Microsoft Auto 3.0 Platform Overview

Proven technology adapted for the auto industry

Published: March 2008

For the latest information, please see:
http://www.microsoft.com/windowsautomotive/default.mspx

Abstract

Automakers and auto suppliers are being challenged to meet a rapidly growing demand for in-car information, navigation, entertainment, and communication systems. However, controlling cost, delivering products to market in a timely manner, incorporating ever-evolving technologies, and delivering the added value needed to stand out in a competitive environment may all delay progress toward meeting market demand. Consumers demand a high level of quality and reliability and expect rich features that are easy to use. At the same time, the systems they desire are often complex to create. The in-vehicle environment provides limited computing resources and in-car applications usually need to be integrated across many different vehicle makes and models.

To help address these challenges, Microsoft, working in partnership with the automotive industry, has developed Microsoft® Auto 3.0. Microsoft Auto 3.0 is a powerful, scalable, and flexible software platform and reference hardware design that helps the auto industry deliver rich, integrated in-car infotainment systems faster, easier, and at lower cost. Microsoft Auto 3.0 provides a flexible solution that can help accelerate the time-to-market, expanding the capabilities of automakers and suppliers to make the most of the opportunities created by today's demanding consumers.

Contents

Introduction

Customers today demand the ability to stay connected with all of their information and entertainment sources while traveling in their cars. They want their mobile devices—such as mobile phones, portable navigation devices, and portable music players—seamlessly integrated into their vehicles.

According to research carried out by ABI Research, approximately 40 million navigation systems were sold in 2007, and 28 million of those were portable navigation devices. When this data is combined with research from the U.S. Department of Transportation and National Highway Traffic and Safety Administration estimating that Americans spend more than 500 million commuter hours per week in their vehicles and that 73 percent of mobile phone users talk on their phones while driving, it is clear that consumers increasingly want to stay connected, even when they are behind the wheel.

Microsoft is at the forefront of this trend, partnering with the automotive, mobile, and consumer electronics industries to provide enabling technologies that can help these industries quickly bring feature-rich, innovative, and cost-effective solutions to the market. The Microsoft® Auto 3.0 platform—a development platform that includes an extensive software framework and hardware reference design specifically engineered for the automotive industry—empowers application designers to develop a variety of integrated solutions that can help customers get to their destination while keeping them connected to the people, information, and digital entertainment that matters to them.

This white paper introduces Microsoft Auto 3.0 and describes the platform’s unique features. The paper discusses the challenges faced by automakers and suppliers in bringing integrated solutions to market, and it explains how the Microsoft Auto 3.0 platform helps overcome those challenges by providing the foundation for quickly and reliably creating a broad range of extensible, customizable, and advanced in-vehicle solutions.

The Challenges

While the market opportunity is ripe for a new generation of integrated in-car systems, automakers and suppliers face a paradoxical set of design requirements: rich (and typically very complex) features, high quality and reliability, rapid time-to-market, and competitive cost. While a vehicle is typically designed four years in advance, many consumer electronic devices have a life span as short as six months. This means that by the time the system actually hits the market, the consumer-facing technologies prevalent at design time might be close to obsolete, making “updateability” an important consideration.

For automotive manufacturers, there are the additional challenges of device manufacturing, vehicle integration, adaptability (devices are generally integrated into several vehicle models, thus amortizing the system cost to the automaker), and extensibility.

Meeting the Challenges

Microsoft is helping the automakers and suppliers meet these challenges through Microsoft Auto 3.0, a hardware and software reference design platform that empowers automakers and suppliers to more quickly and easily build solutions that provide consumers with the features they demand.

What Is Microsoft Auto 3.0?

Microsoft Auto 3.0 provides automakers, suppliers, and developers with the building blocks they need to quickly and reliably create a broad range of advanced in-vehicle solutions that meet the growing needs of automotive consumers.

Microsoft Auto 3.0 is based on the leading embedded operating system from Microsoft, Windows® Embedded CE 6.0 R2. Microsoft Auto 3.0 installs on top of Windows Embedded CE 6.0 R2 and adds automotive-specific functionality; this extends the capabilities of the operating system, yet maintains the flexibility of a Windows Embedded CE 6.0 R2 environment.

The Microsoft Auto 3.0 software stack includes all underlying drivers for the included hardware reference design, supporting middleware and application framework, and device management tools. The design uses technologies from various Microsoft groups, including Digital Rights Management, the Digital Media Division, Microsoft® Visual Studio®, Windows Embedded CE 6.0 R2, Windows®, and Windows Mobile®. The reuse of these existing, widely deployed technologies and components helps to ensure the stability of the designs.

Using various combinations of the Microsoft Auto system abilities, suppliers can make a variety of scenarios possible, including:

- Speech-based interaction with the user.
- Ability to play media from a USB storage device or to connect a media device to a USB port.
- Location information using GPS.
- Ability to easily and securely update itself to obtain new applications, features, or even operating system updates.
- Integration with mobile devices using *Bluetooth®* wireless technology.

All the other features of a navigation system (such as a graphical user interface, CAN/MOST access, and radio integration) can be added by any supplier.

Microsoft Auto 3.0 includes a wide variety of functionality, including:

- *Bluetooth®* 2.0+EDR support.
    - Including a range of profiles and technologies used for integration of mobile phones and portable media players.
- Hands-free phone integration of *Bluetooth®*-enabled mobile phones.
    - Access to the contacts on the connected mobile phone.
    - Send and receive Short Message Service (SMS) messages.
    - Rich call control and multiple-call scenarios.
    - Compatibility with a wide range of mobile phones.
- Integration of portable media players.
    - Support for a wide range of portable multimedia players (PMPs), including iPod, Zune®, Creative, and others.
    - Play media from an inexpensive USB storage device, or other removable storage.
    - Indexing and access to the metadata stored in the digital music brought into the car.
    - Play DRM-protected music from most media players.

- In-car device updatability.
    - Device management technology in Microsoft Auto 3.0 makes a range of update scenarios possible.
    - Ability to provide new applications in addition to service updates.
    - Updates can be applied by the owner of the vehicle, saving time and cost.
- Rich speech experiences.
    - Speech engines from Nuance Communications and Siemens AG are included for development purposes.
    - Speech components help integrate multiple applications into the speech system.
    - The tools and power are included to create rich speech experiences, as demonstrated by the Ford SYNC.
- Rich development tools.
    - Microsoft Auto 3.0 is built on Windows Embedded CE 6.0 R2.
    - Integration with Microsoft® Platform Builder and Visual Studio provides a rich development environment.
- Board Support Packages (BSP) are included for:
    - Freescale i.MX31 processor.
    - Texas Instruments Jacinto processor and the EVM prototyping circuit board.

Why Microsoft Auto 3.0?

Because the common underlying application functionality provided by Microsoft Auto 3.0 has been tested and implemented across multiple solutions with multiple automakers, the base software platform is known to be stable and robust. Automakers and suppliers can reuse their software components, reducing the time-to-market for their devices.

With Microsoft Auto 3.0, applications are developed for the platform rather than for a specific device model, resulting in amortized costs through easy portability. Microsoft Auto 3.0 provides the application functionality without specifying the user experience, so that automakers or suppliers can provide the unique experience that their customers demand.

Microsoft Auto 3.0 provides automakers and suppliers with:

- A general software platform, empowering suppliers to build a rich family of automotive devices, such as portable devices, gateways, and full head units.
- The ability to create and maintain a consistent platform, making connected-car experiences possible.

Microsoft Automotive Business Unit

Microsoft created the Microsoft Automotive Business Unit (ABU) in 1995 in response to the growing desire of consumers to remain connected to their information and entertainment while traveling in their cars. The ABU—a multidisciplinary group composed of product developers and business leaders in North America, Japan, and Germany—is a dedicated partner to the automotive industry. The ABU provides innovative technologies and flexible software to help deliver reliable, easy-to-implement, and cost-effective in-car infotainment solutions that can help automakers and suppliers distinguish themselves in the marketplace.

Available Now

Drivers and passengers can now experience Microsoft technology in the Fiat Blue&Me™ and the Ford Sync.

