IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661,p-ISSN: 2278-8727, Volume 18, Issue 3, Ver. III (May-Jun. 2016), PP 01-11
www.iosrjournals.org

IMPLEMENTATION OF WINDOW LISTING OPTIMIZATION AND KOREAN SMART SPELLER FOR IN-VEHICLE INFOTAINMENT

Mrs. Sarah Iram L. Indikar[1], Dr. P.A. Vijaya[2], Mr. Shivanand Kumbar[3]
[1, 2]*(Dept. of ECE, BNMIT, Visvesvaraya Technological University, India)*
[3]*(Dept. of HMI Infotainment, Harman International India Pvt. Ltd, India)*

Abstract:*The project deals with the implementation of two important features in HMI (Human Machine Interface) for MAN/SCANIA infotainment. The two feature implementations are Window Listing Optimization and Korean Smart Speller. A technique called smart speller is used while entering navigation address details, where the system speller/ keyboard displays only the next available character instead of the entire alphabet set. The available characters are displayed based on a database that is stored in the middleware of navigation. The other characters that are not available are not highlighted on the speller view;thereby it eases the effort of the user in entering characters. Thesecond feature that is implemented for the infotainment head unit is window listing optimization. It optimizes the overall performance of media application browse and the entire system. It reduces the number of calls in the D-Bus and hence the delay is reduced. It also reduces the system congestion thereby increasing the efficiency and performance of the system. As part of window listing, the songs are downloaded as a page with fixed items as and when user browses through these items. Until the user stops browsing for a particular item, none of the items on the list are downloaded. The list of items is updated only when user stops browsing for items.*

Keywords: *HMI, D-Bus, HFP, A2DP,VNC*

I. Introduction

Automotive infotainment is the fastest growing technologies in the infotainment industry. Consumers demand for PC-like quick responsiveness, human-machine interface and excellent efficiency from every device on the go. Meeting these growing demands for the best possible consumer experience, within time to market deadlines, is increasingly a part of the infotainment domain.

Infotainment systems are hardware devices used in vehicles to provide navigation services, connectivity (connected car) and audio / visual entertainment. Most vehicles these days have infotainment units that provide entertainment, connectivity with devices such as personal navigation systems and smart phones with a hands-free (Bluetooth connectivity) car kit. Smart phone connectivity links the product life cycle gap between car infotainment systems and smart cell phones. It employs the advances made in a smart phone technology in these infotainment systems. IVI (In-Vehicle Infotainment) features include, device integration for HFP (Hands-free Profile) and A2DP (Advanced Audio Distribution Profile), head-up display, driver assistance information, interior personalization and Cloud based infotainment (server based).

Many automotive component suppliers are engaged in integrating the already existing navigation system in car with the car audio and smart phone, using a hands-free kit or Bluetooth. Cluster dashboards integrated with head-up display, infotainment systems, steering wheel controls etc. along with smart phone connectivity is the future trend. Standardizations like Mirror Link, which uses Virtual Network Computing (VNC), and have added value to handset integration that work on android platforms. Not only are Infotainment systems used in entertainment, but they are also used in road assistance for driver safety, with additional features such as video & data recording from rear view & night vision cameras and also black box recording feature. The addition of an augmented head-up display ensures the focus of the driver on the screen and also provides more information with distance. The market opportunity for IVI systems that create an exclusive in-car user experience is growing in large scale. The interweaving of the car's IVI system with mobile device and other applications can follow through that experience of connected car. The foremost mobile applications used for music streaming such as Spotify and iTunes, have media player which use different approaches to get user experience data. This confluence enables the procreation of innovative applications that interact with the ECU (Electronic Control Unit) for remote monitoring, control in-car multimedia experiences, internet connectivity, navigation and safety [1].

IVI Intelligence can be used in various aspects and with different implementation methods. One of the approaches is, with accelerated growth in smart phone and Cloud computing technologies, consumers are gabbling for music live streaming, Internet radio and smart phones. Advanced infotainment features integrate user behavior and the next generation of data based infotainment systems.

