

**Research & Vehicle Technology**

**“Infotainment Systems Product Development”**

**Feature – List Browser Protocol**

**AHU Infotainment Subsystem Part Specific Specification (SPSS)**

Version 1.4

**UNCONTROLLED COPY IF PRINTED**

**Version Date: August 30, 2017**

**FORD CONFIDENTIALF**

**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Ver** | **Notes** | |
| **April 1, 2015** | **1.0** | **Initial Release** |  |
|  |  |  |  |
| **October 26, 2015** | **1.1** | **Updated Release** |  |
|  | LBPv2-SR-REQ-128955/B-Item Descriptor Attribute+ | | rpaquet2 - Added Radio Mixed Presets to the table. |
|  | LBPv2-SR-REQ-128955/C-Item Descriptor Attribute | | sberg15: Added item descriptor 0x61 Phone Call Category and 0xA4 Radio Source |
|  | LBPv2-SR-REQ-128954/B-Avaliable List Servers | | sberg15: Added information that list server 0x07 RadioDataService1 is used for "Basic" EPG; added new list server 0x09 RadioDataService3 for "Advanced" EPG data. |
|  | STR-202865/B-Functional Definition | | rpaquet2 - Added Server List Update |
|  | STR-202923/B-Requirements | | sberg15: added server structure for DadioDataService3 - Advanved EPG; |
|  | LBP-REQ-019753/B-Source Icons (TcSE ROIN-301569-1) | | sberg15: Separated AM/FM icon ID into a single AM and FM ID; Added icon ID for DAB; |
|  | LBPv2-REQ-128758/B-List Server Generic Radio 1 - Radio List Structure for Cluster HMI | | sberg15: Changed server structure to support audio source icons in the root list. |
|  | LBP-REQ-132690/B-List Server RadioDataService1 - EPG List Structure for Centerstack HMI | | sberg15: removed Advanced EPG Root server structure. |
|  | LBP-REQ-132691/B-List Server RadioDataService2 - Journaline List Structure for Centerstack HMI | | sberg15: Reduced number of list items in root list from 65534 to 51. |
|  |  |  |  |
| **March 24, 2016** | **1.2** | **Updated Release** |  |
|  | STR-202861/B-List Data Structure | | rpaquet2 - removed 19787 and replaced it with 205695. |
|  | LBPv2-SR-REQ-128955/D-Item Descriptor Attribute | | rpaquet2 - Updated 0xA3 Radio Mixed Presets to include a parameter for PresetState.  sberg15 - Updated 0xA2 Radio Journaline Data to include a parameter for ChildStatus. |
|  | LBPv2-IIR-REQ-130597/B-LBPClient\_LBPServer | | rpaquet2 - removed MD 130388 PrefetchState\_St. |
|  |  |  |  |
| **December 22, 2016** | **1.3** | **Updated Release** |  |
|  | LBPv2-SR-REQ-128954/C-Avaliable List Servers | | rpaquet2 - Added 0x0A Radio 2 for providing source list from AHU to rear display. |
|  | LBPv2-MD-REQ-130389/B-ListServerUpdate\_Ind | | sberg15: corrected typo in ListServer literal All decription field; Added value 0xFFFF All Lists to parameter ListID. |
|  | LBP-FUR-REQ-019741/B-Unavailable Source (TcSE ROIN-294529-2) | | rpaquet2 - Added note to indicate response is for source internal to the responder. |
|  | LBPv2-REQ-128758/C-List Server Generic Radio 1 - Radio List Structure for Cluster HMI | | sberg15: added clarification for server structure notation. |
|  |  |  |  |
| **August 30, 2017** | **1.4** | **Updated Release** |  |
|  | LBPv2-SR-REQ-128955/E-Item Descriptor Attribute | | sberg15: added 0xA5 Radio Mixed Station item descriptor |
|  | LBPv2-SR-REQ-128954/D-Avaliable List Servers+ | | rpaquet2 - Added new list server for Considerate Prompts POI lists |
|  | LBPv2-SR-REQ-128954/E-Avaliable List Servers | | sberg15: Added value 0x0C Radio Data Service 4 for Mixed Station List |

**Table of Contents**

[Revision History 2](#_Toc491870197)

[1 Architectural Design 5](#_Toc491870198)

[1.1 LBP-SV-REQ-019803/A-Static View (TcSE ROIN-40393-1) 5](#_Toc491870199)

[1.2 LBP-CLD-REQ-019804/A-List Browser Client (TcSE ROIN-159174-1) 6](#_Toc491870200)

[1.3 LBP-CLD-REQ-019805/A-List Browser Server (TcSE ROIN-159175-1) 6](#_Toc491870201)

[1.4 List Data Structure 6](#_Toc491870202)

[1.4.1 LBP-SR-REQ-019800/A-List Data Structure (TcSE ROIN-177591-1) 6](#_Toc491870203)

[1.4.2 LBP-SR-REQ-019780/A-Item Index Attribute (TcSE ROIN-40421-1) 6](#_Toc491870204)

[1.4.3 LBP-SR-REQ-019799/A-Object Type Attribute (TcSE ROIN-173633-1) 7](#_Toc491870205)

[1.4.4 LBP-SR-REQ-019801/A-Object State Attribute (TcSE ROIN-177707-1) 7](#_Toc491870206)

[1.4.5 LBP-SR-REQ-019781/A-Data Type Attribute (TcSE ROIN-40318-2) 7](#_Toc491870207)

[1.4.6 LBPv2-SR-REQ-128955/E-Item Descriptor Attribute 7](#_Toc491870208)

[1.4.7 LBPv2-SR-REQ-205695/A-Activation Event Attribute 10](#_Toc491870209)

[1.5 List Hierarchy 10](#_Toc491870210)

[1.5.1 Root List 10](#_Toc491870211)

[1.5.2 Parent-Child Lists and List Entries 10](#_Toc491870212)

[1.5.3 LBP-SR-REQ-019782/A-List Identifier (TcSE ROIN-40422-2) 11](#_Toc491870213)

[1.5.4 LBP-SR-REQ-019788/A-Root Index (TcSE ROIN-40430-1) 11](#_Toc491870214)

[1.5.5 LBPv2-SR-REQ-128954/E-Avaliable List Servers 11](#_Toc491870215)

[1.6 Interface Requirements 11](#_Toc491870216)

[1.6.1 LBPv2-IIR-REQ-130597/B-LBPClient\_LBPServer 11](#_Toc491870217)

[1.6.2 LBPv2-IIR-REQ-130599/A-LBPServer\_LBPClient 13](#_Toc491870218)

[1.7 Navigating Devices and List Servers 17](#_Toc491870219)

[1.7.1 LBP-SR-REQ-019786/A-Requesting List Contents (TcSE ROIN-40428-1) 17](#_Toc491870220)

[1.7.2 LBP-SR-REQ-019790/A-Selecting List Entry (TcSE ROIN-40732-2) 18](#_Toc491870221)

[1.7.3 LBP-SR-REQ-019791/A-Tracking (TcSE ROIN-41670-1) 18](#_Toc491870222)

[1.7.4 LBP-SR-REQ-019784/A-SetItem - Audio Resource Request (TcSE ROIN-40314-1) 19](#_Toc491870223)

