INTERNATIONAL STANDARD

ISO 20080

First edition
2019-03

AMENDMENT 1
2021-11

Road vehicles — Information for remote diagnostic support — General requirements, definitions and use cases

AMENDMENT 1

Véhicules routiers — Information pour support de diagnostic à distance — Exigences générales, définitions et cas d'utilisation

AMENDEMENT 1

Reference number
ISO 20080:2019/Amd.1:2021(E)

ISO copyright office
CP 401 • Ch. de Blandonnet 8
CH-1214 Vernier, Geneva
Phone: +41 22 749 01 11
Email: copyright@iso.org
Website: www.iso.org

Published in Switzerland

Foreword

ISO (the International Organization for Standardization) is a worldwide federation of national standards bodies (ISO member bodies). The work of preparing International Standards is normally carried out through ISO technical committees. Each member body interested in a subject for which a technical committee has been established has the right to be represented on that committee. International organizations, governmental and non-governmental, in liaison with ISO, also take part in the work. ISO collaborates closely with the International Electrotechnical Commission (IEC) on all matters of electrotechnical standardization.

The procedures used to develop this document and those intended for its further maintenance are described in the ISO/IEC Directives, Part 1. In particular, the different approval criteria needed for the different types of ISO documents should be noted. This document was drafted in accordance with the editorial rules of the ISO/IEC Directives, Part 2 (see www.iso.org/directives).

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO shall not be held responsible for identifying any or all such patent rights. Details of any patent rights identified during the development of the document will be in the Introduction and/or on the ISO list of patent declarations received (see www.iso.org/patents).

Any trade name used in this document is information given for the convenience of users and does not constitute an endorsement.

For an explanation of the voluntary nature of standards, the meaning of ISO specific terms and expressions related to conformity assessment, as well as information about ISO's adherence to the World Trade Organization (WTO) principles in the Technical Barriers to Trade (TBT) see www.iso.org/iso/foreword.html.

This document was prepared by Technical Committee ISO/TC 22, *Road vehicles*, Subcommittee SC 31, *Data communication*.

Any feedback or questions on this document should be directed to the user's national standards body. A complete listing of these bodies can be found at www.iso.org/members.html.

Road vehicles — Information for remote diagnostic support — General requirements, definitions and use cases

AMENDMENT 1

Annex A

Replace Annex A with the following:

Annex A
(informative)

Implementation based on ISO 20078 – Road Vehicles – Extended Vehicle (ExVe) – Web Services

A.1 Introduction

A.1.1 General

This annex contains a web service specification of the use cases listed in this document. The web service specification is based on the ISO 20078:2021 series. All web services are defined as REST APIs, using JSON for the transfer of content.

The ISO 20078:2021 series is indispensable for implementing web services according to this annex.

A.1.2 Security

All REST APIs are using OAuth2 compatible framework for access control, OpenID Connect compatible framework for identification purposes and HTTPS for securing the transfer, see ISO 20078-3:2021 for details. The exact details of how to obtain access is described by each offering party.

A.1.3 Error codes

A.1.3.1 ISO 20078

The HTTP status codes (error codes) listed for each REST API are described in ISO 20078-2:2021.

A.1.3.2 General error conditions

General error conditions are valid for all use cases. Table A.1 presents the mapping of error conditions in 5.5 to REST API errors:

Table A.1 — Mapping of error conditions to REST API

Error condition	HTTP status code	ExveErrorId	Example
Request currently not possible to perform by the ExVe	503	20080-1000	{ "dtcReadout": { "id": "abcde-12345-ghjke-67474", "messageTimestamp": "2016-02-24T09:23:46Z", "exveErrorId": "20080-1000", "exveErrorMsg": "Request currently not possible to perform by the ExVe", "vehicleId": "12345678909876543" } }

A.1.3.3 Use case specific error conditions

Use case specific errors are mapped to HTTP status codes in each REST API.

A.2 Resources

A web service is exposing access to one or more resources. To be able to access a resource through a web service, access needs to be granted. This can be done either directly to the resource or through a container.

Table A.2 maps the ISO 20080 (this document) use cases to REST APIs and resources. In some cases, mapping of a use case to a REST API provides little standardization benefit, as it is highly offering party specific.

Table A.2 — Mapping of use cases to REST APIs

UC	Use case name	REST API	Resource(s)	Comment
01	Use case discovery	resourceReadout	Not applicable	
02	Identify ECUs installed in the vehicle	ecuReadouts	ECU readout	
03	Read diagnostic trouble codes (DTCs)	dtcReadouts	DTC readout	
04	Read readiness codes	readinessCodeReadouts	Readiness code readout	
05	Read DTC snapshot data	dtcSnapshotReadout	DTC snapshot readout	
06	Read selected diagnostic parametric dynamic data	parameterReadout	Parameter readout	
07	Read malfunction indicator status	malfunctionIndicatorReadout	Malfunction indicator readout	
08	Clear DTCs	clearDtcJob	Clear DTC job	
09	Adjust the settings of a selected system	Not applicable/no standardized API due to differences between offering parties.	Not applicable	System setting input and result are offering party specific.
10	Activation of actuators	Not applicable/no standardized API due to differences between offering parties.	Not applicable	Actuator input and result are offering party specific.
11	Activate a self-test routine	Not applicable/no standardized API due to differences between offering parties.	Not applicable	Self-test input and result are offering party specific.

ISO 20078-2:2021 defines new syntax for resource versioning (used in Accept and Content-Type HTTP headers). Existing implementations of ISO 20080 based on the ISO 20078:2019 series version can still be used. For new implementations, support of the ISO 20078:2021 series is recommended (see Table A.3 for details).

Table A.3 — Mapping of REST API resource representation used in ISO 20080:2019 and ISO 20080:2019/Amd.1:2021 versions

UC	REST API	Resource version according to ISO 20078-2:2019	Resource version according to ISO 20078-2:2021
01	resourceReadout	application/x.exve.org.resourcereadout.v1+json	application/json; exve-resourceversion= resourcereadout.v1.0
02	ecuReadouts	application/x.exve.org.ecureadout.v1+json	application/json; exve-resourceversion= ecureadout.v1.0
03	dtcReadouts	application/x.exve.org.dtcreadout.v1+json	application/json; exve-resourceversion= dtcreadout.v1.0
04	readinessCodeReadouts	application/x.exve.org.readinesscodereadout.v1+json	application/json; exve-resourceversion= readinesscodereadout.v1.0
05	dtcSnapshotReadout	application/x.exve.org.dtcsnapshotreadout.v1+json	application/json; exve-resourceversion= dtcsnapshotreadout.v1.0
06	parameterReadout	application/x.exve.org.parameterreadout.v1+json	application/json; exve-resourceversion= parameterreadout.v1.0
07	malfunctionIndicatorReadout	application/x.exve.org.malfunctionIndicatorreadout.v1+json	application/json; exve-resourceversion= malfunctionindicatorreadout.v1.0
08	clearDtcJob	application/x.exve.org.cleardtcjob.v1+json	application/json; exve-resourceversion= cleardtcjob.v1.0

A.3 REST API information model

The rest API information model in Figure A.1 is compiled from the use cases in this document and used as a base for designing the REST APIs.

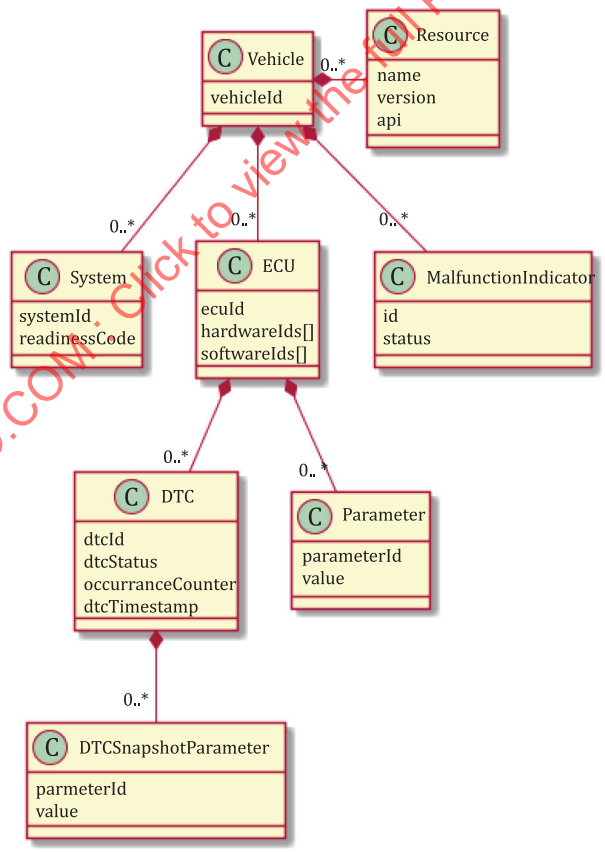

Figure A.1 — REST API information model

A.4 REST APIs

A.4.1 Use case 01 – resourceReadout

The resourceReadout API follows the asynchronous interaction pattern, as the processing time of the request can vary dependent on the offering party implementation (see Figure A.2). In some cases, it is possible to return the result immediately, whereas sometimes the accessing party needs to poll the API until the result is ready.

The accessing party starts by posting a resource readout request (see Table A.4). If the result is available immediately, the result is returned directly (see Table A.5). If the result is not available immediately, a status is returned instead of the readout. The accessing party is supposed to poll the request status until the processing is completed. When the processing is completed, the accessing party will receive the result.

The readout status and the completed readout will be available for a limited time after being created. This time is specified by the offering party.

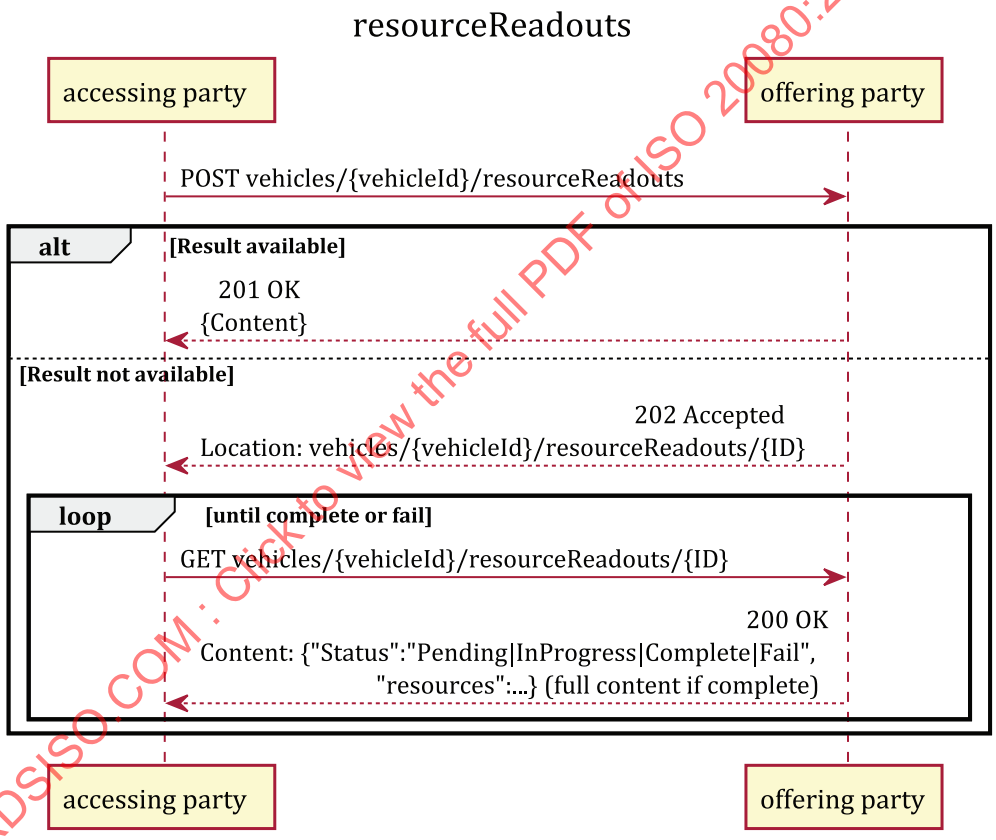

Figure A.2 — resourceReadouts sequence diagram

Table A.4 — POST resourceReadout

POST vehicles/{vehicleId}/resourceReadouts				
Description	This API creates a readout of available resources to the accessing party for one vehicle. If the result is available immediately, the result is returned. If the result is not available, a location to the resource readout is returned. This location shall be polled until the result is available.			
Parameters	vehicleId	string	required	The vehicle identifier of the vehicle to read from

