Document Title	Requirements on Communication Management
Document Owner	AUTOSAR
Document Responsibility	AUTOSAR
Document Identification No	716

Document Status	published
Part of AUTOSAR Standard	Adaptive Platform
Part of Standard Release	R21-11

Document Change History

Date	Release	Changed by	Description
2021-11-25	R21-11	AUTOSAR Release Management	• Editorial changes • Removed RESTful communication
2020-11-30	R20-11	AUTOSAR Release Management	• E2E protection – Events and methods – Traceability to Safety Requirements • Service uniqueness: Offered service with Fully Qualified Service ID • Raw Data Streaming requirements set to valid • Communication Groups (draft)
2019-11-28	R19-11	AUTOSAR Release Management	• Added new chapter Requirements Guidelines • Introduced – Signal2Service Translation – Service Versioning – Raw Data Streaming Communication – Changed Document Status from Final to published • Additional E2E support
2019-03-29	19-03	AUTOSAR Release Management	• No content changes

2018-10-31	18-10	AUTOSAR Release Management	• Minor changes and bugfixes
2018-03-29	18-03	AUTOSAR Release Management	• Automatic Reconnection of Proxies • E2E Protection of Methods • REST Network Binding • Minor changes and bugfixes
2017-10-27	17-10	AUTOSAR Release Management	• Introduction of Fields • Introduction of E2E protected communication • Introduction of RESTful communication • Queuing of events • Minor changes and bugfixes
2017-03-31	17-03	AUTOSAR Release Management	Initial release

Disclaimer

Table of Contents

1 Scope of document

This document specifies requirements on Communication Management of the AUTOSAR Adaptive Platform.

2 Conventions to be used

The representation of requirements in AUTOSAR documents follows the table specified in [TPS_STDT_00078], see Standardization Template, chapter Support for Traceability [1].

The verbal forms for the expression of obligation specified in [TPS_STDT_00053] shall be used to indicate requirements, see Standardization Template, chapter Support for Traceability [1].

2.1 Requirements Guidelines

2.1.1 Requirements quality

2.1.2 Requirements identification

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as follows.

Note that the requirement level of the document in which they are used modifies the force of these words.

- MUST: This word, or the adjective "LEGALLY REQUIRED", means that the definition is an absolute requirement of the specification due to legal issues.
- MUST NOT: This phrase, or the phrase "MUST NOT", means that the definition is an absolute prohibition of the specification due to legal issues.
- SHALL: This phrase, or the adjective "REQUIRED", means that the definition is an absolute requirement of the specification.
- SHALL NOT: This phrase means that the definition is an absolute prohibition of the specification.
- SHOULD: This word, or the adjective "RECOMMENDED", means that there may exist valid reasons in particular circumstances to ignore a particular item, but the full implications must be understood and carefully weighed before choosing a different course.
- SHOULD NOT: This phrase, or the phrase "NOT RECOMMENDED", means that there may exist valid reasons in particular circumstances when the particular behavior is acceptable or even useful, but the full implications should be understood and the case carefully weighed before implementing any behavior described with this label.
- MAY: This word, or the adjective "OPTIONAL", means that an item is truly optional. One vendor may choose to include the item because a particular market-

place requires it or because the vendor feels that it enhances the product while another vendor may omit the same item.

An implementation, which does not include a particular option, SHALL be prepared to interoperate with another implementation, which does include the option, though perhaps with reduced functionality. In the same vein an implementation, which does include a particular option, SHALL be prepared to interoperate with another implementation, which does not include the option (except, of course, for the feature the option provides.)

2.1.3 Requirements status

3 Acronyms and abbreviations

The glossary below includes terms, acronyms and abbreviations relevant to AP_RS_CommunicationManagement that are not included in the AUTOSAR Glossary [2].

Terms:	Description:
Fully qualified service ID	A fully qualified name of a service used as a system-wide unique identifier, e.g. 'com.someOEM.adas.collisionwarner'.
Data ID	A unique identifier of an instance of data transmitted. In case of events, this maps to a specific instance of an event.

4 Requirements Specification

This chapter describes all requirements driving the work to define the Communication Management.

4.1 Functional overview

The AUTOSAR Adaptive Platform Communication Management provides services for the network and protocol independent communication between applications. This document therefore includes requirements on

- Communication between applications
  - Signature of the communication API
  - Behavior of the communication API
  - On-wire protocol for inter-ECU and inter-machine data communication to classic and adaptive platform
- Service discovery
  - Scope
  - Service registry
  - On-wire protocol for inter-ECU and inter-machine service discovery
- Configuration of middleware for communication aspects (register services)

The AUTOSAR Adaptive Platform Communication Management provides requirements for the safety mechanisms related to the communication, precisely for the E2E supervision. They are defined in [3].

4.2 Functional Requirements

4.2.1 General requirements

[RS_CM_00001]{DRAFT} **The Communication Management shall provide a standardized header file structure for each service.** ⌈

Description:	The Communication Management shall provide a standardized header file structure for each service. The application uses the standardized header files which are independent of the underlying actual Communication Management implementation.