Fiat Blue&Me

Fiat Auto Group and Microsoft jointly developed the infotainment system Blue&Me, which empowers customers to connect their personal mobile devices with the integrated solution found in many vehicle models from Fiat, Alfa Romeo, Lancia, and Fiat Light Commercial Vehicles. The competitively priced Microsoft Auto–based infotainment package comes with *Bluetooth®* wireless technology and USB connectivity, letting drivers listen to music from their personal USB storage devices. The integrated voice-controlled hands-free phone kit in the vehicles connects to a large number of mobile phone models, in addition to digital USB storage devices. The Blue&Me system is based on a modular structure, and it can be easily updated to support different services. For example, customers can download language packs from the Fiat Web site and update their system to support a language not originally installed.

Ford Sync

Ford Sync is a factory-installed fully-integrated in-car communications and entertainment system developed by Microsoft and Ford. The Microsoft Auto–based Ford Sync is a consumer device gateway similar to the Fiat Blue&Me, but with significant new features. Ford Sync provides drivers with hands-free voice-activated control over mobile phones and digital music players. Ford Sync automatically connects phones and music players with the vehicle's in-car microphone and sound system. Most popular media players work with Ford Sync, including iPod, Zune, "Plays for Sure" players, and most USB storage devices. Supported audio formats include MP3, AAC, WMA, and WAV.

Ford Sync is based on an ARM 11 processor, 64 MB of DRAM, and 256 MB of flash memory. Customers can use the USB port to update the software to work with the newest personal electronic devices—this is an important advantage, as customers tend to change devices more frequently than they change vehicles.

Ford Sync debuted in the fall of 2007 on 12 different 2008 models of Ford, Mercury, and Lincoln vehicles. By the end of 2009, Ford will install Sync on all vehicle models.

Continental AG Multi Media Platform (MMP)

Continental AG is using Microsoft Auto for its Multi Media Platform (MMP), which provides powerful, secure, flexible, and easy-to-update in-vehicle multimedia systems. The Continental MMP software architecture clearly separates vehicle functions and the functions related to entertainment features, making it possible to quickly react to future innovations and market trends. In the MMP hardware design, Continental uses a scalable concept to ensure top performance at high integration. For high-end systems, another CPU and graphics processor provides additional power for online services and 3D graphics applications.

The hardware is equipped with standard consumer electronics interfaces, so that mobile devices (such as USB storage devices, iPods, or SD cards) can be easily networked with the MMP. The MMP also provides a *Bluetooth®* wireless technology interface, so that consumers can use mobile phones for hands-free phone calls or mobile data services.

Microsoft Auto 3.0 Platform Overview

Microsoft Auto 3.0 is designed to implement state-of-the-art infotainment systems for today's automotive industry; it is an ideal application development platform because it provides a rich programming environment that empowers application developers to add functionality. The flexible Microsoft Auto 3.0 platform targets a wide range of devices, including connectivity gateways, connected radios, multimedia devices, and navigation head units.

The following diagram gives an overview of the various building blocks that make up the Microsoft Auto 3.0 system. The light red sections represent components provided by Microsoft Auto 3.0. The dark blue sections represent components provided by suppliers or third parties. The light blue vertical strip represents developmental tools; these tools bring the entire system together and cross all layers of the software stack. In the combined dark blue and light red section, some of the components are provided by Microsoft Auto 3.0, and some are provided by third parties.

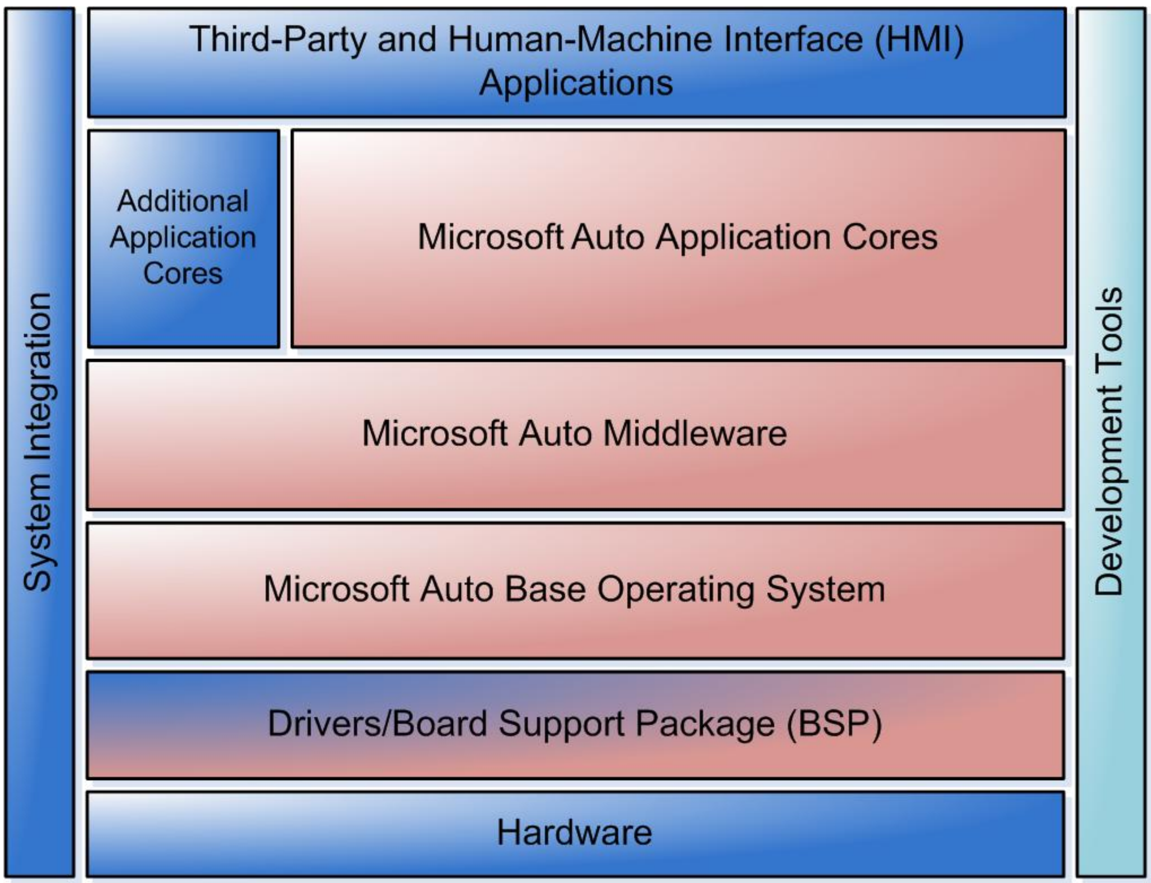

Figure 1 Microsoft Auto 3.0 platform

The Microsoft Auto 3.0 platform is made up of the following components:

- **Hardware.** Provided by the suppliers.
- **Drivers/Board Support Package (BSP).** Provided by both Microsoft and by suppliers.

- **Microsoft Auto base operating system.** Windows Embedded CE 6.0 R2, written from the ground up to provide a real-time operating system for embedded devices.
- **Microsoft Auto middleware.** The Microsoft Auto middleware is much of the "plumbing" for the software platform. Examples include the *Bluetooth®* wireless technology stack, the hands-free mobile phone service, and the media player engine.
- **Microsoft Auto application cores.** The application cores are the most visible part of the software platform. The APIs are organized and structured in a similar way to the desktop version of Windows so that computer knowledge and techniques can be reused. This makes it possible for new development resources to be deployed and made productive more quickly. The applications have been designed so that the Human-Machine Interface (HMI) is easily separable; for example, the media player is composed of a media player core and a supplier-provided application HMI.
- **Third-party and HMI applications.** Third-party and HMI applications interact with the user. This portion of the application can be easily changed without disturbing the underlying application.

Hardware Reference Design

Microsoft Auto 3.0 provides a hardware reference design based on the Freescale™ i.MX31 microprocessor. (Suppliers provide the final hardware.) The Microsoft Auto 3.0 software is compatible with other ARM-based microprocessors, and also includes a board support package for the Texas Instruments Jacinto processor.

Functionality and features of the system rely on some fundamental primitives supported by the base system hardware components (such as management of device power states and transitions, management of NAND flash, and support for data transfer over USB ports)—these are supported by the hardware reference design. For example, the flash driver provides support for primitives required for reading and writing to NAND flash to various system components, including file systems and low-level boot loaders. The flash driver is responsible for handling bit errors and bad blocks, helping to ensure power-safe operation of the flash memory and distributing flash writes evenly to maximize flash life. The hardware reference design uses 256 MB NAND flash memory to store the operating system image, the applications, the speech engines, and additional application data. The actual requirements of a specific device depend on the scenarios and data that are required to be supported. Additional details of the hardware reference design are shown in Appendix 1.

