Document Title	Explanation of Application Interfaces of the HMI, Multimedia and Telematics Domain
Document Category	Explanation (EXP)
Document UID	272
Document Status	published
Part of AUTOSAR Standard	Classic Platform
Part of Standard Release	R26-11

Document Change History

Date	Release	Changed by	Description
2026-11-26	R26-11	AUTOSAR Release Management	• No content changes
2025-11-27	R25-11	AUTOSAR Release Management	• No content changes
2024-11-27	R24-11	AUTOSAR Release Management	• No content changes
2023-11-23	R23-11	AUTOSAR Release Management	• No content changes
2022-11-24	R22-11	AUTOSAR Release Management	• No content changes
2021-11-25	R21-11	AUTOSAR Release Management	• No content changes
2020-11-30	R20-11	AUTOSAR Release Management	• No content changes
2019-11-28	R19-11	AUTOSAR Release Management	• No content changes • Changed Document Status from Final to published
2018-10-31	4.4.0	AUTOSAR Release Management	• Editorial changes



△

2017-12-08	4.3.1	AUTOSAR Release Management	• Editorial changes
2016-11-30	4.3.0	AUTOSAR Release Management	• Editorial changes
2014-10-31	4.2.1	AUTOSAR Release Management	• Editorial changes
2013-03-15	4.1.1	AUTOSAR Administration	• Replaced “Complex Device Driver” by “Complex Driver”
2010-02-02	3.1.4	AUTOSAR Administration	• Initial release

Disclaimer

This work (specification and/or software implementation) and the material contained in it, as released by AUTOSAR, is for the purpose of information only. AUTOSAR and the companies that have contributed to it shall not be liable for any use of the work.

The material contained in this work is protected by copyright and other types of intellectual property rights. The commercial exploitation of the material contained in this work requires a license to such intellectual property rights.

This work may be utilized or reproduced without any modification, in any form or by any means, for informational purposes only. For any other purpose, no part of the work may be utilized or reproduced, in any form or by any means, without permission in writing from the publisher.

The work has been developed for automotive applications only. It has neither been developed, nor tested for non-automotive applications.

The word AUTOSAR and the AUTOSAR logo are registered trademarks.

Table of Contents

1 Purpose of this Document 5

2 Acronyms and abbreviations 6

3 Introduction 7

4 Basic Architecture 8
- 4.1 Application Service SWC . . . 9
- 4.2 Application Controller SWC . . . 10
- 4.3 UI Devices SWCs . . . 10
- 4.4 Example: AUTOSAR PDC application . . . 11

5 Description of Software Compositions and Components 13
- 5.1 Button Panel Component . . . 15
  - 5.1.1 Simple button interface . . . 16
  - 5.1.2 Toggle state interface . . . 16
  - 5.1.3 Multi state button interface . . . 16
  - 5.1.4 Multi state output interface . . . 17
  - 5.1.5 Boundless rotary knob interface . . . 17
- 5.2 InterDomainController Composition . . . 17

6 Glossary 19

7 References 20

1 Purpose of this Document

The purpose of this document is to provide explanations on the ports and interfaces standardized for the Multimedia (MM), Telematics (T) and Human-Machine-Interface (HMI) sub-domains.

Up to this version only some simple UI(User Interface)-Device (e.g. buttons, knobs etc.) interfaces for the HMI sub-domain have been standardized.

Chapter 3 gives a brief introduction to the structure and characteristics of the MM/T/HMI domain. Besides that high-level requirements towards the MM/T/HMI are described in the same chapter. Chapter 4 gives an overview of the domain architecture for the MM/T/HMI domain. In chapter 5 the standardized MM/T/HMI ports and interfaces are explained.

2 Acronyms and abbreviations

Abbreviation / Acronym:	Description:
HMI	Human Machine Interface
LED	Light Emitting Diode
MM/T/HMI	Multimedia / Telematics / Human Machine Interface
PDC	Park Distance Control
SWC	Software Component
UI	User Interface

Table 2.1: Acronyms and abbreviations used in the scope of this Document

3 Introduction

The MM/T/HMI domain is complex and fairly large, with many interactions between SW components of different types fulfilling various purposes. The MM/T/HMI domain includes SW components of different application types, such as Multimedia, Telematics or HMI applications. Moreover, some SW components can be shared by two or more application types. For example, the video display and its associate SW components are shared by Multimedia and HMI applications. The HMI domain by itself is also complex and, is designed and handled differently by car OEM's.