Table A.4 *(continued)*

<table>
<tr><td rowspan="3">Request headers</td><td colspan="2">Host</td><td>required</td><td>According to HTTP/1.1 RFC 2616</td></tr>
<tr><td colspan="2">Authorization</td><td>required</td><td>Bearer {token}</td></tr>
<tr><td colspan="2">Accept</td><td>required</td><td>application/json; exve-resourceversion=resourcereadout.v1.0; charset=utf-8</td></tr>
<tr><td>Response headers</td><td colspan="2">Location</td><td colspan="2">Absolute URI of the /resourceReadouts endpoint</td></tr>
<tr><td rowspan="11">Response (success)</td><td>201</td><td colspan="3">Example (result available immediately):
{
"resourceReadout": {
"id": "abcde-12345-ghjke-67474",
"asyncStatus": "Complete",
"messageTimestamp": "2016-02-24T09:23:46Z",
"vehicleId": "12345678909876543",
"resources": [
{
"name": "DTC Readout",
"version": "1",
"api":
"https://example.org/vehicles/12345678909876543/dtcReadouts"
},
{
"name": "ECU Readout",
"version": "1",
"api":
"https://example.org/vehicles/12345678909876543/ecuReadouts"
}
]
}
}</td></tr>
<tr><td>202</td><td colspan="3">Result is not available immediately. Location of resourceReadout will be returned, e.g. vehicles/{vehicleId}/resourceReadouts/{id}, see Location header.</td></tr>
<tr><td rowspan="9">Error codes</td><td>400</td><td colspan="3">Bad Request</td></tr>
<tr><td>401</td><td colspan="3">Unauthorized</td></tr>
<tr><td>403</td><td colspan="3">Forbidden</td></tr>
<tr><td>404</td><td colspan="3">Not Found</td></tr>
<tr><td>406</td><td colspan="3">Not Acceptable</td></tr>
<tr><td>500</td><td colspan="3">Internal Server Error</td></tr>
<tr><td>501</td><td colspan="3">Not Implemented</td></tr>
<tr><td>503</td><td colspan="3">Service Unavailable</td></tr>
<tr><td>505</td><td colspan="3">Version Not Supported</td></tr>
<tr><td>Access</td><td colspan="2">Any resource</td><td colspan="2">Access to any resource will give access to this API.</td></tr>
<tr><td>JSON schema</td><td colspan="4">See Clause A.5.</td></tr>
</table>

Table A.5 — GET resourceReadout

<table>
<tr><td colspan="6">GET vehicles/{vehicleId}/resourceReadouts/{id}</td></tr>
<tr><td>Description</td><td colspan="5">This API returns a readout of available resources to the accessing party for one vehicle. The id of the readout is returned when posting the request. If the resource readout is not completed, the readout status is returned. If the resource readout is completed, the result is returned.</td></tr>
<tr><td rowspan="2">Parameters</td><td>vehicleId</td><td>string</td><td>required</td><td colspan="2">The vehicle identifier of the vehicle to read from</td></tr>
<tr><td>id</td><td>string</td><td>required</td><td colspan="2">Id of the DTC readout</td></tr>
<tr><td rowspan="3">Request headers</td><td>Host</td><td>required</td><td colspan="3">According to HTTP/1.1 RFC 2616</td></tr>
<tr><td>Authorization</td><td>required</td><td colspan="3">Bearer {token}</td></tr>
<tr><td>Accept</td><td>required</td><td colspan="3">application/json; exve-resourceversion=resourcereadout.v1.0; charset=utf-8</td></tr>
<tr><td>Response headers</td><td>Content-Type</td><td colspan="4">application/json; exve-resourceversion= resourcereadout.v1.0;charset=utf-8</td></tr>
<tr><td>Response (success)</td><td>200</td><td colspan="4">Example (result available):
<pre>{
"resourceReadout": {
   "id": "abcde-12345-ghjke-67474",
   "asyncStatus": "Complete",
   "messageTimestamp": "2016-02-24T09:23:46Z",
   "vehicleId": "12345678909876543",
   "resources": [
      {
      "name": "DTC Readout",
      "version": "1",
      "api":
"https://example.org/vehicles/12345678909876543/dtcReadouts"
      },
      {
      "name": "ECU Readout",
      "version": "1",
      "api":
"https://example.org/vehicles/12345678909876543/ecuReadouts"
      }
      ]
   }
}</pre></td></tr>
</table>

Table A.5 *(continued)*

		Example (result not available):
		{
		"resourceReadout": {
		"id": "abcde-12345-ghjke-67474",
		"asyncStatus": "InProgress",
		"asyncWait": 10000,
		"asyncEstimatedComplete": "2016-02-24T09:24:00Z",
		"messageTimestamp": "2016-02-24T09:23:46Z",
		"vehicleId": "12345678909876543"
		}
		
		}

Error codes	400	Bad Request
	401	Unauthorized
	403	Forbidden
	404	Not Found
	406	Not Acceptable
	500	Internal Server Error
	501	Not Implemented
	503	Service Unavailable
	505	Version Not Supported
Access	Any resource	Access to any resource will give access to this API.
JSON schema	See Clause A.5.	

A.4.2 Use case 02 – ecuReadouts

The ecuReadout API follows the asynchronous interaction pattern, as the processing time of the request can vary dependent on the offering party implementation (see Figure A.3). In some cases, it is possible to return the result immediately, whereas sometimes the accessing party needs to poll the API until the result is ready.

The accessing party starts by posting an ECU readout request (see Table A.6). If the result is available immediately, the result is returned directly. If the result is not available immediately, a status is returned instead of the readout. The accessing party is supposed to poll the request status until the processing is completed (see Table A.7). When the processing is completed, the accessing party will receive the result.

The readout status and the completed readout will be available for a limited time after being created. This time is specified by the offering party.

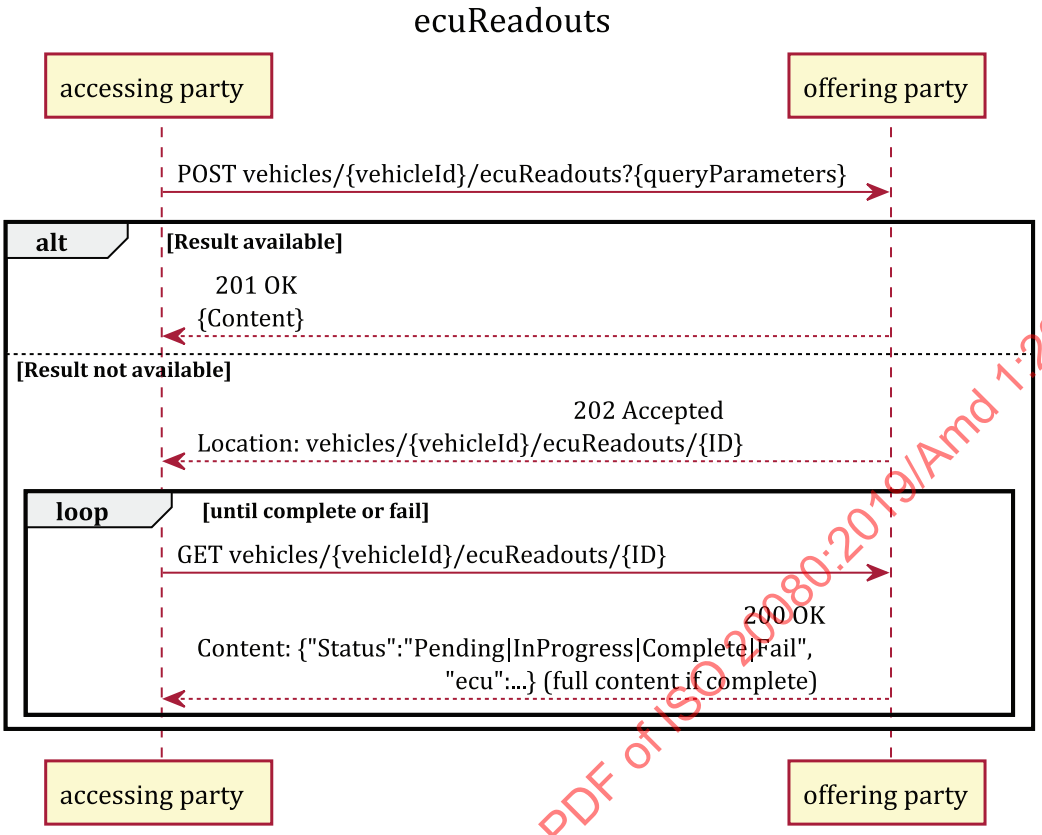

Figure A.3 — ecuReadouts sequence diagram

Table A.6 — POST ecuReadout

POST vehicles/{vehicleId}/ecuReadouts/ecuId={ecuId}				
Description	This API creates a readout of ECUs for one vehicle. If the result is available immediately, the result is returned. If the result is not available, a location to the ECU readout is returned. This location shall be polled until the result is available.			
Parameters	vehicleId	string	required	The vehicle identifier of the vehicle to read from
	ecuId	string	optional	Return this ECU id only Default: Return all ECUs
Request headers	Host	required	According to HTTP/1.1 RFC 2616	
	Authorization	required	Bearer {token}	
	Accept	required	application/json; exve-resourceversion=ecureadout.v1.0; charset=utf-8	
Response headers	Location	Absolute URI of the /ecuReadouts endpoint		