They can:

- Manually personalizing the media playlist or automate the choice of entertainment based on user behavior data. User behavior data is a derivative of analysis of inputs from VR (Voice Recognition), based on the mood of the vehicle owner.
- Procure entertainment for IVI from Cloud based resources instead of local media databases like DVD players or Bluetooth streaming audios. This feature requires cloud based efficient servers for interaction with user.
- Share and store media files like photos, music and videos including Cloud based media databases like Twonky, Plex and air video. Sharing of such media files will result in saving local space and having more media choices between various groups. Software such as Universal plug and play or digital living network alliance helps user to enjoy personal and online music, photos and videos. It is also used to share favorite media with PCs, TVs, stereos and other devices which connected to your network.
- Vehicle interior personalization can be incorporated in the form of ambient & mood lighting and cluster rearrangement to include user’s choice of music. This gives a possibility to alter the car’s dashboard and look of other interior components.
- Integration of technology by extending the HMI (Human-Machine Interface) of smart phones with navigation units has been achieved. It is done by using VCN, hands-free or Bluetooth interfaces. An additional feature that can be included is the registering of vehicle health and status reports on an infotainment device. It is a possible scenario because an intelligent IVI acts like a standalone ECU and can easily be interlaced by a gateway to communicate to other ECU's. An application which is dependent on the infotainment unit can utilize standard OBD (On Board Diagnostic) tests to provide vital health information [1].

The system architecture should be built such that it is feasible for intelligent profiling. Such intelligent systems can for e.g. be used to detect kids in a vehicle through an interactive voice recognition system or an image processing system, using the human detection algorithm. It can also check if the driver is alone by interactive voice recognition or image processing methods and then choose a pre-set music or make a mood-based profile to help them relax and enjoy. Profiling can be achieved in two ways i.e. manually or semi-automatically. Semi-automatic profiling uses many inputs such as voice recognition for identifying persons, a human detection algorithm to detect user images and even noise level detection can be done. More intelligent features can be added by getting Cloud based songs, depending on the user situation. Broadcasting a user requested song through interactive radio is a challenge which needs support from Cloud or a FM radio service provider [1].

HARMAN is one such company which is a leading global provider of premium audio and infotainment solutions. The company has acclaimed brands and savors a legacy of leading-edge innovation and premium quality in infotainment system solutions. They operate worldwide through a highly interactive network of talented employees and channel partners, sited to serve both the emerging and established markets [2].

1.1Objective of the project

HARMAN is currently developing a MAN/SCANIA infotainment unit for heavy vehicles. The MAN Group is one of Europe's leading commercial vehicle, engine and mechanical engineering companies and SCANIA is one of the world’s leading manufacturers of trucks and buses for heavy transport applications, industrial and marine engines [2].

There are mainly two types of infotainment screens that are being developed:

- J5 i.e. basic infotainment system with features like media (audio), radio and phone connectivity. The J5 screen is 5 inches wide.
- J6 i.e. advanced infotainment system with basic features and additional features like media (video playback), indigenous navigation system, front camera and park assist camera support and extended services like car play/mirror link connectivity, internet browser and remote control application. The J6 screen is 7 inches wide.

Their variants are:

- MAN J5/J6 for the company MAN.
- SCANIA J5/J6 for the company SCANIA.

The aim of this project is to develop the below mentioned HMI features for the MAN/SCANIA Infotainment.

1.1.1 Korean Smart Speller

To improvise the user experience while entering navigation address details, a technique called smart speller is used where the system speller/ keyboard is made smart enough to display only the available characters making it easy for the user to enter desired text. This feature of Korean smart speller is implemented in line with the generic smart speller as the base. As per the Korean speller implementation, single byte character inputs are taken and composed into a multi- byte Korean word to get the desired navigational details in Korean language for e.g. country name , state name , street name, postal code etc.

1.1.2 Window Listing Optimization Technique

In Media application the user is allowed to browse through a list of songs. Here a maximum of 25000 songs are supported by the system. When the user is trying to browse through the list of all songs, downloading the complete list of 25000 songs becomes really slow and puts heavy load on the D-Bus in turn resulting in sluggish behavior of the overall system. To optimize the overall performance of media browse, a technique called window listing is implemented. As part of window listing the songs are downloaded as a page with fixed items as and when user browses through these items.

II. Architectural Design

The HMI of any infotainment system consists of:

1) Input/output devices
2) HMI layer
3) Application layer

There are a number of interfaces added to connect the different layers. Below these layers lies the operating system. The Fig 1 depicts a simple HMI system [3].Currently, HARMAN International is dealing with the IVI project for MAN and SCANIA companies. HARMAN is the parent company behind an array of legendary brands that includes Harman Kardon®, JBL®, Mark Levinson® and Infinity®. It is a leading global provider of premium audio and infotainment solutions [2].