Approaches and requirements for the HMI are heterogeneous. Some systems are very simple, while others are very complex. Some system designs desire to maintain full control of the HMI logic. Other system designs require a generic approach where the bulk of the HMI core software is easily reusable across systems (i.e.: different car models). The later approach requires powerful HMI core software that is unaware of the HMI design flow and which is instructed by other components to execute the HMI design flow. Of course, both approaches are valid and should be covered by the suggested architecture. Each approach calls for different type of HMI application interfaces. The domain complexity as well as the different views and requirements on the domain architecture calls for a well defined layered and flexible architecture approach. The aim of the architecture is to provide access to most demanded functionalities and allow different architecture designs for the HMI without imposing a particular design.

The architecture aims to:

- Separate "functional core" world from HMI world
- Separate HMI logic from HMI presentation
- Centralize the management of HMI resources (e.g.: audio and display)
- Abstract the physical devices (e.g. Keyboard)

4 Basic Architecture

This section describes the architecture of a simple system and the terminology used to differentiate different types of application components (SWC).

The basic principle for the architecture is **separation of HMI parts from non-HMI parts** and of OEM-specific parts from non-OEM-specific parts.

A typical architecture includes the following types of SWCs as depicted in the overview shown next.

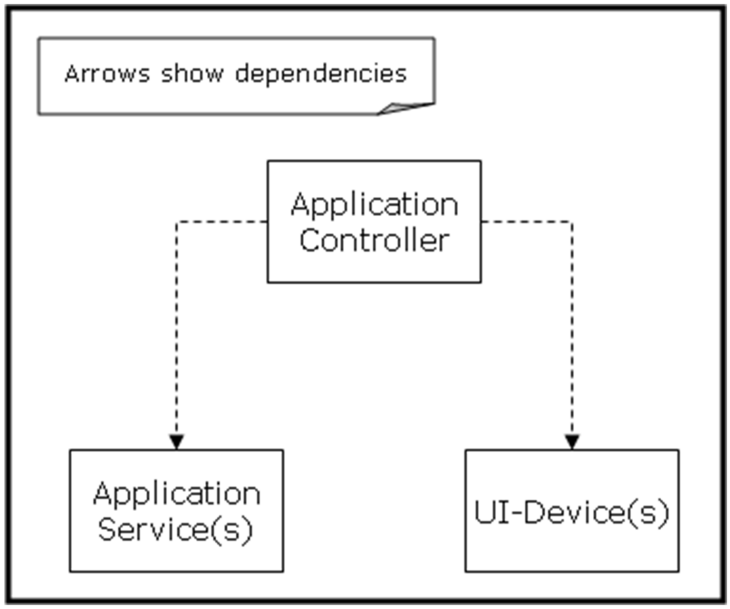


Figure 4.1: Three SWC types and their dependencies

The three types of SWCs have the following responsibilities:

1. **Application Service SWC**

   Implementation of the main functionality, e.g. PDC (Park Distance Control), CD/-Media Player or mirror adjustment. This functionality is independent of the HMI (Look & Feel)

2. **UI-Device SWC**

   Representation of all User Interface (UI) devices (input and output devices). The UI-Device definition should be unaware of the functionality it is connected to. A button SWC switching the PDC functionality on/off should, for example, **only** deliver pressed/not pressed information, **not** PDC on/off information (for the button to be reusable).Examples of input devices are buttons, joysticks, microphones, touch screens.Examples of output devices are screens, LEDs, buzzers, vibration devices.

3. **Application Controller**

The Application Controller maps the Application Services to UI-Devices. It is responsible for both the behavior (HMI logic) and the connection to the UI-Devices (HMI). This component will in most cases be highly OEM specific.

This architecture enables the reuse of the Application Services in different HMI scenarios (e.g. HMIs with or without displays, speech recognition etc.). This way the Application Services are independent of the HMI.

The encapsulation of the HMI- and OEM-specific aspects in the Application Controller allows the reuse of both the Application Services and the UI-Devices – only the Application Controller needs to be changed to give the desired behavior.

Data Flow

The following overview shows the data flow for Application Service / Application Controller / UI-Device system. Note that there should *never* be a data flow directly between the Application Service and the UI-Device, since a basic principle is that the UI-Device should be generic and not know anything about the Application Service.

