

SmartDeviceLink as an Open Innovation Platform for Connected Car Features and Mobility Applications

Jeffrey Yeung, Omar Makke, Perry MacNeille, and Oleg Gusikhin
Ford Motor Company

ABSTRACT

SmartDeviceLink (SDL) is open-source software that connects the vehicle's infotainment system to mobile applications. SDL includes an open-source software development kit (SDK) that enables a smart-device to connect to the vehicle's human-machine interface (HMI), read vehicle data, and control vehicle sub-systems such as the audio and climate systems. It is extensible, so other convenience subsystems or brought-in aftermarket modules can be added. Consequently, it provides a platform for cyber-physical systems that can integrate wearables, consumer sensors and cloud data into an intelligent vehicle control system. As an Open Innovation Platform, new features can be rapidly developed and deployed to the market, bypassing the longer vehicle development cycles. This facilitates a channel for rapid prototyping and innovation that is not constrained by the traditional process of automotive parts development, but is rather on the timeline of software development. This allows parties other than the OEM, including third party developers, universities, and startups, to quickly integrate their applications into the car. By digitizing the controls into a programmable application program interface (API), we can incorporate machine learning and ambient intelligence to learn user preferences and needs over time and intelligently pre-set them automatically. This can disrupt the traditional model of the human machine interaction. This is particularly important as transportation moves towards autonomous vehicles. In this paper, we discuss some specific examples using SDL to enhance the transportation experience, including intelligent air quality and climate control, enhanced radio interface, mobility service applications, and innovative intelligent vehicle features.

CITATION: Yeung, J., Makke, O., MacNeille, P., and Gusikhin, O., "SmartDeviceLink as an Open Innovation Platform for Connected Car Features and Mobility Applications," *SAE Int. J. Passeng. Cars – Electron. Electr. Syst.* 10(1):2017, doi:10.4271/2017-01-1649.

INTRODUCTION

The automotive industry is experiencing an era of expanded technology opportunity, particularly in the areas of vehicle electronics and connectivity. The industry has developed methods of innovation and product development suited to the production of very robust commodity products in a highly optimized, global manufacturing system. In recent years, vehicle infotainment systems have been combined with internet services via the wireless cellular data network to create new opportunities. This combination disrupts the traditional development process significantly. New methods of innovation and development are needed for this era of rapid development.

Open innovation is a topic within the field of innovation management. The term has been promoted by Henry Chesbrough from the Center for Open Innovation at the University of California. He has written books and papers on the subject [1, 2], although the idea has been discussed as far back as 1960s [3]. The fundamental premise is that properly managed inflows and outflows of knowledge can improve innovation within a firm while creating new markets of the technology outside the firm.

In a new business model, a good understanding of the value proposition, target market, value chain, revenue mechanism(s), value network, and competitive strategy is needed to extract value from the technology. Developing this understanding often involves a process of trial and error and can take considerable time. A firm with a new technology needs to develop the business model very quickly. Most of the time, firms and industries operate with a closed innovation strategy where incremental technological improvements drive product differentiation, meet new government regulations, market changes, etc. These firms frequently compete by being lean and robust, but lack the resources to consistently create value from rapidly changing technology.

The value network, or ecosystem, is developed through an evolutionary process that begins pretty chaotically but ends in business model(s) that add value. Early in the process, members of the ecosystem have varied characteristics, motivations, and contributions to make. As the ecosystem matures through trial and error, the necessary operational components become clearer, and some members may drop out while others may commit more resources. There can be mergers, spinoffs and selloffs until a stable business is established, commoditization sets in and a closed innovation model is once again preferred.

In software development, open source is a key enabler for an open innovation platform. The ecosystem consists of member firms working together on a common software platform that allows innovators to develop technology and business models utilizing road vehicle entertainment systems for live testing for their concepts. In the following sections, the authors introduce their open innovation platform and several use cases, hoping to encourage technology entrepreneurs to join their ecosystem.

Open Innovation Platform

SDL is the focus of an open innovation ecosystem intended to discover new businesses arising from recent technology advancements. These new businesses may be based on unexpected combinations of hardware, software, and ecosystem members that are admittedly difficult to predict at this point. Therefore, the adaptive nature of open innovation approach is necessary.

