Moblin for IVI - Software Architecture Overview

VERSION	**V0.8**
VERSION DATE	**16-MAY-08**

1 Introduction

The digital lifestyle has become a reality in terms of the commodities and technologies available to the consumer. In the home and office of today, a wide range of connected devices and services provide a seamless experience around information and entertainment access. The competitiveness and potential of this market has led to a mature ecosystem capable of churning out a new generation of affordable devices and services every six months.

At the same time, automotive manufacturers today face a tremendous challenge in trying to bridge the historically long development cycles of a vehicle, to the ever-changing multimedia-based demands of the consumer.

The Moblin In-Vehicle Infotainment (IVI) Software Architecture encourages the development of Open Infotainment Platforms (OIPs), based on interoperable, standards-based hardware and software solutions.

The issues of affordability and compatibility will mean a change within the current automotive supplier structure. It would mean a move away from the vertical “one-off” model in place today towards a horizontal PC-like supplier ecosystem. This would enable greater re-use of architectural elements across 1st tier suppliers and greater portability for automotive OEMs.

OIPs enable manufacturers to scale software across devices, leading to a decrease in total-cost-of-ownership and quicker product development times. Major requirements unique to in-vehicle-infotainment include:

- Power state management
- Fast boot
- File system integrity and robustness
- CAN-bus functionality
- MOST (in-vehicle) interconnectivity

The Moblin IVI software framework is based on the Linux operating system and related open source components that provide a software infrastructure to create an automotive platform.

1.1 About this document

This document is intended to provide an overview of the architecture of the Moblin IVI software, which can be used to develop an OIP.

The term "user" refers to a human interacting with the system through the HMI, and the term "client" refers to any functional component of the system that requires a service provided by another component of the system, typically anyone endeavoring to architect, develop and

finally deploy applications based on the Moblin IVI architecture for the in-vehicle-infotainment (IVI) market.

For example, a video playback application may be considered a client of the component that provides the video codec.

Upon reading this document, a user would gain insight into what makes up the Moblin IVI software architecture.

1.2OIP-Based Infotainment System

Any IVI system based on OIP can be organized into several physical configurations, with the software framework providing logical grouping of functionalities to adapt to a certain configuration chosen by the implementer of a car infotainment system.

The following are some examples of typical system configurations:

1.2.1All-in-one System

In this model, all hardware components for the car infotainment system (Radio, Automotive, Media, Storage, Navigation and Networking, etc) are in one device.