Table A.6 *(continued)*

<table>
<tr><td>Response (success)</td><td>201</td><td>Example (result available immediately):
<pre>{
"ecuReadout": {
   "id": "abcde-12345-ghjke-67474",
   "asyncStatus": "Complete",
   "messageTimestamp": "2016-02-24T09:23:46Z",
   vehicleId": "12345678909876543",
   "receivedTimestamp": "2016-02-24T09:23:46Z",
   "ecus": [
         {
         "ecuId": "ABC",
         "hardwareIds": ["1234567"],
         "softwareIds": ["9876543"]
      }, {
         "ecuId": "DEF",
         "hardwareIds": ["2345678"],
         "softwareIds": ["8976543"]
      }, {
         "ecuId": "GHI",
         "hardwareIds": ["3456789"],
         "softwareIds": ["7896543","7896555"]
         }
      ]
   }
}</pre></td></tr>
<tr><td></td><td>202</td><td>Result is not available immediately. Location of ecuReadout will be returned, e.g. vehicles/{vehicleId}/ecuReadouts/{id}, see Location header.</td></tr>
<tr><td>Error codes</td><td>400</td><td>Bad Request</td></tr>
<tr><td></td><td>401</td><td>Unauthorized</td></tr>
<tr><td></td><td>403</td><td>Forbidden</td></tr>
<tr><td></td><td>404</td><td>Not Found</td></tr>
<tr><td></td><td>406</td><td>Not Acceptable</td></tr>
<tr><td></td><td>500</td><td>Internal Server Error</td></tr>
<tr><td></td><td>501</td><td>Not Implemented</td></tr>
<tr><td></td><td>503</td><td>Service Unavailable</td></tr>
<tr><td></td><td>505</td><td>Version Not Supported</td></tr>
<tr><td>Access</td><td colspan="2">ECU readout</td><td>Full access to this API</td></tr>
<tr><td>JSON schema</td><td colspan="3">See Clause A.5.</td></tr>
</table>

Table A.7 — GET ecuReadout

<table>
<tr><td colspan="6">GET vehicles/{vehicleId}/ecuReadouts/{id}</td></tr>
<tr><td>Description</td><td colspan="5">This API returns the ECU readout for one vehicle. The id of the readout is returned when posting the request. If the ECU readout is not completed, the readout status is returned. If the ECU readout is completed, the result is returned.</td></tr>
<tr><td rowspan="2">Parameters</td><td>vehicleId</td><td>string</td><td>required</td><td colspan="2">The vehicle identifier of the vehicle to read from</td></tr>
<tr><td>id</td><td>string</td><td>required</td><td colspan="2">id of the ECU readout</td></tr>
<tr><td rowspan="3">Request headers</td><td>Host</td><td>required</td><td colspan="3">According to HTTP/1.1 RFC 2616</td></tr>
<tr><td>Authorization</td><td>required</td><td colspan="3">Bearer {token}</td></tr>
<tr><td>Accept</td><td>required</td><td colspan="3">application/json; exve-resourceversion=ecureadout.v1.0; charset=utf-8</td></tr>
<tr><td>Response headers</td><td>Content-Type</td><td colspan="4">application/json; exve-resourceversion=ecureadout.v1.0; charset=utf-8</td></tr>
<tr><td>Response (success)</td><td>200</td><td colspan="4">Example (result available):
<pre>{
   "ecuReadout": {
      "id": "abcde-12345-ghjke-67474",
      "asyncStatus": "Complete",
      "messageTimestamp": "2016-02-24T09:23:46Z",
      "vehicleId": "12345678909876543",
      "receivedTimestamp": "2016-02-24T09:23:46Z",
      "ecus": [
         {
            "ecuId": "ABC",
            "hardwareIds": ["1234567"],
            "softwareIds": ["9876543"]
         }, {
            "ecuId": "DEF",
            "hardwareIds": ["2345678"],
            "softwareIds": ["8976543"]
         }, {
            "ecuId": "GHI",
            "hardwareIds": ["3456789"],
            "softwareIds": ["7896543","7896555"]
         }
      ]
   }
}</pre></td></tr>
</table>

Table A.7 *(continued)*

		Example (result not available):
		{
		"ecuReadout": {
		"id": "abcde-12345-ghjke-67474",
		"asyncStatus": "InProgress",
		"asyncWait": 10000,
		"asyncEstimatedComplete": "2016-02-24T09:24:00Z",
		"messageTimestamp": "2016-02-24T09:23:46Z",
		"vehicleId": "12345678909876543"
		}
		}

Error codes	400	Bad Request
	401	Unauthorized
	403	Forbidden
	404	Not Found
	406	Not Acceptable
	500	Internal Server Error
	501	Not Implemented
	503	Service Unavailable
	505	Version Not Supported
Access	ECU readout	Full access to this API
JSON schema	See Clause A.5.	

A.4.3 Use case 03 – dtcReadouts

The dtcReadout API follows the asynchronous interaction pattern, as the processing time of the request can vary dependent on the offering party implementation (see Figure A.4). In some cases, it is possible to return the result immediately, whereas sometimes the accessing party needs to poll the API until the result is ready.

The accessing party starts by posting a DTC readout request (see Table A.8). If the result is available immediately, the result is returned directly. If the result is not available immediately, a status is returned instead of the readout. The accessing party is supposed to poll the request status until the processing is completed (see Table A.9). When the processing is completed, the accessing party will receive the result.

The readout status and the completed readout will be available for a limited time after being created. This time is specified by the offering party.

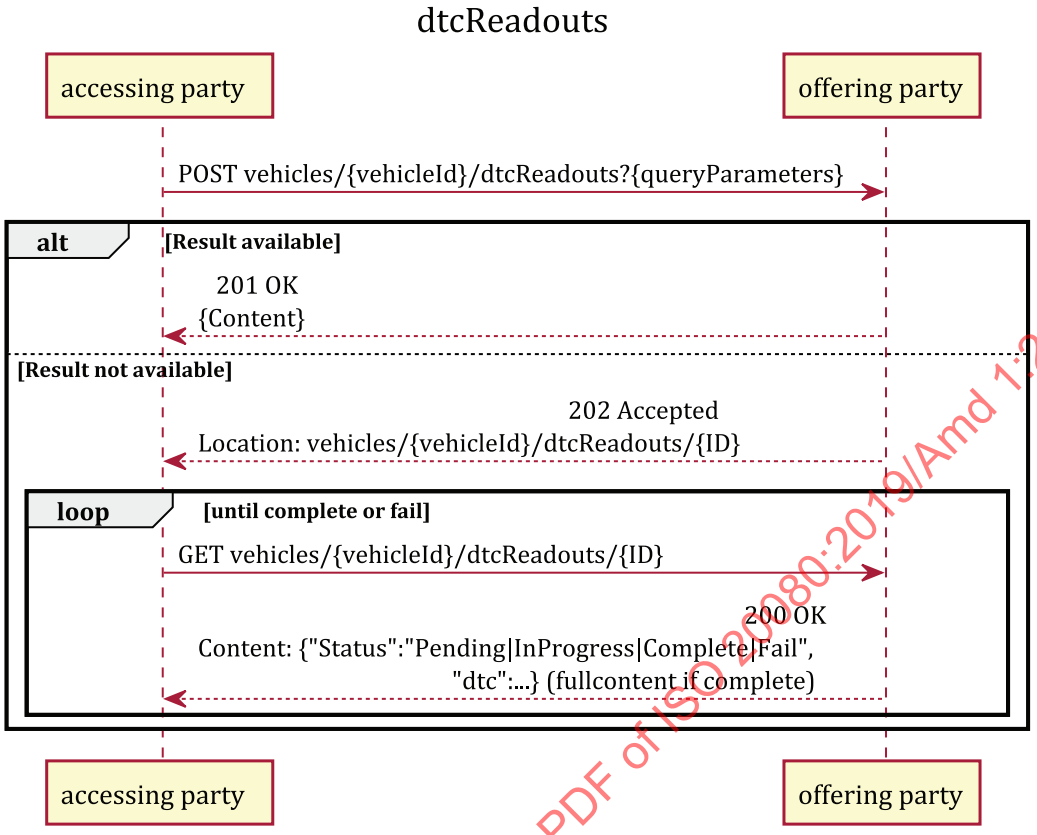

Figure A.4 — dtcReadouts sequence diagram

Table A.8 — POST dtcReadout

POST vehicles/{vehicleId}/dtcReadouts?ecuId={ecuId}&dtcStatus={dtcStatus}				
Description	This API creates a readout of DTCs for one vehicle. If the result is available immediately, the result is returned. If the result is not available, a location to the DTC readout is returned. This location shall be polled until the result is available.			
Parameters	vehicleId	string	required	The vehicle identifier of the vehicle to read from
	ecuId	string	optional	Returns DTCs from this ECU id only Default: Return DTCs from all ECUs
	dtcStatus	string	required	Returns DTCs with this status only.
Request headers	Host	required	According to HTTP/1.1 RFC 2616	
	Authorization	required	Bearer {token}	
	Accept	required	application/json; exve-resourceversion=dtcreadout.v1.0; charset=utf-8	
Response headers	Location	Absolute URI of the /dtcReadouts endpoint		

Table A.8 *(continued)*

<table>
<tr><td>Response (success)</td><td>201</td><td colspan="2">Example (result available immediately):
{
"dtcReadout": {
"id": "abcde-12345-ghjke-67474",
"asyncStatus": "Complete",
"messageTimestamp": "2016-02-24T09:23:46Z",
"vehicleId": "12345678909876543",
"receivedTimestamp": "2016-02-24T09:23:46Z",
"dtcs": [
{
"dtcId": "123456",
"status": "ACTIVE",
"occurrenceCounter": 3,
"ecuId": "ABC",
"dtcTimestamp": "2016-01-20T08:23:46Z"
},
{
"dtcId": "345678",
"status": "ACTIVE",
"occurrenceCounter": 1,
"ecuId": "ABC",
"dtcTimestamp": "2016-01-07T14:56:10Z"
}
]
}
}</td></tr>
<tr><td></td><td>202</td><td colspan="2">Result is not available immediately. Location of dtcReadout will be returned, e.g. vehicles/{vehicleId}/dtcReadouts/{id}, see Location header.</td></tr>
<tr><td>Error codes</td><td>400</td><td>Bad Request</td><td>DTC status not valid</td></tr>
<tr><td></td><td>401</td><td colspan="2">Unauthorized</td></tr>
<tr><td></td><td>403</td><td colspan="2">Forbidden</td></tr>
<tr><td></td><td>404</td><td>Not Found</td><td>ECU Id not valid</td></tr>
<tr><td></td><td>406</td><td colspan="2">Not Acceptable</td></tr>
<tr><td></td><td>500</td><td colspan="2">Internal Server Error</td></tr>
<tr><td></td><td>501</td><td colspan="2">Not Implemented</td></tr>
<tr><td></td><td>503</td><td colspan="2">Service Unavailable</td></tr>
<tr><td></td><td>505</td><td colspan="2">Version Not Supported</td></tr>
<tr><td>Access</td><td colspan="2">DTC readout</td><td>Full access to this API</td></tr>
<tr><td>JSON schema</td><td colspan="3">See Clause A.5.</td></tr>
</table>