The hardware reference design is not mandatory to build devices based on Microsoft Auto 3.0. A supplier can choose one of the supported processor platforms that ship with Microsoft Auto 3.0, or they may choose to develop a Windows Embedded CE 6.0 Board Support Package (BSP) for another processor family. Documentation is available on the requirements for transforming a standard Windows Embedded CE 6.0 BSP into one that can fully support the functionality of the Microsoft Auto 3.0 platform. The layered architecture, the operating system, middleware, and applications should not be significantly affected when repurposing a standard Windows Embedded CE 6.0 BSP into one that supports Microsoft Auto 3.0.

Board Support Package

The Board Support Package includes hardware-specific components such as a boot loader, an OEM adaptation layer (OAL), device drivers, and run-time image configuration files.

The OAL includes the lowest-level components that initialize the CPU, peripherals, and other hardware modules. The OAL interacts directly with the hardware and abstracts the specifics of the hardware from the upper layers. The OAL is typically built when developing the BSP; it is built as a library (HAL.lib) and linked into the overall kernel executable (Nk.exe).

Operating System

The base operating systems layer provides the platform-level features of the Microsoft Auto 3.0 system. This layer includes support for:

- File systems.
- Windowing and focus management.
- Speech API components that expose the chosen speech recognition and text-to-speech (TTS) engines.
- Access to operations supported by the various hardware modules on the system.
- Networking transports and protocols (TCP/IP and *Bluetooth®* wireless technology).

The base operating system is built on top of Windows Embedded CE 6.0 R2. The Windows Embedded CE 6.0 R2 kernel exposes core system functionality and provides the infrastructure through which the rest of the system software can interface with the OAL. The Windows Embedded CE 6.0 R2 kernel is primarily responsible for process and thread management, loader, predictable thread scheduling, memory management, and interrupt handling and support for system calls.

Features of Windows Embedded CE 6.0 R2 include:

- A Win32 API subset, including file and memory management, device and service management, threads and process management, and networking stacks.
- Platform Builder 6.0 and other powerful development tools.
- Multilanguage support.
- Windows® Internet Explorer® Web Browser Control for Windows CE (based on Internet Explorer 6), with a user interface that can be replaced by the automaker.
- Rich multimedia support through the Microsoft® DirectShow® API, with support for a variety of formats such as Windows Media® Audio and MP3.
- High-performance graphics support through the Microsoft® Direct3D® Mobile API and the Microsoft® DirectX® API.

Middleware and Application Cores

The next layers up in the software stack are the Microsoft Auto 3.0–specific middleware and services. These components define the heart of the Microsoft Auto 3.0 platform; they distinguish this platform from standard Windows CE and from the other platforms based on Windows CE technology.

A detailed view of the system building blocks that make up the software platform components for Microsoft Auto 3.0 is shown in Figure 2.

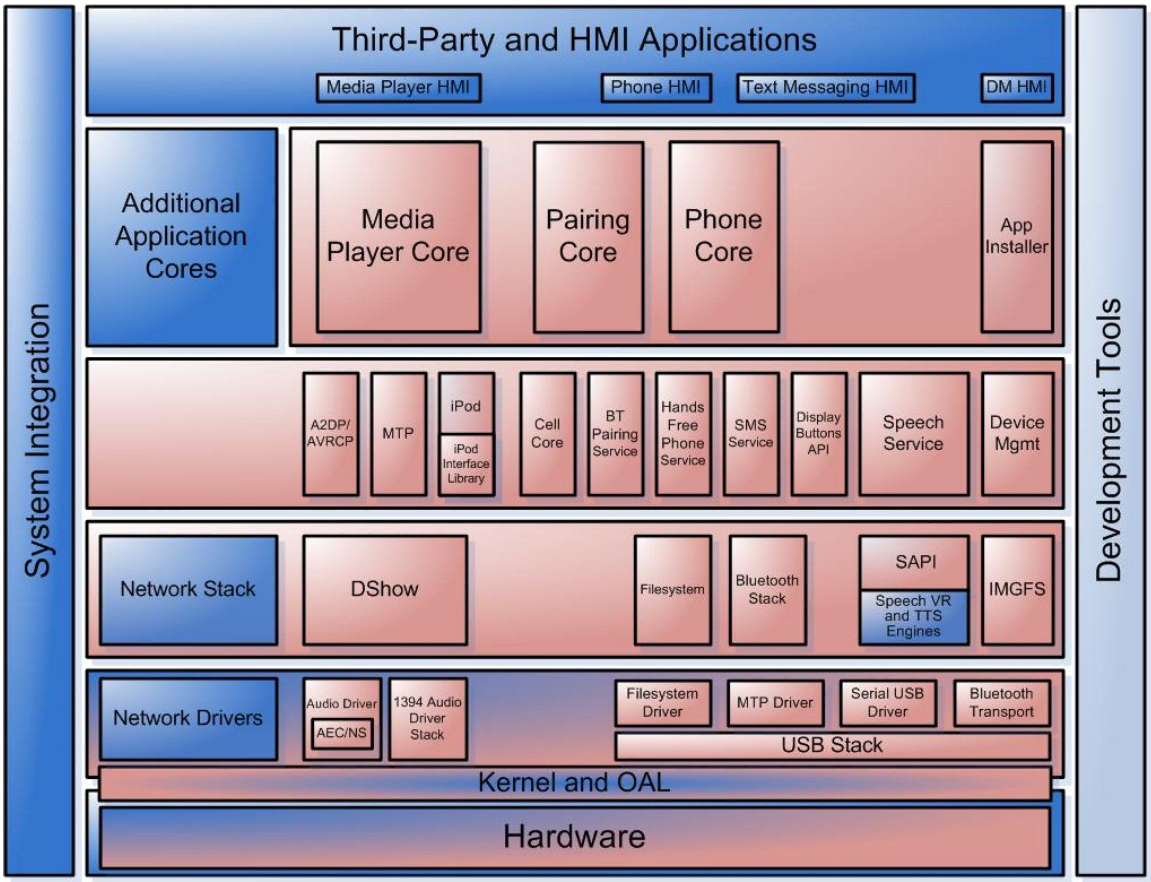

Figure 2 Platform components

The middleware components provide a stable foundation for connectivity and consumer device applications. Additional components adapting the platform to specific requirements (for example, the MOST network stack) can be added by third parties or by suppliers.

The middleware layer provides the infrastructure support required to easily develop applications while using the powerful features exposed by the base platform. Services in this layer include the following:

- **Speech service.** Provides a robust speech API that sits on top of the Microsoft Speech API (SAPI). This lets many applications share the recognition and text-to-speech engines and provides common controls that make it easy for applications to have a rich dialog with the user with minimal application work.

- **Display service.** Provides support for applications that interact with the user through display messages, including both a display driver that supports connections to multiple different types of vehicle displays and a display API that makes it possible for an application to write to the display after a connection to the display is established.
- **Buttons support.** Provides the ability to use buttons that may have different underlying hardware (in different vehicle models) in a uniform way using a notion of virtual keys and key-up and key-down events.

Additional services include vehicle networking and diagnostics services, hands-free mobile phone service, data communication and connection management, and device management service.

Applications written for Microsoft Auto 3.0 can use a broad array of C/C++ APIs. The base Windows Embedded CE 6.0 R2 system exposes an API for its services (including memory and process management, file systems, registry access, and so on). The automotive-specific platform components expose APIs for support of functionality such as vehicle networking, speech, hands-free telephony, and diagnostics.

Bluetooth Wireless Technology

Microsoft Auto 3.0 provides a modular *Bluetooth*® 2.0+EDR–compatible software stack that is built on the Windows Embedded CE 6.0 *Bluetooth®* stack. Deployed Microsoft Auto 3.0–based systems that make use of the Microsoft Auto 3.0 *Bluetooth®* solution may be updated in-field with the device management sub-system. The platform supports a pairing service, hands-free functionality, dial-up networking client, OPP/OBEX, streaming audio using A2DP, and the serial port protocol.