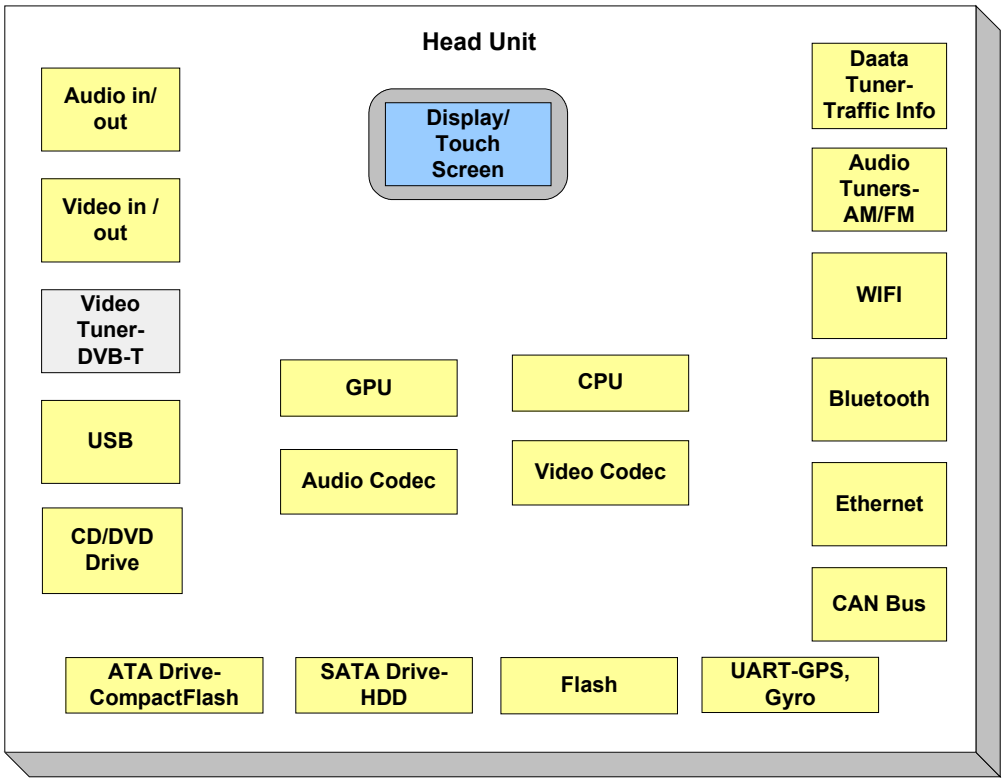

Figure 1 – System All-in-one Configuration

1.2.2Multiple Device Configurations

In this model, hardware components are distributed into multiple devices connected via a network mechanism (CAN or MOST based vehicle network, Ethernet, etc.). For CAN and MOST specifications refer to Appendix – 2.

- **Multiple Device Configurations: Example – 1:**

  Configuration: Head Unit (Hardware components: Media, Automotive, Networking, Navigation and Storage) – Radio Unit (Hardware components: Radio Tuners, Data tuner, CD/DVD drives)

  ▪Head Unit: The Head Unit is the central module that controls and maintains the devices and interfaces that constitute the vehicle infotainment/management system. It also presents the information and media to the user and collects user inputs. It also provides connectivity to external network through Wi-Fi/Ethernet and external devices such as mobile devices via Bluetooth and USB. A device bridge is provided to establish a MOST based network to other in-vehicle devices for extending the feature set to radio tuning, multiple CD/DVD changers, etc.

  ▪Radio Unit: The Radio Unit has the primary responsibility to tune to audio sources such as AM/FM transmissions and data sources such as traffic information. The radio unit also includes CD/DVD drives. The radio unit allows connections to the head unit via a network bridge.

