

Keys to Building a Successful In-Vehicle Infotainment & Automotive System



Steven Yee, Director of Applications Engineering, BSQUARE Corporation

There was a time – not all that long ago – when a state-of-the-art automotive entertainment system consisted only of an AM radio. It wasn't a stereo system. There was only one speaker. If the radio was truly advanced, it had a series of mechanical buttons that you could use to bookmark a few of your favorite stations. Another mechanical level controlled the climate: it turned on a heater, a fan, or was off entirely.

Introduction

We have come a long way.

Today's systems not only have more speakers, they offer many more features. We have progressed far beyond in-dash cassette decks and CD players to offer in-dash navigation systems and easy-to-use climate controls that let you punch in a specific temperature.

We have even farther to go.

Who knows what the next must-have feature will be? What is almost certain is that it will arrive faster than the advances that preceded it. The pace of innovation is accelerating dramatically. Consider in-car navigation systems. Just a few years ago, such systems were available only as third-party accessories. They existed only as standalone units, powered through an accessory port and attached to the vehicle by way of a suction cup.

In a very short period of time, navigation systems became part of the vehicle. They are no longer an afterthought. They are completely incorporated into the design of the vehicle. And they are no longer a feature found only on top-of-the-line luxury cars. They are becoming standard equipment on more affordable models.

Just as the feature set changes, so should the development approach. Automakers and system builders no longer have the time to develop unique products for each vehicle model or even brand. This outdated approach is not only excessively expensive, it's also time consuming. Development resources are better spent keeping up with advances in technology and demand from the marketplace.

And the makers of one component have to be aware of how their product fits in the overall system. Years ago, virtually any radio could be plugged into any car. Today, all the components need to talk to each other.

For example, cell phone hands-free technology is now built in many vehicles. Today's radios need to be aware of when the user is on the phone and route the audio of the phone call over the speaker system instead of the regular entertainment feed. As the feature set grows, the systems also need to be easier for the driver to control. More systems now respond to voice commands, include interactive touch-screen displays, or both.

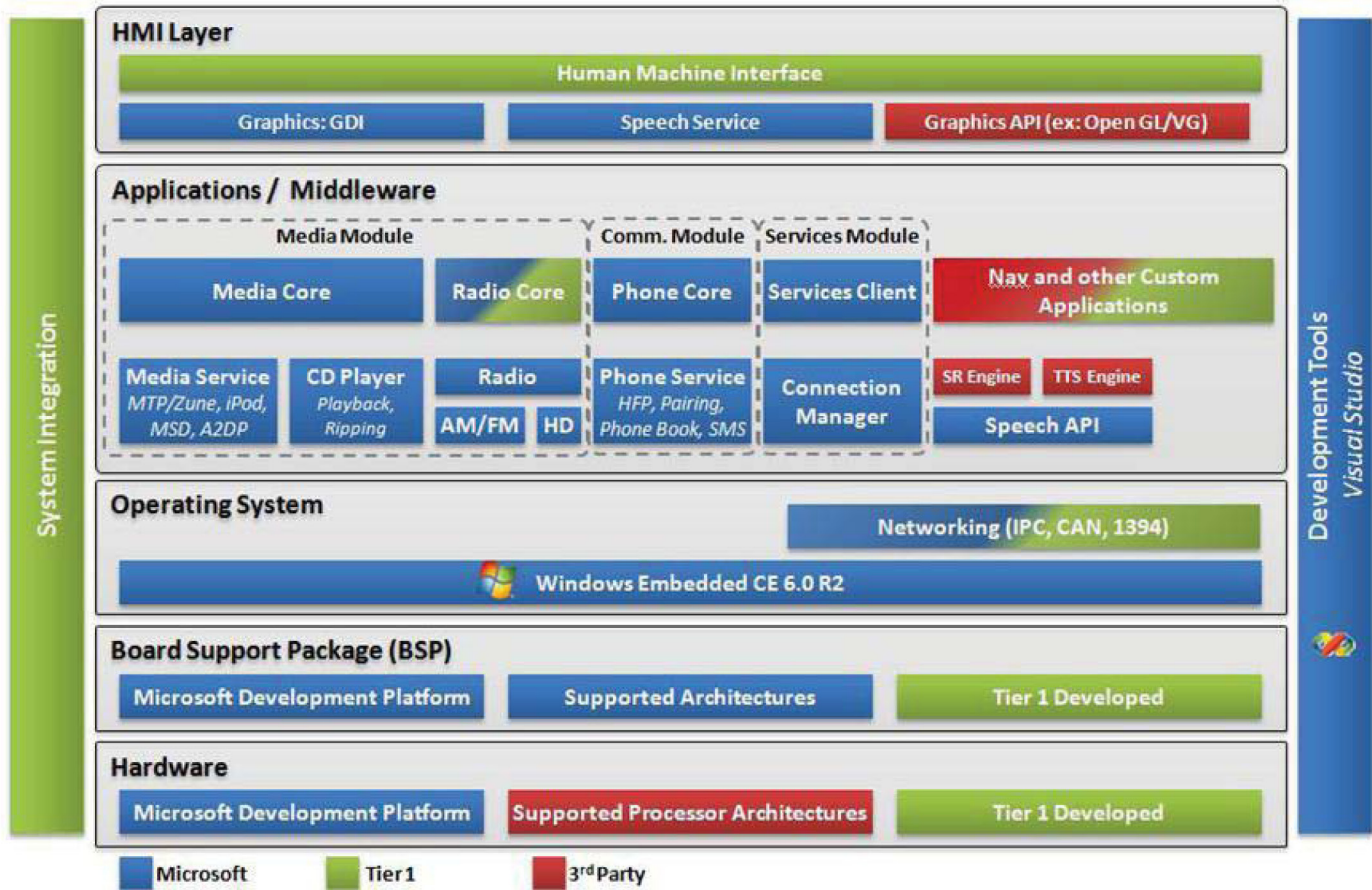
Today's systems are no longer contained entirely within the car. Many now communicate with the outside world to download up-to-the-minute traffic and other data, or send information, such as the health of the vehicle's components, to a central server.