▽

△

Rationale:	The application code shall be reusable for different AUTOSAR Adaptive platform implementations.
Dependencies:	–
Use Case:	The application developers implement their code against the standardized header files.
Supporting Material:	–

⌋(*RS_Main_00060*)

[RS_CM_00002]{DRAFT} **The service header files shall define the namespace for the respective service.** ⌈

Description:	The service header files shall define the namespace for the respective service to uniquely identify each service instance.
Rationale:	The application code shall be reusable for different AUTOSAR Adaptive platform implementations and for different vehicle lines.
Dependencies:	–
Use Case:	To avoid conflicts with other applications and other services each service shall have its own namespace.
Supporting Material:	–

⌋(*RS_Main_00060*)

[RS_CM_00003]{DRAFT} **The Communication Management shall define how language specific data types are derived from modeled data types.** ⌈

Description:	The Communication Management shall define how language specific data types, e.g. C++ data types, are derived from modeled data types.
Rationale:	The Communication Management shall support different language bindings.
Dependencies:	–
Use Case:	The Communication Management supports C++ language binding and therefore has to define the modeled data types in C++.
Supporting Material:	–

⌋(*RS_Main_00060*)

[RS_CM_00004]{DRAFT} **Communication Management shall support the translation between signal-based and service-oriented communication** ⌈

Description:	The Communication Management shall support the translation between signal-based and service-oriented communication in the protocol binding
Rationale:	Adaptive Platform restricts communication to Service-oriented communication, the rest of the vehicle however still uses Signal-based communication means - therefore a translation of these two approaches shall be performed.
Dependencies:	–
Use Case:	Data which is produced on a Can ECU is needed at an Adaptive machine in a safe and secure manner. The protocol binding is able to de-serialize and serialize signal-based Can messages and provide the payload to the adaptive application.
Supporting Material:	–

⌋(*RS_Main_00652*)

4.2.2 Service-oriented communication

4.2.2.1 Communication between applications

[RS_CM_00200] The Communication Management shall transform Fully Qualified Service IDs to communication protocol specific Service IDs ⌈

Description:	The Communication Management shall transform Fully Qualified Service IDs to communication protocol specific Service IDs. Fully Qualified Service IDs are used within the application code by the developer and need to be defined to enable cooperation of services of different vendors. Communication protocol specific Service IDs may be used within the messages on the network and may be needed if the communication protocol service ID space was not designed for Fully Qualified Service IDs.
Rationale:	Binary of application shall be unaware of communication protocol specific Service IDs.
Dependencies:	–
Use Case:	One platform is used in multiple vehicle lines but the service IDs are different for the platform in the two vehicle lines. Communication Binding still uses Fully Qualified Service IDs. Communication Management transforms Fully Qualified to SOME/IP Service IDs.
Supporting Material:	see Adaptive Platform Scenarios

⌋(*RS_Main_00140*)

[RS_CM_00204] The Communication Management shall map the protocol independent Service Oriented Communication to the configured protocol binding and shall execute the protocol accordingly. ⌈

Description:	The Communication Management shall map the protocol independent Service Oriented Communication to the configured protocol binding and shall execute the protocol accordingly. The application code shall use service oriented communication independently of the actually configured protocol. It is the responsibility of the Communication Management to realize the specific protocol.
Rationale:	Binary of application shall be unaware of communication protocol specifics.
Dependencies:	–
Use Case:	One application is used in multiple vehicle lines but the used communication protocols are different in the two vehicle lines, e.g. in one case it uses SOME/IP and in another case it uses local IPC.
Supporting Material:	–

⌋(*RS_Main_00140, RS_Main_01001*)

[RS_CM_00315]{DRAFT} **The Communication Management shall support a change of the configured protocol binding without requiring a re-compilation of the adaptive application** ⌈

Description:	Since the selection of a particular network protocol binding is an integrator driven deployment decision, any change in the selection of a particular network protocol binding or changes in the various attributes and parameters of a particular network protocol binding shall be possible without requiring a re-compilation of the involved adaptive applications. The required changes to the involved adaptive application shall be limited to a re-linking (either static or dynamic) of the involved adaptive application.
Rationale:	Binary of application shall be unaware of concrete configured protocol binding. Concrete protocol binding shall be configurable/changeable upon deployment time of the binary of the application.
Dependencies:	–
Use Case:	Binary of application shall be usable in various different deployment scenarios (e.g., using SOME/IP protocol binding in one deployment scenario and local IPC-based protocol binding in some other deployment scenario.)
Supporting Material:	see Adaptive Platform Scenarios

⌋(*RS_Main_00140*)

[RS_CM_00205] The Communication Management shall realize the SOME/IP service discovery protocol, the SOME/IP protocol and the E2E supervision (E2E protocol). ⌈