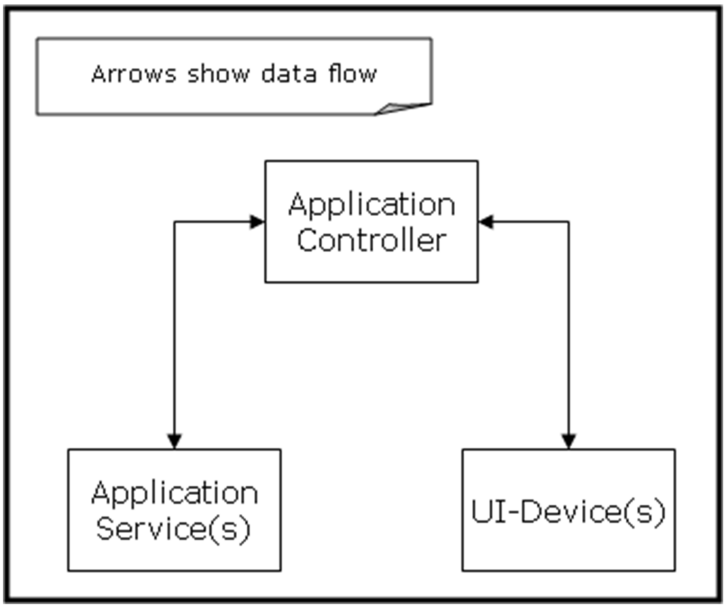


Figure 4.2: Data flow between the three SWC types

4.1 Application Service SWC

Application services are the SWCs implementing the functionality of an HMI related application. There are a lot of examples for HMI related applications:

- Park distance control application
- AM/FM tuner application

- Navigation application
- Climate control application

Application services do not have to be specific to the MM/T/HMI domain, but can also be derived from any other domain interacting with the user, e.g. driving dynamics functionality. The ports and interfaces of some of these applications are standardized within AUTOSAR, others maybe standardized in different Standardization Groups, e.g. the MOST cooperation.

The HMI basic architecture defined in this document has to be able to cope with this requirement. Therefore the application controller SWC type is introduced.

4.2 Application Controller SWC

The Application Controller can be seen as the HMI specific wrapper of the Application Services. The HMI specific parts of the application realized by the Application Controller include the following aspects:

- Decide which data provided by the Application Service has to be presented to the user in which way (audio, video, etc.)
- Interpret the user input and control the functionality of the Application Service
- Decide at which point in time the Application Service wants to be active (“request for the focus”)

Each OEM will define its specific HMI behavior (especially for each vehicle variant). Therefore the Application Controller will be different for each HMI system.

4.3 UI Devices SWCs

UI Device SWCs provide input from and output to the user. The components are the access points to the input and output hardware devices. Such devices are for instance buttons, joysticks, microphones, screens, LEDs, buzzers or others.

Please note there are some domain specific UI devices which are not handled by this WP like the accelerator pedal. These domain specific UI devices are strongly coupled to the specific application service and are hardly reusable.

Input UI Device SWCs retrieve the input signals via the I/O Abstraction Layer or Complex Driver from the input hardware. They then convert the received input signals and provide them via standardized application interfaces to other (potentially remote) SWCs.

Output UI Device SWCs are used to provide output to the user via output hardware devices. The UI Device SWC converts the output data received from other SWCs and passes it via the I/O Abstraction Layer to the output hardware device.

4.4 Example: AUTOSAR PDC application

This section explains the usage of UI device interfaces based on a Park Distance Control (PDC) application example. This is only an example and is **not related** in anyway to the actually standardized AUTOSAR PDC ports and interfaces which are explained in the "Explanation of Application Interfaces of the Body and Comfort Domain" document [1].

Based on the MM/T/HMI architecture introduced in the previous section the example PDC functionality is realized using the following SWCs:

1. PDC application service SWC
2. PDC application controller SWC
3. PDC Button SWC
4. PDC LED SWC

The PDC application service provides the basic functionality of the PDC (handle sensors, compute distance to obstacles, etc.). The main purpose of the PDC Controller is to realize the communication between the application specific PDC application service SWC and the PDC application service agnostic UI devices.

The PDC application controller uses two interfaces to communicate with the PDC application service.

- PDC Service Interface
- PDC Status Interface

These interfaces are provided by the PDC domain experts and must be independent from the user interface provided to the user.