[1.7.5 LBP-SR-REQ-019785/A-Label of List (TcSE ROIN-40317-2) 19](#_Toc491870224)

[1.7.6 LBP-SR-REQ-019795/A-Client requests invalid parent-child list (TcSE ROIN-31400-1) 19](#_Toc491870225)

[1.7.7 LBP-SR-REQ-019796/A-Client selects invalid entry (TcSE ROIN-31407-1) 19](#_Toc491870226)

[1.7.8 LBP-SR-REQ-019797/A-Client selects invalid parent-child list (TcSE ROIN-31414-1) 19](#_Toc491870227)

[1.7.9 LBP-SR-REQ-019798/A-SetItem - Server Response (TcSE ROIN-160332-1) 19](#_Toc491870228)

[1.7.10 LBP-SR-REQ-129269/A-Data Prefetch 19](#_Toc491870229)

[2 Functional Definition 20](#_Toc491870230)

[2.1 LBP-FUN-REQ-019707/A-Request Root List (TcSE ROIN-293807-1) 20](#_Toc491870231)

[2.1.1 Use Cases 20](#_Toc491870232)

[2.1.2 White Box View 20](#_Toc491870233)

[2.2 LBP-FUN-REQ-019710/A-Browse a parent List from Root List (TcSE ROIN-293810-1) 20](#_Toc491870234)

[2.2.1 Use Cases 21](#_Toc491870235)

[2.2.2 White Box View 21](#_Toc491870236)

[2.3 LBP-FUN-REQ-019713/A-Selecting an Entry from Root List (TcSE ROIN-293813-1) 21](#_Toc491870237)

[2.3.1 Use Cases 21](#_Toc491870238)

[2.3.2 White Box View 22](#_Toc491870239)

[2.4 LBP-FUN-REQ-019716/A-Browsing Child List from Parent List (TcSE ROIN-293816-1) 22](#_Toc491870240)

[2.4.1 Use Cases 22](#_Toc491870241)

[2.4.2 White Box View 23](#_Toc491870242)

[2.5 LBP-FUN-REQ-019719/A-Selecting and Entry from a parent/Child List (TcSE ROIN-293819-1) 23](#_Toc491870243)

[2.5.1 Use Cases 23](#_Toc491870244)

[2.5.2 White Box View 23](#_Toc491870245)

[2.6 LBP-FUN-REQ-019722/A-Traversing up the Hierarchy from child to parent list (TcSE ROIN-293822-1) 24](#_Toc491870246)

[2.6.1 Use Cases 24](#_Toc491870247)

[2.6.2 White Box View 24](#_Toc491870248)

[2.7 LBP-FUN-REQ-019725/A-Traversing up the Hierarchy from Parent to Root (TcSE ROIN-293825-1) 25](#_Toc491870249)

[2.7.1 Use Cases 25](#_Toc491870250)

[2.7.2 White Box View 25](#_Toc491870251)

[2.8 LBP-FUN-REQ-019728/A-Browsing down a list (TcSE ROIN-293828-1) 26](#_Toc491870252)

[2.8.1 Use Cases 26](#_Toc491870253)

[2.8.2 White Box View 26](#_Toc491870254)

[2.9 LBP-FUN-REQ-019731/A-Browsing up a list (TcSE ROIN-293831-1) 27](#_Toc491870255)

[2.9.1 Use Cases 27](#_Toc491870256)

[2.9.2 White Box View 27](#_Toc491870257)

[2.10 LBP-FUN-REQ-130790/A-Prefetch inidcation 28](#_Toc491870258)

[2.10.1 Use Cases 28](#_Toc491870259)

[2.10.2 White Box View 28](#_Toc491870260)

[2.11 LBPv5-FUN-REQ-132838/A-List Browser Icon and Structured Data 29](#_Toc491870261)

[2.11.1 Requirements 29](#_Toc491870262)

[2.12 LBP-FUN-REQ-132987/A-New device connected 33](#_Toc491870263)

[2.12.1 Use Cases 33](#_Toc491870264)

[2.12.2 White Box View 34](#_Toc491870265)

[2.13 LBP-FUN-REQ-130789/A-Server list update 35](#_Toc491870266)

[2.13.1 Use Cases 35](#_Toc491870267)

[2.13.2 White Box View 35](#_Toc491870268)

[3 Appendix: Reference Documents 37](#_Toc491870269)

# Architectural Design

The List Browser Protocol is a general way for devices (e.g. radio head unit, remote display) to store and share information across the network. The strategy is based upon devices storing information in a standardized list data structure which can then be used for sharing information between clients and servers. The protocol has been developed to allow any client the ability to browse through and select entries from a list of entries located on any device also implementing this protocol. Lists can be navigated up and down to select entries and traversed forward and backward between parent and child lists. The protocol is flexible in that one common strategy can be applied to browse multiple types of data (e.g. song lists, play lists, audio sources, folders, etc.). The interface can be developed to support the specific data types but can also be flexible and generic to allow new types of data to be accessed without requiring a software update on the client.

The following figure is a high-level view of the LBP. In the figure a device is shown which contains multiple list severs each dedicated to specific functional area. Other implementation may have one list server dedicated to several functional areas.

Client

(e.g. Remote Display)

Device

(e.g. Radio Head

Unit)

Data

--

--

--

--

Data

--

--

--

--

--

Data

--

--

--

List Server\_FA\_1

Internal Memory

Client browses lists and selects entries via control commands

Data

--

--

--

--

Data

--

--

--

--

--

List Server\_FA\_2

Internal Memory

List Data Structures

This primary section serves to define the underlining strategy of the list browser protocol. The actual usage of the strategy will be defined in other Functional Area Specifications (FAS) with references back to this primary section. The following sub-sections shall provide further descriptions of the list data structure, hierarchy within list and sub-lists, and navigating through lists.

## LBP-SV-REQ-019803/A-Static View (TcSE ROIN-40393-1)

The following is a logical view of the list browser protocol architecture.



## LBP-CLD-REQ-019804/A-List Browser Client (TcSE ROIN-159174-1)

The List Brower Client is responsible for addressing list servers and requesting/activating items. The client also is responsible for the HMI output of the information received from the server.

## LBP-CLD-REQ-019805/A-List Browser Server (TcSE ROIN-159175-1)

The List Brower Server is responsible for navigating the lists, keeping track of the active list and the providing information back to the client.

## List Data Structure

The List Data Structure is an information interface which allows a server to present its features and other types of information to clients in a standardized form. By using this protocol and the data structures defined, servers can utilize a standard interface for sharing information with clients.

### LBP-SR-REQ-019800/A-List Data Structure (TcSE ROIN-177591-1)

The list data structure is realized as a container for sub-lists and list entries. Each list data structure shall contain the following attributes for each item (list or entry) in a list:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Item Index** | **Data Type** | **Activation Event** | **Object Type** | **Object State** | **Item Descriptor** |
| 0x0001 | 0x00-0xFF | Supported/Not Supported | Entry/List | Active/In-Active | {Descriptor Tag} |
| .. | .. | .. | .. | .. | .. |
| 0xFFFF | .. | .. | .. | .. | .. |

The actual implementation of the storage of the list data structures is left to the device implementer but the method for exchange of information is defined herein.

