Product Ready Specification

目次 *Contents*

変更履歴 Changes

Ver	Date	Change point	Author

1. 目的/Purpose

AGL 仕様書 **v1** と AGL 実装 **v9.0.2** に大きなギャップがある

There is a large gap between AGL specifications v1 and AGL implementation v9.0.2

① **agl_spec_v1（2015）**

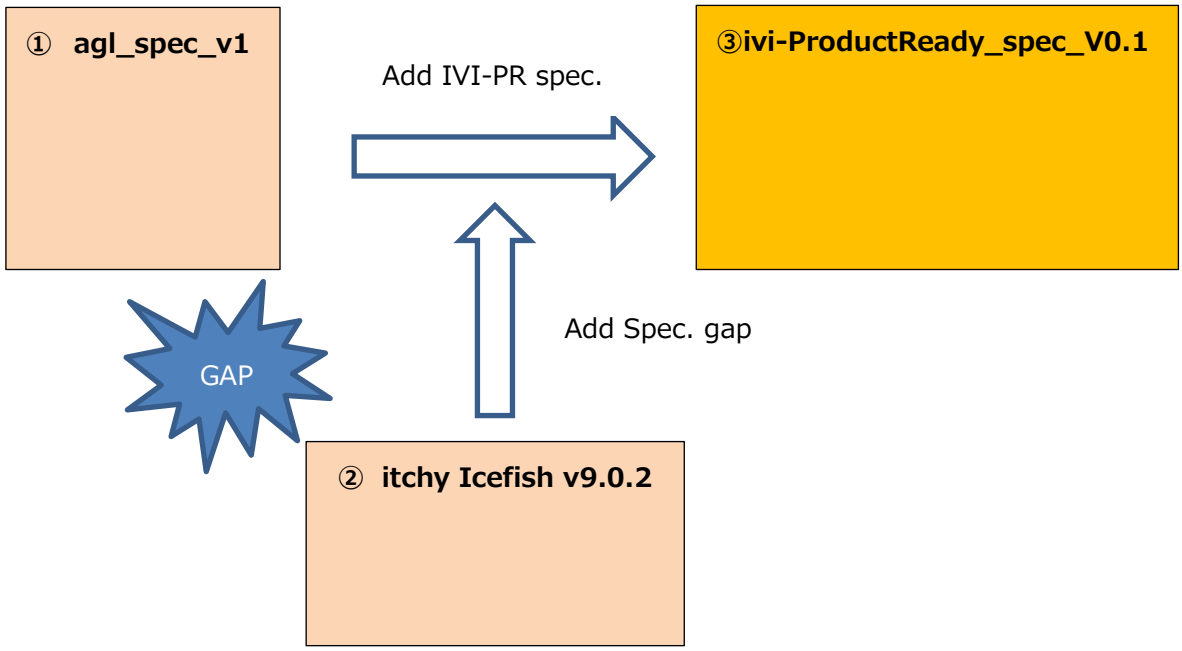
② **itchy Icefish v9.0.2（2020）**

本仕様書では ProductReady として開発する機能に関して、①、②の仕様差分と新規仕様を追記

③ ***ivi-ProductReady_spec_V0.1*** を作成

In this specification, the difference specification of ① and ② and the new specification as Product Ready are added.