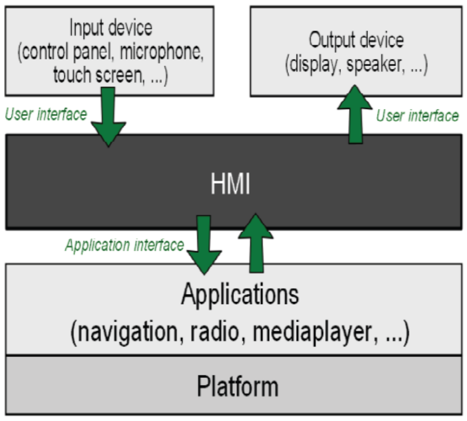
Input device (control panel, microphone, touch screen, ...)
Output device (display, speaker, ...)
User interface
User interface
HMI
Application interface
Applications (navigation, radio, mediaplayer, ...)
Platform

Fig1. Simplified HMI System

HMI mainly consists of UI screens for taking input and displaying output data, a logical layer for storing and processing the data, abstract service layer having abstract class for defining inter HMI interfaces and SVC-IPC layer which communicates with under lying Application services using D-BUS. The Fig 2 shows the proposed architecture and the different layers present inside.

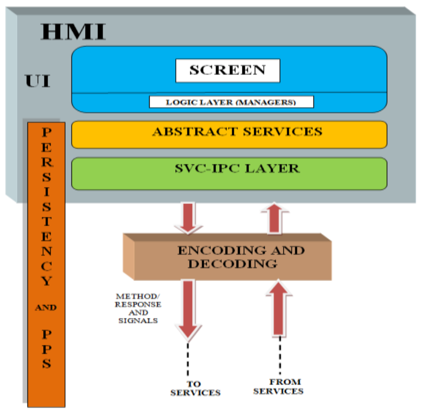

Fig2. Proposed architecture for the Infotainment system

Screens/UI

The UI screens are made of Buttons, text fields and image fields widgets, each of which are arranged based on designs provided by customer. The project uses touch screen for user interaction. The following steps explain the events that take place on a touch.

When user presses any button on the screen, the TOUCH EVENT is processed in these screen files and given for further processing to Logical layer. When the system receives some request from underlying service the same is handled and processed by Logical Layer and then processed data is passed to screen to be displayed.

Logical Layer

Each HMI applications (e.g. Phone, Media, Tuner and Navigation) will be having multiple screens designs and one manager class to manage these screens, which is also called Logical layer.

The manager will be a singleton class running throughout the program execution.

Few of the main roles played by managers are:
1) Stores and manages all application data throughout program execution.
2) Manages application screens.
3) Acts as communication mechanism between screens and SVC-IPC layer.
4) Can communicate with other managers inside HMI for inter application data handshake.

Abstract Service

Abstract Service mainly works as Abstract class for all API calls declarations to which has to be made to underlying application layers. This layer is also used for making API calls from Logic Layer to SVC-IPC layer. Here Logical layer calls a function of Abstract service layer which interns calls the virtual function to SVC-IPC for underlying API call.

SVC-IPC

This layer mainly works as communicator between HMI layer and underlying application services. On any button pressed in screens the final call comes till this layer, then the API data is filled in appropriate format and D-BUS message is sent to respective application service. Now when the response for the above API call is arrived, it is registered and received from D-BUS in this layer data is abstracted and passed to Logical layer for storing and displaying on screens.

Hence the complete HMI communication process can be explained as below.
1) On press of some widget on screens CONTROL EVENT is raised and received in UI Layer.
2) After extracting that button related data, a function call is made to Logic layer with these data.
3) Logic Layer again makes a function call to abstract service layer with the same data, here in Abstract service layer an EVENT is generated and caught back in same layer. This is done to release control thread.
4) After catching internal event back, Abstract service calls virtual function in SVCIPC service. Here the Data is arranged in proper format and API call is made to underlying service using D-BUS.
5) Once the response comes back for the API call it is received in SVCIPC layer. The data is extracted from DBUS call and filled in proper structure.
6) Then an event SERVICE EVENT is generated with these data.
7) Logical layer catches this event. The data is processed and stored in logical layer and LOGIC EVENT is raised with the processed data.
8) Currently visible screen receives this event and displayed the data appropriately on the buttons for it to be visible to user.
9) If the same data has to be used by some other application then the LOGIC EVENT is also registered and received by manager for that application and processed.

III. Tools Used And Workflow

The below mentioned tools are mainly used for software and project development. This section gives an overview these tools and the workflow adopted for the implementation of various features based on the requirement.

3.1 Software development and tracking tools

The tools mentioned below are used for software development and tracking.
1) *Doors*: Doors is used for defining and tracking CRS (Customer Requirement Set) and FRS (Functional requirements Set).
2) *JIRA*: These requirements are broken down in to features logged in to JIRA for development cycle steps tracking. These steps include assigning to developer, analyzing, development, review, integration, verification, closing.

