H2020 Grant Agreement No. 857125

D3.2 Service Architecture Specification

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement no. 857125.

Deliverable no.	D3.2
Work package	WP3
Dissemination level	PU
Due date of deliverable	2020-04-01
Actual submission date	2020-04-30

Author(s)			
Partner	**First name**	**Last name**	**Email**
Fraunhofer	Stefan	Rilling	stefan.rilling@iais.fraunhofer.de
AC	Peter	Fröhlich	peter.fröhlich@agricircle.com
Contributor(s)			
Partner	**First name**	**Last name**	**Email**
AEF	Volker	Zippel	volker.zippel@claas.com
AEF	Stephan	Adam	adamstephan@johndeere.com
AEF	Alexander	Hammerschmidt	a.hammerschmidt@dke-data.com
AEF	Slawi	Stesny	slawi.stesny@agcocorp.com
AEF	Franz	Kraatz	franz.kraatz@krone.de
AEF	Hara	Spathi	Hara.spathi@aef-online.org
AC	Mallku	Caballero	mallku.caballero@agricircle.com

Ethics & Security Ass.	This deliverable is excluded from ethics & security assessment
Approval Date	-
Remarks	-

Gender-proofing	Gender-sensitive language Gender aspects in research not applicable to this deliverable
Proofing date	2020-04-01
Remarks	-

Version	**Date**	**Released by**	**Comments**	**Document status**
0.1	2020-03-03	Stefan Rilling	Initial version	Draft
0.2	2020-03-26	Franz Kraatz	New sections	Draft
0.3	2020-04-03	Stefan Rilling	Progress in multiple sections	Draft
0.4	2020-04-07	Stefan Rilling	Relation to the use cases	Draft

0.5	14.04.2020	A. Hammerschmidt	Add the glossary definition, security and different data formats chapter	Draft
0.6	21.04.2020	Stefan Rilling	Version for internal Review	Draft

Table of Content

List of Figures

List of Tables

Glossary of Terms

Agricultural Software	This can be an on-premise application, cloud solution or software running on a single device like a PC, tablet or smartphone. Agricultural software can receive or send data to provide extra services (documentation, evaluation, planning, analytics, etc.). Agricultural software can be connected to other systems or devices via internet or directly. FMIS (Farm Management Information Systems) is an example of agricultural software.
ATLAS app	An application capable of running in the ATLAS app engine.
ATLAS data services	ATLAS data services are corresponding to the OASIS definition “a mechanism to enable access to one or more capabilities, where the access is provided using a prescribed interface and is exercised consistent with constraints and policies as specified by the service description”[1].
ATLAS Network	The “ATLAS Network” describes a set of rules, protocols, flows and services which enable participants of the network in the agronomical sector to exchange information and data. The network consists of participants with different roles. Those providing information or services and those consuming information and services or even participants covering both roles.
ATLAS Resource Identifier	An Identifier to identify a unique resource in the ATLAS Network. The Identifier could be a combination of the participant id and the unique resource id in the participant system.
Capabilities	Capabilities are a list of technical message types, services or commands which a communication unit, an agricultural software or a data platform supports
Capabilities Service	Every endpoint in ATLAS Network has to report his capabilities of sending and receiving messages after onboarding and when the capabilities change (e.g. an

[1] https://www.oasis-open.org/committees/tc_home.php?wg_abbrev=soa-rm

	application functionality is extended by a new module or data format).
Command	A command can be sent to the ATLAS Network from any endpoint. The most common type of command includes a message of a specific message type, that shall be forwarded to one or more endpoints.
Communication Units	These are mobile software applications, mostly running on embedded systems or mobile devices to make a connection to other systems. CU's can communicate with the Internet in a variety of ways. CU's can buffer data to compensate for lost connections. Machine combinations can communicate with other systems via one communication unit.
Data Consumer	The Data Consumer is a participant in the ATLAS Network which mainly takes information of the Data Providers and computes those for further usage.
Data Owner	The Data Owner is the legal owner of any information flowing through the network. An information can also be person related.
Data Platform	This is a software solution from a machine manufactures or other providers. A Data Platform can be an on-premise or cloud solution. It interacts between machines, sensors (SR) and Agricultural software (AS). Machines and sensors (SR) communicate via Communication Units (CU) to the Data platforms. A Data Platform can manage telemetry, agronomic or business data. A Data platform can provide additional services (documentation, device service, fleet management, etc.)
Data Provider	The Data Provider is a participant in the ATLAS Network which is mainly providing information into the network for further use.
EFDI	EFDI (Extended FMIS Data Interchange) is the working title of an upcoming standard of the Agricultural Industry Electronics Foundation (AEF). It defines message formats for network communication between machines and FMIS as well as between different

	FMIS, including the transfer of live telemetric data from machines.
Endpoint	An endpoint is an addressable entity in the ATLAS Network, which is the communication entry point for a single application instance or machine.
Endpoint Descriptions	The endpoint description contains detailed information about an endpoint. Important part of this information is the list of the technical message types, services or commands the endpoint supports.
Information Type	A summary of different Technical message types.
ISOBUS DDI	A DDI (literally: Data Dictionary Identifier) represents a device or process parameter in the ISOBUS norm. Over 600 DDIs are defined in ISO 11783-11, see http://dictionary.isobus.net/isobus/. DDIs are used, for example, in device descriptions to describe properties, the supported settings, and the provided data of a machine. DDIs are also used to identify the data records in telemetry messages.
ISOBUS, ISO 11783, taskdata	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) ISO 11783-10 describes the file-based and batch-oriented interchange format between machines and FMIS, using XML (ISOXML) 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. The ISOBUS standard specifies only the data but not how it is exchanged between machines and FMIS.
IT Service	A definition according to ITIL®: "A means of delivering value to customers by facilitating outcomes customers want to achieve without the ownership of specific costs and risks. The term ‘service’ is sometimes used as a synonym for core service, IT

	service or service package." A service is understood as a functional service of a subsystem, with which a contribution to the function of the overall system.
Message	A message is an information or perhaps a request, that is sent from an endpoint to any other endpoint. A message can also be the payload of a command.
Network Participant	A network participant in the ATLAS context represents an organization (service consumer or provider) which takes part in the network in order to digitally exchange agricultural data.
Onboarding Service	The onboarding is needed in order to connect the communication units, agricultural software or data platforms and create the data exchange endpoints.
Registry Service	A trusted directory of ATLAS Participants and the ATLAS Data Services they offer, accessible via APIs.
Sensor	This is a device which detects or measures a physical property and records, indicates, or otherwise responds to it. The specific input could be light, heat, motion, moisture, pressure, or any one of a great number of other environmental phenomena. The output is generally a signal that is converted for further processing.
Service Provider	A service provider provides IT services that consist of a combination of experts, processes and technologies. The type and scope of IT services are defined by a service level agreement (SLA).
SLA	A service-level agreement (SLA) is a commitment between a service provider and a client. Particular aspects of the service – quality, availability, responsibilities – are agreed between the service provider and the service user.
Technical message types	The technical message type describes the type of the content of an ATLAS message, such as an EFDI live telemetry message, or a JPEG image. Each technical message type must be assigned to an information type, which represents its meaning and purpose.

Abbreviations and Acronyms

A	Assumptions
AC	AgriCircle
ADS	ATLAS data service
AEF	Agricultural Industry Electronics Foundation
API	Application programming interface
AR	Augment Reality
AS	Agricultural software
ATLAS	Agricultural Interoperability and Analysis System
AVI	Audio Video Interleave
CAN	Controller Area Network
CRUD	Create, read, update, delete
CU	Communication Unit
DDI	Data Dictionary Identifier
EFDI	Extended Farm Management Data Interface
EU	European Union
FMIS	Farm Management Information System
FR	Functional requirements
GDPR	General Data Protection Regulation
GPS	Global Position System
HTTPS	Hypertext Transfer Protocol Secure
IAN	Interoperable Apps network
IDP	Identity Provider
IPN	Interoperable platform network
ISO	International Organization for Standardization
ISV	Independent Software Vendor
IT	Information Technology
JPEG	Joint Photographic Experts Group
LTP	Linked Third Party
MPEG	Moving Picture Experts Group
MQTT	Message Queuing Telemetry Transport
NFR	Non functional requirements
NGO	Non-Governmental Organization
NIR	Near-infrared
NPK	Nitrogen, Phosphorus, Potassium
OASIS	Organization for the Advancement of Structured Information Standards

OAuth	Open Authorization
OEM	Original Equipment Manufacturer
OS	Operating system
PC	Personal Computer
PDF	Portable Document Format
REST	Representational state transfer
SDK	Software development kit
SLA	Service Layer Agreement
SME	Small and medium-sized enterprise
SR	Sensor
TIM	Tractor Implement Management
TLS	Transport Layer Security
VR	Virtual reality
WMV	Windows Media Video
XML	Extensible Markup Language

1 Introduction

The aim of this deliverable is the high-level description of the ATLAS interoperability architecture. This architecture and its implementations will form the foundation of the various technical solutions developed in ATLAS. It will allow the interconnection of sensors, agricultural machines and data services in a reliable and secure way. A reference implementation of this architecture will be created within the project and evaluated along a multitude of real-world use cases.