③ Create ***ivi-ProductReady_spec_V0.1***



2. 適用範囲/Scope of Application

IVI-ProductReady-Profile

3. 関連文書と用語集/Related Specifications Terms and Definitions

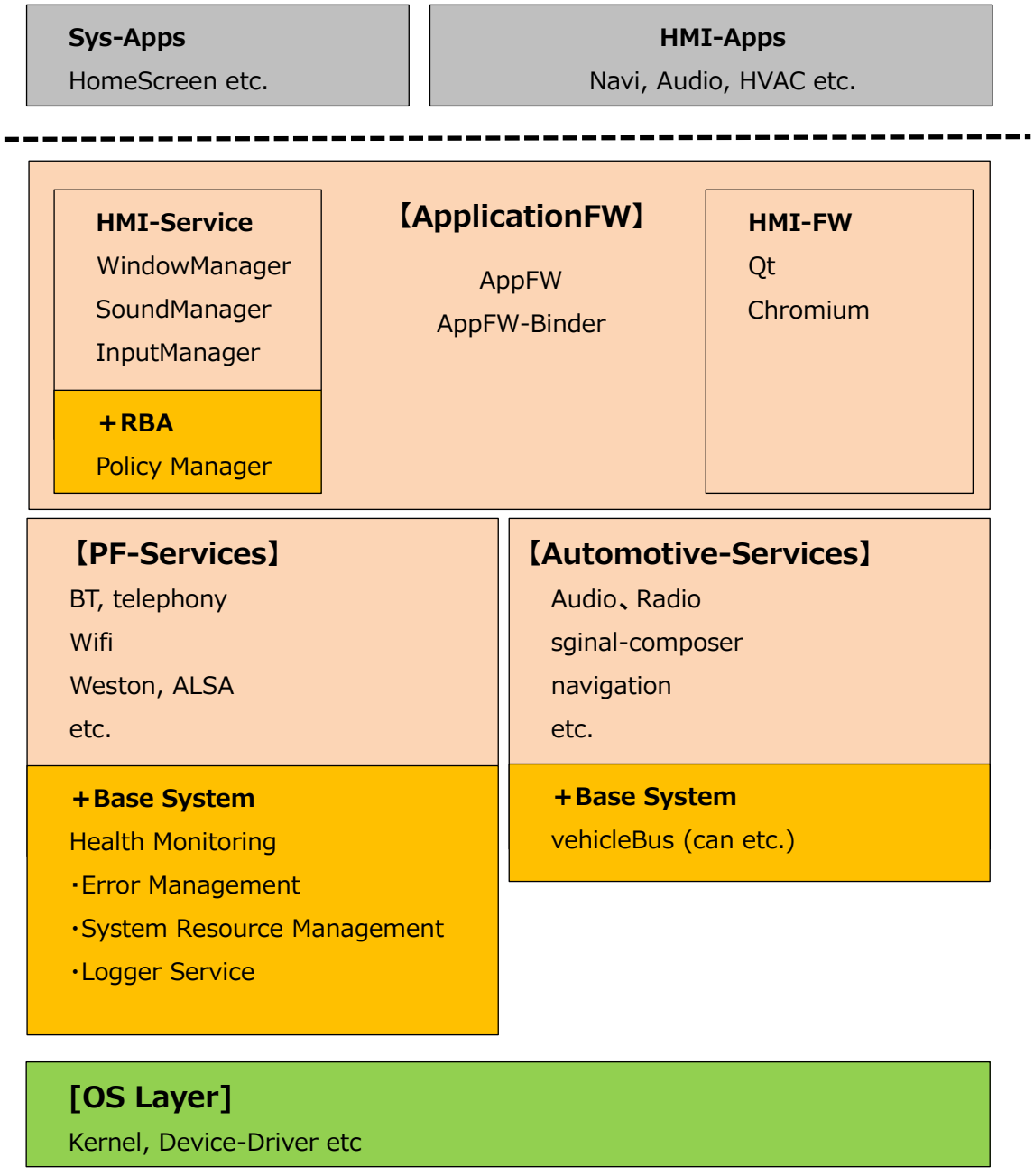
関連文書 Related Specifications

	文書名
	agl_spec_v1_280515.pdf

用語 Terms and Definitions

語句 Terms	説明 Explanation

4. 全体アーキテクチャ/Entire Architecture V0.1（Trial）



現時点で V0.1 に追加予定のパッケージ

Packages currently planned to be added to V0.1

5. App FW

5.1. App Framework

5.1.1. AGL spec and implementation

[agl_spec_v1]

No description

[itchy Icefish v9.0.2]

AGL ではアプリとサービス及び、サービスとサービスの間の仕組みとして AFW（Binder）がある。
In AGL, there is AFW (Binder) as a mechanism between applications and services and between services.

AGL のすべてのアプリとサービスは Binder を利用している。
All AGL apps and services use Binder.

[ivi-ProductReady_spec_V0.1]

IVI-PR では、BaseSystem とのサービス通信として AGLBinder と同等な機能を持つ NSFW を利用する。
参考として、サービス間通信としての機能要件を以下に示す。
IIVI-PR uses NSFW, which has the same functions as AGL Binder, as service communication with the Base System.
For reference, the functional requirements for inter-service communication are shown below.

また、NSFW の機能概要を「Appendix8.1」に添付する
In addition, an overview of NSFW functions is attached to "Appendix 8.1".

5.1.2. Inter-service communication FW

【Requirements List】

#	Term	Description
1	Pub/Sub 機能 Pub/Sub function required	Callback/イベント購読登録 /Callback/event subscription Callback/イベント購読解除 /cancel callback/event subscription サービス利用状態 /service use status アプリケーションデータ設定/取得 /application data set/get
2	Session 保持 Session holding required	セッションオープン/クローズ /session open/close セッションイベント通知 /session event notification メッセージキューオープン/クローズ /message quene open/close
3	Communication の種類 Requires Communication types	同期通信 /sync communication 非同期通信 /asynchronous communication レスポンスデータ送信 /send response data メッセージデータ受信 /receive message data 非ブロッキングメッセージ受信 /none-blocked message receive 受信メッセージ破棄 /delete received message
4	子 Thread による通信 Communication by child Thread	子スレッド起動_ループなし /start sub thread(no loop) 子スレッド起動_ループあり(子スレッド間通信) /start sub thread_has loop(sub thread communication) タイマー（コールバック） /timer(callback) タイマー（API） /Timer(API)
5	RingBuffer の操作 Operation of RingBuffer	RingBuffer の読込み/書込み /read/write ringBuffer Configuration File の読込み/書込み /read/write configuration file

#	Term	Description
6	Log の制御 Log control required	初期化 /Initialize、ログ出力 /log output 強制クリア / force clear ログ出力の制御ワード（設定・取得） /control word of log output(set・get) ログファイル操作 /logfile handle LogLevel（設定・取得） /LogLevel(Set・Get) リアルタイムログ出力切替設定 /Change to realtime log output ログイベント送信 /send log event
7	Shared Memory 操作 Shared Memory operations	共有メモリオブジェクトをオープンする(SharedMemory) / Open shared memory object(SharedMemory) 共有メモリオブジェクトのオープン状態を確認する (SharedMemory) / Check the open state of the shared memory object (SharedMemory). CNSSharedMem クラスを用いてオープンした共有メモリオブジェクトをクローズする。Close the shared memory object by using the CNSSharedMem class 共有メモリオブジェクトからデータ読み込み(SharedMemory) / Read data from shared memory object(ShardMemory) 共有メモリオブジェクトのデータをファイルへ書き出し (SharedMemory) / Write the data of the shared memory object to a file(SharedMemory)

6. PF Services

6.1. Persistent Storage

6.1.1. AGL spec and implementation

[agl_spec_v1]

No description

[itchy Icefish v9.0.2]

No implementation

[ivi-ProductReady_spec_V0.1]

IVI-PR では、不揮発領域のデータを読み出す・書き込む等、以下の機能が必要
The following functions such as reading / writing data in the non-volatile area are required.