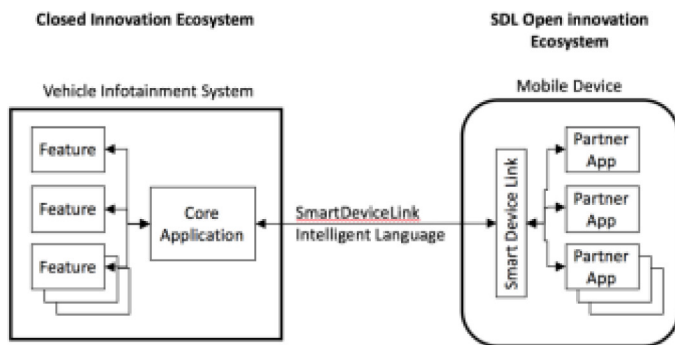


Figure 1. Innovation Ecosystems

The SDL Open Innovation ecosystem is intended to involve members from large and small firms that may develop software, hardware or both. Currently the SDK is targeted to mobile devices that support iOS and Android operating systems but is open to new hardware devices if they support Bluetooth, WiFi and/or USB.

The ecosystem also accommodates members from other automotive OEMs, suppliers and after-market companies that wish to implement the core application (Figure 1) on their infotainment platforms. The openness of the platform makes possible many different alliances of development partners, OEMs, suppliers and aftermarket companies. The nature of the software interfaces is such that members can have very different business priorities, development cycles, revenue models and technologies.

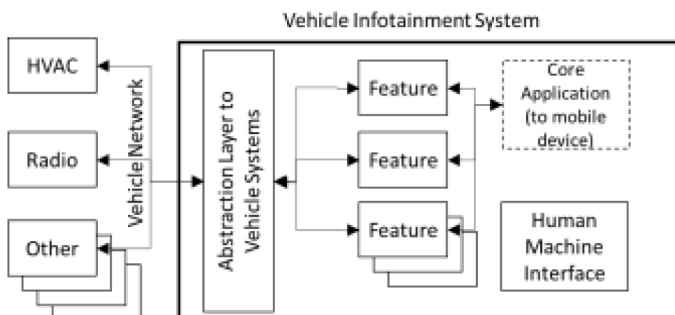


Figure 2. Vehicle Systems Flowchart

SDL allows mobile apps to read vehicle data and control vehicle sub-systems such as the audio and climate systems. It is extensible, other convenience sub-systems or even brought-in aftermarket modules can be added. Figure 2 shows how SDL Core interfaces with the rest of the vehicular sub-systems and features. These features include navigation systems, climate control, GPS, geo-fencing, speech recognition, radio, audio equalizer, etc.

SDL facilitates a channel for rapid prototyping and innovation that is not constrained by the traditional process of automobile parts development but is rather on the timeline of software development. This allows parties other than the OEM, including third party developers, universities, and startups, to quickly integrate their applications into the car using mobile device apps. By digitizing the controls into an open source programmable application program interface (API), it is possible to incorporate machine learning and ambient intelligence to learn user preferences over time and intelligently pre-set them automatically. This can disrupt the traditional model of the human machine interaction.

This paper describes a system to connect the rider to the vehicle. It first begins with an introduction into SmartDeviceLink and its capabilities to show how it can be used as an Open Innovation Platform for the vehicle. This is followed by examples of innovative features for the Connected Car using SDL.

The first use case is an intelligent climate control system, integrating brought-in sensors and wearable devices for automated climate and air quality control. This utilizes air-quality sensors to detect both the interior and exterior air quality and adjust the fan speed and recirculation vent appropriately to maintain clean air within the vehicle cabin. This has particular use in high-density cities high air pollution. Furthermore, we discuss the usage of wearable devices with cloud data to create an intelligent climate control system based on real-time feedback from the vehicle occupants to improve the user experience.

The second example is for a hybrid radio system, which combines internet content providers with broadcast radio services. We will demonstrate an example of hybrid radio implementation with a mobile app called NextRadio. NextRadio receives the FM radio in the car and combines it with metadata received on the smartphone to enhance the radio experience in the vehicle. Album art, song information, and music genre are added to the display. Additionally, it enables selection of radio stations within range of the vehicle by genre or current song, none of which is possible with just FM radio.

The third example is a mobility application, specifically ride-sharing. In this case, SDL acts as the medium to enable not the driver but the passenger to control the vehicle sub-systems from their phone, either in a taxi or ride-share or in a large passenger vehicle. This allows for dynamic control of their preferences from their phone based on various meta-data, which we can learn about them over time. This also has great implications for autonomous vehicles, which may lack a familiar console interface to control these options.

We discuss the implications of this system as a whole and how it fits into the current trends in Mobility. Finally, we also touch on some other examples including audio equalizer and auto-high beam.