3) *ELVIS*: Once the delivered features are tested the defects are created and logged in ELVIS for defect fix steps tracking. These steps include categorizing, Processing, Integration, verification and reproduction.
4) *PERFORCE*: PERFORCE is a version control tool. Complete Development Source code and Resources are stored in a server and to enable multiple developers from various locations to check out this source code and check in their changes this version control tool is required.

Another tool called Open HMI Studio is used for project development related to Graphics. It also has an option for regression testing. Some of the other key features of this tool are 2D Screen authoring/3D Scene authoring, animation authoring/Language authoring/Menu Tree authoring/Flow Chart authoring etc. It is also used for Source Code Generation, and Compilation [4].

3.2 Language and Operating System

The project is developed using C++ language. QNX is the OS used. It is a commercial Unix-like real-time operating system, aimed primarily at the embedded systems market.

3.3 Workflow for feature implementation and development

1) Discussion with customers for the requirement and feature clarification. Once the requirements are clear a CRS (Customer Request Set) is built .this CRS gives an upper view of how customer wants features to work.
2) With the CRS the next step is internal technical lead discussions. With the take away from the technical discussions, a TRS (Technical Request Set) is developed.
3) The TRS then is broken down into small feature segments for the work to be divided to the developers.
4) Each feature segment is logged into a software called JIRA and assigned to developers.
5) Developer completes development within a planned date and sends the feature for integration to the integration team which compiles and integrates all the feature segments.
6) After the integration is complete the testing team tests feature for its correct implementation and functionality.
7) If any defect or bug is found it is logged into a software called ELVIS and a ticket is created for the developer of that particular feature to fix the defect/bugs. The defect is then fixed and sent back for verification for the clearance. Once verified the feature is approved as defect free else a ticket is raised again in the same way as mentioned above. This process is to ensure that the implementation is totally defect-free.

IV. Design Implementation And Results

This work has been carried out at Harman International India Pvt. Ltd. It has been successfully designed, implemented and verified for different test cases. The implemented features are as below.

4.1 Implementation of Korean Smart Speller

The working model of a generic smart speller is shown in Fig 3. As shown in the diagram the smart speller is directly associated with the navigation HMI and belong to the same HMI layer. The speller and navigation logic layer communicate with each other by sending events and requests and receiving the confirmation to the request.

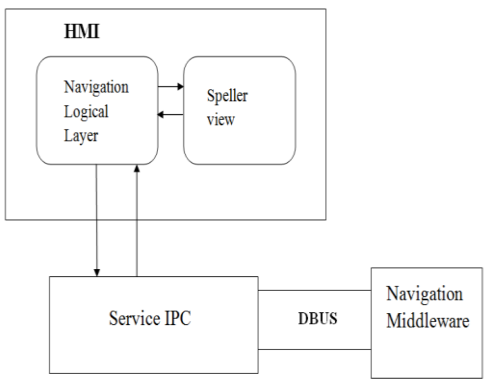

Fig3. Block Diagram for Smart speller

The sequence diagram for a generic smart speller depicting the various events and requests that are made between the navigation layer and the Korean speller view are show in Fig 4.

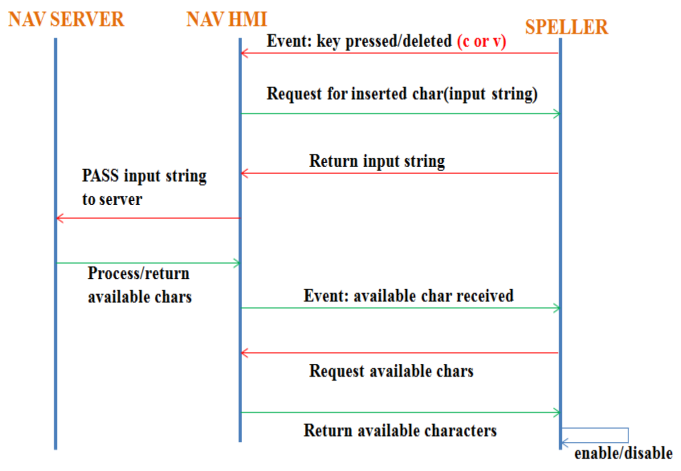
NAV SERVER
NAV HMI
SPELLER
Event: key pressed/deleted (c or v)
Request for inserted char(input string)
Return input string
PASS input string to server
Process/return available chars
Event: available char received
Request available chars
Return available characters
enable/disable

Fig4. Sequence diagram for Korean Smart Speller

4.2 Implementation of Window Listing Optimization Technique

While designing an embedded system and working with Real time Operating system two of the most crucial factors that decide the Performance, Stability, Efficiency and the Life Cycle of the system are Time and Memory. Care needs to be taken in management of the Time and Memory Factors while designing, implementing and optimizing the system. Since an embedded system is known for its real time operation, any

The basic idea behind window listing technique is to download only those songs which user tries to browse, at the same time making the entire list available to user. When user selects a category in the media browse, the count of songs belonging to that category is requested from media middleware first. Next, on receiving this count an empty list with the received count number is created and first page of items are downloaded and made available to the user.

