

	Title: agrirouter Connectivity-Platform Integration Guide Part 2 Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

agrirouter Connectivity- Platform Integration Guide (Part 2)

	Title: agrirouter Connectivity-Platform Integration Guide Part 2 Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

author(s): DKE-Data GmbH & Co. KG
last modification: 9/4/2018 5:08:00 PM
filename: DKE_agrirouter_Integration_Guide_Part_2_v1.3.DOCX
version: 1.3

 DKE DATA	Title: agrirouter Connectivity-Platform Integration Guide Part 2 Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

Content

1. Purpose	12
1.1. Purpose of this document.....	12
1.2. Purpose of further documents	12
2. Epilog.....	13
3. How to use this document – an advice on development and more	15
4. General conventions	16
4.1. Numbering convention	16
4.2. String Identifiers convention.....	16
4.3. TimeStamp conventions.....	16
4.4. Time Synchronisation.....	16
5. Environments.....	17
5.1. Quality assurance.....	17
5.2. Productive	17
5.3. Areas.....	17
6. Development tools	18
6.1. REST using Postman	18
6.1.1. Check onboarding	18
6.1.2. Adding a Certificate.....	18
6.2. MQTT using Paho	20
6.2.1. Format Certificate for Paho	20
6.2.2. Setup Paho Certificate	20
6.2.3. Setup the connection criteria of Paho:.....	21
6.2.4. Basic functions	21
6.3. agrirouter protobuf toolset	22
7. Authentication Process.....	23
7.1. Authentication of farming software	23
7.1.1. Preparation.....	23
7.1.2. Authentication Process Overview.....	25
7.1.3. Generating an authentication URL	26
7.1.4. Perform authentication	26
7.1.5. Analyse result.....	27
7.2. Authentication for CUs and non-cloud-software	28
8. Onboarding App Instances.....	29
8.1. General Overview.....	29
8.2. Workflow for Farming Software and Telemetry Systems.....	30
8.3. Workflow for CUs	31
8.4. Workflow for Virtual CUs	31
8.5. Creating a Registration Code	32
8.5.1. Using the agrirouter UI	32
8.5.2. As result of the authentication	33
8.5.3. Telemetry Platforms	33
8.6. Collecting and setting up relevant information.....	34
8.6.1. ApplicationID and X-Agrirouter-ApplicationID.....	34
8.6.2. CertificationVersionID.....	34
8.6.3. Setting up private and public key	35

 DKE DATA	Title: agrirouter Connectivity-Platform Integration Guide Part 2	
	Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

8.7.	onboarding Endpoint URLs	36
8.8.	Signing requests.....	36
8.9.	Sending a Verification Request	37
8.9.1.	General	37
8.9.2.	Request Information.....	37
8.9.3.	Verification Result	38
8.10.	Sending Onboarding Request	39
8.10.1.	Request information for signed onboarding.....	39
8.10.2.	Request information for CU onboarding.....	41
8.10.3.	Response	41
8.11.	Reonboarding.....	44
8.11.1.	What is re-onboarding?	44
8.11.2.	Why should an endpoint be re-onboarded?	44
8.11.3.	How does re-onboarding work?	44
9.	Revoking Accounts	45
9.1.	Process of Revoking	45
9.1.1.	Creating Revoke Command	45
9.1.2.	Header	45
9.1.3.	Body	46
9.1.4.	Response	46
10.	Architecture of an endpoint.....	47
10.1.	General Overview.....	47
10.2.	Inbox	48
10.3.	Outbox.....	49
10.4.	Feed.....	50
10.5.	Subscription List.....	50
11.	Basics of communication	51
11.1.	General communication convention	51
11.2.	Prequisits for communication	51
11.3.	Different layers, "The onion principle".....	51
11.4.	The protocol layer.....	52
11.4.1.	REST.....	52
11.4.2.	MQTT	54
11.4.3.	Comparison of protocols	55
11.5.	Abstraction of communication workflows.....	55
12.	Communication of App Instance and Endpoint.....	56
12.1.	Endpoint Adresses	56
12.2.	Assigning a result to a request	57
12.3.	Flow for sending messages.....	57
12.4.	Flow of commands	58
12.5.	Flow of requesting messages from the feed.....	58
12.6.	Terms	59
13.	Building Messages and Commands	60
13.1.	General	60
13.2.	Chunking big messages	60
13.3.	Encode content to its Technical Message Type	61
13.4.	Encode content in Protobuf	61

 DKE DATA	Title: agrirouter Connectivity-Platform Integration Guide Part 2 Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

13.4.1. Getting the agrirouter protobuf files	61
13.4.2. Reading Protobuf definitions	63
13.4.3. Including Protobuf in your project.....	63
13.5. Chunking Messages.....	64
13.5.1. Chunking parameters.....	64
13.5.2. Preparing and creating chunked messages	64
13.6. Building the Envelope.....	65
13.7. Building the payload	65
13.8. Connecting envelope and PayLoad.....	65
13.9. Message container.....	66
13.9.1. Encode in Base64	66
13.9.2. Go on in Protobuf	66
13.10. Adding the Timestamp	66
13.11. Add Message to List of Messages.....	66
13.11.1. JSON.....	66
13.11.2. Protobuf.....	66
13.12. Build message.....	67
13.12.1. JSON setup.....	67
13.12.2. Protobuf setup	67
14. Sending a Request	69
14.1. General Prequesits.....	69
14.1.1. Routings available.....	69
14.1.2. Publishing only with available subscriptions	69
14.2. Using MQTT	69
14.3. Using REST	69
14.3.1. Address	69
14.3.2. Header	69
14.3.3. Body	70
14.4. Header	70
14.5. Body	70
15. Receiving Results.....	71
15.1. General	71
15.2. MQTT	71
15.3. REST.....	71
15.4. Format of received messages	71
15.4.1. JSON Result	71
15.4.2. Protobuf Result	72
15.4.3. ParameterList.....	73
15.5. Decoding Base64	73
15.6. <i>Delimiting Envelope and Payload.</i>	73
15.7. Envelope	73
15.7.1. <i>response_code</i>	73
15.7.2. type	74
15.7.3. application_message_id.....	74
15.7.4. message_id.....	75
15.7.5. timestamp.....	75
15.8. Payload	75
15.8.1. Messages.....	75
16. Overview of agrirouter commands	76

 DKE DATA	Title: agrirouter Connectivity-Platform Integration Guide Part 2 Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

16.1.	Describing the endpoint.....	77
16.1.1.	Capabilities Command	77
16.1.2.	Subscription Command	78
16.1.3.	Sending a teamset using device Descriptions.....	81
16.2.	Reading the feed.....	82
16.2.1.	Paging.....	82
16.2.2.	Message Query and Filtering	82
16.2.3.	Call for Message Header List	82
16.2.4.	Call for specific messages.....	85
16.2.5.	Call for message deletion.....	88
16.3.	Controlling the outbox	89
16.3.1.	Call for message list confirmation.....	89
16.4.	Read the eco system.....	90
16.4.1.	Call for endpoints, that support a technical message type	90
16.4.2.	Call for filtered list of endpoints, that support a specific message type	92
16.5.	Commands for telemetry platforms	93
16.5.1.	Onboarding a Virtual CU	93
16.5.2.	Removing a Virtual Cu	94
17. Technical Message types.....		95
17.1.	General	95
17.2.	iso:11783:-10:device_description:protobuf - Teamset/EFDI Device Description.....	96
17.2.1.	Definition	96
17.2.2.	Command.....	96
17.2.3.	Result.....	96
17.3.	iso:11783:-10:time_log:protobuf - EFDI TimeLog.....	97
17.3.1.	Command.....	97
17.4.	iso:11783:-10:taskdata:zip - TaskData.....	98
17.4.1.	Definition	98
17.4.2.	Command.....	98
17.4.3.	Format.....	98
17.5.	shp:shape:zip - Shape.....	99
17.5.1.	Definition	99
17.5.2.	Command.....	99
17.5.3.	Format.....	99
17.6.	img: bmp - Bitmap Image	100
17.6.1.	Definition	100
17.6.2.	Command.....	100
17.6.3.	Format.....	100
17.7.	img: jpg – JPEG Image	101
17.7.1.	Definition	101
17.7.2.	Command.....	101
17.7.3.	Format.....	101
17.8.	img: png – PNG Image	102
17.8.1.	Definition	102
17.8.2.	Command.....	102
17.8.3.	Format.....	102
17.9.	vid: avi – AVI Video Files.....	103
17.9.1.	Definition	103
17.9.2.	Command.....	103
17.9.3.	Format.....	103

 DKE DATA	Title: agrirouter Connectivity-Platform Integration Guide Part 2 Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

17.10. vid:wmv – WMV Video Files	104
17.10.1. Definition	104
17.10.2. Command.....	104
17.10.3. Format.....	104
17.11. vid:mp4 – MPEG4 Video Files	105
17.11.1. Definition	105
17.11.2. Command.....	105
17.11.3. Format.....	105
18. Session workflow	106
18.1. General Overview.....	106
18.2. Request sending Frequency	107
19. IDs, IDs, IDs – Which is which?	108
19.1. Overview	108
19.2. Ids in different Environments.....	108
19.3. Practical differentiation between SensorAlternateId and DeviceAlternateId	108
19.3.1. An App sends a TaskData.....	108
19.3.2. A CU sends a TaskData.....	108
19.3.3. A telemetry Platform sends a TaskData from a virtual CU	108
19.4. accountId.....	109
19.4.1. Description	109
19.4.2. Where to find.....	109
19.5. applicationId	110
19.5.1. Description	110
19.5.2. Where to find.....	110
19.6. capabilityAlternateId	111
19.6.1. Description	111
19.6.2. Where to find.....	111
19.7. certificationVersionId	112
19.7.1. Description	112
19.7.2. Where to find.....	112
19.8. deviceAlternateId.....	113
19.8.1. Description	113
19.8.2. Where to find.....	113
19.9. endpointId	114
19.9.1. Description	114
19.9.2. Where to find.....	114
19.10. gatewayId.....	115
19.10.1. Description	115
19.10.2. Where to find.....	115
19.11. sensorAlternateId	116
19.11.1. Description	116
19.11.2. Where to find.....	116
19.12. TeamsetContextId	117
19.12.1. Description	117
19.12.2. Where to find.....	117
20. Certificates and keys	118
20.1. Definitions	118
20.2. Agrirouter public keys.....	118

	Title: agrirouter Connectivity-Platform Integration Guide Part 2 Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

20.3. Application keys	119
20.4. Endpoint certificate	120
21. Usage Metrics	121
22. General Certification process.....	123
22.1. General	123
22.2. Out of focus.....	123
22.3. Certification institutes	123
22.4. Tests	123
22.4.1. For Farming Software and Telemetry platforms	123
22.4.2. For Communication Units.....	123
23. Annex A (Glossary)	124
24. Annex B (Shortings).....	127
25. Annex C (Message Codes and Handling).....	128

 DKE DATA	Title: agrirouter Connectivity-Platform Integration Guide Part 2 Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

Table of Figures

Figure 1 agrirouter Solution.....	13
Figure 2 Using Postman for onBoarding	18
Figure 3 Adding the certificate in Postman	19
Figure 4 ARTS User Interface	22
Figure 5 Application overview.....	23
Figure 6 Application settings	24
Figure 7 fuf.me is an alias for localhost.....	24
Figure 8 Authentication Workflow	25
Figure 9 Application authentication screen	26
Figure 10 Example of an authentication result	27
Figure 11 Authentication for non-cloud-applications	28
Figure 12 General onboarding workflow	29
Figure 13 onboarding Workflow for a farming software or telemetry platform	30
Figure 14 Agrirouter onboarding process for a CU	31
Figure 15 Requesting a registration code in agrirouter UI.....	32
Figure 16 Registration code for a CU	32
Figure 17 Finding the applicationID	34
Figure 18 Finding the certificationVersionID	34
Figure 19 Generating Private and Public Key in the agrirouter UI	35
Figure 20 Revoke Process.....	45
Figure 21 Architecture of an endpoint.....	47
Figure 22 Inbox within an agrirouter endpoint	48
Figure 23 Outbox within an agrirouter endpoint.....	49
Figure 24 The feed within an agrirouter endpoint	50
Figure 25 Subscription List within an agrirouter endpoint.....	50
Figure 26 The onion principle for a non-telemetry message.....	51
Figure 27 The onion principle for a telemetry message	52
Figure 28 Request and Response in HTTP	53
Figure 29 REST Communication with the outbox	53
Figure 30 Request and Response using MQTT	54
Figure 31 Receiving Result from the outbox in MQTT	54
Figure 32 Abstraction of a Call or Message Sending to the Inbox	55
Figure 33 Abstraction of a result from the outbox	55
Figure 34 Send Message or Command to agrirouter	57
Figure 35 Command Flow.....	58
Figure 36 Command flow for reading the feed	58
Figure 37 Workflow of creating a message.....	60
Figure 38 Workflow of creating a chunked message.....	61
Figure 39 Selecting Technical Message Types in the agrirouter UI	95
Figure 40 General messaging workflow	106

	<p>Title: agrirouter Connectivity-Platform Integration Guide Part 2</p> <p>Author: DKE-Data GmbH & Co. KG</p>	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

Figure 41 agrirouter account ID	109
Figure 42 agrirouter endpoint software ID	110
Figure 43 agrirouter endpoint software version ID	112
Figure 44 agrirouter endpoint ID	114
Figure 45 Private and public key of an application in agrirouter	119

	<p>Title: agrirouter Connectivity-Platform Integration Guide Part 2</p> <p>Author: DKE-Data GmbH & Co. KG</p>	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

Document history

version	date	author	Description
1.0	11.07.2018	DKE-Data GmbH & Co. KG	First release
1.2	25.07.2018	DKE-Data GmbH & Co. KG	Added Remark on sending frequency Fixed Error in documentation of Results from the outbox (Single message instead of array) Link zum agrirouter toolset
1.3	04.09.2018	DKE-Data GmbH & Co. KG	Correction of Parameter id in onboarding (was Id) Fixed Error: Requests to Outbox are done using GET, not POST Added Definition for TeamsetContextID Added TypeURLs Added List of Message Types Added clarification on VirtualCUs Added external ID to endpoint list

Review

version	name	Department/Company	Location
1.0	13.07.2017	DKE-Data GmbH & Co. KG	
1.3	03.09.2018	DKE-Data GmbH & Co. KG	

	<p>Title: agrirouter Connectivity-Platform Integration Guide Part 2</p> <p>Author: DKE-Data GmbH & Co. KG</p>	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

1. Purpose

1.1. Purpose of this document

The purpose of this document is to create deep knowledge about the thirdparty integration into the agrirouter solution. This document shall include all information necessary to develop an integration with the agrirouter interface in an application. If you experience any problems with missing information in this document, please send an email to support@my-agrirouter.com mentioning "Integration Guide Part2" in the topic.

1.2. Purpose of further documents

This document is a follow up on the document "Integration Guide Part 1". If you are not yet familiar with the agrirouter topic, please read this document first. It can be found here: <https://my-agrirouter.com/support/>

There are several documents including further information on agrirouter and the used message formarts. Please check the list of additional resources at the end of this document.

	Title: agrirouter Connectivity-Platform Integration Guide Part 2 Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

2. Epilog

Digitization in agriculture makes good progress. One unsolved problem so far are the different and customized developments that are used for the data exchange, for instance between machine and software products of different manufacturers. DKE-Data GmbH & Co. KG has a solution for this.

The company DKE-Data GmbH & Co. KG (DKE is the German abbreviation for digital communication and development) is currently developing a cross-manufacturer agrirouter Solution that is far more than merely hardware and software. Customers are making use of a service: safe and cross-product data transport.

The agrirouter Solution allows farmers to transfer data between farming machinery from several manufacturers and several farming applications from different application providers. Users can configure data transfer permissions centrally on agrirouter Solution, to control what data can be exchanged between which endpoint. Data can be published on the agrirouter Solution and routed to interested and allowed endpoints.

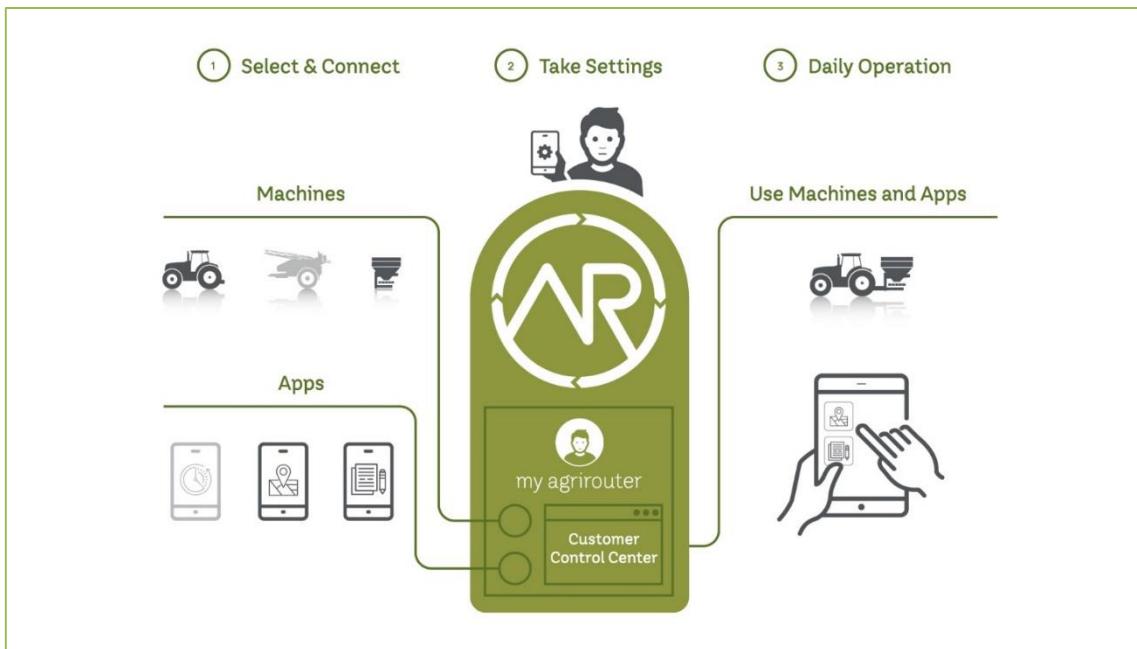


Figure 1 agrirouter Solution

With central maintenance of data routing between endpoints, there is no need to configure network connections between all data sources and targets on each machine/ or in each application.

The agrirouter Solution provides a common cross-vendor data exchange mechanism, based on a common message format, which can be implemented by different machine manufacturers and application providers.

The transmission and evaluation of live telemetry data is getting more and more important. With the agrirouter Solution, machines can send telemetry data to the agrirouter, where data is automatically distributed to several applications, based on permissions and configuration. For live telemetry data, the agrirouter filters the content of the messages according to the user-defined permissions.

This way it can be achieved that one application gets only application rate data, for example, while another one gets only machine-related data such as fuel consumption.

Technically, the agrirouter Solution is a cloud system, which provides the messaging service and a web site with self-services for the different user types.

	Title: agrirouter Connectivity-Platform Integration Guide Part 2 Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

When the recipient of a message is offline, the message is buffered on the agrirouter until the recipient is online again. However, the message is not buffered forever. After a maximum buffer time, the buffered messages are removed from the buffer even if they are not yet delivered.

The agrirouter Solution also stores additional data needed for rights management and message processing. This includes the list of endpoints attached to agrirouter account, and the technical device description of the connected agricultural machines. This information is needed for message routing and filtering and for automatic adjustment of message content by manufacturer-specific adapters. In addition, the agrirouter stores usage statistics data needed for billing and for ensuring a smooth and efficient function.

 DKE DATA	Title: agrirouter Connectivity-Platform Integration Guide Part 2 Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

3. How to use this document – an advice on development and more

This chapter shall describe a possible way of handling the integration with the agrirouter. It is not the only way of doing it, of course. For a valid and well communicated agrirouter integration, you should follow a mixture of business and development decisions:

1. **Get basic knowledge:** First, read the basic concepts of agrirouter architecture and ecosystem; described in part 1 of this guide.
2. **Decide, what you want to implement:**
 - a. **Read** the several non-agrirouter-relevant technical message types (e.g. TaskData, Images, Telemetry) as a general overview of possible capabilities.
 - b. **Decide, which messages are relevant** for you and which information you could consume or provide. This way, you could e.g. forward the requirement for a TaskController Integration or a Camera attachment possibility at a CU to the relevant departments and it's more common that it will be there in time.
 - c. **Synchronize your plans:** Dependend on your role in the company make sure, management, marketing, sales and developers have the same idea of the upcoming product
3. **Decide, how to implement:** Once you made these decisions, let's not care about the messages for a while and work out a possible architecture:
 - a. **Read** about the onion principle of communication and the command flow.
 - b. **Check** the general concepts around agrirouter like Protobuf, MQTT, REST and EFDI
 - c. **Check** for libraries and frameworks e.g. at the DKE GitHub.
 - d. **Decide**, how to build your architecture to make your system compatible for agrirouter or to build an interface service.
4. **Create an application in agrirouter**, to receive applicationID and CertificationID
5. **Start Development:**
 - a. **Build** your application and read about the several messages in detail to implement them
6. **Do prerelease work**
 - a. **PreCreate market place entry:** Even though your app is not yet tested, prepare the marked place form for the release of your product. Should be done by a sales or management person
 - b. **Check the markedplace entry:** Should be done by a developer: Read the marked place entry and make sure, you support all the described functionality as described in the marked place entry and in the integration guide or linked documents.
7. **Test your app:** agrirouter provides the possibility to select test users for a not yet certified app. Select some customers or your internal staff to check, if your app runs smooth with agrirouter and perhaps with common other endpoints. For a farming software, a corresponding Communication Unit could be such an endpoint
8. **Certify your app:** Select your desired test institute and let your app be certified for public release.
9. **Release to public:** Activate your market place entry and let people know, that your app is up and running.

 DKE DATA	Title: agrirouter Connectivity-Platform Integration Guide Part 2	
	Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

4. General conventions

4.1. Numbering convention

Within the agrirouter interface, there are several IDs or counters, that are of type int64. Due to difference between the protobuf implementation of C++ vs. e.g. Java, the ID "0" cannot be used.

Therefore:

All numberings begin with 1

4.2. String Identifiers convention

For **globally unique** instance ID we suggest using Uniform Resource Name (URN) according to RFC5141. Basically, this enabled an identification to be of any kind that is defined by the official IANA registered URN namespaces (<https://www.iana.org/assignments/urn-namespaces/urn-namespaces.xhtml>).

For example this could be a UUID if the CU is an App on a consumer device with a generated identity: urn:uuid:f81d4fae-7dec-11d0-a765-00a0c91e6bf6

Also this could identify an OEMs device with a Product Identification Information: urn:iso:std:iso:11783:-12:2013:ed-2:tech:pii:1234567890ABC*Brand Name*Product Name*.

4.3. TimeStamp conventions

If there is nothing different mentioned in the relevant chapter, all timestamps in messaging, onboarding etc. shall be in UTC. The common format for timestamps is ISO; e.g. "2018-06-14T14:53:28.123Z".

4.4. Time Synchronisation

As agrirouter exchanges data between different applications in different environments, it is very important to have a common current time. Therefore, you should match the result timestamp of an agrirouter command with your local time and compensate too big differences (>1 minute), when setting the timestamp of your agrirouter commands.

The best practice here is to request the endpoint list on a regular basis and take the timestamp of the result.

If your application has access to the GPS time, this should be enough to keep the same time as the agrirouter.

Please note:

The agrirouter support team also suggests using this compensation time in your TaskController- and DataLogger- Data. Even though, this is not relevant for a proper communication with the agrirouter, it will avoid timing problems in the analysis of data with applications, that receive your data.

Example:

If multiple machines with multiple CUs work on a field, you'll never be able to match their EFDI data, if you have different times

	Title: agrirouter Connectivity-Platform Integration Guide Part 2 Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

5. Environments

agrirouter currently supports 2 environments of the whole agrirouter connectivity platform.

5.1. Quality assurance

The quality assurance environment (“QA”) is used to develop and test applications in a save and uncritical eco system. Please note, that the quality assurance environment shall not be used by productive customers.

5.2. Productive

The productive environment is used by your endcustomers with tested and certified apps. This environment is very scalable and made for a huge number of users.

5.3. Areas

Currently, there is only one Area: EU1. Every app instance is onboarded to environments in this area. With upcoming availability of the agrirouter in more and more countries, there might be additional areas created. This could happen for performance reasons or legal requirements of different countries. If new areas become required, this document will be updated, providing all the required information.

	<p>Title: agrirouter Connectivity-Platform Integration Guide Part 2</p> <p>Author: DKE-Data GmbH & Co. KG</p>	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

6. Development tools

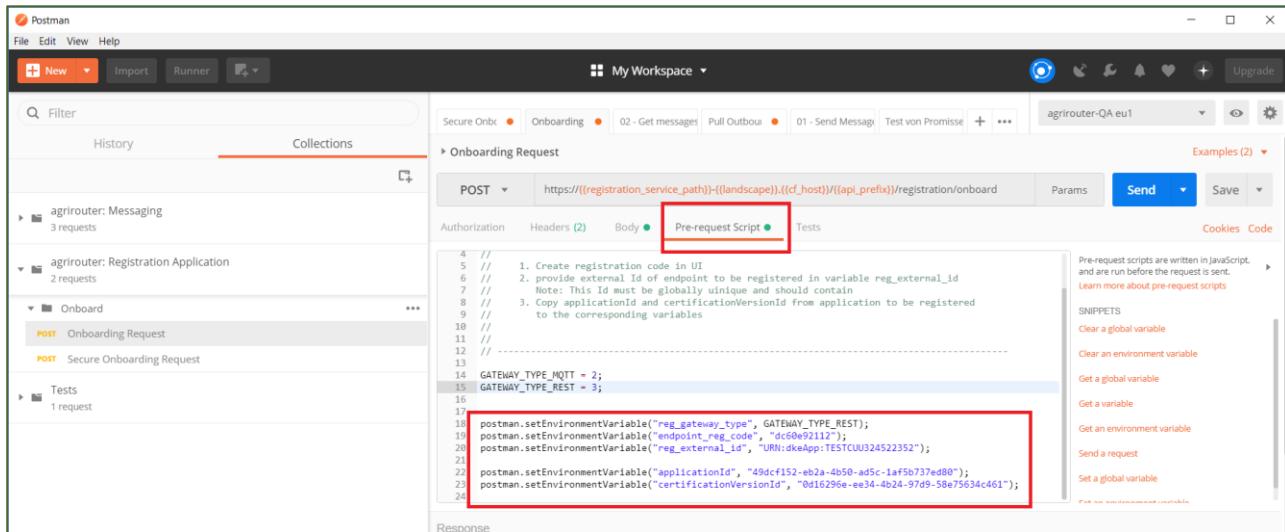
6.1. REST using Postman

Postman is a tool to send http requests and analyze results, usable for full test chains for online APIs. You can download postman here: <https://www.getpostman.com/>

With this integration guide, a postman collection is delivered, including the endpoint definitions and abstractions for the different environments.

6.1.1. Check onboarding

To do an onboarding request, open the Agrirouter: Application Registration/OnBoard/Onboarding Request
Open the Pre-Request-Script and fill the variable field names



```

4 // 1. Create registration code in UI
5 // 2. provide external Id of endpoint to be registered in variable reg_external_id
6 // Note: This Id must be globally unique and should contain
7 //        3. Copy applicationId and certificationVersionId from application to be registered
8 //        to the corresponding variables
9 //
10 //
11 //
12 //
13
14 GATEWAY_TYPE_MQTT = 2;
15 GATEWAY_TYPE_REST = 3;
16
17
18 postman.setEnvironmentVariable("reg_gateway_type", GATEWAY_TYPE_REST);
19 postman.setEnvironmentVariable("endpoint_reg_code", "dc60w92112");
20 postman.setEnvironmentVariable("reg_external_id", "URN:dkeApp:TESTCLU324522352");
21
22 postman.setEnvironmentVariable("applicationId", "49df1f52-eb2a-4b50-ad5c-1af5b737ed80");
23 postman.setEnvironmentVariable("certificationVersionId", "0d16296e-ee34-4b24-97d9-58e75634c461");
24

```

Figure 2 Using Postman for onBoarding

6.1.2. Adding a Certificate

For an SSL Secured communication, add the certificate delivered with the onboarding request:

Copy&paste ENCRYPTED PRIVATE KEY section to new file

Replace all \n with new lines.

Save both sections (ENCRYPTED PRIVATE KEY and CERTIFICATE) in separate files (e. g. DEVICE_MYEXTERNALID.key, DEVICE_MYEXTERNALID.crt)

Provide both in Postman Settings / Certificates, including the passphrase provided in the onboarding response.

	Title: agrirouter Connectivity-Platform Integration Guide Part 2 Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2		Confidence: external
Status: final	Version: 1.3	September 4, 2018

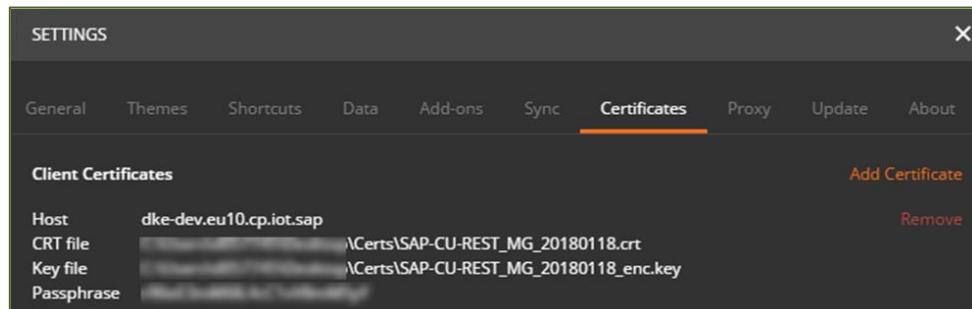


Figure 3 Adding the certificate in Postman

 DKE DATA	Title: agrirouter Connectivity-Platform Integration Guide Part 2	
	Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

6.2. MQTT using Paho

Paho is an Eclipse tool to connect to an MQTT Broker.