Description:	The Communication Management shall realize the SOME/IP service discovery protocol, the SOME/IP protocol and the E2E supervision (E2E protocol). The protocols are described in AUTOSAR SOME/IP Service Discovery Protocol specification, AUTOSAR SOME/IP Protocol specification and AUTOSAR E2E Protocol Specification. SOME/IP and E2E protocols shall be realized as independent protocol layers, without dependencies between each other.
Rationale:	SOME/IP and E2E are supported in both AUTOSAR classic and adaptive.
Dependencies:	–
Use Case:	Radar, Camera and SensorFusion applications communicate by Ethernet using SOME/IP protocol. Safety-related applications communicating over a non-safety-related bus (e.g. Ethernet, CAN).
Supporting Material:	–

⌋(*RS_Main_01002*, *RS_Main_00010*)

[RS_CM_00499]{DRAFT} **Service Contract Versioning shall be supported by the AUTOSAR** ⌈

Description:	Service Contract Versioning shall be supported by AUTOSAR.
Rationale:	Service Contract Versioning is an essential feature for SOA systems
Dependencies:	–
Use Case:	Service catalogue, a system which contains different Service versions
Supporting Material:	–

⌋(*RS_Main_00060*, *RS_Main_01002*, *RS_Main_00140*)

[RS_CM_00500] Service Contract Version for a Service Interface ⌈

Description:	The Service Interface shall contain a Service contract version number
Rationale:	The versioning number is used to differentiate different Service configurations (scope Service interface and behaviour)
Dependencies:	–
Use Case:	Service catalogue, a system which contains different Service versions
Supporting Material:	–

⌋(*RS_Main_00060*, *RS_Main_01002*, *RS_Main_00140*)

[RS_CM_00501] Service Contract Versioning for all Transport Deployment Protocols ⌈

Description:	The transport deployment protocols shall support Service contract versioning
Rationale:	To support Service contract from design phase to deployment phase the transport deployment protocols need to support versioning
Dependencies:	–
Use Case:	Support of service contract versioning inside AUTOSAR Adaptive platform
Supporting Material:	–

⌋(*RS_Main_00060*, *RS_Main_01002*, *RS_Main_00140*)

4.2.2.2 Service discovery

[RS_CM_00101] Communication Management shall provide an interface to offer services ⌈

Description:	Application developers shall be able to offer services provided by their application for usage by other applications.
Rationale:	To support communication a mechanism is needed to offer provided services to other applications, which are able to use them.
Dependencies:	–
Use Case:	Application “A” offers a wall clock service to other applications.
Supporting Material:	–

⌋(*RS_Main_01002*, *RS_Main_00060*, *RS_Main_00140*)

[RS_CM_00108]{DRAFT} **Service Communication – Uniqueness of offered service** ⌈

Description:	The Communication Management shall check the offered service for uniqueness (i.e., a service with the same ServiceIdentifier and InstanceIdentifier was already previously offered). Uniqueness shall be applied locally and, if possible, on system/vehicle-wide level using a Fully Qualified Service ID for identification purposes.
Rationale:	To support communication a mechanism is needed to offer provided services to other applications, which are able to use them.
Dependencies:	–
Use Case:	Application “A” offers a wall clock service to other applications.
Supporting Material:	–

⌋(*RS_Main_01002*, *RS_Main_00060*, *RS_Main_00140*)

[RS_CM_00102] Communication Management shall provide an interface to find services ⌈

Description:	Application developers shall be able to find all service instances provided by other applications at runtime.
Rationale:	To establish communication during runtime a mechanism is needed to find provided services based on the type of the service and the concrete service instance.
Dependencies:	–
Use Case:	Application “A” searches for a wall clock service provided by another application. Communication Management finds all available matching service instances and the application can select the right one.
Supporting Material:	–

⌋(*RS_Main_01002*, *RS_Main_00060*, *RS_Main_00140*)

[RS_CM_00103] Communication Management shall provide an interface to subscribe to a specific event provided by an instance of a certain service ⌈

Description:	Application developers shall be able to subscribe to one specific event inside of one selected service instance.
Rationale:	After finding instances of a service type, it shall be possible to subscribe to certain events of a specific instance.
Dependencies:	–
Use Case:	Application “A” subscribes to the power on event of the application controlling the ignition lock.
Supporting Material:	–

⌋(*RS_Main_01002*, *RS_Main_00060*, *RS_Main_00140*)

[RS_CM_00104] Communication Management shall provide an interface to stop the subscription to an event of a service instance ⌈

Description:	Application developers shall be able to stop an active subscription to an event of a service instance by the application.
Rationale:	After subscribing to an event of a specific service instance, it shall be possible to stop the subscription later on.
Dependencies:	–
Use Case:	Application “A” stops the subscription to the power on event and receives no longer such events.
Supporting Material:	–

⌋(*RS_Main_01002*, *RS_Main_00060*, *RS_Main_00140*)

[RS_CM_00106] Communication Management shall provide a means to monitor the state of the subscription to an event ⌈

Description:	Application developers shall be able to query the current state of a subscription to an event of a service instance by the application or to get a notification about the change of the current state of a subscription to an event of a service instance.
Rationale:	It shall be possible to monitor/query the actual state of a subscription.
Dependencies:	–
Use Case:	An Application wants to keep track of the subscription state to the power on event and get notified if it changes.
Supporting Material:	–

⌋(*RS_Main_01002*, *RS_Main_00060*, *RS_Main_00140*)