A dedicated team of core architects from beneficiaries AC, Fraunhofer, AEF and AEF's LTPs is responsible for the creation of the reference architecture described in this document. Based on lessons learned from the on-going work in ATLAS, further discussions inside the member organizations and the reference implementation, the architecture will be continuously evolved as needed.

2 Document overview

The foundation for the architecture specification is the outcome of deliverable "D3.1 – Requirement analysis". The work in this deliverable started with the collection of architecture drivers ("3.1 Architecture drivers") emerging from the stakeholder groups of the ATLAS consortium: farmers, machinery manufacturers and data service providers. In the process of the work conducted for this deliverable, the requirements and architecture drivers were further refined and viewed from a software architecture and development process.

The high-level architecture derived from the drivers consists of two parts, (i) the ATLAS Network and (ii) the ATLAS AppEngine, which complement each other. Chapter "3.2 ATLAS Network" and "3.3 ATLAS AppEngine - On-premise Computing Platform" provide insights about both approaches. Additionally, this section includes chapters about central components identified ("3.4 ATLAS central components"), version handling for ATLAS ("3.5 Version handling for capabilities in the architecture"), security ("3.6 Integration of security in the architecture"), data forwarding ("3.7 Handling data forwarding in the architecture ") and GDPR ("3.8 Handling GDPR in the architecture").

Chapter 4 provides one example use-case for each of the architecture approaches. "4.1 Fertilization Use Case" explains how the ATLAS AppEngine together with App's can be adapted and chapter "4.2 Platform independent cross brand machine tracking" adapts the ATLAS Network architecture for integration scenarios.

The final chapter "5 Conclusion" provides lessons learned and an outlook, how the architecture work will continue.

3 ATLAS architecture

The technical concept of ATLAS to achieve the interoperability of sensors, agricultural machines and data processing services is based on two approaches which complement each other:

- ATLAS Network: The ATLAS Network aims for a service-oriented architecture which enables the connection of sensor-networks, data platforms and data processing services in a consistent and easy way.
- ATLAS AppEngine: The ATLAS AppEngine is a self-contained computing reference platform on which apps may be easily installed and executed. This reference platform should be able to be adopted easily for different use-cases (e.g. on-machine or on-premise).

With the two approaches ATLAS can connect different sensors, machines and actors locally (e.g. on the machine or on the farm) and on demand. It delivers data to data consumers, where data can be stored, processed or be used by evaluation services. Furthermore, there will be frameworks in the network for knowledge exchange and decision support as well as the infrastructure to store and process data, granting access to single users, user communities, SMEs, NGOs and other stakeholders to drive know-how, efficiency and sustainability in the agricultural sector.

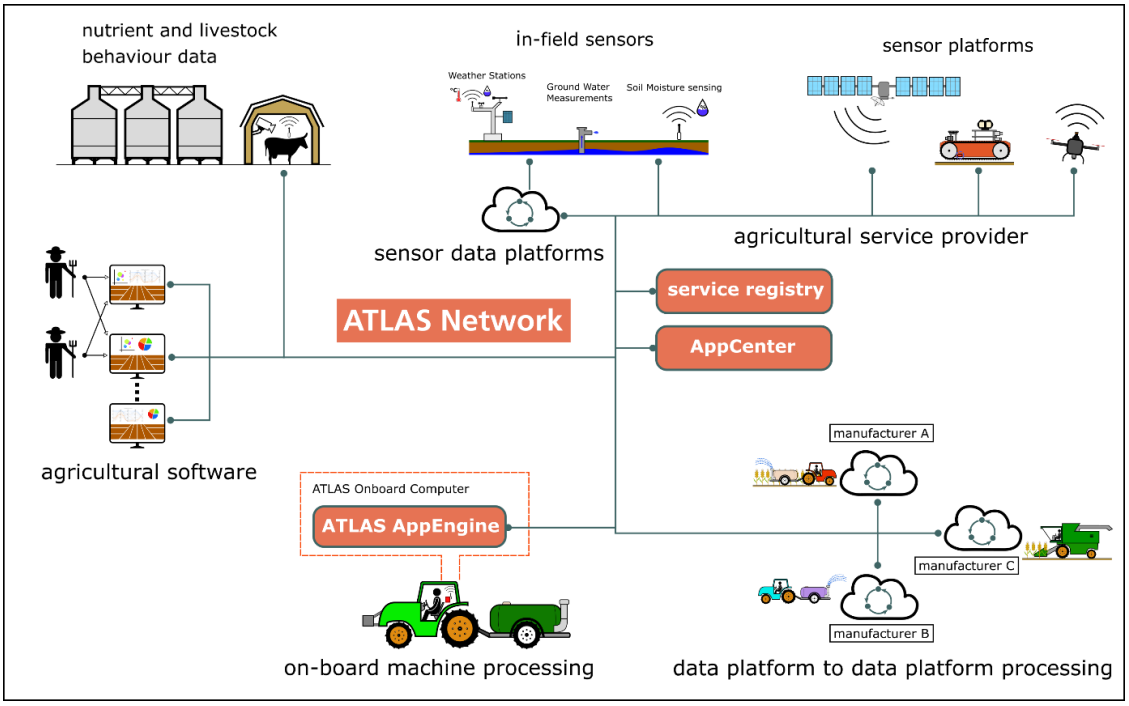

Figure 1: The ATLAS Network overview.

Figure 1 shows how the different participants and technologies can interconnect via the ATLAS Network.

3.1 Architecture drivers

The following section contains a list of assumptions, functional and non-functional requirements which have been identified as architecture drivers for the ATLAS architecture.

Table 1: Assumptions

No.	Description
A-001	GDPR conformity has to be taken into consideration.
A-002	Data exchange will be secured on transport layer.
A-003	Solution will support (occasional) offline scenarios, where machines act in areas without permanent internet connection.
A-004	Data Sources will provide the temporal resolution to the dataset as defined in ATLAS.
A-005	Source systems will be able to map their User/Access rights management to the ATLAS Access management.
A-006	Endpoint metrics which could be used for later invoicing scenarios and central billing systems are out of scope.
A-007	Message level encryption lies in the responsibility of the participating service if needed.
A-008	Code of Conduct for Ag data sharing based on contractual agreements has to be taken into consideration.[2]
A-009	Assumption is that the technical (architectural) solution will probably need a logically central registry service to be able to identify any participants in the network.
A-010	A mixed fleet is a group of equipment, independently from the owner or brand.
A-011	Consumer systems have to provide basic meta information for billing.
A-012	Only "certified" and "tested" participants can take part in ATLAS Network.
A-013	ATLAS central components will not store any transactional data.
A-014	Data access from implements to machines is only available through the current open standards like for example ISOBUS. Specific use cases will bring us in the need to discuss the currently available solutions.
A-015	ATLAS sets up a certification body(ies) for ATLAS “Label” governance, data platform certification (maintain self-certification tools), AppEngine certification, app certifications (maintain self-certification tools where relevant, and more where necessary). ATLAS defines a binding charter which lists conditions, rights and obligations for every participant (machinery vendors and ISVs) in the ATLAS Network. E.g., the obligation for any ATLAS participant to enable the “pairing" and relevant API access with any other ATLAS participant for relevant capabilities in a non-discriminatory manner.
A-016	All ATLAS project members (direct and indirect) commit to adhere to the ATLAS Charter (obtain the ATLAS Label) in the course of the project.
A-017	ATLAS defines the concept of AppEngine: the specifications of a “box” and operating system capable of supporting agricultural apps execution to automate sequences of operations as well as to enable low-latency real-time operations.

[2]https://cema-agri.org/publications/19-brochures-publications/37-eu-code-of-conduct-on-agricultural-data-sharing

A-018	For quality assurance it will be needed to define specific Non-Functional Requirements (NFRs) for specific use cases in terms of latency, computing time, availability etc.
A-019	Each qualified request from a registered ATLAS participant needs to get a qualified answer. Some requests may require pairing first.
A-020	ATLAS needs a central registry (one truth only), central certification (quality assurance) and documentation on a certain level.
A-021	Data exchange in the ATLAS Network should be non-discriminatory at all times.