In order to keep the example simple the distance feedback to the user is being omitted in this example. The example PDC application controller only uses two channels to communicate with the user interface.

- A simple hard button to switch the PDC functionality on or off.
  (PDC Button SWC)
- A LED to give feedback to the driver if the PDC functionality is switched on or off.
  (PDC LED SWC)

Figure 4.3 depicts the specification of the PDC functionality with introduced SWCs. The Simple Button Interface realizes the communication channel for the PDC hard button. The data field "pressed" uses a one bit encoding to express if the button was pressed / released. The Toggle State Interface realizes the LED feedback communication channel. The data field "active" uses a one bit encoding to express if the LED is active / inactive. Both UI devices are independent from the PDC functionality.

Both interfaces use the AUTOSAR sender/receiver communication paradigm.

The Simple Button Interface and the Toggle State Interface could easily be connected to other application controllers without having to change the UI device interfaces.

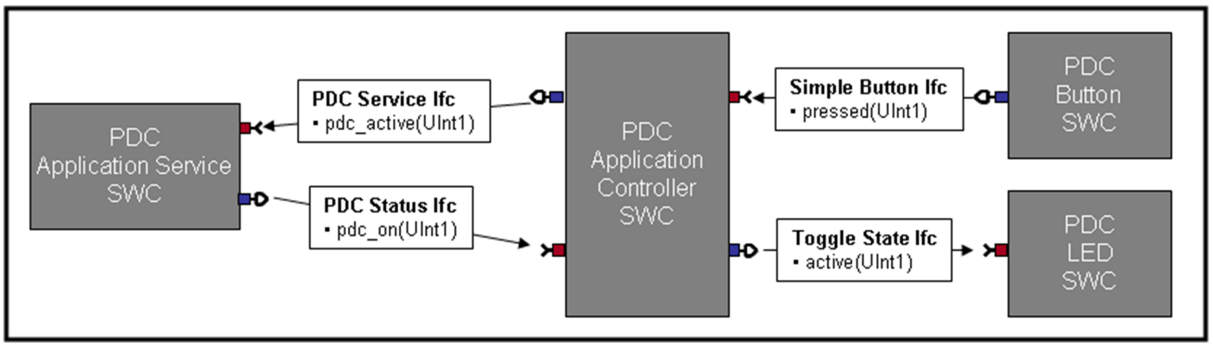


Figure 4.3: Example PDC application using UI device interfaces

5 Description of Software Compositions and Components

This chapter introduces the parts of the basic architecture that are standardized within AUTOSAR.

Figure 5.1 depicts which parts of the HMI basic architecture are defined by AUTOSAR. The interfaces and ports of some of the application services are standardized within AUTOSAR. The Application Controllers are OEM specific and are therefore not standardized within AUTOSAR. Some of the UI Device interfaces will be standardized within AUTOSAR.