For these reasons – and others – it's often best to develop your unique software on top of a proven, stable platform that provides user interface, voice command, communications, and other fundamental services. This paper will share best practices for using Microsoft Auto 4.0 as that base.

Build Upon a Rich Platform

All these demands may sound difficult to meet, but they are not if you build your device on top of a rich operating system, particularly one designed for automobiles. By using an abstracted approach, you should be able to easily make changes to the interface and add third-party capabilities like voice command without affecting your fundamental code.

There are a number of rich OS on the market you can choose from, including Windows CE, Linux, Android, Wind River, and QNX.



Microsoft Auto 4.0 Architecture

Beside the base rich OS, we also start seeing company adding more Auto specific feature on top of the OS to target this vertical market. Microsoft Auto is one of these products and we will take a closer look at it this article.

Microsoft Auto 4.0 is a version of Windows Embedded CE 6.0 R2 specially customized for automobiles. It includes a media core that not only supports radios and CD players, but also any media player that drivers are likely to have. Its communications module contains support for Bluetooth hands-free calling. And its Human Machine Interface (HMI) layer makes it easy to plug-in voice command software, among other things.

Let's examine that middleware in more detail.

Media Core Architecture

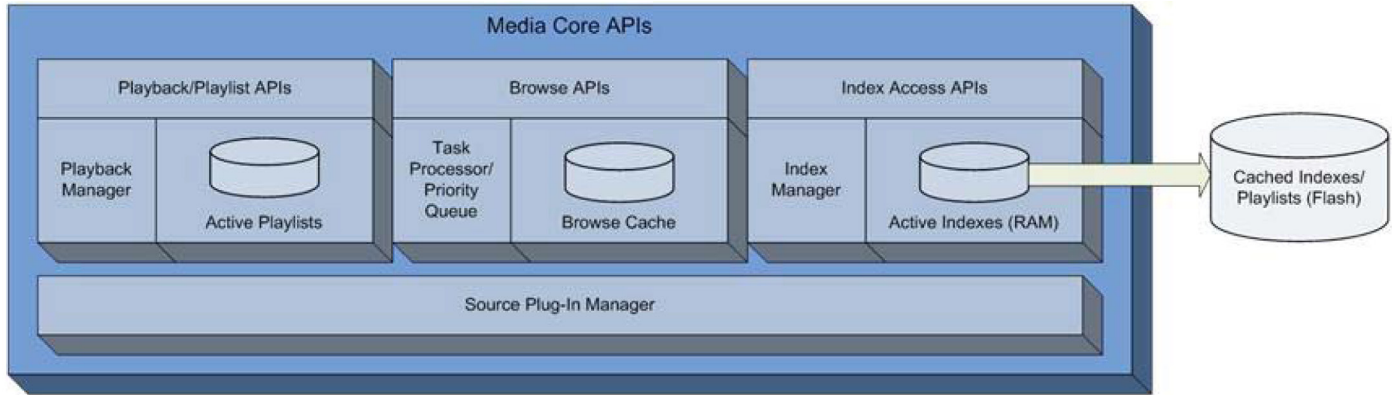
The ability to play media files from devices like the Apple iPod and other MP3 players is becoming a standard feature on many vehicles. The media core in Microsoft Auto 4.0 has the greatest potential to save automotive developers time. There are countless media players on the market. The media core gives you one set of APIs to play and browse their contents, regardless of whether the device in use is a media player with a proprietary format or a flash drive loaded with MP3s.

This pre-built media core can also save you money. The interface to access the Apple iPod is a closely guarded secret. To provide iPod support, you would have to license a special chip from Apple and incorporate that into your design. The media core in Microsoft Auto 4.0 includes support for the iPod.

Even in devices without a proprietary interface, processing media files can be challenging. You need to be able to handle anything users enter as metadata. Different devices and content creators may also have different tagging standards. If you try to build the media support yourself, you need to support multiple languages and be able to handle error conditions.

This approach also saves you testing effort. Microsoft has tested its media core with wide variety of consumer devices. That testing can be tough to do yourself given the huge number of variables.

While Microsoft Auto 4.0 can reduce development and testing time, there are some special issues you should consider when developing your system.



Media Core Architecture

Depending on your needs, you may need to code DirectShow and driver support. You will also need to write some audio bus code to tell the system where to route the audio.