First, read the getting Started Guide: <https://help.sap.com/viewer/e765b2a5b99540ce84da397c20cc1993/Cloud/en-US/4954c48d016f40eeb5fcfa1cd596b909.html>

Secondly, download and install Paho: <https://help.sap.com/viewer/e765b2a5b99540ce84da397c20cc1993/Cloud/en-US/53ad3e974dba4d49970113e0fd80ab07.html>

6.2.1. Format Certificate for Paho

The certificate needs to be formatted to work with PAHO:

Copy the whole certificate field, replace the \n with new Lines (\r\n) and save it in an *.pem file

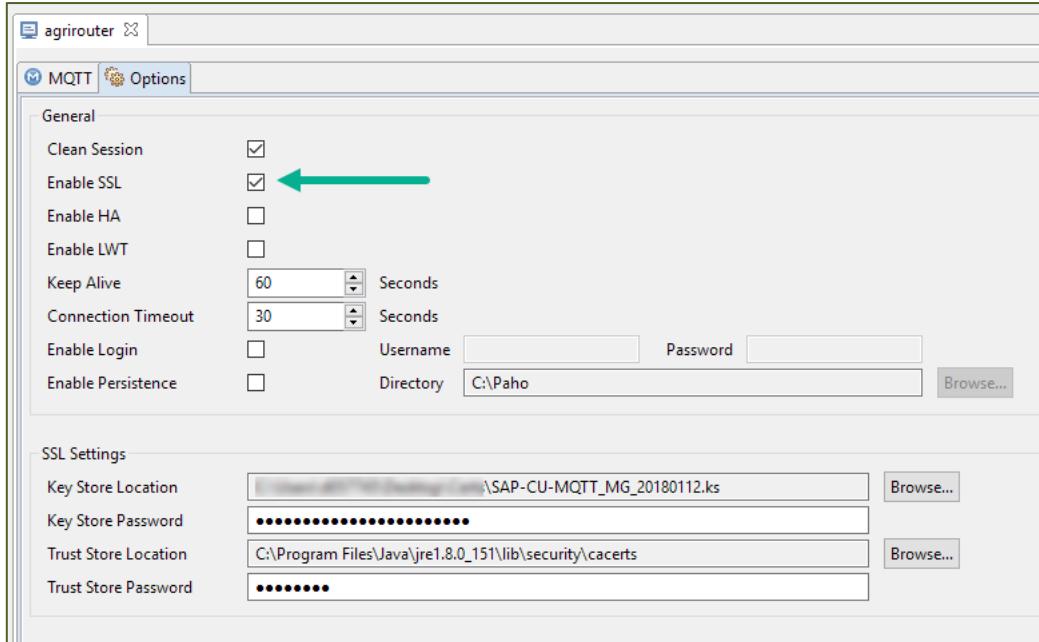
Install openssl.

Format the certificate:

```
$ openssl pkcs12 -export -in SAP-CU-MQTT_MG_2018012.pem -inkey SAP-CU-MQTT_MG_2018012.pem -out mqtt_client.ks
Enter pass phrase for SAP-CU-MQTT_MG_2018012.pem:
Enter Export Password:
Verifying - Enter Export Password:
```

6.2.2. Setup Paho Certificate

To setup the certificate, open the connection options and enable SSL. Then add the certificate file



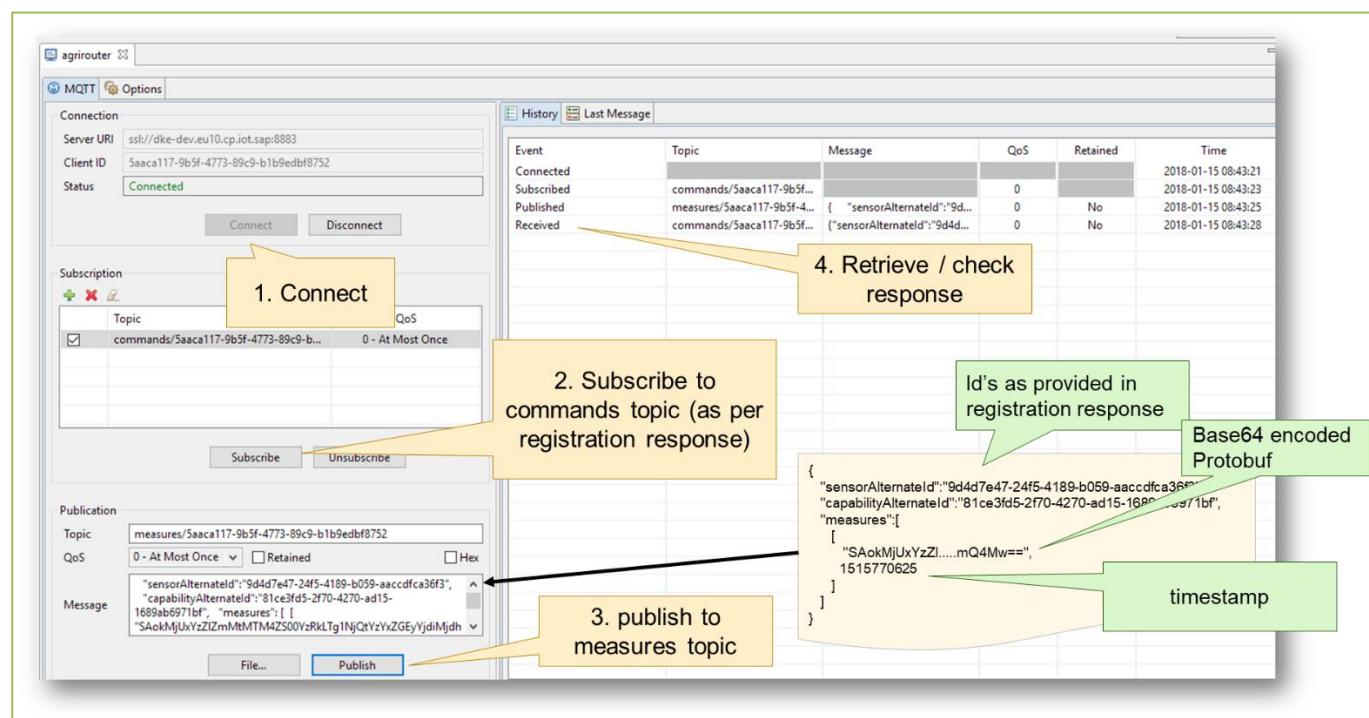
Further information: <https://help.sap.com/viewer/e765b2a5b99540ce84da397c20cc1993/Cloud/en-US/711642e551f4477c882c5561853ee7c7.html>

 DKE DATA	Title: agrirouter Connectivity-Platform Integration Guide Part 2	
	Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

6.2.3. Setup the connection criteria of Paho:



6.2.4. Basic functions



	<p>Title: agrirouter Connectivity-Platform Integration Guide Part 2</p> <p>Author: DKE-Data GmbH & Co. KG</p>	
Identification: agrirouter Integration Guide Part 2		Confidence: external
Status: final	Version: 1.3	September 4, 2018

6.3. agrirouter protobuf toolset

The agrirouter protobuf toolset ("ARTS") is a Java Project to check or create several parameters for the agrirouter communication. It can for example decode received Base64 Strings to their Protobuf representation. The tool includes a command line as well as a user interface.

The project can be found here: <https://github.com/saschadoemer/agrirouter-protobuf-toolset>

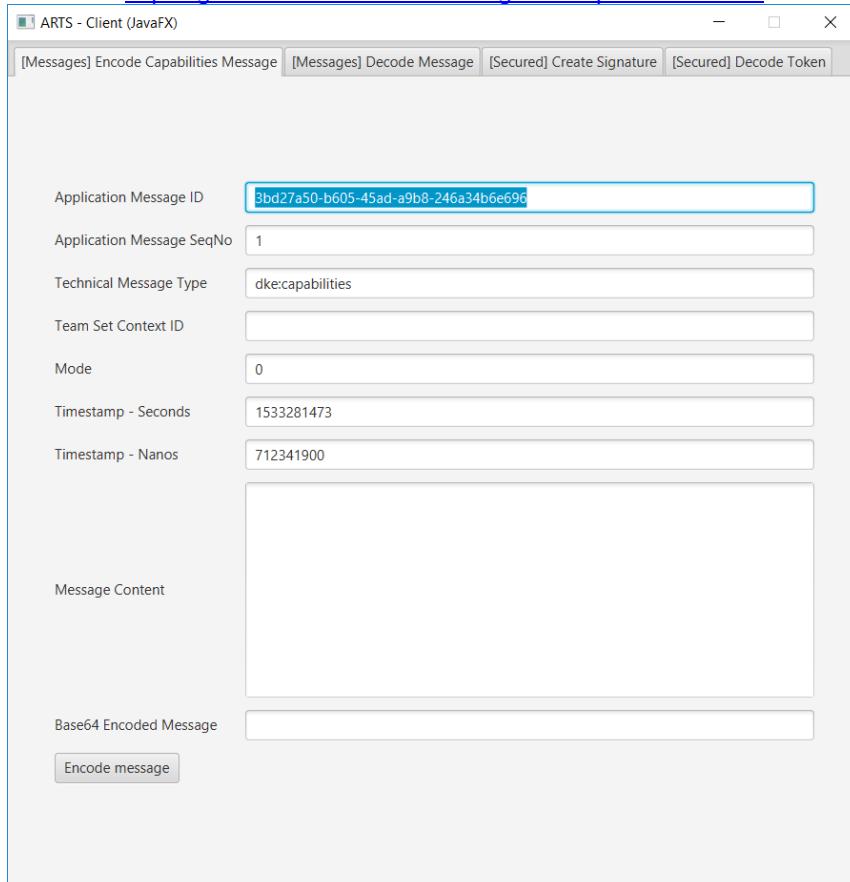


Figure 4 ARTS User Interface

Remark

The ARTS-Project is an open source project, feel free to participate and extend its functionality.

Please note:

This tool is not an official part of the agrirouter eco system and DKE-Data is not liable for it.

 DKE DATA	Title: agrirouter Connectivity-Platform Integration Guide Part 2 Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

7. Authentication Process

The authentication process was designed to give application providers of consuming applications (e.g. farming software, but in general every application that consumes data) the chance to make sure, they know, which account is assigned to which agrirouter account. Otherwise, fake accounts could consume data on their costs.

Remark: As the authentication process – in difference to the rest of the communication – provides a request to your agrirouter integration and not the response to a request from your agrirouter integration, the documentation of the parameters is done in a little different way and there is no description for a HTTP Response code, because we don't get one.

7.1. Authentication of farming software

As application providers of farming software have to pay for the receiving of raw data, the application provider should make sure, that only such agrirouter accounts can onboard one of his applications, that he can assign to one of his users.

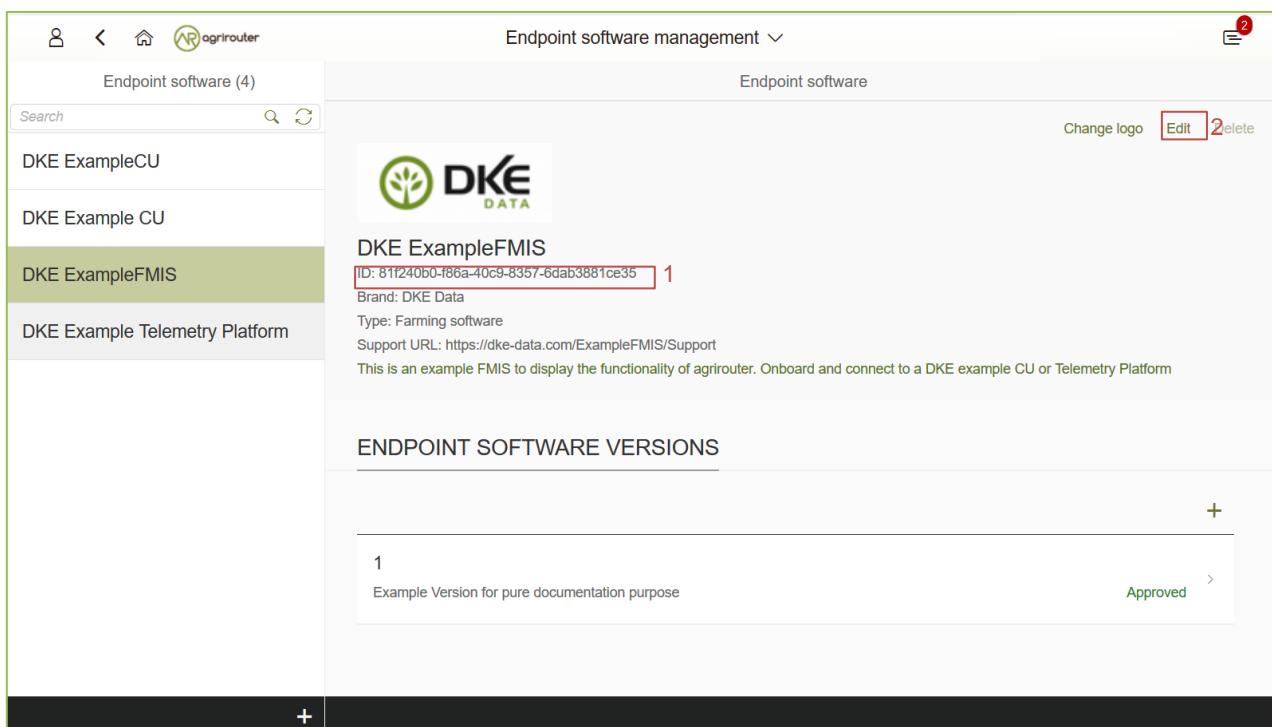
Remark: As the user has to log on to agrirouter, the authentication process requires a browser UI!

7.1.1. Preparation

To provide authentication, the developer first has to setup the application.

In the developers UI, he has to:

- Copy the application ID (1)
- Call Edit (2)
- Provide a redirect URL, to which agrirouter shall deliver the authentication result using HTTP 301 Browser redirect.(3)
- Provide a public key for his encryption or create a key pair on the agrirouter frontend



The screenshot shows the 'Endpoint software management' section of the agrirouter interface. On the left, a sidebar lists existing applications: 'DKE ExampleCU', 'DKE Example CU', 'DKE ExampleFMIS' (which is highlighted in green), and 'DKE Example Telemetry Platform'. The main area is titled 'Endpoint software' and shows a single entry for 'DKE ExampleFMIS'. This entry includes the ID '81f240b0-f86a-40c9-8357-6dab3881ce35' (with a red box around it), the brand 'DKE Data', type 'Farming software', and support URL 'https://dke-data.com/ExampleFMIS/Support'. A note below states 'This is an example FMIS to display the functionality of agrirouter. Onboard and connect to a DKE example CU or Telemetry Platform'. Below this, a section titled 'ENDPOINT SOFTWARE VERSIONS' shows a single version entry with ID '1', status 'Approved', and a note 'Example Version for pure documentation purpose'. At the bottom of the page is a large black footer bar with a '+' button.

Figure 5 Application overview

 DKE DATA	Title: agrirouter Connectivity-Platform Integration Guide Part 2	
	Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2		Confidence: external
Status: final	Version: 1.3	September 4, 2018

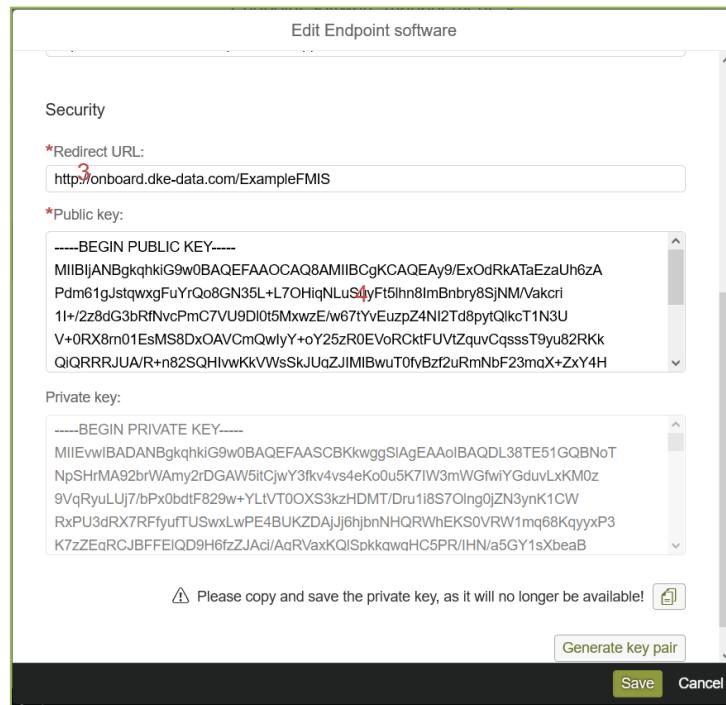


Figure 6 Application settings

Remark

If you do not have an official domain, where your application is reachable online, you can use <http://fuf.me>. This URL is a synonym for localhost or 127.0.0.1, but it meets the requirements of the agrirouter UI for redirect URLs. Simply check Tracert/TraceRoute:

```
Microsoft Windows [Version 10.0.16299.492]
(c) 2017 Microsoft Corporation. Alle Rechte vorbehalten.

C:\Users\frank>tracert fuf.me

Routenverfolgung zu fuf.me [127.0.0.1]
über maximal 30 Hops:

 1  <1 ms    <1 ms    <1 ms  LAPTOP-JNS882J8 [127.0.0.1]

Ablaufverfolgung beendet.
```

Figure 7 fuf.me is an alias for localhost

 DKE DATA	Title: agrirouter Connectivity-Platform Integration Guide Part 2	
	Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

7.1.2. Authentication Process Overview

The authentication process works as follows:

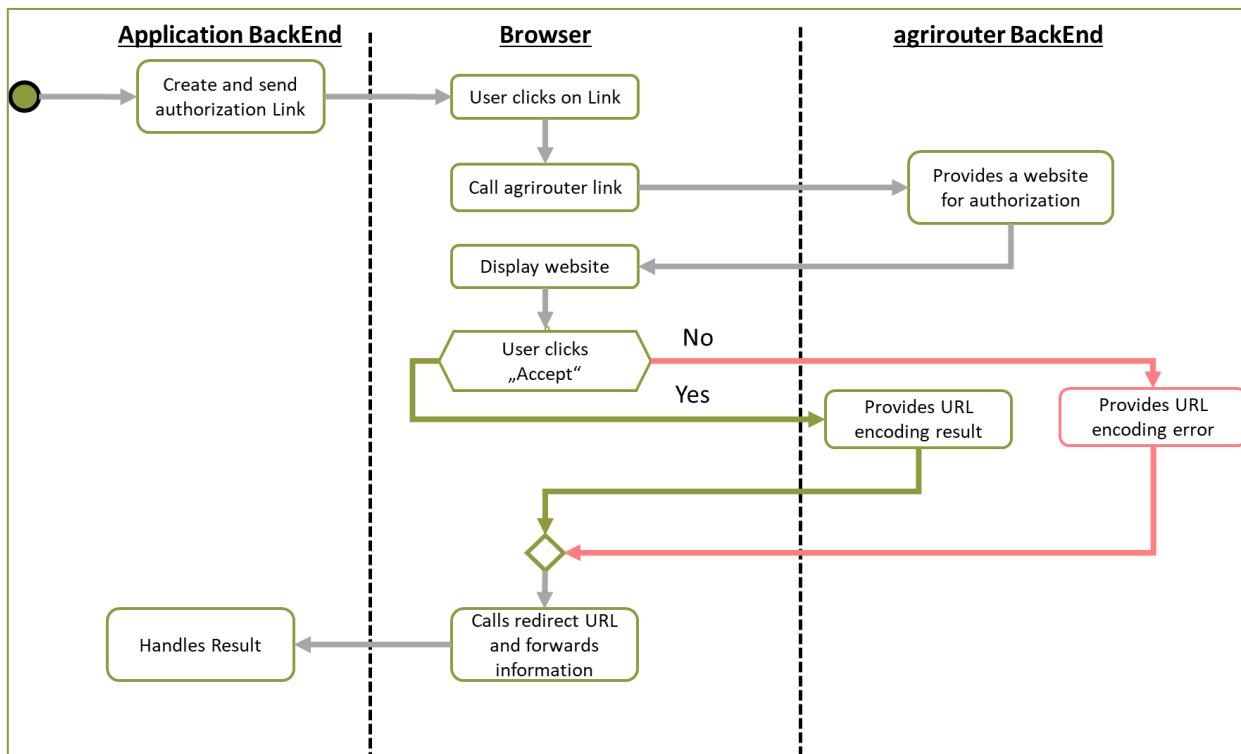


Figure 8 Authentication Workflow

To better understand, what happens here, try the following:

1. Call <https://httpbin.org/get> in your browser. You'll get a JSON view of the get request
2. Call <https://httpbin.org/get?Param1=Value1&Param2=Value2> in your browser. You'll get a view of the get request
⇒ <https://httpbin.org> simply echoes the request that is send to the page. That's important to understand

Remark

For testing purpose, you can just enter the url <https://httpbin.org/get> in your applications redirect URL (see below) to see the result of authentication.

Remark

The step "user clicks on Link" might not be needed, applications could handle that different.

For example the application could send a redirect (HTTP Status 300) to directly redirect the user to the agrirouter Connection Website. The description "user clicks on Link" is simply the most understandable description we could come up with.

 DKE DATA	Title: agrirouter Connectivity-Platform Integration Guide Part 2 Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

7.1.3. Generating an authentication URL

Area	Environment	URL
EU1	Quality Assurance	https://agrirouter-qa.cfapps.eu1.hana.ondemand.com/
EU1	Productive	https://goto.my-agrirouter.com/

The authentication Link is a HTTP GET Request that has to be called from a browser.

Method	Address
GET	/application/{{applicationID}}/authorize

To provide a link for authentication, create a link like this:

`{{agrirouter-url}}/application/{{applicationID}}/authorize?{{response_type}}&{{state}}&{{redirectURL}}`

Parameter	Example Value	Remark
<code>{{agrirouter-url}}</code>	see above	Differentiates between QA and Live system
<code>{{applicationID}}</code>	Noted from the agrirouter UI	
<code>{{response_type}}</code>	<code>response_type=onboard</code>	Possible values: verify: only verify the user, onboard: verify user and create a Registration Code (Token)
<code>{{state}}</code>	<code>state=w4st556dr543d4wr4s4</code>	A number to identify the request result on server side. The provided Number should be: <ul style="list-style-type: none"> • Unique • Not guessable
<code>{{ redirect_uri}}</code>		Could extend your entered redirect URL

Caution:

Calling this link will deliver a website to log in to agrirouter, therefore, this link has to be called through a browser!

Remark

The response type onboard can be used to onboard farming applications without having to create a Registration Code in the agrirouter UI.

7.1.4. Perform authentication

When the user clicks on the link, the agrirouter website is called. If the user is currently not logged in, he has to log in. After logging in, he is delivered a website to authorize the connection between agrirouter and the application provider:

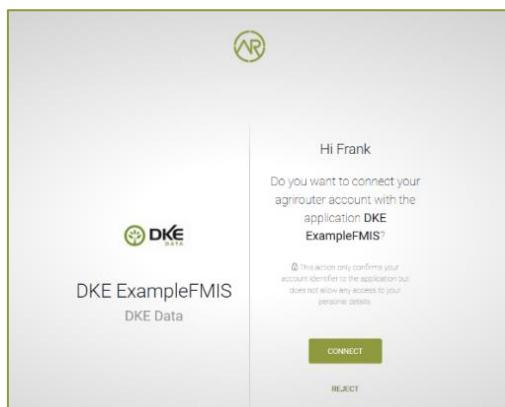


Figure 9 Application authentication screen

 DKE DATA	Title: agrirouter Connectivity-Platform Integration Guide Part 2	
	Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

7.1.5. Analyse result

agrirouter sends an HTTP 301 redirect to the browser, encoding the authentication result in a get queue attached to the Redirect URL entered in the developers' application settings.

The browser reacts in requesting this redirect URL which performs a GET request at the endpoint of the address.
The following parameters will be delivered in the GET-Queue:

Position	Name	Type	Description
1	signature	String	A signature to verify, that the source of the message is the agrirouter
2	state	String	The value that was passed to the agrirouter in parameter State
3	token	String	A base64 encoded JSON Object as Result
(3)	error	String	If error is delivered, user declined connection!

```
{
  - args: {
    signature: "H45ZxgsWJZzYZgyBCrxnUAKpdDKuJy0BwXYztenNZI73+Rxb/pFhtc6x7YUAgMAQE1qXV0mM9hrXpXwEhdqr21+NRAEY9",
    state: "b862103e-7689-4a7a-9cab-f60c98448215",
    token: "eyJhY2NvdW50IjoiNTA5ZTY1YWEtODRkZi00YzRkLWJmOTgtYzMyOGYwODQ2NGM1IiwicmVnY29kZSI6ImQ2ND",
  },
  - headers: {
    Accept: "text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8",
    Accept-Encoding: "gzip, deflate",
    Accept-Language: "en,de-DE;q=0.9,de;q=0.8,en-US;q=0.7",
    Cache-Control: "max-age=0",
    Connection: "close",
    Host: "httpbin.org",
    Upgrade-Insecure-Requests: "1",
    User-Agent: "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)"
  },
  origin: "88.130.2.214",
  url: "http://httpbin.org/get?state=b862103e-7689-4a7a-9cab-f60c98448215&token=eyJhY2NvdW50IjoiNTA5ZTY1YWEtODRkZi00YzRkLWJmOTgtYzMyOGYwODQ2NGM1IiwicmVnY29kZSI"
}
```

Figure 10 Example of an authentication result

7.1.5.1. Checking for errors

If the result includes a parameter **error**, the request was declined. Possible values:

Value	Description
request_declined	The user clicked on "decline"

7.1.5.2. Checking authenticity

Before analyzing the result, which is encoded in the **token**, it should be made sure, that the result (provided to the browser and from there to the application providers server) is really provided by the agrirouter.

Steps:

- concatenate **state** and **token** from the query
- create the SHA256 hash of the concatenated string
- verify the authenticity of the **signature** with the agrirouter public key and generated hash

Remark

The public keys can be found at Certificates and keys

 DKE DATA	Title: agrirouter Connectivity-Platform Integration Guide Part 2 Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

7.1.5.3. Analyse the result in token

The result token is a base64 encoded JSON object including the following parameters:

Name	Type	Description
account	String	The unique id of the user account on agrirouter , that will be provided to you in the metrics exports for billing
regcode	String	If response-type=onboard, this regcode will deliver a Registration Code equal to clicking the generate TAN-Button in the agrirouter ui
expires	DateTime	The date and time (in UTC), when the regcode becomes invalid

Example:

```
{
  "account": "31c83d5d-c307-42f9-80b1-6fc9324823b8",
  "regcode": "f75bfb41b",
  "expires": "2018-02-27T10:49:04.901Z"
}
```

7.2. Authentication for CUs and non-cloud-software

To perform authentication for software, that is not provided as a cloud solution, a small cloud onboarding service could be created to handle the onboarding communication:

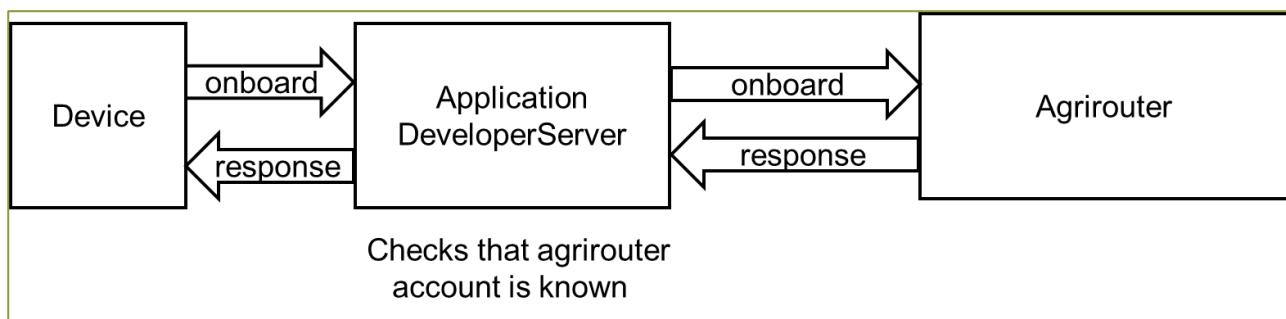


Figure 11 Authentication for non-cloud-applications

 DKE DATA	Title: agrirouter Connectivity-Platform Integration Guide Part 2	
	Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

8. Onboarding App Instances

8.1. General Overview

The onboarding workflow is needed to onboard app instances and create endpoints in the agrirouter. The result of the onboarding process has to be stored.