Now, for further browsing user can perform various operations such as:

a. Go to next /previous page
b. Scroll through the list
c. Drag and drop the list
d. Dragging the scroll bar slider

The block diagram for window listing optimization is shown in Fig 5.

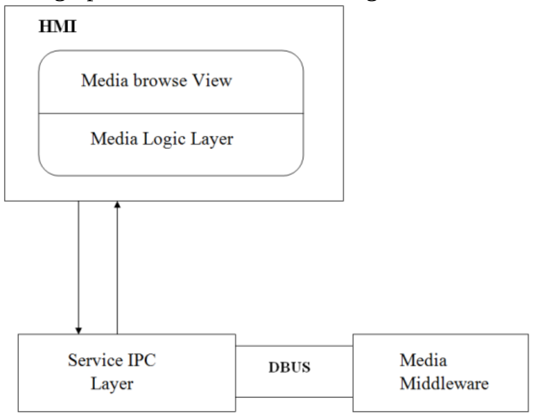

Fig5. Block Diagram for Window Listing Optimization Technique

The Fig 6 next shows the functional flow for the optimized media browse. When compared to the normal browsing implementation in the window listing technique, here once the user enters browse view and selects any category, the “selectItem” request is sent to service layer and the response of “selectItem” is received. Here the count is also received based on the number of items present in the list for that category.

When the count is zero i.e. no items in that category then the process ends. But when count is non-zero we create an empty list and request for the first page item using “getItems”. Both these operations are performed parallel to each other thereby savings that much amount of time. Once requested data is received with the response of “getItems”, the items are updated in the already created list.

The user can drag or scroll to traverse in the list. When the user does this, it is doubtful as to which item the user will stop at after the drag or scroll. Since we are creating a list of 5 items, it is necessary to know which the current top item on the list is. Only when this item is known the next four items can be updated correctly. Hence care is taken to find the current top item accordingly.

If the user browses further in the list, the same process of requesting a single page data and updating in the list is repeated. As a result the delay due to large data fetch and the list creation is not observed anymore. The user experience is also efficiently improved. Congestion caused earlier due to requesting of complete list of items at once through the "getItems" from a particular category is not present in this implementation.

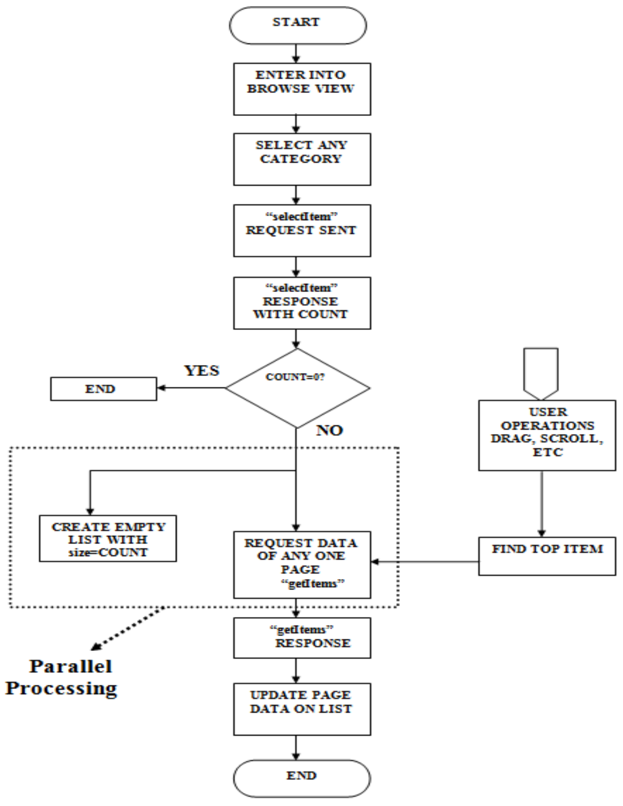
START
ENTER INTO BROWSE VIEW
SELECT ANY CATEGORY
"selectItem" REQUEST SENT
"selectItem" RESPONSE WITH COUNT
COUNT=0?
YES
END
NO
USER OPERATIONS DRAG, SCROLL, ETC
CREATE EMPTY LIST WITH size=COUNT
REQUEST DATA OF ANY ONE PAGE "getItems"
FIND TOP ITEM
"getItems" RESPONSE
Parallel Processing
UPDATE PAGE DATA ON LIST
END

Fig6. Flow chart for Optimized Media Browse

The sequence diagram below explains the various events and requests made in the form of a sequence flow along with the algorithm to achieve the window listing optimization. The sequence diagram for the window listing optimization is shown below in Fig 7.