[RS_CM_00107] Communication Management shall provide a means to automatically update a proxy instance in case of restart of the offered service ⌈

Description:	Communication Management shall automatically update the proxy instance of a service in a way that clients do not have to renew/reinstanciate their proxy instances. The reconnection has to fulfill all the requirements for the initial connection attempt.
Rationale:	It shall be possible to use a proxy instance independably of wether a server instance has been restarted and/or the handle of the proxy instance changes.
Dependencies:	–
Use Case:	Save the client application from the effort of keeping track of subscription state and resubscribing in case of restarts on server side
Supporting Material:	–

⌋(*RS_Main_01002*, *RS_Main_00060*, *RS_Main_00140*)

[RS_CM_00105] Communication Management shall provide an interface to stop offering services ⌈

Description:	Application developers shall be able to stop the offering services which the application started to offer before.
Rationale:	After the offering of a service, it shall be possible to stop the offering of the service later on.
Dependencies:	–
Use Case:	Application “A” stops offering the wall clock service.
Supporting Material:	–

⌋(*RS_Main_01002*, *RS_Main_00060*, *RS_Main_00140*)

[RS_CM_00700]{DRAFT} **The Service Discovery shall evaluate the service version compatibility for service connection** ⌈

Description:	The Service Discovery shall evaluate the Service version compatibility for the service connection
Rationale:	Only Service versions of providers and clients which are compatible can be connected.
Dependencies:	–
Use Case:	Support of service contract versioning inside AUTOSAR Adaptiven platform
Supporting Material:	–

⌋(*RS_Main_00060*, *RS_Main_01002*, *RS_Main_00140*)

[RS_CM_00701] Service Versioning Blacklist ⌈

Description:	A Service client instance can specify provider Service version which shall not be used for connection.
Rationale:	This feature enables last minute changes to "block" provider versions.
Dependencies:	–
Use Case:	Last minute changes/patches on OEM site
Supporting Material:	–

⌋(*RS_Main_00060*, *RS_Main_01002*, *RS_Main_00140*)

4.2.2.3 Communication

4.2.2.3.1 Events

[RS_CM_00201] Communication Management shall provide an API to send events to other applications ⌈

Description:	Application developers shall be able to provide data in form of events to other applications.
Rationale:	After offering a service, it shall be possible to send events of the respective service to all subscribed applications.
Dependencies:	–
Use Case:	Application “A” sends the power on event upon turning the key in the ignition lock.
Supporting Material:	–

⌋(*RS_Main_01002*, *RS_Main_00060*, *RS_Main_00140*)

[RS_CM_00223]{DRAFT} **The Communication Management shall protect the transmission of events using E2E protocol. The E2E Protection has to be executed behind the event API.** ⌈

Description:	Application developers shall be able to have an E2E-protected event-based communication, regardless of the bus used.
Rationale:	It shall be ensured that communication failure modes introduced by the communication bus (on the E2E-protected serialized data) which are detectable by the E2E protocol are detected by Communication Management.Note: It depends on the used communication type (periodic/ non-periodic) and the application which failure modes are to be detected.
Dependencies:	–
Use Case:	Application “A” receives an E2E-protected speed (as a part of an event). In case of a corruption or a loss, this is detected by a periodic polling by application and by E2E checks (CRC and a stuck-at counter), reported by Communication Management by E2E result. As a result, the application could enforce the safe state of its function, e.g. refusing to open tail gate.
Supporting Material:	[3]

⌋(*RS_Main_01002, RS_Main_00060, RS_Main_00140, RS_Main_00010, RS_SAF_-21601*)

[RS_CM_00202] Communication Management shall provide an API to the application to poll received events ⌈

Description:	Application developers shall be able to query whether a certain event has been received from another application and read that data at the same time.
Rationale:	After subscribing to an event of a specific service instance, it shall be possible to receive events send by the server and access them in a polling-based style.
Dependencies:	–
Use Case:	Application “A” polls for the receiving of the power on event and is able to access the corresponding data.
Supporting Material:	–

⌋(*RS_Main_01002, RS_Main_00060*)

[RS_CM_00203] Communication Management shall trigger the application on reception of an event ⌈

Description:	Application developers shall be able to let the platform trigger the application when a new event has been received from another application. The platform shall not deliver the data directly, but instead provide a mechanism to read the data upon request.
Rationale:	After subscribing to an event of a specific service instance, it shall be possible to receive events send by the server and access them in a event-based style through triggering of a processing function.
Dependencies:	–
Use Case:	Application “A” gets triggered whenever receiving the power on event and is able to access the corresponding data.