NAND 使用 / Use NAND
CacheDRAM 使用 / Use CacheDRAM
BackupDRAM 使用 / Use BackupDRAM
リアルタイム書込 / realtime write
Backup 周期 / Backup cycle

IVI-PR では、BaseSystem が持つ不揮発性メモリへのアクセスの仕組みを追加する
VI-PR adds a mechanism for accessing the non-volatile memory of the Base System.

6.1.2. PersistentStorage

【Requirements List】

#	Term	Description
1	バックアップ領域からデータを読み出す [Read data from backup]	指定されたオフセットとデータサイズにより、ID（文字列）で指定されたバックアップ領域からデータを読み出す。 Reading data from the offset of backup area according to the ID specified by string.
2	バックアップ領域へデータを書き込む [Write data to backup]	指定されたオフセットとデータサイズにより、ID(文字列)で指定されたバックアップ領域へデータを書き込む。 Writing data to backup area from the offset according to the ID specified by string.
3	バックアップ領域への Fill [Fill data to backup]	指定されたオフセットとデータサイズにより、ID(文字列)で指定されたバックアップ領域へ指定したパターンで書込む。 Filling pattern data to backup area from the offset according to the ID specified by string.
4	バックアップ領域のサイズ取得 [Get data size of backup]	ID(文字列)で指定されたバックアップ領域のサイズを取得する。 Getting data size in backup according to the area ID specified by string.
5	バックアップデータのチェック [Check validity of backup data]	ID（文字列）で指定されたバックアップデータをチェックする。 Checking backup area data's validity according to the area ID specified by string.
6	backup_manager サービスの起動 [backup_manager service start]	backup_manager サービスを起動する。 backup_manager service start.
7	backup_manager サービスの終了 [backup_manager service stop]	backup_manager サービスを終了する。 backup_manager service stop.

#	Term	Description
8	バックアップ領域のデータを読み込み、バックアップ領域へデータを書き込む [Read and Write data to backup]	バックアップ領域のデータを読み込み、バックアップ領域へデータを書き込む Read and Write data to backup
9	バックアップ領域のデータを削除 [Delete data size of backup]	ID(文字列)で指定されたバックアップ領域のデータを削除する。 Deleting data in backup according to the area ID specified by string.

6.2. Health Monitoring

6.2.1. AGL spec and implementation

[agl_spec_v1]

5.1.5 Health Monitoring
Platform monitoring service such as watchdog or active monitoring

Spec v1 では車両信頼性を確保するための「Health Monitoring」を要件としているが、具体的な要件がない。
Spec v1 requires “Health Monitoring” to ensure vehicle reliability, but there is no specific requirement.

[itchy Icefish v9.0.2]

車両信頼性に対応する機能として、“HeartBeat”のみ実装されている
"HeartBeat" is implemented as a function that supports vehicle reliability.

[ivi-ProductReady_spec_V0.1]

IVI-PR では以下のような信頼性要件を IVI に追加する
IVI-PR adds the following reliability requirements to IVI

- サービスの異常検知と異常処理の実行
- CPU やメモリの監視
- 異常時のログ保存機能、周期ログの実行

- Service error detection and error handling can be executed
- CPU and memory monitoring
- Log saving function at the time of abnormality, periodic log can be executed

6.2.2. System start/stop、and Error Management

【Requirements List】

#	Term	Description
1	システム起動 [System start]	Configuration ファイルに設定された順序に従って他の常駐型サービスを起動 Start other resident services according to the order set in the configuration file
2	システム終了 [System shutdown]	Configuration ファイルに設定された順序に従って他のサービスを終了 Terminate other services according to the order set in the configuration file
3	異常検出と異常処理 [Anomaly detection and Abnormal processing]	異常検知［Anomaly detection］ ・サービスの生存確認による異常検出［Abnormality detection by confirmation of survival］ ・サービス異常終了［Services abnormally shutdown］ ・ステムメモリ枯渇の検知[System memory shortage] 異常処理［Abnormal processing］ ・システムリセット［System-Reset］ ・プロセス再起動［Process-Restart］ ・異常ログ出力［Abnormal log output］ ・Rob ログ保存［Store Rob(*1）log］ その他　［Others］ ・モデル切り替え［Change model］ ・電源状態遷移［Power State Transition](*2)

(*1）Rob（Record of Behavior）
ドライバーに違和感を覚えさせる制御を行った状況などの記録
Own system sound state, record status such as controlled let the driver feel uncomfortable.

(*2）独自の電源マイコンとの連携が必要なため、スタブで対応します。
Since it is necessary to link with a unique power supply microcomputer, stubs

are used.

・停止 Stop

・乗車中 Pre-Boot

・駐乗車起動 Background-Boot

・通常起動 Normal-Boot

【Implementation Example】

(1) システム起動 [System start]

system_manager は起動すると、Configuration ファイルに設定された順序に従って他の常駐型サービスを起動する。サービスはグループ化することができ、system_manager はこのグループ単位でサービスを起動する。

When system_manager is started, it starts the other resident services in line with the order in Configuration file. The services can be grouped and system_manager starts the services per the group.