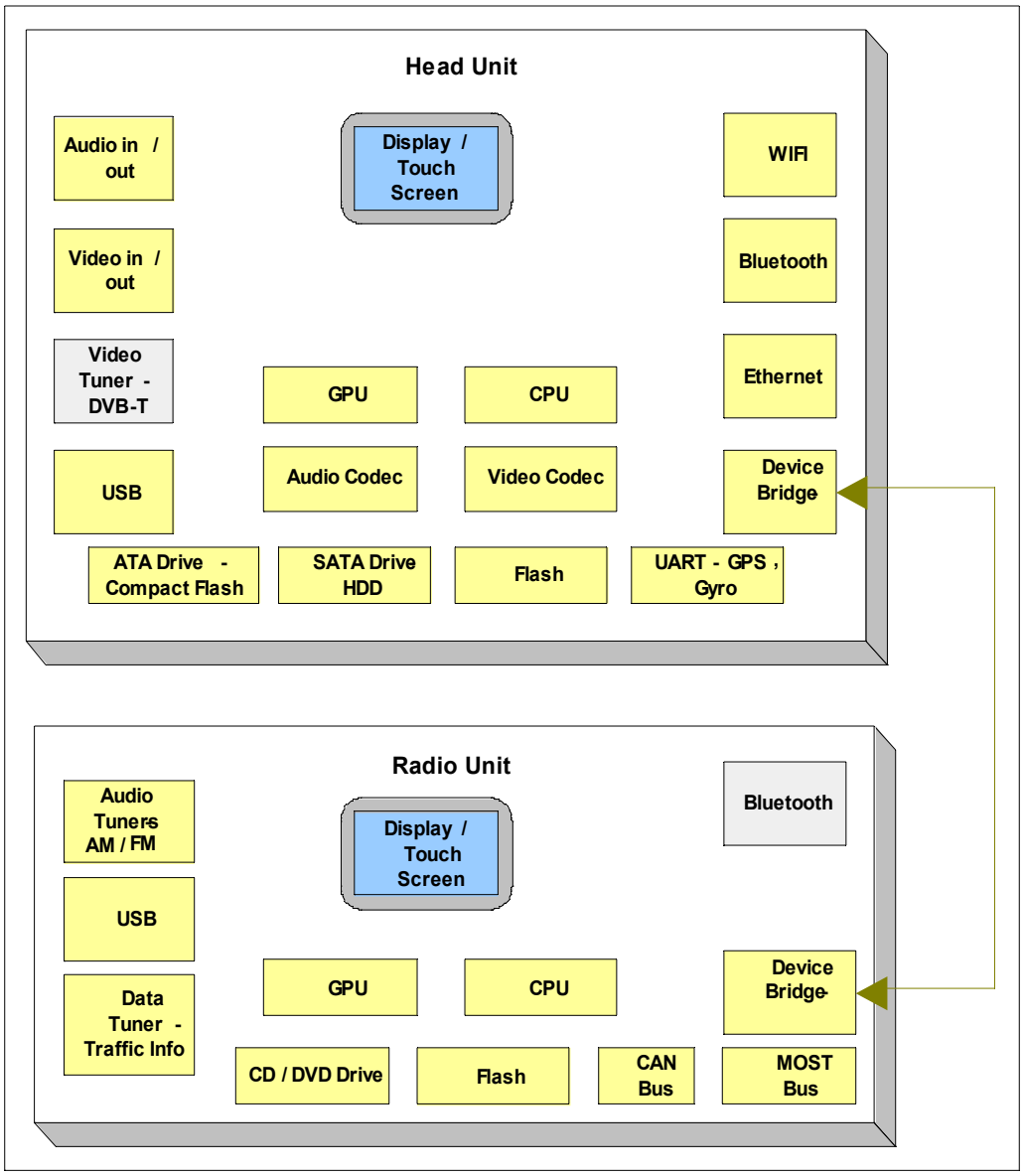

Figure 2 – System Multiple Device Configurations: Example –1

1.3 Software Framework

The Software framework defines the software interfaces and components that run on the Head Unit. This Software framework allows the Head Unit to run interactive user applications and provides a mechanism to control and process data originating from the Radio Unit. The framework does not define applications or the Human Machine Interface (HMI).

The software framework has a layered architecture, which supports vertical functional groupings and horizontal/vertical component bindings. The following layers are defined and sub-systems are identified in Figure 3.

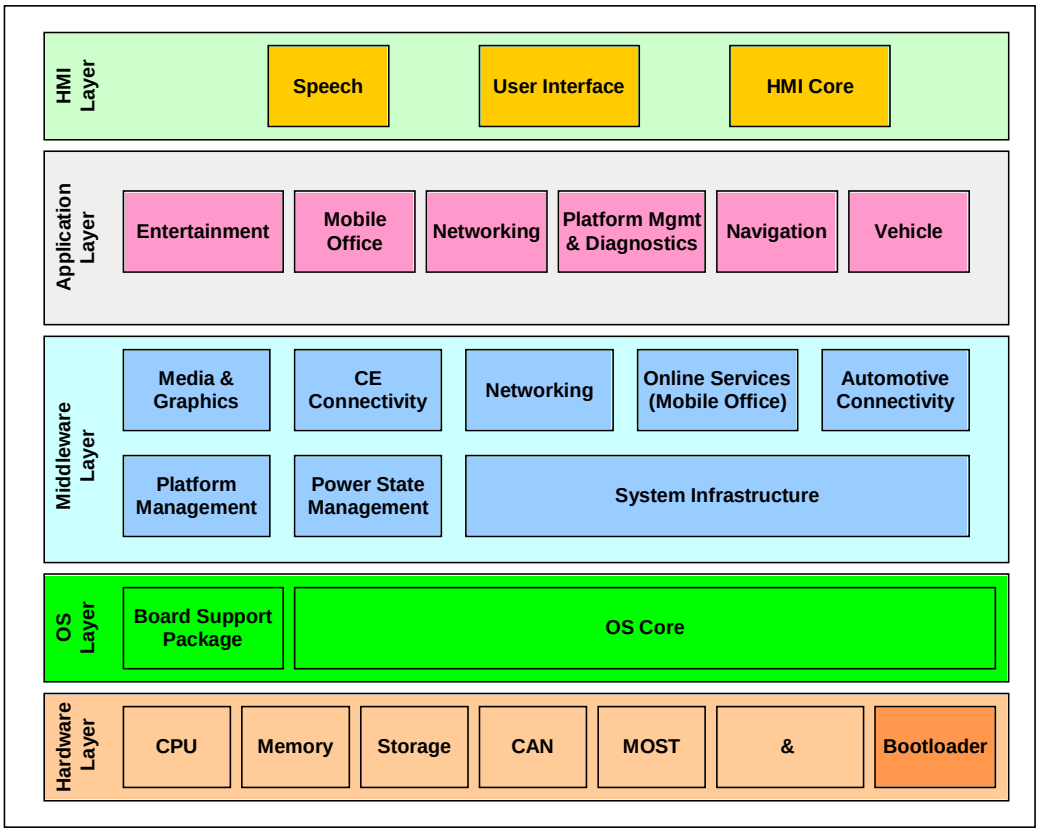

Figure 3 – Software Framework Overview

- **Hardware Layer:**

  The Hardware Layer lists the basic core platform components of the platform required for an infotainment system. It also includes any software components necessary to boot the OS Layer, including firmware, bootloader, etc.