▽

△

Supporting Material:	–

⌋(*RS_Main_01002*, *RS_Main_00060*)

[RS_CM_00206]{DRAFT} **Communication Management shall queue received events** ⌈

Description:	Communication Management shall queue received events with configurable queue length and policy.
Rationale:	The application wants to ensure that it receives the last n events, n being the queue length.
Dependencies:	–
Use Case:	An application polls received events and wants to get the last n events received since the last polling.
Supporting Material:	–

⌋(*RS_Main_01002*, *RS_Main_00060*)

[RS_CM_00224]{DRAFT} **The communication management shall provide the E2E information of the received event to the application.** ⌈

Description:	The communication management shall provide the E2E information of the received event to the application.
Rationale:	In case of reception of invalid E2E check result, the application shall be able to perform an appropriate error handling. The access to the event data is identical for safety-related and non-safety-related data.
Dependencies:	–
Use Case:	Application “A” polls gets invalid E2E check result and as a result it switches to a safe state.
Supporting Material:	The provided E2E information shall be, for each event in the queue: E2E status, E2E state, and the sample. Note that in case applications are triggered, there may be a need of an application-level detection of timeouts. This is because in case of delay or loss, the event will not arrive and E2E check will not be performed.

⌋(*RS_Main_01002*, *RS_Main_00060*, *RS_Main_00010*, *RS_SAF_21601*)

4.2.2.3.2 Methods

[RS_CM_00211] Communication Management shall provide an interface to provide methods to other applications ⌈

Description:	Application developers shall be able to provide methods which can be called by other applications.

▽

△

Rationale:	After offering a service, it shall be possible for other applications to call methods of the service and get the respective result.
Dependencies:	–
Use Case:	Application “A” calls the “getCurrentTime” method of the wall clock service provided by application “B”.

⌋(*RS_Main_01002*, *RS_Main_00060*, *RS_Main_00140*)

[RS_CM_00212] Communication Management shall provide an interface to call methods of other applications synchronously ⌈

Description:	Application developers shall be able to synchronously call methods provided by other applications. It is required that the result is available when the method call returns.
Rationale:	After finding a service, it shall be possible for an application to call a method of the service as a synchronous service call: the calling application wants to wait for the completion of the service method execution and have the result available before continuing.
Dependencies:	–
Use Case:	Application “A” calls the “getCurrentTime” method of the wall clock service provided by application “B” and wants to stop processing until the result has been received.
Supporting Material:	–

⌋(*RS_Main_01002*, *RS_Main_00060*, *RS_Main_00140*)

[RS_CM_00213] Communication Management shall provide an interface to call service methods asynchronously ⌈

Description:	Application developers shall be able to asynchronously call methods provided by other applications. It is not required that the result is available when the method call returns. Either the calling application checks for the completion of the service method execution by itself or it is notified on the completion.
Rationale:	After finding a service, it shall be possible for an application to call a method of the service as an asynchronous service call: the calling application does not want to wait for the completion of the service method execution and continues without the result available.
Dependencies:	–
Use Case:	Application “A” calls the “getCurrentTime” method of the wall clock service provided by application “B” asynchronously and can do further processing until the “getCurrentTime” method execution is finished.
Supporting Material:	–

⌋(*RS_Main_01002*, *RS_Main_00060*, *RS_Main_00140*)

[RS_CM_00400]{DRAFT} **Communication Management shall protect the transmission of methods using E2E protocol.** ⌈

Description:	Communication Management shall, transparent to the application, protect the transmission of methods using E2E protocol.
Rationale:	It shall be ensured that communication failure modes introduced by the communication bus (on the E2E-protected serialized request or response data) which are detectable by the E2E protocol are detected at the client side by Communication Management. Note: It depends on the used communication type (periodic/ non-periodic) and the application which failure modes are to be detected.
Dependencies:	–
Use Case:	E2E protected method calls in client-server based communication
Supporting Material:	[3]

⌋(*RS_Main_01002, RS_Main_00060, RS_Main_00010, RS_Main_00140 RS_SAF_-21601*)

[RS_CM_00401]{DRAFT} **The communication management shall provide the E2E information of the received method call to the application.** ⌈

Description:	The communication management shall provide the E2E information of the received method call to the application.
Rationale:	In case of reception of invalid E2E check result, the application shall be able to propagate detected E2E failure modes to the response data provided to the client. The access to the request data is identical for safety-related and non-safety-related data.
Dependencies:	–
Use Case:	Application “B” provides a method and this method is called by application “A” and receives with the request invalid E2E check result and as a result the same invalid E2E data are added to the response data
Supporting Material:	The provided E2E information shall be E2E status, E2E state and object data.

⌋(*RS_Main_01002, RS_Main_00060, RS_Main_00010 RS_SAF_21601*)

[RS_CM_00402]{DRAFT} **Communication Management shall support a decision for applying the method call based on E2E results.** ⌈

Description:	Communication Management shall provide the E2E information to the application that has received the request for the method call and decide based on the E2E information if the called method will be applied or not.
Rationale:	In case of reception of invalid E2E check result, the application shall be able to skip the received method call. Requests which have been corrupted during data transmission are not valid and could result in unintended results. The user of these results is not aware of this.
Dependencies:	–
Use Case:	Application “B” provides a method and this method is called by application “A” and receives with the request invalid E2E check result and as a result the called method is not applied

▽

△

Supporting Material:	–

⌋(*RS_Main_01002*, *RS_Main_00060*, *RS_Main_00010*)

[RS_CM_00225]{DRAFT} **Communication Management shall provide an interface to call fire&forget service methods** ⌈