ただし、AGL では Linux 標準の Sysytemd を採用しているため、本機能は BaseSystem 内のサービス起動にのみ使用する。

However, since AGL uses the Linux standard Systemd, this function is used only for starting services within the BaseSytem.

(2) システム終了 [System shutdown]

system_manager は、Configuration ファイルに設定された順序に従って他のサービスを終了する。システム終了は、Power Supply Sub-System（ハードウェアユニット）からの要求、サービスからの要求、及び異常検出等に基づいて行う。

system_manager terminates the other services in line with the order in Configuration file. The system shutdown is processed based on Power Supply Sub-System requests, the service requests and malfunction detections etc.

(3) 異常検出 [Malfunction detections]

system_manager は各種異常を検出すると、その異常を通知し、種々の異常処理を行う。異常処理の内容は、予め静的に Configuration ファイルとして規定する。規 定する種類としては、システムリセット、再起動（事象発生のプロセス）、発生事象の履歴保存等である。

When system_manager detects every kinds of the malfunctions, it executes the various failure processing. The contents of the failure processing are statically

prescribed in Configuration file in advance.　The prescribed ones are the system reset, the restart (event occurrence processes) and the occurred event log storing.

A)　Heart Beat

管理対象サービスとの間で定期的に通信（Heart Beat リクエスト）することによって生存確認を行う。サービスの応答がないと異常と判断し、Configuration ファイルの設定に従い、サービスの再起動、終了、あるいはシステムリセットを行う。

system_manager confirms the behaviors by the regular communication (Heart Beat request) to the controlled services.　When it has no responses from the services, system_manager recognizes the malfunctions and restarts and terminates the services or resets the systems in line with the set-up in Configuration file.

B)　サービス異常終了 [Services abnormally shutdown]

起動したプロセスの状態を監視する。異常終了を検知すると、Configuration ファイルの設定に従い、サービスの再起動、終了、あるいはシステムリセットを行う。

system_manager monitors the process status after start.　When it detects the abnormal shutdowns, it restarts and terminates the services or resets the systems in line with the set-up in Configuration file.

C)　システムメモリ枯渇 [System memory shortage]

resource_manager と連携してシステムメモリを監視する。一定時間規定の閾値を下回るとシステムメモリ不足と判定し、システムリセットを行う。

system_manager monitors the system memory in cooperation with resource_manager.　When the value gets lower than the certain level for the specified time, it recognizes the system memory shortage and resets the system.

(4)　異常ログ出力 [Abnormal log output]

system_manager は、サービスに対して異常ログ出力機能を提供する。アプリケーションからの要求に応じて、保存対象のログを USB ストレージデバイス、SD カード等に保存する。

system_manager provides the function to output the abnormal log for the services.　It stores the targeted log in USB storage device and SD card etc due to the requests from the applications.

(5)　モデル切り替え [Change model]

System_manager はモデル仕様依存処理を config の inithook に実装し、libsmconf.so を置き換えることによりモデル切り替えを実現

system_manager implements model specifications-dependent processing in inithook of config and realizes a model change by rearranging libsmconf.so.

(6) 電源状態遷移 [Power State Transition]

電源状態に応じてシステムの動作を制御するため、power_service から電源状態遷移要求を受信し、管理下にある常駐型サービスに通知する。

In order to control the system according to the power state, notify the managed resident service of a power state transition received from power_service.

(7) Rob ログ保存 [Store Rob log]

システム再起動時に RoB ログを保存する機能を呼び出してリセット履歴を保存する。

Call the function to save RoB log at system restart and save reset history.

6.2.3. System Resource Management

【Requirements List】

#	Term	Description
1	CPU 負荷の監視 [CPU load monitoring]	高負荷状態が一定時間異常継続している状態に至ると、CPU を占有する上位プロセスの LOGGING を行う。 execute LOGGING of the upper level process occupying CPU when the high load status is abnormally continued during the specified time.
2	システム情報の取得 [System information provision]	残メモリ情報、NAND フラッシュへの書き込み可否状態、ネットワークデバイスの通信量 Obtain the residual memory information, write access status to NAND flash and communication volume in the network device
3	システムメモリの監視 [System memory monitoring]	残メモリ量が閾値を下まわると、EVENT を発行 Issue an event when the residual memory falls below the certain level.

4	ウォッチドックタイマ(WDT)の更新 [Watch Doc Timer (WDT) Update]	マイコン内蔵の WDT により、ソフトウェアの暴走検出を行う。 Monitor the error detection for the software by WDT including microcomputer.
5	デバッグ表示情報[Debug display information provision]	デバッグ表示するための残メモリ情報、CPU 負荷情報、CMA の残メモリ情報を提 Provide the residual memory information, CPU load information and residual memory information for CMA
6	ACC-OFF 時の最小メモリ残量情報のログ出力 [Log output of the minimum residual memory information at ACC-OFF]	起動以降の最小メモリ残量情報を保持し、ACC-OFF を検出したときログ出力する。 Maintain the minimum residual memory information after the start and output the log at ACC-OFF detection.

【Implementation Example】

(1) CPU 負荷の監視 [CPU load monitoring]

CPU 負荷を監視し、高負荷状態が一定時間異常継続している状態に至ると、CPU を占有する上位プロセスの LOGGING を行う。

Monitor CPU load and execute LOGGING of the upper level process occupying CPU when the high load status is abnormally continued during the specified time.