- **Operating System (OS) Layer:**

  The OS Layer includes the Linux operating system and all the packages configured with the kernel and root file system build. This layer also includes drivers and board specific software that are integrated to the board support package (BSP).

- **Middleware Layer:**

  The Middleware provides a rich set of components and interfaces to realize all functional areas of the Application layer. These middleware components fall into the following categories:

  - ***Open source packages:***

    Provides infrastructure elements such as networking, codecs, graphics libraries, etc., using open source packages.

  - ***Moblin IVI Specific components:***

    These components are developed to address the lack of open-source support for certain functionality or to aggregate and control multiple lower level modules.

  - ***Third party Stacks Integrated:***

    The third party stacks implement specific protocol stacks like Bluetooth, Speech, MOST, etc, where tested and proven third party solutions are available.

- **Application Layer:**

  The Application layer includes applications and tools that provide car infotainment and vehicle management functionalities. The application and Tier-1 developers, keeping in focus the specific car infotainment system they are building, develop these applications, by utilizing the underlying interfaces provided by the Moblin IVI software framework. The Moblin IVI software architecture identifies and functionally groups known applications in the car infotainment domain, so that appropriate components and interfaces needed to realize the application layer can be designed/integrated into the middleware and OS layers.

- **HMI Layer:**

  The Human Machine Interface (HMI) is the central interface to the user of the system. The HMI has control of the display of the HMI Head Unit and has the responsibility to process and react to all user inputs coming into the system, such as audio in, touch screen input etc.

  Any state information is located in the underlying applications or the middleware. Any state change that is relevant for the HMI is reported to the HMI by inter-process

communication mechanism. Conceptually there is a central event queue in the HMI where all such notifications arrive and cause changes in the graphical elements presented to the user on the screen or trigger speech output. The HMI typically performs graphical rendering activities. An exception to this rule may be the Navigation application, which does the map rendering on its own. The HMI defines the view port for this map rendering.

2System Infrastructure

2.1Hardware Layer

2.1.1Description

The hardware layer lists the basic core platform components of the platform required for an infotainment system.

2.1.2Components

The hardware layer of a typical OIP is shown in Figure 4.

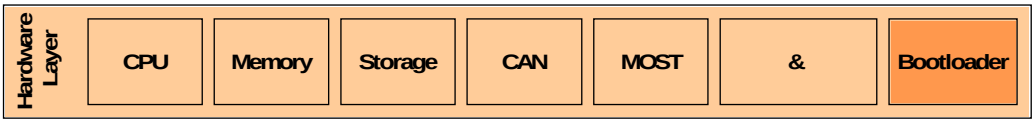

Figure 4 – Hardware Layer

2.1.3Hardware

The following hardware components are required as part of an infotainment platform.

Hardware Function	Description
Processor	CPU supporting frequencies for sufficient integer and floating point performance Provides sufficient entries in instruction and data TLB
Memory Controller	Support low cost memory
Video Decoder	MPEG2, MPEG4, VC1, WMV9, H.264 (main and high profile level 4.1), DiVX
Graphics Engine	GFX performance: Fill rate of at least 400 and 3DMark'05 score of 120
HD Audio	Audio Support is required; at least two channels
Display	800x600 15 bpp
I/O	Minimum USB 2.0
Other I/O	MOST, CAN, SPI, Bluetooth, UART, SDIO, Ethernet

2.1.3.1Bootloader

The bootloader is responsible for starting the operating system. It consists of two components the bootloader itself and a configuration mechanism.

The following include the components that makeup the bootloader:

Component	Functional Description	Realization
Boot Mode Storage Device	Provides an interface to set and get the boot mode, which is necessary to enter the recovery mode.	New: Boot Mode Storage Device
Bootloader	Required to start the operating system according to the settings of the Boot Mode Storage Device.	New: bootloader based on grub

2.2OS Layer

2.2.1Description

The OS Layer consists of a standard embedded Linux operating system, including both the Linux file system. This provides the foundation for the middleware components.