With the *Bluetooth*® wireless technology software stack, devices (such as PDAs and mobile phones) can be paired and connected with Microsoft Auto 3.0 devices. The software stack supports a number of *Bluetooth*® wireless technology profiles used in automotive solutions (including PBAP, A2DP, AVRCP, HFP, and DUN). Windows Embedded CE 6.0 R2 includes additional *Bluetooth*® wireless technology profile implementations that have not been explicitly tested with Microsoft Auto 3.0, but could provide a basis for additional solutions. Appendix 2 provides further information about the Microsoft Auto 3.0 *Bluetooth*® stack.

Bluetooth Pairing Core

The *Bluetooth®* wireless technology pairing core is one of the application cores used to provide core services to Microsoft Auto 3.0 applications; many Microsoft Auto 3.0 applications depend on this API to pair and communicate with *Bluetooth®* wireless technology–enabled devices.

Media Core

With the media core, developers can work with any of the wired device types through a single standard interface, providing the customer with a single experience regardless of the type of wired media device that is connected. The user experience can be driven by buttons on the display or by voice, depending on the requirements of the supplier and automaker.

The media core module is responsible for actual media playback, metadata indexing, hardware event handling (for example, power and speech), keeping all the metadata of all tracks, and maintaining a "now playing" list with history and shuffling ability. All sorting of metadata is done at this level. The media core also supports use of the A2DP and AVRCP *Bluetooth®* profiles to enable playback of music wirelessly from phones and other devices that support those *Bluetooth®* profiles. The media core adds media capabilities that include:

- **Zune support**. Microsoft Auto 3.0 also offers a software add-on package that lets a device fully interact with all available Zune devices for playback of audio content through a USB connection. The Zune support includes full support for all DRM-protected content purchased through the Zune Marketplace or obtained through the Zune Pass subscription service.
- **MTP device support**. MTP-based devices from companies such as Sansa, Creative, and iRiver are also supported, including the DRM-protected content on those devices. These types of devices are connected via USB.
- **iPod support**. Older generation iPods are supported through 2-wire style connections; newer generation models that require a 1-wire connection with Apple Authentication are also supported. Playback of all audio content (including content protected by the Fairplay DRM mechanisms) is supported, as are the popular iPhone and iPod Touch.
- **Mass storage device support**. The media core supports the ability for a user to bring in digital media that is not DRM-protected into the vehicle on mass storage devices, such as USB storage devices and SD cards.
- **Supported media formats**. The media core can access, index, and play files that are in WMA, MP3, PCM WAV, and AAC formats. The media core also supports playlists, including those in M3U and WPL formats in addition to the native formats supported on iPods and Zunes.
- **Browsing media files**. Regardless of the type of the physically connected device, the media core creates a full index of the audio media on the device based on the metadata included in the digital media files. This helps ensure that the user experience is the same, regardless of the device type attached. The user interface can be driven by a folder-based hierarchy or can be filtered by a variety of available metadata, including track name, album name, artist, and genre. This index is maintained per device by the media core, and the index for each known device can also be persisted (enabling fast access to the same device the next time it is brought into the vehicle).
- **Playback control**. The media core makes it possible to have full playback control over the media files that it has indexed, including play, pause, fast forward, rewind, next track, and previous track.

Phone Core

The phone core provides connectivity and control of mobile phones to make and receive phone calls using the *Bluetooth®* HFP profile. The phone core supports an automatic download of the contact list from the mobile phone. (The user's contact list can be presented either by display in the car or by voice, depending on the application Human-Machine Interface.)

- **Automatic synchronization of contacts**. The phone core can download the phonebook from mobile phones that support the functionality through *Bluetooth®* wireless technology. After the initial phonebook download, the data is persisted on the device, so the user has almost instant access to the information the next time the phone is connected to the device. If new contacts have been added to or deleted from the phone since the last time it was connected to the device, the phone core automatically downloads the new phonebook in the background while the customer maintains access to the persisted phonebook. Contacts can be automatically downloaded via PBAP, SyncML, or AT commands. The phonebook can also be filled through an OPP/OBEX vCard object that is push-initiated by the user.

- **Extensive call control capabilities**. 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. Call waiting information is also supported. The phone core also downloads and makes the mobile phone's call history available, so that the application can expose that information to the customer as well. The device can also send the phone call audio to the handset by toggling the privacy mode of the phone.
- **SMS message support**. The SMS router provides access to the SMS messages that are received by the *Bluetooth®*-connected phone, for those phones that support SMS message access through AT commands.
- **Phone status support.** The phone core supports the ability to get the carrier name, battery level, network state, and signal strength information for those phones that expose that information over *Bluetooth®* wireless technology.
- **Broad handset compatibility.** Microsoft Auto 3.0 regularly ships device compatibility updates, empowering partners to issue updates to their in-car devices that include support for the latest mobility devices available on the market.

Speech Service

Microsoft Auto 3.0 provides a speech-based user interface enabled by the speech service, which can host a variety of SAPI 5.0-compliant speech recognition (SR) and text-to-speech (TTS) engines. The speech service also uses a speech prompt engine, which, when combined with pre-recorded and dynamically synthesized prompts, creates a more natural user experience. The speech engines are interchangeable, providing for inexpensive support for several languages.

The speech service provides a set of intuitive, high-level speech controls designed to enable rapid speech application development for non-SAPI experts. It also performs system-wide speech-related bookkeeping and management services, such as managing the global grammar and arbitrating access to the speech focus.

Language availability is limited only by the speech engine and TTS vendor language catalogs. There are three voice recognition and TTS engine combinations included in Microsoft Auto 3.0 for development purposes only (two sets are from Nuance [Scansoft] for both VR and TTS, another comes with VR from Siemens AG and TTS from SVOX). The supplier and automaker must choose a speech engine vendor and obtain licensing through the vendor for the engines and languages, and must then tune the system before deployment.

GPS

Microsoft Auto 3.0 supports GPS positioning of the vehicle; it then communicates this position to applications that need the information. Microsoft Auto 3.0 uses the Windows Embedded CE 6.0 R2 GPS Intermediate Driver (GPSID) architecture—a developer can create a driver that feeds the GPSID with GPS data (either from a GPS chip onboard the device, GPS available on the vehicle network, a *Bluetooth®* wireless technology connection that is feeding GPS signals, or another source). Since the developer controls the functionality of the source of the GPS data, that data can be raw feeds directly from a GPS chip or it can be corrected GPS data based on dead-reckoning algorithms. The GPSID architecture then offers multiplexed application access to the GPS data through a standard interface at the top edge of the GPSID component.

Connection Manager

The connection manager is the central component for managing connections on the Microsoft Auto 3.0 platform. The connection manager provides an API to let applications request connections, specify priorities, and close connections after use.

A supported connectivity feature is the firewall module. Windows Embedded CE 6.0 R2 brings an integrated firewall module to Microsoft Auto 3.0 that is configured using registry settings; with this module, Microsoft Auto 3.0 is flexible enough to support a wide range of firewall scenarios and applications.

Device Management Sub-System

The device management (DM) sub-system supports mechanisms for updating applications, configuration settings, and system software on Microsoft Auto 3.0 at different stages during its life cycle (see Figure 4). Generally, the device management scenarios that are enabled are determined by the policies of the automaker. The key goal of DM is to maintain a unified software inventory on the Microsoft Auto 3.0–based device, regardless of the update method.

Supported DM use cases include:

- Installing application software, a complete image, or critical patches and fixes.
- Uninstalling or reinstalling application software.
- Activating or deactivating installed applications.
- Adjusting the user’s setup.

The provisioning mechanism in Microsoft Auto 3.0 relies on the download of standard CAB files to the device. There is a standard component available that unpacks, verifies, and installs CAB files; a limited scripting capability makes it possible to modify registry and other device-specific settings. The CAB unpacking mechanism is principally independent of the CAB file contents, so this mechanism could also be used to download installable content for other electronic control units networked to the Microsoft Auto 3.0–based device.