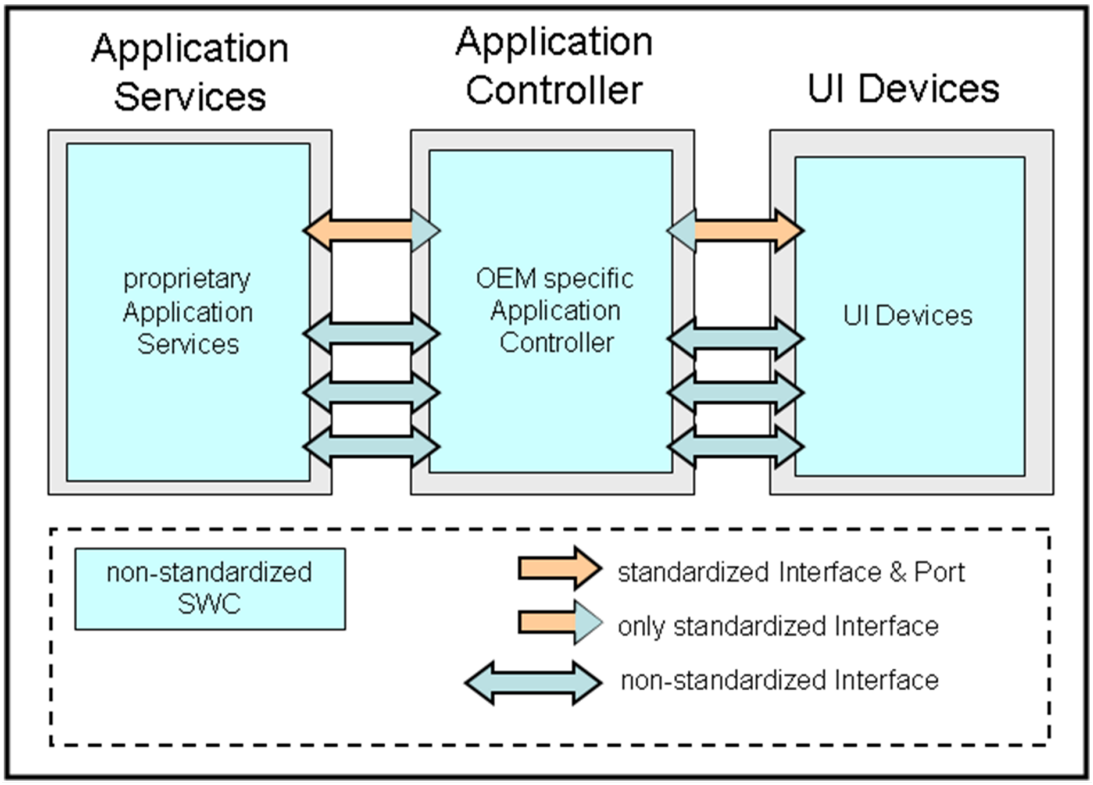


Figure 5.1: Overview of the proposed HMI architecture and the standardized components

Figure 5.2 shows an example of a real-world system using the standardized AUTOSAR Application Service Interfaces and Ports and the standardized UI Device Interfaces.

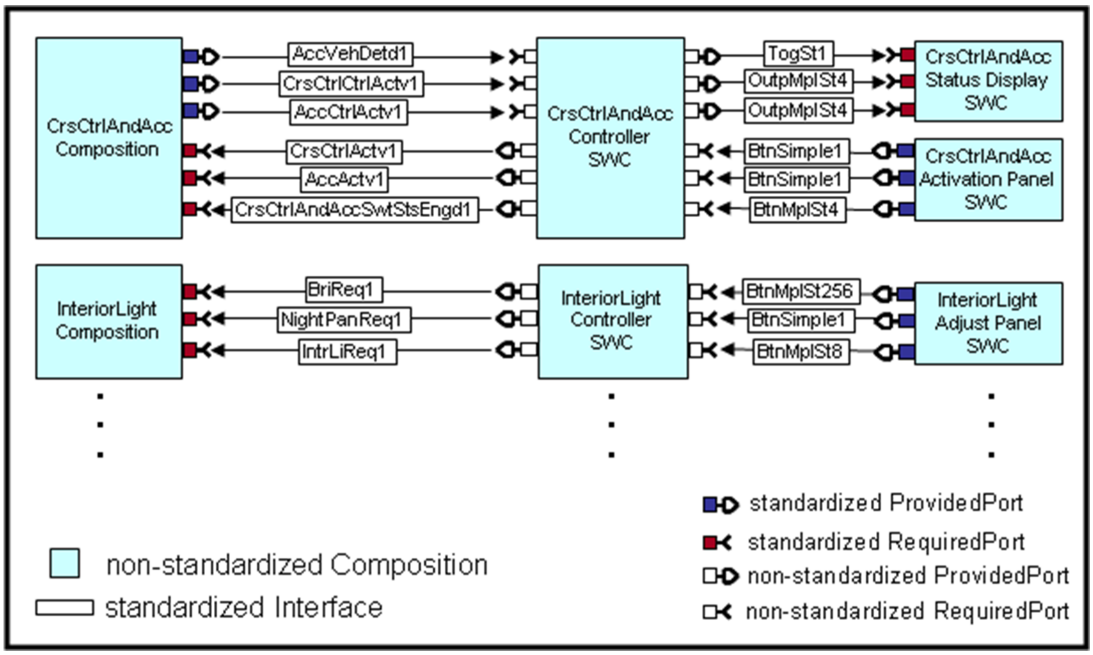


Figure 5.2: Standardized AUTOSAR Ports and Interfaces used in a real-world system

The presentation of the application behavior to the user (HMI behavior) is always vendor and even vehicle variant specific (hard buttons vs. touch-screen). Standardizing a mapping from Application Services to UI Devices is not in the scope of AUTOSAR. Therefore, no Application Controller ports are standardized as they define a system's UI Device mapping.

