすい環境を整えていくことが重要だと考えています。

また最近では、品質不正問題のように、自らの開発を第三者に対して論理的な証拠を示して説明できるようなプラットフォームが求められています。私たちは「ConTrack」で、こういったニーズをキャッチアップし、お客様の開発に貢献したいと考えています。

さらに、さまざまな開発活動を自動化することで、なるべく人手を介さずに済むようにツールを進化させていきます。人が判断するところはどうしても残ると思いますが、人間が携わらなくてもよい単純作業はどんどん自動化することで、ヒューマンエラーの抑止、開発効率の向上につなげられるからです。

そして、お客様自身の手で、こういった活動をやりやすくするためのAPIをどんどん拡充する予定です。

最後に、お客様のドキュメント管理を最大効率化し、それがQCD向上に確実に役立てるよう、また、不正防止にも活用できる「真のDocOps環境の実現」を、私たちは今後も目指していきます。

ベリサーブの拠点

当社は、お客様の多様なニーズにお応えするため、国内外に拠点を展開しています。
地域に根ざしたサービスの提供と、各エリアの特性を生かした事業運営を推進しています。