Table A.9 — GET dtcReadout

<table>
<tr><td colspan="6">GET vehicles/{vehicleId}/dtcReadouts/{id}</td></tr>
<tr><td>Description</td><td colspan="5">This API returns the DTC readout for one vehicle. The id of the readout is returned when posting the request. If the DTC readout is not completed, the readout status is returned. If the DTC readout is completed, the result is returned.</td></tr>
<tr><td rowspan="2">Parameters</td><td>vehicleId</td><td>string</td><td>required</td><td colspan="2">The vehicle identifier of the vehicle to read from</td></tr>
<tr><td>id</td><td>string</td><td>required</td><td colspan="2">id of the DTC readout</td></tr>
<tr><td rowspan="3">Request headers</td><td>Host</td><td>required</td><td colspan="3">According to HTTP/1.1 RFC 2616</td></tr>
<tr><td>Authorization</td><td>required</td><td colspan="3">Bearer {token}</td></tr>
<tr><td>Accept</td><td>required</td><td colspan="3">application/json; exve-resourceversion=dtcreadout.v1.0; charset=utf-8</td></tr>
<tr><td>Response headers</td><td>Content-Type</td><td colspan="4">application/json; exve-resourceversion=dtcreadout.v1.0; charset=utf-8</td></tr>
<tr><td>Response (success)</td><td>200</td><td colspan="4">Example (result available):
<pre>{
   "dtcReadout": {
      "id": "abcde-12345-ghjke-67474",
      "asyncStatus": "Complete",
      "messageTimestamp": "2016-02-24T09:23:46Z",
      "vehicleId": "12345678909876543",
      "receivedTimestamp": "2016-02-24T09:23:46Z",
      "dtcs": [
         {
            "dtcId": "123456",
            "status": "ACTIVE",
            "occurrenceCounter": 3,
            "ecuId": "ABC",
            "dtcTimestamp": "2016-01-20T08:23:46Z"
         },
         {
            "dtcId": "345678",
            "status": "ACTIVE",
            "occurrenceCounter": 1,
            "ecuId": "ABC",
            "dtcTimestamp": "2016-01-07T14:56:10Z"
         }
      ]
   }
}</pre></td></tr>
</table>

Table A.9 *(continued)*

		Example (result not available):
		{
		"dtcReadout": {
		"id": "abcde-12345-ghjke-67474",
		"asyncStatus": "InProgress",
		"asyncWait": 10000,
		"asyncEstimatedComplete": "2016-02-24T09:24:00Z",
		"messageTimestamp": "2016-02-24T09:23:46Z",
		"vehicleId": "12345678909876543"
		}
		}

Error codes	400	Bad Request
	401	Unauthorized
	403	Forbidden
	404	Not Found
	406	Not Acceptable
	500	Internal Server Error
	501	Not Implemented
	503	Service Unavailable
	505	Version Not Supported
Access	DTC readout	Full access to this API
JSON schema	See Clause A.5.	

A.4.4 Use case 04 – readinessCodeReadouts

The readinessCodeReadout API follows the asynchronous interaction pattern, as the processing time of the request can vary dependent on the offering party implementation (see Figure A.5). In some cases, it is possible to return the result immediately, whereas sometimes the accessing party needs to poll the API until the result is ready.

The accessing party starts by posting a readiness code readout request (see Table A.10). If the result is available immediately, the result is returned directly. If the result is not available immediately, a status is returned instead of the readout. The accessing party is supposed to poll the request status until the processing is completed (see Table A.11). When the processing is completed, the accessing party will receive the result.

The readout status and the completed readout will be available for a limited time after being created. This time is specified by the offering party.

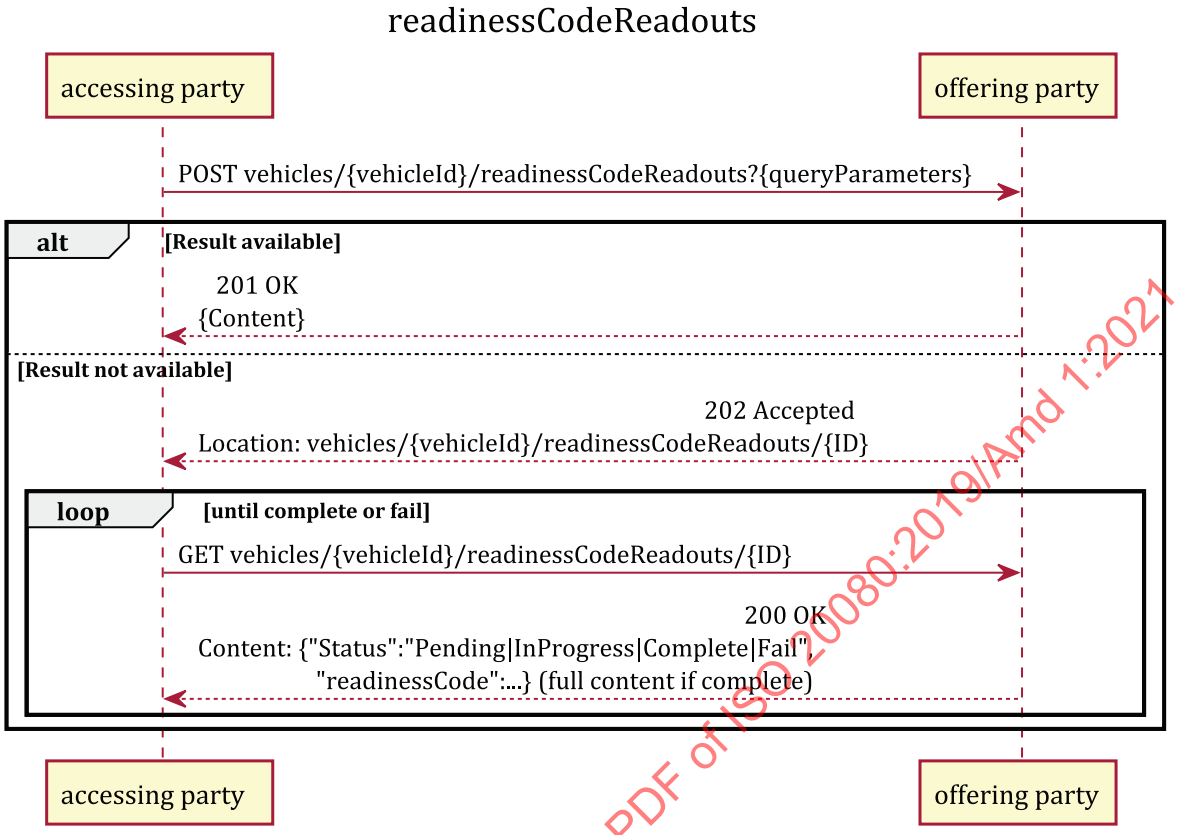

Figure A.5 — readinessCodeReadouts sequence diagram

Table A.10 — POST readinessCodeReadout

POST vehicles/{vehicleId}/readinessCodeReadouts				
Description	This API creates a readout of readiness codes for one vehicle. If the result is available immediately, the result is returned. If the result is not available, a location to the readiness code readout is returned. This location shall be polled until the result is available.			
Parameters	vehicleId	string	required	The vehicle identifier of the vehicle to read from
Request headers	Host	required	According to HTTP/1.1 RFC 2616	
	Authorization	required	Bearer {token}	
	Accept	required	application/json; exve-resourceversion= readinesscodereadout.v1.0; charset=utf-8	
Response headers	Location	Absolute URI of the /readinessCodeReadouts endpoint		

Table A.10 *(continued)*

Response (success)	201	Example (result available immediately):
		{
		"readinessCodeReadout": {
		"id": "abcde-12345-ghjke-67474",
		"asyncStatus": "Complete",
		"messageTimestamp": "2016-02-24T09:23:46Z",
		"vehicleId": "12345678909876543",
		"receivedTimestamp": "2016-02-24T09:23:46Z",
		"readinessCodes": [
		{
		"systemId": "EGR",
		"isReady": true
		}, {
		"systemId": "SCR",
		"isReady": false
		}
		]
		}
		}

	202	Result is not available immediately. Location of readinessCodeReadout will be returned, e.g. vehicles/{vehicleId}/readinessCodeReadouts/{id}, see Location header.
Error codes	400	Bad Request
	401	Unauthorized
	403	Forbidden
	404	Not Found
	406	Not Acceptable
	500	Internal Server Error
	501	Not Implemented
	503	Service Unavailable
	505	Version Not Supported
Access	Readiness code readout	Full access to this API
JSON schema	See Clause A.5.	

Table A.11 — GET readinessCodeReadout

<table>
<tr><td colspan="6">GET vehicles/{vehicleId}/readinessCodeReadouts/{id}</td></tr>
<tr><td>Description</td><td colspan="5">This API returns the readiness code readout for one vehicle. The id of the readout is returned when posting the request. If the readiness code readout is not completed, the readout status is returned. If the readiness code readout is completed, the result is returned.</td></tr>
<tr><td rowspan="2">Parameters</td><td>vehicleId</td><td>string</td><td>required</td><td colspan="2">The vehicle identifier of the vehicle to read from</td></tr>
<tr><td>id</td><td>string</td><td>required</td><td colspan="2">id of the readiness code readout</td></tr>
<tr><td rowspan="3">Request headers</td><td>Host</td><td>required</td><td colspan="3">According to HTTP/1.1 RFC 2616</td></tr>
<tr><td>Authorization</td><td>required</td><td colspan="3">Bearer {token}</td></tr>
<tr><td>Accept</td><td>required</td><td colspan="3">application/json; exve-resourceversion=readinesscodereadout.v1.0; charset=utf-8</td></tr>
<tr><td>Response headers</td><td>Content-Type</td><td colspan="4">application/json; exve-resourceversion=readinesscodereadout.v1.0; charset=utf-8</td></tr>
<tr><td>Response (success)</td><td>200</td><td colspan="4">Example (result available):
<pre>{
"readinessCodeReadout": {
   "id": "abcde-12345-ghjke-67474",
   "asyncStatus": "Complete",
   "messageTimestamp": "2016-02-24T09:23:46Z",
   "vehicleId": "12345678909876543",
   "receivedTimestamp": "2016-02-24T09:23:46Z",
   "readinessCodes": [
         {
         "systemId": "EGR",
         "isReady": true
      }, {
         "systemId": "SCR",
         "isReady": false
         }
      ]
   }
}</pre>Example (result not available):
<pre>{
"readinessCodeReadout": {
   "id": "abcde-12345-ghjke-67474",
   "asyncStatus": "InProgress",
   "asyncWait": 10000,
   "asyncEstimatedComplete": "2016-02-24T09:24:00Z",
   "messageTimestamp": "2016-02-24T09:23:46Z",
   "vehicleId": "12345678909876543"
   }
}</pre></td></tr>
</table>

Table A.11 *(continued)*

Error codes	400	Bad Request
	401	Unauthorized
	403	Forbidden
	404	Not Found
	406	Not Acceptable
	500	Internal Server Error
	501	Not Implemented
	503	Service Unavailable
	505	Version Not Supported
Access	Readiness code read-out	Full access to this API
JSON schema	See Clause A.5.	

A.4.5 Use case 05 – dtcSnapshotReadouts

The dtcSnapshotReadout API follows the asynchronous interaction pattern as the processing time of the request can vary dependent on the offering party implementation (see Figure A.6). In some cases, it is possible to return the result immediately, whereas sometimes the accessing party needs to poll the API until the result is ready.

The accessing party starts by posting a DTC snapshot readout request (see Table A.12). If the result is available immediately, the result is returned directly. If the result is not available immediately, a status is returned instead of the readout. The accessing party is supposed to poll the request status until the processing is completed (see Table A.13). When the processing is completed, the accessing party will receive the result.