Table 2: Functional requirements

No.	Description
FR-001	ATLAS shall provide a functionality that allows the service provider to propagate the service availability based on region and/or country.
FR-002	ATLAS shall provide a functionality that allows the service provider to propagate availability of object attributes based on region and/or country.
FR-003	Every participant should be able to limit and/or suspend connectivity to another endpoint in order to secure own matters justified on justifiable reasons. This applies to any possible centralized service in ATLAS as well.
FR-004	The architecture must respect multiple communication patterns related to the use cases.
FR-005	Resources in the ATLAS Network which are crucial for the functionality of the network and the capabilities of the participants need to be uniquely identifiable, but the uniqueness is not limited to only those resources.

Table 3: Non functional requirements

No.	Description
NFR-001	Processes to assess system security needs to be established.
NFR-002	The architecture needs to consider that specific use cases need faster or slower response times in order to fulfill the use case. It must be assured that specific use cases are respected.
NFR-003	Services for support and diagnosis tasks should be foreseen.
NFR-004	The architecture needs to consider eventual inconsistency as specific participants may not be available at a given time.
NFR-005	A transparent request and response pattern needs to be used to ensure clear service states.
NFR-006	At least parts of the architecture and APIs and their documentation shall reach a maturity level to be able to be put into a standardization process.
NFR-007	All ATLAS architecture and documentation need to be freely available for further development and research. All contributors agree that the final product, pattern, system or solution is open source.
NFR-008	All solutions or architectures should be independent from any programming language. This ensure the possibility for a maximum of participants to be able to take part in the ATLAS Network. Any technological decision should not be relying on specific software or hardware product.

3.2 ATLAS Network

The ATLAS Network shall overcome the missing interoperability currently met in the agricultural sector. To overcome this, the network needs to meet the requirements of the farmers, OEMs, software providers and other stakeholders. In order to do so ATLAS is envisioning a kind of service mesh architecture (see Figure 2) where every participant stays autonomous and is responsible for implementing and providing the services offered. With this architecture central components are kept to a minimum and only a central service registry and app centre services are needed as first place to go.

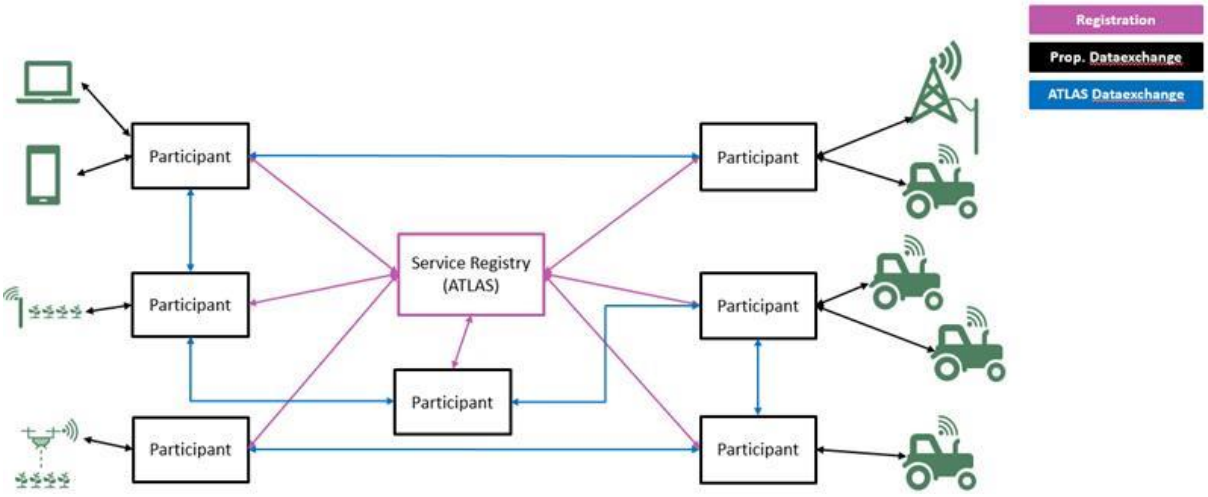

Figure 2: ATLAS Service Mesh Network

3.2.1 ATLAS service setup

To be able to take part in the ATLAS Network each participant needs to register its services in the ATLAS Registry so that it can be found and used by other participants of the ATLAS Network. In the ATLAS Network each participant stays autonomous and is responsible for implementing and providing the services offered in the network. This also includes security and consent management. Figure 3 shows a high-level architecture of two participants registering their services in the network. Each participant offers an own software (OEM/agricultural Software) which accesses proprietary backend services (agricultural software services/ OEM services) and via those the needed data. The access to the participant software, backend services as well as the ATLAS data services (ADS) is secured via an identity provider (IDP) owned by the participant. Additionally, it is possible for every participant's service following the ATLAS rules to join other hosted ATLAS compatible registries. User consent which needs to be captured from every service in order to be able to secure the agreed data access and transfer will be managed in a consent management system owned by the participant.

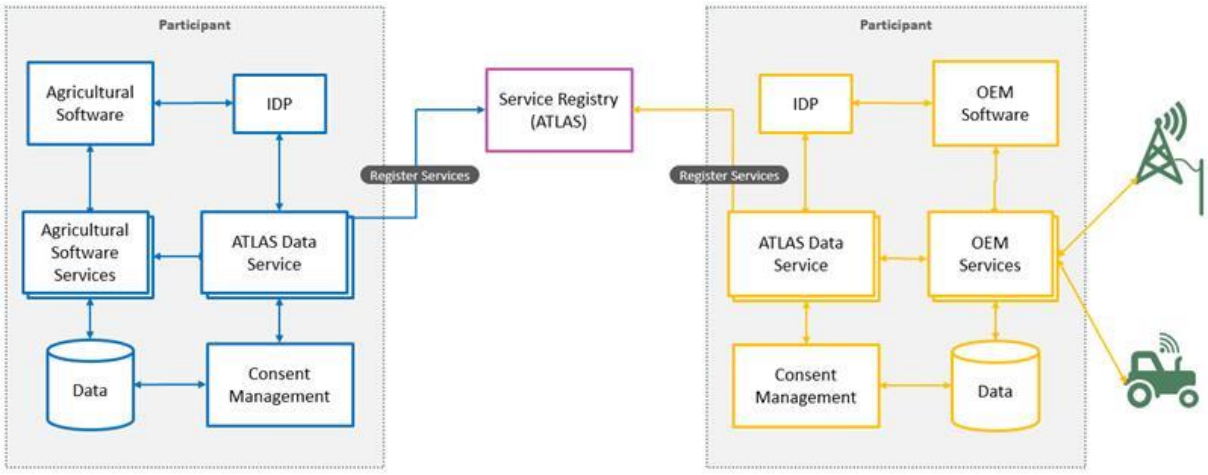

Figure 3: High Level Service Architecture

- **Identity Provider (IDP)**: The identity provider acts as a central service in the participant setup. Its function is the storage of identities and the authentication of identities for services upon request.
- **Participant software**: Each participant in the network can have its own software solution for the services it offers. If the participant has an own software and is connected to another participant, he can use and/or show the data provided by the other participant in his own software. (Agricultural Software; OEM Software)
- **Participant service**: Each participant can have own proprietary services. These services can function as backend services for own software solutions or hardware on the field etc. (Agricultural Software Service; OEM Service)
- **Consent Management**: A consent management system is used to gather and track the user consent. The ATLAS Network will not provide a solution for capturing and tracking the consent of a user. It is assumed that each participant has its own consent management solution in place, which can be used to gather and track user consent in the ATLAS context.
- **Data**: Data collected by services and stored by the participant for further usage.
- **Service Registry**: A centralized registry service where ATLAS data services can register in order to be found and used. For further information see section 3.4.1.

3.2.2 ATLAS data services

The instances of the ATLAS data service pattern will be the central component of the ATLAS interconnectivity network. The following describe the generic setup of the ATLAS data service (service template) and is shown in Figure 4.

The ATLAS service template comprises the following five layers:

Defined by ATLAS

- **Service Interface**: The service interface is responsible to establish the connection to the corresponding partner. ATLAS will support multiple communication channel types like REST, MQTT, WebSocket, etc.
- **Service Layer Mapping**: The service layer mapping defines a generic exchange protocol with the associated generic data exchange format. The exchange protocol defines a generic set of different basic functions for data exchange (e.g. GET, ACK, SHOW), which in turn are used in different combinations for the realization of scenarios (e.g. file exchange or data stream). Based on these scenarios, the various use cases of the next layer can be mapped, and a reusability of existing scenarios is always guaranteed. New use cases can be created very easily and implemented in existing systems with little effort.
- **Supported Format**: Specific data formats are provided so that the data exchange can be adapted to the specific requirements of the individual use cases. Building on the generic protocol / data exchange format, various data formats such as ISOXML, PDF or JPEG are integrated in this layer. In addition, it is very easy to integrate new data formats for new use cases or more specific requirements.