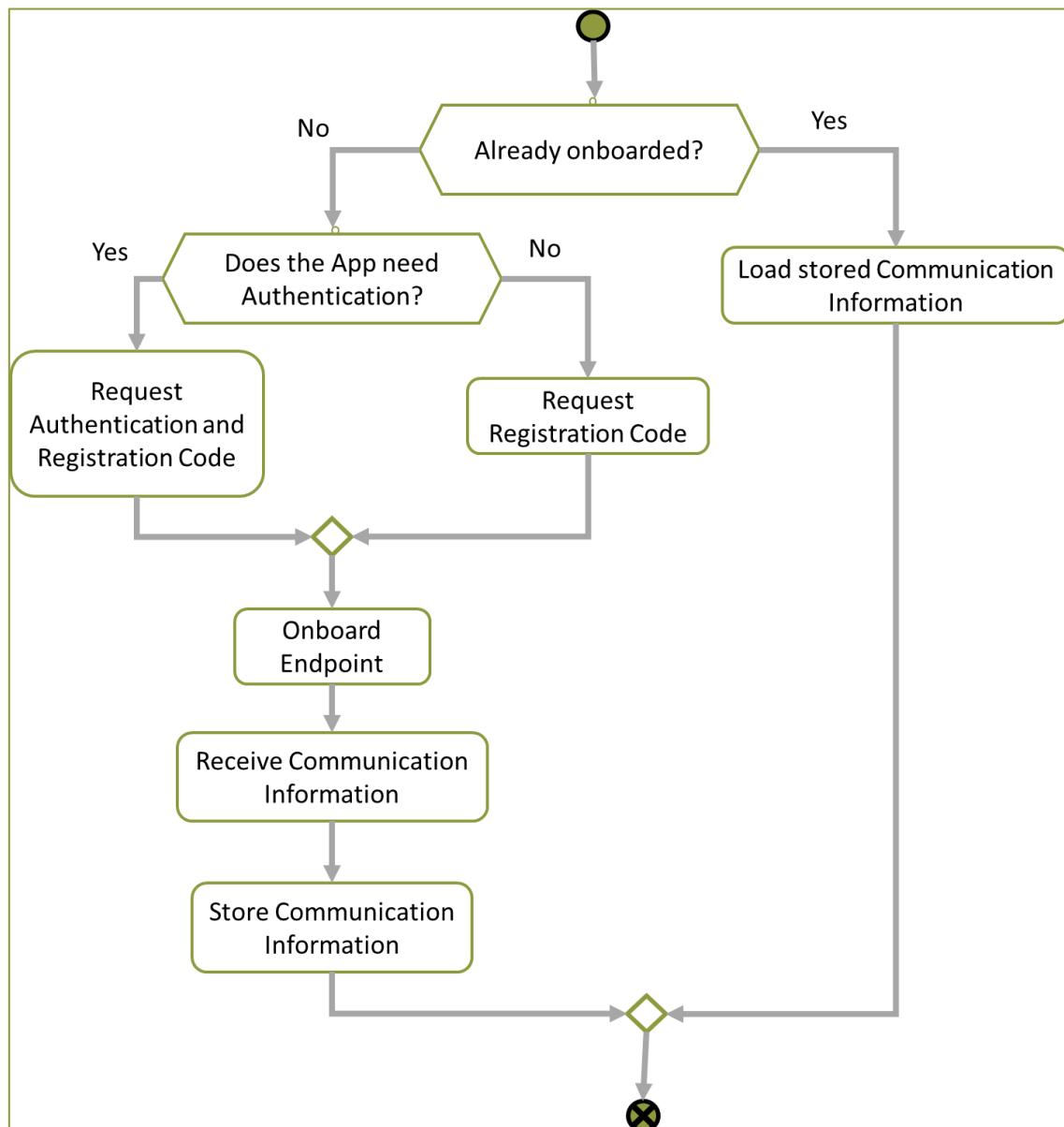


Figure 12 General onboarding workflow

 <p>DKE DATA</p>	<p>Title: agrirouter Connectivity-Platform Integration Guide Part 2</p> <p>Author: DKE-Data GmbH & Co. KG</p>	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

8.2 Workflow for Farming Software and Telemetry Systems

Telemetry systems and Farming Software need to assign their own userID to the accountID of the user at the agrirouter. Therefore, a verification should be performed before onboarding.

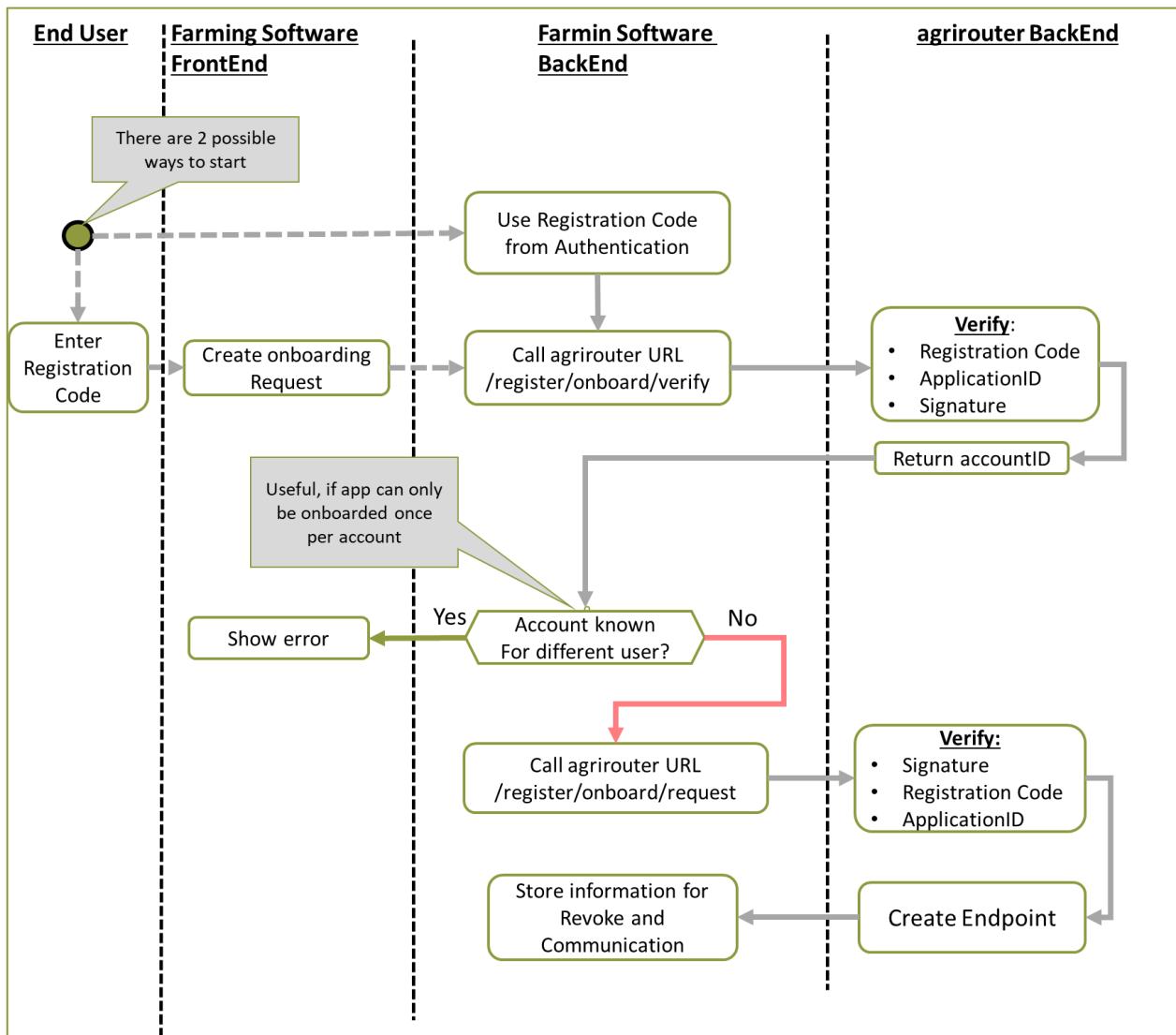


Figure 13 onboarding Workflow for a farming software or telemetry platform

 DKE DATA	Title: agrirouter Connectivity-Platform Integration Guide Part 2	
	Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

8.3. Workflow for CUs

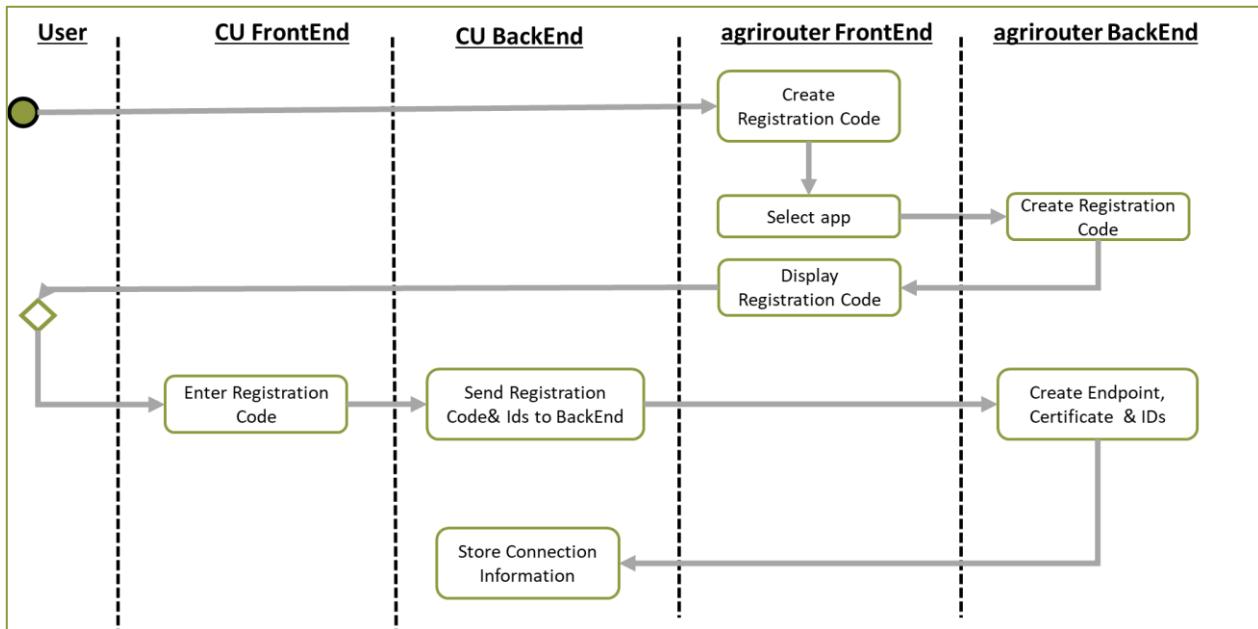


Figure 14 Agrirouter onboarding process for a CU

Remark

The CU onboarding mechanism needs less steps than the Farming Software and Telemetry software, as it is only required for those applications consuming data to know which user is assigned to which account.

The following parts of onboarding are **not** required for CU onboarding:

- Authentication
 - o Entering certificates in the agrirouter UI
 - o Adding a redirect URL in the agrirouter UI
- Verification
- onboarding with a Signed request

8.4. Workflow for Virtual CUs

Onboarding virtual CUs through a telemetry platform is done using a command. This will in particular be described later. See 16.5 Commands for telemetry platforms

From the end user's perspective, it is mostly dependend on the telemetry platform, if there is any selection mechanism or if it is done automatically.

	<p>Title: agrirouter Connectivity-Platform Integration Guide Part 2</p> <p>Author: DKE-Data GmbH & Co. KG</p>	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

8.5. Creating a Registration Code

8.5.1. Using the agrirouter UI

The Registration code for a new CU can be created by the end user clicking “Generate TAN”(1) in agrirouters’ control center. He has to select the desired CU (2) and gets a 10-digit code consisting of letters and numbers (3). A CU needs an interface to input this registration code.

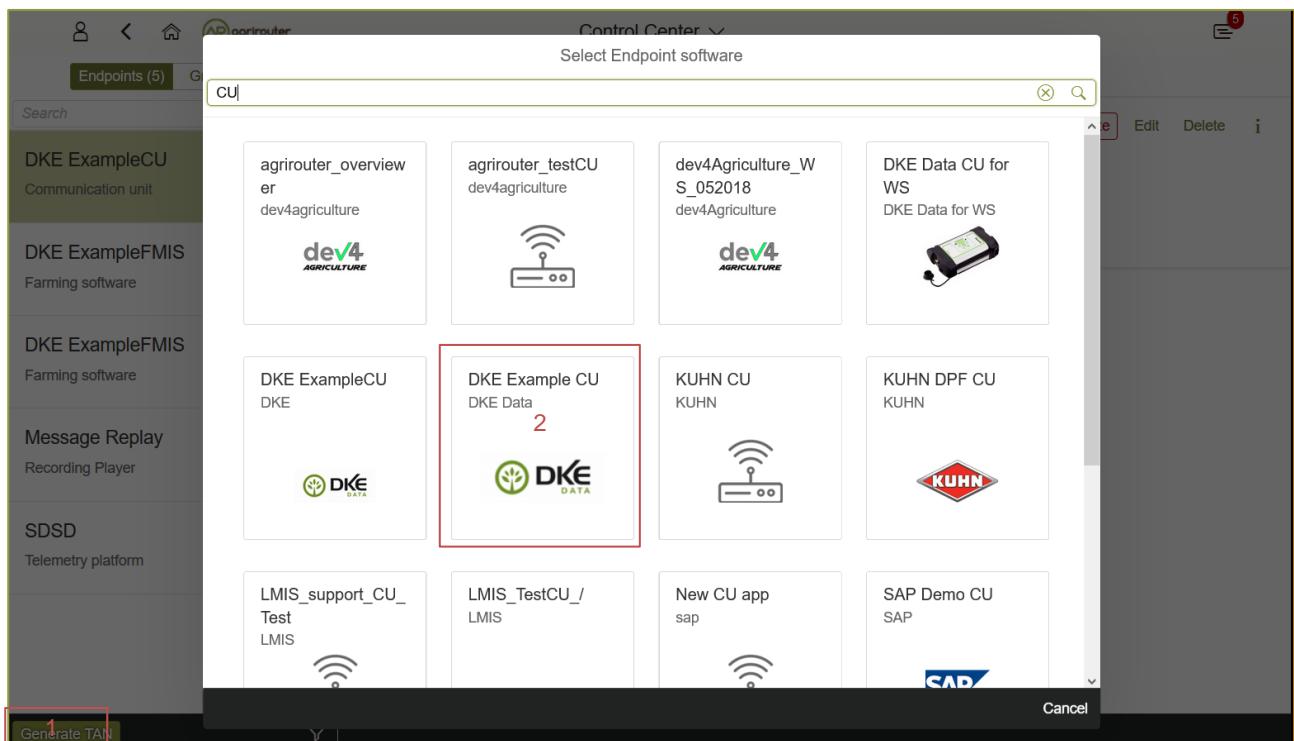


Figure 15 Requesting a registration code in agrirouter UI

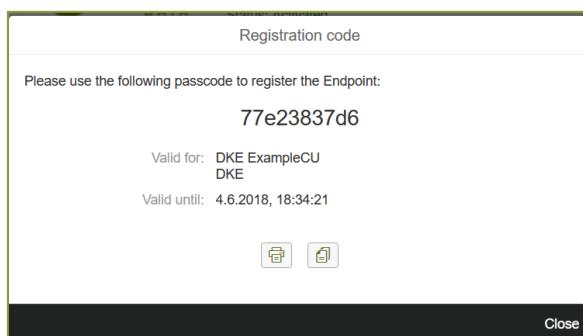


Figure 16 Registration code for a CU

	Title: agrirouter Connectivity-Platform Integration Guide Part 2 Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

8.5.2. As result of the authentication

If the authentication process is done using parameter `response_type=onboard`, the result will include a regcode. This regcode is a TAN.

8.5.3. Telemetry Platforms

A telemetry platform has to be onboarded using the authentication process. Once it is onboarded, it can onboard virtual CUs by itself. Therefore, it can use a special command. as Commands are described in a later chapter, please refer to 16.5.1 Onboarding a Virtual CU.

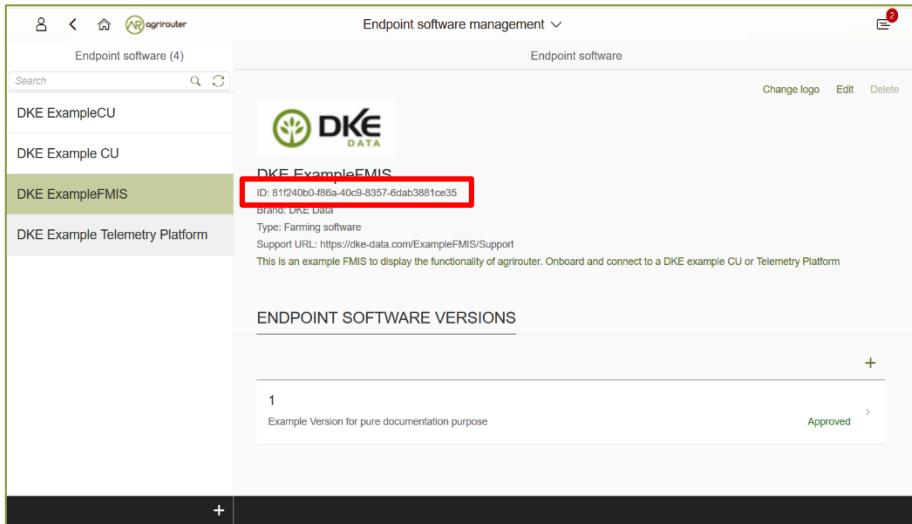
 DKE DATA	Title: agrirouter Connectivity-Platform Integration Guide Part 2 Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

8.6. Collecting and setting up relevant information

The onboarding request requires several different information:

8.6.1. ApplicationID and X-Agrirouter-ApplicationID

Both IDs are the same, they can be found in the agrirouter software endpoint UI of the developers account:

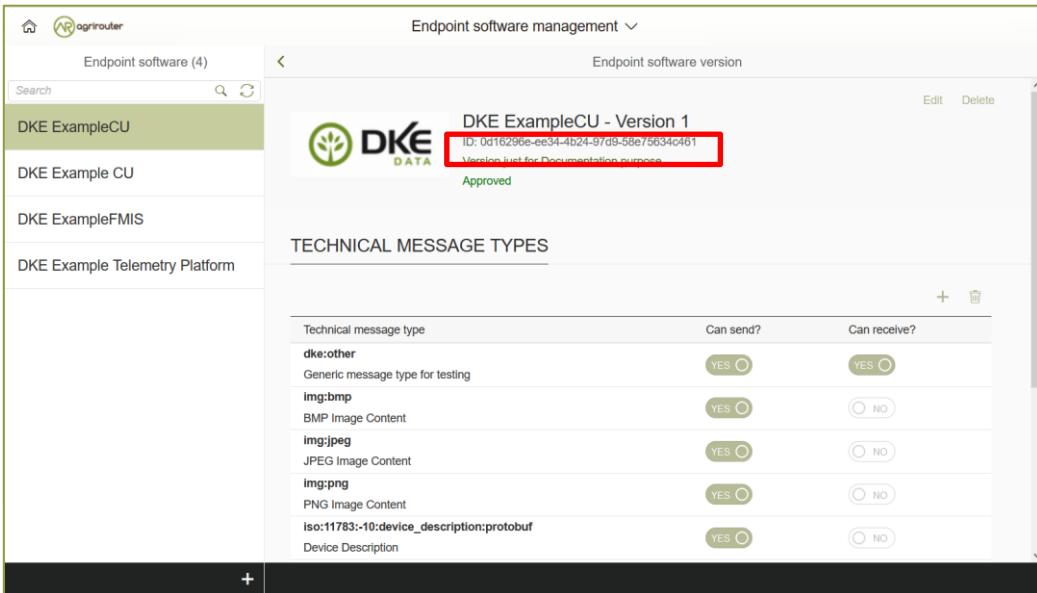


The screenshot shows the agrirouter software management interface. On the left, there's a sidebar with a search bar and a list of endpoint software entries: DKE ExampleCU, DKE Example CU, DKE ExampleFMIS (which is highlighted in green), and DKE Example Telemetry Platform. The main panel is titled "Endpoint software management" and shows the details for "DKE ExampleFMIS". It includes a logo for DKE DATA, the ID "ID: 812460b0-f86a-40c9-8357-6dbab3881ce35", the brand "Brand: DKE Data", type "Farming software", support URL "https://dke-data.com/ExampleFMIS/Support", and a note about it being an example FMIS. Below this, there's a section titled "ENDPOINT SOFTWARE VERSIONS" with one entry: "1 Example Version for pure documentation purpose" with status "Approved".

Figure 17 Finding the applicationID

8.6.2. CertificationVersionID

The CertificationVersionID is the ID unique to this specific app certification process. It can be found when clicking on the required Endpoint Software Version:



The screenshot shows the agrirouter software management interface, specifically the "Endpoint software version" section for "DKE ExampleCU". It displays the entry "DKE ExampleCU - Version 1" with the ID "ID: 0d16296e-ee34-4b24-97d9-58e75634c461" and a note "Version just for Documentation purpose". Below this, there's a section titled "TECHNICAL MESSAGE TYPES" with a table:

Technical message type	Can send?	Can receive?
dke:other	<input checked="" type="radio"/> YES	<input checked="" type="radio"/> YES
Generic message type for testing	<input checked="" type="radio"/> YES	<input type="radio"/> NO
img:bmp	<input checked="" type="radio"/> YES	<input type="radio"/> NO
BMP Image Content	<input checked="" type="radio"/> YES	<input type="radio"/> NO
img:jpeg	<input checked="" type="radio"/> YES	<input type="radio"/> NO
JPEG Image Content	<input checked="" type="radio"/> YES	<input type="radio"/> NO
img:png	<input checked="" type="radio"/> YES	<input type="radio"/> NO
PNG Image Content	<input checked="" type="radio"/> YES	<input type="radio"/> NO
iso:11783-10:device_description:protobuf	<input checked="" type="radio"/> YES	<input type="radio"/> NO
Device Description	<input checked="" type="radio"/> YES	<input type="radio"/> NO

Figure 18 Finding the certificationVersionID

	<p>Title: agrirouter Connectivity-Platform Integration Guide Part 2</p> <p>Author: DKE-Data GmbH & Co. KG</p>	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

8.6.3. Setting up private and public key

Remark

This step is not required for CUs

As the requests need to be signed (see 8.8 Signing requests), the public key has to be stored within the agrirouter. This can be done, calling “Edit” on the Endpoint Software Management Screen (see Figure 17 Finding the applicationID). The agrirouter UI offers the possibility to create a key pair, you can however create your own one and just store the public key on agrirouter.

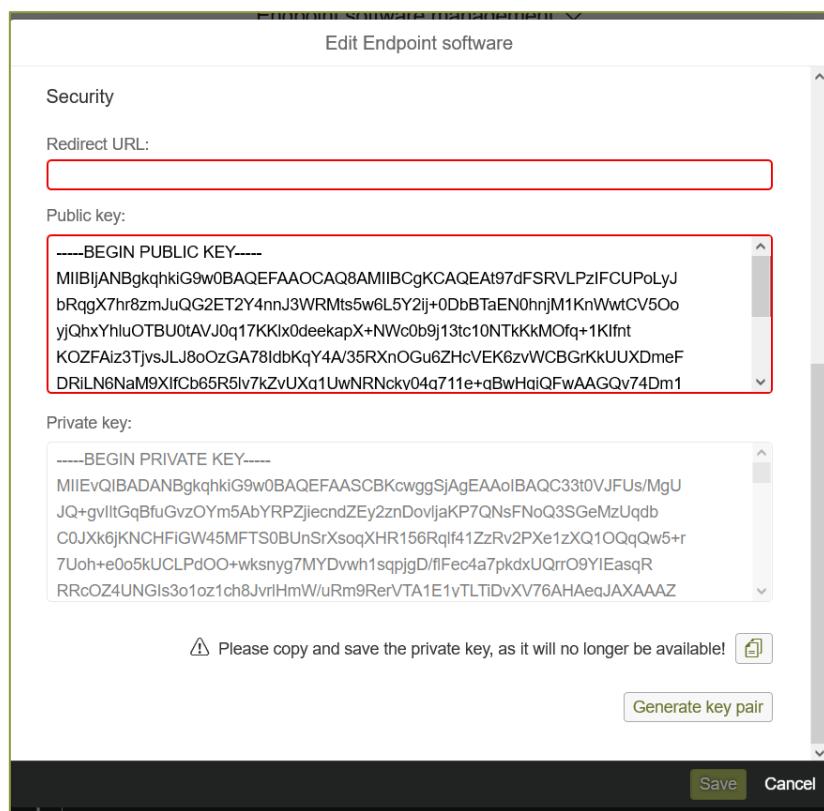


Figure 19 Generating Private and Public Key in the agrirouter UI

 DKE DATA	Title: agrirouter Connectivity-Platform Integration Guide Part 2	
	Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

8.7.onboarding Endpoint URLs

The endpoint URL differs, depending on your desired geolocation and the Quality Assurance or Productive Environment. The request must be a HTTP Post request to

Environment	Area	URL
Quality Assurance	EU1	https://agrirouter-registration-service-hubqa-eu1.cfapps.eu1.hana.ondemand.com
Production	EU1	https://agrirouter-registration-service.cfapps.eu1.hana.ondemand.com

Remark

Like every URL in this document, these URLs might change in the future or there might be additional ones for new Areas.

8.8.Signing requests

For onboarding, the agrirouter must be sure, that the requests actually come from an instance of the app specified in the request. Therefore, the payload must be signed with the applications private key. The corresponding public key must be maintained by the developer in agrirouter per application, see Setting up private and public key.

A Payload encryption is not needed since all communication is encrypted with TLS

All signatures used for the onboarding and revocation process shall be created by:

- hashing the request body (SHA256)
- then using the private key to create an RSA signature of the hash
- Create HEX representation of this hash
- Add the hex string as "X-Agrirouter-Signature" to the header of the HTTP call

Agrirouter will look up the public key for the app id specified and verify the signature.

 DKE DATA	Title: agrirouter Connectivity-Platform Integration Guide Part 2 Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

8.9. Sending a Verification Request

The verification request is used to actually check, if the endpoint is for the desired application and account before actually onboarding it.

8.9.1. General

The address for the verification request is as follows:

Method	Address
POST	api/v1.0/registration/onboard/verify

8.9.2. Request Information

The app instance has to send a HTTP Post request.

8.9.2.1. Header Information

8.9.2.1.1. For CU onBoarding

For CUs, this is not required or available

8.9.2.1.2. For FarmingSoftware and Telemetry Platform onBoarding

The Request shall include the following header information:

Name	Type	Description
Authorization	String	"Bearer "+ the TAN
Content-Type	String	application/json
X-Agrirouter-ApplicationId	String	[Application Id]
X-Agrirouter-Signature	[Signature]	see 8.8 Signing requests

8.9.2.2. Body Information

The request body includes the same parameters as the onboarding requests body:

The request body is a JSON object including the following Parameters and is equal for any type of onboarding:

Position	Name	Type	Description
1	id	String	The unique ID of the endpoint; we advice to create a URN
2	applicationId	String	The application ID for the application, provided in the agrirouter developer UI
3	certificationId	String	The ID of the certification software version provided in the agrirouter developer UI
4	gatewayId	String	The desired communication protocol after onboarding 2: MQTT 3: REST Example: "2"
5	certificationType	String	Type of the desired certificate; Possible values: PEM,P12
6	UTCTimestamp	String	A Timestamp like this: 2018-06-20T07:29:23.457Z
7	timeZone	String	A TimeZone like this: "+03:00"

 DKE DATA	Title: agrirouter Connectivity-Platform Integration Guide Part 2 Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

Example:

```
{
  "id": "urn:myapp:snr00003234",
  "applicationId": "e0eb00ff-e2ef-4429-85f5-2559aceedd6d",
  "certificationVersionId": "e0eb00ff-e2ef-4429-85f5-2559aceedd6d",
  "gatewayId": "3",
  "UTCTimestamp": "2018-06-04T12:00:03.000Z",
  "timeZone": "+02:00"
}
```

8.9.3. Verification Result

The result is a HTTP response code with a JSON object in the Body

8.9.3.1. Result codes

There are different result HTTP Status codes indicating the result

Code	Description
200	The validation was successful
400	There was an error in the request
401	The request was unauthorized; the provided registration code was unknown

8.9.3.2. Body Information

8.9.3.2.1. Success

For a successful result, the body will include a JSON object like this:

```
{
  "accountId": "4823443c-fd0d-44a7-81a6-06104455945a"
}
```

It includes the accountId, so that an app provider can check, if this accountId might already be known. For apps, that can be onboarded only once (like an FMIS, where it doesn't make any sense to have 2 of the same kind), this would mean, that onboarding is not needed.

8.9.3.2.2. Failure

In case of Failure, an error message is provided.

	Title: agrirouter Connectivity-Platform Integration Guide Part 2	
	Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

8.10. Sending Onboarding Request

To onboard a new endpoint, the endpoint has to send an onboarding request providing the registration code to agrirouter. The request is a HTTP POST request.

Remark: There is no MQTT onboarding mechanism, so onboarding always has to be done using REST.

8.10.1. Request information for signed onboarding

Remark

This is the onboarding request for Farming Software and Telemetry platforms, not for CUs.

8.10.1.1. General

The address for the onboarding request is as follows:

Method	Address
POST	api/v1.0/registration/onboard/request

8.10.1.2. Header Information

8.10.1.2.1. For FarmingSoftware and Telemetry Platform onBoarding

The Request shall include the following header information:

Name	Type	Description
Authorization	String	"Bearer " + the registration code <i>Remark: There is a space between bearer and registration code!</i>
Content-Type	String	application/json
X-Agrirouter-ApplicationId	String	[Application Id]
X-Agrirouter-Signature	[Signature]	see 8.8 8.8

8.10.1.3. Body Information

The request body is a JSON object including the following Parameters and is equal for any type of onboarding:

Position	Name	Type	Description
1	id	String	The unique ID of the endpoint; we advice to create a URN
2	applicationId	String	The application ID for the application, provided in the agrirouter developer UI
3	certificationVersionId	String	The ID of the certification software version provided in the agrirouter developer UI
4	gatewayId	String	The desired communication protocol after onboarding 2: MQTT 3: REST Example: "2"
5	certificateType	String	Type of the desired certificate; Possible values: PEM,P12
6	UTCTimestamp	String	A Timestamp like this: 2018-06-20T07:29:23.457Z
7	timeZone	String	A TimeZone like this: "+03:00"

	Title: agrirouter Connectivity-Platform Integration Guide Part 2 Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

Example:

```
{
  "id": "mydeviceid",
  "applicationId": "e0eb00ff-e2ef-4429-85f5-2559aceeedd6d",
  "certificationVersionId": "e0eb00ff-e2ef-4429-85f5-2559aceeedd6d",
  "gatewayId": "3",
  "UTCTimestamp": "2018-06-04T12:00:03.000Z",
  "timeZone": "+02:00"
}
```

	<p>Title: agrirouter Connectivity-Platform Integration Guide Part 2</p> <p>Author: DKE-Data GmbH & Co. KG</p>	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

8.10.2. Request information for CU onboarding

Remark

This is the onboarding request for CUs.

8.10.2.1. General

The address for the onboarding request is as follows:

Method	Address
POST	api/v1.0/registration/onboard

8.10.2.2. Header Information

The Request shall include the following header information:

Name	Type	Description
Authorization	String	"Bearer "+ the TAN
Content-Type	String	application/json

8.10.2.3. Body Information

The request body is a JSON object including the following Parameters and is equal for any type of onboarding:

Position	Name	Type	Description
1	id	String	The unique ID of the endpoint; we advice to create a URN
2	applicationId	String	The application ID for the application, provided in the agrirouter developer UI
3	certificationId	String	The ID of the certification software version provided in the agrirouter developer UI
4	gatewayId	String	The desired communication protocol after onboarding 2: MQTT 3: REST
5	certificationType	String	Type of the desired certificate; Possible values: PEM,P12

Example:

```
{
  "id": "mydeviceid",
  "applicationId": "e0eb00ff-e2ef-4429-85f5-2559aceedd6d",
  "certificationVersionId": "e0eb00ff-e2ef-4429-85f5-2559aceedd6d",
  "gatewayId": "3",
}
```

8.10.3. Response

8.10.3.1. Response code

The request has several possible response codes indicating Success or Failure:

 DKE DATA	Title: agrirouter Connectivity-Platform Integration Guide Part 2 Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

Code	Possible problem
201	Success; Analyze onboarding result to get started
400	The Request was invalid
401	Unauthorized; meaning, that one of the given header parameters is wrong. Refer to the error message

8.10.3.2. Success

On success, the HTTP response code will be 201.