The readout status and the completed readout will be available for a limited time after being created. This time is specified by the offering party.

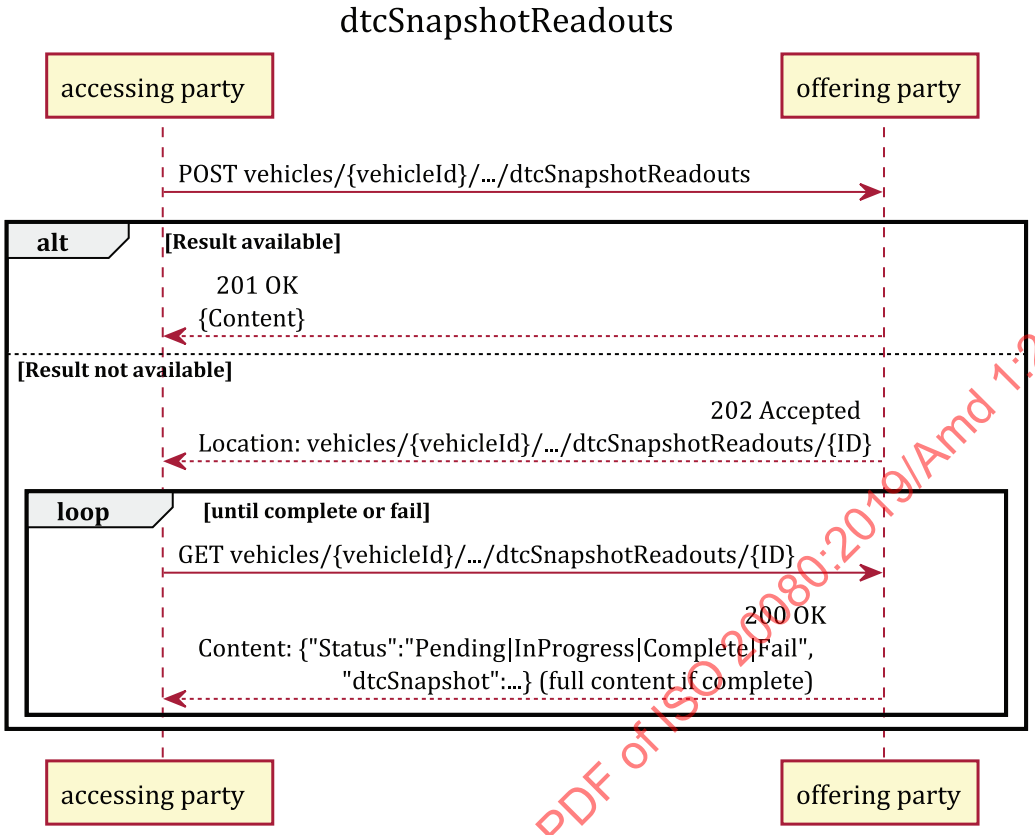

Figure A.6 — dtcSnapshotReadouts sequence diagram

Table A.12 — POST dtcSnapshotReadout

POST vehicles/{vehicleId}/ecuId/{ecuId}/dtcId/{dtcId}/dtcSnapshotReadouts				
Description	This API creates a readout of a DTC snapshot from one vehicle. If the result is available immediately, the result is returned. If the result is not available, a location to the DTC snapshot readout is returned. This location shall be polled until the result is available.			
Parameters	vehicleId	string	required	The vehicle identifier of the vehicle to read from
	ecuId	string	required	The id of the ECU to read from
	dtcId	string	required	The id of the DTC associated with the snapshot
Request headers	Host	required	According to HTTP/1.1 RFC 2616	
	Authorization	required	Bearer {token}	
	Accept	required	application/json; exve-resourceversion= dtcsnapshotreadout.v1.0; charset=utf-8	
Response headers	Location	Absolute URI of the /dtcSnapshotReadouts endpoint		

Table A.12 *(continued)*

<table>
<tr><td>Response (success)</td><td>201</td><td colspan="2">Example (result available immediately):
{
"dtcSnapshotReadout": {
"id": "abcde-12345-ghjke-67474",
"asyncStatus": "Complete",
"messageTimestamp": "2016-02-24T09:23:46Z",
"vehicleId": "12345678909876543",
"receivedTimestamp": "2016-02-24T09:23:46Z",
"ecuId": "ABC",
"dtcId": "123456",
"dtcSnapshotParameters": [
{
"name": "1234",
"value": "83"
},
{
"name": "5678",
"value": "2"
},
{
"name": "9012",
"value": "27"
}
]
}
}</td></tr>
<tr><td></td><td>202</td><td colspan="2">Result is not available immediately. Location of dtcSnapshotReadout will be returned, e.g. vehicles/{vehicleId}/ecuId/{ecuId}/dtcId/{dtcId}/dtcSnapshotReadouts/{id}, see Location header.</td></tr>
<tr><td>Error codes</td><td>400</td><td colspan="2">Bad Request</td></tr>
<tr><td></td><td>401</td><td colspan="2">Unauthorized</td></tr>
<tr><td></td><td>403</td><td colspan="2">Forbidden</td></tr>
<tr><td></td><td>404</td><td>Not Found</td><td>ECU ID not valid
DTC ID not valid</td></tr>
<tr><td></td><td>406</td><td colspan="2">Not Acceptable</td></tr>
<tr><td></td><td>500</td><td colspan="2">Internal Server Error</td></tr>
<tr><td></td><td>501</td><td colspan="2">Not Implemented</td></tr>
<tr><td></td><td>503</td><td colspan="2">Service Unavailable</td></tr>
<tr><td></td><td>505</td><td colspan="2">Version Not Supported</td></tr>
<tr><td>Access</td><td colspan="2">DTC snapshot readout</td><td>Full access to this API</td></tr>
<tr><td>JSON schema</td><td colspan="3">See Clause A.5.</td></tr>
</table>

Table A.13 — GET dtcSnapshotReadout

<table>
<tr><td colspan="6">GET vehicles/{vehicleId}/ecuId/{ecuId}/dtcId/{dtcId}/dtcSnapshotReadouts/{id}</td></tr>
<tr><td>Description</td><td colspan="5">This API returns the DTC snapshot from one vehicle. The id of the readout is returned when posting the request. If the DTC snapshot readout is not completed, the readout status is returned. If the DTC snapshot readout is completed, the result is returned.</td></tr>
<tr><td rowspan="4">Parameters</td><td>vehicleId</td><td>string</td><td>required</td><td colspan="2">The vehicle identifier of the vehicle to read from</td></tr>
<tr><td>ecuId</td><td>string</td><td>required</td><td colspan="2">The id of the ECU to read from</td></tr>
<tr><td>dtcId</td><td>string</td><td>required</td><td colspan="2">The id of the DTC associated with the snapshot</td></tr>
<tr><td>id</td><td>string</td><td>required</td><td colspan="2">id of the DTC snapshot readout</td></tr>
<tr><td rowspan="3">Request headers</td><td>Host</td><td>required</td><td colspan="3">According to HTTP/1.1 RFC 2616</td></tr>
<tr><td>Authorization</td><td>required</td><td colspan="3">Bearer {token}</td></tr>
<tr><td>Accept</td><td>required</td><td colspan="3">application/json; exve-resourceversion= dtcsnapshotreadout.v1.0;charset=utf-8</td></tr>
<tr><td>Response headers</td><td>Content-Type</td><td colspan="4">application/json; exve-resourceversion= dtcsnapshotreadout.v1.0; charset=utf-8</td></tr>
<tr><td>Response (success)</td><td>200</td><td colspan="4">Example (result available):
<pre>{
"dtcSnapshotReadout": {
"id": "abcde-12345-ghjke-67474",
"asyncStatus": "Complete",
"messageTimestamp": "2016-02-24T09:23:46Z",
"vehicleId": "12345678909876543",
"receivedTimestamp": "2016-02-24T09:23:46Z",
"ecuId": "ABC",
"dtcId": "123456",
"dtcSnapshotParameters": [
{
"name": "1234",
"value": "83"
},
{
"name": "5678",
"value": "2"
},
{
"name": "9012",
"value": "27"
}
]
}
}</pre></td></tr>
</table>

Table A.13 *(continued)*

		Example (result not available):
		{
		"dtcSnapshotReadout": {
		"id": "abcde-12345-ghjke-67474",
		"asyncStatus": "InProgress",
		"asyncWait": 10000,
		"asyncEstimatedComplete": "2016-02-24T09:24:00Z",
		"messageTimestamp": "2016-02-24T09:23:46Z",
		"vehicleId": "12345678909876543"
		}
		}

Error codes	400	Bad Request
	401	Unauthorized
	403	Forbidden
	404	Not Found
	406	Not Acceptable
	500	Internal Server Error
	501	Not Implemented
	503	Service Unavailable
	505	Version Not Supported
Access	DTC snapshot readout	Full access to this API
JSON schema	See Clause A.5.	

A.4.6 Use case 06 – parameterReadouts

The parameterReadout API follows the asynchronous interaction pattern, as the processing time of the request can vary dependent on the offering party implementation (see Figure A.7). In some cases it is possible to return the result immediately, whereas sometimes the accessing party needs to poll the API until the result is ready.

The accessing party starts by posting a parameter readout request (see Table A.14). If the result is available immediately, the result is returned directly. If the result is not available immediately, a status is returned instead of the readout. The accessing party is supposed to poll the request status until the processing is completed (see Table A.15). When the processing is completed, the accessing party will receive the result.

The readout status and the completed readout will be available for a limited time after being created. This time is specified by the offering party.