Defined by ATLAS participant (optional)

- **Data Format Conversion**: Since only specified data formats can be supported in ATLAS or only a specific version of the data format is used, there is the possibility for every participant to convert the data format supported in ATLAS to its specific data format. In the end, it is up to each participant to decide whether to use the data format offered by ATLAS or to convert the data into their own format.
- **Proprietary/Specific Data Format**: Each participant in the ATLAS Network can use their own proprietary data format within their own system. On the other hand, he is also responsible for converting the proprietary format into the ATLAS data format.

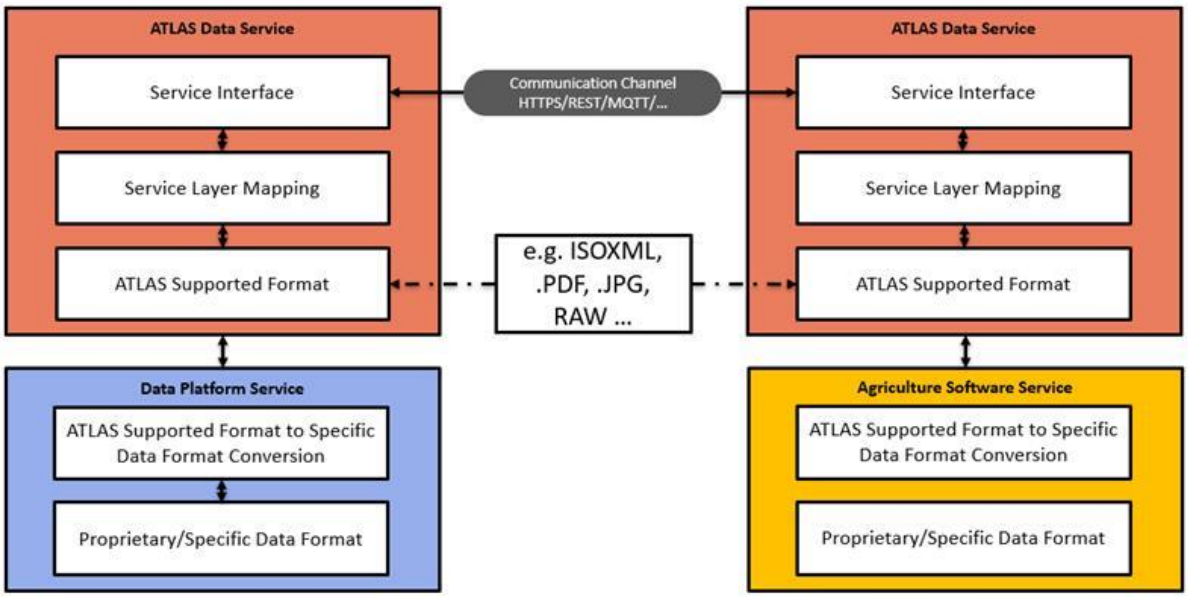

Figure 4: ATLAS Data Service - Generic View

3.2.3 Integration of different data formats

Currently there are various standardised, non-standardised, proprietary and other manufacturer-dependent own data formats on the agricultural market available.

To be able to exchange the different standardised, non-standardised, proprietary and other manufacturer-dependent own data formats, the following technical message type and information type concept will be recommended.

3.2.3.1 The technical message type

The technical message type describes the type of the content of an ATLAS message, for example an EFDI live telemetry message, or a JPEG image. Each technical message type must be assigned to an information type, which represents its meaning and purpose.

The technical message types are the basis for the capabilities listed in an endpoint description, and for the subscriptions of endpoints (endpoints subscribe to a technical message type, not the information type - e. g. if an application subscribes to jpeg image, it will subscribe to jpeg, not to image).

The technical messages types need to be assigned to information types. A technical message type can be assigned to exactly one information type.

There shall be a way to create new technical message types with following requirements:

Technical message type:

- Identifier = an unique identifier
- Description
- Information type

3.2.3.2 The information message type

To specify what kind of information data may be transported. The information type refers to the meaning or purpose, and not to the technical data format. Each information type has one or more technical message type assigned. The information type is a summary of different technical message types.

The information types provided will be used by the End-User, where data exchange is created on the level of information types.

There shall be a way to create new information types with following requirements:

Information message type:

- Identifier = an unique identifier
- Title and description

3.2.3.3 Live telemetry messages

A live telemetry message contains data points with data from the machines in one team set. 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 team set.

DDIs (Data Dictionary Identifier) are used, for example, in device descriptions to describe properties, the supported settings, and the provided data of a machine. DDIs are also used to identify the data records in telemetry messages.

A filtering for specific DDIs could not be recommend, as this would have been too complex for the end user. ATLAS Network end users can filter telemetry data for DDI categories that abstract all DDIs into following categories for example:

Table 4: Different categories for live telemetry

Live telemetry category	DDIs (Data Dictionary Identifier)[3]
Machine data: Data related to the machine characteristics (not process relevant).	54 Minimum Tillage Depth 55 Maximum Tillage Depth 59 Minimum Seeding Depth 60 Maximum Seeding Depth ….
Application Data: Data related to what is applied to the field (e.g. fertiliser, seeds, plant protection, dry matter, etc.).	1 Setpoint Volume Per Area Application Rate 2 Actual Volume Per Area Application Rate 3 Default Volume Per Area Application Rate 4 Minimum Volume Per Area Application Rate …
Guidance and geo data Data related to geographical and guidance information.	505 Tramline Control Level 506 Setpoint Tramline Control Level 507 Tramline Sequence Number …
GPS geo position GPS geo position (north and east coordinates) where the telemetry data was measured or logged.	… Will be defined during the ATLAS project …
General work data Task and lifetime counter or average values (counters that are not relevant for application and/or yield).	… Will be defined during the ATLAS project …
Fuel and exhaust fluid consumption data Data related to the consumption of fuel and exhaust fluid (DEF) (total energy consumption).	… Will be defined during the ATLAS project …
Crop and yield data Properties of harvested material.	… Will be defined during the ATLAS project
Process data Data related to the main working process of the machine.	… Will be defined during the ATLAS project …
….	…

[3] http://dictionary.isobus.net/isobus/

3.2.3.4 Maintain information and technical message types

In order to ensure extensibility of the ATLAS interface in terms of new message formats to be supported, it must be possible to define and add new information types and technical message types in an easy way, provided that the new types do not require any special logic for data exchange or filtering.

Such new information and technical message types should be added by the source system administrators, using a dedicated administration user interface or one other way.

However, only message formats that have a high significance and acceptance in the industry and are therefore helpful for the exchange of agricultural data are added.

Table 5: Example of technical and information message types

Technical Message Type	Information Type	Description
iso-11783-10 taskdata	TaskData	ISO compliant set of Taskdata
iso-11783-10 device_description	Telemetry	ISO compliant set of device descriptions connected
iso-11783-10 time_log	Telemetry	ISO compliant life telemetry data based on the device description
bmp	Image	A Bitmap File
jpeg	Image	A JPEG File
png	Image	A Portable Network Graphics File
shape	Shape	Shape dataset e.g. as a ZIP-File
pdf	PDF	A PDF document
video avi	Video	An AVI Video

Technical Message Type	Information Type	Description
video mp4	Video	A MPEG4 Video
video wmv	Video	A WMV Video
..	..	..

3.2.4 Data service pairing

In the architecture are two different levels for pairing two systems together. The first stage of the connection takes place at the technical level. Two systems establish a technically verified and secure connection for data exchange. Building on this, the users of the systems establish a legal release for their data to the users of other systems. This means only open data can be obtained freely from all users from other systems. All other data are subject to access restrictions and must be approved by the owner for third-party access. In the following, these two levels are examined in more detail.

3.2.4.1 Pairing between services

In order for two systems to establish a connection to each other for data exchange, several steps are required. Initially, the two systems are not known to each other and there is no further descriptive information. Only the ATLAS Registry is known to the systems, since they have also entered themselves here. This registry is therefore also the central point of contact for connecting the individual systems to one another.

First, the requesting system can get all systems that provide a corresponding service with a compatible version by searching the ATLAS Registry. From this, the requesting system selects the desired entry and uses the information stored here to initiate the connection to the other system. During the establishment of the connection, further information necessary for the connection is exchanged between the two systems and configurations are defined. This ensures that both systems exchange the data in the same way on a secure transmission path.