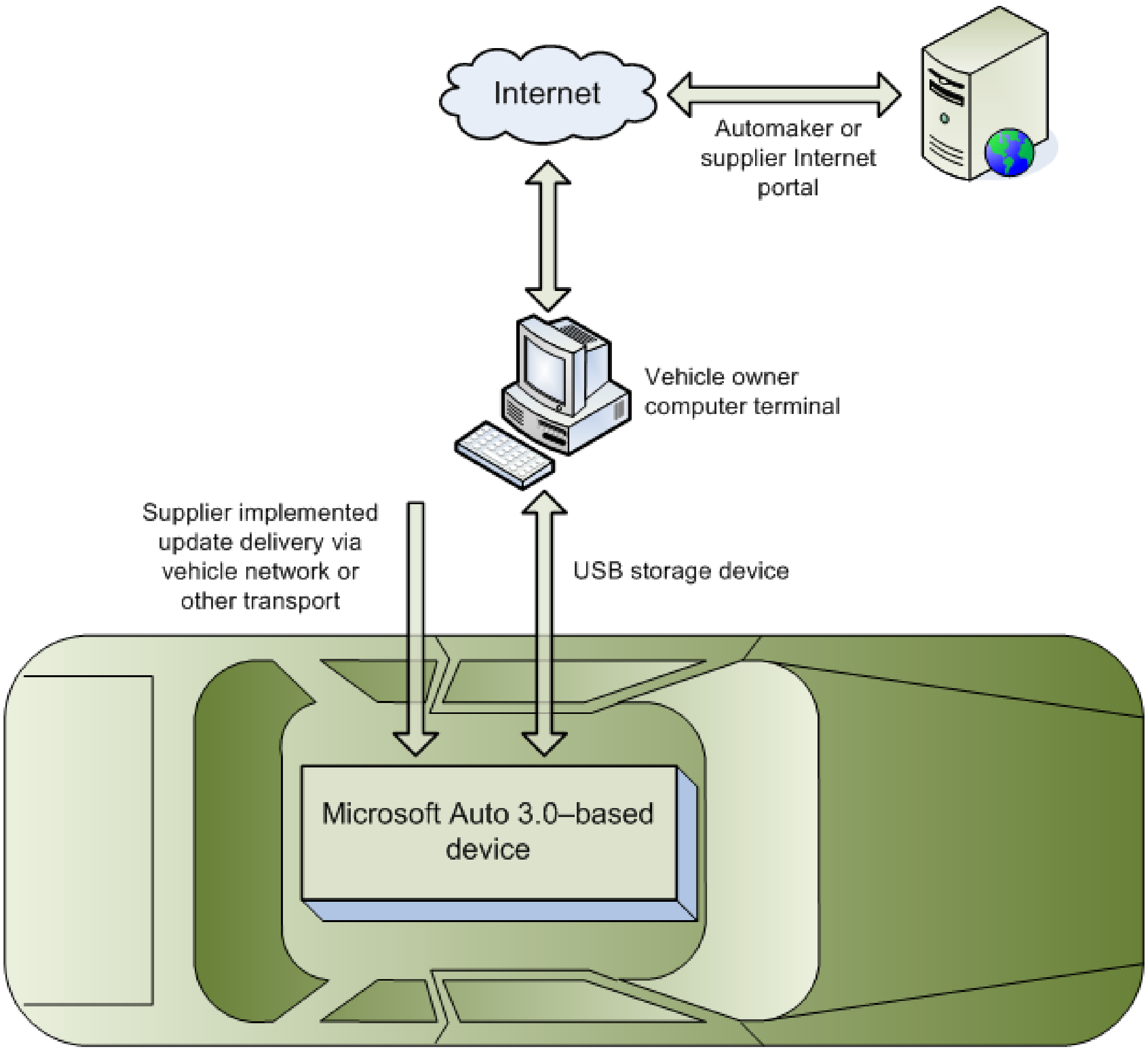

Figure 3 Device management

The overall DM solution in Microsoft Auto 3.0 is primarily based on core Windows Embedded CE DM components, with some enhancements:

- **Single installation API.** A single installation API is used instead of several installer executables. The installation API can handle all types of installation packages. This enables effective reboot handling because information on the installation status can be exchanged more easily, and installation can be paused, stopped, and resumed by using an API instead of an executable file. The API supports compressed CAB files for better bandwidth utilization.
- **Support for unattended uninstall/upgrades.** CAB file uninstallations are, by default, delayed until key-off. This makes uninstallation of an application in use nonintrusive to the customer.
- **Super CAB files.** Super CAB files (PKS) bundle packages together into one larger package. These may involve a set of application CABs or a set of system updates.

The device update process has two steps:

- Obtaining the actual file (the update image) onto the device (delivered on a USB storage device or an alternate delivery method implemented by the supplier).
- Performing the actual update by verifying and extracting the update image and installing.

Suppliers can select whether updates will be obtained from a USB storage device —the correct CABs and INF file is attached to the USB host port and an update is initiated manually—or automatically based on an ignition cycle.

The installer is involved in actual verification and installation. The installer maintains a persistent installation database that can be accessed by other applications and services to determine the software inventory and package installation status. The installer itself is a DLL that exposes an installation API (as opposed to a standalone process).

For system-update packages, the installer triggers the Image Update mechanism. This causes a reboot into the Update Loader (ULDR). The ULDR verifies and applies updates to the operating system. Update progress and status is carefully tracked across reboots and reported back to the Installer DB.

Using a centralized installation database to maintain the state of each downloaded package, the DM engine can continue the DM session after a reboot. Reboots are usually delayed until key-off. When the device is turned on again after a reboot, the DM session continues from where it left off.

Security Sub-System

The security sub-system helps to ensure that Microsoft Auto 3.0 does not accept or run untrusted executable code on the system. Microsoft Auto 3.0 is engineered with adherence to the Trustworthy Computing (TwC) security standards enforced at Microsoft. The most critical security mechanism for Microsoft Auto is its application/operating system trust model. Authenticode code signing technology is used to help ensure that Microsoft Auto does not install or run untrusted code.

It is possible to configure the Microsoft Auto 3.0 platform so that all updates to the image (whether they be applications or system updates) must be signed. For each of the different types of updates possible, there are different certificate stores to enforce this policy. Some examples include:

- **Secure Image Download.** At the lowest level, certificates in the secure image downloader ensure that only certain signed images can be used to completely re-flash the system.
- **Operating System Image Updates.** Each of the image update packages has a certificate store (stored in a binary blob called Device Side Manifest [DSM] in the package).
- **Application Installation.** Application CABs are signature verified upon download from the USB storage device. Certificates in the installer policy store determine whether the CAB should be accepted or not.
- **Device Authentication.** A Service Credential object is provided to applications for enabling device authentication.

- **Other Security Mechanisms.** Several additional sub-systems employ mechanisms to help protect against threats that apply to them. For example:
  - *Bluetooth®* wireless technology connection with an external device is made over a secure link.
  - Application CABs can be signed with a flag that prevents their installation from USB storage devices to limit piracy concerns.
  - Diagnostics Security Access is implemented to allow certain operations to be performed only if the client presents the necessary evidence.

To manage certificates and encryption keys on the device, Microsoft Auto 3.0 (through Windows Embedded CE 6.0) provides CryptoAPI.

Reliability Sub-System

Several subcomponents in Microsoft Auto 3.0 provide support for system reliability.

- **Hardware watchdog.** Helps ensure that the system is reset if it appears locked.
- **System health monitoring.** Hosted in SysHealth.exe and contains a launch monitor (helps ensure that the set of applications considered critical for system functionality start up normally), memory monitor (watches currently available memory and schedules a deferred or immediate reboot if it is below a low or critically low threshold), periodic reboot (configurable by the supplier), and RTL_ZONE logging (collects a retail message log for the system).
- **Process monitor.** Provides a timer-based software watchdog; applications can request a process termination or a system reboot if they happen to hang.
- **Reliability service.** Provides the ability to request system reboots and enforces logic for the system to shut down completely if recurrent reboots are detected within a short period of time.
- **Flash driver.** Implements several mechanisms to help ensure a long flash life, including wear-leveling, bad block handling, main and spare-area error correcting codes, and protection against flash sector corruption because of unexpected power failure.
- **Backup installer**. System switches to the backup installer database (DB) upon detection of any kind of corruption in the active DB, the installer switches back to the backup DB.

Whenever the system schedules a planned reboot, it notifies applications through power management events. Planned reboots occur in situations such as installer scheduled reboots (upon application or system updates) and SysHealth reboots. Whenever the system suspends or reboots in a planned manner, it flushes data to the disk. If the system has to reboot because of catastrophic conditions (such as a battery disconnection, a reset from the hardware watchdog, or a critical process failure), it is not possible to notify applications to save data.