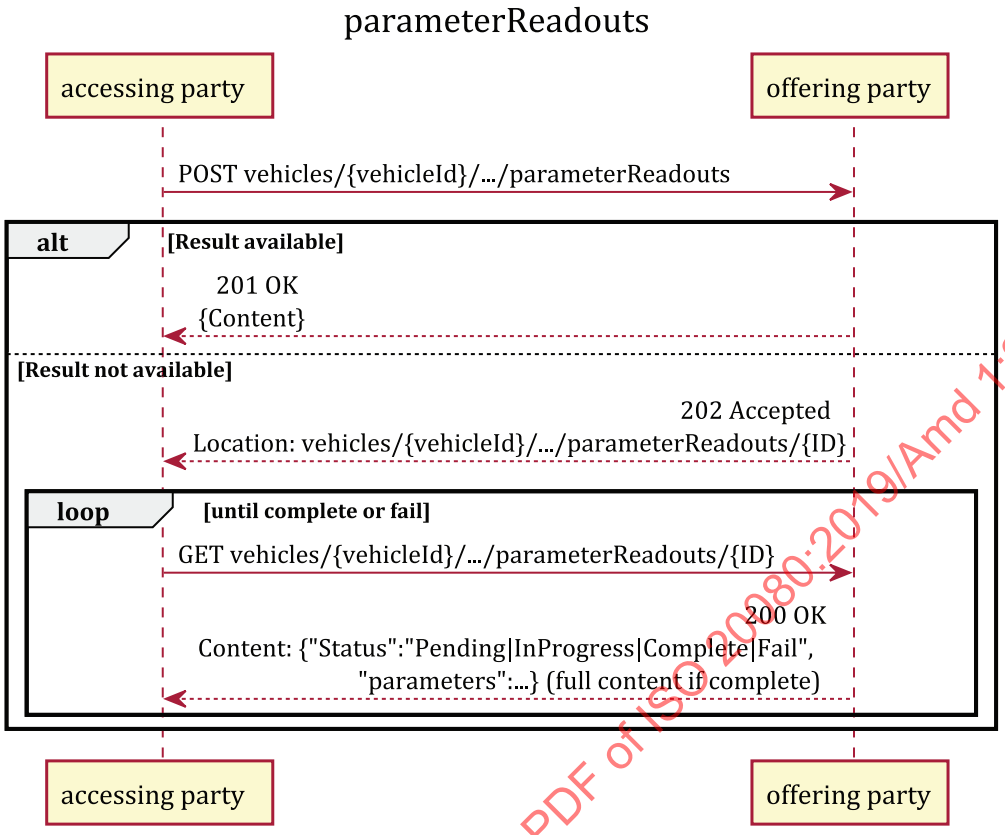

Figure A.7 — parameterReadouts sequence diagram

Table A.14 — POST parameterReadout

POST vehicles/{vehicleId}/ecuId/{ecuId}/parameterReadouts?paramId={paramId}*				
Description	This API creates a readout of parameters from one ECU in a vehicle. If the result is available immediately, the result is returned. If the result is not available, a location to the parameter readout is returned. This location shall be polled until the result is available.			
Parameters	vehicleId	string	required	The vehicle identifier of the vehicle to read from
	ecuId	string	required	The id of the ECU to read from
	paramId	string	required	The parameter id(s) to read. *) The paramId can occur many times.
Request headers	Host	required	According to HTTP/1.1 RFC 2616	
	Authorization	required	Bearer {token}	
	Accept	required	application/json; exve-resourceversion= parameterreadout. v1.0; charset=utf-8	
Response headers	Location	Absolute URI of the /parameterReadouts endpoint		

Table A.14 *(continued)*

<table>
<tr><td>Response (success)</td><td>201</td><td colspan="2">Example (result available immediately):
<pre>{
"parameterReadout": {
"id": "abcde-12345-ghjke-67474",
"asyncStatus": "Complete",
"messageTimestamp": "2016-02-24T09:23:46Z",
"vehicleId": "12345678909876543",
"receivedTimestamp": "2016-02-24T09:23:46Z",
"ecuId": "ABC",
"parameters": [
{
"id": "1234",
"value": "648"
},
{
"id": "5678",
"value": "1000"
}
]
}
}</pre></td></tr>
<tr><td></td><td>202</td><td colspan="2">Result not available immediately. Location of parameterReadout will be returned, e.g. vehicles/{vehicleId}/ecuId/{ecuId}/parameterReadouts/{id}, see Location header.</td></tr>
<tr><td>Error codes</td><td>400</td><td>Bad Request</td><td>Number of parameters exceeded</td></tr>
<tr><td></td><td>401</td><td colspan="2">Unauthorized</td></tr>
<tr><td></td><td>403</td><td colspan="2">Forbidden</td></tr>
<tr><td></td><td>404</td><td>Not Found</td><td>ECU Id not valid
Parameter Id not valid</td></tr>
<tr><td></td><td>406</td><td colspan="2">Not Acceptable</td></tr>
<tr><td></td><td>500</td><td colspan="2">Internal Server Error</td></tr>
<tr><td></td><td>501</td><td colspan="2">Not Implemented</td></tr>
<tr><td></td><td>503</td><td colspan="2">Service Unavailable</td></tr>
<tr><td></td><td>505</td><td colspan="2">Version Not Supported</td></tr>
<tr><td>Access</td><td colspan="2">Parameter readout</td><td>Full access to this API</td></tr>
<tr><td>JSON schema</td><td colspan="3">See Clause A.5.</td></tr>
</table>

Table A.15 — GET parameterReadout

<table>
<tr><td colspan="6">GET vehicles/{vehicleId}/ecuId/{ecuId}/parameterReadouts/{id}</td></tr>
<tr><td>Description</td><td colspan="5">This API returns the parameter readout from one ECU in a vehicle. The id of the readout is returned when posting the request. If the parameter readout is not completed, the readout status is returned. If the parameter readout is completed, the result is returned.</td></tr>
<tr><td rowspan="3">Parameters</td><td>vehicleId</td><td>string</td><td>required</td><td colspan="2">The vehicle identifier of the vehicle to read from</td></tr>
<tr><td>ecuId</td><td>string</td><td>required</td><td colspan="2">The id of the ECU to read from</td></tr>
<tr><td>id</td><td>string</td><td>required</td><td colspan="2">Id of the parameter readout</td></tr>
<tr><td rowspan="3">Request headers</td><td>Host</td><td>required</td><td colspan="3">According to HTTP/1.1 RFC 2616</td></tr>
<tr><td>Authorization</td><td>required</td><td colspan="3">Bearer {token}</td></tr>
<tr><td>Accept</td><td>required</td><td colspan="3">application/json; exve-resourceversion= parameterreadout.v1.0; charset=utf-8</td></tr>
<tr><td>Response headers</td><td colspan="2">Content-Type</td><td colspan="3">application/json; exve-resourceversion= parameterreadout.v1.0; charset=utf-8</td></tr>
<tr><td>Response (success)</td><td>200</td><td colspan="4">Example (result available):
<pre>{
"parameterReadout": {
"id": "abcde-12345-ghjke-67474",
"asyncStatus": "Complete",
"messageTimestamp": "2016-02-24T09:23:46Z",
"vehicleId": "12345678909876543",
"receivedTimestamp": "2016-02-24T09:23:46Z",
"ecuId": "ABC",
"parameters":[
{
"id": "1234",
"value": "648"
},
{
"id": "5678",
"value": "1000"
}
]
}
}</pre></td></tr>
</table>

Table A.15 *(continued)*

		Example (result not available):
		{
		"parameterReadout": {
		"id": "abcde-12345-ghjke-67474",
		"asyncStatus": "InProgress",
		"asyncWait": 10000,
		"asyncEstimatedComplete": "2016-02-24T09:24:00Z",
		"messageTimestamp": "2016-02-24T09:23:46Z",
		"vehicleId": "12345678909876543",
		"ecuId": "ABC"
		}
		}

Error codes	400	Bad Request
	401	Unauthorized
	403	Forbidden
	404	Not Found
	406	Not Acceptable
	500	Internal Server Error
	501	Not Implemented
	503	Service Unavailable
	505	Version Not Supported
Access	Parameter readout	Full access to this API
JSON schema	See Clause A.5.	

A.4.7 Use case 07 – malfunctionIndicatorReadouts

The malfunctionIndicatorReadout API follows the asynchronous interaction pattern, as the processing time of the request can vary dependent on the offering party implementation (see Figure A.8). In some cases, it is possible to return the result immediately, whereas sometimes the accessing party needs to poll the API until the result is ready.

The accessing party starts by posting a malfunction indicator readout request (see Table A.16). If the result is available immediately, the result is returned directly. If the result is not available immediately, a status is returned instead of the readout. The accessing party is supposed to poll the request status until the processing is completed (see Table A.17). When the processing is completed, the accessing party will receive the result.

The readout status and the completed readout will be available for a limited time after being created. This time is specified by the offering party.

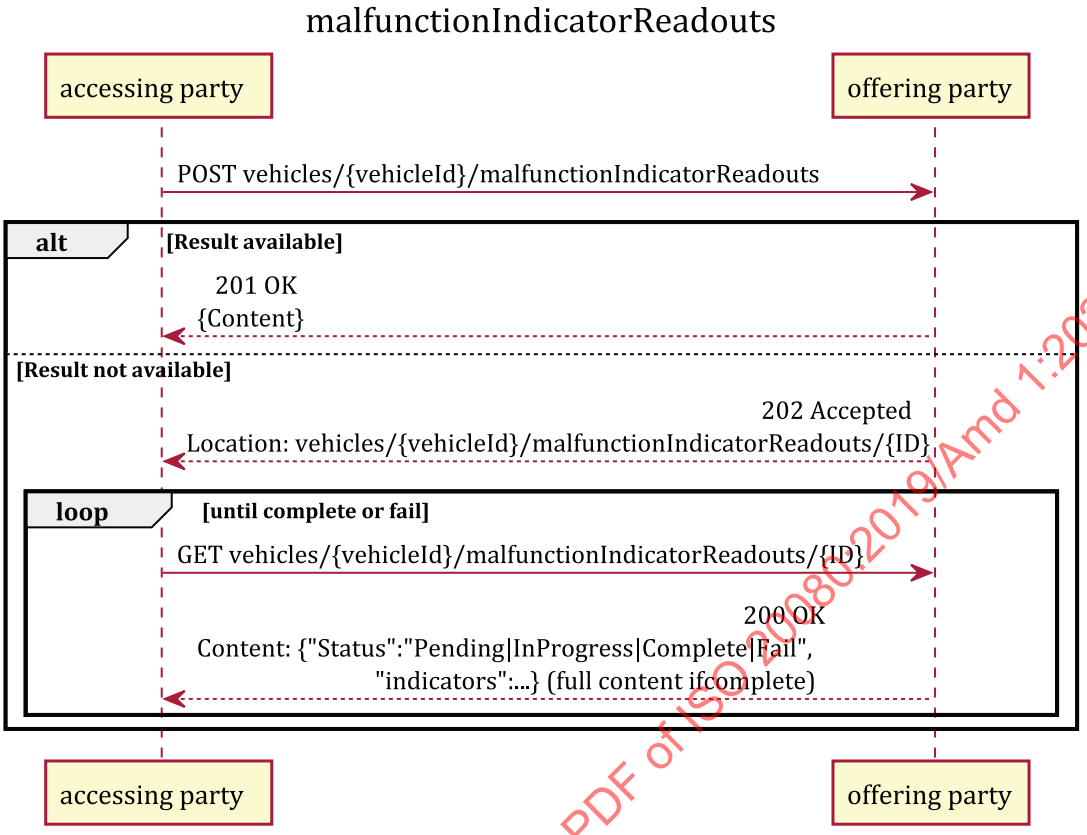

Figure A.8 — malfunctionIndicatorReadouts sequence diagram

Table A.16 — POST malfunctionIndicatorReadout

POST vehicles/{vehicleId}/malfunctionIndicatorReadouts				
Description	This API creates a readout of malfunction indicators from one vehicle. If the result is available immediately, the result is returned. If the result is not available, a location to the malfunction indicator readout is returned. This location shall be polled until the result is available.			
Parameters	vehicleId	string	required	The vehicle identifier of the vehicle to read from
Request headers	Host	required	According to HTTP/1.1 RFC 2616	
	Authorization	required	Bearer {token}	
	Accept	required	application/json; exve-resourceversion= malfunctionindicatorreadout.v1.0; charset=utf-8	
Response headers	Location	Absolute URI of the /malfunctionIndicatorReadouts endpoint		

Table A.16 *(continued)*

Response (success)	201	Example (result available immediately):
		{
		"malfunctionIndicatorReadout": {
		"id": "abcde-12345-ghjke-67474",
		"asyncStatus": "Complete",
		"messageTimestamp": "2016-02-24T09:23:46Z",
		"vehicleId": "12345678909876543",
		"receivedTimestamp": "2016-02-24T09:23:46Z",
		"indicators": [
		{
		"id": "A",
		"status": "active"
		},
		{
		"id": "B",
		"status": "yellow"
		}
		]
		}
		}

	202	Result is not available immediately. Location of malfunctionIndicatorReadout will be returned, e.g. vehicles/{vehicleId}/malfunctionIndicatorReadouts/{id}, see Location header.
Error codes	400	Bad Request
	401	Unauthorized
	403	Forbidden
	404	Not Found
	406	Not Acceptable
	500	Internal Server Error
	501	Not Implemented
	503	Service Unavailable
	505	Version Not Supported
Access	Malfunction indicator readout	Full access to this API
JSON schema	See Clause A.5.	