(2) システム情報の取得 [System information provision]

残メモリ情報、NAND フラッシュへの書き込み可否状態、電源を入れてからのネットワークデバイスの通信量を取得する。

Obtain the residual memory information, write access status to NAND flash and communication volume in the network device after the power on.

(3) システムメモリの監視 [System memory monitoring]

システム全体のメモリを監視する。残メモリ量が閾値を下まわると、EVENT を発行する。

Monitor the memory in the whole system. Issue an event when the residual memory falls below the certain level.

(4) ウォッチドックタイマ(WDT)の更新 [Watch Doc Timer (WDT) Update]

MM マイコン内蔵の WDT により、ソフトウェアの暴走検出を行う。

FIFO および RR プロセスが、長時間 CPU を占有する状況が継続すると、異常と判断し WDT を発火させる。

Monitor the error detection for the software by WDT including MM microcomputer.

Recognizes the malfunction and triggers WDT when FIFO process and RR process keep occupying CPU for the long time.

(5) デバッグ表示情報の提供 [Debug display information provision]

デバッグ表示するための残メモリ情報、CPU 負荷情報、CMA の残メモリ情報を提供する。

これらは、ダイアグ画面からログ出力を Release から Debug に切り替えることにより LOG に出力される。

Provide the residual memory information, CPU load information and residual memory information for CMA for the debug display.

These are outputted to LOG by switching the log output to Debug from Release at the Diagnosis screen.

(6) ACC-OFF 時の最小メモリ残量情報のログ出力 [Log output of the minimum residual memory information at ACC-OFF]

起動以降の最小メモリ残量情報を保持し、ACC-OFF を検出したときログ出力する。

Maintain the minimum residual memory information after the start and output the log at ACC-OFF detection.

6.2.4. Logger Service

【Requirements List】

#	Term	Description
1	ログ保存要求機能 [Function to request the log storing]	不揮発メモリに保存する Save to non-volatile area (eMMC/USB/SD etc.)
2	周期読み込み機能 [Function to periodically read]	設定に従ってログの周期読み込みを行う Periodically read the log according to the settings

【Implementation Example】

(1) ログ保存要求機能 [Function to request the log storing]

他モジュールで作成したログを、保存要求受信時に読み込んで保存する。

At the receipt of the storing request read the log generated by other modules and save it.

(2) 周期読み込み機能 [Function to periodically read]

他モジュールで書き込まれたログを周期的に読み込む。

Periodically read the log written by other modules.

7. Automotive Services

7.1. Vehicle Bus

7.1.1. AGL spec and implementation

[agl_spec_v1]

7.5.5 Automotive Devices

Spec v1 では、車両で利用するデバイスを要件としているが、具体的な要件はない。

Spec v1 requires devices to be used in vehicles, but there are no specific requirements

[itchy Icefish v9.0.2]

AGL では通信デバイス（CAN、Positioning）に対して汎用的な実装を追加している。

AGL adds a general implementation for communication devices (CAN, Positioning).

(e.g.) AGL-Service CAN、Signal-Composer etc.

[ivi-ProductReady_spec_V0.1]

IVI-PR では通信デバイスに以下のような要件を追加する

IVI-PR adds the following reliability requirements

- 通信デバイスからの詳細情報取得
- 通信デバイスの通信途絶監視・復帰
- Acquisition of detailed information from communication devices
- Communication blackout monitoring / recovery of communication devices

7.1.2. CAN

【Requirements List】

#	Term	Description
1	CAN を利用した車両データの送受信 Transmission and reception of vehicle data using CAN	CAN デバイスとのデータ送受信 Data transmission/reception with CAN device
2	通信途絶からの復帰等の信頼性機能 Reliability functions such as recovery from communication interruption	CAN データ通信途絶・復帰を監視する Check the can data communication suspension or resume

【Implementation Example】

(1)can_hal へ CAN データを送信する。
Send CAN data to can_hal.

(2)can_hal から CAN データを受信する。
Receive CAN data from can_hal.

(3)CAN データ通信途絶・復帰を監視する。
Check the can data communication suspension or resume.

(4)CAN コマンド制御処理。
Transmission control of CAN command.

(5)CAN サービスの Availability を通知する。
Notifies the CAN service availability.

(6)送信した CAN データを配送する(エコーバック)
Distrubute CAN data transmitted by AVN.

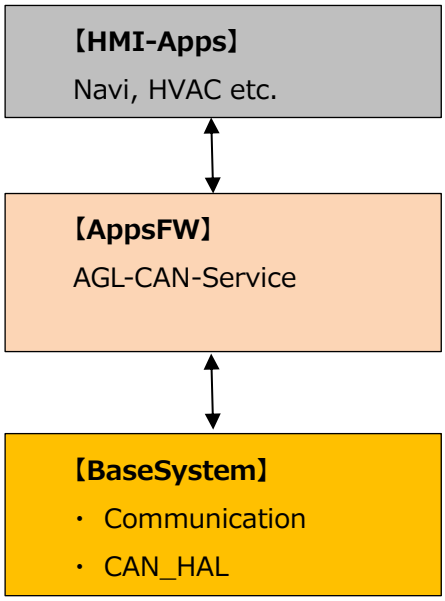


Fig. CAN implementation example

7.1.3. Positioning

【Requirements List】

#	Term	Description
1	GPS デバイスの情報 GPS Device management	緯度経度、高度、速度(*1)、方位情報が必要 longitude and latitude, altitude, speed, heading data
2	センサデバイスの情報 SensorDevice(*2) management	ジャイロスコープ、加速度計、車速パルス、車両リバースが必要 Gyroscope, Accelerometer, Speed pulse, Vehicle Reverse

(*1)(*2)

AGL 標準 HW にデバイスがない場合、スタブで対応します。

If there is no device in the AGL standard HW, it will be supported by a stub.