Description:	Application developers shall be able to call fire&forget methods provided by other applications for a "best effort" approach. The application is not expecting any kind of acknowledge or handshake from the provider of the service method. It even accepts, that the call will not even reach the provider of the service method.
Rationale:	After finding a service, it shall be possible for an application to call a method of the service as an fire&forget service call: the calling application does not want to get any information about the service method execution.
Dependencies:	–
Use Case:	Application “A” calls the “setCurrentTime” fire&forget method of the wall clock service provided by application “B” and will continue its execution independently of the method execution within “B”.
Supporting Material:	–

⌋(*RS_Main_01002*, *RS_Main_00060*, *RS_Main_00140*)

[RS_CM_00214]{DRAFT} **Communication Management shall provide an interface to query the result of an asynchronously called service method** ⌈

Description:	Application developers shall be able to retrieve the result of an asynchronously called method. The method to query the result can be called at any time after the called method has returned: if it is called before completion of the service method execution, it returns instantly; if it is called after completion, it returns providing the result.
Rationale:	After calling a service method asynchronously, the application shall be able to get the result of this service method.
Dependencies:	–
Use Case:	Application “A” calls the “getCurrentTime” method of the wall clock service provided by application “B” asynchronously. After “getCurrentTime” method execution has completed, Application “A” accesses the result.
Supporting Material:	–

⌋(*RS_Main_01002*, *RS_Main_00060*)

[RS_CM_00215] Communication Management shall trigger the application on completion of an asynchronously called service method ⌈

Description:	Application developers shall be able to let the platform trigger the application when the result of an asynchronously called method is available.
Rationale:	After asynchronously calling a method of a specific service instance, it shall be possible to trigger a processing function on the availability of the method result.
Dependencies:	–
Use Case:	Application “A” calls the “getCurrentTime” method of the wall clock service provided by application “B” asynchronously. After “getCurrentTime” method execution has completed, Application “A” a specific function of Application “A” is called to notify that the result of “getCurrentTime” is available.
Supporting Material:	–

⌋(*RS_Main_01002*, *RS_Main_00060*)

[RS_CM_00403]{DRAFT} **Communication management shall provide an interface to detect delay of E2E protected service responses at the client side by supervision of a predefined response deadline.** ⌈

Description:	Communication management shall provide an interface to detect delayed service responses at the client side by supervision of a predefined response deadline.
Rationale:	A delayed response shall be detected and the application can apply a safety related error reaction.
Dependencies:	–
Use Case:	Client is sending a method call. Client is awaiting the response within 300ms. After reaching the deadline the fault is detected at client side.
Supporting Material:	–

⌋(*RS_Main_01002*, *RS_Main_00060*, *RS_Main_00010*, *RS_Main_00140 RS_SAF_-21601*)

[RS_CM_00404]{DRAFT} **The communication management shall provide the E2E information of the method response to the application.** ⌈

Description:	The communication management shall provide the E2E information of the method response to the application.
Rationale:	In case of reception of invalid E2E check result, the application shall be able to perform an appropriate error handling. The access to the response data is identical for safety-related and non-safety-related data.
Dependencies:	–
Use Case:	Application “A” requests a method call and receives with the response an invalid E2E check result and as a result it switches to a safe state.
Supporting Material:	Note, there may be a need of an application-level monitoring of a deadline to stop waiting for a response.

⌋(*RS_Main_01002*, *RS_Main_00060*, *RS_Main_00010 RS_SAF_21601*)

4.2.2.3.3 Fields

[RS_CM_00216]{DRAFT} **Communication Management shall provide an interface which aggregates methods to receive a notification on a changed field value as well as explicitly getting and setting the field value** ⌈

Description:	The interface shall aggregate the methods to receive a notification that the field value has changed, explicitly get the field value and explicitly set the field value. It shall also be possible to aggregate any non-empty subset of these methods.
Rationale:	To access properties held in a central location it is necessary to be able to query and modify the value as well as get notifications on change of the value.
Dependencies:	–
Use Case:	The consumers of sensor data would like to know the update rate of the sensor data and want to be able to change this value if they have different requirements.
Supporting Material:	–

⌋*(RS_Main_01002, RS_Main_00060)*

[RS_CM_00227]{DRAFT} **Communication Management shall trigger the application on reception of a notification on field value change** ⌈

Description:	Application developers shall be able to let the platform trigger the application when a new notification on field value change has been received from another application. The platform shall not deliver the data directly, but instead provide a mechanism to read the data upon request.
Rationale:	It shall be possible to receive notifications on field value change sent by the server and access them in a event-based style through triggering of a processing function.
Dependencies:	–
Use Case:	A consumer of sensor data gets triggered on reception of a modified update rate.
Supporting Material:	–

⌋*(RS_Main_01002, RS_Main_00060)*

[RS_CM_00217] Communication Management shall provide a method to remotely set the field value ⌈

Description:	Communication Management shall provide a method to remotely set the field value.
Rationale:	The application wants to change the value of a field provided by another application.
Dependencies:	–
Use Case:	A consumer of sensor data would like to modify the update rate.