The result is a json object including the information required for onboarding.

Position	Name	Type	Description
1	authentication	Object	Includes all Authentication information
1.1	certificate	String	The certificate required for communication; Public AND Private Key
1.2	secret	String	The passkey for the certificate
1.3	type	String	Type of Certificate; PEM or PK12
2	capabilityAlternateId	String	A value that just has to be saved and sent in several scenarios
3	connectionCriteria	Object	Includes all information required for further communication
3.1	gatewayId	String	Assigned gateway; 2= MQTT, 3=REST
3.2	host	String	Only for MQTT: The broker address
3.3	port	String	Only for MQTT: The broker port
3.4	measures	String	Endpoint URL of the inbox or Topic, when using MQTT
3.5	commands	String	Endpoint URL of the outbox or Topic, when using MQTT
3.6	client	String	MQTT only: The ClientID of the endpoint
4	deviceAlternateId	String	The device ID used to mark the source of a message from this device
5	sensorAlternateId	String	The deviceID used to mark the source of the communication from this device

Example:

```
{
  "authentication": {
    "certificate": "-----BEGIN ENCRYPTED PRIVATE KEY-----\n... \n-----END ENCRYPTED PRIVATE KEY-----\n-----BEGIN CERTIFICATE-----\n... \n-----END CERTIFICATE-----\n",
    "secret": "77R8cjOGi9yTCBt2",
    "type": "PEM"
  },
  "capabilityAlternateId": "7bc8ab05-a0de-40db-a259-7defb1265e9",
  "connectionCriteria": {
    "gatewayId": "3",
    "measures": "https://dke-qa.eu1.cp.iot.sap/iot/gateway/rest/measures/c067272a-d3a7-4dcf-ab58-5c45ba66ad60",
    "commands": "https://dke-qa.eu1.cp.iot.sap/iot/gateway/rest/commands/c067272a-d3a7-4dcf-ab58-5c45ba66ad60"
  },
  "deviceAlternateId": "c067272a-d3a7-4dcf-ab58-5c45ba66ad60",
  "sensorAlternateId": "5564ce96-385f-448a-9502-9ea3c940a259",
}
```

 DKE DATA	Title: agrirouter Connectivity-Platform Integration Guide Part 2	
	Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

}

Remark

Save all those information, you'll need them for communication with the agrirouter.

8.10.3.3. Failure

On Failure, a JSON object including an error message is received, e.g.:

```
{
  "error": {
    "code": "0110",
    "message": "Signing header is invalid, the request has timedout, or UTCTimestamp is not provided",
    "target": "",
    "details": []
  }
}
```

Possible Error codes

Error Code	Error Text	Comment
0010	The account is inactive	
0011	Unknown account.	
0020	The account is not Approved for use with this application	
0021	The provided application certification is unknown	
0022	The provided application certification is not in the proper status.	
0023	The endpoint was previously onboarded and is blocked for use.	
0024	The provided application certification is not accepted for this request.	
0100	Invalid payload.	
0101	The certification Id provided is not valid for this request.	
0102	The gateway Id provided is not valid	
0103	The certificated type provided is not valid	
0104	The gateway id provided is not valid for this device. The gateway Id can not be changed when reonboarding an existent device in account	
0105	The application id provided is not valid.	
0105	You've made too many requests in a short period of time.	
0106	Missing \$(constants.Validation.SIGNATURE.APPLICATION_ID_HEADER_FI ELD} or \$	
0107	Invalid signature	
0108	Application cannot be validated as application header is not the same as the application associated with the given registration code.	
0109	Application header is not the same as the application associated with the given endpoints or multiple applications found for the given endpoints.	

	Title: agrirouter Connectivity-Platform Integration Guide Part 2	
	Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

8.10.3.3.1. Handling Signature Issues

If you experience problems with an invalid signature, (Code 107), try the following:

- make sure, you encoded the whole body
- compare the signature with the result of the node server tool delivered with the postman collection
- check the timestamp. If your local time is ahead of or too far behind agrirouter servers time, it will not recognize the signature as valid. The agrirouter HTTP answer includes a timestamp reporting agrirouters server time.

8.11. Reonboarding

8.11.1. What is re-onboarding?

Re-onboarding allows an endpoint to get new client certificates and updates on URLs / topics for measures and commands Internally, the endpoint gets the same identification (deviceAlternateId, sensorAlternateId and capabilityAlternateId remain the same), thus all rules, filters and buffered messages remain valid and do not get lost

8.11.2. Why should an endpoint be re-onboarded?

If an app instance loses parts of its onboarding information, it can be reonboarded using the onboarding process. Additionally, the endpoint URLs might change. Even though, this is currently not planned, it could be a result of scaling.

8.11.3. How does re-onboarding work?

The client needs to send the same onboarding request as for 1st first onboarding, including a new registration code provided by the end-user.

Remark

The external Id **must be the same** as for the initial onboarding, as this is used to map the endpoint to the existing

Capabilities may be sent, but do not have to (these are stored within the agrirouter from initial onboarding)

Reonboarding has to be done, when an app is updated in a way, that its certification version id changes.

 DKE DATA	Title: agrirouter Connectivity-Platform Integration Guide Part 2	
	Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

9. Revoking Accounts

If the contract with a customer ends, app providers should revoke their connection with the customers account. This can be done using a similar process

9.1. Process of Revoking

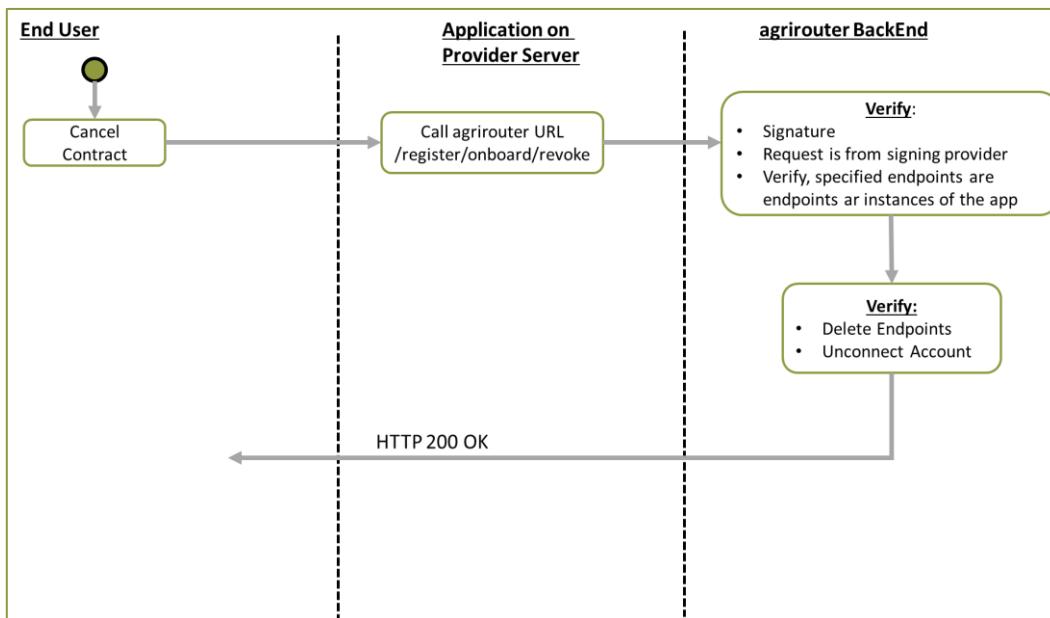


Figure 20 Revoke Process

9.1.1. Creating Revoke Command

9.1.1.1. General

Deleting a CU is done through a DELETE-Request to the following endpoint:

Method	Address
DELETE	/registration/onboard/revoke

Remark

Deleting virtual CUs is done using a command. See 16.5.2 Removing a Virtual Cu

9.1.2. Header

The Request shall include the following header information:

Name	Type	Description
Content-Type	String	application/json
X-Agrirouter-ApplicationId	String	[Application Id]
X-Agrirouter-Signature	[Signature]	see 8.8 Signing requests

 DKE DATA	Title: agrirouter Connectivity-Platform Integration Guide Part 2 Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2		Confidence: external
Status: final	Version: 1.3	September 4, 2018

9.1.3. Body

The body includes a JSON object:

Position	Name	Type	Description
1	accountId	String	The ID of the users account
2	endpointId	Array of String	An Array of endpointIDs, that shall be deleted.
3	UTCTimestamp	String	A timestamp like this: 2018-06-20T07:29:23.457Z
4	timeZone	String	A time zone like this: "+03:00"

Example:

```
{
  "accountId": "4823443c-fd0d-44a7-81a6-06104455945a",
  "endpointIds": [
    "3823443c-ed0d-44a7-81a6-06104455945a",
    "7873443c-fd0d-44a7-81a6-06104455945a"
  ],
  "UTCTimestamp": "2018-06-25T18:08:17.661890Z",
  "timeZone": "+00:00",
}
```

9.1.4. Response

The response is an empty message using a HTTP Return code to indicate the result:

Code	Description
204	Revoke was successful
400	A validation failure occurred
401	The Request was unauthorized

 DKE DATA	Title: agrirouter Connectivity-Platform Integration Guide Part 2	
	Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2		Confidence: external
Status: final	Version: 1.3	September 4, 2018

10. Architecture of an endpoint

10.1. General Overview

The endpoint can be seen as the interface between agrirouter and an App Instance. It receives messages from the app instance and can deliver messages to this app instance.

On the other side, it is part of the agrirouter and forwards messages to routing and addressing, receives addressed messages and can handle the subscriptions.

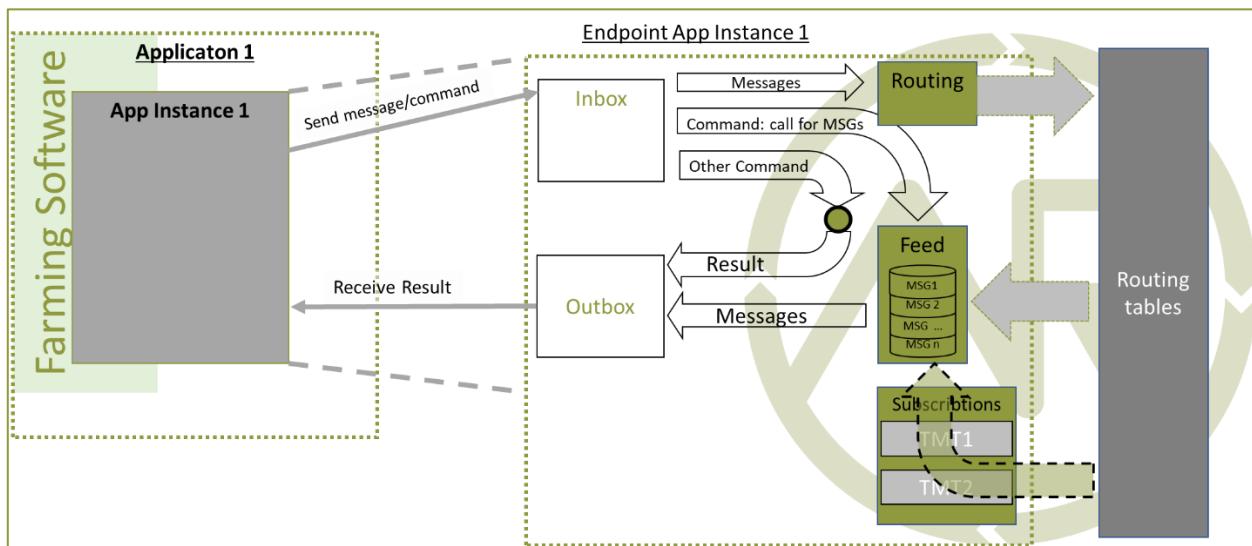


Figure 21 Architecture of an endpoint

Remark

The terms “subscription” and “subscribing” are well known by MQTT addicted developers. In this document however, subscriptions describe a technique of the agrirouter endpoint. It works like an MQTT subscription, it’s however important to notice, that MQTT is mostly not the context of these words in the document except for the relevant chapter 8.6.2!

 DKE DATA	Title: agrirouter Connectivity-Platform Integration Guide Part 2	
	Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

10.2. Inbox

The inbox is used to receive messages or commands from an app instance.
The address of this inbox is delivered with the Response of the onboarding request as “measures”.

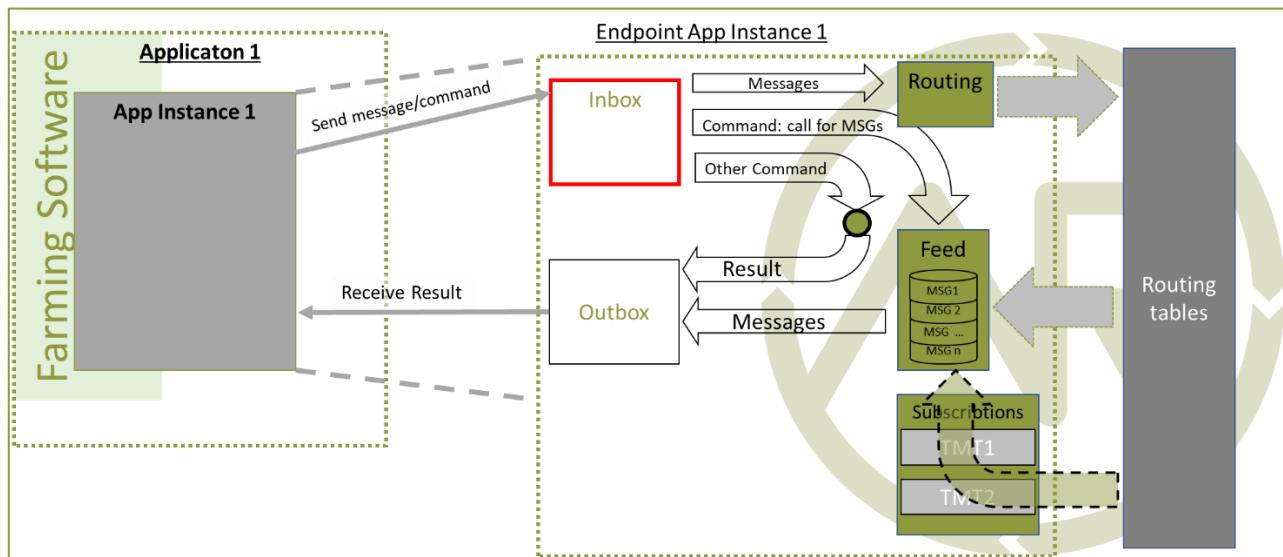


Figure 22 Inbox within an agrirouter endpoint

Remark

In terms of agrirouter onboarding result, the inbox is called measures due to the used standardized gateway. In this integration guide, we call it inbox as this is, what it really is in terms of agrirouter

 DKE DATA	Title: agrirouter Connectivity-Platform Integration Guide Part 2	
	Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

10.3. Outbox

The outbox is used to deliver messages to an app instance.

The address of this outbox is delivered with the Response of the onboarding request as “commands”.

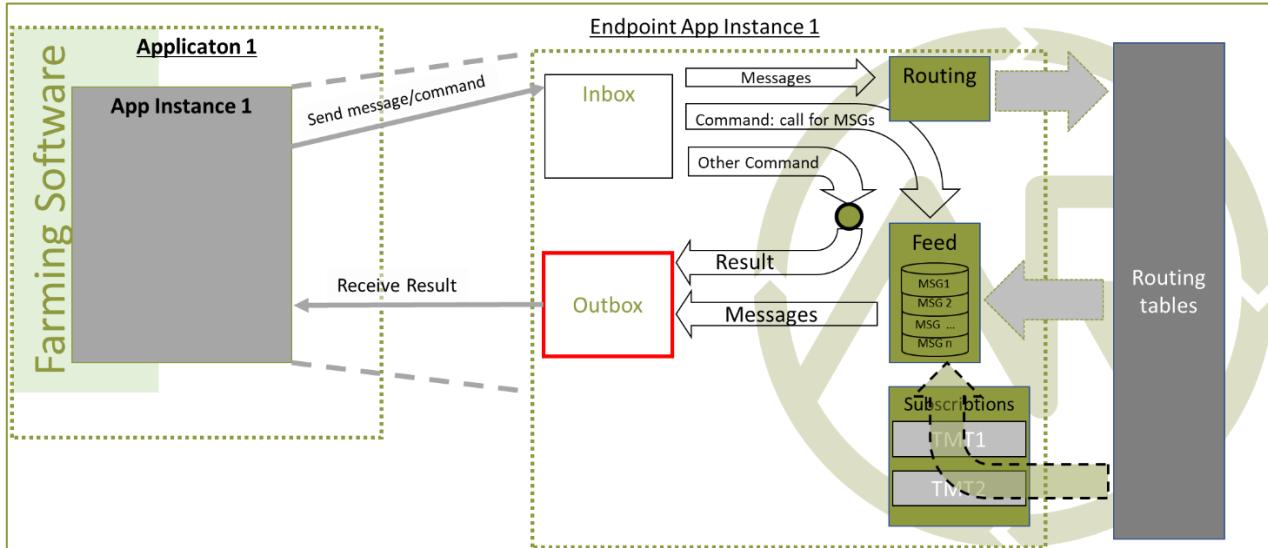


Figure 23 Outbox within an agrirouter endpoint

Remark

In terms of agrirouter onboarding result, the outbox is called commands due to the used standardized gateway. In this integration guide, we call it outbox as this is, what it really is in terms of agrirouter

 DKE DATA	Title: agrirouter Connectivity-Platform Integration Guide Part 2 Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

10.4. Feed

The feed receives all messages from the agrirouter due to addressing of messages and subscriptions of the endpoint, given there is a routing for that. Messages in the feed are stored for a maximum of 4 weeks, before they are deleted. The user will get informed about that, if there are messages, that were not delivered for 3 weeks or more. An app instance can either request the messages to be forwarded to its outbox or request to delete the messages. If it requests the forwarding to the outbox, the messages are only deleted from the feed, when the app instance confirms having received those messages.

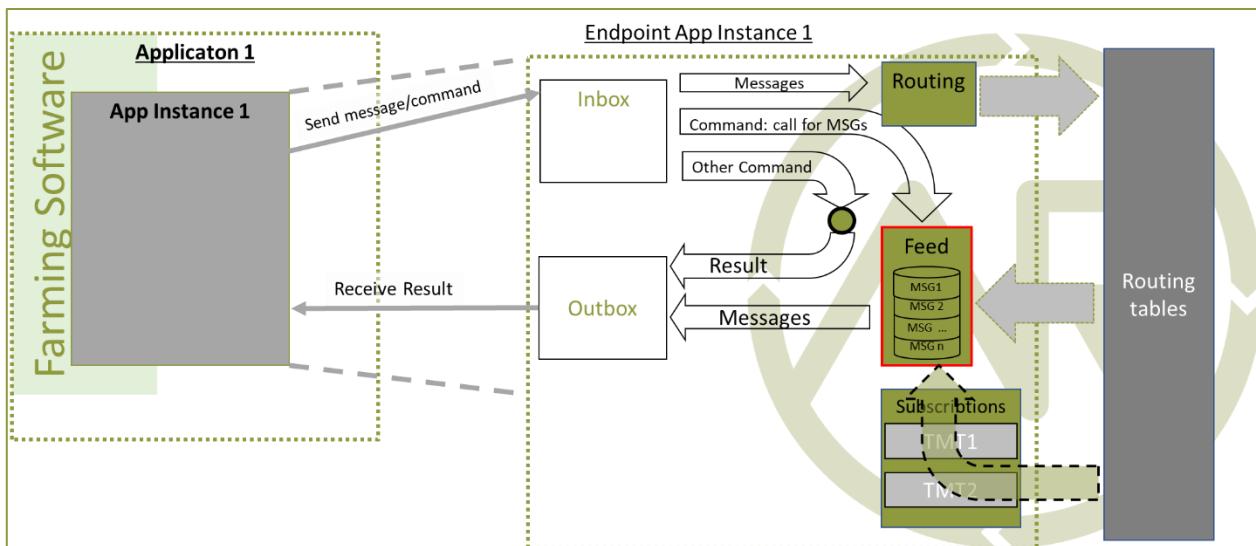


Figure 24 The feed within an agrirouter endpoint

10.5. Subscription List

The subscription list includes a list of all Message Types and DDIs, the endpoint is subscribed for. Whenever a message is published on the agrirouter account, this will lead to putting a copy of this message into the feed, given, there is a route for that.

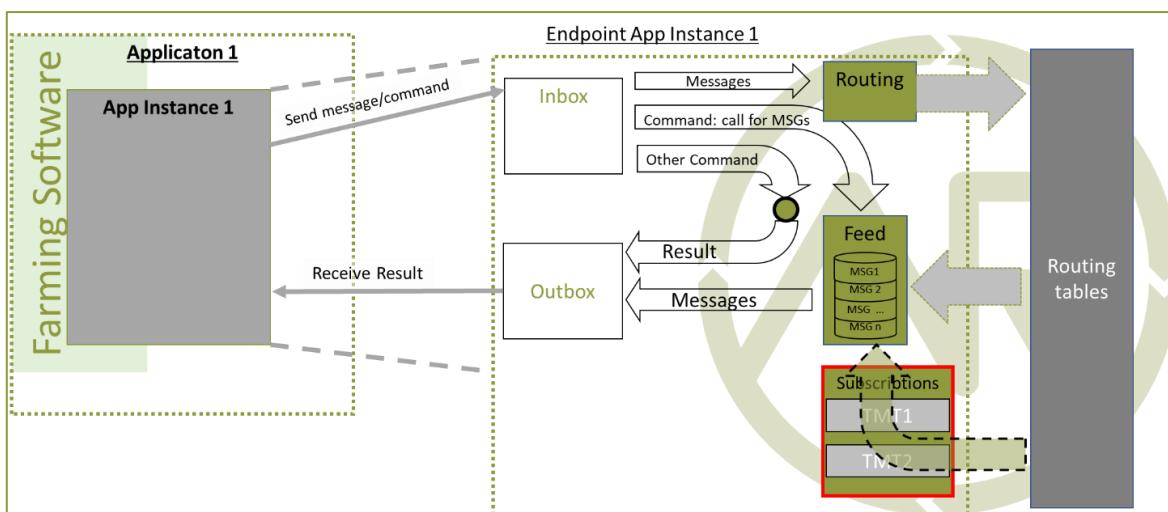


Figure 25 Subscription List within an agrirouter endpoint

 DKE DATA	Title: agrirouter Connectivity-Platform Integration Guide Part 2	
	Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

11. Basics of communication

11.1. General communication convention

Communication with the agrirouter includes communication with inbox AND outbox, unimportant, if an app only sends messages or only consumes messages. Every message or command is sent to the inbox, results are always delivered through the outbox.

11.2. Prequisites for communication

Every communication is introduced by the App Instance. The agrirouter endpoint will never contact the app instance.

After application registration, the communication is secured using the SSL certificate retrieved during the onboarding process. The agrirouter provides PK12 certificates as well as PEM certificates which can be set within the SSL context of the request to ensure authenticated communication with the agrirouter.

Once after the validation has happened, the agrirouter provides all necessary security informations to establish a messaging connection. The developer is obliged to store this certificate, or the tokens secured and encrypted within the application. Receiving the certificate is described in 8 Onboarding .

11.3. Different layers, “The onion principle”

agrirouter is a platform, that is mostly used to transport messages **through** it and not towards it.

Therefore:

the content is encapsulated in

messages and commands, which are encapsulated into

the request to and the result from the agrirouter, which are encapsulated into

the protocol layer of REST using Request and Response or MQTT using Publish and Subscribe.

With except to the EFDI telemetry messages (DeviceDescription and live data), agrirouter doesn't analyse the messages inside the agrirouter request. It just checks the technical message type and the addressing to determine the recipients based on the routings and subscriptions.

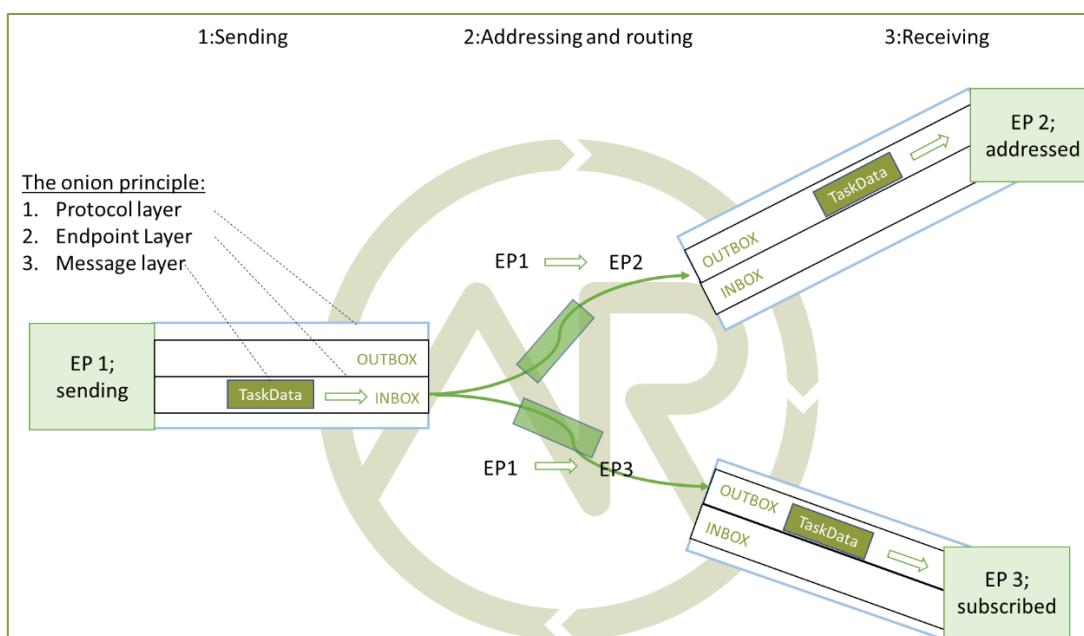


Figure 26 The onion principle for a non-telemetry message

 DKE DATA	Title: agrirouter Connectivity-Platform Integration Guide Part 2	
	Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

An exception is the EFDI telemetry messages.

The device descriptions are needed by the agrirouter to

- determine the relevant CU when sending a Message directly to a machine
- filter for DDIIs that are allowed to be sent to specific endpoints

The life telemetry data is analysed, so that a filtering for value categories like fuel consumption is possible.

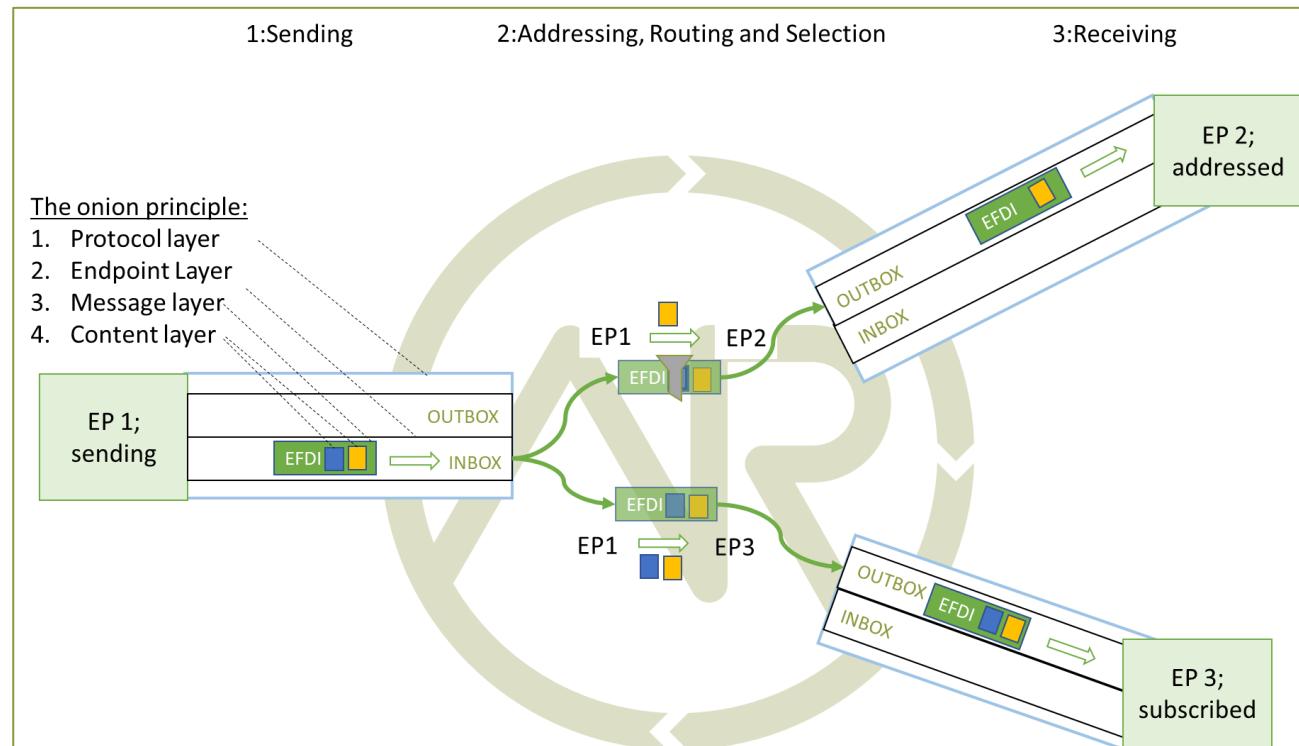


Figure 27 The onion principle for a telemetry message

11.4. The protocol layer

The protocol layer describes the basic communication layer. All agrirouter communication is based on a TCP-IP communication. When onboarding an endpoint, the developer can choose between REST and MQTT.

