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

Document Status	Final
Part of AUTOSAR Standard	Adaptive Platform
Part of Standard Release	19-03

Document Change History			
Date	**Release**	**Changed by**	**Description**
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

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 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].

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 spread across several sections in this document.

4.2 Functional Requirements

4.2.1 General requirements

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

Type:	draft

▽

△

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] The service header files shall define the namespace for the respective service. ⌈

Type:	draft
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] The Communication Management shall define how language specific data types are derived from modeled data types. ⌈

Type:	draft
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*)

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 ⌈

Type:	draft
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. ⌈

Type:	draft
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] The Communication Management shall support a change of the configured protocol binding without requiring a re-compilation of the adaptive application ⌈

Type:	draft
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). ⌈

Type:	draft
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_00222] The Communication Management shall transform Fully Qualified Service IDs, its instance and Event ID or Method ID to E2E Data ID. ⌈

Type:	draft
Description:	The Communication Management shall transform Fully Qualified Service IDs, its instance and Event ID or Method ID to E2E Data ID.
Rationale:	E2E Supervision is bus independent and operates on Data IDs.
Dependencies:	−
Use Case:	E2E supervision used for communication protection, for file system protection.
Supporting Material:	see PRS E2E Supervision in Foundation

⌋*(RS_Main_00140, RS_Main_00010)*

4.2.2.2 Service discovery

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

Type:	draft
Description:	Application developers shall be able to offer services provided by their application for usage by other applications. The services shall be offered 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:	−

s ⌋*(RS_Main_01002, RS_Main_00060, RS_Main_00140)*

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

Type:	draft
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 ⌈

Type:	draft
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 ⌈

Type:	draft
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 ⌈

Type:	draft
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 ⌈

Type:	draft
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 ⌈

Type:	draft
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*)

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 ⌈

Type:	draft
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] Communication Management shall protect the transmission of data using E2E protocol, hidden behind the event API. ⌈

Type:	draft
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) are detected by Communication Management.
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:	–

⌋(RS_Main_01002, RS_Main_00060, RS_Main_00140, RS_Main_00010)

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

Type:	draft
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 ⌈

Type:	draft
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] Communication Management shall queue received events ⌈

Type:	draft
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] Communication Management shall provide the E2E information to the application that polls received events and to the application that is triggered on reception of an event. ⌈

Type:	draft
Description:	Communication Management shall provide the E2E information to the application that polls received events and to the application that is triggered on reception of an event. This interface shall be cleanly encapsulated in the interface, so that the API usage is identical only with the exception of the E2E information.
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)*

4.2.2.3.2   Methods

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

Type:	draft
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 ⌈

Type:	draft
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 ⌈

Type:	draft
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] Communication Management shall protect the transmission of methods using E2E protocol. ⌈

Type:	draft
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) are detected at the client side by Communication Management.
Dependencies:	–
Use Case:	E2E protected method calls in client-server based communication
Supporting Material:	–

⌋*(RS_Main_01002, RS_Main_00060, RS_Main_00010, RS_Main_00140)*

[RS_CM_00401] Communication Management shall provide the E2E information to the application that is triggered to run the called service method. ⌈

Type:	draft
Description:	Communication Management shall provide the E2E information to the application that has received the request for the method call. This interface shall be cleanly encapsulated, so that the API usage is identical only with the exception of the E2E information.
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_CM_00402] Communication Management shall support a decision for applying the method call based on E2E results. ⌈

Type:	draft
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] Communication Management shall provide an interface to call fire&forget service methods ⌈

Type:	draft
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] Communication Management shall provide an interface to query the result of an asynchronously called service method ⌈

Type:	draft
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 ⌈

Type:	draft
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] 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. ⌈

Type:	draft
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_CM_00404] Communication Management shall provide the E2E information to the application that is triggered after completion of a called service method. ⌈

Type:	draft
Description:	Communication Management shall provide the E2E information to the application that has sent the request for the method call. This interface shall be cleanly encapsulated, so that the API usage is identical only with the exception of the E2E information. The provided E2E information shall be E2E status, E2E state, E2E counter, Data ID, CRC, hash over the deserialized object.
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)*

4.2.2.3.3 Fields

[RS_CM_00216] 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 ⌈

Type:	draft
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] Communication Management shall trigger the application on reception of a notification on field value change ⌈

Type:	draft
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 ⌈

Type:	draft
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 ⌈

Type:	draft
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] 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 ⌈

Type:	draft
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 event shall provide the field value to the client immediately after the first subscription.
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] Communication Management shall provide a method to send notifications on data change to other applications ⌈

Type:	draft
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 ⌈

Document ID 716: AUTOSAR_RS_CommunicationManagement
— AUTOSAR CONFIDENTIAL —

Type:	draft
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 ⌈

Type:	draft
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 RESTful Communication

4.2.3.1 Communication between applications

[RS_CM_00300] The Communication Management shall provide a framework to support the RESTful communication paradigm introduced by [3]. ⌈

Type:	draft
Description:	The Communication Management shall provide a framework to support RESTful communication where there is no information about clients tracked by the server.
Rationale:	RESTful communication is stateless by it's nature. The server application does not track any state of its clients. To get a detailed introduction of the RESTful communication paradigm see [3].