### LBP-SR-REQ-019780/A-Item Index Attribute (TcSE ROIN-40421-1)

The item index is used to denote the position of the item within the list that was specified by the ActiveListID field. Entries shall be numbered so that the first item in a list begins with index number 0x0001, the second position is 0x0002, and so on. The largest addressable item in an ActiveListID is 0xFFFE (65534). 0xFFFF is reserved.

### LBP-SR-REQ-019799/A-Object Type Attribute (TcSE ROIN-173633-1)

The Object Type attribute within the list data structure is used to indicate if the object is a "List" object or an "Entry" object. "List" objects are lists that can be navigated via the GetItemInfo.Rq() method or selected for activation via the SetItemInfo.Rq() method. "Entry" objects can only be selected for activation via the SetItemInfo.Rq() method.

### LBP-SR-REQ-019801/A-Object State Attribute (TcSE ROIN-177707-1)

The Object State attribute within the list data structure shall be used to indicate if the object is currently activated/in use on the server. This attribute typically pertains to "Entry" objects but may be used by "List" objects.

For example, an entry within a list of song tracks represents a track that is actively playing on the server. When the browsing client requests the list of tracks from the server the server will indicate, via the Object State attribute, that this item is currently active within the list. The client may then provide an indication to the user that this entry is currently active on this server.

### LBP-SR-REQ-019781/A-Data Type Attribute (TcSE ROIN-40318-2)

The Data Type attribute within the list data structure is used to denote the information stored in the Item Descriptor attribute for each item. The Data Type attribute shall be used to indicate the type of information the item represents. For example, if the data type is "Album" then the client knows that this item represents the name of an album.

### LBPv2-SR-REQ-128955/E-Item Descriptor Attribute

Each item shall also possess the Item Descriptor attribute which can also be used by the client for HMI purposes. The Item Descriptor is typically a large text string which contains a concatenated set of text fields. Each of these fields can be parsed by the client and used for HMI purposes.

The following table defines the link between Data Type Attribute and the Item Descriptor Attribute:

|  |  |  |  |
| --- | --- | --- | --- |
| **Encoding** | **Data Type** | **Item Descriptor** | **Descriptor Length** |
| 0x00 | General Reserved |  |  |
| 0x01 | General Reserved |  |  |
| 0x02 | Generic Text | {Generic Text} | 25 characters + EOS |
| … | General Reserved |  |  |
| 0x1F | General Reserved |  |  |
| 0x20 | Media Type | {SourceIcon}{MediaSourceName} | {$0-$FF represents Source icon, $0 = Invalid }{25 characters + EOS} |
| 0x21 | Metadata Category | {CatIcon}{CatName} | {$0-$FF represents Category icon, $0 = Invalid}{25 characters + EOS} |
| … | Media Reserved |  |  |
| 0x3F | Media Reserved |  |  |
| 0x40 | Navigation POI | {Direction}{Distance}{POIName} | {$0-$FF Represents direction icon, $0 = Invalid }{8 characters + EOS}{25 characters + EOS} Info: POI name could also be a destination name. |
| … | Navigation Reserved |  |  |
| 0x5F | Navigation Reserved |  |  |
| 0x60 | Phone CallerID | {CallTypeIcon}{PhoneTypeIcon}{CallerID} | {$0-$FF Represents call type icon, $0 = Invalid }{$0-$FF Represents phone type icon, $0 = Invalid }{25 characters + EOS} |
| 0x61 | Phone Call Category | { CallTypeIcon }{CatName} | {$0-$FF Represents call type icon, $0 = Invalid }{25 characters + EOS} |
| … | Phone Reserved |  |  |
| 0x7F | Phone Reserved |  |  |
| 0x80 | BT Device | {DeviceID}{ConnectedIcon}{DeviceName} | {$0-$F Represents the index of the bonded BT device, $0 = Invalid}{$0-$FF Represents the connected icon, $0 = Invalid}{25 characters + EOS} |
| … | BT Device Reserved |  |  |
| 0x9F | BT Device Reserved |  |  |
| 0xA0 | Radio Station | {RadioBand}{IndexNumber}{Frequency}  {StationIcon}{StationName} | {$1 = AM, $2 = FM, $3 = DAB, $4 = SDARS, $5 = HD, $0 = Invalid}{$0-$1E Represents the stored station number, $0 = Invalid}{}{$0-$FFFF Represents the frequency, BlockID, SDARS channel number}{$0-$FF Represents a station icon (e.g. HD), $0 = Invalid}{16 characters + EOS} |
| 0xA1 | Radio EPG Data | {Icon}{Hour}{Minute}{ProgramName} | {$0-$F Represents EPG program icon, $0 = Invalid }{$0-$17 Represents the hour, $FF = Invalid}{$0-$3B Represents the minute, $FF = Invalid}{128 characters + EOS} |
| 0xA2 | Radio Journaline Data | {JournalineListEntry}{ChildStatus} | {50 characters + EOS}{$1 = NotAvailable, $2 = Available, $0 = Invalid} |
| 0xA3 | Radio Mixed Presets | {PresetNumber}{PresetState}{RadioBand}{Frequency}{HD Number}{StationNameShort}{StationNameLong} | {$1-$1E, $0 = Invalid}{$0 = Empty, $1 = Available} {$1 = AM, $2 = FM, $3 = DAB, $4 = SDARS, $0 = Invalid} {$0-$FFFF Represents the frequency, DAB BlockID, SDARS channel number}  {HD Number = $0 - $F} {StationNameShort =  9 characters max + EOS,  RDS Markets:  PSName = 8 characters max,  No PSName = xxx.yyMHz,  Non RDS Markets: AM = xxxx kHz,  FM = xxx.y MHz,  AM HD = AM,  FM HD1 = FM,  FM HD2+ = xxx.y HDz,  SDARS: Shortname = 8 characters max,  DAB: ServiceName = 8 characters DAB No Station Name = Blockxxx to APIM and Block xxx to the IPC or MFD}  {StationNameLong =  DAB:  ServiceName = 16 character long name max plus EOS  AMFM and SDARS = EOS} |
| 0xA4 | Radio Source | {SourceIcon}{RadioSourceName} | {$0-$FF represents Source icon, $0 = Invalid }{25 characters + EOS} |
| 0xA5 | Radio Mixed Station | {SourceIcon}{StationNameLong}{StationNameShort} | {$0-$FF represents Source icon, $0 = Invalid}{StationNameLong = DAB: ServiceName = 16 character long name max plus EOS;  AMFM and SDARS = EOS}{StationNameShort = 9 characters max + EOS, RDS Markets:  PSName = 8 characters max} |
| … | Radio Reserved |  |  |
| 0xAF | Radio Station Reserved |  |  |

**Note:** A list server can contain multiple data types.

### LBPv2-SR-REQ-205695/A-Activation Event Attribute

The Activation Event attribute is used to indicate whether an item can be selected or not via the SetItemInfo.Rq() method. Entry Objects that do not support an Activation Event should be indicated as not selectable via the HMI (e.g. empty preset etc.). List Objects may or may not support an Activation Event depending on the server's usage of the object. If a List Object supports an Activation Event it can be selected via SetItemInfo.Rq() method or entered via the GetItemInfo.Rq() method.