You also need to consider how you want to index media files. If the user plugs in a flash drive with 5,000 MP3s, the system could grind as it goes through and indexes every one. You need to decide if you want the user to wait for that process to be completed, if you want to pause the indexing process if the user wants to do something else, or if you want to increase processing capacity, perhaps by using a special graphics processor to free your CPU for tasks like this.

Also, be aware that Microsoft Auto 4.0 will not index image or video files. The media core will show or play them, but not by title only.

Phone Core Architecture

As more government jurisdictions crack down on driving distractions, the ability to control the infotainment system and make phone calls hands-free, using voice commands, is also an essential feature.

While drivers in the past were content to add-on hands-free calling devices, they have grown to expect a much more integrated approach. In a growing number of vehicles, the standard equipment includes not only a built-in speaker phone, but one that also supports voice commands.

The phone core in Microsoft Auto 4.0 includes built-in support for Bluetooth pairing, hands-free calling over Bluetooth connections, as well as phonebook and SMS support.

Bluetooth Pairing Architecture

The Bluetooth Pairing Core consists of two parts — the service component and the in-proc DLL which provides the API that abstracts the IOCTL calls to the service. The service

is implemented as a standard service with standard service API functions. The API is a simple wrapper around the task of packing function arguments into IOCTL calls to the service.

The core functionality provided by the service is the paired device list management capabilities. The service maintains three paired device lists, for phone, media, and other devices. The service also handles the Bluetooth pairing procedure, although not the Bluetooth connecting procedure. It handles the management of the paired devices using an access ordered list with a maximum length of 12 paired devices.

Calling and Phonebook Functions

The pre-existing calling functions are quite extensive. The phone core lets the application answer incoming calls, dial a new call, switch between calls, hang up calls, and redial the last number called. It also displays call waiting and call history information, and offers a privacy mode which sends the call to the handset.

It also automatically synchronizes contacts. From phones that support this process, the phone core can download the phonebook information over the Bluetooth connection and store it so that it is displayed almost instantly the next time the user pairs device. New or changed contact information is automatically updated in the background. Just as with the media core, a lot of testing is done for you already. Microsoft has tested hundreds of phones with Microsoft Auto 4.0. Support is also growing constantly. Microsoft regularly ships device compatibility updates, allowing you to update your device to support the latest products your customers may be using.