【Implementation Example】

(1) GPS

・ GPS 情報を取得する

Get GPS information

・ GPS のリセットを要求する

Request GPS reset

・ GPS 時刻を提供する

Provide GPS time

(2) Sensor

・ Sensor データを提供する

Provide Sensor data

8. Appendix

8.1. NSFW

8.1.1. NSFW

NSFW は以下の機能を提供している。

1. HANDLE：インスタンスの内部構造を隠ぺいするために用いる型が提供。
2. Dispatcher：メッセージ受信とコールバック関数登録の仕組みが提供。
3. 1 対 1 のプロセス間通信：任意のプロセスと通信を行う仕組みを提供。
4. 1 対多のプロセス間通信：多数のプロセスと通信を行う仕組みを提供。
5. Session：通信したいサービスとのセッション(通信経路)を確立する仕組みを提供。
6. Abnormal Monitor ：framework_unified における異常監視する機能を提供。

NSFW provide functions as follows.

1.HANDLER:provide variable type to hide the internal construction of instance
2.Dispatcher:provide mechanism for receiving message and registering callback function
3.one-to-one communication between process: provide mechanism for communicating with any process
4.one-to-many communication between process: proved mechanism for communicating with many process
5.Session: provide mechanism for build session with communication target
6.Abnormal Monitor: provide mechanism of abnormality monitoring

8.1.2. ユースケースと API 一覧 [Use case and API lists]

setup

#	Term	Description
1	Callback/イベント購読登録 /Callback/event subscription	Callback 関数を登録しておくことにより、対応するメッセージ／イベントを受信すると、Dispatcher が Callback 関数を実行する If callback function is registered,when receiving the correspond message and event,Dispatcher willexecute the callback function
2	Callback/イベント購読解除 /cancel callback/event subscription	Callback 関数の登録解除を実施すると、Dispatcher は同 Callback 関数を実行しない。 If cancel the callback function register,Dispatcher will not execute the same callback function.
3	サービス利用状態 /service use status	サービスは機能提供の可否状態をクライアントに通知する責務を持つ。 Service should notify client whether function can be used ot not.
4	アプリケーションデータ設定/取得 /application data set/get	アプリケーションデータは Dispatcher にて保持され、任意のタイミングで設定／取得可能である。 Application data is maintained by Dispatcher and be set/get at any time.
5	DeferMessage /DeferMessage	DeferMessage により、受信済のメッセージを任意のタイミングで再度受信させることができる。 The Message that has been received can be received again by DeferMessage.

Session

	Term	Description
1	セッションオープン/クローズ /session open/close	本処理はクライアント/サービス間のセッションを確立する機能を提供する。クライアントからサービスへセッションのオープンを要求し、サービスが生成したセッションをクライアントへ返す。クライアントは取得したセッションを利用してサービスとの通信を行う。セッションが不要となった際は、クライアントからサービスへセッションのクローズを要求し、サービスがセッションを破棄する。 This process provides function to build client/server session.Client send session open request to service,then service creates session and return it to client.Client use this session to communicate with service.If Client does not want to use session any more,send session close request to service,then service will delete the session.
2	セッションイベント通知 /session event notification	本処理は、確立されたセッション上にて Publish-Subscribe 方式の通信を行う機能を提供する。サービスは、Publish されたコマンドを Subscribe されたクライアントへ配送する。 Provide communication in Publish-Subscribe way between build session.Service sends Publish command to Subscribe client.
3	メッセージキューオープン/クローズ /message quene open/close	メッセージの送受信を行うために必要なメッセージキューのオープン、クローズのためのシーケンスを以下に記載する。送信、受信、同期通信等の用途に応じてコールする API を切り分ける必要がある。 Sequence of message queue open/close, which is needed when send and receive message.The call API should be spared correspond to send,receive, sync communication etc.

communication

#	Term	Description
1	同期通信 /sync communication	同期メッセージの送信による同期通信を行う Sequence of sync communication by sending sync message
2	非同期通信 /asynchronous communication	非同期メッセージの送信は、クライアントから XXXSendMsg(serviceName,CMD,prm,session)を呼び出すことで実現する。 Call XXXSendMsg(serviceName,CMD,prm,session) ny client to send asynchronous message.
3	レスポンスデータ送信 /send response data	同期通信、非同期通信におけるレスポンス応答 response data by sync and asynchronous communication way.
4	メッセージデータ受信 /receive message data	メッセージを受信した際に添付されている受信データを取得するためのシーケンスを以下に記載する。 get received data which is attached when receiving message.
5	非ブロッキングメッセージ受信 /none-blocked message receive	メッセージを受信する際に非ブロッキング形式で受信を行う receiving message at none-blocked way
6	受信メッセージ破棄 /delete received message	メッセージキューに蓄積されているメッセージ情報を破棄する deleting received message that is saved in message queue