11.4.1. REST

REST is a wellknown principle for online apis. For more information on REST, please see following resources:

- https://en.wikipedia.org/wiki/Representational_state_transfer
- <https://code.tutsplus.com/tutorials/a-beginners-guide-to-http-and-rest--net-16340>

REST uses HTTP requests, that result in an HTTP Response.

To avoid request timeouts, agrirouter will return an HTTP Status Code 201(Processing) and the app instance will have to poll for a request confirmation.

Remark:

REST is based on single, request only methods, therefore, the agrirouter cannot contact any endpoint. This means, that the endpoint for example has to poll for new messages in the outbox.

 DKE DATA	Title: agrirouter Connectivity-Platform Integration Guide Part 2	
	Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

11.4.1.1. Communication Workflow with the Inbox

Using REST, an app instance just receives a HTTP 200 “OK” Response. The App Instance sends a HTTP Post request over an SSL secured Connection.

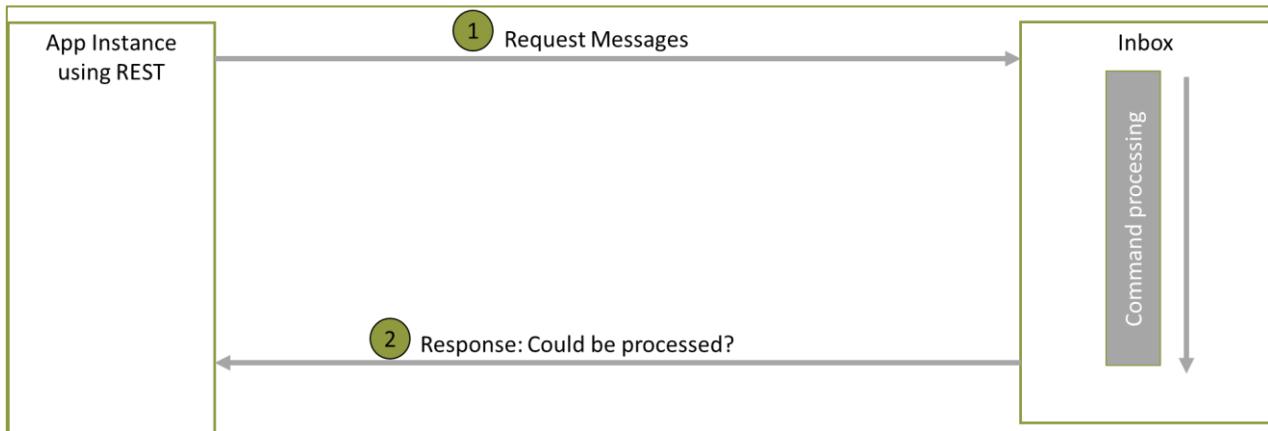


Figure 28 Request and Response in HTTP

11.4.1.2. Communication with the Outbox

Using REST, the communication with the Outbox requires polling:

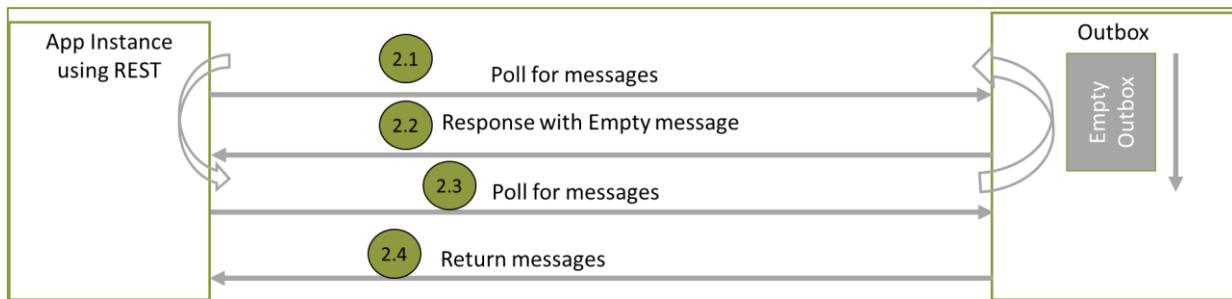


Figure 29 REST Communication with the outbox

 DKE DATA	Title: agrirouter Connectivity-Platform Integration Guide Part 2 Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

11.4.2. MQTT

MQTT is a subscription based protocol. Therefore, the client can be informed by agrirouter, that new messages are available. For further information, please refer to one of the following resources:

- <https://mqtt.org/>

The agrirouter provides an MQTT Broker serverside, so, an app instance has to connect to this server with its client. agrirouter provides one MQTT Server per Endpoint, so there is no danger or chance to subscribe for messages of another endpoint

11.4.2.1. Communication with the Inbox

Using MQTT, the app instance will publish the request and after a while, the agrirouter will publish the response. Polling is not required.



Figure 30 Request and Response using MQTT

11.4.2.2. Communication with the Outbox

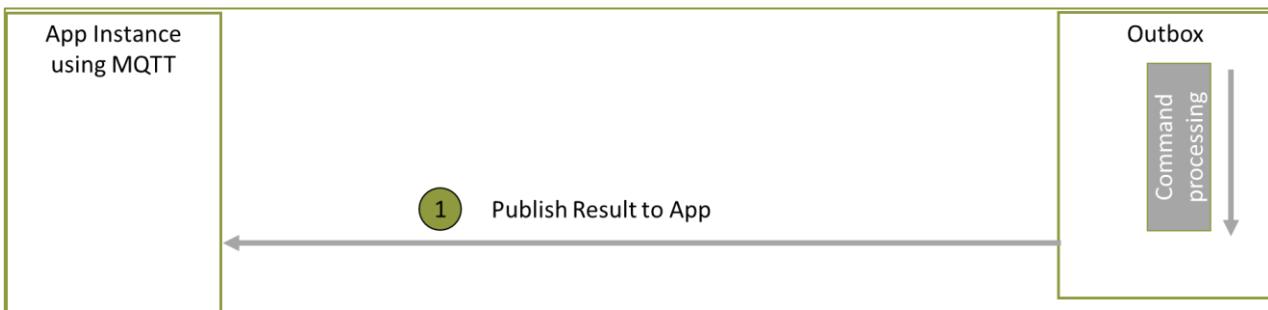


Figure 31 Receiving Result from the outbox in MQTT

If there are messages available in the outbox, agrirouter will simply publish them to the MQTT App Instance.

 DKE DATA	Title: agrirouter Connectivity-Platform Integration Guide Part 2 Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

11.4.3. Comparison of protocols

Both protocols have several advantages and disadvantages. To select the right protocol for your needs, check the following table

Topic	MQTT	REST
Can be used for onboarding		X
Can send JSON	X	X
Can send raw protobuf		X
Needs no polling	X	
Steps for Call and Result	3	min. 4; polling

11.5. Abstraction of communication workflows

To avoid graphs with too many arrows, we simplify the upcoming requests, abstracting MQTT and REST. Whatever protocol you use, a Request and Response in this document will look like this:

11.5.1.1. Abstraction of Calls to the Inbox

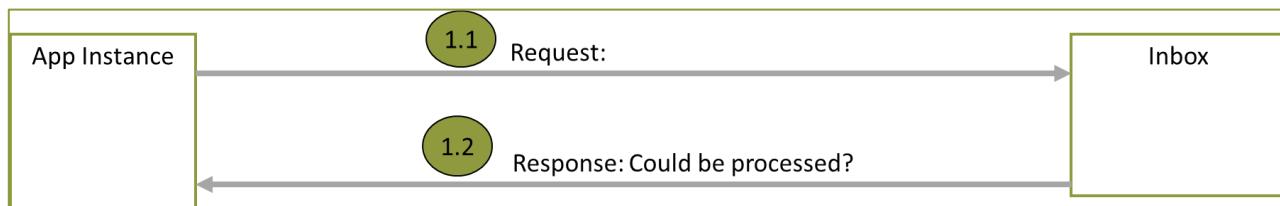


Figure 32 Abstraction of a Call or Message Sending to the Inbox

11.5.1.2. Abstraction of Results from Outbox

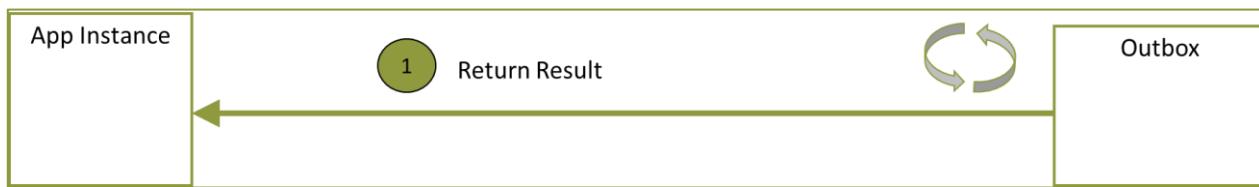


Figure 33 Abstraction of a result from the outbox

	Title: agrirouter Connectivity-Platform Integration Guide Part 2 Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

12. Communication of App Instance and Endpoint

12.1. Endpoint Adresses

The endpoint addresses of the inbox and outbox are delivered with the onboarding request. Please refere to 8.6.3

 DKE DATA	Title: agrirouter Connectivity-Platform Integration Guide Part 2	
Author: DKE-Data GmbH & Co. KG		
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

Sending Onboarding Request

12.2. Assigning a result to a request

The assignment between messages sent to the Inbox and their corresponding message in the outbox is done by comparing the application_message_id provided by the app.

Exception:

If a message is not correctly encoded, so that the agrirouter cannot decode it, there will be no application_message_id in the result.

12.3. Flow for sending messages

Sending messages to the agrirouter creates an ACK-Message in the outbox of the agrirouter

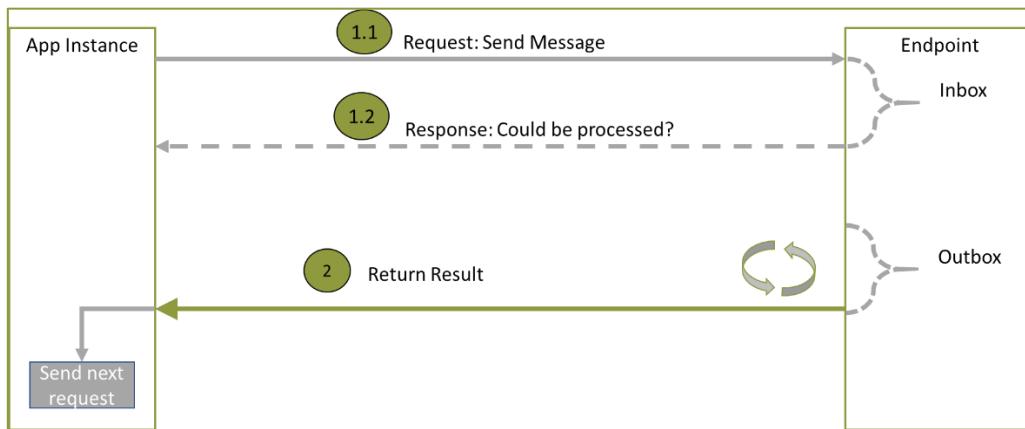


Figure 34 Send Message or Command to agrirouter

 DKE DATA	Title: agrirouter Connectivity-Platform Integration Guide Part 2	
	Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

12.4. Flow of commands

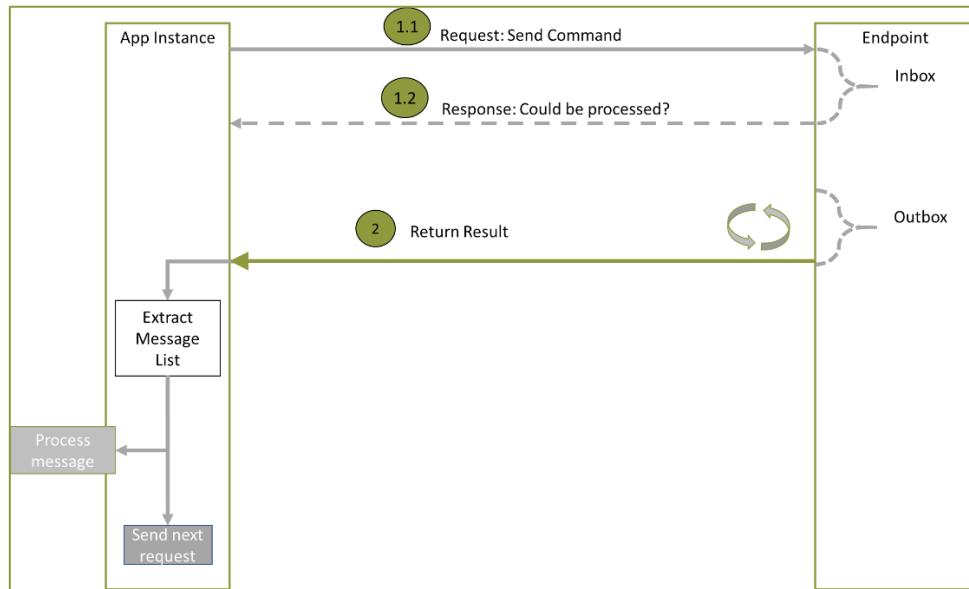


Figure 35 Command Flow

12.5. Flow of requesting messages from the feed

If the command is a feed command requesting messages from the feed, the app instance has to confirm the receipt of the message, so that it is deleted from the feed.

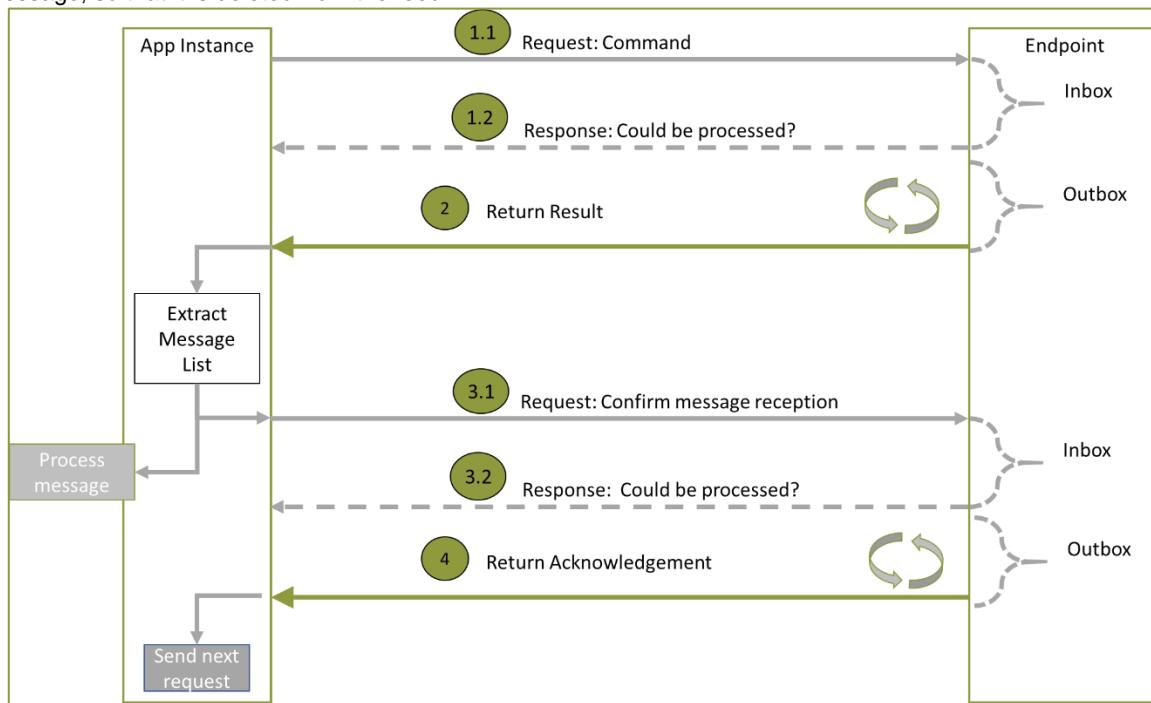


Figure 36 Command flow for reading the feed

	Title: agrirouter Connectivity-Platform Integration Guide Part 2 Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2		Confidence: external
Status: final	Version: 1.3	September 4, 2018

12.6. Terms

A command-process consists of a call (1.1) to the inbox. This call consists of the request and a processing status response. Agrirouter will return a result through the outbox. When the app instance receives this result successfully, it has to call for confirmation at the inbox to clear the outbox. If a message list was delivered, this will also delete the received messages from the feed.

 DKE DATA	Title: agrirouter Connectivity-Platform Integration Guide Part 2	
	Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

13. Building Messages and Commands

13.1. General

Creating a message includes several steps. This can be seen in the graph below:

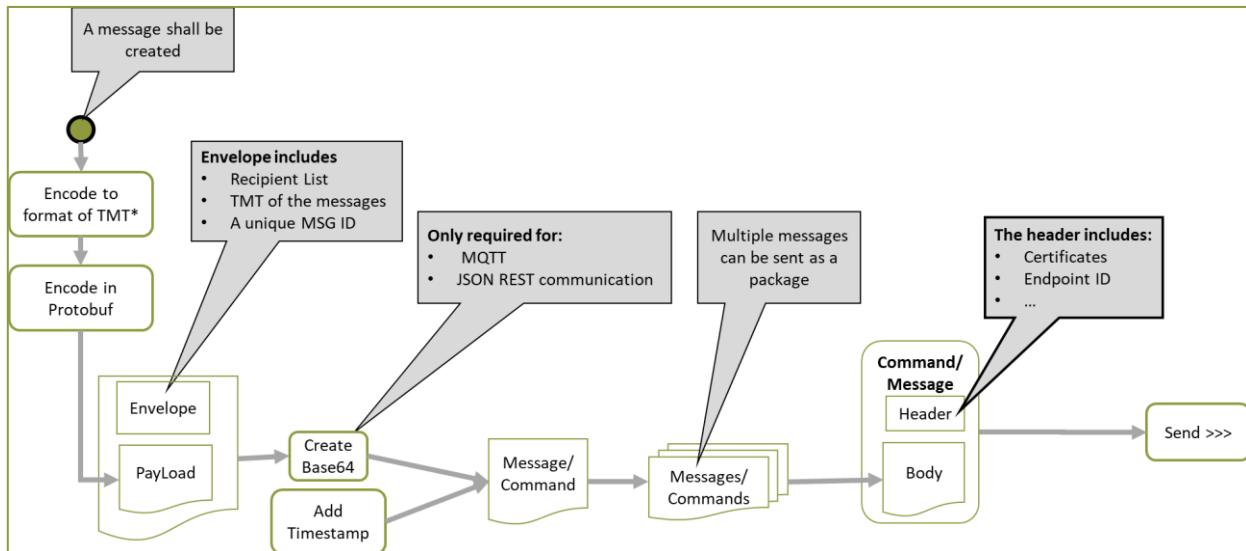


Figure 37 Workflow of creating a message

13.2. Chunking big messages

Exchanging data using a mobile connection can be very critical, as a stable internet connection is not always given. To avoid wasting data volume and time, agrirouter offers a possibility to cut messages into multiple smaller parts and send them one by another. If the internet connection breaks down inbetween, only one small package is not delivered and has to be send again.

Remark

agrirouter messages can have a maximum size of 1 MB each. If a message is too large, agrirouter will respond with a HTTP code 413 "Request too large". Please note, that the relevant size value is the size of the protobuf format, which might increase with conversion to Base64.

Remark

EDFI telemetry messages cannot be chunked, as they are analyzed for the routing of telemetry data on agrirouter.

 DKE DATA	Title: agrirouter Connectivity-Platform Integration Guide Part 2	
	Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

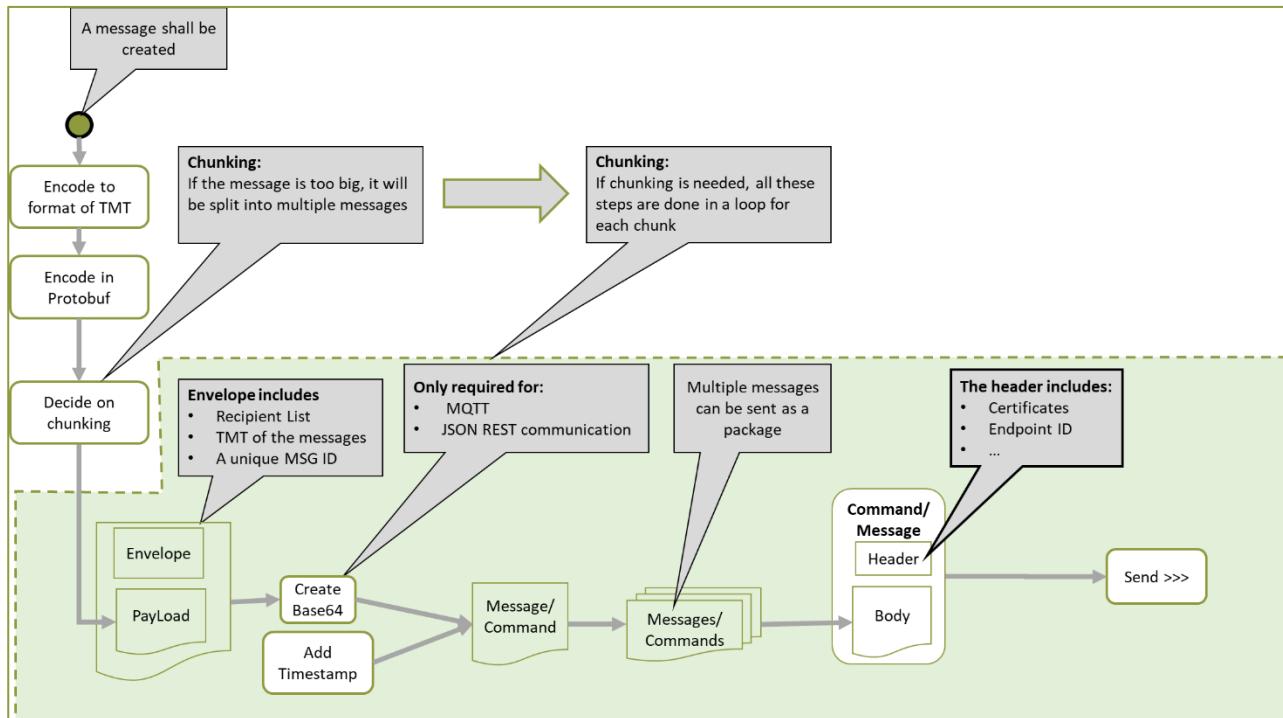


Figure 38 Workflow of creating a chunked message

Remark:

Sending multiple messages in one Request is only possible, if the overall size of all messages payloads is beyond the chunking maximum size value.

The overall body of the message to the inbox shall not include more payload than allowed without chunking. So, when a message is too big for one request and needs to be chunked, these chunks have to be sent in multiple messages, they can not be added as multiple chunks to one message.

13.3. Encode content to its Technical Message Type

Encoding a message in its required technical message type depends on the message type. The message types for agrirouter Commands are described further in 16 Overview of agrirouter commands.

13.4. Encode content in Protobuf

agrirouter messages and commands use protobuf 3 to encode content. Please refer to <https://developers.google.com/protocol-buffers/> for a full documentation of the format and a lot of development resources.

The agrirouter uses the node.js lib: <https://github.com/dcodeIO/protobuf.js>

Remark

As the payload of a message is defined as an any message, the TypeURL has to be set to the corresponding message type. Otherwise, agrirouter will not be able to forward or process the message.

13.4.1. Getting the agrirouter protobuf files

Protobuf definitions are part of the integration guides attachments, they are delivered as .proto files. They can also be found at the github: <https://github.com/DKE-Data/agrirouter-api-protobuf-definitions>

	Title: agrirouter Connectivity-Platform Integration Guide Part 2 Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

The repository delivers all protobuf files, that are owned by the DKE. It might be, that you need additional protocol buffers like the EFDI telemetry messages definitions, that cannot be delivered in this github account due to intellectual property rights. You will however find instructions, how to get those protobuf definition files.

	<p>Title: agrirouter Connectivity-Platform Integration Guide Part 2</p> <p>Author: DKE-Data GmbH & Co. KG</p>	
Identification: agrirouter Integration Guide Part 2		Confidence: external
Status: final	Version: 1.3	September 4, 2018

13.4.2. Reading Protobuf definitions

Basically, you can think of a protobuf as a C Struct or a JSON object. It includes data of several data types with the possibility of default values or optional parameters and lists.

An example of a protobuf with several descriptions

```
message MyProtobuf{ //The main class of this element
    enum Direction { // An enumeration of values
        SEND = 0;
        RECEIVE = 1;
        SEND_RECEIVE = 2;
    }
    message subBuffer{
        int64 x=1;
        int64 y=2;
    }
    string name=1; // Field 1 of the structure is a string called "name"
    int64 age=2;
    Direction direction=3; //A field of type direction, that describes an enum
    repeated string hobbies=4; //A list with 0 to n elements possible
    repeated subBuffer positions=5;
}
```

13.4.3. Including Protobuf in your project

Protobuf is available for multiple programming languages such as Java, C++, Java Script, Python, etc. etc.
The protobuf compiler creates sourcecodes for your desired language. Please refer to <https://github.com/google/protobuf> for a list of implementations

 DKE DATA	Title: agrirouter Connectivity-Platform Integration Guide Part 2	
	Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

13.5. Chunking Messages

Important note:

The segmentation of message does not apply to the telemetry related data as it is described in EFDI telemetry messages. A maximum number of chunks (max. size for binaries) will be defined in the project

The max chunk size is capped at 1MB. This size might be decreased during the project for performance reasons.

13.5.1. Chunking parameters

The protobuf ChunkComponent can be found in commons/chunk.proto.

Name	Type	Description
context_id	String	A unique ID for this chunk. The number shall be equal for each part of the chunk and help the receiving endpoint to identify the chunk
current	int64	The current index of this chunk within the whole chunk starting with 1
total	int64	The total number of chunks, this message consists of
total_size	int64	The total size of the whole chunk in bytes.

13.5.2. Preparing and creating chunked messages

If it is recognized, that a message needs to be split into multiple chunks, starting from here, the single message is sent to agrirouter by splitting the message body and creating multiple requests to the agrirouter, each including a new chunk element.

Remark

agrirouter does neither check nor inform about Chunks, that have not yet been delivered to agrirouter. It will forward the single parts and the receiving endpoint(s) will have to take care of realigning the parts.

 DKE DATA	Title: agrirouter Connectivity-Platform Integration Guide Part 2	
	Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

13.6. Building the Envelope

The envelope is a protobuf structure of type agrirouter.request.RequestEnvelope.

The parameters as overview:

Parameter	Type	Description
application_message_id	String	A unique ID for this message. UUID required
application_message_seq_no	int64	An indicator, in which order the client sent the message. The smallest Sequence number must be >0.
technical_message_type	string	The TMT; see 17 Technical Message
team_set_context_id	string	The relevant teamset for this message; just in case, it changes
mode	mode	DIRECT, PUBLISH or PUBLISH_WITH_DIRECT
recipients	string(repeated)	A list of endpoint IDs to forward the message to
chunk_info	ChunkComponent	The chunking information for split messages
timestamp	timestamp	The timestamp, when the message was created

For the timestamp format definition, please refer to:

<https://github.com/google/protobuf/blob/master/src/google/protobuf/timestamp.proto>

Remark:

The application_message_sequence_no shall not be 0, as this might lead to misbehavior in any C++ Implementation of the agrirouter interface. To be consistent with every endpoint, it shall also not be done in other languages, even though they do not have a problem with that on their side of the agrirouter.

13.7. Building the payload

The structure of the payload depends on the technical message type. Its always some kind of protobuf structure, please refer to the chapters on technical message types for further information.

13.8. Connecting envelope and PayLoad

Envelope and content are packaged into one container by using the technique of "Delimited Messages". Please note, that this is **not** simply copying both memory buffers into one buffer. Please refer to <https://developers.google.com/protocol-buffers/docs/techniques#streaming>

Remark

Note that this concept is not supported in all protobuf libraries (in Java and node.js it is, in C++ it is not)

If building streaming is required for the language and libraries, you use, Note, that Delimited messages are attached to each other like this: Length1,Content1,Length2,Content2,... The variable size of Length is the length of a varint; see <https://developers.google.com/protocol-buffers/docs/encoding#varints>.

A solution for C++ can be found here: <https://stackoverflow.com/questions/2340730/are-there-c-equivalents-for-the-protocol-buffers-delimited-i-o-functions-in-ja/>

 DKE DATA	Title: agrirouter Connectivity-Platform Integration Guide Part 2	
	Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

13.9. Message container

The message needs to be packaged into a message container, that includes the message itself and a timestamp. Going forward from this step, the encoding can either be in protobuf or JSON. For MQTT, it has to be JSON, for REST, it can be both.

13.9.1. Encode in Base64

This step is only required, if your app instance communicates with its endpoint using MQTT or JSON based REST.

Encode the serialized binary protobuf stream into a base64 string. All further steps will be done in JSON from now on.

13.9.2. Go on in Protobuf

agrirouter REST endpoints are also capable of exchanging Protobuf.

When using Protobuf, the whole message including the upcoming steps will be encoded in Protobuf. The container is an element of type any in the message; see further steps.

13.10. Adding the Timestamp

The Timestamp and the message now have to be packaged into one JSON or Protobuf object with the timestamp of the message sending time. This timestamp shall use UTC.

Remark:

The timestamp is the time of recording the message, not the timestamp of sending it.

13.11. Add Message to List of Messages

The object can now be added to the list of messages, that shall be sent to the endpoint at once. It's important to know, that all these messages have the same recipient list.

13.11.1. JSON

The message List is a JSON array of Message containers and called measures in the following:

[message, timestamp]

13.11.2. Protobuf

The protobuf container can be found here:

It looks as follows:

```
message Measure {
    repeated google.protobuf.Any values = 1;
}
```

Remark

Please note, that this definition is part of the message definition below (when it comes to definition of the whole message).

 DKE DATA	Title: agrirouter Connectivity-Platform Integration Guide Part 2	
	Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

13.12. Build message

To have a fully compatible message, we now need to take the list of messages and add a header describing the sending endpoint.