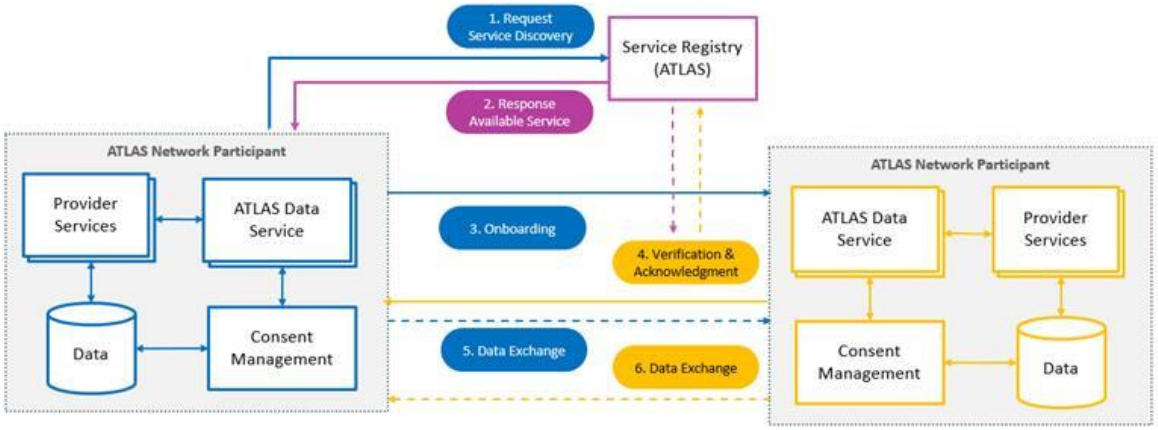

Figure 5: Pairing steps for connection establishment between systems

Figure 5 shows the individual steps from searching the registry, establishing the connection and the final connection for the secure data exchange.

3.2.4.2 Pairing for data exchange

After the systems have established a connection to each other for the data exchange, the actual data access is still missing. For data access a distinction between open source and user-related data must be made. Because with the open source data no further access release is required. Once the connection between the two systems has been established, the open source data can be obtained from third parties without any restrictions.

The situation is different with user-related data. Here the consuming user must first request for the provision to the data. The request for data access is communicated to the owner of the data by his system. Here the owner of the data has several options.

1. Data access is denied
2. Data access is set up for a limited time (can be revoked at any time)
3. Data access is set up unlimited (can be revoked at any time)

For this access, the consuming user receives a token, which must now be given for each request for the data. The token links the access rights of the requesting user with the data. Using the token, access to the data can also be withdrawn by the owner at any time by setting the token to invalid. Decoupling is carried out so that there is a defined withdrawal of access rights in the consuming systems.

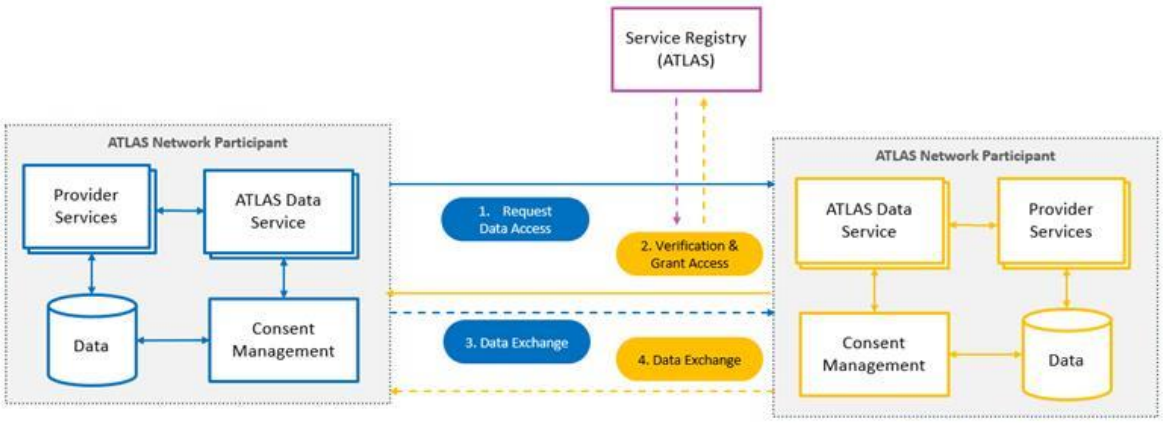

Figure 6: Pairing steps for data access between data sources and users

Figure 6 shows the described process between the systems for the release of data to third parties by the owner of the data.

3.3 ATLAS AppEngine - On-premise Computing Platform

The ATLAS architecture identifies the need for a self-contained computing reference platform, the AppEngine, on which apps may be easily installed and executed. An app is a fit for purpose and self-contained software module that may interact, via the AppEngine SDK, with machinery, sensors, cloud service and apps executing on other nearby AppEngines.

The AppEngine aims to provide a platform that supports apps to function with little or no internet connectivity, such as for tractors operating in remote rural areas, as well as enabling apps that require very low latency when processing nearby sensor data to actuate adjustments in real-time on local devices.

There is no single AppEngine “box”; AppEngines may be built into machinery or provided as appliances. In the case of tractors, an AppEngine is not meant to be a replacement of vendors’ built-in MICSs but an add-on that may be retrofitted, integrating seamlessly with existing equipment. Adherence to the AppEngine reference specifications guarantees that the same app binary may run on any AppEngine. Of course, AppEngines may have different capabilities, in terms of processing power, storage of Inputs/Outputs.

3.3.1 ATLAS apps

3.3.1.1 Real-time apps

These apps typically interact with the environment (robotics, machinery) in real-time based on data exchanged with sensors and/or apps on other tractors (e.g. platoon management)

3.3.1.2 Job apps

Lean apps that download data (e.g. prescriptions), from cloud services that perform the heavy processing lifting, and are then able to perform their operations off-line, typically on a tractor where a job is a combination of a metric per GPS position, a driving path, a tractor guidance and implement management including Headland management and folding.

3.3.1.3 Utility apps

Utility apps that provide information to human operators without otherwise influencing their environment (e.g. drivability information on a field). Information from these apps can be shown on a terminal or via a connected device in a head up display or on gadgets like AR/VR glasses.

3.3.1.4 Platoon apps

Platoon apps enable the cooperation of different AppEngine-based machines on a common task (typically use case involves tractors in a field) via a local/mesh network. These scenarios are enabled by deploying the same platoon apps on all the cooperating devices. This approach enables a lean architecture; by enabling app siblings to interact with proprietary formats, there is no need for complex app interoperability standards.

3.3.2 AppEngine SDK

The AppEngine SDK (see Figure 7) is a mediation layer that enables apps to interact with their environment by abstracting the low level technologies and transports in a standardised API. The AppEngine SDK is also responsible for restricting access of apps to specific information or controls on the basis of the permissions that were granted for the app in the ATLAS registry.

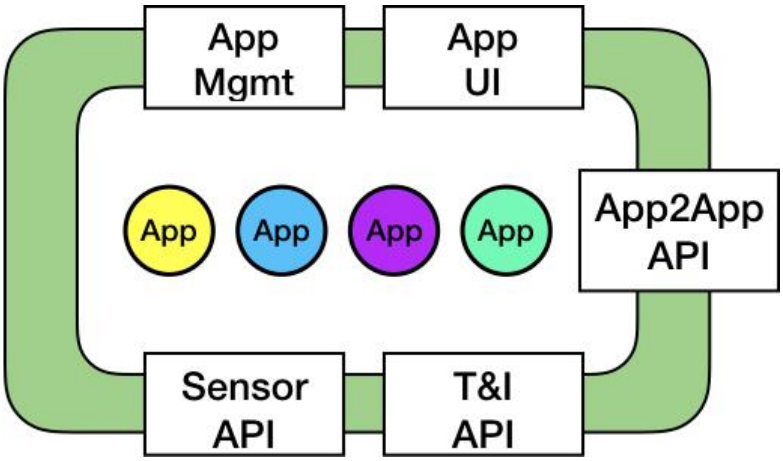

Figure 7: AppEngine SDK

Depending on the use case, different AppEngine implementation may provide only some of the SDK modules detailed below. The only module that is required in all implementations is the AppEngine Management.

3.3.2.1 T&I API

The Tractor and Implement API is an abstraction layer that enables apps to retrieve information or send controls to the machinery. The abstraction will leverage existing standards, where applicable (ISOBUS, TIM, Steering and Sequence control) but may also cover “standard” types of robotics appendages via ad-hoc connectivity.

3.3.2.2 App2App API

The App2App API abstracts the low-level connectivity layers (e.g. various 802.11 variants) for inter communication between sibling platoon apps. Application level messages and data formats are private to each platoon app and are out of ATLAS scope

3.3.2.3 Sensor API

The Sensor API abstracts the low-level connectivity layers to and from sensors, and defines formats for common data and measures. It is recommended that best practice format is prescribed for ubiquitous measure types but custom/proprietary payloads remain possible

3.3.2.4 App UI

The AppEngine SDK provides user interface support to enable rich user interactions at runtime.