Thread,Timer

#	Term	Description
1	子スレッド起動_ループなし /start sub thread(no loop)	メインループを持たない子スレッドの起動。メインループが必要な場合は自作および次項の子スレッド起動_ループありにて実現すること starting sub thread that has no main loop.If main loop is need,implement it with self made and next sequence [start sub thread_has loop]
2	子スレッド起動_ループあり (子スレッド間通信) /start sub thread_has loop(sub thread communication)	メインループを持つ子スレッドの起動、および子スレッド、親スレッド間の通信 sub thread(has mainloop) start,communication between sub thread and parent thread.
3	タイマー（コールバック） /timer(callback)	本処理はコールバック関数の登録とタイマーの開始処理が一体となったタイマーの機能を提供する。タイムアウトした際に対象 API で登録したコールバック関数が呼ばれる。 Provide function to timer which combines callback function register and timer start process.When timeout happens,call the registered API object.
4	タイマー（API） /Timer(API)	本処理はタイマーの基本的な機能を提供する。本処理を使用するには事前にコールバック関数を登録する必要があり、タイムアウトした際に対象 API で登録したコールバック関数が呼ばれる。 Provide basic function of timer handle.Before using this API,callback function should be registered.When timeout happens,the registered API object will be called
5	タイマー（NSTimer クラス） /timer(NSTimer class)	本処理はクラスを用いたタイマーの機能を提供する。本処理を使用するには事前にコールバック関数を登録する必要があり、タイムアウトした際に対象 API で登録したコールバック関数が呼ばれる。 Provide function of Timer class.Before using this API,callback function should be registered.When timeout happens,the registered API object will be called

Ringbuffer,configuration

#	Term	Description
1	RingBuffer の読込み/書込み /read/write ringBuffer	RingBuffer の読み込み／書き込み。本は、書き込み完了および読み込み完了をメッセージ通信にて通知するケースである。 ringbuffer read/write.This term described the case of notifying the completion pf reading and writing by message.
2	Configuration File の読込み/書込み /read/write configuration file	Configuration は、設定ファイルの読み込み及び更新機能を提供する。Configuration には、以下の機能用 API が提供されている。 1.コンフィギュレーションファイルの解析 2.コンフィギュレーションファイルから値の読込み 3.コンフィギュレーションファイルへの値の書込み Configuration provides function of reading configurationfile and update function.Functions of Configuration are as follows. 1.analysis configuration file 2.read value from configuration file 3.write value to configuration file

Log

#	Term	Description
1	初期化 /Initialize	アプリ起動時に XXX_SET_ZONES()をコールすることでログ機能の初期化を行う。
2	ログ出力 /log output	ログを出力する際には PASLOG()もしくは NsLogData()を使用する。 Use XXXLOG() or NsLogData() when output log.
3	強制クリア / force clear	ログの設定やメモリ上のログをクリアする際に使用する。 Clear log configuration or log that be saved.
4	ログ出力の制御ワード（設定・取得） /control word of log output(set・get)	シーケンスの順番でコールすることは必須ではない。制御ワードを設定、取得する際に使用する API である。 Do not need call the APIs by the order as the sequence describes.This API is used to set/get control word
5	ログファイル操作 /logfile handle	ログファイルの数や名前を設定する際に使用する。 Set log file number and name.
6	LogLevel（設定・取得） /LogLevel(Set・Get)	ログレベルを切り替える際に使用する。 Change log level.
7	リアルタイムログ出力切替設定 /Change to realtime log output	リアルタイムログの出力を切り替える際に使用する。 Change to realtime log output.
8	ログイベント送信 /send log event	本処理はログイベントを送信する機能を提供する。本 API がコールされると登録されているプリケーションに「SS_MSG_EVTLOG」、「SS_MSG_LOGGERCNT」イベントを通知する。 This API provides function to send log event.And notify「SS_MSG_EVTLOG」、「SS_MSG_LOGGERCNT」event of Application who calls and register the API

Shared Memrory（CNSSharedMemReader）

#	Term	Description
1	共有メモリオブジェクトをオープンする /Open shared memory object.	CNSSharedMemReader クラスを用いて共有メモリオブジェクトをオープンする。 Open the shared memory object using the CNSSharedMemReader class.
2	共有メモリオブジェクトのオープン状態を確認する /Check the open state of the shared memory object.	CNSSharedMemReader クラスインスタンス内の共有メモリオブジェクトのオープン状態を確認する。 Check the open state of the shared memory object in the instance of the CNSSharedMemReader class.
3	共有メモリオブジェクトをクローズする /Close shared memory object.	CNSSharedMemReader クラスを用いて共有メモリオブジェクトをクローズする。 Close the shared memory object using the CNSSharedMemReader class.
4	共有メモリオブジェクトからデータ読み込み /Read data from shared memory object.	共有メモリオブジェクトからデータを読み込む。 Read data from shared memory object.
5	共有メモリオブジェクトのデータをファイルへ書き出し /Write the data of the shared memory object to a file.	共有メモリオブジェクトのデータをファイルへ書き出す。 Write the data of the shared memory object to a file.
6	共有メモリオブジェクトのデータサイズを取得 /Get the data size of the shared memory object.	共有メモリオブジェクトのデータサイズを取得する。 Get the data size of the shared memory object.
7	共有メモリオブジェクトのデータ読み出しポインタを初期化 /Initializes the data read pointer of the shared memory object.	共有メモリオブジェクトの読み出し位置ポインタを書き込み位置ポインタと同一に初期化する。 Initializes the read position pointer of the shared memory object as same as the write position pointer.