Parameters List:

Position	Name	Type	Description
1	sensorAlternateId	String	The source of this message, e.g. the CU or a device
2	capabilitiesAlternateId	String	An internal value
3	measures	Array	An Array of messages and Timepoints
3.1	message	Base64/Protobuf	A base64 encoded message
3.2	timestamp	Timestamp	The timestamp of recording

13.12.1. JSON setup

The JSON setup looks like this:

```
{
    "sensorAlternateId": "{{sensorAlternateId}}",
    "capabilityAlternateId": "{{capabilityAlternateId}}",
    "measures": [ ["{{encoded_request}}", "{{$timestamp}}"] ]
}
```

13.12.2. Protobuf setup

The protobuf message can be found here: <https://help.sap.com/viewer/643f531cbf50462c8cc45139ba2dd051/Cloud/en-US/e97b63e35f9a4bdbab72075e7bd37ccf.html>

It looks as follows:

```
syntax = "proto3";
import "google/protobuf/any.proto";
package gateway;

option java_package = "com.sap.iotservices.common.protobuf.gateway";
option java_outer_classname = "MeasureProtos";

message MeasureRequest {
    string capabilityAlternateId = 1;
    string sensorAlternateId = 2;
    string sensorTypeAlternateId = 3;
    int64 timestamp = 4;
    repeated Measure measures = 5;

    message Measure {
        repeated google.protobuf.Any values = 1;
    }
}
```

	Title: agrirouter Connectivity-Platform Integration Guide Part 2 Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

}

}

Remark

Note, that there is a repeated Element (=Array) in a repeated element (also Array). This inner array shall always consist of 2 Elements; First is the message, second is the timestamp.

	<p>Title: agrirouter Connectivity-Platform Integration Guide Part 2</p> <p>Author: DKE-Data GmbH & Co. KG</p>	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

14. Sending a Request

The message or command is sent to the agrirouter using a protocol dependent request to its agrirouter inbox.

14.1. General Prequesits

14.1.1. Routings available

To send a message to another endpoint, there is an existing routing required. This also applies to published messages

Remark

New routings might take a few seconds to minutes to be activated.

14.1.2. Publishing only with available subscriptions

If you publish a message, it will only reach an endpoint, if there is at least one subscribed endpoint.

14.2. Using MQTT

The endpoint address of the MQTT broker is delivered as host+port in the onboarding request. The inbox is the topic delivered with the "measures" attribute of the onboarding request.

To send a message, publish it.

In general, the hosts should be as follows:

Area	Environment	URL
EU1	Quality Assurance	https://dke-qa.eu1.cp.iot.sap/
EU1	Productive	https://dke.eu1.cp.iot.sap/

14.3. Using REST

Using REST, the message is sent as a single HTTP call using HTTPS and the endpoint certificates delivered with the onboarding Request.

Messages and commands are sent to the inbox. The address of the inbox is returned with the onboarding request ("measures") and should look like this:

Method	Address
POST	/iot/gateway/rest/measures/{{deviceAlternateId}}

14.3.2. Header

As the communication works via https, the SSL Certificate information delivered with the onboarding Result is needed. In most communication libraries, the developer just has to provide

- the certificate, which is the public key,
- the SSLKey, which is the private key
- the KeyPassword for the private key

Most libraries are than able to handle the communication.

	<p>Title: agrirouter Connectivity-Platform Integration Guide Part 2</p> <p>Author: DKE-Data GmbH & Co. KG</p>	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

For further information see the curl documentation on the parameters used for a communication. For curl_easy_setopt(:

- CURLOPT_SSLKEY
- CURLOPT_KEYPASSWD
- CURLOPT_SSLCERT

The header of the HTTP Request for the REST implementation also requires

Content-Type: "application/json"

or

Content-Type: "application/x-protobuf"

14.3.3. Body

The body consists of the message package, that was created in the previous paragraphs. Receiving a Response HTTP Messages always get confirmed by the agrirouter. For the HTTP REST Gateway, the agrirouter will simply send a HTTP Status 200 OK. For MQTT, there will not be any response.

14.4. Header

The Header has a HTTP Status 200

14.5. Body

The Body of the response is empty.

Remark

The result code of this simply informs you, if the endpoint was reachable and if the inbox was able to forward the message

 DKE DATA	Title: agrirouter Connectivity-Platform Integration Guide Part 2	
	Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

15. Receiving Results

15.1. General

Results for commands and messages from calls for messages are returned to the outbox. An application instance receives them from the outbox and can than decode the data.

The address of the outbox is returned with the onboarding request ("commands") and should look like this:

Method	Address
GET	/iot/gateway/rest/commands/{{deviceAlternateId}}

Remark

Any feed query message has to be confirmed to be deleted from the feed. If you do not confirm a feed message, you will receive it over and over again, until you confirm it.

15.2. MQTT

The endpoint address of the MQTT broker is delivered as host+port in the onboarding request. The outbox is the topic delivered with the "commands" attribute of the onboarding request. The app instance has to subscribe for this topic.

The addresses can be found in chapter 14.2 Using MQTT.

15.3. REST

The REST endpoint address of the outbox is delivered with the "command" attribute of the onboarding request. An application instance has to perform a GET request to this endpoint to receive all messages from the outbox.

15.4. Format of received messages

Agrirouter returns a message, so, there can be multiple responses in one response package.

The result format depends on the setting of the Accept-Header in the GET Request to the Outbox:

Accept Header	Will Receive
application/json	JSON
application/x-protobuf	Native Protobuf

15.4.1. JSON Result

```
[
  {
    "sensorAlternateId": "49dcf152-eb2a-4b50-ad5c-1af5b737ed80",
    "capabilityAlternateId": "49dcf152-eb2a-4b50-ad5c-1af5b737ed80",
    "command": {
      "message":
        "fskjlaflasdkjlasdfkjalkefnFSDLKDFJL33221sdldgkjglfdkjldkgjDFAFKJLKJSFDSFEE
SXFRRDGDGRDGSRDDGRddrrrg354grdgIODIO35445DGDGLKKJWE3333425H1SJK=="
    }
  }
]
```

	<p>Title: agrirouter Connectivity-Platform Integration Guide Part 2</p> <p>Author: DKE-Data GmbH & Co. KG</p>	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

15.4.2. Protobuf Result

```

syntax = "proto3";

import "google/protobuf/any.proto";

package gateway;

option java_package = "com.sap.iotservices.common.protobuf.gateway";
option java_outer_classname = "CommandResponseProtos";

message CommandResponse {
    string capabilityAlternateId = 1;
    string sensorAlternateId = 2;
    Command command = 3;

    message Command {
        repeated google.protobuf.Any values = 1;
    }
}

syntax = "proto3";

import "CommandResponse.proto";

package gateway;

option java_package = "com.sap.iotservices.common.protobuf.gateway";
option java_outer_classname = "CommandResponseListProtos";

message CommandResponseList {
    repeated CommandResponse commands = 1;
}

```

 DKE DATA	Title: agrirouter Connectivity-Platform Integration Guide Part 2 Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

15.4.3. ParameterList

The result of a request to the outbox is an **array** of messages each including the following parameters:

Name	Type	Description
capabilityAlternateId	String	Equals value from onboarding
sensorAlternateId	String	Source of the message
command	String	The Base64 encoded message or a protobuf object, if agrirouter sent a protobuf object.

As this Array can include multiple messages, all further steps need to be done in a loop.

15.5. Decoding Base64

In case, a JSON Object was received, the String inside the Command is a Base64 encoded protobuf object. The app instance has to decode the message into a protobuf object then.

The Base64 string should be convertible to a byte stream, that includes a ResponseEnvelope and a ResponsePayloadWrapper.

15.6. Delimiting Envelope and Payload

Envelope and payload of each message are packaged into one delimited message. Separating those parts requires the delimiting functionality of protobuf. To know the format for the payload, the envelope has to be decoded first.

15.7. Envelope

The envelope is a protobuf object, that can be found in message agrirouter.response.ResponseEnvelope

Parameters are:

Position	Name	Type	Description
1	response_code	int64	A HTTP-like response code indicating the result
2	type	ResponseBodyType	An indicator of the correct decoding protobuf description
3	application_message_id	String	the corresponding message, that this answer is assigned to.
4	message_id	String	The agrirouter internal message ID used to be written to the agrirouter log and for confirmation.
5	timestamp	TimeStamp	The timestamp of the Result creation

15.7.1. response_code

The response code indicates success or failure of a request.

HTTP Response Code	Description
200	request processed/received successfully.
201	request processed/received and data created (this would be valid in cases for response to subscriptions, capabilities, teamset context messages, etc)
400	validation of request failed, messages in the body specifies the type of request failure
413	request too large
503	request cannot be processed at the moment, probably due to temporary overload

	<p>Title: agrirouter Connectivity-Platform Integration Guide Part 2</p> <p>Author: DKE-Data GmbH & Co. KG</p>	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

15.7.2. type

The type is required to determine the correct schema for the body:

The following List shows the corresponding Protobuf schemata for each ResponseType

ResponseBodyType	Schema (.proto)	Comment
MESSAGES	agrirouter.feed.response.MessageQueryResponse	A list of messages as result of a request for messages
ACK	There is no Schema for this, the payload will just have a size of 0 bytes	An acknowledgement of a message
ACK_WITH_MESSAGES	agrirouter.commons.Messages	An acknowledgement with messages; e.g. warnings.
ACK_WITH_FAILURE	agrirouter.commons.Messages	Represents a failure response in correlation to a message sent to the agrirouter
ACK_FOR_FEED_HEADER_LIST	agrirouter.feed.response.HeaderQueryResponse	Used for acknowledgements for feed envelope requests
ACK_FOR_FEED_MESSAGE	agrirouter.feed.response.MessageQueryResponse	Used for acknowledgements for feed payload requests.
ACK_FOR_FEED_FAILED_MESSAGE	agrirouter.feed.response.FailedMessageQueryResponse	Response for failed requests to feed queries
ENDPOINTS_LISTING	agrirouter.response.payload.account.ListEndpointsResponse	Used for a response which contains the endpoint listing
CLOUD_REGISTRATIONS	agrirouter.cloud.registration.OnboardingResponse	The result of a Virtual CU registration

Analysing the results is described in the specific chapters of process description for several

15.7.3. application_message_id

Indicates to which request this response is to be mapped (app message id is provided by sending application)

 DKE DATA	Title: agrirouter Connectivity-Platform Integration Guide Part 2	
	Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

15.7.4. message_id

The message_id is a unique ID of this message on the agrirouter. Use this id for the confirmation or deletion request.

15.7.5. timestamp

Time point, when this message was created by the sender.

15.8. Payload

There are several possible payload protobuf formats. Specific messages will be described in the following chapter, in this chapter we will only describe general payload answers.

15.8.1. Messages

The agrirouter.common.messages element includes a message as result of a command.

```

syntax = "proto3";
package agrirouter.common;

message Message {
    string message = 1;                                // Message text in
English Only
    string message_code = 2;                            // Globally defined
message code
    map<string, string> args = 3;                      // List of argument
names and values which would be inserted into the message text
}

message Messages {
    repeated Message messages = 1;                    // Collection of the
Message object listed above used in scenario's when there are multiple
messages in a response
}

```

It includes an array of Message:

Position	Name	Type	Description
1	message	string	A human readable description of the message including wildcard fields
2	message_code	string	A standardized code to analyse by a program
3	args	map<string,string>	A map of field+value pairs to add specific information to a standardized message

To display the message, replace all fields in the message with the corresponding values from the map.

The possible codes can be found in the attached file ErrorCodes.xlsx, specific values will be described in more detail within the corresponding Commands chapter.

 DKE DATA	Title: agrirouter Connectivity-Platform Integration Guide Part 2 Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

16. Overview of agrirouter commands

This chapter shall give you an overview of all the commands, that are relevant to control the agrirouter communication

Following a list of all commands; specific definitions follow in the next subchapters.

MessageType	Schema (.proto)	Comment
dke:capabilities	agrirouter.request.payload.endpoint.CapabilitySpecification	Endpoint to announce its capabilities in terms of technical message types that can be sent / received
dke:subscription	agrirouter.request.payload.endpoint.Subscription	Endpoint to subscribe for a certain technical message type, so that it receives published messages of this type
dke:feed_header_query	agrirouter.feed.request.MessageQuery	Endpoint to query for metadata of messages in its message feed (type, size, sender, time sent etc.)
dke:feed_message_query	agrirouter.feed.request.MessageQuery	Endpoint to query for messages in its message feed
dke:feed_confirm	agrirouter.feed.request.MessageConfirm	Endpoint to confirm that it has received a certain message (or set of messages)
dke:feed_delete	agrirouter.feed.request.MessageDelete	Endpoint to delete messages from its message feed
dke:list_endpoints	agrirouter.request.payload.account.ListEndpointsQuery	Endpoint to get a list of endpoints to which messages of a certain type can be sent (considering routing rules in place)
dke:list_endpoints_unfiltered	agrirouter.request.payload.account.ListEndpointsQuery	Endpoint to get a list of endpoints to which messages of a certain type can be sent (not considering routing rules)
dke:cloud_onboard_endpoints	agrirouter.cloud.registration.OnboardingRequest	A cloud system to create virtual CU endpoints in a paired agrirouter account
dke:cloud_offboard_endpoints	agrirouter.cloud.registration.OffboardingRequest	A cloud system to delete (offboard) virtual CU endpoints in a paired agrirouter account
iso:11783:-10:device_description:protobuf	efdi.ISO11783_TaskData	Communication Unit to send a device description message including a teamset Context ID
iso:11783:-10:time_log:protobuf	efdi.TimeLog	Live telemetry data according to ISO11783 TimeLog specification, encoded in protobuf

Remark

Those are the commands, that have the agrirouter as destination. Messages have other endpoints as destination. They are sent to the agrirouter like commands, but the agrirouter recognizes, that he only has to route and perhaps filter them.

 DKE DATA	Title: agrirouter Connectivity-Platform Integration Guide Part 2	
	Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

16.1. Describing the endpoint

This chapter will list all messages to tell the agrirouter, what the endpoint is capable of, who it is and what it wants to receive.

16.1.1. Capabilities Command

16.1.1.1. Definition

Every endpoint has to report his capabilities of sending and receiving messages after onboarding and when the capabilities change (e.g. an application functionality is extended by a new module). This is required from agrirouter to know, if specific messages can be routed to an endpoint and to show the correct choosable Information Types in the User Interface for routing.

Remark:

agrirouter specific commands are not part of the Capabilities, each application just has to support all of them.

Remark:

For the certification process it is required to send the capabilities message multiple times. Therefore, it must either be sent on manual request (request must be performable by the test processing person) or on reconnection/restart of the application.

The list of technical message types can include any TechnicalMessageFormat mentioned in 17 Technical Message types.

16.1.1.2. Command

Command	dke:capabilities
Protobuf Schema	agrirouter.request.payload.endpoint.CapabilitySpecification
TypeURL	types.agrirouter/agrirouter.request.payload.endpoint.CapabilitySpecification

Parameters are an Array of elements of the following:

Position	Name	Type	Description
1	capabilities	Array of capabilities	The list of capabilities of this endpoint
2	app_certification_id	String	The application ID assigned by agrirouter
3	app_certification_version_id	String	The ID of the apps certification version

The technical message type can be a technical message type of DKE or a proprietary message type.

Position	Name	Type	Description
1	technical_message_type	String	The name of the capability
2	direction	Direction	The message direction, that is part of the capabilities

Direction can be one of those values:

Value	Direction
0	Send
1	Receive
2	Send & Receive

Those are the values of an Enumeration, see protobuf files.

16.1.1.3. Result

ResultCode	ACK
Protobuf Schema	agrirouter.response.payload.endpoint.CapabilityResponse

 DKE DATA	Title: agrirouter Connectivity-Platform Integration Guide Part 2	
	Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

typeURL

types.agrirouter.agrirouter.response.payload.endpoint.CapabilityResponse

The result is a list of endpoints, each with a list of the capabilities, it can receive. Therefore, the capabilities messages can be used to get an overview of all endpoints available to send messages to.

CapabilityResponse.

Position	Name	Type	Description
1	application_message_id	String	The ID assigned with the request
2	recipients	Recipient (repeated)	A list of endpoints and their capabilities

Recipient is a subelement, describing an endpoint

Position	Name	Type	Description
1	id	String	The ID of the endpoint (devicealternateID)
2	technical_message_type	String (repeated)	A list of capabilities, this endpoint supports

16.1.1.4. Errors

If the message was malformed, an ACK_WITH_FAILURE will be returned. If the same capabilities message is sent twice, an ACK_WITH_MESSAGES will be returned with a message indicating, that the capabilities did not change.

16.1.1.5. Example

```
{
  "capabilities": [
    {
      "technicalMessageType": "iso:11783:-10:device_description:protobuf",
      "direction": 0
    },
    {
      "technicalMessageType": "iso:11783:-10:time_log:protobuf",
      "direction": 0
    },
    {
      "technicalMessageType": "dke:other",
      "direction": 1
    },
    {
      "technicalMessageType": "img/jpeg",
      "direction": 1
    },
    {
      "technicalMessageType": "iso:11783:-10:taskdata:zip",
      "direction": 2
    }
  ],
  "appCertificationId": "a94764fc-6704-47c2-aa39-9ca2490e0eb8",
  "appCertificationVersionId": "6776445b-09a0-49f9-9df0-f17ebe73e595"
}
```

16.1.2. Subscription Command

16.1.2.1. Definition

 DKE DATA	Title: agrirouter Connectivity-Platform Integration Guide Part 2 Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

The subscription command is used to subscribe for a list of technical message types. Being subscribed for a technical message type means, that your endpoint will receive a message of such TMT, whenever any other endpoint sends such a TMT to "publish" and a routing between the sender and your applications endpoint is given.

Remark

- Subscriptions can be defined for a specific message type, not for a specific sender.
- Each new subscription list sent by an endpoint deletes old subscriptions
 - ⇒ Always send all required subscriptions in one Command.
- Sending a new capabilities message will delete all subscriptions
 - ⇒ Always send all capabilities in one capabilities message

Remark

An app instance has to remember its subscription list on it own, there is no way to request this list.

Best Practice

To avoid mismatches between publishing applications and applications only receiving addressed messages, it is advised to subscribe for any technical message type and DDI, your application shall handle, if there are no specific reasons not to do so.

Remark:

For the certification process it is required to send the subscriptions message multiple times. Therefore, it must either be sent on manual request (request must be performable by the test processing person) or on reconnection/restart of the application.

The list of technical message types can include any TechnicalMessageFormat mentioned in 17 Technical Message types.

16.1.2.2. Command

Command	dke:subscription
Protobuf Schema	agrirouter.request.payload.endpoint.Subscription
TypeURL	types.agrirouter/agrirouter.request.payload.endpoint.Subscription

Position	Name	Type	Description
1	technical_message_type	Subscription (Repeated)	A list of subscriptions

It is an array, each entry is of type agrirouter.request.payload.endpoint.MessageTypeSubscriptionItem

Position	Name	Type	Description
1	technical_message_type	String	The technical message type
2	ddis	uint32(repeated)	A list of ddis, only relevant for the EFDI telemetry message type
3	position	bool	Shall the GPS position be delivered? Only relevant for EFDI telemetry message type

The DDIs field is only required, if the subscription is used for a subscription of Telemetry values. Same applies to Position.

16.1.2.3. Result

ResultCode	ACK
Protobuf Schema	None; Simply 0 bytes of answer
typeURL	""

In case of success, an Acknowledgement is received.

In case of failure, an Acknowledgement with Message or an Acknowledgement with Failure is received. In both cases, the protobuf format will be agrirouter.common.messages. See 15.8.1 Messages.

 DKE DATA	Title: agrirouter Connectivity-Platform Integration Guide Part 2	
	Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2		Confidence: external
Status: final	Version: 1.3	September 4, 2018

16.1.2.4. Errors

Errors will be reported using ACK_WITH_FAILURE. For a list of possible errors, see the error list.

16.1.2.5. Example

```
{
  "technicalMessageTypes": [
    {
      "technicalMessageType": "iso:11783:-10:time_log:protobuf",
      "ddi": [17, 23, 24],
      "positions": false
    },
    {
      "technicalMessageType": "img/jpeg",
      "ddi": [],
      "positions": false
    }
  ]
}
```

	Title: agrirouter Connectivity-Platform Integration Guide Part 2 Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2		Confidence: external
Status: final	Version: 1.3	September 4, 2018

16.1.3. Sending a teamset using device Descriptions

16.1.3.1. Definition

The teamset describes a list of devices attached to the communication unit. The most common devices are agricultural machines connected to the CU as ISO11783-10 TaskControllers. For more information on ISO11783-10, please refer to <https://aef-online.org>

Remark:

For the certification process it is required to send the device description message multiple times. Therefore, it must either be sent on manual request (request must be performable by the test processing person) or on reconnection/restart of the application.

For the format, please refer to 17.2 iso:11783:-10:device_description:protobuf - Teamset/EFDI Device Description

 DKE DATA	Title: agrirouter Connectivity-Platform Integration Guide Part 2	
	Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

16.2. Reading the feed

These commands are used to control the delivery of messages from the feed to the app instance through the outbox.

16.2.1. Paging

Due to the huge number of messages, that could be delivered with a single call for messages, it would be a bad idea, to deliver all those message headers in one big package. Therefore, a paging mechanism was introduced to split the whole list of messages or message headers. into several pages, each including a maximum number of messages or a maximum overall size of the Result.

All pages will be the result of one single GET Request, if all messages are delivered to the outbox, when the endpoint requests them. If not all messages have been delivered from feed to the outbox at this point in time, the endpoint will have to poll for the messages. In MQTT, all Pages are delivered one after each other.

Remark

If an app instance misses one package, it should first request the messages of the headers, it already received to shorten the list of headers delivered when requesting again. This only works for self containing message, it is critical, if the missed package includes parts of a chunked message.

16.2.2. Message Query and Filtering

Reading the messages, the message headers from the feed or deleting messages from the feed can be done using a message filter including the following parameters:

Position	Name	Type	Description
1	message_ids	String(repeated)	A list of agrirouter message IDs
2	senders	String(repeated)	A list of message senders
3	validity_period	Validity Period	A time span, that the messages were sent in

The validity Period has following parameters.

Position	Name	Type	Description
1	sentFrom	Timestamp	Begin of the time span
2	sentTo	Timestamp	End of the time span

A message with id, source and timestamp fits the filter, if

- (id is part of messageIDs or messageIDs is empty) AND
- (source is part of senders or senders is empty) AND
- (timestamp is within validity_period or validity_period is empty)

Remark:

Make sure, that you never request a validity period with a sentFrom older than 4 Weeks (28 days); agrirouter will return an error message instead of the header list for that. In case, you request a period within those 28 days and receive an error anyway, compare your local time with the agrirouter answer to avoid a time difference.

16.2.3. Call for Message Header List

16.2.3.1. Definition

To receive the Message ids of all available messages or a specific filtered list of available message IDs, an App Instance can call for the message headers available in the feed to be delivered to the Outbox

16.2.3.2. Command

Command	dke:feed_header_query
Protobuf Schema	agrirouter.feed.request.MessageQuery
TypeURL	types.agrirouter/agrirouter.feed.request.MessageQuery

 DKE DATA	Title: agrirouter Connectivity-Platform Integration Guide Part 2 Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

See 16.2.2 Message Query and Filtering.

16.2.3.3. Result

ResultCode	ACK_FOR_FEED_HEADER_LIST
Protobuf Schema	agrirouter.feed.response.HeaderQueryResponse
typeURL	types.agrirouter/agrirouter.feed.response.HeaderQueryResponse

The Result is a list of message headers (“envelopes”) without the message payload.

Position	Name	Type	Description
1	queryMetrics	QueryMetrics	A summarize of the response
2	page	Page	The current page of the message
3	chunk_contexts	ChunkComponent (repeated)	A list of all chunk contexts
4	feed	Feed (repeated)	A message from the feed

The Query metrics informs about several result parameters:

Position	Name	Type	Description
1	total_messages_in_query	int32	The total number of all messages headers in the response
2	max_count_restriction	int32	The maximum count of messages per page

The Paging information is included in the page parameter:

Position	Name	Type	Description
1	number	int32	The index of the current page
2	total	int32	The total number of pages

The chunk context is an Array of available chunk contexts within this messages. If there are multiple of them, this means, that there are multiple chunked messages to be realigned.

The chunk context is described in 13.2 Chunking big messages

The feed includes an array of message headers

Position	Name	Type	Description
1	sender	string	Endpoint ID of the sender
2	receiver	string	Endpoint ID of the receiver
3	header	Header (repeated)	An array of message headers

Remark:

As a telemetry platform can receive messages for multiple Virtual CUs, the receiver field is used to determine the correct virtual CU.

Position	Name	Type	Description
1	message_id	string	The message ID to request, delete or confirm the message
2	technical_message_type	string	The technical message type of the message
3	team_set_context_id	string	The teamset ID to assign the message to the correct context.
4	chunk_contextID	string	The chunk context corresponding to the chunk list
5	payload_size	int64	The size of the payload
6	sent_timestamp	timestamp	The timestamp, that was provided by the sender. It should be the recording time point

	Title: agrirouter Connectivity-Platform Integration Guide Part 2	
	Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

7	sequence_number	int64	The sequence number to determine the correct order for messages, that were recorded at the same time point
---	-----------------	-------	------------------------------------------------------------------------------------------------------------

Remark:

The result is sorted ascending by the senderID.

Within the sender-receiver package, the messages are sorted primary by the timestamp and secondary by the sequence number.

The header is a list of Sender+Receiver-Tuples with a sublist of message headers. This sublist is sorted by the timestamp and – if multiple messages with the same timestamp exist – the Message Sequence Number.

16.2.3.4. Errors

If the message was incorrect, an ACK_WITH_FAILURE will be reported. For specific error messages, see the error list.

 DKE DATA	Title: agrirouter Connectivity-Platform Integration Guide Part 2	
	Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

16.2.4. Call for specific messages

16.2.4.1. Definition

Every app Instance can request a single or a list of messages to be forwarded from the feed to the outbox by its message ids

16.2.4.2. Command

Command	dke:feed_message_query
Protobuf Schema	agrirouter.feed.request.MessageQuery
typeURL	types.agrirouter/agrirouter.feed.request.MessageQuery

See 16.2.2Message Query and Filtering.

16.2.4.3. Result

ResultCode	ACK_FOR_FEED_MESSAGE
Protobuf Schema	types.agrirouter/agrirouter.feed.response.MessageQueryResponse

The Result is a list of message headers ("envelopes") without the message payload.

Position	Name	Type	Description
1	queryMetrics	QueryMetrics	A summarize of the response
2	page	Page	The current page of the message
3	messages	FeedMessage (repeated)	A message from the feed

The Query metrics informs about several result parameters:

Position	Name	Type	Description
1	total_messages_in_query	int32	The total number of all messages headers in the response
2	max_count_restriction	int32	The maximum number of messages per page

The Paging information is included in the page parameter:

Position	Name	Type	Description
1	number	int32	The index of the current page
2	total	int32	The total number of pages

The messages include an array of messages

Position	Name	Type	Description
1	header	Header	The header of the message
2	content	any	The payload in the corresponding protobuf format

The header includes the whole envelope of a message

Position	Name	Type	Description
1	receiver_id	string	The receiver; might be a secondary endpoint like a virtual CU behind a telemetry platform
2	technical_message_type	string	The technical message type of the message
3	team_set_context_id	string	The teamset ID to assign the message to the correct context.
4	chunk_contextID	string	The chunk context corresponding to the chunk list
5	payload_size	int64	The size of the payload

	Title: agrirouter Connectivity-Platform Integration Guide Part 2	
	Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

6	sent_timestamp	timestamp	The timestamp, that was provided by the sender. It should be the recording time point
7	sequence_number	int64	The sequence number to determine the correct order for messages, that were recorded at the same time point
8	sender_id	string	The endpoint ID of the sender
9	created_at	Timestamp	The timestamp, when this message was added to the receiving endpoints feed

Remark:

The result is sorted ascending by the senderID.

Within the sender-reciever package, the messages are sorted primary by the timestamp and secondary by the sequence number.

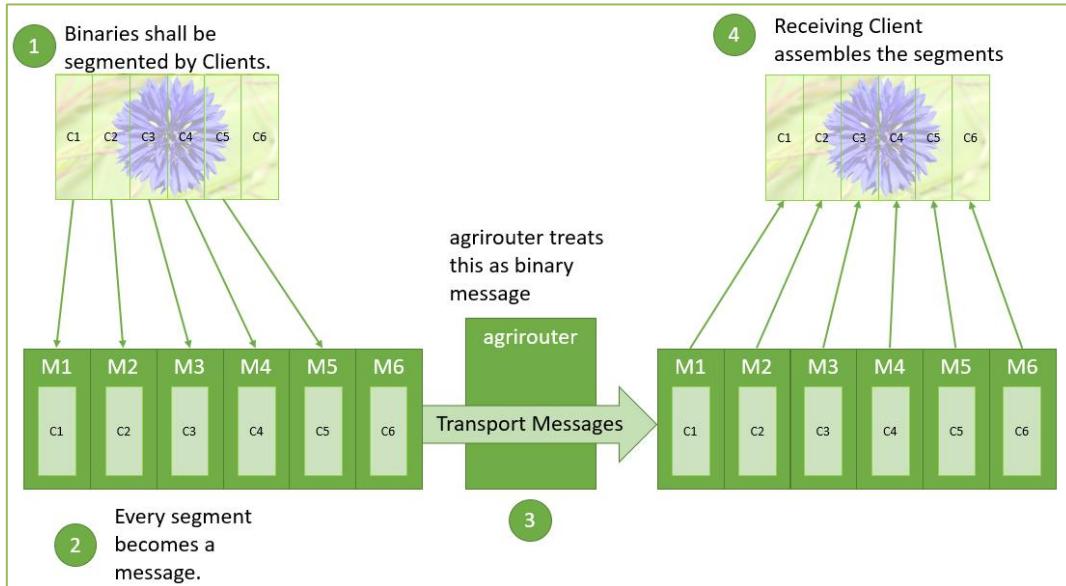
16.2.4.4. Errors

If the message was incorrect, an ACK_WITH_FAILURE will be reported. For specific error messages, see the error list.

 DKE DATA	Title: agrirouter Connectivity-Platform Integration Guide Part 2	
	Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

16.2.4.5. Chunked messages

Messages sent to the agrirouter can be split into multiple chunks, if the message format is not EFDI.



Only those message, that were not created by the agrirouter and that are not of type EFDI can be chunked.

16.2.4.5.1. Recognizing chunked messages in the feed

To recognize chunked messages, request the message header query and see, if you find different chunk contexts.