## List Hierarchy

This protocol assembles lists, sub-lists, and entry data structures into a hierarchy which is useful when representing information that can be categorized in a hierarchal manner.

### Root List

At the highest level exists the "Root List" and at the minimum the Root List must exist on the server. The contents of the root list are specific to each device. Root Lists can contain sub-lists and root entries.

### Parent-Child Lists and List Entries

At the next level exist sub-lists and list entries. List and sub-lists can better be defined as having a parent-child relationship. Therefore, the next levels in the hierarchy can be defined as the Parent lists followed by Child lists. Parent lists can contain child lists and parent entries. Child Lists contain child entries and also sub-lists (i.e. become parents and spawn further child lists).

List Entries exist at the bottom of the hierarchy and are the lowest selectable element which can be traversed in a list (i.e. an entry can not be navigated any further).

The following diagram illustrates the general relationship between the kinds of lists and how they form a hierarchy.

Root List

0x0001

0x0002

0x0003

…

Item Index

Data Type

Item Descr

0xXX

{Descriptor}

0xXX

{Descriptor}

0xXX

{Descriptor}

0xXX

{Descriptor}

Parent List

0x0001

0x0002

0x0003

…

Item Index

Data Type

Item Descr

0xXX

{Descriptor}

0xXX

{Descriptor}

0xXX

{Descriptor}

0xXX

{Descriptor}

Parent List

0x0001

0x0002

0x0003

…

Item Index

Data Type

Item Descr

0xXX

{Descriptor}

0xXX

{Descriptor}

0xXX

{Descriptor}

0xXX

{Descriptor}

Parent-Child List

0x0001

0x0002

0x0003

…

Item Index

Data Type

Item Descr

0xXX

{Descriptor}

0xXX

{Descriptor}

0xXX

{Descriptor}

0xXX

{Descriptor}

Child List

0x0001

0x0002

0x0003

…

Item Index

Data Type

Item Descr

0xXX

{Descriptor}

0xXX

{Descriptor}

0xXX

{Descriptor}

0xXX

{Descriptor}

Figure 1 – Lists-Entry Hierarchy

### LBP-SR-REQ-019782/A-List Identifier (TcSE ROIN-40422-2)

A unique list ID, which is assigned by the list server, identifies each list within the list server. The list ID shall be constant during the life of the list.

### LBP-SR-REQ-019788/A-Root Index (TcSE ROIN-40430-1)

Each list server shall have a Root List associated to List\_ID 0x0000. The Root List shall always be accessible from any point in a list.

### LBPv2-SR-REQ-128954/E-Avaliable List Servers

The following are the available addressable list servers:

|  |  |  |
| --- | --- | --- |
| **List Server ID** | **Name** | **Comment** |
| 0x01 | Navigation Info | Used for browsing Nav POIs, previous dest, etc. |
| 0x02 | Generic Media 1 | Used for browsing media data (e.g. Artist names, titles, etc.) |
| 0x03 | Phone Info | Used for browsing phone book, call lists, etc. |
| 0x04 | Generic List 1 | Contains only items with Data Types 0,1,2. |
| 0x05 | Radio 1 | Used for browsing radio data (e.g. Preset lists for AM/FM/DAB/SDARS, etc.) started from cluster. |
| 0x06 | Remote CD1 | Used for browsing media data (e.g. Artist names, titles, etc.) |
| 0x07 | Radio Data Service1 | Used to transmit DAB data service Basic EPG. |
| 0x08 | Radio Data Service2 | Used to transmit DAB data service Journaline. |
| 0x09 | Radio Data Service3 | Used to transmit DAB data service Advanced EPG. |
| 0x0A | Radio 2 | Used for browsing radio Source information (e.g. AM,FM,SAT,DAB,CD.) started from Rear Display. |
| 0x0B | Considerate Prompt 1 | Static List used to transmit POI list related to Considerate Prompts |
| 0x0C | Radio Data Service4 | Used to transmit a mixed station list to the client. |
| 0x0D – 0xFF | Reserved |  |

## Interface Requirements

### LBPv2-IIR-REQ-130597/B-LBPClient\_LBPServer

#### LBPv2-MD-REQ-130601/A-ItemInfo\_Rq

Message Type: Request

This request signal is used to get or set list content from the list browse server.

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Literals** | **Value** | **Description** |
| OpCode | - | - | Parameter OpCode is used to distinguish between requesting list data or select an executable item. |
|  | Inactive | 0x0 |  |
|  | GetItemInfo\_Rq | 0x1 | Value GetItemInfo\_Rq is used to request list data from a list object. |
|  | SetItemInfo\_Rq | 0x2 | Value SetItemInfo\_rq is used by the Client to activate a selected item index during browsing of the device. The value shall only be used on items which support an ActivationEvent. Parameter NbrOfItems and StartItemInd shall be set to 0x0 in this case. |
|  | Reserved | 0x3 |  |
|  |  |  |  |
| SetListServ | - | - | Parameter setListServ is used to address the requested list server. |
|  | Inactive | 0x00 |  |
|  | ServerID\_1 | 0x01 |  |
|  | … | 0x02 |  |
|  | All | 0xFF | Value All is used address all available list servers. |
|  |  |  |  |
| ActiveListID | - | - | Parameter ActiveListID contains the list ID from which the items are being selected. This information is provided by the server's response method. |
|  | Root | 0x0000 |  |
|  | List ID1 | 0x0001 |  |
|  | …. | 0x0002 |  |
|  | List ID 65534 | 0xFFFE |  |
|  | Reserved | 0xFFFF |  |
|  |  |  |  |
| ItemIndex | - | - | Parameter ItemIndex contains the selected item Index for the requested list. |
|  | BrowseActiveListID | 0x0000 | Value BrowseActiveListID is used if the client is only browsing up/down the entries of the active list. In this case this value shall be set to 0x0000 (BrowseActiveListID) which indicates to the server to provide the items requested for the active list. |
|  | EntryIndex\_1 | 0x0001 |  |
|  | …. |  |  |
|  | ParentOfActiveListID | 0xFFFF | Value ParentOfActiveListID is used if the client needs to retrieve the parent list of the active list. In this case this value shall be set to 0xFFFF (ParentOfActiveListID). This is an indication to the server to navigate to the parent list of the active list and provide that data as requested. |
|  |  |  |  |
| NbrOfItems |  |  | Parameter NbrOfItems is used to indicate how many items to provide in the response message. |
|  | Inactive | 0x0 |  |
|  | 1 item | 0x1 |  |
|  | … |  |  |
|  | 63 items | 0x3F |  |
|  |  |  |  |
| StartItemInd |  |  | Parameter StartItemInd is used to adreess the index value of where to start the request in the response message |
|  | Inactive | 0x0000 |  |
|  | 1 | 0x0001 |  |
|  | … |  |  |
|  | 65534 | 0xFFFE |  |
|  | Reserved | 0xFFFF |  |

### LBPv2-IIR-REQ-130599/A-LBPServer\_LBPClient

#### LBPv2-MD-REQ-130602/A-ItemInfo\_Rsp

Message Type: Response

This response signal is used to transmit the requested list content to the list browse client.