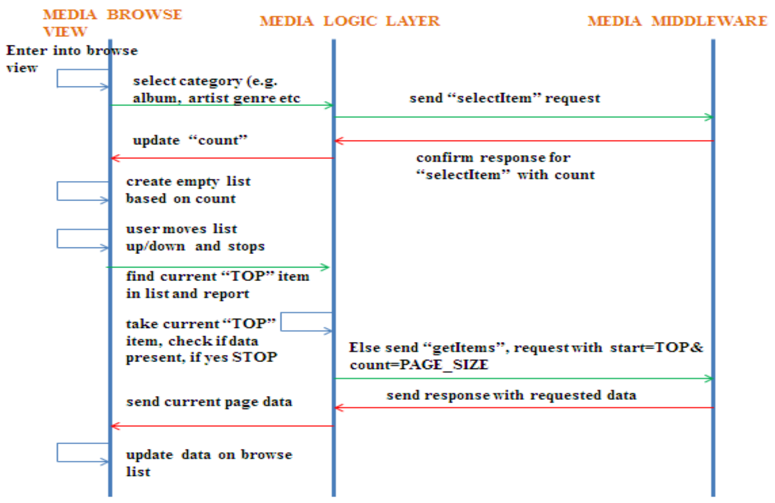
MEDIA BROWSE VIEW
MEDIA LOGIC LAYER
MEDIA MIDDLEWARE
Enter into browse view
select category (e.g. album, artist genre etc
send "selectItem" request
update "count"
confirm response for "selectItem" with count
create empty list based on count
user moves list up/down and stops
find current "TOP" item in list and report
take current "TOP" item, check if data present, if yes STOP
Else send "getItems", request with start=TOP& count=PAGE_SIZE
send current page data
send response with requested data
update data on browse list

Fig7. Sequence Diagram for Window listing optimization

4.3 Results

This section provides the results for the above feature implementations. All these simulations have been carried out in Microsoft Visual C++ and the graphics are integrated using Open HMI Studio. The manual testing is carried out on the head unit.

4.3.1 Results for Korean Smart Speller

The Europe map is used as a database for the Korean smart speller. The view below in Fig 8 is of the navigation menu in Korean language. The language has been changed to Korean in setup settings menu

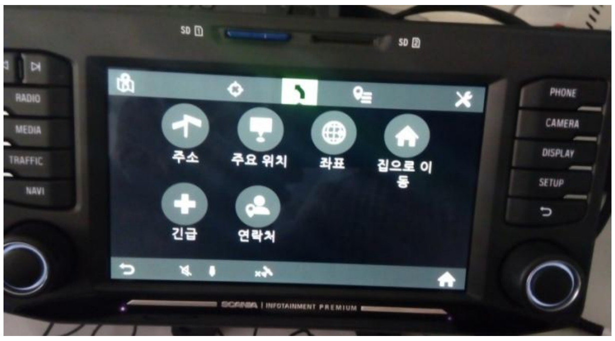

Fig8. Korean Language Navigation View

The Fig 9 below shows the submenu that appears asking for details to be entered.

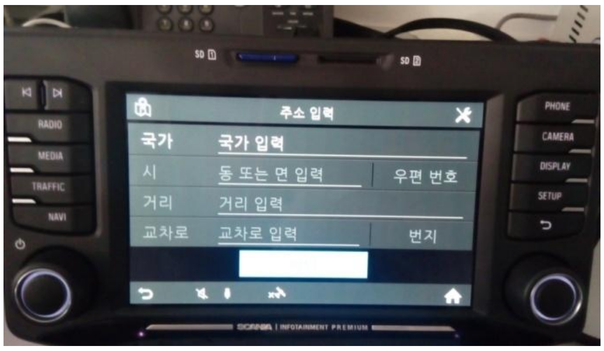

Fig9. Address Submenu for Korean Language

The Hangul keyboard view appears as shown next in Fig 10.