16.2.4.5.2. Pulling chunked messages from the feed

Chunked messages can be pulled like any other message type. make sure to request all chunks at once, so that you can make sure, that the message can be rebuild successfully before confirming chunks, which would delete them from the feed.

	<p>Title: agrirouter Connectivity-Platform Integration Guide Part 2</p> <p>Author: DKE-Data GmbH & Co. KG</p>	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

16.2.5. Call for message deletion

16.2.5.1. Definition

An app instance can delete message from its feed, if it does not want to consume them. Therefore, it sends a list of message IDs or a validity period or a list of senders to the inbox.

16.2.5.2. Command

Command	dke:feed_delete
Protobuf Schema	agrirouter.feed.request.MessageQuery
typeURL	types.agrirouter/agrirouter.feed.request.MessageQuery

See 16.2.2Message Query and Filtering.

16.2.5.3. Result

ResultCode	ACK
Protobuf Schema	None, simply 0 bytes
typeURL	""

16.2.5.4. Errors

If the message was incorrect, an ACK_WITH_FAILURE will be reported. For specific error messages, see the error list.

 DKE DATA	Title: agrirouter Connectivity-Platform Integration Guide Part 2 Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

16.3. Controlling the outbox

To make sure, that no message gets lost due to e.g. a loss of internet connection while delivering a message, the app instance has to confirm the receipt of every message.

16.3.1. Call for message list confirmation

16.3.1.1. Definition

Once a message was downloaded from the outbox, the Client has to confirm, that it properly received this message/those messages.

Remark:

As long as the receipt of a message is not confirmed, it will be delivered again and again by agrirouter, whenever the client tries to download messages, it called for. When a message is confirmed, it will be deleted from the feed.

16.3.1.2. Command

Command	dke:feed_confirm
Protobuf Schema	agrirouter.feed.request.MessageConfirm
typeURL	types.agrirouter/agrirouter.feed.request.MessageConfirm

MessageConfirm is simply an array of message IDs.

16.3.1.3. Result

ResultCode	ACK
Protobuf Schema	None, simply 0 bytes
typeURL	""

16.3.1.4. Errors

If the message was incorrect, an ACK_WITH_FAILURE will be reported. For specific error messages, see the error list.

 DKE DATA	Title: agrirouter Connectivity-Platform Integration Guide Part 2 Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

16.4. Read the eco system

To communicate with all the other endpoints in the eco system of the specific agrirouter account, it is required to read,

- which endpoints are available
- which TMT they can send
- which TMT they can receive
- which TMT they are subscribed for

16.4.1. Call for endpoints, that support a technical message type

16.4.1.1. Definition

Every App Instance can call for a list of endpoints, that are capable of receiving or sending specific messages. This request does not filter for available routings.

16.4.1.2. Command

Command	dke:list_endpoints_unfiltered
Protobuf Schema	agrirouter.request.payload.account.ListEndpointsQuery
typeURL	types.agrirouter/agrirouter.request.payload.account.ListEndpointsQuery

Position	Name	Type	Description
1	technical_message_type	String	The technical message type that should be part of the routing
2	direction	Direction	The direction, the routing shall support

The direction describes, if messages shall be sendable or receivable:

Nr	Direction
0	Send
1	Receive
2	Send and Receive

16.4.1.3. Result

ResultCode	ENDPOINTS_LISTING
Protobuf Schema	agrirouter.response.payload.account.ListEndpointsResponse
typeURL	types.agrirouter/agrirouter.response.payload.account.ListEndpointsResponse

The endpoint response includes a list of available endpoints:

Position	Name	Type	Description
1	endpoint_id	String	The endpoint ID to address a message to
2	endpoint_name	String	The name of the endpoint as displayed in the agrirouter UI
3	endpoint_type	String	The type of Endpoint, see below
4	status	String	The current activity status of the endpoint
5	message_types	MessageTypes	See below
6	external_id	String	The id provided as "id" in the onboarding request

The endpoint Type can be one of the following Values:

Code	Description	Comment
application	Software Application	Software Application

	<p>Title: agrirouter Connectivity-Platform Integration Guide Part 2</p> <p>Author: DKE-Data GmbH & Co. KG</p>	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

foreignMachine	External Machine	Machine shared by a paired agrirouter account
machine	Machine	Agricultural machine
messageRecordingReplay	Recording Player	Can replay recorded messages
pairedAccount	Paired Account	Represents a paired Agri-Router account with which you can exchange messages
endpointGroup	Group	A group of other assets (applications, machines) with common rules
virtualCU	Telemetry Platform	Connection via Telemetry Platform

The status can be one of the following values:

active
inactive

The Message object describes technical message type and direction.

Position	Name	Type	Description
1	technical_message_type	String	The technical message type that should be part of the routing
2	direction	Direction	The direction, the routing shall support

The direction describes, if messages shall be sendable or receivable:

Nr	Direction
0	Send
1	Receive
2	Send and Receive

Remark:

The endpoint requesting the endpoint list is also part of this answer.

Remark:

The concept of groups is only used to give a little more comfort to the end user when setting rules. It is currently not possible to address a message to the group.

16.4.1.4. Errors

If the message was incorrect, an ACK_WITH_FAILURE will be reported. For specific error messages, see the error list.

	<p>Title: agrirouter Connectivity-Platform Integration Guide Part 2</p> <p>Author: DKE-Data GmbH & Co. KG</p>	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

16.4.2. Call for filtered list of endpoints, that support a specific message type

16.4.2.1. Definition

This request is used for request a list of endpoints, that support a technical message type and has corresponding routings to the requesting endpoint.

16.4.2.2. Command

Command	dke:list_endpoints
Protobuf Schema	agrirouter.request.payload.account.ListEndpointsQuery
typeURL	types.agrirouter/agrirouter.request.payload.account.ListEndpointsQuery

For further information, see 16.4.1.2Command

16.4.2.3. Result

ResultCode	ENDPOINTS_LISTING
Protobuf Schema	agrirouter.response.payload.account.ListEndpointsResponse
typeURL	types.agrirouter/agrirouter.response.payload.account.ListEndpointsResponse

For further information, see 16.4.1.3

16.4.2.4. Errors

If the message was incorrect, an ACK_WITH_FAILURE will be reported. For specific error messages, see the error list.

	<p>Title: agrirouter Connectivity-Platform Integration Guide Part 2</p> <p>Author: DKE-Data GmbH & Co. KG</p>	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

16.5. Commands for telemetry platforms

Telemetry platforms on- and offboard virtual CUs, this requires 2 commands.

Remark

If a telemetry platform does not provide any functionality of a farming software, those are the only 2 messages exchanged with the agrirouter by the Telemetry platform as any other communication will be done by the virtual CUs.

16.5.1. Onboarding a Virtual CU

16.5.1.1. Definition

A telemetry platform can onboard its own CUs, which will show up in the agrirouter UI as own endpoints. These virtual CUs communicate

16.5.1.2. Command

Command	dke:cloud_onboard_endpoints
Protobuf Schema	agrirouter.cloud.registration.OnboardingRequest
typeURL	types.agrirouter/agrirouter.cloud.registration.OnboardingRequest

The onboarding Request includes an Array of EndpointRegistrationDetails

Position	Name	Type	Description
1	id	String	The unique ID of the Virtual CU
2	Name	String	The name, that shall be displayed in the agrirouter UI

For the ID, please respect the Naming convention in 4.2String Identifiers convention

16.5.1.3. Result

ResultCode	CLOUD_REGISTRATIONS
Protobuf Schema	agrirouter.cloud.registration.OnboardingResponse

The Result is an Array of EndpointRegistrationDetails and Failure Notifications

Position	Name	Type	Description
1	onBoardedEndpoints	EndpointRegistrationDetails (Repeated)	The onboarding information for the endpoints
2	failures	Failures(Repeated)	A list of failed onboardings

For all successfully registered Endpoints, following information will be delivered:

Position	Name	Type	Description
1	id	String	The ID external ID provided with the onboarding Request as id
2	deviceAlternateID	String	The ID used to mark messages sent to agrirouter as coming from this Cloud account
3	sensorAlternateID	String	The ID used to mark messages sent to agrirouter as coming from this virtual CU
4	capabilityAlternateID	String	An internal value needed to be delivered with any message
5	endpointID	String	The endpointID to address this Virtual CU

	<p>Title: agrirouter Connectivity-Platform Integration Guide Part 2</p> <p>Author: DKE-Data GmbH & Co. KG</p>	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

The **deviceAlternateId** is equal for the telemetry platform and each of its virtual CUs. This means, that the messages for all virtual CUs are received through one feed and one outbox. The effected CU however is marked by its **sensorAlternateId**.

The **endpointId** is used to offboard a CU.

16.5.1.4. Errors

Errors of a partly successful request are delivered with the Response.

If any of the onboarding Requests failed, failures includes a list of failed requests:

Position	Name	Type	Description
1	id	String	The ID external ID provided with the onboarding Request as id
2	message	Message	The error message

For the message format, refer to 15.8.1 Messages

16.5.2. Removing a Virtual Cu

16.5.2.1. Definition

This command removes a virtual CU from the endpoint list of the users account.

Attention

Removing the endpoint leads to a dump of all data in the in- or outbox, subscription list and feed of this endpoint.

Your customers data will be lost!

Do not remove an endpoint, just because it is currently offline!

16.5.2.2. Command

Command	dke:cloud_offboard_endpoints
Protobuf Schema	agrirouter.cloud.registration.OffboardingRequest
typeURL	types.agrirouter/agrirouter.cloud.registration.OffboardingRequest

16.5.2.3. Result

ResultCode	ACK
Protobuf Schema	None, simply 0 bytes
typeURL	“”

16.5.2.4. Errors

See list of error codes.

	<p>Title: agrirouter Connectivity-Platform Integration Guide Part 2</p> <p>Author: DKE-Data GmbH & Co. KG</p>	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

17. Technical Message types

17.1. General

Technical message types are several standardized formats, that agrirouter apps should be capable of reading and creating. These technical message types are packaged into information types for the end user to make it easier for him to create the routings based on the type of data and not on the format of it.

The possible technical message types can be found in the agrirouter UI when creating a new application:

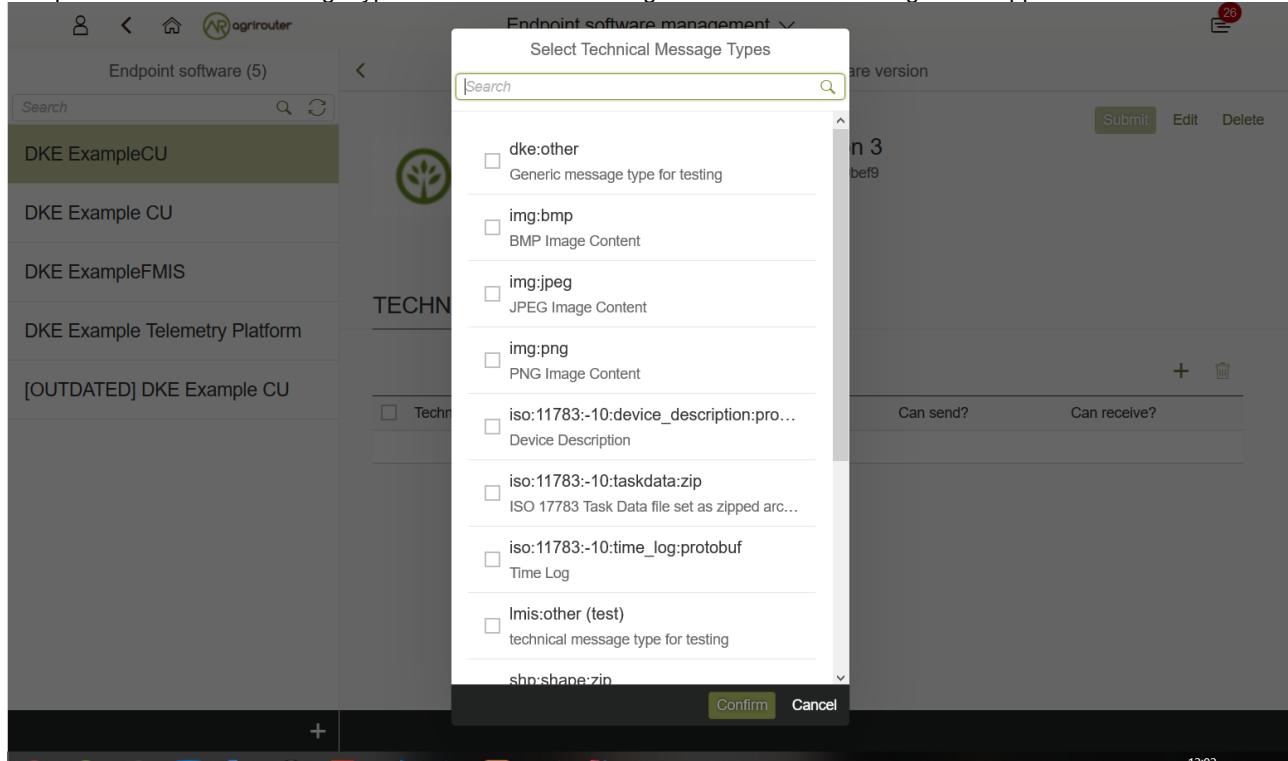


Figure 39 Selecting Technical Message Types in the agrirouter UI

It can be any of these formats:

Technical Message Type	Information Type	Description
iso:11783:-10:taskdata:zip	TaskData	A zipfile including a TaskData Set
iso:11783:-10:device_description:protobuf	Telemetry	A set of device descriptions connected to a CU. Also called "TeamSet"
iso:11783:-10:time_log:protobuf	Telemetry	Life telemetry data based on the devicedescription
img:bmp	Image	A Bitmap File
img/jpeg	Image	A JPEG File
img:png	Image	A Portable Network Graphics File
shp:shape:zip	Shape	A zip file including a Shape dataset
vid:avi	Video	An AVI Video
vid:mp4	Video	A MPEG4 Video
vid:wmv	Video	A wmv Video

	Title: agrirouter Connectivity-Platform Integration Guide Part 2 Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

17.2. iso:11783:-10:device_description:protobuf - Teamset/EFDI Device Description

17.2.1. Definition

This format is used to report devices, that are attached to a CU to the agrirouter. These devices including the CU are known as a teamset, having their own teamsetID.

17.2.2. Command

Command	iso:11783:-10:device_description:protobuf
Protobuf Schema	EFDI; available at AEF Repository
TypeURL	iso:11783:-10:device_description:protobuf

The teamset message is part of EFDI and therefore currently not available here. It will be available by AEF.

The AEF protobuf files can be found here: <https://github.com/aefev/fmisei-spec/tree/master/proto-v3-grpc> .

Remark

Access to the AEF repository can be requested at the AEF Office, see <https://aef-online.org>

17.2.3. Result

The result indicates, if the teamset could be forwarded to an addressed endpoint or – if it was published – if there was a subscribed endpoint, who received the message. If no endpoint received the message, there will be an ACK_WITH_FAILURE

Remark:

If a CU sends a teamset, applications are able to address messages directly to the devices. These messages are then delivered to the CU.

	Title: agrirouter Connectivity-Platform Integration Guide Part 2 Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

17.3. iso:11783:-10:time_log:protobuf - EFDI TimeLog

The EFDI timelog is comparable to ISOXML Binary Log data. the format with all its parameters can be received from the AEF repository.

17.3.1. Command

Command	iso:11783:-10:time_log:protobuf
Protobuf Schema	EFDI; available at AEF Repository
TypeURL	iso:11783:-10:time_log:protobuf

	Title: agrirouter Connectivity-Platform Integration Guide Part 2	
	Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

17.4. iso:11783:-10:taskdata:zip - TaskData

17.4.1. Definition

This format is used to exchange a TaskData.zip Files. It can forward the binary Zip-File-Content, but currently no file name or other meta data.

17.4.2. Command

Command	iso:11783:-10:taskdata:zip
Protobuf Schema	-
TypeURL	""

17.4.3. Format

The zip file has to be formatted to a Base64-encoding of its binary data. It must be sent as a string.

	Title: agrirouter Connectivity-Platform Integration Guide Part 2	
	Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

17.5. shp:shape:zip - Shape

17.5.1. Definition

This format is used to exchange application maps as a zip file.

17.5.2. Command

Command	shp:shape:zip
Protobuf Schema	-
TypeURL	""

17.5.3. Format

The zip file has to be formatted to a Base64-encoding of its binary data. It must be sent as a string.

The zip file must at least include the following files:

- SHP: The main file including the geometry
- SHX: The index file indexing the geometry features
- DBF: The dBASE Table, including the attribute information of the features

	Title: agrirouter Connectivity-Platform Integration Guide Part 2 Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

17.6. img:bmp - Bitmap Image

17.6.1. Definition

This format is used to exchange a bitmap file. It can forward the binary bmp-Content, but currently no file name or other meta data.

17.6.2. Command

Command	img:bmp
Protobuf Schema	-
TypeURL	""

17.6.3. Format

The bmp file has to be formatted to a Base64-encoding of its binary data. It must be sent as a string.

	Title: agrirouter Connectivity-Platform Integration Guide Part 2 Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2		Confidence: external
Status: final	Version: 1.3	September 4, 2018

17.7. img:jpg – JPEG Image

17.7.1. Definition

This format is used to exchange a bitmap file. It can forward the binary jpeg-Content, but currently no file name or other meta data.

17.7.2. Command

Command	img:jpg
Protobuf Schema	-
TypeURL	""

17.7.3. Format

The bmp file has to be formatted to a Base64-encoding of its binary data. It must be sent as a string.

	Title: agrirouter Connectivity-Platform Integration Guide Part 2 Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

17.8. img/png – PNG Image

17.8.1. Definition

This format is used to exchange a bitmap file. It can forward the binary png-Content, but currently no file name or other meta data.

17.8.2. Command

Command	img:png
Protobuf Schema	-
TypeURL	""

17.8.3. Format

The bmp file has to be formatted to a Base64-encoding of its binary data. It must be sent as a string.

	Title: agrirouter Connectivity-Platform Integration Guide Part 2	
	Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

17.9. vid:avi – AVI Video Files

17.9.1. Definition

This format is used to exchange an avi video file. It can forward the binary avi-Content, but currently no file name or other meta data.

17.9.2. Command

Command	vid:avi
Protobuf Schema	-
TypeURL	""

17.9.3. Format

The bmp file has to be formatted to a Base64-encoding of its binary data. It must be sent as a string.

	Title: agrirouter Connectivity-Platform Integration Guide Part 2	
	Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

17.10. vid:wmv – WMV Video Files

17.10.1. Definition

This format is used to exchange an wmv video file. It can forward the binary wmv-Content, but currently no file name or other meta data.

17.10.2. Command

Command	vid:wmv
Protobuf Schema	-
TypeURL	""

17.10.3. Format

The bmp file has to be formatted to a Base64-encoding of its binary data. It must be sent as a string.

	Title: agrirouter Connectivity-Platform Integration Guide Part 2	
	Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

17.11. vid:mp4 – MPEG4 Video Files

17.11.1. Definition

This format is used to exchange an MPEG4 video file. It can forward the binary mp4-Content, but currently no file name or other meta data.

17.11.2. Command

Command	vid:mp4
Protobuf Schema	-
TypeURL	""

17.11.3. Format

The bmp file has to be formatted to a Base64-encoding of its binary data. It must be sent as a string.

 DKE DATA	Title: agrirouter Connectivity-Platform Integration Guide Part 2	
	Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

18. Session workflow

18.1. General Overview

Every application instance communicates to the agrirouter in sessions. The first message can be sent after onboarding is done. It shall report the capabilities.

These capabilities must match those stated in the application version certification or a subset of them. If the endpoint wants to 'listen' to certain messages, it must subscribe to the appropriate technical message types, so that published messages get routed to this endpoint. Messages however will only get routed, if there is a routing for that

Every request sent to the agrirouter inbox will be acknowledged by the agrirouter, indicating whether the request could be processed successfully, with warning or it could not be processed because of errors; in that case, an error code and an error message in English are returned. Communication Layers.

After that, the teamset is reported and a communication workflow can begin. The Workflow is as follows:

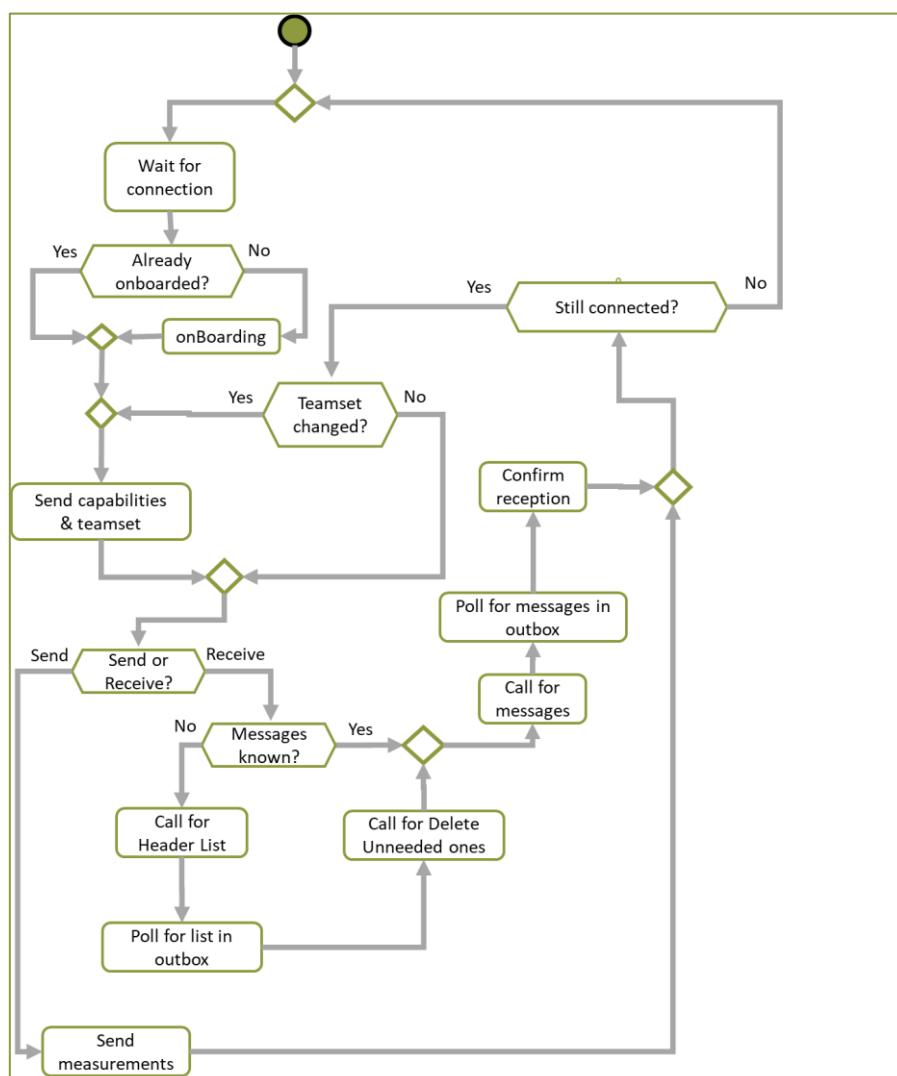


Figure 40 General messaging workflow

	<p>Title: agrirouter Connectivity-Platform Integration Guide Part 2</p> <p>Author: DKE-Data GmbH & Co. KG</p>	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

Remark:

The displayed workflow is just one possible solution. It is up to the app provider or his developer to find the most practical workflow of requesting, which messages are available and requesting or deleting those messages.

18.2. Request sending Frequency

agrirouter needs some time to process commands and forward messages, that are received through one of the endpoints. To avoid mixing up communication and requesting the next message before receiving the result of the last command, the agrirouter support team advices, to have a minimum time lapse between sending of 2 requests to the agrirouter:

Environment	Advised time between 2 requests
Quality assurance	10 s
Production	5 s

Remark

EFDI Log Telemetry messages can be collected over several seconds and than be sent in one request to the agrirouter.

 DKE DATA	Title: agrirouter Connectivity-Platform Integration Guide Part 2 Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

19. IDs, IDs, IDs – Which is which?

19.1. Overview

The agrirouter api uses several IDs, which can be a little confusing. But don't worry, here is a short description, which ID is used for what:

Name	Description
accountId	A UUID describing the account of an end user. In one account, there can be multiple endpoints. One account however has one set of login credentials for the agrirouter
applicationId	A UUID identifying an application regardless of the software version
capabilityAlternateId	This is an Id that is delivered with the onboarding Request and has to be delivered with any message
certificationVersionId	A UUID representing a Certification for one app. an app has to be certified every time, a new version with changed capabilities against agrirouter are implemented.
deviceAlternateId	The Id of the sender of a message
endpointId	The endpointId is the unique address of an Application Instance on the agrirouter. It's used to address the outbox and inbox and to directly address an endpoint from a different endpoint
gatewayId	This is a simple enumeration to determine the protocol type of the communication. It's either 2=MQTT or 3=REST
sensorAlternateId	The Id of the recorder or the final receiver of a message

19.2. Ids in different Environments

It is important to keep in mind, that all Ids including applicationId or certificationVersionId might differ between QA and Productive environments.

19.3. Practical differentiation between SensorAlternateId and DeviceAlternateId

sensorAlternateID and deviceAlternateId seem to be pretty equal, but the context differs. Here are some practical examples

19.3.1. An App sends a TaskData

For apps, the solution is pretty easy:

sensorAlternateId = deviceAlternateId = The endpoint representing the users account within the app.

19.3.2. A CU sends a TaskData

Assume, a CU wants to send a taskdata.zip. Even though, it could include machine data - perhaps of multiple machines, the source of this taskdata is the CU. Therefore:

sensorAlternateId = deviceAlternateId = The CU

19.3.3. A telemetry Platform sends a TaskData from a virtual CU

The virtual CU as well as the telemetry platform are endpoints in the agrirouter. The platform wants to mark the virtual CU as source of the taskdataset.

deviceAlternateId: The telemetry platform is source of the message

sensorAlternateId: The virtual CU is source of the content of the message

 DKE DATA	Title: agrirouter Connectivity-Platform Integration Guide Part 2	
	Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

19.4. accountId

19.4.1. Description

The Unique Identifier for an endusers or developers account. Relevant for billing.
A UUID describing the account of an end user. In one account, there can be multiple endpoints. One account however has one set of login credentials for the agrirouter UI.

19.4.2. Where to find

19.4.2.1. As API/As Developer

The value is delivered with the authentication request. The value is also part of the billing metrics

19.4.2.2. As End User

The value can be found in the agrirouter UI endpoint Information.

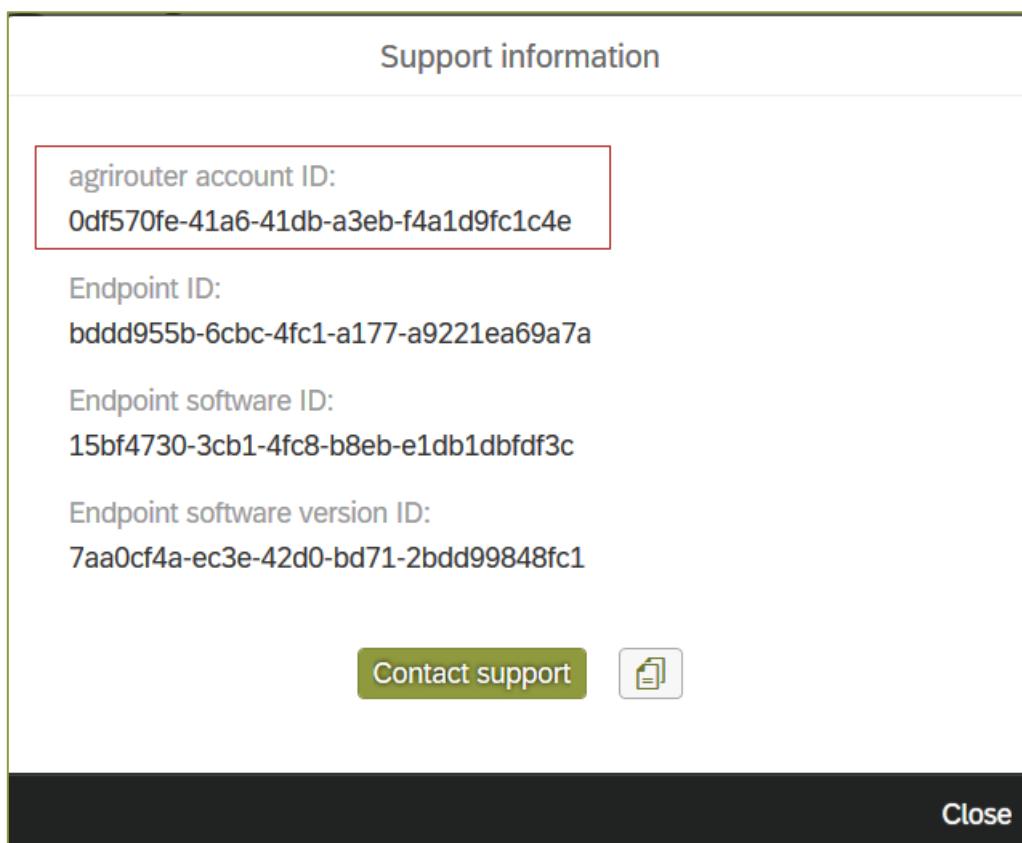


Figure 41 agrirouter account ID

 DKE DATA	Title: agrirouter Connectivity-Platform Integration Guide Part 2	
	Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

19.5. applicationId

19.5.1. Description

A UUID identifying an application regardless of the software version.