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Literals** | **Value** | **Description** |
| RspListServ | - | - | Parameter RspListServ is used to indicate which list server is providing the response. |
|  | Inactive | 0x00 |  |
|  | ServerID\_1 | 0x01 |  |
|  | … |  |  |
|  | Reserved | 0xFF |  |
|  |  |  |  |
| OpCodeRsp | - | - | Parameter OpCodeRsp is used to return the value of the OpCode associated with the data in the response message. |
|  | Inactive | 0x00 |  |
|  | GetItemInfoRsp | 0x01 |  |
|  | SetItemInfoRsp | 0x02 |  |
|  |  |  |  |
| ActiveListID | - | - | Parameter ActiveListID shall contain the current list ID from which the items are being selected. |
|  | Root | 0x0000 |  |
|  | List ID1 | 0x0001 |  |
|  | …. | 0x0002 |  |
|  | List ID 65534 | 0xFFFE |  |
|  |  |  |  |
| ParentListID | - | - | The parameter ListID shall contain the parent list ID of the current ActiveListID. |
|  | Root | 0x0000 |  |
|  | List ID1 | 0x0001 |  |
|  | …. | 0x0002 |  |
|  | List ID 65534 | 0xFFFE |  |
|  | Reserved | 0xFFFF |  |
|  |  |  |  |
| NbrOfItemsRtn | - | - | Parameter NbrOfItemsRtn is used to indicate how many items provided in the response message |
|  | 0 | 0x0000 |  |
|  | 1 | 0x0001 |  |
|  | …. |  |  |
|  | 30 | 0x1E |  |
|  |  |  |  |
| NbrItemsInSelection |  |  | Parameter NbrItemsInSelection is used to indicate the total number of all items in the active list id. |
|  | 0 | 0x0000 |  |
|  | 1 | 0x0001 |  |
|  | …. |  |  |
|  | 65535 | 0xFFFF |  |
|  |  |  |  |
| *stringItemContent (Array(1.. NbrOfItemsRtn) of record (ItemIndex, DataType, ActivationEvent, ObjectType, ObjectState, ItemDescriptor)* |  |  |  |
|  |  |  |  |
| ItemIndex | List Title | 0x0000 | Parameter ItenIndex is used to associate an index to each list item. |
|  | 1 | 0x0001 |  |
|  | … |  |  |
|  | 65535 | 0xFFFF |  |
|  |  |  |  |
| DataType | Reserved | 0x00 |  |
|  | DataTypeID\_1 | 0x01 |  |
|  | … |  |  |
|  | Reserved | 0xFF |  |
|  |  |  |  |
| ActivationEvent |  |  | Parameter ActivationEvent is used to indicate if a list entry is able to be activated or not. |
|  | Not Supported | 0x0 |  |
|  | Supported | 0x1 |  |
|  |  |  |  |
| ObjectType |  |  | Parameter ObjectType is used to indicate if the returned list entry is a List or Entry Object. List objects can contain additional data, entry objects can be executed. |
|  | List Label | 0x0 |  |
|  | Entry Object | 0x1 |  |
|  | List Object | 0x2 |  |
|  |  |  |  |
| ObjectState |  |  | The parameter ObjectState is used to indicate if the returned list entry is the active on the server or not. |
|  | Inactive | 0x0 |  |
|  | Active | 0x1 |  |
|  |  |  |  |
| ItemDescriptor |  |  | The parameter Item Descriptor is typically a large text string which contains a concatenated set of text fields. Each of these fields can be parsed by the client and used for HMI purposes. |
|  | {Descriptor Tag} |  |  |

Example ItemContent:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Item Index** | **Data Type** | **ActivationEvent** | **ObjectType** | **Item Descriptor** |
| 0x0000 | 0x02 | No | List Label | {Generic Text} |
| 0x0001 | 0x00-0xFE | Yes/No | .. | {Descriptor Tag} |
| .. | .. | .. | .. |  |
| 0xFFFE | .. | .. | .. |  |

#### LBPv2-MD-REQ-130389/B-ListServerUpdate\_Ind

Message Type: Indication

This indication method is used to indicate a server list update to the client. This indication can be used to start fetching the updated list content. This method shall not be used during initial server list build up.

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Literals** | **Value** | **Description** |
| ListServer | - | - | Parameter ListServer is used to address the list server which is/was updated. |
|  | Inactive | 0x00 |  |
|  | ServerID\_1 | 0x01 |  |
|  | … | 0x02 |  |
|  | All | 0xFF | Value All is used to address all available list servers. |
|  |  |  |  |
| ListID | - | - | The parameter ListID is used to indicate which server list is getting updated. |
|  | Root | 0x0 | Root List is getting updated |
|  | ListID1 | 0x0001 | ListID1 is getting updated |
|  | ListID2 | 0x0002 | ListID2 is getting updated |
|  | … |  |  |
|  | List ID65534 | 0xFFFE | ListID65534 is getting updated |
|  | All Lists | 0xFFFF | Value All Lists is used to indicate that all Lists within a server have been changed. |
|  |  |  |  |
| ParentID | - | - | The parameter ParentID is used to indicate the parent list which contains the updating ListID. |
|  | Root | 0x0 | Root List is ParentID |
|  | ListID1 | 0x0001 | ListID1 is ParentID |
|  | ListID2 | 0x0002 | ListID2 is ParentID |
|  | … |  |  |
|  | List ID65534 | 0xFFFE | ListID65534 is ParentID |
|  | Reserved | 0xFFFF |  |
|  |  |  |  |
| EntryIndex | - | - | The parameter EntryIndex shall be used to indicate which list entry from the ParentID is getting updated. This could be used to indicate this update via HMI output (e.g. grey out list entry etc.) |
|  | NoDataExists | 0x0000 | NoDataExisis shall be set if the root list is getting updated. |
|  | EntryIndex1 | 0x0001 | Entry 1 from parent list is getting updated. |
|  | EntryIndex2 | 0x0002 | Entry 2 from parent list is getting updated. |
|  | … |  |  |
|  | EntryIndex65534 | 0xFFFE | Entry 65534 from parent list is getting updated. |
|  | Reserved | 0xFFFF |  |
|  |  |  |  |
| ListStatus | - | - | The parameter is used to transmit the different states of the server list. |
|  | Inactive | 0x0 |  |
|  | Valid | 0x1 | Valid is set if the list is updated and ready. |
|  | Updating | 0x2 | Updating is set if the list is getting updated with new data. |
|  | Reserved | 0x3 |  |

## Navigating Devices and List Servers

The process of an HMI client browsing the information available on a particular device begins with the client first accessing a particular list server present on the device. Once an HMI client has accessed a particular list server, the client can scroll up and down through items in the list and then request an entry in the list. The client can also move forward and backward between lists and sub-lists by requesting the particular (parent or child) list from the list server.

Several methods are provided for navigating the hierarchy and retrieval of information, GetItemInfo.Request(), SetItemInfo.Request(), and GeItemtInfo.Response(). The client utilizes the GetItemInfo.Request() methods for requesting item information and utilizes the SetItemInfo.Request for item activation. The server utilizes the GetItemInfo.Response() method for responding to information requests. The details of these methods are defined in the Interface Requirements portion of this section.