▽

△

Supporting Material:	–

⌋(*RS_Main_01002*, *RS_Main_00060*)

[RS_CM_00218] Communication Management shall provide a method to remotely get the field value ⌈

Description:	Communication Management shall provide a method to remotely get the field value.
Rationale:	The application wants to know the value of a field provided by another application.
Dependencies:	–
Use Case:	A consumer of sensor data would like to know the sensor's current update rate.
Supporting Material:	–

⌋(*RS_Main_01002*, *RS_Main_00060*)

[RS_CM_00219]{DRAFT} **Communication Management shall provide an interface which aggregates methods to send a notification on value change and to register a get and set function for the field value** ⌈

Description:	The interface shall aggregate the methods to send a notification on value change, and to register a get and set function for the field value. It shall also be possible to aggregate any non-empty subset of these methods. In case the method to send a notification is included in the aggregation, the first subscription shall trigger an event to the client with the field value.
Rationale:	To share properties held in a central location with multiple consumers the providing application shall offer methods to query the current property value, modify the value and notify all consumers of changes to the value.
Dependencies:	–
Use Case:	The provider of sensor data notifies the consumers about the update rate. It also provides methods to the consumers to query and modify the update rate.
Supporting Material:	–

⌋(*RS_Main_01002*, *RS_Main_00060*)

[RS_CM_00226]{DRAFT} **Communication Management shall provide a method to send notifications on data change to other applications** ⌈

Description:	Application developers shall be able to send notifications on change of a field value to other applications.
Rationale:	To share properties held in a central location with multiple consumers the providing application shall be able to notify all consumers of changes to the value.
Dependencies:	–

▽

△

Use Case:	A provider of sensor data notifies the consumers that the current update rate has changed.
Supporting Material:	–

⌋(*RS_Main_01002*, *RS_Main_00060*)

[RS_CM_00220] Communication Management shall trigger the set method of the application which provides the field ⌈

Description:	Communication Management shall trigger the set method of the application which provides the field.
Rationale:	When other applications want to modify the current value of a field the Communication Management shall trigger the respective set method of the providing application.
Dependencies:	–
Use Case:	A provider of sensor data offers the consumers a method to modify the current update rate.
Supporting Material:	–

⌋(*RS_Main_01002*, *RS_Main_00060*)

[RS_CM_00221] Communication Management shall trigger the get method of the application which provides the field ⌈

Description:	Communication Management shall trigger the get method of the application which provides the field.
Rationale:	When other applications query the current value of a field the Communication Management shall trigger the respective get method of the providing application.
Dependencies:	–
Use Case:	A provider of sensor data offers the consumers a method to query the current update rate.
Supporting Material:	–

⌋(*RS_Main_01002*, *RS_Main_00060*)

4.2.3 Raw Data Streaming Communication

[RS_CM_00410] The Communication Management shall provide an API to support reading and writing raw data streams that has no datatype information ⌈

Description:	The Communication Management shall provide an API to support reading and writing raw data streams that has no datatype information.
Rationale:	A separate API is needed, since using SOME/IP services for raw data streams requires typed data and serialization, which introduces unnecessary overhead.
Dependencies:	–
Use Case:	Exchanging binary data between an ECU running adaptive applications and sensors in a car.
Supporting Material:	–

⌋(*RS_Main_00140*, *RS_Main_00060*)

[RS_CM_00411] Application developers shall be able to send and receive raw binary data streams independent of the underlying network protocol ⌈

Description:	Application developers shall be able to send and receive raw binary data streams independent of the underlying network protocol
Rationale:	The application API for Raw Data Streams shall be static. The network protocol and configuration shall only be handled in the deployment information.
Dependencies:	–
Use Case:	Sensors using different network protocols can be changed without updating the application code.
Supporting Material:	–

⌋(*RS_Main_00140*)

[RS_CM_00412] The Communication Management shall provide TCP/IP Sockets as network protocol for Raw Data Streams ⌈

Description:	The Communication Management shall provide TCP/IP Sockets as network protocol for Raw Data Streams
Rationale:	Sockets is the most common transport mechanism used for ethernet communication
Dependencies:	–
Use Case:	Exchanging binary data between an ECU running adaptive applications and sensors in a car
Supporting Material:	–

⌋(*RS_Main_00280*)

4.2.4 Communication Group

[RS_CM_00600]{DRAFT} **Creation of CommunicationGroups** ⌈

Description:	CommunicationManagement shall support creation of preconfigured CommunicationGroups.
Rationale:	CommunicationGroups enable easy exchange of information between group of Processes that works together.
Dependencies:	–
Use Case:	State Management can create CommunicationGroup to synchronize efforts that are needed when system switch from one power mode to another. Requests from State Management can be send to all participants and responses could be tracked.
Supporting Material:	–

⌋(*RS_Main_00140*, *RS_Main_00060*, *RS_Main_00460*)

[RS_CM_00601]{DRAFT} **Provide origin of information** ⌈

Description:	CommunicationManagement shall support differentiation of origin of messages exchanged inside a CommunicationGroup.
Rationale:	Dedicated group members should be able to know from whom message arrived. Without this information it will not be possible to check if all group members replied.
Dependencies:	–
Use Case:	If Processes want to coordinate actions across group, it will need to check if all Processes that should perform an action replayed to re-quest.
Supporting Material:	–

⌋(*RS_Main_00140*, *RS_Main_00060*, *RS_Main_00460*)

4.3 Non-Functional Requirements

There is no non-functional requirement.