Human-Machine Interface Layer

The Microsoft Auto application model enables a clear separation of Human-Machine Interface (HMI) from the core application logic. The HMI framework, technically part of the Windows application framework, facilitates the separation of the HMI portion of the application from the computational or processing portion. This lets the core of the application be written once; the look and feel of the user interface can then be readily customized. The core applications can be updated without change to the HMI, and vice versa.

The supplier or automaker can write applications in the HMI layer that take advantage of the application cores and middleware components. Microsoft Auto 3.0 provides the flexibility to fit into almost any user interface paradigm the automaker may choose—the middleware offers all of the core functionality that lets the supplier present it to the customer in a way that meets the requirements of each individual automaker.

The HMI sample applications included in Microsoft Auto 3.0 include the media player, phone application, device management HMI (for installing applications and image updates from an attached USB storage device), *Bluetooth®* wireless technology pairing application, and SMS reader.

Delivering a Microsoft Auto 3.0–Based Solution

Application designers can rely on the rich, familiar development environment provided by Microsoft Auto 3.0. For a “future-proof” device, Microsoft Auto 3.0 uses a secure, standard, and stable update and installation technology from Windows Embedded CE 6.0 R2 and Windows Mobile.

Microsoft Auto 3.0 supports a powerful API set and provides an intuitive development framework that application developers can use. The toolset and framework are familiar to the general development community (which might not necessarily have automotive software development experience).

Microsoft Auto 3.0 provides easy access to various ports and peripherals during the development and testing phase—both for system developers and application developers. This includes mechanisms in the hardware and in the software image to enable easy development, download, testing, and debugging of system images and applications. Additionally, Microsoft Auto 3.0 enables designs to meet requirements from various testing phases, such as unit testing, functional testing, system-integration testing, and in-vehicle integration testing.

The Microsoft Auto 3.0 development tools include:

- Platform Builder 6.0
- Visual Studio 2005
- Development hardware
- Microsoft Auto 3.0 Platform Development Kit

Platform Builder 6.0

Platform Builder is a standard integrated development environment (IDE) for Windows CE–based devices. Platform Builder 6.0 ships with Windows Embedded CE 6.0 R2 and runs as an add-on to Visual Studio 2005—providing a more consistent development experience for both application and platform developers.

Microsoft Auto 3.0 uses Platform Builder 6.0 primarily for the following scenarios:

- Developing (edit, compile, debug) of native C and C++ code on the platform.
- Flashing a development board with new software images.
- Monitoring the debug trace output from a running system.
- Using remote tools to measure and monitor various parameters of the device.
- Executing some automated tests of the device.

Visual Studio 2005

Visual Studio 2005, the standard development suite for desktop Windows development, features an extensive set of tools, configuration samples, and guidelines that improve productivity from the initial design to final testing and tuning.

Visual Studio 2005 is required for hosting Platform Builder 6.0. Using the Platform Builder mode, the developer can create and debug the actual operating system platform on which the HMI applications will run. Using Visual Studio in application mode enables development and debugging of applications running on top of the platform image already running on the device.

Development Hardware

The development and testing for Microsoft Auto 3.0 is typically done on prototype hardware rather than through software emulators. The development hardware should have available either a serial port and a USB device or an Ethernet port for connecting to a desktop computer. Microsoft Auto 3.0 does provide a hardware reference design—details are given in Appendix 1.

Platform Development Kit

Microsoft Auto 3.0 ships as a Platform Development Kit (PDK) that installs on top of Windows Embedded CE 6.0 R2; it is used in conjunction with Visual Studio 2005 and Platform Builder 6.0 for development of the complete software stack for a specific device. The Microsoft Auto 3.0 PDK includes source code and binaries for the two supported processor platforms (MARPF1 and TI Jacinto EVM Board), binaries for the Microsoft Auto middleware components, documentation, and the command-lines tools necessary to create, modify, and extend a Microsoft Auto 3.0–based device image.

Summary

Microsoft Auto 3.0 provides a proven, reliable, and extensible software platform and hardware reference design on which automakers can distinguish themselves by building innovative solutions to help drive sales and customer loyalty. It provides automakers, suppliers, and developers with the building blocks they need to quickly and reliably create a broad range of advanced in-vehicle solutions that meet the growing needs of automotive consumers and set them apart from the rest of the field. Integrated features such as *Bluetooth®* wireless technology, speech recognition, and media player support are built into the platform for rapid deployment without heavy development time. Application designers can use the comprehensive software solution that is provided, or they can easily extend Microsoft Auto 3.0 to their own unique implementation. Developers may also reuse existing, proven software components, freeing up development resources and saving design cycles. The Microsoft Auto 3.0 platform is modular, so developers can use the components they wish and exclude or substitute others—from the hardware reference to any of the middleware components. The hardware reference design, along with robust software tools, also results in faster time-to-market. Many software components can be reused across product lines and models, saving time and money.

Microsoft Auto 3.0 provides:

- A reliable, robust platform.
- Increased productivity and reduced costs.
- Flexibility, scalability, and extensibility.

Reliable, Robust Platform

With support for industry-specific hardware and software, Microsoft Auto 3.0 provides a reliable, high-quality platform. Microsoft Auto 3.0 uses the familiar WIN32 API set that contains a powerful interactive developer environment that meets a wealth of automotive-relevant requirements. The Windows CE 6.0 core is reliable, extensively tested and widely used—this provides the foundation and tools for quickly and reliably creating a broad range of extensible, customizable, and advanced in-vehicle solutions.

Increased Productivity and Reduced Costs

Microsoft Auto 3.0 enables automakers and suppliers to develop their infotainment solutions with fewer design cycles and get them into production more quickly. Microsoft Auto 3.0 uses a comprehensive middleware stack, eliminating the need for additional hardware and removing the inefficiencies of creating ad hoc software solutions; the result is faster, less expensive and more reliable integration.

Flexible, Scalable, and Extensible

Automakers and suppliers can help ensure that their systems stay current with the latest devices and services, thanks to the inherently flexible design and built-in software update mechanisms of Microsoft Auto 3.0. Unique solutions and customized user interfaces can also be easily built on top of the existing platform. The open architecture is designed specifically to enable a designer to easily extend functionality with custom solutions.

Glossary

A2DP

Advanced Audio Distribution Profile. A2DP defines how high quality audio (stereo or mono) can be streamed from one device to another over a *Bluetooth®* wireless technology connection.

AAC

Advanced Audio Coding. AAC is a standardized, lossy compression and encoding scheme for digital audio that is designed to be the successor of the MP3 format. AAC generally achieves better sound quality than MP3 at many bit-rates.

API

Application Programming Interface. An API is a source code interface that an operating system or library provides to support requests for services to be made of it by computer programs.

ARM

Advanced RISC Machine or Acorn RISC Machine. ARM is a 32-bit RISC processor architecture that is widely used in many embedded designs. Because of their power-saving features, ARM CPUs are dominant in the mobile electronics market, where low power consumption is critical.

ASX

Advanced Stream Redirector. One of the three Windows Media metafile formats (ASX, WAX, and WVX). The ASX file is a metafile (a file that contains data about another file) a reference to an ASF file.

AT commands

The Hayes command set, also called the AT (for attention) command set, is used by dial-up modems. The command set consists of a series of short strings that combine together to produce complete commands for operations such as dialing, hanging up, and changing the parameters of the connection.

AVRCP

Audio/Video Remote Control Profile. AVRCP is designed to provide a standard interface to control devices to let a single remote control to control all of the audio/visual equipment to which a user has access.

Binary blob

Binary Large Object. A binary blob is an object file that is loaded into the kernel of an open source operating system (not usually code running outside the kernel such as BIOS).

Bluetooth® wireless technology

An industrial specification for wireless personal area networks, named after the 10th century king of Denmark, King Harold Bluetooth. *Bluetooth®* enables connection and information exchange between devices such as mobile phones, laptops, personal computers, printers, digital cameras, and video game consoles over a secure, globally unlicensed short-range radio frequency.

Bluetooth® 2.0+EDR version introduced an Enhanced Data Rate (EDR) of 3.0 Mbit/s (basic signaling rate, the practical data transfer rate is 2.1 Mbit/s).

Boot loader

A program whose only job is to load software to start the operating system.

CAB file format

The Microsoft Windows native compressed archive format. It supports compression and digital signing, and is used in a variety of Microsoft installation engines, including Setup API, Device Installer, AdvPack (for the installation of ActiveX components from Internet Explorer) and Windows Installer).