User Interface

The Apple iPod has completely revolutionized what people expect from a user interface. How your user interface works and looks plays a major role.

Most people are highly visual and they often evaluate the quality of an item by what they see. Imagine a consumer is trying to decide between two GPS units. Aside from comparing the specifications on any marketing materials, the shopper has no way of determining which unit is backed by superior technology. The only easily accessible benchmark is the appearance of each unit.

More attractive units are also considered much easier to use, whether or not that's actually the case. Scientists call this the Aesthetic-Usability Effect. Repeated studies have found consumers truly believe the attractive devices are better. They are more likely to use these devices and they believe they are easier to learn.

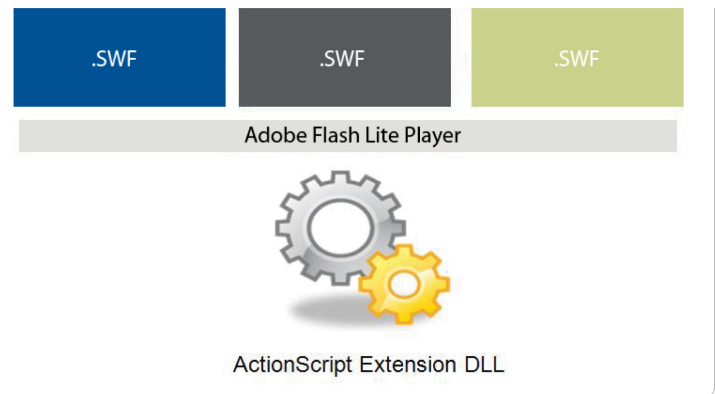
Making a great first impression is also critical. Once a user has decided that a device is easy to use — or not — it's difficult, if not impossible, to change that perception. People are generally stubborn in their views.

Keeping the interface attractive and easy to use is somewhat challenging given that drivers expect more information from their vehicle. They demand more information about fuel efficiency and the status of vehicle systems, more options for climate control, and more detail to help them get where they are going and to find places to eat, see a movie, or fuel up along the way. They want tremendous detail, yet expect the interface is free from clutter.

Since today's infotainment systems are capable of so much and since attractiveness and ease-of-use are paramount goals, a touch screen — and a large one at that — typically provides the best means for interacting with the user. Given how high Apple and other manufacturers have set the standard for appearance, it's critical that you invest significant effort in your user interface in order to compete.

After making such an investment in the user interface, you will want to maximize its mileage — that is you will want to be able to reuse it as much as possible.

This does not mean, however, that you will want to use an identical interface even in different models within the same make of cars. As we've already discussed, appearance is incredibly important and auto manufacturers will insist that their higher-end vehicles appear more luxurious than cheaper models, even if the core technology in each device is the same.



Adobe flash 'skinning' engine.

It's best to make a user interface that you can easily customize or re-skin. You should be able to quickly hide features that are not available on budget models. You will also want to be able to rebrand your interface with minimal effort, both for different vehicle manufacturers and for different models within the same line. A device going in a luxury car should truly look more luxurious than its budget counterpart. If you want to sell your device to several different automakers, you should ensure that each manufacturer gets a unique look. Microsoft Auto 4.0 includes two user interface components: the graphical user interface and the voice command interface. Both are extremely important.

You will need third-party to handle voice commands. While Microsoft Auto 4.0 provides the basic framework for that functionality, you will need a third-party engine.

The graphical user interface is another area where you will almost certainly want to go beyond the graphics capabilities provided by Microsoft. Microsoft does not provide OpenGL graphics acceleration, but this is something you should seriously consider adding in order to provide a graphical interface that meets or exceeds today's high user expectations.

Re-skinning is yet another area where you may want to add a third-party technology, like Adobe Flash. The easier it is to customize your interface, the more potential markets you have for your technology. Web technologies and Adobe Flash give you the tools to make stunning interfaces, yet are relatively easy to modify.