3.3.3 AppEngine Management

The management functions are designed to enable interactions with the AppEngine and the ATLAS AppCenter.

3.3.3.1 AppEngine pairing

This function enables the association of an AppEngine instance with an ATLAS AppCenter account so that end-users may later select on which AppEngine instance a particular app should be installed.

3.3.3.2 AppEngine OS/SDK Updates

This function enables the update of AppEngine OS & SDK over the air (typically for security updates).

3.3.3.3 App Management

This function provides the means of installing and uninstalling apps on the AppEngine as well as other general administration activities such as certificate renewals (if applicable).

Installing an app also involves the establishment of a link between the app and its companion service, more specifically, between an app on a particular AppEngine instance and a corresponding user account on the companion service. The pairing information is stored by the AppEngine and accessible to the app.

Connections between AppEngine or apps and cloud-based services are always initiated on the on-premise as the AppEngine may be off-line or have dynamic ip-addresses.

3.3.3.4 AppEngine Feature Support

Different AppEngine instance may offer different type of features. For instance, a tractor-based AppEngine will typically be able to access ISOBUS/TIM/Steering/Sequence Control functions via the T&I API, whereas a farm-based AppEngine will not. This API provides the ATLAS AppCenter with the means to determine whether an ATLAS app is compatible with a given AppEngine instance.

3.3.4 AppEngine Certification

A certification process will be defined to certify AppEngine implementations. In particular, for AppEngines destined to be installed on tractors and connecting to the ISOBUS/TIM/Steering/Sequence Control, an AEF ISOBUS certification will be required.

3.3.5 Safety Considerations

Some app functions can impact machinery operations in a way that could be potentially hazardous. For instance, an app influencing a tractor's speed or automatically folding/unfolding implements should do so in a way that is safe to people on or nearby the machine, particularly for tractors that do not have built-in safeguards to override these commands in case of necessity.

Apps requiring high safety class permissions will have to undergo a stricter certification process before being approved in the registry.

3.4 ATLAS central components

The ATLAS architecture requires two central components that will be operated under the authority of a non-profit governance body in order to ensure trustworthiness in the ecosystem: (i) the ATLAS Registry that serves as a trusted directory for ATLAS services/apps and participants, and (ii) the ATLAS AppCenter on which registered apps may be searched and installed on AppEngines (see Figure 8).

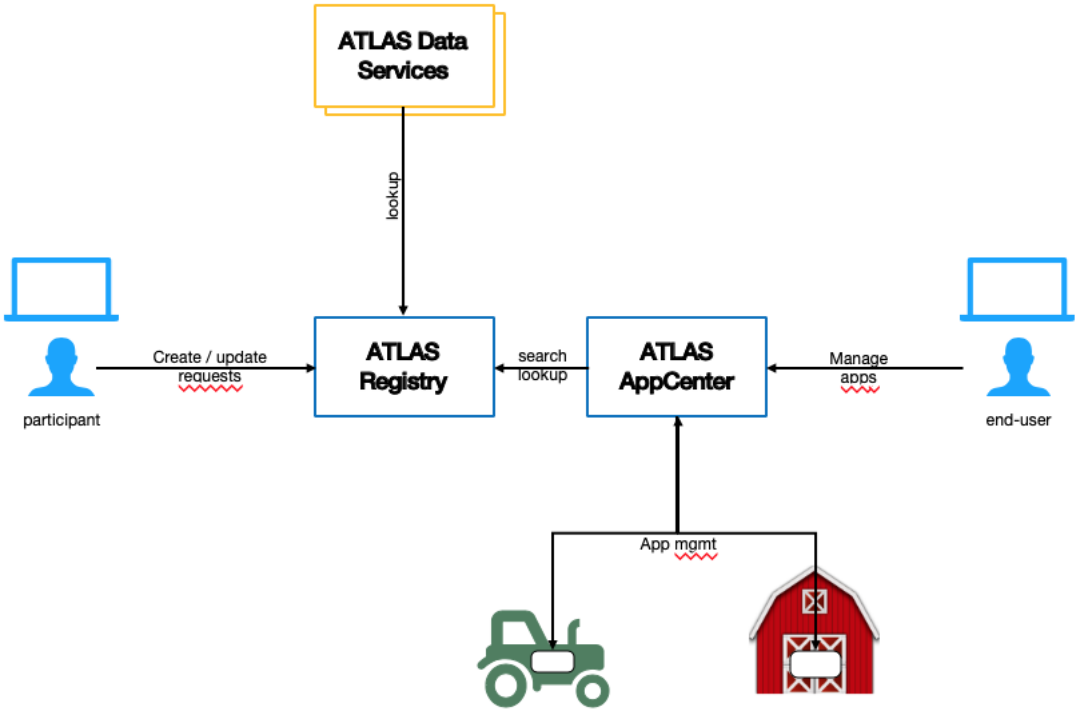

Figure 8: ATLAS central components

3.4.1 ATLAS Registry

The ATLAS Registry is the central component that serves as a trusted directory for ATLAS data services/apps and ATLAS participants. Participants are entered into the registry after their identity has been reasonably verified, at which point they may start submitting a request to register their ATLAS Services. Upon verification that the ATLAS Service information is valid and that the service complies with ATLAS requirements, the service entry is recorded and becomes searchable by other ATLAS participants.

Every ATLAS service must be registered in the ATLAS Registry. Additional registries may be created by interest groups that wish to define stricter domain-specific certifications for the purpose of ascertaining stricter quality standards, for instance. Nevertheless, while ATLAS participants (e.g. FMIS) may promote services with a specific certification, by connecting to the appropriate registry, they must also connect to the ATLAS Registry to allow farmers to use standard services as well.

3.4.1.1 Participant Registration Request

Independent Software Vendors wishing to provide ATLAS data services first need to register their identity and contact information via the Participant Registration Request API.

- participant name
- participant contact information
- … (full details are outside the scope of the high-level architecture)

3.4.1.2 Service Registration Request

Registered participants may submit Service Registration Requests via this API to request the creation or update of a service registry entry by providing the following information:

- participant id
- service name
- service base url
- description
- provided capabilities to other services
- required capabilities from other services
- optional capabilities from other services
- required AppEngine permissions for companion app (where applicable)
- companion app:
    - name
    - description
    - download url
    - required permissions
    - optional permissions
    - hash on binary code
- … (full details are outside the scope of the high-level architecture)

3.4.1.3 Capabilities and Permissions

Service Capabilities

Capabilities are defined as granular endpoint accesses that may be either provided or consumed. An endpoint identifies both the type of resource and the operation it performs (CRUD). The detailed specifications of capabilities are outside the scope of this document.

It is the serving service's responsibility to verify that an operation (endpoint) requested by a client lies within the capability scope declared by the client. A normalised dictionary of capabilities will be defined to enable meaningful interoperation and searches.

App Permissions

Permissions correspond to specific AppEngine SDK operations that an app requires to function. A normalised dictionary of permissions will be defined to enable meaningful interoperation. Furthermore, the dictionary may categorize each permission based into safety classes; for instance, permissions to commands that impact machinery operations will be in a higher safety class than permission to access sensor data. Apps requiring permissions of higher safety classes may then undergo a more extensive verification process prior to acceptance in the ATLAS Registry.

3.4.1.4 ATLAS Registry Lookup and Search

These APIs may be used by ATLAS data platforms (e.g. FMIS) to retrieve details about a given ATLAS data service, or to search for services based on various criteria such as capabilities.

3.4.1.5 Service Registry Entry Approval Process

The service registry entry approval process must validate the correctness of the entry request, at the minimum, ascertain the existence of the provided services capabilities endpoints. If the request includes a companion app entry, the approval process also verifies the availability of the download URL and of the app's integrity (by means of a hash, for instance).

Most of this process may be automated, but additional manual validation must be performed for services with companion apps. A human expert should validate the a-priori relevance of the requested permission with respect to the stated function of the app. For apps requesting highly safety-sensitive permissions, an additional certification process may be required where it is deemed that the AppEngine certification does not provide sufficient guarantees.

3.4.2 ATLAS AppCenter

The AppCenter provides a user interface to browse and search for apps, and the means for farmers to install, update and uninstall apps on their AppEngines. Farmers wishing to use ATLAS apps must register for an AppCenter account.

3.4.2.1 App Publication

There is no specific app publication process. Since all apps are described in the context of a companion service in the ATLAS Registry, the AppCenter can directly connect to the Registry to retrieve all necessary information about apps to be presented.

Updating an app is done via an update to its companion service entry.

3.4.2.2 AppEngine Pairing

The AppCenter, in conjunction with the AppEngine, will provide a mechanism to let farmers securely pair AppEngine instances with their AppCenter account.