▽

△

Dependencies:	–
Use Case:	Exchange data in a data-oriented way between adaptive applications.
Supporting Material:	–

⌋(*RS_Main_00140*, *RS_Main_01003*)

[RS_CM_00301] The Communication Management shall provide an abstraction of network protocols for RESTful services. ⌈

Type:	draft
Description:	The Communication Management shall provide an abstraction of network protocols for RESTful services.
Rationale:	This abstraction is needed to ensure portability of the adaptive applications. For example, the same RESTful adaptive application can be used with HTTP/1.1 or a IPC binding without changing any application code.
Dependencies:	–
Use Case:	Portability of adaptive applications.
Supporting Material:	–

⌋(*RS_Main_00140*, *RS_Main_00505*)

[RS_CM_00304] The Communication Management shall support URIs according to RFC3986 [4] to identify data. ⌈

Type:	draft
Description:	An (relative) URI shall act as a global identifier to data. The URI shall use the syntax of RFC3986 [4]
Rationale:	Individual resources are identified in requests using URIs as resource identifiers. The resources themselves are conceptually separate from the representations that are returned to the client. For example, the server does not send its database, but rather, some HTML, XML or JSON that represents some database records.
Dependencies:	–
Use Case:	Identify RESTful service resources.
Supporting Material:	–

⌋(*RS_Main_00060*, *RS_Main_00140*)

[RS_CM_00309] The Communication Management shall provide a way to match requests to corresponding server handlers and vice versa. ⌈

Type:	draft
Description:	For each client request there may be an own function on server side which handles the request. The client side shall be able to match the response to the previously send request.
Rationale:	−
Dependencies:	−
Use Case:	Routing of RESTful requests to functions implemented by an RESTful adaptive application.
Supporting Material:	−

⌋(*RS_Main_00140*)

[RS_CM_00310] The Communication Management shall provide an interface to install request handlers. ⌈

Type:	draft
Description:	For each client request there may be an own function on server side which handles the request. The server shall be able to register multiple handler functions.
Rationale:	−
Dependencies:	−
Use Case:	Routing of RESTful requests to functions implemented by an RESTful adaptive application.
Supporting Material:	−

⌋(*RS_Main_00060*, *RS_Main_00140*)

[RS_CM_00311] The Communication Management shall provide type aliases for abstraction of standard C++ components. ⌈

Type:	draft
Description:	
Rationale:	Standard C++ components may be critical in terms of resource control and safety. Therefore an abstraction shall be present.
Dependencies:	−
Use Case:	−
Supporting Material:	−

⌋(*RS_Main_00060*, *RS_Main_00140*)

4.2.3.2 Network binding

[RS_CM_00302] The Communication Management shall provide the means to offer RESTful services via multiple network protocols simultaneously. ⌈

Type:	draft
Description:	The Communication Management shall provide the means to offer RESTful services via multiple network protocols simultaneously.
Rationale:	An adaptive server application can offer it's RESTful service over multiple network bindings at the same time (e.g. IPC and HTTP/1.1).
Dependencies:	–
Use Case:	Use different network bindings for the same adaptive server application at the same time.
Supporting Material:	–

⌋(*RS_Main_00140*, *RS_Main_00505*)

[RS_CM_00314] The Communication Management shall provide Websockets to establish event communication. ⌈

Type:	draft
Description:	The event-based communication between RESTful services shall be realized with the Websocket protocol. See RFC 6455 [5] for details.
Rationale:	–
Dependencies:	–
Use Case:	Transport RESTful events between adaptive applications located on different machines.
Supporting Material:	–

⌋(*RS_Main_00505*, *RS_Main_01007*)

[RS_CM_00312] The Communication Management shall provide HTTP/1.1 to transport RESTful requests and responses. ⌈

Type:	draft
Description:	The communication between RESTful services shall be realized with the HTTP/1.1 protocol. See RFC 2616 [6] for details.
Rationale:	
Dependencies:	–
Use Case:	Transport RESTful requests and replies between adaptive applications located on different machines.
Supporting Material:	–