Figure 5 – OS Layer

2.2.1.1Board Support Package (BSP)

Specifies the hardware specific Linux kernel configuration as well as provides hardware specific device drivers. In addition, boot loaders may be provided as part of the BSP.

2.2.1.2OS Core

Specifies Linux OS Kernel and Driver facilities required for the Moblin IVI software:

- **Standard Linux kernel configuration:** A default configuration that specifies the items required to be enabled in the kernel by the application environment.

▪**Standard C library and Unix-like environment:** GNU C Library is provided along with a number of standard Linux applications. This environment creates a basic embedded Linux based operating system.

▪**Crypto framework:** The crypto support is needed to allow applications and middleware to implement authentication schemes. OpenSSL is integrated to provide the crypto support.

▪**IPC:** Along with standard Linux IPC mechanisms, d-bus will provide additional messaging and communication infrastructure for applications and processes.

▪**Networking Support:** Standard networking infrastructure and protocols are supported.

▪**File Systems:** Many standard file systems like JFFS2, Ext2, and Ext3 are available.

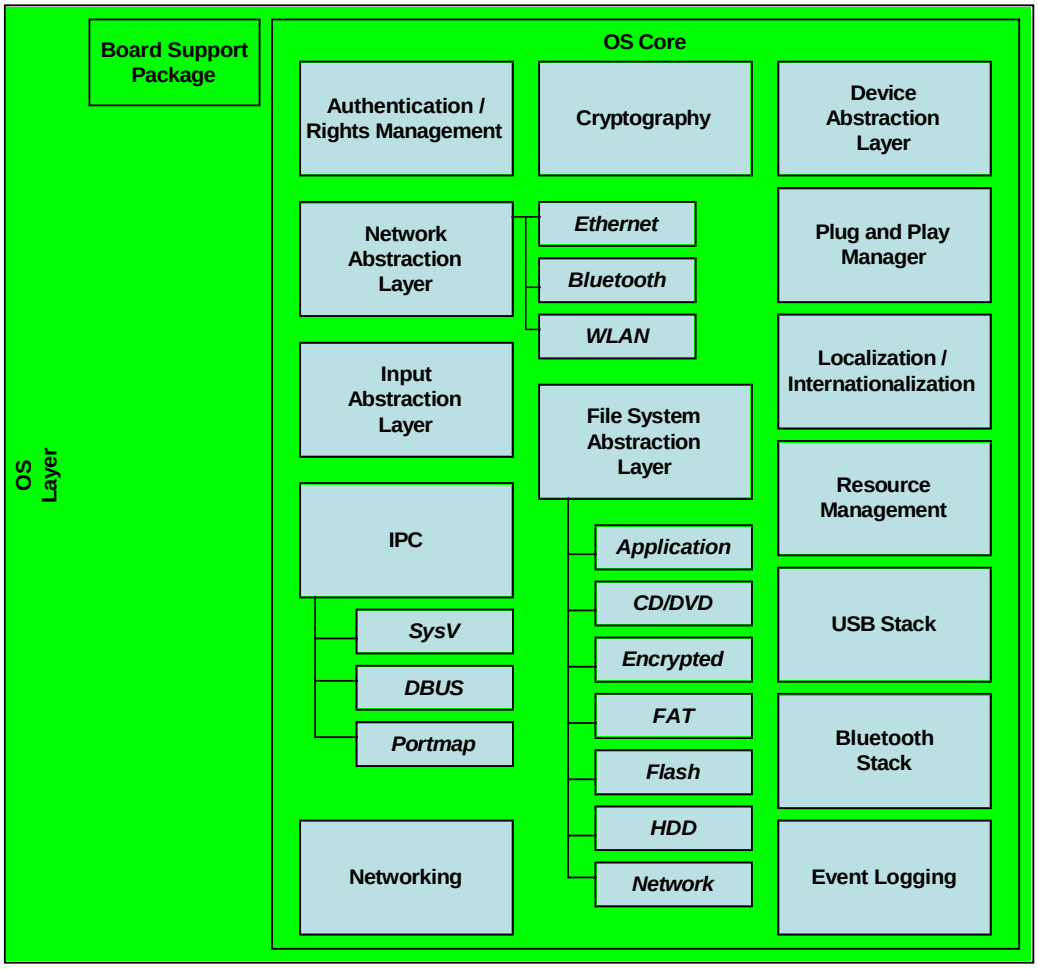

Figure 6 – OS Core Overview