SMARTDEVICELINK OVERVIEW

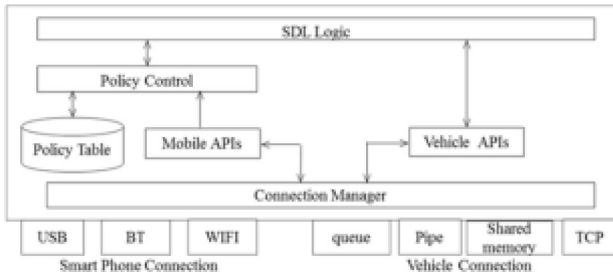


Figure 3. SDL Architecture

SmartDeviceLink (SDL) is an open source project pioneered by Ford Motor Company that connects in-vehicle infotainment systems to smartphone applications allowing automakers the opportunity to provide customers with highly integrated connected experiences, and app developers with new and exciting ways of connecting with their customers. It is managed by the open source community, SmartDeviceLink Consortium Inc. (SDLC), which includes companies such as Ford Motor Company and Toyota Motor Company. It is open to OEMs, suppliers and app developers who are integrating with SDL or have plans to integrate with SDL in the future [4].

SDL comprises of head unit software, known as SDL core, and mobile SDKs for Android, iOS, and cloud configuration. SmartDeviceLink can be treated as a specification which can be implemented by arbitrary developers. Nevertheless, open source implementation is provided. SDL core's implementation is provided in C++. It supports several transport protocols: Bluetooth, WiFi and USB. SmartDeviceLink supports both media and non-media apps. Media apps are dedicated to audio streaming and provide an alternative user interface (UI) to the native media UI, which usually include FM/AM/XM, and CD. Non-media apps extend the built-in applications, and normally read vehicle data and provide added functionality to the driver. The head unit defines four states called HMI_LEVEL for each connected mobile app: FOREGROUND, LIMITED, BACKGROUND, and NONE.

When the app is selected from the head unit, it opens a predefined UI templates and is put in FOREGROUND state, which gives the app all of its allowed permissions. Once opened, the driver may switch to a different screen on the head unit, such as the navigation screen. If the driver navigates to a different screen on the head unit, the app enters BACKGROUND state where most of its permissions are lost. A LIMITED mode is defined for media apps. This mode is entered if the HMI is displaying the predefined UI template, but does not have propriety. This can happen if an app is streaming music and the driver receives a phone call for example.

Mobile applications can communicate with SDL through the SDL software development kit (SDK), which is available for Android and iOS platforms [5]. The SDK makes the app discoverable by the vehicle's head unit. It exposes a set of remote procedure calls (RPCs) through a defined set of application programming interface (API). In brief, the app instantiates an instance of the SDL proxy class, provided by the SDK, which handles the communication between the app and the vehicle. The RPCs are methods of the proxy class. Moreover, the proxy class receives the vehicle's notifications and makes them available for the mobile app.

Different OEM's can implement SDL and provide different capabilities. Once the proxy starts within the app and connects with the vehicle, basic information about vehicle capabilities such as vehicle's language, audio capabilities, etc. are immediately made available to the app. The app can further query for specific capabilities through the proxy APIs. Although the RPC implementation is not directly exposed to the app developer, it is worth noting that the RPC protocols are implemented as JSON strings.

A remote control extension for SDL is created and is available in the public repository. The extension consists of additions to SDL core inside the vehicle and to the mobile SDK for Android and iOS. Three major RPC's responsible for remote control, and their corresponding APIs are shown in Table 1. These APIs provide enough abstraction for mobile apps to control vehicle modules with different capabilities inside vehicles by different OEMs.

Table 1. SDL Remote Control APIs

API	Parameters	Description
<code>getInteriorVehicleCapabilities()</code>	Zone	Returns supported modules the vehicle is equipped with (Radio, Climate Unit...).
<code>getInteriorVehicleData()</code>	Zone, Module	Reads module data. Modules are obtained from <code>getInteriorVehicleCapabilities</code> .
<code>setInteriorVehicleData()</code>	Zone, Module, Data	Control API. Sets the data of the Module for the specified zone.

Security Considerations

As with any open source platform and wireless applications, a major consideration for any system is security. In the case of SmartDeviceLink, the base security mechanism is pre-defined, but the OEM can extend it to add additional layers of security. Apps during testing phases can also have more privileges than during release.

The core of SmartDeviceLink security is the policy table, which is hosted in the cloud by the OEM. Each SmartDeviceLink App requires an App ID to be generated by the OEM, which identifies the app in the policy table. Each App ID is associated with an explicit set of RPCs which the app can execute in each of the four HMI_LEVEL states of the app. If the App ID is not present in the table or if the app attempts to execute an RPC not listed in its HMI state, access is denied. In this way, the OEM has central control over all app