### LBP-SR-REQ-019786/A-Requesting List Contents (TcSE ROIN-40428-1)

The browsing client shall utilize the GetItemInfo.Rq() method for managing the retrieval of data from a particular list server. To browse through a device via the HMI, the client must address a particular list server. For example, to view the contents of a USB device, the client shall target the USB list server and based on the list data structures (retrieved earlier) stored on this server the HMI can then browse through parent-child lists (e.g. artists, albums, etc.).

To support an example, imagine a server contains the following items:

0x0001

0x0002

0x0003

…

Item Index

Data Type

Item Descr

0x02

{Name\_x}

0x01

{List\_A}

0x01

{List\_B}

0xXX

{Descriptor}

0x0001

0x0002

0x0003

…

Item Index

Data Type

Item Descr

0xXX

{Descriptor}

0xXX

{Descriptor}

0xXX

{Descriptor}

0xXX

{Descriptor}

List\_B

(A\_ID = 0x0013, P\_ID = 0x0000)

0x0001

0x0002

0x0003

…

Item Index

Data Type

Item Descr

0x01

{List\_B1}

0xXX

{Descriptor}

0xXX

{Descriptor}

0xXX

{Descriptor}

0x0001

0x0002

0x0003

…

Item Index

Data Type

Item Descr

0xXX

{Descriptor}

0xXX

{Descriptor}

0x01

{List\_B13}

0xXX

{Descriptor}

0x0001

0x0002

0x0003

…

Item Index

Data Type

Item Descr

0x02

{Name\_xx}

0x02

{Name\_xy}

0x03

{TrackName\_x}

0xXX

{Descriptor}

List\_A

(A\_ID = 0x0011, P\_ID = 0x0000)

List\_B1

(A\_ID = 0x0022, P\_ID = 0x0013)

List\_B13

(A\_ID = 0x0029, P\_ID = 0x0022)

(SvrID = 0x13) Root

(A\_ID = 0x0000, P\_ID = 0x0000)

Where A\_ID is the ActiveListID and P\_ID is the ParentListID of the ActiveList. If the client is in the root list and wishes to request the contents of item 3 (List\_B) within the root list, then the request parameters would be set to the following, ActiveListID = 0x0000 (root is being browsed), ItemIndex = 0x0003 (item selected), NumberOfItems = 0x05 (number of entries requested), StartItemInd = 0x0001 (starting index of entries to return).

The response back from the server would be A\_ID = 0x0013, P\_ID = 0x0000 with the data structure depicted for List\_B in the figure above.

If the client now wishes to select item 1 (List\_B1) within the active sub-list, then the request parameters would be set to the following, ActiveListID = 0x0013 (active list), ItemIndex = 0x0001 (item selected), NumberOfItems = 0x05 (number of entries requested), StartItemInd = 0x0001 (starting index of entries to return)..

The response back from the server would be A\_ID = 0x0022, P\_ID = 0x0013 with the data structure depicted for List\_B1 in the figure above.

### LBP-SR-REQ-019790/A-Selecting List Entry (TcSE ROIN-40732-2)

The browsing client shall utilize the SetItemInfo.Rq() method for activation of a particular item within the ActiveListID. This method can only be used by items which support ActivationEvents.

To support the example described previously: If the active list is reported A\_ID = 0x0022, P\_ID = 0x0013 with the data structure depicted for List\_B1 in the figure above. If the client now wishes to select and activate item 3, assuming the ActivationEvent is supported for this item, within the active sub-list, then the settings for the SetItemInfo.Rq() method would be as follows: ActiveListID = 0x0022 (active list), ItemIndex = 0x0003 (item selected).

### LBP-SR-REQ-019791/A-Tracking (TcSE ROIN-41670-1)

The server shall be responsible for tracking the navigation between the lists.

### LBP-SR-REQ-019784/A-SetItem - Audio Resource Request (TcSE ROIN-40314-1)

If a SetItemInfo.Rq() requires a change to the audio resources, the server shall be responsible for issuing the request for connection of the respective audio resource.

### LBP-SR-REQ-019785/A-Label of List (TcSE ROIN-40317-2)

Each list within a list server shall have an associated textual label/title. The server shall provide the label of each list to the client.

Within GetItemInfo.Rsp(), the ItemIndex equal to 0x0000 shall be used to contain the label of the active list. The Data Type for the label shall be set equal to 0x02 (Generic text). The ActivationEvent shall be set to "No" for labels. The Object Type shall be set to 0x00 (List Label).

The label shall be provided when the StartItemIndex of GetItemInfo.Rq() is equal to 0x0001.

The label shall not be counted as a member of NbrItemsInSelection.

The label shall be counted as a member of NbrOfItemsRtn by both the client and the server. Therefore, when the client requests information starting at index 0x0001and would like five items of information returned, the client must request NbrOfItems equal to six. Likewise the server will respond back with NbrOfItemsRtn equal to six. With a starting index greater then 0x0001 with five items requested, the client would request NbrOfItems equal to five with the server responding back with NbrOfItemsRtn equal to five.

### LBP-SR-REQ-019795/A-Client requests invalid parent-child list (TcSE ROIN-31400-1)

If the client requests a parent-child that is either out of range or does not exist, the server shall respond back with CES = 0x14 (Final Result – Failure, requested index out of range).

### LBP-SR-REQ-019796/A-Client selects invalid entry (TcSE ROIN-31407-1)

If the client selects invalid entry that is either out of range or does not exist, the server shall respond back with CES = 0x14 (Final Result – Failure, requested index out of range).

### LBP-SR-REQ-019797/A-Client selects invalid parent-child list (TcSE ROIN-31414-1)

If the client selects a parent-child that is either out of range or does not exist, the server shall respond back with CES = 0x14 (Final Result – Failure, requested index out of range).

### LBP-SR-REQ-019798/A-SetItem - Server Response (TcSE ROIN-160332-1)

Upon reception of a valid SetItemInfo.Rq(), the server shall provide a response with CES = 0x01 (Final Result – Success). All fields after the CES field will not be transmitted.

### LBP-SR-REQ-129269/A-Data Prefetch

While the List Browse Client is in prefetch state then the List Browse Server shall ignore the activation event attribute if the client requests for list objects which have an activation event supported. This is required to avoid e.g. activation of a radio band when requesting its list content.