5 Requirements Tracing

The following table references the AUTOSAR main requirements specified in [4] and links to the fulfillment of these.

Feature	Description	Satisfied by
[RS_Main_00010]	AUTOSAR shall support the development of safety related systems	[RS_CM_00205] [RS_CM_00223] [RS_CM_00224] [RS_CM_00400] [RS_CM_00402] [RS_CM_00403]
[RS_Main_00010 RS_SAF_21601]	No description	[RS_CM_00401] [RS_CM_00404]
[RS_Main_00060]	AUTOSAR shall provide a standardized software interface for communication between Applications	[RS_CM_00001] [RS_CM_00002] [RS_CM_00003] [RS_CM_00101] [RS_CM_00102] [RS_CM_00103] [RS_CM_00104] [RS_CM_00105] [RS_CM_00106] [RS_CM_00107] [RS_CM_00108] [RS_CM_00201] [RS_CM_00202] [RS_CM_00203] [RS_CM_00206] [RS_CM_00211] [RS_CM_00212] [RS_CM_00213] [RS_CM_00214] [RS_CM_00215] [RS_CM_00216] [RS_CM_00217] [RS_CM_00218] [RS_CM_00219]

		[RS_CM_00220] [RS_CM_00221] [RS_CM_00223] [RS_CM_00224] [RS_CM_00225] [RS_CM_00226] [RS_CM_00227] [RS_CM_00400] [RS_CM_00401] [RS_CM_00402] [RS_CM_00403] [RS_CM_00404] [RS_CM_00410] [RS_CM_00499] [RS_CM_00500] [RS_CM_00501] [RS_CM_00600] [RS_CM_00601] [RS_CM_00700] [RS_CM_00701]
[RS_Main_00140]	AUTOSAR shall provide network independent communication mechanisms for applications	[RS_CM_00101] [RS_CM_00102] [RS_CM_00103] [RS_CM_00104] [RS_CM_00105] [RS_CM_00106] [RS_CM_00107] [RS_CM_00108] [RS_CM_00200] [RS_CM_00201] [RS_CM_00204] [RS_CM_00211] [RS_CM_00212] [RS_CM_00213] [RS_CM_00223] [RS_CM_00225] [RS_CM_00315] [RS_CM_00410] [RS_CM_00411] [RS_CM_00499] [RS_CM_00500] [RS_CM_00501] [RS_CM_00600] [RS_CM_00601] [RS_CM_00700] [RS_CM_00701]
[RS_Main_00140 RS_SAF_21601]	No description	[RS_CM_00400] [RS_CM_00403]
[RS_Main_00280]	AUTOSAR shall support standardized automotive communication protocols	[RS_CM_00412]
[RS_Main_00460]	AUTOSAR shall standardize methods to organize mode management on Application, ECU and System level	[RS_CM_00600] [RS_CM_00601]
[RS_Main_00652]	AUTOSAR shall support the translation between signal-based and service-oriented communication	[RS_CM_00004]

[RS_Main_01001]	AUTOSAR shall support intra ECU communication	[RS_CM_00204]
[RS_Main_01002]	AUTOSAR shall support service-oriented communication	[RS_CM_00101] [RS_CM_00102] [RS_CM_00103] [RS_CM_00104] [RS_CM_00105] [RS_CM_00106] [RS_CM_00107] [RS_CM_00108] [RS_CM_00201] [RS_CM_00202] [RS_CM_00203] [RS_CM_00205] [RS_CM_00206] [RS_CM_00211] [RS_CM_00212] [RS_CM_00213] [RS_CM_00214] [RS_CM_00215] [RS_CM_00216] [RS_CM_00217] [RS_CM_00218] [RS_CM_00219] [RS_CM_00220] [RS_CM_00221] [RS_CM_00223] [RS_CM_00224] [RS_CM_00225] [RS_CM_00226] [RS_CM_00227] [RS_CM_00400] [RS_CM_00401] [RS_CM_00402] [RS_CM_00403] [RS_CM_00404] [RS_CM_00499] [RS_CM_00500] [RS_CM_00501] [RS_CM_00700] [RS_CM_00701]
[RS_SAF_21601]	Communication Management shall provide mechanisms for detection of errors during the exchange of information among software components, by considering all faults listed in the ISO standard (ISO 26262:6-2018 D.2.4).	[RS_CM_00223] [RS_CM_00224]

6 References

[1] Standardization Template
AUTOSAR_TPS_StandardizationTemplate

[2] Glossary
AUTOSAR_TR_Glossary

[3] Requirements on E2E
AUTOSAR_RS_E2E

[4] Main Requirements
AUTOSAR_RS_Main