Two "virtual helper compositions" which act as placeholders for the Application Controllers and UI Device SWCs of real-world systems are used in the AUTOSAR Application Interface model. The virtual ButtonPanel SWC is a placeholder for the UI Device SWCs and the InterDomainController replaces the Application Controllers (see Figure 5.3).

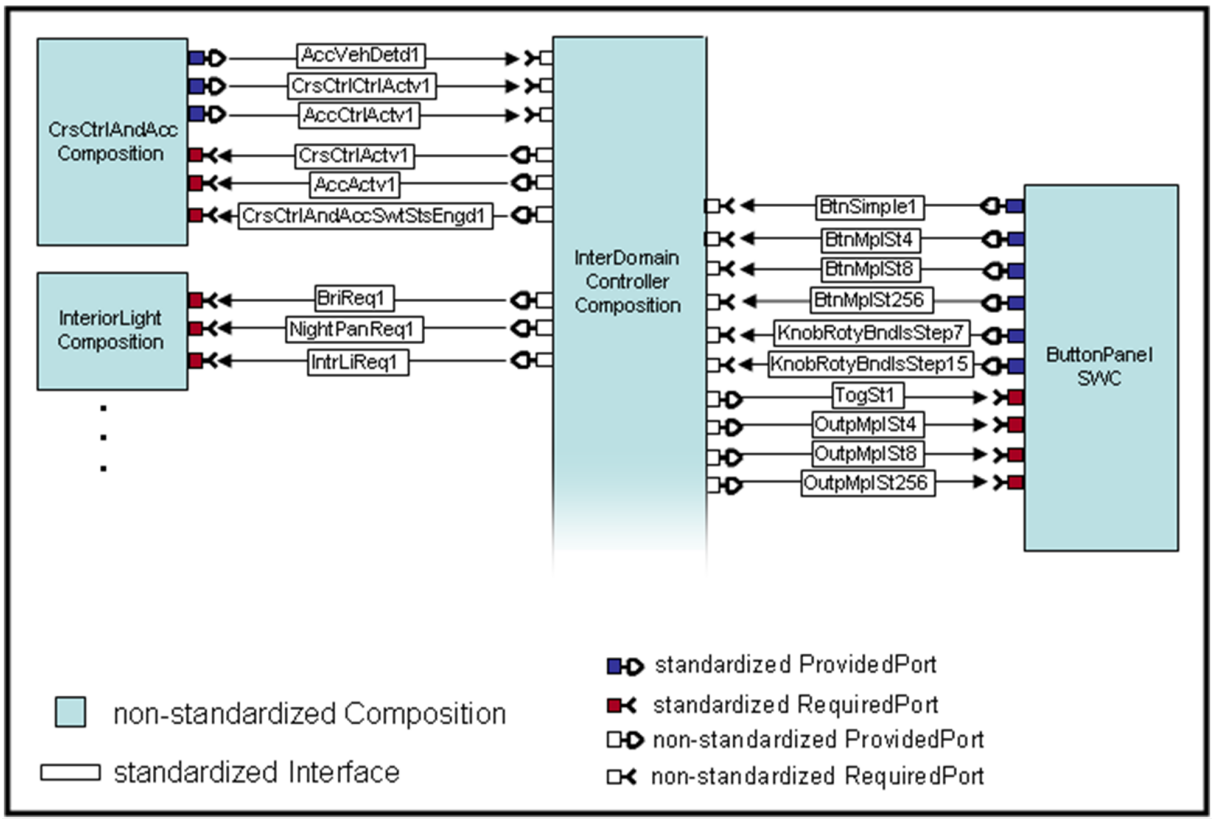


Figure 5.3: Architecture with the InterDomainController and ButtonPanel

5.1 Button Panel Component

The ButtonPanel is a “virtual helper SWC” acting as a placeholder for all the system specific UI Device SWCs which a real-world system would include (see Figure 5.3).

The ButtonPanel aggregates a set of standardized Ports and Interfaces for simple UI Devices like hard buttons, switches, LEDs etc.

The UI-Device Interfaces are standardized with the following principles in mind:

- A UI-Device interface should never be specific to certain functionality.
- A UI-Device interface should be standardized on the most fundamental level possible. If more advanced functionality is required (e.g. a hold long signal after N ms), then that functionality must be layered on top of the UI-Device.