# Functional Definition

## LBP-FUN-REQ-019707/A-Request Root List (TcSE ROIN-293807-1)

### Use Cases

#### LBP-UC-REQ-019708/A-Request Root List (TcSE ROIN-292216-1)

|  |  |
| --- | --- |
| **Actors** | System |
| **Pre-conditions** | Infotainment Network is active. |
| **Scenario Description** | Client requests Root List from server. |
| **Post-conditions** | Root List is retrieved from server. |
| **List of Exception Use Cases** | N/A |
| **Interfaces** | G-HMI, Vehicle System Interface |

### White Box View

#### White Box Scenarios

##### LBP-SD-REQ-019709/A-Request Root List from List Server (TcSE ROIN-39796-1)

**Linked Elements**

LBP-UC-REQ-019762/A-Request Root List (TcSE ROIN-30390-1)

Sequence Diagram



## LBP-FUN-REQ-019710/A-Browse a parent List from Root List (TcSE ROIN-293810-1)

### Use Cases

#### LBP-UC-REQ-019711/A-Browse a Parent List from the Root List (TcSE ROIN-292217-1)

|  |  |
| --- | --- |
| **Actors** | System |
| **Pre-conditions** | Infotainment Network is active.  The root list has been obtained from the server.  Root list contains parent lists. |
| **Scenario Description** | The client requests a parent list within the root list for browsing.  For example, the root list contains several folders and the user selects a folder to browse/read its contents. |
| **Post-conditions** | Server responds with requested parent list structure. |
| **List of Exception Use Cases** | N/A |
| **Interfaces** | G-HMI, Vehicle System Interface |

### White Box View

#### White Box Scenarios

##### LBP-SD-REQ-019712/A-Browse a Parent List from the root list (TcSE ROIN-39798-1)

**Linked Elements**

LBP-UC-REQ-019764/A-Browse a Parent List from the Root List (TcSE ROIN-30397-1)

Sequence Diagram



## LBP-FUN-REQ-019713/A-Selecting an Entry from Root List (TcSE ROIN-293813-1)

### Use Cases

#### LBP-UC-REQ-019714/A-Selecting an Entry from the Root List (TcSE ROIN-292218-1)

|  |  |
| --- | --- |
| **Actors** | System |
| **Pre-conditions** | Infotainment Network is active.  The root list has been obtained from the server.  The root list contains selectable entries. |
| **Scenario Description** | The client selects an entry from the root list. |
| **Post-conditions** | Server responds with activation of the selected entry. |
| **List of Exception Use Cases** | N/A |
| **Interfaces** | G-HMI, Vehicle System Interface |

### White Box View

#### White Box Scenarios

##### LBP-SD-REQ-019715/A-Selecting an Entry from the root list (TcSE ROIN-39799-1)

**Linked Elements**

LBP-UC-REQ-019766/A-Selecting a Entry from the Root List (TcSE ROIN-30421-1)

Sequence Diagram



## LBP-FUN-REQ-019716/A-Browsing Child List from Parent List (TcSE ROIN-293816-1)

### Use Cases

#### LBP-UC-REQ-019717/A-Browsing a Child List from a Parent List (TcSE ROIN-292219-1)

|  |  |
| --- | --- |
| **Actors** | System |
| **Pre-conditions** | Infotainment Network is active.  The client is browsing a parent list which contains child lists. |
| **Scenario Description** | The client requests a child list from within a parent list.  For example, the user is browsing Folder\_A (parent) which contains several folders (children). The user selects one of the child folders for browsing. |
| **Post-conditions** | The server responds with contents of selected child list. |
| **List of Exception Use Cases** | N/A |
| **Interfaces** | G-HMI, Vehicle System Interface |

### White Box View

#### White Box Scenarios

##### LBP-SD-REQ-019718/A-Browse a Child List from the Parent list (TcSE ROIN-39800-1)

**Linked Elements**

LBP-UC-REQ-019768/A-Browsing a Child List from a Parent List (TcSE ROIN-30427-1)

Sequence Diagram



## LBP-FUN-REQ-019719/A-Selecting and Entry from a parent/Child List (TcSE ROIN-293819-1)

### Use Cases

#### LBP-UC-REQ-019720/A-Selecting an Entry from a Parent-Child List (TcSE ROIN-292220-1)

|  |  |
| --- | --- |
| **Actors** | System |
| **Pre-conditions** | Infotainment Network is active.  The parent-child list contains selectable entries. |
| **Scenario Description** | The client selects an entry from the parent-child list. |
| **Post-conditions** | Server responds with activation of the selected entry. |
| **List of Exception Use Cases** | N/A |
| **Interfaces** | G-HMI, Vehicle System Interface |

### White Box View

#### White Box Scenarios

##### LBP-SD-REQ-019721/A-Selecting an Entry from Parent/Child List (TcSE ROIN-39801-1)

**Linked Elements**

LBP-UC-REQ-019770/A-Selecting an Entry from a Parent-Child List (TcSE ROIN-30439-1)

Sequence Diagram



## LBP-FUN-REQ-019722/A-Traversing up the Hierarchy from child to parent list (TcSE ROIN-293822-1)

### Use Cases

#### LBP-UC-REQ-019723/A-Traversing up the Hierarchy from Child to Parent (TcSE ROIN-292221-1)

|  |  |
| --- | --- |
| **Actors** | System |
| **Pre-conditions** | Infotainment Network is active.  Client currently browsing a Child List. |
| **Scenario Description** | The client requests the parent list of the current child list. For example, the user is browsing Folder\_B which is a child of Folder\_A and wishes to view the content list of Folder\_A. |
| **Post-conditions** | Server responds back with the Parent List of the current Child List. |
| **List of Exception Use Cases** | N/A |
| **Interfaces** | G-HMI, Vehicle System Interface |

### White Box View

#### White Box Scenarios

##### LBP-SD-REQ-019724/A-Traversing up the hieracy from child to parent list (TcSE ROIN-39802-1)

**Linked Elements**

LBP-UC-REQ-019772/A-Traversing up the hierarchy from Child to Parent (TcSE ROIN-30445-1)

Sequence Diagram



## LBP-FUN-REQ-019725/A-Traversing up the Hierarchy from Parent to Root (TcSE ROIN-293825-1)

### Use Cases

#### LBP-UC-REQ-019726/A-Traversing up the Hierarchy from Parent to Root (TcSE ROIN-292222-1)

|  |  |
| --- | --- |
| **Actors** | System |
| **Pre-conditions** | Infotainment Network is active.  Client currently browsing a Parent List. |
| **Scenario Description** | The client requests the root list of the current parent list. For example, the user is browsing Folder\_A which is a Parent in the Root List and wishes to view the content list of the Root list. |
| **Post-conditions** | Server responds back with the Root List of the current Parent list. |
| **List of Exception Use Cases** | N/A |
| **Interfaces** | G-HMI, Vehicle System Interface |

### White Box View

#### White Box Scenarios

##### LBP-SD-REQ-019727/A-Traversing up the hieracy from parent list to root list (TcSE ROIN-39810-1)

**Linked Elements**

LBP-UC-REQ-019774/A-Traversing up the hierarchy from Parent to Root (TcSE ROIN-30451-1)

Sequence Diagram



## LBP-FUN-REQ-019728/A-Browsing down a list (TcSE ROIN-293828-1)

### Use Cases

#### LBP-UC-REQ-019729/A-Browsing down a List (TcSE ROIN-292223-1)

|  |  |
| --- | --- |
| **Actors** | System |
| **Pre-conditions** | Infotainment Network is active.  Client is currently browsing a list. |
| **Scenario Description** | The client is browsing a list and requests the next index of entries. |
| **Post-conditions** | Server responds back with requested index of entries. |
| **List of Exception Use Cases** | N/A |
| **Interfaces** | G-HMI, Vehicle System Interface |

### White Box View

#### White Box Scenarios

##### LBP-SD-REQ-019730/A-Browse down a list of entries (TcSE ROIN-39811-1)

**Linked Elements**

LBP-UC-REQ-019776/A-Browsing down a list (TcSE ROIN-30463-1)