⌋(*RS_Main_00505*)

[RS_CM_00313] The Communication Management shall provide a JSON-based serialization for the payload of RESTful requests and responses. ⌈

Type:	draft
Description:	The RESTful communication shall use JSON to de/serialize the payload. See RFC 7159 [7] for details.
Rationale:	–
Dependencies:	–
Use Case:	Transfer RESTful data payload between adaptive applications.
Supporting Material:	–

⌋(*RS_Main_00505*, *RS_Main_01007*)

4.2.3.3 Data representation

[RS_CM_00305] The Communication Management shall represent data as an tree of objects. ⌈

Type:	draft
Description:	All data shall be represented to the adaptive application as an Object in an tree-like Object Graph
Rationale:	The Object Graph represents the data in a tree-like structure which can be traversed and manipulated by RESTful adaptive applications.
Dependencies:	–
Use Case:	Abstraction of the used payload format (e.g. JSON or XML).
Supporting Material:	–

⌋(*RS_Main_00140*, *RS_Main_01003*)

[RS_CM_00306] The Communication Management shall provide an Object Graph which is independent of the used serialization format. ⌈

Type:	draft
Description:	The Object Graph shall be independent of the used serialization format.
Rationale:	The Object Graph shall be able to represent any tree-like data structure.
Dependencies:	–
Use Case:	Abstraction of the used payload format (e.g. JSON or XML).
Supporting Material:	–

⌋(*RS_Main_00140*)

[RS_CM_00307] The Communication Management shall provide an Object Graph where each Object is strongly typed. ⌈

Type:	draft
Description:	Every Object in the Object Graph shall be strongly typed.
Rationale:	–
Dependencies:	–
Use Case:	Abstraction of the used payload format (e.g. JSON or XML)
Supporting Material:	–

⌋(RS_Main_00140, RS_Main_01003)

[RS_CM_00308] The Communication Management shall provide methods to read and manipulate the Object Graph ⌈

Type:	draft
Description:	The Object Graph shall be modifiable by the applications
Rationale:	–
Dependencies:	–
Use Case:	Abstraction of the used payload format (e.g. JSON or XML)
Supporting Material:	–

⌋(RS_Main_00060, RS_Main_00140, RS_Main_01003)

4.3   Non-Functional Requirements (Qualities)

5 Requirements Tracing

The following table references the AUTOSAR main requirements specified in [8] 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_00222] [RS_CM_00223] [RS_CM_00224] [RS_CM_00400] [RS_CM_00401] [RS_CM_00402] [RS_CM_00403] [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_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_00304] [RS_CM_00308] [RS_CM_00310] [RS_CM_00311] [RS_CM_00400] [RS_CM_00401] [RS_CM_00402] [RS_CM_00403] [RS_CM_00404]

[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_00200]
		[RS_CM_00201]
		[RS_CM_00204]
		[RS_CM_00211]
		[RS_CM_00212]
		[RS_CM_00213]
		[RS_CM_00222]
		[RS_CM_00223]
		[RS_CM_00225]
		[RS_CM_00300]
		[RS_CM_00301]
		[RS_CM_00302]
		[RS_CM_00304]
		[RS_CM_00305]
		[RS_CM_00306]
		[RS_CM_00307]
		[RS_CM_00308]
		[RS_CM_00309]
		[RS_CM_00310]
		[RS_CM_00311]
		[RS_CM_00315]
		[RS_CM_00400]
		[RS_CM_00403]
[RS_Main_00505]	No description	[RS_CM_00301]
		[RS_CM_00302]
		[RS_CM_00312]
		[RS_CM_00313]
		[RS_CM_00314]
[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_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_Main_01003]	AUTOSAR shall support data-oriented communication	[RS_CM_00300] [RS_CM_00305] [RS_CM_00307] [RS_CM_00308]
[RS_Main_01007]	AUTOSAR communication shall assure quality of service on communication	[RS_CM_00313] [RS_CM_00314]

6 References

[1] Standardization Template
AUTOSAR_TPS_StandardizationTemplate

[2] Glossary
AUTOSAR_TR_Glossary

[3] REST: Architectural Styles and the Design of Network-based Software Architectures

[4] RFC 3986, Uniform Resource Identifier (URI): Generic Syntax

[5] RFC 6455, The WebSocket Protocol

[6] RFC 2616, Hypertext Transfer Protocol – HTTP/1.1

[7] RFC 7159, The JavaScript Object Notation (JSON) Data Interchange Format

[8] Main Requirements
AUTOSAR_RS_Main