3.4.2.3 AppCenter Browse and Search

The AppCenter web application will enable end-users to search for apps and view their companion web services, details.

3.4.2.4 App Management

In a personal space of the AppCenter, users will be able to view their paired AppEngines and to manage (install, uninstall, update, list) their apps. These operations will be queued in such a way as to guarantee their eventual completion, in case AppEngine instances are currently not in line, or connection issues occur.

A mechanism will be provided to link an app instance (within an AppEngine) with its companion service so that inter-communication may securely be carried out.

3.4.3 Validation and Certification

The ATLAS Registry approval process aims at being as lean and quick as possible to enable a fluent and agile open ecosystem. This approval process will include basic certification processes (where possible, fully automated) to ensure the ability of the service to properly integrate in the ATLAS Network. The basic certification will validate the ability to establishing pairing to other services and may be extended to validate further core features over time. However, we also recognise opportunity for other governance bodies to define and validate stronger certification processes in given domains. Such certified services provide end users with stronger a-priory quality guarantees.

The means to enable these external certification bodies is via community registries (see Figure 9). Community Registries should conform to the ATLAS Registry API but are operated independently from the ATLAS central components by third party organizations. These organizations are free to define their own guidelines, processes and quality control that a service must undergo in order to be included in their Community Registry. The Community registries may only certify services that are already registered in the ATLAS Registry, thereby ensuring universal basic conformity for all services.

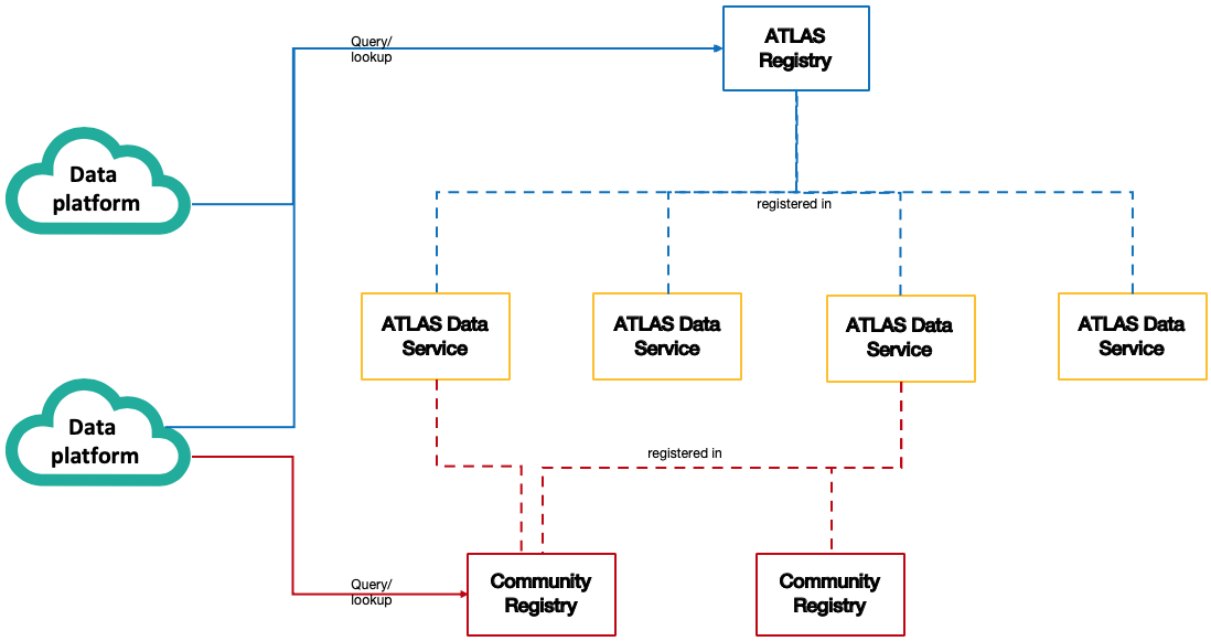

Figure 9: Community registries

The presence of an ATLAS data service in a Community registry is therefore a guarantee of certification in that community. ATLAS Data Platforms may promote ATLAS services originating from specific Community registries to their end users but they must also provide them with the means to access and configure basic-certified services available in the ATLAS Registry.

3.5 Version handling for capabilities in the architecture

As the architecture and the resulting service patterns and interface specifications will be developed through a process of concurrent circles of prototyping, inspection and adoption a versioning will be used to ensure compatibility of the designed components of deliverables within the ATLAS Network over time.

To counter the different demands for versioning there will not be just one version for all but a set of solutions and designs all versioned individually and by specific ways of versioning.

While a standardized document versioning through i.e. a GIT-repository is suitable for documents there will be the need of i.e. URL-versioning for REST service endpoints, which then refer to a specific version for an interface specification document.

All components and their type of versioning are still to define at the current level of maturity for the architecture. The following example shows a possible versioning structure:

Versions: (EXAMPLE)

ATLAS 1.5: (consists of)

Part 1: ATLAS Interoperable platforms network 1.5.0 (consists of)

1.1: ATLAS IPN Service Registry 1.2.3

1.2: ATLAS IPN Data provider 1.4.2

1.3: ATLAS IPN Data consumer 1.5.6

Part 2: ATLAS Interoperable Apps network 1.4.1

2.1: ATLAS IAN AppEnvironment 1.4.2

2.2: ATLAS IAN App Companion Services 1.2.0

3.6 Integration of security in the architecture

In the architecture there are different areas in which security aspects have to be considered and integrated. This includes, for example, data security, the transmission security of information, but also security aspects for app management between the App Center and App Engine. To have not too many different technologies making implementation difficult, a uniform concept should be developed. Depending on the requirements, this concept can be used to integrate an appropriate and, if possible, already existing technology directly into the architecture.

3.6.1 Security for data exchange

The security controls and sharing capabilities will vary between different on-premise or cloud-based solutions. When different on-premise or cloud-based solutions are to be used for storing information classified as restricted, then the following four security level criteria need to be met:

1. Transport Layer Security **(**TLS**)** as a secure transport protocol (e.g. HTTPS) must be available.

2. Integration with the **registration services**. (see chapter: ATLAS Registry)

3. **Application key pair** (public & private) as a key pair of the connected applications in the ATLAS Network. This is used to create a signature for the first onboarding and easy verification during data exchange process.

4. **Endpoint certificate (optional)** is used to encrypt the communication in the ATLAS Network. It can be delivered with the onboarding request.

Provider's service has undergone third party security testing (i.e. vulnerability scan/assessment or audit) and has continuous monitoring in place for system intrusions / unauthorized access.

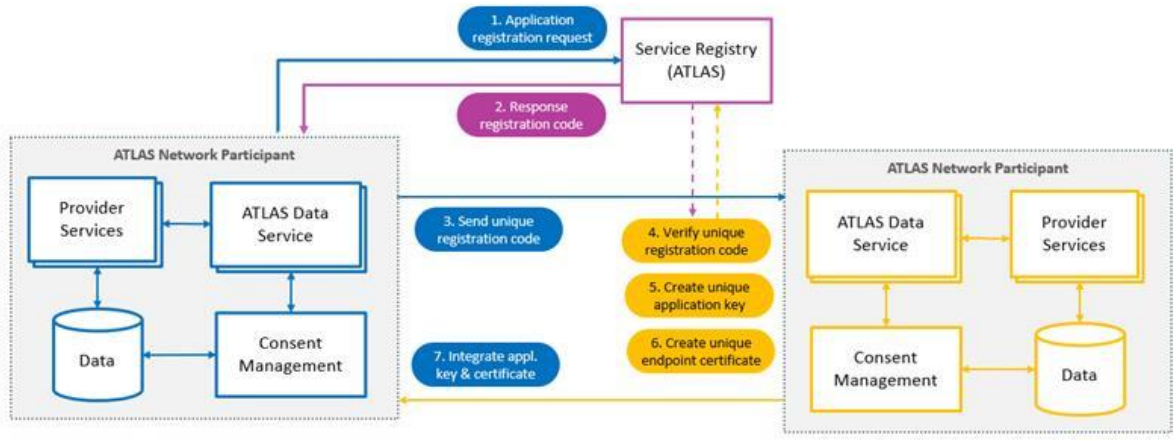

Figure 10: Registration and onboarding flow for secure data exchange

Figure 10 shows the individual steps from the registry and the final connection for the secure data exchange.

First, a data platform must be registered in the ATLAS Registration Service (see chapter: ATLAS Registry). This includes communicating the capabilities of the data platform, including which data formats can be supported. After successful certification, a unique registration code for the ATLAS Network is provided.

To ensure that the registration code is consistent and unique, it should have at least the two following components:

- Identifier = an unique application identifier
- Version = an unique application identifier for the application version

With this registration code a request can be sent from a data platform to the agriculture software. After successful matching in the ATLAS service and internal validation, a confirmation is sent as a verification to the data platform.