Please Note: The Button Panel Component is not intended to be taken as is for a real system. It is only a virtual example SWC used to group the standardized input and output interfaces.

5.1.1 Simple button interface

Simple button interface	
Description	Used to send a simple user request. This request is provided by a button that supports the two different values “pressed” and “not pressed”. (e.g. turn Park Distance Control functionality on/off).
Comments	• Long press handling is not a part of simple button. • Output value reflects physical position of button.

5.1.2 Toggle state interface

Toggle state interface	
Description	Accept simple value information (activated / deactivated). A typical realization is a LED that shows if the according functionality is active or inactive (e.g. Park distance control functionality state is turned on/off).
Comments	• Other states (e.g. blinking state) are not supported.

5.1.3 Multi state button interface

Multi state button interface	
Description	Used to send a user request with multiple value information to the according SWC when the button/switch/knob etc. has more than two physical positions. The range for the state information may differ from small to large values. State values are always discrete values.
Comments	• The multi state button can also be used for slider / gauge UI devices, especially if these UI devices have a fixed physical state (state is not being handled in software) • The current state value should represent a physical position of the button / slider / gauge etc. • “0” (zero) state value should represent state “off”.

5.1.4 Multi state output interface

Multi state output interface	
Description	Accept state value information for UI hardware devices with multiple possible states. As for the multi state buttons, state value information with different ranges (see Multi State Button description) is used. (e.g. seat heating adjustment with value range 0-3 as state information.
Comments	• “0” (zero) state value should represent state “off”.

5.1.5 Boundless rotary knob interface

Boundless rotary knob interface	
Description	Used to send state output for input devices which does not have distinct user-recognizable physical positions. (e.g. knob/wheel without start/stop position). The movement (e.g. 1 step clockwise) influences the function (e.g. increase volume), not the physical position of the device.
Comments	• The value transmitted is the number of steps/moves since the last transmitted value. • Note that the selected value depth (e.g. 5 bit for +/- 15 steps) is not correlated to the number of physical positions. It rather limits how many steps the user can apply when turning the rotary knob. • The selected value depth is correlated to the update interval. A large update interval enables the user to turn the knob further and thereby making more steps.

5.2 InterDomainController Composition

The InterDomainController is a “virtual helper composition” (see chapter 5). It is not intended for usage within real-world AUTOSAR systems. It is merely a placeholder taking the role of the various proprietary and highly system specific Application Controllers a real-world system would include (see Figure 5.2).

The InterDomainController composition is responsible for handling all communication between the standardized AUTOSAR application service ports and the Ports of the ButtonPanel (see 5.1).

For all the user input required by the standardized AUTOSAR application services it provides a Port which provides the required user input. For the feedback to the user which is being provided by the standardized AUTOSAR application service Ports it provides corresponding RequiredPorts.

A real-world application controller would map the application services' user in- and output to UI-Devices. As mentioned in chapter 5 standardizing such a mapping is not in the scope of AUTOSAR. Therefore, the InterDomainController does not have a corresponding UI-Device attached to it for each and every required/provided user input/output. It only has one port for each standardized UI-Device interface to indicate which role it takes in the overall architecture.

6 Glossary

Several widely used terminologies need to have a unique and shared definition in the scope of this document.

Application A software (or program) that is specified to the solution of a problem of an end user requiring information processing for its solution.

An Application has a functional part (Application Service) and an HMI / Behaviour part (Application Controller).

Domain Set of vehicle functions put together with reference to a certain semantic homogeneity. In AUTOSAR, there are 5 functional domains: Body and comfort, power train, chassis, P&P safety, HMI/Telematics/Multimedia. A domain may be divided into sub domains.

Architecture The fundamental organization of a system embodied in its components, their static and dynamic relationships to each other, and to the environment, and the principles guiding its design and evolution.

HMI logic The allowed and desired sequence of user interactions.

Global HMI Logic The allowed and desired sequence of user interactions between various applications in the system.

Specific HMI Logic The allowed and desired sequence of user interactions within a given application.

Modality The type of output channel to the user, e.g. audio or video output.

7 References

[1] Explanation of Application Interfaces of the Body and Comfort Domain
AUTOSAR_CP_EXP_AIBodyAndComfort