Shared Memrory（CNSSharedMemWriter）

#	Term	Description
1	共有メモリオブジェクトをオープンする (SharedMemoryWriter) /Open shared memory object (SharedMemoryWriter).	CNSSharedMemWriter クラスを用いて共有メモリオブジェクトをオープンする。 Open the shared memory object using the CNSSharedMemWriter class.
2	共有メモリオブジェクトのオープン状態を確認する (SharedMemoryWriter) /Check the open state of the shared memory object (SharedMemoryWriter).	CNSSharedMemWriter クラスインスタンス内の共有メモリオブジェクトのオープン状態を確認する。 Check the open state of the shared memory object in the instance of the CNSSharedMemWriter class.
3	共有メモリオブジェクトをクローズする (SharedMemoryWriter) /Close shared memory object (SharedMemoryWriter	CNSSharedMemWriter クラスを用いて共有メモリオブジェクトをクローズする。 Close the shared memory object using the CNSSharedMemWriter class.
4	共有メモリオブジェクトにデータを書き込む /Write the data to shared memory object.	CNSSharedMemWriter クラスを用いて共有メモリオブジェクトにデータを書き込む。 Write the data to shared memory object using the CNSSharedMemWriter class.
5	共有メモリバッファーをクリアする /Clear the shared memory buffer.	CNSSharedMemWriter クラスを用いて共有メモリバッファーをクリアする。 Clear the shared memory buffer using CNSSharedMemWriter class.

Shared Memrory（CNSSharedMem）

#	Term	Description
1	共有メモリオブジェクトをオープンする (SharedMemory) / Open shared memory object(SharedMemory)	CNSSharedMem クラスを用いて、共有メモリオブジェクトをオープンする Open the shared memory object using the CNSSharedMem class.
2	共有メモリオブジェクトのオープン状態を確認する (SharedMemory) / Check the open state of the shared memory object (SharedMemory).	CNSSharedMem クラスインスタンス内の共有メモリオブジェクトのオープン状態を確認する。 Check the open state of the shared memory object in the instance of the CNSSharedMem class.
3	共有メモリオブジェクトをクローズする (SharedMemory) / Close shared memory object (SharedMemory).	CNSSharedMem クラスを用いてオープンした共有メモリオブジェクトをクローズする。 Close the shared memory object by using the CNSSharedMem class.
4	共有メモリオブジェクトからデータ読み込み (SharedMemory) / Read data from shared memory object(ShardMemory).	CNSSharedMem クラスのインスタンスを用いて、共有メモリオブジェクトからデータを読み込む Read the data from shared memory object by using the instance of the CNSSharedMem class.
5	共有メモリオブジェクトのデータをファイルへ書き出し (SharedMemory) / Write the data of the shared memory object to a file(SharedMemory).	CNSSharedMem クラスのインスタンスを用いて、読み込んだ共有メモリオブジェクトのデータをファイルへ書き出す。 Write the read data of the shared memory object to a file by using the instance of CNSSharedMem class.

#	Term	Description
6	共有メモリオブジェクトのデータサイズを取得 (ShraedMemory) / Get the data size of the shared memory object(SharedMemory).	CNSShraedMem クラスのインスタンスを用いて、読み込んだ共有メモリオブジェクトのデータサイズを取得する。 Get the read data size of the shared memory object by using the instance of CNSSharedMem class.
7	共有メモリオブジェクトのデータ読み出しポインタを初期化 (SharedMemory) / Initializes the data read pointer of the shared memory object(SharedMemory).	共有メモリオブジェクトの読み出し位置ポインタを書き込み位置ポインタと同一に初期化する。 Initializes the read position pointer of the shared memory object as same as the write position pointer.
8	共有メモリオブジェクトにデータを書き込む (SharedMemory) / Write the data to shared memory object(SharedMemory).	CNSSharedMem クラスを用いて共有メモリオブジェクトにデータを書き込む。 Write the data to shared memory object by using the CNSSharedMem class.
9	共有メモリバッファーをクリアする(SharedMemory) /Clear the shared memory buffer(SharedMemory).	CNSSharedMem クラスを用いて共有メモリバッファーをクリアする。 Clear the shared memory buffer by using CNSSharedMem class.

Misc

#	Term	Description
1	Mutex /Mutex	ロック機能を担う。 Lock function.
2	RWLock /RWLock	リードロック、ライトロック機能を担う。 Read lock and write lock function
3	Version /Version	バージョン情報を取得する。 Get version information.
4	HSMEvent /HSMEvent	ステートマシンを保有するサービスについて、起動・子スレッド生成・コールバック登録を行う start,create sub thread,register callback of service that has state machine
5	セッション管理(data pool) /session management(data pool)	本処理は、セッションオープン/クローズの機能をクラス化したものとして提供する。Publish-Subscribe 方式の通信については、notification_persistent_service の機能を利用する。 Provide session open/close function as a class.Publish-Subscribe communication way will use function of notification_persistent_service.
6	メイン処理 /Main process	メッセージを受信し、コールバックの実行を行う。 receive the message and execute register callback.
7	シーン別電源の状態遷移 /State transition of Situation based power management	シーン別電源のメッセージ受信(通常起動、乗車中起動、乗車中状態への遷移、駐乗車起動、駐乗車状態への遷移、 終了要求）とコールバックファンクションの呼び出しを行う。 Receive power message by scene (Normal-startup, Riding startup, Transition for riding state, Parking and riding startup, Transition for parking and riding state and Stop request) and call callback-function.