19.5.2. Where to find

19.5.2.1. As API

The value cannot be found by the api, it has to be entered by the developer.
The developer can find the ID in his developer account:

19.5.2.2. As End User

The value can be found in the agrirouter UI endpoint Information

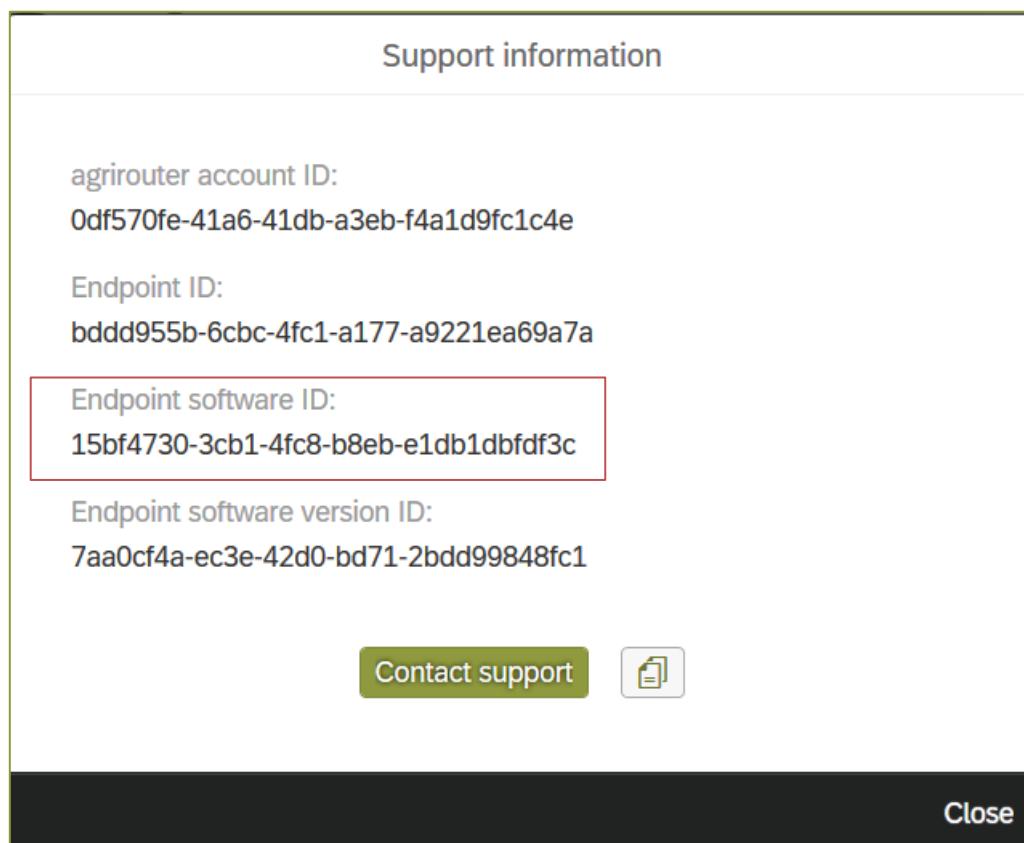


Figure 42 agrirouter endpoint software ID

	Title: agrirouter Connectivity-Platform Integration Guide Part 2 Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2		Confidence: external
Status: final	Version: 1.3	September 4, 2018

19.6. capabilityAlternateld

19.6.1. Description

This is a value required by the IoT Gateway of the agrirouter. It has no further meaning for the endpoint or app instance and shall just be delivered with requests.

19.6.2. Where to find

19.6.2.1. As API

The value is delivered with the onboarding request

19.6.2.2. As End User

The value cannot be found by an end user and has no meaning for him.

 DKE DATA	Title: agrirouter Connectivity-Platform Integration Guide Part 2	
	Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

19.7. certificationVersionId

19.7.1. Description

A UUID representing a certification for one app. an app has to be certified every time, a new version with changed capabilities against agrirouter are implemented.

19.7.2. Where to find

19.7.2.1. As API

The value cannot be found by the API. It has to be entered by the developer.
The developer can find the certificationVersionID in his endpoint software overview:

19.7.2.2. As End User

The value can be found in the agrirouter UI endpoint Information

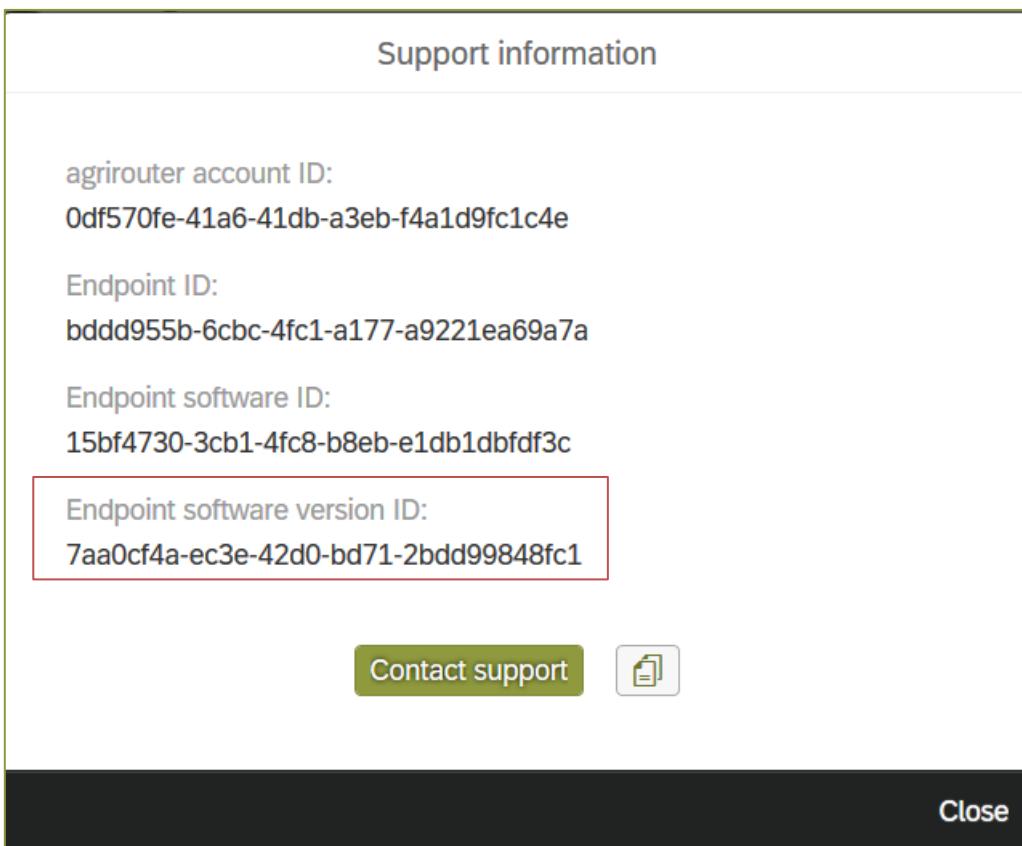


Figure 43 agrirouter endpoint software version ID

	Title: agrirouter Connectivity-Platform Integration Guide Part 2 Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

19.8. deviceAlternateld

19.8.1. Description

The deviceAlternateID represents the source of an agrirouter command, but not necessarily the source of the message itself. E.g. a telemetry platform would mark itself as deviceAlternateId and the virtual CU as source of message (content)

19.8.2. Where to find

19.8.2.1. As API

The value is delivered with any agrirouter message. To create message, take the context relevant endpointId. Context is the command, e.g. an onboarding request.

19.8.2.2. As End User

This ID cannot be found in the UI by the end user

 DKE DATA	Title: agrirouter Connectivity-Platform Integration Guide Part 2	
	Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

19.9. endpointId

19.9.1. Description

The endpointID is the unique address of an Application Instance on the agrirouter. It's used to address the outbox and inbox and to directly address an endpoint from a different endpoint.
An endpoint can be an Application, a CU, a Virtual CU, a machine or a Telemetry platform.

19.9.2. Where to find

19.9.2.1. As API

The value is delivered with the onboarding request.

19.9.2.2. As End User

The value can be found in the agrirouter UI endpoint Information

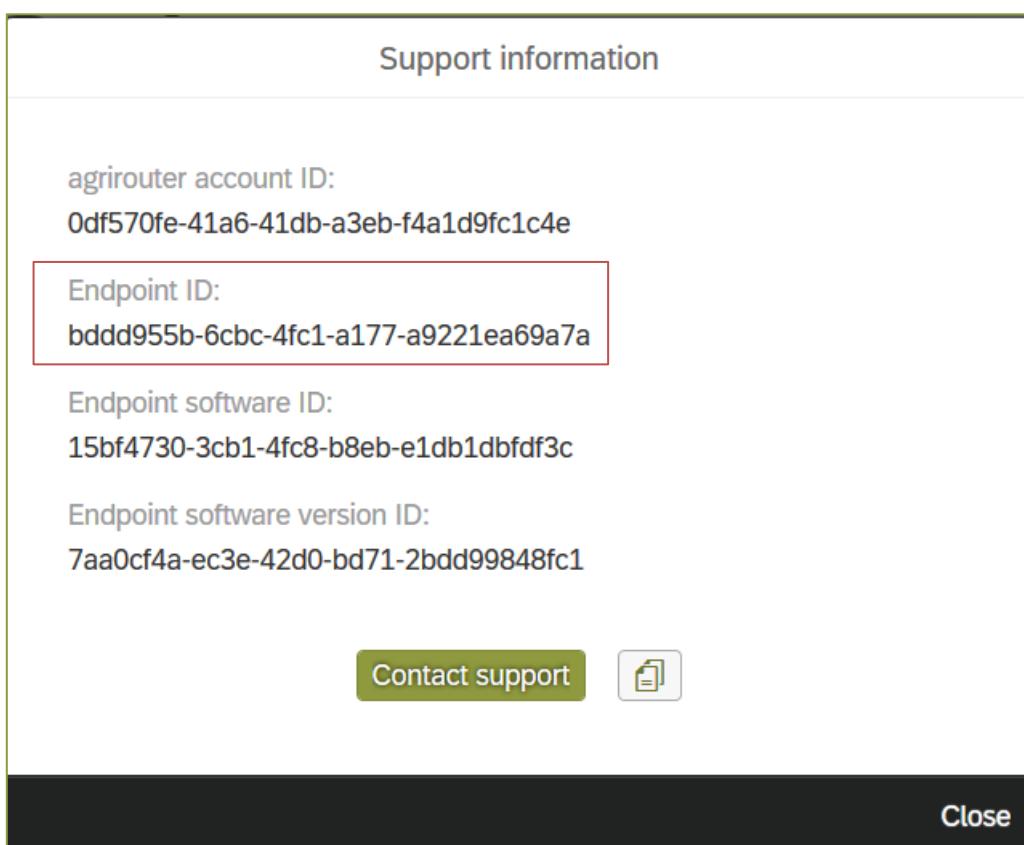


Figure 44 agrirouter endpoint ID

	Title: agrirouter Connectivity-Platform Integration Guide Part 2 Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2		Confidence: external
Status: final	Version: 1.3	September 4, 2018

19.10. gatewayId

19.10.1. Description

The gatewayId is an enumeration to determine the used protocol for all communication after the onboarding:

2=MQTT

3=HTTP

19.10.2. Where to find

19.10.2.1. As API

see above

19.10.2.2. As End User

see above

	Title: agrirouter Connectivity-Platform Integration Guide Part 2 Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

19.11. sensorAlternateId

19.11.1. Description

The sensorAlternateID is an endpointID representing the parent telemetry platform for a Virtual CU.
It can be used to on- and offboard new virtual CUs.

19.11.2. Where to find

19.11.2.1. As API

The value is delivered with any agrirouter message. To create message, take the context relevant endpointID. Context is the “source” of a message, e.g. a machine.

19.11.2.2. As End User

The id is an endpointID and can be found like the endpoint Id

 DKE DATA	Title: agrirouter Connectivity-Platform Integration Guide Part 2 Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

19.12. TeamsetContextId

19.12.1. Description

The TeamsetContextId is used to describe a unique combination of different machines and CUs attached to a CU or Virtual CU. It must be defined by the CU creating a URN:

```
urn:hash::algorithm:value
```

The agrirouter team advises to use a hashing algorithm of either md5 or sha256.

Examples (the key is the hash of "Hello World"):

MD5:

```
urn:hash::md5:b10a8db164e0754105b7a99be72e3fe5
```

Sha256:

```
urn:  
hash::sha256:a591a6d40bf420404a011733cfb7b190d62c65bf0bcd32b57b277d9ad9f146e
```

19.12.2. Where to find

19.12.2.1. As API

The TeamsetContextId is sent in the envelope of every message. This information is forwarded through agrirouter so that it can be received by an app instance receiving this message.

19.12.2.2. As End User

An end user cannot see this value.

	<p>Title: agrirouter Connectivity-Platform Integration Guide Part 2</p> <p>Author: DKE-Data GmbH & Co. KG</p>	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

20. Certificates and keys

20.1. Definitions

Within this document, there are several public or private keys mentioned, that are relevant for the agrirouter. The following table shall give an overview of the different Keys/Certificates and their usage:

Name	Description	Usage
agrirouter public key	A certificate to proof the identity of the agrirouter. Only the public key is available to developers	Verify the signature in the authentication process
Application Key Pair	The key pair, that can be provided to or created by the agrirouter when creating a new software.	Create the signature for the onboarding Process
Endpoint Certificate	The certificate of an endpoint, used for the encrypted communication with the agrirouter	Standard communication in REST or MQTT; "Everything after onboarding"

20.2. Agrirouter public keys

These are the public keys used by the agrirouter.

Environment	Public Key
EU1 QA	-----BEGIN PUBLIC KEY----- MIIBljANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAY8xF9661acn+iS+QS+9Y 3hvTUVcismzbuvxHgHA7YeoOUFxyj3lkaTnXm7hzQe4wDEDgwpJSGAzxIISUXe 8EsWLorg5O0tRexx5SP3+kj1i83DATBJCXP7k+bAF4u2FVJphC1m2BfLxeLGljzx VAS/v6+EwvYaT1Al9FFqW/a2o92lsVPOh9oM9eds3IBOAbH/8XrmVleHofw+XbTH 1/7MLD6IE2+HbEeY0F96nioXArqQWXcjUQsTch+p0p9eqh23Ak4ef5oGcZhNd4yp Y8M6ppvIMiXkgWSPJevCJhxRJRmndY+ajYGx7CLePx7wNvxXWtkng3yh+7WiZ/Y qwIDAQAB -----END PUBLIC KEY-----
EU1 Production	-----BEGIN PUBLIC KEY----- MIIBljANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAcxD31sYtzH9NTfZ6n8H +h/QgOaoTL9GAakplAsdwYSLjBpgYMZOHIgkdM9ksRP8WsITChtZtxrCnBjR8bap ekPT/pM9zPZINEPxUlyJNwwTWjzTJP03+Yr07Q8v8fTJ5VWzAHlHtGQ/sI7yXA8 pzruTNre1MzxO3Ikjt2Q2e7CVXAp1b53BghgsppL9Bi7NK1R+vdWSs0B1Db/Gj alOkWUnhivTjRMX61RGDCQSVSEaX12EvJX7FooAsW3NFeZCgeZGWEa5ZMALLiBL4 GNASOOHju7ewlYjkyGIRxxAoc3C0w5dg1qlLiAFWToYwgDOcUpLRjU/7bzGiGvp8 RwIDAQAB -----END PUBLIC KEY-----

	<p>Title: agrirouter Connectivity-Platform Integration Guide Part 2</p> <p>Author: DKE-Data GmbH & Co. KG</p>	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

20.3. Application keys

The keys of the application can be found at the endpoint software overview when selecting an application and clicking on edit.

Edit Endpoint software

Security

*Redirect URL:

*Public key:

```
----BEGIN PUBLIC KEY----
MIIBIjANBgkqhkiG9w0BAQEFAOCQ8AMiIBCgKCAQEAY9ExOdRkATAEzaUh6zA
Pdm61gJstqwxgfUyRQo8GN35L+l7OHiqNlUsuyFt5lh8lmBnby8SjNM/Vakcri
1I+/2z8dG3bRFNvcPmC7VU9Dl0t5MxwzE/w67YYvEuzpZ4NI2Td8pytQlkct1N3U
V+0RX8m01EsMS8DxOAVCmQwlyY+oY25zR0EVoRCktFUVtZquvCqssT9yu82RKk
QIQRRRJUAR+n82SQHlwKkVVsSkJuqZJMLBwuT0fyBzf2uRmNbF23mqX+ZxY4H
```

Private key:

```
----BEGIN PRIVATE KEY----
MIIEvwIBADANBgkqhkiG9w0BAQEFAASCBKkwggSlAgEAAoIBAQDL38TE51GQBNoT
NpSHRMa92brWAmr2rDGAW5itCjwY3fkv4vs4eKoou5K7IW3mWGfwIYGduvLxKM0z
9VqRyuLUj7/bPx0bdtF829w+YLVT0XS3zHDMDT/Dru18S70mg0jZN3ynk1CW
RxPU3dRX7RFyufTUSwxLwPE4BUKZDAJj6hjbhNHQRWhEKS0VRW1mq68KqyyxP3
K7zZEaRCJBFFEIQD9H6fZJAc/AqRVaxKQISpkkawqHC5PR/lHN/a5GY1sXbeaB
```

⚠ Please copy and save the private key, as it will no longer be available! [\[copy\]](#)

Figure 45 Private and public key of an application in agrirouter

Only the public key will be stored in the agrirouter, the private key has to be stored by your app in a secure way.

 DKE DATA	Title: agrirouter Connectivity-Platform Integration Guide Part 2	
	Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

20.4. Endpoint certificate

The endpoint certificate is used to encrypt the communication with the agrirouter. It is delivered with the onboarding request.

```
{
  "deviceAlternateId": "6f1d952b-538e-4269-94b7-02bf51e83413",
  "capabilityAlternateId": "81ce3fd5-2f70-4270-ad15-1689ab6971bf",
  "sensorAlternateId": "aed40673-8e32-4f10-8cc8-3db2b58ed1bd",
  "connectionCriteria": {
    "gatewayId": "3",
    "measures": "https://dke-
qa.eul.cp.iot.sap/iot/gateway/rest/measures/6f1d952b-538e-4269-94b7-
02bf51e83413",
    "commands": "https://dke-
qa.eul.cp.iot.sap/iot/gateway/rest/commands/6f1d952b-538e-4269-94b7-
02bf51e83413"
  },
  "authentication": {
    "type": "PEM",
    "secret": "yY2uU1vV8aA1yY8uU1vV1cC",
    "certificate": "-----BEGIN ENCRYPTED PRIVATE KEY-----
\nMIIE6zAdBgoqhkiG9w0BDA\nnVD8E3qSEsvWS1Z93XPji\nn-----END ENCRYPTED PRIVATE
KEY-----
\n-----BEGIN CERTIFICATE-----
\nMIIEPzCCAyegAwIBAgIOAIjM....sV4DpbNKJ1Hut6000kzGCI+gsE=\n-----END
CERTIFICATE-----\n"
  }
}
```

 DKE DATA	Title: agrirouter Connectivity-Platform Integration Guide Part 2 Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

21. Usage Metrics

agrirouter collects and measures all information (event types) regarding data exchange in real time. For each message transported through agrirouter, it includes the following parameters:

column	type	Comment
id	UUID	The ID of the message
dh_message_id	UUID	The agrirouter internal ID of this message
account_id	UUID	The agrirouter account id, in which this message was sent
endpoint_id	UUID	The endpoint ID of the endpoint, for which the event happened
tech_msg_type_id	String	The technical message type of this message
event_type	Enum	The current status of the message; see below
volume	Integer	The total volume of the encoded message
buffer_duration	Integater	How long did the message persist in the feed, until it was delivered or deleted
month	Integer	
day	Integer	
hour	Integer	
year	Integer	
created_on	TimeStamp	When was this message delivered to the feed?

The event type is one of the following:

Event	Description
SENT	Message was sent by specified endpoint
ERROR	Message was returned because of an error (e. g. missing team context ID)
DELIVERED	Message was delivered to specified endpoint (but not yet confirmed)
CONFIRMED	Message was confirmed by specified endpoint
DELETED	Message was deleted by specified endpoint
OFFBOARDED	Message was deleted, because the specified endpoint was deleted / offboarded
TIMEOUT	Message was deleted because of a timeout (4 weeks rule)

Sent and Error mark events of the inbox, all the other events are events of feed or outbox.

The aggregated result will have the following structure:

column	type	Comment
app_provider_id	uuid	derived from the endpoint ID
app_id	uuid	derived from endpoint ID; this is the field over which the aggregate will be calculated
account_id	uuid	taken directly from raw data
endpoint_id	uuid	taken directly from raw data
event_type	character varying(15)	taken directly from raw data; second field over which the aggregate will be calculated
total_messages	integer	number of messages in aggregate

	<p>Title: agrirouter Connectivity-Platform Integration Guide Part 2</p> <p>Author: DKE-Data GmbH & Co. KG</p>	
Identification: agrirouter Integration Guide Part 2		Confidence: external
Status: final	Version: 1.3	September 4, 2018

total_volume	integer	total data volume in aggregate
average_duration	integer	average buffer time of aggregate
month	integer	
day	integer	third key field over which the aggregate will be calculated
hour	integer	
year	integer	
created_on	timestamp with time zone	

In any agrirouter developer account (account management -> export metrics), these usage statistics can be exported for your applications.

Currently, all messages delivered to an endpoints feed are relevant for billing; unimportant, if those were delivered or deleted. This might change in the future, all relevant information to change the billing mechanism are collected.
As changing the billing mechanism will impact app providers costs, this would be communicated and discussed in advance.

 DKE DATA	Title: agrirouter Connectivity-Platform Integration Guide Part 2 Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

22. General Certification process

22.1. General

Every Application needs to be certified to be onboardable to the agrirouter. The certification makes sure, that the communication protocol of the agrirouter was successfully implemented and the application is able to communicate with the agrirouter.

A certification is needed any time, the app changes its capabilities.

22.2. Out of focus

The certification will not check, if an application “understands” the data sent over agrirouter, it will only check, if the application can send and receive data marked as such Technical message types, that were selected in the Certification Version.

22.3. Certification institutes

DKE and the agrirouter support team will release a list of certified partners, that are able and allowed to do the certification process. Please refer to <https://my-agrirouter.com/support/>

22.4. Tests

22.4.1. For Farming Software and Telemetry platforms

The test for farming software and telemetry platforms will check the following workflows:

- onboarding of a Farming Software or Telemetry Platform
- sending messages
- receiving messages

22.4.2. For Communication Units

The test for communication units will check the following workflows:

- Onboarding
- Capabilities
- Receive data
 - Respecting capabilities
- Send data
 - Respecting capabilities
- Reconnection: If a CU loses its connection to the agrirouter, it will be tested, that the communication will be reestablished, once the connection to the agrirouter is reestablished.

	<p>Title: agrirouter Connectivity-Platform Integration Guide Part 2</p> <p>Author: DKE-Data GmbH & Co. KG</p>	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

23. Annex A (Glossary)

#	Concept	Description
1.	Application	In the agrirouter context, an application is a product which can connect to the agrirouter. The applications are not running on the agrirouter, but they connect to the agrirouter over the network.
2.	Application Instance	An application instance is the communication partner for an endpoint in an agrirouter account. An application instance represents a single software product or user account in a cloud software.
3.	Communication Unit (CU)	An application acts as a communication unit when <ul style="list-style-type: none"> Its main purpose is to provide data collected by sensor systems like GPS or machine sensors It manages the agrirouter communication for these machines. It sends machine data to agrirouter, and receives and processes messages addressed to machines
4.	agrirouter Account	An agrirouter account is a logical space on the agrirouter, that is used by one person or company. (e.g. a farm or contractor)
5.	agrirouter Message Format	An agrirouter command consist of a agrirouter command header (also called envelope) and the content (sometimes also called payload). The header contains fields such as the sender ID, the list of recipients, and the technical message type.
6.	agrirouter routings	agrirouter users can maintain routings to specify what data may go from which sender/source to which recipient. Without a corresponding routing, no data is transferred.
7.	Capabilities	Capabilities are a list of technical message types, an app instance supports to send or receive
8.	Command	A command is sent to the agrirouter from any endpoint. It consists of a header and an agrirouter control message (e.g. a teamset Message). The most common type of command includes a message of a specific message type, that shall be forwarded to one or more endpoints
9.	Directly Connected Endpoints	A directly connected endpoint is any application, that communicates with its own endpoint at the agrirouter.
10.	EFDI	<p>EFDI (Extended FMIS Data Interchange) is the working title of an upcoming standard of the Agricultural Industry Electronics Foundation (AEF).</p> <p>It defines message formats for network communication between machines and FMIS as well as between different FMIS, including the transfer of live telemetry data from machines.</p> <p>The information that are transferred is based on ISOBUS concepts. It uses, for example, ISOBUS DDLs, and even the message structures are based on ISOXML elements. As the technical serialization format, EFDI does not use XML, but Google protocol buffers.</p> <p>The agrirouter uses EFDI messages as the payload format for some message types, for example for live telemetry data and device descriptions.</p> <p>At the time of this writing, EFDI is still in an early stage. The working group is just finalizing the internal definition of the first message types (live telemetry data, device descriptions).</p>
11.	Endpoint	In the agrirouter context, an endpoint is an addressable entity in an agrirouter account, which is the communication entry point for a single app instance or machine.
12.	Endpoint Descriptions	The endpoint description contains detail information about an endpoint. Important part of this information is the list of the technical message types, the endpoint supports.
13.	Farm Management Information System (FMIS)	An application providing farm management functionality. An FMIS may, for example receive data from machines, send settings and tasks to machines, and exchange data with other applications.
14.	Information Type	A summarize of different Technical message types

	<p>Title: agrirouter Connectivity-Platform Integration Guide Part 2</p> <p>Author: DKE-Data GmbH & Co. KG</p>	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

15.	ISOBUS, ISO 11783, taskdata	<p>Wikipedia: "ISO 11783, known as Tractors and machinery for agriculture and forestry -- Serial control and communications data network (commonly referred to as "ISO Bus" or "ISOBUS") is a communication protocol for the agriculture industry based on the SAE J1939 protocol (which includes CANbus) ." (source: https://en.wikipedia.org/wiki/ISO_11783)</p> <p>ISO 11783-10 describes the file-based and batch-oriented interchange format between machines and FMIS, using XML (ISOMXML) and binaries with XML header data. This is called a taskdata file set, even though it can contain many other data (fields, products, crop, worker, and many more) and need not even contain a task.</p> <p>The ISOBUS standard specifies only the data but not how it is transferred between machines and FMIS.</p>
16.	ISOBUS DDI	<p>A DDI (literally: Data Dictionary Identifier) represents a device or process parameter in the ISOBUS norm. Over 500 DDIs are defined in ISO 11783-11, see http://dictionary.isobus.net/isobus/ .</p> <p>DDIs are used, for example, in device descriptions to describe properties, the supported settings, and the provided data of a machine.</p> <p>DDIs are also used to identify the data records in telemetry messages.</p> <p>agrirouter users can filter telemetry data for DDI Categories, that abstract all DDIs into 10 categories. A list of DDIs and their Category assignment can be found here:</p> <p>A filtering for specific DDIs could not be provided, as this would have been too complex for the end user.</p>
17.	Live Telemetry Messages	<p>A live telemetry message contains data points with data from the machines in one teamset. Each data point at least contains the time when it was logged. Additionally, each data point can have a geo-position. Each data point may contain many log entries. Each log entry contains the value of a specific parameter (DDI) of a specific component or function (device element) of one of the machines in the teamset.</p> <p>The user-defined agrirouter routings define which parameters of which machine may go to which recipient. In addition, recipients can subscribe for certain parameters they want to get (sender-independent). Recipients will receive those data, when they are sent to the public address.</p> <p>The agrirouter uses the routings and the subscriptions to determine the recipients, and the information each of them gets. To each recipient it delivers a filtered version of the messages, which contains only the allowed and subscribed parameters of the allowed machines.</p>
18.	Machines	<p>Machines are agricultural machinery, in the sense of a ISO11783 device.</p> <p>Machines are tractors, implements such as sprayers, or self-propelled machines like combine harvesters. A machine is described with machine description, which is conceptually based on ISO11783-10 device description.</p> <p>Machines are the sources of the data records in live telemetry messages which applications send via agrirouter.</p> <p>From an abstract view, machines are just sensor networks providing sensor data.</p>

	<p>Title: agrirouter Connectivity-Platform Integration Guide Part 2</p> <p>Author: DKE-Data GmbH & Co. KG</p>	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

19.	Machine Endpoint	<p>A machine endpoint represents one real-world machine in the context of an agrirouter account. The same real-world machine can have endpoints in several agrirouter accounts, but not more than one in the same account.</p> <p>A message can be addressed to a machine as the recipient. This tells the agrirouter to deliver the message to that CU, to which the machine is connected. If the addressed machine is currently not connected, the agrirouter puts the message in the machine's feed and delivers it as soon as some CU reports that the machine is now connected. Since machines can be attached to different CUs at different times (connected to different tractor, for example), it is not known in advance which CU that will be.</p> <p>The message itself will be received and processed by the CU, but addressing it to the machine makes sure that it goes to the right one.</p>
20.	Message	A message is an information or perhaps a request, that is sent from an endpoint to any other endpoint. A message is a possible payload of a command
21.	teamset	A teamset is a set of connected machines which work and move together and are connected to the same communication unit
22.	Virtual Communication Units	<p>A virtual communication unit is the equivalent of a communication unit for situations where the teamsets are not directly connected to the agrirouter. Instead they are connected to an external telemetry-enabled cloud service, which itself is connected to the agrirouter. Such a telemetry-enabled cloud application has its own mechanisms for connecting farming machines.</p> <p>For each farm, many machines are connected to the external cloud service, grouped in many teamsets. The cloud application, which makes these machines known on the agrirouter, also reports one virtual communication unit for each teamset.</p>

	Title: agrirouter Connectivity-Platform Integration Guide Part 2	
	Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

24. Annex B (Shortings)

As agrirouter uses multiple, sometimes quite long special terms, these terms are shortened in some of the graphics. Following a list of shorting's and the corresponding full term

Shorting	Full Term
[]	Shorting for a list
...	additional steps; not shown here
App	Application
ar	agrirouter
CAP	Capabilities
CU	Communication Unit
DDI	Data Dictionary Identifier; see ISOBUS DDI
DVC	Device Description/ Device
EP	Endpoint
IT	Information Type
MSG	Message
TMT	Technical Message Type
VCU	Virtual CU; Virtual Communication Unit

	Title: agrirouter Connectivity-Platform Integration Guide Part 2 Author: DKE-Data GmbH & Co. KG	
Identification: agrirouter Integration Guide Part 2	Confidence: external	
Status: final	Version: 1.3	September 4, 2018

25. Annex C (Message Codes and Handling)

The error and message codes are delivered in the github repository, see
<https://github.com/DKE-Data/agrirouter-interface-documentation/blob/develop//docs/SystemMessageCodes.adoc> .