Sequence Diagram



## LBP-FUN-REQ-019731/A-Browsing up a list (TcSE ROIN-293831-1)

### Use Cases

#### LBP-UC-REQ-019732/A-Browsing up a List (TcSE ROIN-292224-1)

|  |  |
| --- | --- |
| **Actors** | System |
| **Pre-conditions** | Infotainment Network is active.  Client is currently browsing a list. |
| **Scenario Description** | The client is browsing a list and requests the previous index of entries. |
| **Post-conditions** | Server responds back with requested index of entries. |
| **List of Exception Use Cases** | N/A |
| **Interfaces** | G-HMI, Vehicle System Interface |

### White Box View

#### White Box Scenarios

##### LBP-SD-REQ-019733/A-Browse up a list of entries (TcSE ROIN-39812-1)

**Linked Elements**

LBP-UC-REQ-019778/A-Browsing up a list (TcSE ROIN-30469-1)

Sequence Diagram



## LBP-FUN-REQ-130790/A-Prefetch inidcation

### Use Cases

#### LBP-UC-REQ-129268/A-Indicate Data Prefetch State

|  |  |
| --- | --- |
| **Actors** | System |
| **Pre-conditions** | Infotainment Network is active. |
| **Scenario Description** | The client system is prefetching data from the list server. |
| **Post-conditions** | The prefetch status gest indicated to the list server. |
| **List of Exception Use Cases** | N/A |
| **Interfaces** | G-HMI, Vehicle System Interface |

### White Box View

#### White Box Scenarios

##### LBP-SD-REQ-130401/A-Indicate data prefetch

Scenarios

Normal Usage

The client is fetching list data from the server.

Constraints

Pre-condition

Network is active.

Post-condition

The prefetching status is indicated to the user.

Sequence Diagram



## LBPv5-FUN-REQ-132838/A-List Browser Icon and Structured Data

### Requirements

#### LBP-FUR-REQ-019735/A-Translations (TcSE ROIN-294523-1)

All text strings must reference the MUI database for full language support and match the current system language as selected in the System Settings menu.

#### LBP-FUR-REQ-019736/A-Get Item Request (TcSE ROIN-294524-2)

If the List Browser Server receives a getitem request of 0xFFFF then return to the parent menu. Refer to SPSS Requirment LBP-GREQ-41673-1 for further clarification.

#### LBP-FUR-REQ-019738/A-Source Name (TcSE ROIN-294526-2)

The {source name} shall be the name of the selected List Browser Server audio source.

#### LBP-FUR-REQ-019741/B-Unavailable Source (TcSE ROIN-294529-2)

If the getitem is requesting an unavailable source, the List Browser Server shall return a Command Execution Status of 0x15 (Final Result-Failure, connected environment not reacting). Refer to requirement TP-GREQ-138094-2-CES.

Note: This requirement applies to sources internal to the responding Server ID (sources contained in the request Module). No response required for Server ID which is requested outside of the module.

#### LBP-FUR-REQ-019743/A-Object State (TcSE ROIN-294531-2)

When the List Browser Server is returning a list for Media sources, the Object State must be given. For example if the USB source is active and you are browsing the media source list, Object State must be Active. Refer to requirement LBP-GREQ-177707-Object State Attribute.

#### LBP-FUR-REQ-019744/A-Response Time (TcSE ROIN-294532-2)

The List Browser Server must respond to a List Browser Client request in <100ms with the fully populated list i.e. the user should never see the 'Please Wait' while navigating through the cluster menus. The Navigation POI list (Active List0x0005) is the only exception and must have a fully populated list in < 5 seconds.

#### LBP-REQ-019753/B-Source Icons (TcSE ROIN-301569-1)

|  |  |
| --- | --- |
| **Icon Name** | **Source Icon ID** |
| Invalid | 0x00 |
| AM | 0x01 |
| CD | 0x02 |
| Sirius | 0x03 |
| Audio Video In | 0x04 |
| User device - USB | 0x05 |
| User device - phone | 0x06 |
| User device - media player | 0x07 |
| User device - SD Card | 0x08 |
| User device - WiFi | 0x09 |
| User Device- Bluetooth Audio | 0x0A |
| Line In | 0x0B |
| FM | 0x0C |
| DAB | 0x0D |
| Reserved | 0x0E…0xFD |
| Blank | 0xFE |
| Unknown Source | 0xFF |

#### LBPv2-REQ-128758/C-List Server Generic Radio 1 - Radio List Structure for Cluster HMI

- Number of Preset banks depends on configuration (e.g. Touch, Non-Touch screen).

- DAB and SDARS can only be available separately based on configuration (EU or NA)



Future protection:

- added to support station list browsing capabilities.

Future protection:

- added to support station list browsing capabilities.

#### LBPv2-REQ-140537/A-List Server Generic Radio 1 - Radio List Structure for Centerstack HMI

- Number of Preset banks depends on configuration (e.g. Touch, Non-Touch screen).

- DAB and SDARS can only be available separately based on configuration (EU or NA)



#### LBP-REQ-132690/B-List Server RadioDataService1 - EPG List Structure for Centerstack HMI



#### LBP-REQ-132691/B-List Server RadioDataService2 - Journaline List Structure for Centerstack HMI



#### LBP-REQ-192091/A-List Server RadioDataService3 - Advanced EPG List Structure for Centerstack HMI



Future protection for C519

## LBP-FUN-REQ-132987/A-New device connected

### Use Cases

#### LBP-UC-REQ-132986/A-New Device Connected

|  |  |
| --- | --- |
| **Actors** | User |
| **Pre-conditions** | Device is initially inserted / connected |
| **Scenario Description** | The list server shall build up the lists from the initially inserted / connected device. The ListBrowseServer shall indicate the initializing status to the ListBrowseClient |
| **Post-conditions** | Initializing status of the ListBrowseServer is indicated to the ListBrowseClient. |
| **List of Exception Use Cases** | E1-List Server Error : If the server is not able to build up the list structure due to any error the ListBrowseServer shall indicate this to the ListBrowseClient. |
| **Interfaces** | G-HMI, Vehicle System Interface |

### White Box View

#### White Box Scenarios

##### LBP-SD-REQ-132996/A-New devide connected

Scenarios

Normal Usage

The list server builds up the list structure of the newly connected device.

Constraints

Pre-condition

New device is connected.

Post-condition

The list server function status is transmitted to the client.

Sequence Diagram



## LBP-FUN-REQ-130789/A-Server list update

### Use Cases

#### LBP-UC-REQ-129267/A-Indicate Server List Update

|  |  |
| --- | --- |
| **Actors** | System |
| **Pre-conditions** | Infotainment Network is active. |
| **Scenario Description** | A server list gets updated with new data. |
| **Post-conditions** | Update of server list is indicated to the client. |
| **List of Exception Use Cases** | Server list is empty: If the server list is empty the list server will indicated the status to the client. |
| **Interfaces** | G-HMI, Vehicle System Interface |

### White Box View

#### White Box Scenarios

##### LBP-SD-REQ-130402/A-Indicate server list update

Scenarios

Normal Usage

A server list is getting updated.

Constraints

Pre-condition

Network is active.

Post-condition

Server list update information is indicated to the client.

Sequence Diagram



# Appendix: Reference Documents

|  |  |
| --- | --- |
| Reference # | Document Title |
| 1 |  |
| 2 |  |
| 3 |  |
| 4 |  |
| 5 |  |