Connectivity

Connectivity is an area ripe for tremendous growth. We have access to more information than ever. For instance, there are web sites that track the lowest gas prices in a particular area. Numerous other web sites provide detailed traffic information, including traffic accidents and expected travel time between two points. With an internet connection, you can pull in all of this available real-time data, providing even more value to the driver.

Social networking is also increasingly popular. Drivers of some hybrid or alternative energy vehicles may appreciate the ability to share their fuel economy with other members of their network. Such a feature would be easy to add with connectivity in the vehicle. It would not only be a selling point to people obsessed with social networking, it could also provide a means of viral marketing for the automaker.

There are a few options you can get connectivity to a vehicle system:

- Cellular modem
- Data over voice
- Satellite & FM radio
- Wi-Fi

Including a cellular modem on your system will provide the closest thing to an “always on” connection. However, if the cost of including such a module doesn’t fit your budget, you can consider using the driver’s cell phone as a modem and provide connectivity via Bluetooth connection. Of course, this approach relies on the end users have a cell phone with data plan.

An alternative is to use the voice channel on the cell phone for connectivity. Data speeds are slow, however, and this process still requires your end user to have a Bluetooth-connected cell phone. Many systems on the market today are using satellites to gather real-time travel, weather, and other information. Satellite is a great way to gather information from content providers, but provides very little ability to upload information.

	Windows Automotive 5.0		Microsoft Auto 4.0	
OS base	CE 5.0		CE 6.0	
UI	Powerful, automotive-ready; AI-UTK for advanced UI design	X	Build/bring your own	•
Navigation	Build/bring your own based on GDI-Sub API	/	Build/bring your own	•
Hands-free phone	Build/bring your own	•	Standard hfp functions, auto-ready	X
Video	CE 5.0 Standard support	•	Build/bring your own	•
Audio/Media	Build/bring your own	•	Automotive-ready for portable music players (PMPs)	X
Hardware	Design your own	/	Reference hardware for device gateway-style devices, or design your own	X
Middleware Services	CE 5.0 Standard	•	Updated bluetooth stack, usb, speech service	X
Speech engines	Build/bring your own	•	Multiple included for development; build/bring your own for shipping	X
Development tools	AST-specially designed for automotive applications	X	Standard CE 6.0 tools	/
Updatable	CE 5.0 Standard	•	Image update, automotive-ready CE installer	X
(X's represent areas where the least development effort is needed, /'s represent some development and •'s represent no development.)				

Differences between Windows Automotive 5.0 and Microsoft Auto 4.0

MSN Direct provides still another method of gathering real-time data. It uses FM radio frequencies to send data to the car. Wi-Fi provides the greatest upload and download capacity, but, of course, availability is the key challenge.

Providing fast and reliable “always on” connectivity is still a challenge in a vehicle. You should design your application to provide a good user experience even when it is not connected.

Windows Automotive

Besides Microsoft Auto 4.0, you may have heard of a product called Windows Automotive. They are two completely different products and the chart at left shows some key differences between them. I highly recommend using Microsoft Auto instead of Windows Automotive for a number of reasons, including the fact that it contains more of the automotive middleware that is likely to help you in your development efforts.

Conclusion

Because of their tremendous capabilities, today’s automotive infotainment systems are more complex than ever. They have much more in common with personal computers than any automotive device of the past. In fact, they’re essentially PCs.

As a result of that complexity, most automotive systems developers are finding easier to build their unique projects on top of an existing, proven platform, such as Microsoft Auto 4.0, which already offers many of the basic automotive features they need. This platform gives them a good head start, allowing them to focus their resources where their products are unique. They can spend more time trying to build a competitive advantage than reinventing the wheel.

Next Steps

Contact your Bsquare Account Manager or call Bsquare at +1.888.820.4500 or +1.425.519.5900
Or email sales@bsquare.com for more information.

For more information, please visit www.bsquare.com. Or email us at sales@bsquare.com

At Bsquare

Bsquare is a solution provider to the global embedded device community. Our teams collaborate with OEMs at any stage in their device development to bring quality products to market faster. Since 1994, Bsquare has been a trusted partner to smart device makers worldwide.

Bsquare Headquarters
Toll-free +1.888.820.4500
Tel: +1.425.519.5900
sales@bsquare.com