Codec

A device or program capable of encoding and decoding a digital data stream or signal. Microsoft Auto 3.0 only provides production-licensed decoders for Windows Media Audio (WMA) and a development license for MP3.

CAN

Controller Area Network. CAN is a computer network protocol and bus standard designed to enable microcontrollers (microprocessor designed for high integration, low power consumption, self-sufficiency, and cost-effectiveness, in contrast to a general-purpose microprocessor) and devices to communicate with each other and without a host computer.

CryptoAPI

Cryptographic Application Programming Interface. An API included with Windows operating systems that provides services so that developers can secure Windows-based applications using cryptography. It is a set of DLLs that provides an abstraction layer that isolates programmers from the code used to encrypt the data.

DirectShow

A multimedia framework/API produced by Microsoft. DirectShow can be used by software developers to perform various operations with media files or stream; DirectShow is based on the Windows Component Object Model (COM) framework and provides a common interface for media across many programming languages. It is an extensible, filter-based framework that can render or record media files on demand.

DirectX

A collection of APIs for handling tasks related to multimedia, especially game programming and video, on Microsoft platforms.

DLL

Dynamic-Link Library. DLLs are implementations of the shared library concept in the Microsoft Windows and OS/2 operating systems, and they have the file extension DLL, OCX (for libraries containing ActiveX controls), or DRV (for earlier system drivers). DLLs can contain code, data, and resources, in any combination.

DMA

Direct Memory Access. DMA enables hardware sub-systems within the computer to access system memory for reading/writing independently of the central processing unit. Many hardware systems use DMA, including disk drive controllers, graphics cards, network cards, and sound cards. DMA channels can transfer data to and from devices with much less CPU overhead.

DRAM

Dynamic Random Access Memory. DRAM is a type of random access memory (RAM) that stores each bit of data in a separate capacitor within an integrated circuit. The capacitor charge is refreshed periodically, as opposed to static memory.

DRM

Digital Rights Management. DRM refers to the access control technologies used by publishers and copyright holders to limit usage of digital media or devices.

DUN

Dial-Up Networking Profile. DUN provides a standard to access the Internet and other dial-up services over *Bluetooth®* wireless technology. DUN can be used to access the Internet from a laptop by dialing up wirelessly on a mobile phone.

Executable

A file whose contents are meant to be interpreted as a program by a computer.

FairPlay

A digital rights management (DRM) technology created by Apple Inc.

FAT

File Allocation Table. FAT is the primary file system for various operating systems. A TFAT is a Transaction Safe FAT.

FIR

Finite Impulse Response filter. A FIR filter is a type of a digital filter, an electronic filter that works by performing digital mathematical operations on an intermediate form of a signal (in contrast to older analog filters).

Flash memory

Non-volatile computer memory that can be electrically erased and reprogrammed.

FPGA

Field-Programmable Gate Array. An FPGA is a semiconductor containing programmable logic components (logic blocks) and programmable interconnects.

Gateway

A computer or a network that enables or controls access to another computer or network.

GDI

Graphics Device Interface. GDI is one of the three core sub-systems (along with the kernel and the Windows API for the user interface of Microsoft Windows). GDI is an interface for representing graphical objects and transmitting them to output devices.

GOEP

Generic Object Exchange Profile. The GOEP provides a basis for other data profiles, and is based on OBEX.

GPIO

General Purpose Input/Output. GPIO devices provide a set of IO ports that can be configured for either input or output.

GPS

Global Positioning System. GPS utilizes at least 24 satellites that transmit precise microwave signals, enabling a GPS receiver to determine its location, speed, direction, and time.

Head unit

A component of a stereo system, either in a vehicle or in a home cinema system, that provides a unified hardware interface for the various components of an electronic media system.

HFP

Hands-Free Profile. HFP is commonly used to enable auto hands-free kits to communicate with mobile phones in the car.

HMI

Human Machine Interface.

JTAG

Joint Test Action Group. JTAG refers to the IEEE 1149.1 standard for test access ports that are used for testing printed circuit boards using boundary scan.

Kernel

The central component of computer operating systems, which manages the system resources (communication between hardware and software components).

M3U

Moving Picture Experts Group Audio Layer 3 Uniform Resource Locator (also MP3 URL). M3U is a computer file format that stores multimedia playlists.

Middleware

Computer software that connects software components or applications. Middleware consists services that enable multiple processes running on one or more computers to interact across a network.

MOST

Media Oriented System Transport. MOST is a serial communication system for transmitting audio, video and control data through fiber-optic cables. This multifunctional, high-performance multimedia network technology based on synchronous data communication requires professional software tools and hardware interfaces.

MP3

MPEG-1 Audio Layer 3. MP3 is a digital audio encoding format used to create a file to store a single segment of audio so that it can be organized or easily transferred between computers and other.

MPEG

Moving Picture Experts Group. MPEG, is a working group of the International Organization for Standardization/International Electrotechnical Commission charged with the development of video and audio encoding standards. MPEG has standardized compression formats and ancillary standards.

MTP

Media Transfer Protocol. MTP is a set of custom extensions to the Picture Transfer Protocol (PTP) from Microsoft. MTP supports the transfer of music files on digital audio players and movie files on portable media players. MTP is closely related to Windows Media Player.

NAND flash memory

Forms the core of the removable USB interface storage devices (USB storage devices) and most memory card formats.

OBEX

OBject Exchange. OBEX is a communications protocol that facilitates the exchange of binary objects between devices. Many PDAs use OBEX to exchange business cards, data, even applications.

OPP

Object Push Profile. OPP defines the requirements for the protocols and procedures to be used by the applications involved in the pushing/pulling of data objects between *Bluetooth®* devices.

PBAP

Phone Book Access Profile. PBAP enables the exchange of Phone Book Objects between devices. It can be used between a car kit and a mobile phone to let the car kit display the name of the incoming caller.

PCM

A term for data encoded as Linear Pulse Code Modulation (LPCM). LPCM is a method of encoding audio information digitally or formats using this method of encoding.

PDA

Personal Digital Assistant. PDAs are handheld (or palmtop) computers. Newer PDAs also have both color screens and audio capabilities, enabling them to be used as mobile phones (smartphones), Web browsers, or portable media players.

PDK

Platform Development Kit. Refers to a collection of tools, documentation, and code that enables a platform developer (likely a supplier) to create, extend, and customize an image for the device. Platform developers have access to all low-level operating system and device-hardware-specific APIs, in addition to application-level public APIs. Platform developers can write new device drivers and modify an image in order to support new hardware.

PMP

Portable Multimedia Player. PMPs are consumer electronics devices that are capable of storing and playing digital media. The data is typically stored on a hard drive, microdrive, or flash memory. Mobile phones are also sometimes referred as PMPs because of their playback capabilities.

PWM module

Pulse Width Modulation module. The purpose of the PWM module is to enable time-critical waveform operations to be handled by the hardware instead of software.

SAPI

Speech Application Programming Interface. SAPI is an API developed by Microsoft to enable the use of speech recognition (converts spoken words to machine-readable input) and speech synthesis (the artificial production of human speech) within Windows applications.

SD card

Secure Digital (SD) card. An SD card is a flash (non-volatile) memory card format developed by Matsushita, SanDisk and Toshiba. SD cards are used in portable devices such as digital cameras, handheld computers, PDAs, mobile phones, and GPS units.

SDIO

Secure Digital Input Output. Devices that support SDIO (typically PDAs, laptops or mobile phones) can use small devices designed for the SD form factor, such as GPS receivers, Wi-Fi or Bluetooth adapters, modems, Ethernet adapters, or other mass storage media such as hard drives.

SDP

Service Discovery Profile. SDPs are network protocols that enable automatic detection of devices and services offered by the devices on a computer network. (For example, the *Bluetooth®* SDP is a profile used to find out which Bluetooth services are offered by the remote device.)

SMS

Short Message Service. SMS is a communications protocol that enables the interchange of short text messages between mobile telephone devices. SMS technology has facilitated the development and growth of text messaging.

SPP

Serial Port Profile. The SPP emulates a serial cable in order to provide an easily implemented wireless replacement for existing RS-232-based serial communications applications, such as familiar control signals. It provides the basis for other profiles such as DUN, FAX, HSP and AVRCP profiles.

SSI

Server Side Includes. SSI is a simple server-side scripting language used for the Web primarily for dynamically including the contents of one file into another file that is served by a Web server.