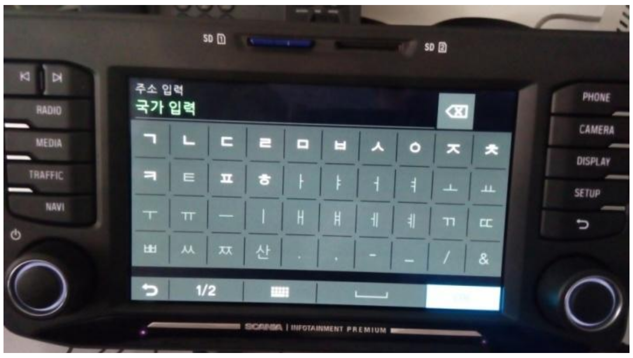

Fig10. Hangul Keyboard View

The country that is entered here will be Germany for which the first character in Korean language is shown below in Fig 11 and next available characters are shown.

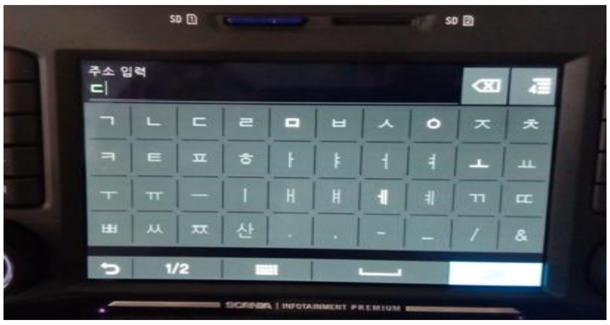

Fig11. First character in Korean language entered

The next character from the available set of characters is entered as shown in the Fig 12.

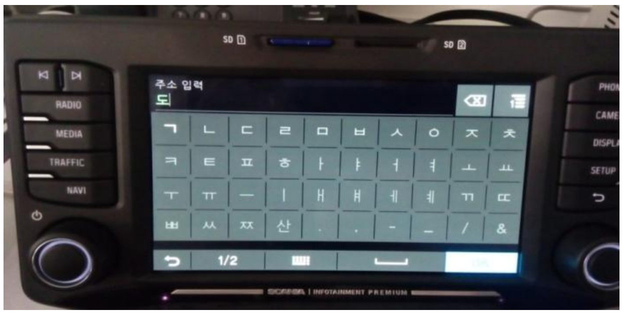

Fig12. Next available character is entered

Further the next available character is displayed as shown in Fig 13.

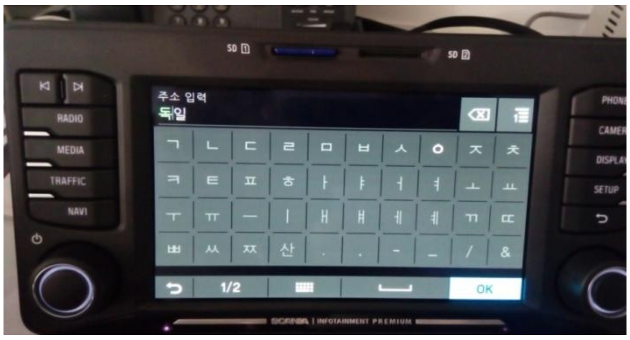

Fig13. Next available character

4.3.2Results for Window Listing Optimization Technique

The results for the window listing optimization technique are shown in this section. It includes the implementation on the head unit, timing optimization and the memory optimization. Here the user can hold and drag any item on the list in upward or downward directions. As seen in the Fig 14 the data is not downloaded in the next page and loading is displayed. Once the user drops the list the data for this page will be downloaded.

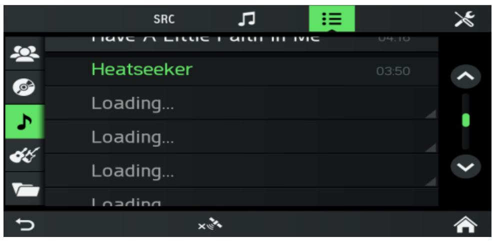

Fig14. List hold and drag

User can quickly go up to any position in the list by directly holding and dragging the scrollbar marker as seen in the Fig 15. Here while the user is still scrolling through the list the data will not be downloaded and loading will be seen

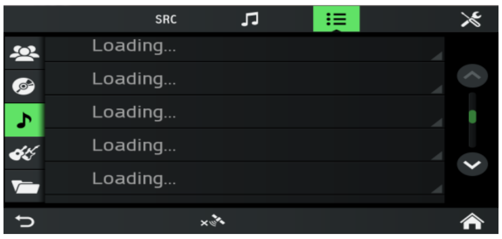

Fig15. Scrollbar marker drag

To scroll quickly through the list user can also long press on the up or down arrow as seen in the Fig 16. When presses and hold on the button the list will quickly scroll and loading will be displayed and at this point the data is still not downloaded. Once the user releases the button the data will be downloaded for that page.