Table A.17 — GET malfunctionIndicatorReadout

GET vehicles/{vehicleId}/malfunctionIndicatorReadouts/{id}				
Description	This API returns the malfunction indicator readout from one vehicle. The id of the readout is returned when posting the request. If the malfunction indicator readout is not completed, the readout status is returned. If the parameter readout is completed, the result is returned.			
Parameters	vehicleId	string	required	The vehicle identifier of the vehicle to read from
	id	string	required	Id of the malfunction indicator readout

Table A.17 *(continued)*

<table>
<tr><td rowspan="3">Request headers</td><td>Host</td><td>required</td><td>According to HTTP/1.1 RFC 2616</td></tr>
<tr><td>Authorization</td><td>required</td><td>Bearer {token}</td></tr>
<tr><td>Accept</td><td>required</td><td>application/json; exve-resourceversion= malfunctionindicatorreadout.v1.0; charset=utf-8</td></tr>
<tr><td>Response headers</td><td>Content-Type</td><td colspan="2">application/json; exve-resourceversion= malfunctionindicatorreadout.v1.0; charset=utf-8</td></tr>
<tr><td>Response (success)</td><td>200</td><td colspan="2">Example (result available):
<pre>{
"malfunctionIndicatorReadout": {
   "id": "abcde-12345-ghjke-67474",
   "asyncStatus": "Complete",
   "messageTimestamp": "2016-02-24T09:23:46Z",
   "vehicleId": "12345678909876543",
   "receivedTimestamp": "2016-02-24T09:23:46Z",
   "indicators": [
         {
         "id": "A",
         "status": "active"
         },
         {
         "id": "B",
         "status": "yellow"
         }
      ]
   }
}</pre>
Example (result not available):
<pre>{
"malfunctionIndicatorReadout": {
   "id": "abcde-12345-ghjke-67474",
   "asyncStatus": "InProgress",
   "asyncWait": 10000,
   "asyncEstimatedComplete": "2016-02-24T09:24:00Z",
   "messageTimestamp": "2016-02-24T09:23:46Z",
   "vehicleId": "12345678909876543"
   }
}</pre></td></tr>
</table>

Table A.17 *(continued)*

Error codes	400	Bad Request
	401	Unauthorized
	403	Forbidden
	404	Not Found
	406	Not Acceptable
	500	Internal Server Error
	501	Not Implemented
	503	Service Unavailable
	505	Version Not Supported
Access	Malfunction indicator readout	Full access to this API
JSON schema	See Clause A.5.	

A.4.8 Use case 08 – clearDtcJobs

The clearDtcJobs API follows the asynchronous interaction pattern, as the processing time of the request can vary dependent on the offering party implementation (see Figure A.9). In some cases, it is possible to return the result immediately, whereas sometimes the accessing party needs to poll the API until the result is ready.

The accessing party starts by posting a clear DTC job request (see Table A.18). If the result is available immediately, the result is returned directly. If the result is not available immediately, a status is returned instead of the readout. The accessing party is supposed to poll the request status until the processing is completed (see Table A.19). When the processing is completed, the accessing party will receive the result.

The readout status and the completed readout will be available for a limited time after being created. This time is specified by the offering party.

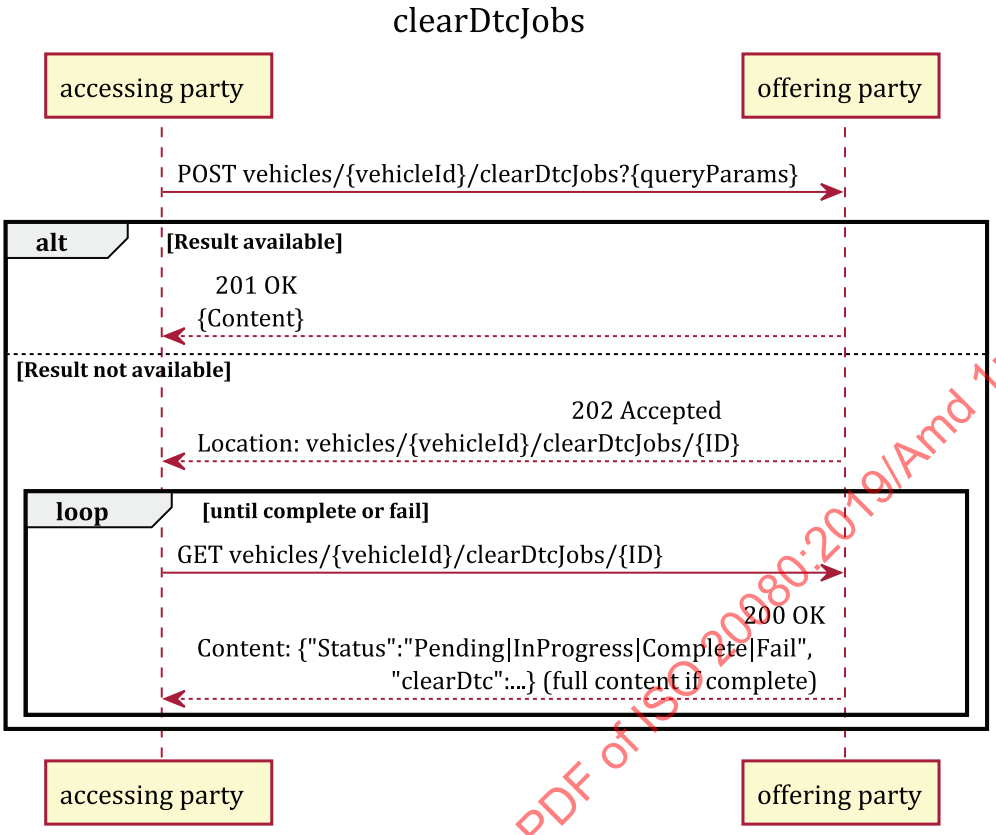

Figure A.9 — clearDtcJobs sequence diagram

Table A.18 — POST clearDtcJobs

POST vehicles/{vehicleId}/clearDtcJobs?ecuId={ecuId}				
Description	This API creates a clear DTC job for one vehicle. If the result is available immediately, the result is returned. If the result is not available, a location to the clear DTC job is returned. This location shall be polled until the result is available.			
Parameters	vehicleId	string	required	The vehicle identifier of the vehicle to read from
	ecuId	string	optional	Clears DTCs on this ECU id only Default: Clears DTCs on all ECUs
Request headers	Host	required	According to HTTP/1.1 RFC 2616	
	Authorization	required	Bearer {token}	
	Accept	required	application/json; exve-resourceversion=cleardtcjob.v1.0; charset=utf-8	
Response headers	Location	Absolute URI of the /clearDtcJobs endpoint		

Table A.18 *(continued)*

Response (success)	201	Example (result available immediately):
		{
		"clearDtcJob": {
		"id": "abcde-12345-ghjke-67474",
		"asyncStatus": "Complete",
		"messageTimestamp": "2016-02-24T09:23:46Z",
		"vehicleId": "12345678909876543",
		"receivedTimestamp": "2016-02-24T09:23:46Z",
		"clearDtc": {
		"confirmation": "OK"
		}
		}
		}

	202	Result is not available immediately. Location of clearDtcJob will be returned, e.g. vehicles/{vehicleId}/clearDtcJobs/{id}, see Location header.
Error codes	400	Bad Request
	401	Unauthorized
	403	Forbidden
	404	Not Found — ECU Id not valid
	406	Not Acceptable
	500	Internal Server Error
	501	Not Implemented
	503	Service Unavailable
	505	Version Not Supported
Access	Clear DTC Job	Full access to this API
JSON schema	See Clause A.5.	

Table A.19 — GET clearDtcJobs

GET vehicles/{vehicleId}/clearDtcJobs/{id}				
Description	This API returns the clear DTC job result for one vehicle. The id of the job is returned when posting the request. If the job is not completed, the job status is returned. If the job is completed, the result is returned.			
Parameters	vehicleId	string	required	The vehicle identifier of the vehicle to read from
	id	string	required	id of the clear DTC job
Request headers	Host	required	According to HTTP/1.1 RFC 2616	
	Authorization	required	Bearer {token}	
	Accept	required	application/json; exve-resourceversion=cleardtcjob.v1.0; charset=utf-8	
Response headers	Content-Type	application/json; exve-resourceversion=cleardtcjob.v1.0; charset=utf-8		

Table A.19 *(continued)*

Response (success)	200	Example (result available):
		{
		"clearDtcJob": {
		"id": "abcde-12345-ghjke-67474",
		"asyncStatus": "Complete",
		"messageTimestamp": "2016-02-24T09:23:46Z",
		"vehicleId": "12345678909876543",
		"receivedTimestamp": "2016-02-24T09:23:46Z",
		"clearDtc": {
		"confirmation": "OK"
		}
		}
		}
		Example (result not available):
		{
		"clearDtcJob": {
		"id": "abcde-12345-ghjke-67474",
		"asyncStatus": "InProgress",
		"asyncWait": 10000,
		"asyncEstimatedComplete": "2016-02-24T09:24:00Z",
		"messageTimestamp": "2016-02-24T09:23:46Z",
		"vehicleId": "12345678909876543"
		}
		}

Error codes	400	Bad Request
	401	Unauthorized
	403	Forbidden
	404	Not Found
	406	Not Acceptable
	500	Internal Server Error
	501	Not Implemented
	503	Service Unavailable
	505	Version Not Supported
Access	Clear DTC Job	Full access to this API
JSON schema	See Clause A.5.	

A.5 JSON schemas

A.5.1 General

A.5.1.1 Introduction

This subclause contains descriptions and graphical representations of the JSON schemas[3] referenced from the REST APIs. The actual schemas can be downloaded from https://standards.iso.org/iso/20080/

ed-1/en/amd/1. All JSON schemas are using the draft 4 version, as it has the best tooling support at the time of writing.

Legend for properties tables:

M - Mandatory

O -Optional

A.5.1.2 Extensible

All JSON schemas are extensible ("additionalProperties" : true) to allow for future versions and/or additional information from the offering party.

A.5.1.3 Asynchronous interaction pattern

All JSON schemas supporting asynchronous REST APIs have some parameters in common (see Figure A.10 and Table A.20). Those are described here and not repeated in each section.