SyncML

Synchronization Markup Language. SyncML is a platform-independent information synchronization standard.

UART

Universal Asynchronous Receiver/Transmitter. UART is computer hardware component (an individual or a part of an integrated circuit) that translates data between parallel and serial forms. UARTs are now commonly included in microcontrollers.

USB

Universal Serial Bus. USB is a serial bus standard for interface devices, designed to both enable peripherals to be connected using a single standardized interface socket and to improve plug-and-play capabilities by letting devices be connected and disconnected without rebooting the computer (called hot swapping).

vCard

A file format standard for electronic business cards.

Wi-Fi

Wireless Fidelity. A wireless technology that promotes standards for the interoperability of wireless local area network products based on the Institute of Electrical and Electronics Engineers (IEEE) 802.11 standards. Common applications for Wi-Fi include Internet and VoIP phone access, gaming, and network connectivity for consumer electronics.

WAV

Waveform Audio Format. A Microsoft and IBM audio file format standard for storing an audio bitstream on a computer. It is a variation of the RIFF (Resource Interchange File Format, a generic meta-format for storing data in tagged chunks) bitstream format method for storing data in “chunks”, and is the main format used on Windows for raw and typically uncompressed audio. The default bitstream encoding is the Microsoft Pulse Code Modulation (LPCM) format.

Win32API

Windows API. WinAPI is the core set of application programming interfaces (APIs) available in the Windows operating systems—Win32 is the 32-bit API. The API consists of functions implemented in system DLLs (kernel32.dll, user32.dll, and gdi32.dll).

Windows Embedded CE

A Windows operating system developed for embedded systems. Windows CE has a distinctly different kernel, not a trimmed-down version of desktop Windows (it should not be confused with Windows XP Embedded, which is based on Microsoft® Windows NT®). Windows Embedded CE 6.0 is supported on Intel x86 (and compatible processors), MIPS, ARM, and Hitachi SuperH processors. Microsoft Auto 3.0 only ships with support for two ARM-based processors.

WMA

Windows Media Audio. WMA is an audio data compression technology developed by Microsoft as part of the Windows Media framework. WMA can refer to the audio file format or its audio codecs.

WPL

Windows Media Player Playlist. WPL is a computer file format that stores multimedia playlists.

Appendix 1: Hardware Reference Design

The following table provides details of the Microsoft Auto 3.0 hardware reference design.

The hardware reference design recommends using 256 MB NAND flash memory to store the operating system image, the applications, the speech engines, and additional application data. The image is partitioned during build time into an IMAGEFS (Image File System) region and one or more TFAT (Transaction Safe FAT) regions. The lower region of the NAND flash holds the IPL (Initial Program Loader), UPL (Update Loader), device parameter store and FPGA configuration modules. The IPL is only updateable via JTAG programming. The Secure JTAG feature of the CPU prevents unauthorized access to this region.

Function	Characteristics

Processor	• Freescale i.MX31, 16/32 bit RISC microprocessor.
	• ARM1136FJ-S core
	• 90 nm CMOS technology
	• Multi Layer 6*5 AHB Smart Speed Crossbar Switch allowing up to 5 bus transactions in parallel
	• Separate 16kB instruction and 16kB data cache
	• 128K second level cache
	• Smart Power Management
	• Enables simultaneous MPEG4 (SP) encoding and decoding
	• Support for real-time video decode
	    • MPEG4 SP (simple profile)
	    • H.264
	    • WMV
	    • RV
	    • MPEG2
	    • DivX
	• Video and image data pre/post-processing support in hardware
	• 400 (533) MHz CPU clock, SDRAM bus at 100 (133)MHz, DDR at 200 (266) MHz
	• 2-D/3-D graphics support (i.MX31 only)
	• Camera Sensor support
	• High-speed USB 2.0 interface:
	    • OTG – high speed
	    • Host1 – high speed
	    • Host2 – full speed
	• Flexible Audio Interconnect Module allows for programmed flexible connection of the various audio ports (I2S)
	• Security Features
	    • Memory Management Unit
	    • Security Controller incl. Secure RAM and Security Monitor
	    • Random Number Generator Accelerator
	    • Secure JTAG controller (with optional disabling)
	    • Universal Unique Identification
	    • Real-time Integrity Checker
	    • High Assurance Boot
	    • Tamper detection
	• Enhanced DMA
	• Timers
	    • Real-time clock

	• Enhanced Periodic Interrupt Timers • General Purpose Timer • Watchdog Timer • PWM Module • I2C • 2 SSI's • 3 CSPI's • 5-ch UART • FIR Module • GPIO
Memory	• 256 MB NAND Flash 8/16 bit • 64MByte DDRAM
CAN	• The CAN controllers can be connected to a vehicle CPU that in turn is connected to the main CPU by a serial connection
Ethernet	• 100Mbs Ethernet port for fast connection to development environment or other purposes
USB 2.0 OTG port	• High-speed (480 Mbs) USB 2.0 port available for both device and host connections (On The Go)
USB 2.0 Host port	• High-speed (480 Mbs) USB port available for external E-Call module connection and (with additional USB 2.0 hub) connection to future extension modules • Optional replaceable by a RS485 UART connection
USB Full speed port	• Full-speed (12 Mbs) USB port for connection to the *Bluetooth®* 2.0 EDR module
Bluetooth	• *Bluetooth®* 2.0 compatible • Supports Enhanced Data Rate (EDR) • Connection to processor through USB
Wireless 802.11 module	• Optional 802.11 module can be connected to CPU SDIO port

Appendix 2: Bluetooth Software Stack

The Microsoft *Bluetooth®* wireless technology stack implementation is a modular, general-purpose software stack; it is linked by default, or modularized, to meet the specific configurations that an automaker needs. The protocol stack makes up the core portion of the *Bluetooth®* wireless technology implementation. Through a *Bluetooth®* wireless technology connection, devices can exchange data and interact with one another via the applications. The HCI software module supports various connections (UART, USB, and PCMCIA) to the *Bluetooth®* wireless technology chip.

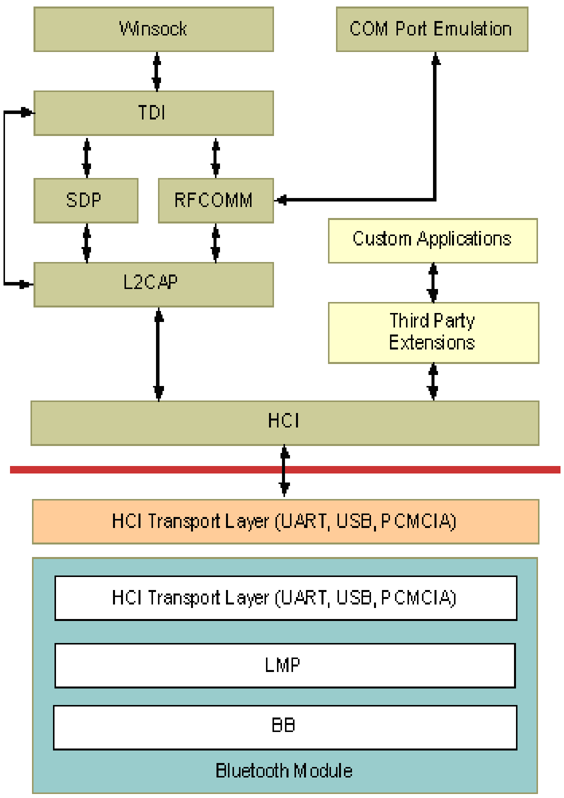

Figure 4 Bluetooth stack

The pairing service provides *Bluetooth®* SDP (Service Discovery Protocol), security, and pairing functionality to Microsoft Auto 3.0 applications. Paired data is stored in the registry. Events fired by this service serve as notifications of pairings to subscribed applications.

The following profiles/protocols are supported and used in Microsoft Auto 3.0:

- Generic Object Exchange 1.1
- Serial Port Profile 1.1
- Phonebook Access Profile-PCE 1.0
- A2DP-SNK 1.0
- AVRCP-Controller 1.0
- HFP-HF 1.5 (backward compatible to HFP 1.0)
- DUN-DT & GW 1.1

Related Links

The following Web pages provide additional information:

For more information about Microsoft Auto 3.0, see:
http://www.microsoft.com/windowsautomotive/ma/default.mspx

For more information about the Automotive Business Unit, see:
http://www.microsoft.com/windowsautomotive/default.mspx