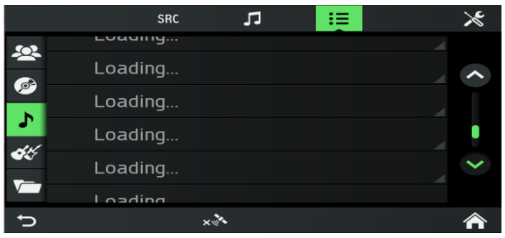

Fig16. Scrollbar button long press

For page by page browsing of data user can just click on up or down arrow as shown in Fig 17 below.

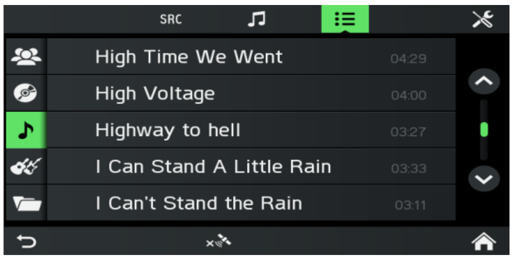

Fig17. Next/Previous page browse

4.4.1 Timing Optimization

In general implementation without optimization, usually the time taken to request the data and update it on the list is found to be increasing with the increase in number of items. But after optimization the time taken will always be constant and minimal as we download only a single page of item each time the request is made by the user.

The Table 1 shows the time taken for different number of items (songs) before and after optimization.

NO. OF ITEMS	TIME TAKEN (Before Optimization hh/mm/ss/ms)	TIME TAKEN (After Optimization hh/mm/ss/ms)
1000	00:00:02:590	00:00:00:676
2000	00:00:03:383	00:00:00:676
5000	00:00:05:324	00:00:00:676
10000	00:00:08:623	00:00:00:676
20000	00:00:10:792	00:00:00:676
25000	00:00:12:096	00:00:00:676

Table1. Timing Optimization

4.4.2 Memory Optimization

A concept called simple list control is used for creating the list for better memory management. As part of simple list control we are using a C++ concept called smart pointer. A smart pointer takes care of freeing the memory on its own whenever it goes out of range in this case whenever the list items are deleted.

NO. OF ITEMS	MEMORY (Before Optimization in Mb)	MEMORY (After optimization in Mb)

100	32.3	25.3
1000	63.3	25.4

Table2. Memory Optimization

V. Conclusion

The development of MAN/SCANIA infotainment is about the automotive infotainment system that will be integrated in heavy vehicles (of MAN and SCANIA).Main features like Media, Radio, phone connectivity, Navigation, Camera and extended features like Car play, Fleet manager, Mirror link, Remote control app are almost in the feature completion phase and majorly going through stability, defect fixing and variant configuration phase.

The entire software cycle consists of getting the requirement set from the customer, breaking them in to technical functional segments and assigning them to the developers for implementation. After implementation the feature goes for integration and testing to find if any bugs/defects are present. The bugs are reported and sent back to developer for fixing. Developer responsibility is to analyze /reproduce the bug and pin point the defect location and provide a stable fix and send it for verification back to testing team. In accordance with the workflow mentioned above, the Korean Smart Speller feature, in conformation with the generic smart speller has been successfully implemented and the results have been verified as per the customer requirement set. The available characters are displayed based on the navigation database that is stored in the head unit making it user friendly and saving entry time. The second implementation of window listing optimization for media browse application has been tested for various test cases and is found to be bug free. All its uses cases with respect to the media application have also been tested and verified successfully. This optimization technique reduces the number of calls in the D-Bus thereby reducing delay and system congestion. The future implementation and development of both these features will be as per the customer requirement.

Acknowledgements

I express my sincere gratitude to Dr. P. A. Vijaya, Professor and Head, Department of Electronics and Communication Engineering, B.N.M Institute Of Technology, Bengaluru, for her constant support and encouragement in the course of the project. Words are inadequate to thank the HMI Infotainment team at HARMAN International India Pvt. Ltd., Bengaluru for being immensely co-operative and helpful. I would also like to thank Mr. Shivanand Kumbar, External guide and Principle Engineer, for his encouragement and cooperation.I would like to expressspecial thanks to my parents & husband for their moral support, love and endless encouragement.

References

White Paper:

[1] Vageesh Kumar, White paper, Mind Tree, *"Intelligent In-Vehicle Infotainment (IVI)"*, 2013.

Websites:

[2] www.harman.com

[3] www.google.com

Documents:

[4] Technical /Functional specification Project MAN/SCANIA HMI concept 452020_SCANIA_MAN_Technical_Functional_Spec_HMI_7.1_2015-07-31.