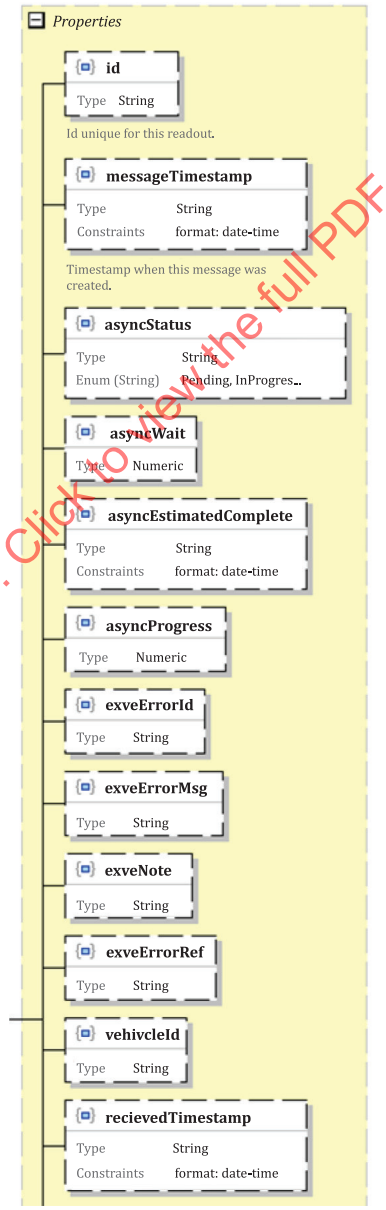

Figure A.10 — Common parameters supporting asynchronous REST APIs

Table A.20 — Common parameters supporting asynchronous REST APIs

Property	M/O	Description
id	M	The id of the readout, allocated when the readout is created
messageTimestamp	M	Timestamp when the JSON document was created, e.g. 2017-04-12T23:20:50.52Z
asyncStatus	M	Reflects the status of the readout. Possible values are: **Pending**: the requested operation has not yet started processing, **InProgress**: the requested operation is processing but is not completed, **Complete**: the requested operation is complete and has succeeded, **Fail**: the requested operation is completed and has failed or timed out.
asyncWait	O	Recommended waiting time in milliseconds until next request for status
asyncEstimatedComplete	O	Estimated completion time of the readout
asyncProgress	O	Progress of the readout in per cent
exveErrorId	O	The id of the error in case something failed
exveErrorMsg	O	The error message in case something failed
exveNote	O	A message regarding the result of the request, it can be filled both when the request was completed and failed.
exveErrorRef	O	A unique reference to the error, it can be used in communication with the offering party to help in resolving problems.
vehicleId	M	The vehicle identifier of the ExVe, e.g VIN
receivedTimestamp	O	Timestamp when the data in the message was received by the ExVe backend server from the vehicle.

A.5.2 Use case 01 – resourceReadout

Figure A.11 and Table A.21 describe the structure and content of the JSON schema.

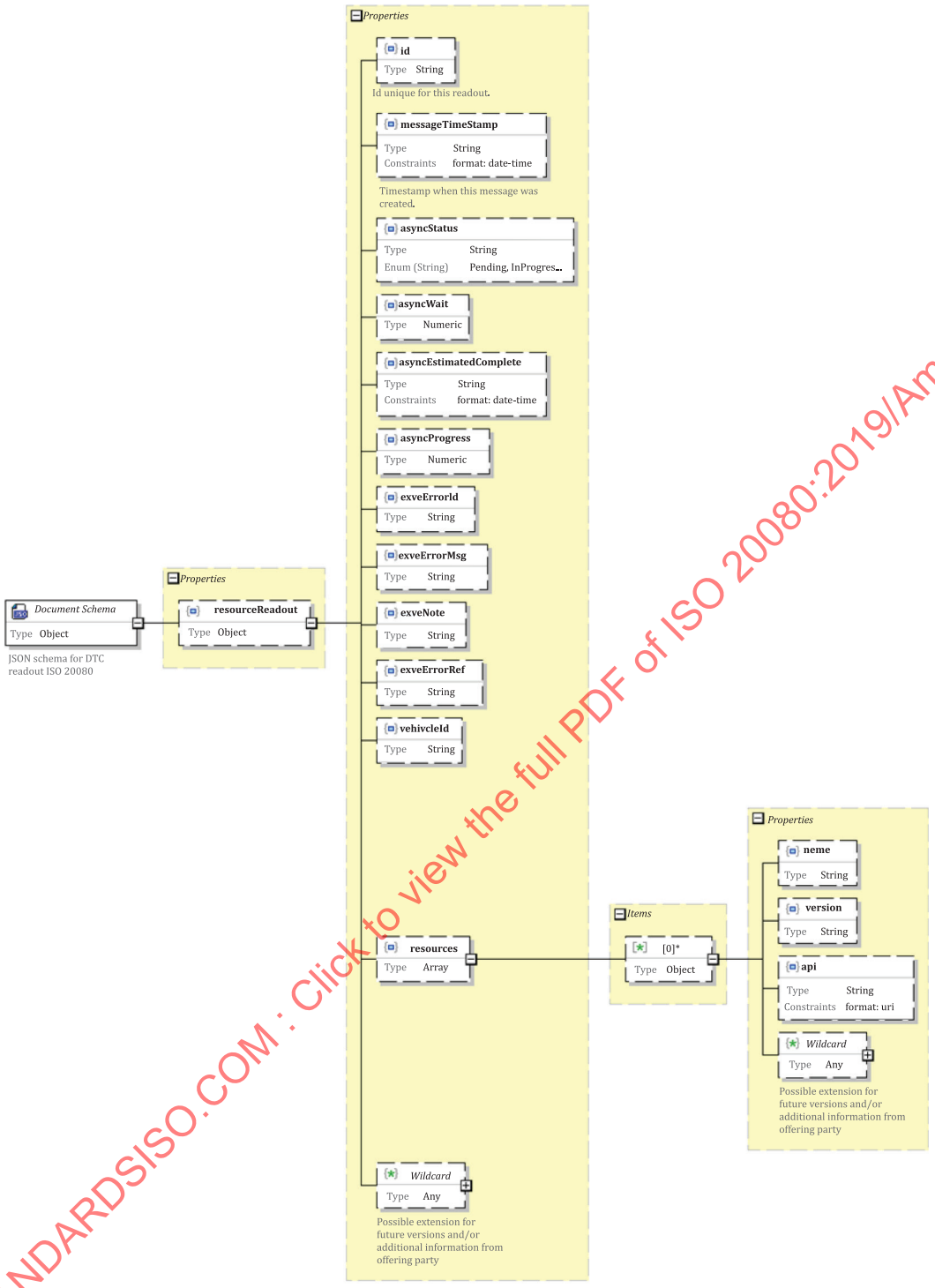

Figure A.11 — UC01-resourceReadout_schema

Table A.21 — UC01-resourceReadout_schema

Property	M/O	Description
resources/name	M	Name of resource
resources/version	M	Version of resource in API
resources/api	M	URI to API exposing resource

A.5.3 Use case 02 – ecuReadout

Figure A.12 and Table A.22 describe the structure and content of the JSON schema.

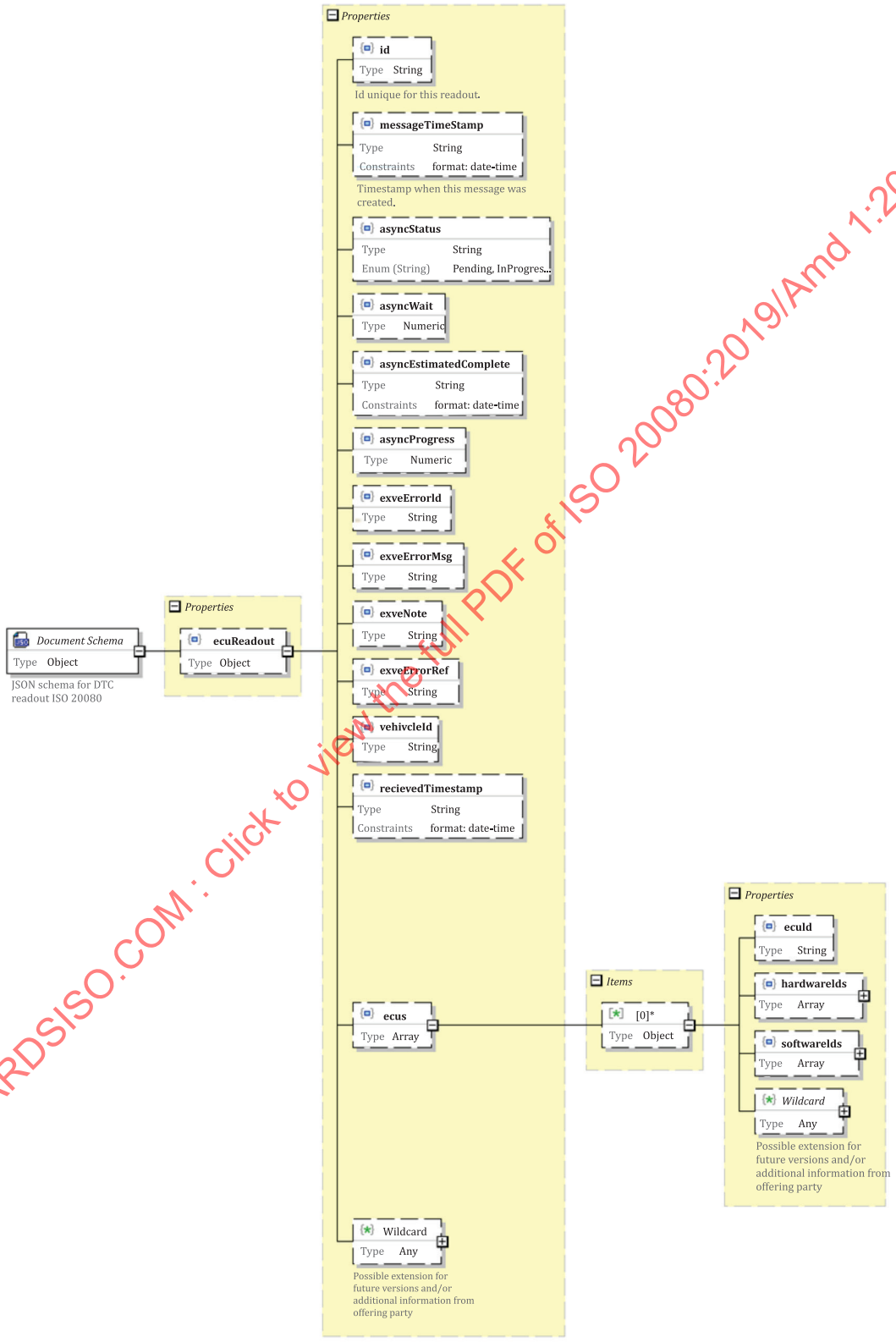

Figure A.12 — UC02-ecuReadout_schema

Table A.22 — UC02-ecuReadout_schema

Property	M/O	Description
ecuId	M	Identifier of the ECU
hardwareIds	M	Identifier(s) of the hardware unit(s) in this ECU
softwareIds	M	Identifier(s) of the software unit(s) in this ECU

A.5.4 Use case 03 – dtcReadout

Figure A.13 and Table A.23 describe the structure and content of the JSON schema.