A unique and unambiguous key pair is then generated and exchanged between the data platform and agriculture software in the ATLAS Network. This key serves as a digital signature and allows secure communication between the parties. With digital signatures it should be practically impossible to forge or falsify a signature or to generate a second message for which this signature is also valid.

In order to exchange data between the data platform and the agriculture software in a secure and encrypted way, a unique endpoint certificate will be generated at the endpoint level. This certificate is used to securely exchange data between these two endpoints.

After successful integration of application key pair (public & private) a secure and encrypted data exchange is possible.

3.6.2 Security for AppEngine

As discussed in section 3.3, in order to manage apps on the AppEngine (install / uninstall) a pairing between the AppEngine and the AppCenter is required. In order to secure this pairing, it is important to establish that the AppEngine instances are owned, or at least controlled by, the AppCenter's account owner. The AppEngine provides an on-board administration user interface whose password is created on first activation. Once logged in to the administration console, the farmer has access to a "Manage Pairing" option. Selecting this option will prompt the farmer for his AppCenter account credentials. With valid credentials, the connection to his AppCenter is established and the farmer can now invoke the pairing command which will register a privately generated token (saved within the AppEngine instance) into his AppCenter account. Similarly, the farmer may un-pair his AppEngine, either from the AppEngine administration interface or from his AppCenter administration interface. The token is then used to authenticate all upstream (AppEngine → AppCenter) or downstream (AppCenter → AppEngine) communications. These operations can obviously only take place when the AppEngine is in a location with Internet access.

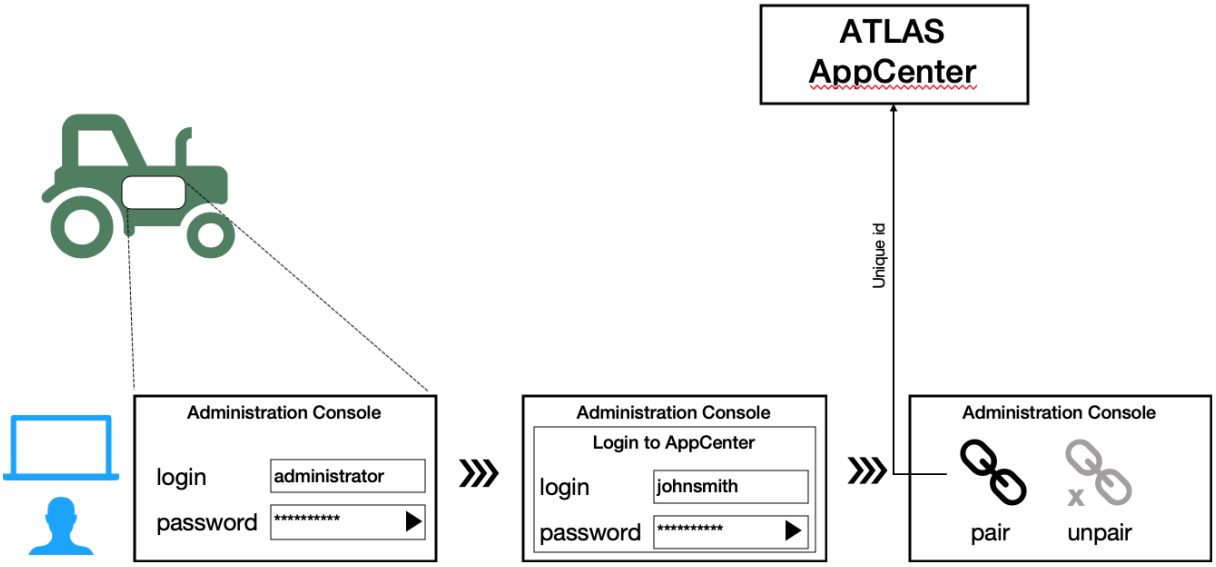

Figure 11: Security - AppEngine Pairing

3.7 Handling data forwarding in the architecture

Focus of ATLAS is to describe how participants of ATLAS can interoperate based on provided use cases. Data forwarding beyond ATLAS cannot be avoided, each participant of ATLAS needs to manage data in all matters himself therefore he can use the ATLAS Resource Identifier. Especially for data forwarding this means that if a participant wants to forward data to a 3rd party, individual agreements between all involved endpoints would be required.

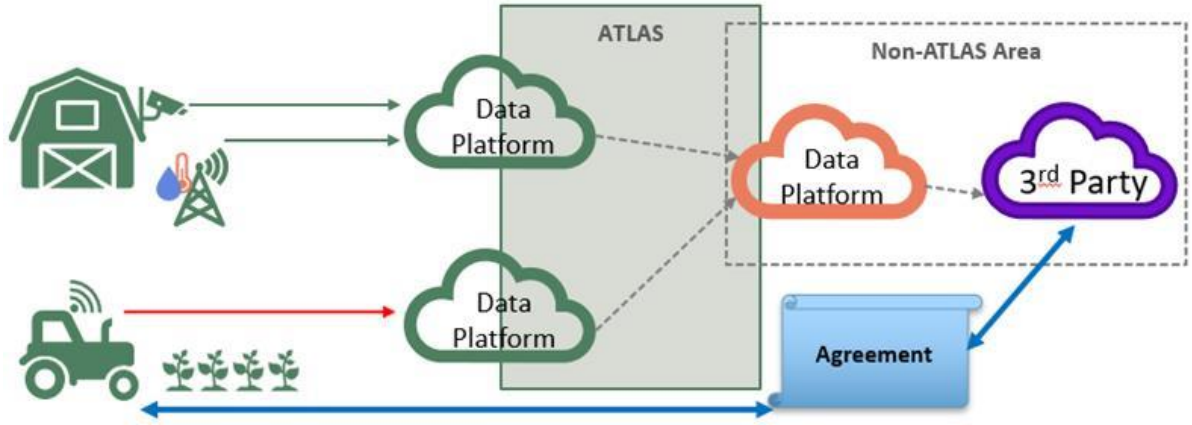

Figure 12: Endpoint agreement for data forwarding

There are not restrictions how a participant has to store the data, thus there is also no specification how a participant must treat data conflicts especially when data is coming from two different sources but providing the same data this can lead to data duplication and deviations that may impact results.

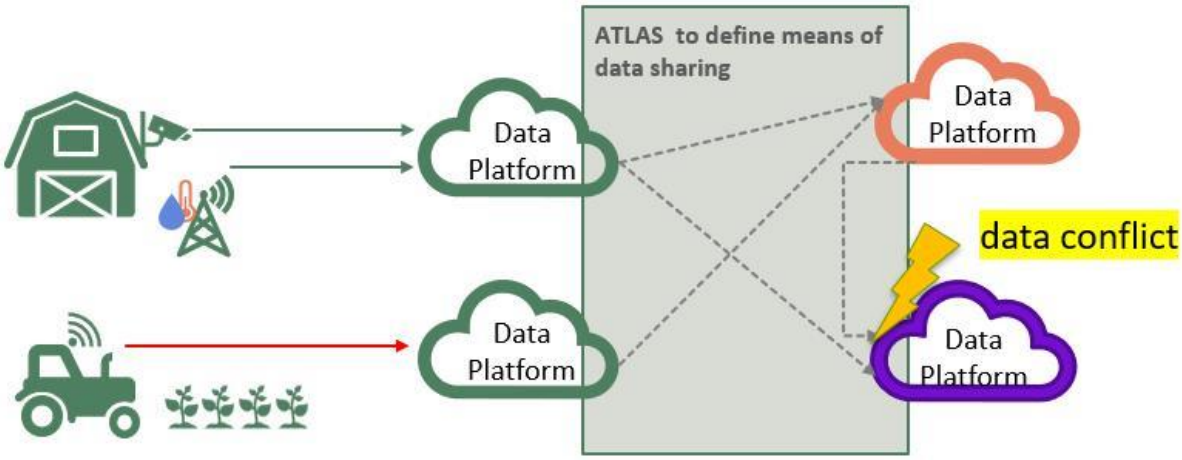

Figure 13: Data conflict with forwarded data

The forwarded data can contain an ATLAS Resource Identifier as meta data information associated with it. This could enable to the data service to identify resources in the network. The function is listed in the ATLAS Registry as a capability.

3.8 Handling GDPR in the architecture

The rules and conditions given through the General Data Protection Regulation (EU) 2016/679 (GDPR) are a relevant constraint which has to be considered in appropriate ways. The ATLAS AppCenter (see section 3.4.2) and the ATLAS Registry (see section 3.4.1) as central components have been identified by the core architecture team to require special emphasis on GDPR conformity. The management structure of ATLAS foresees the installation of dedicated advisory boards to bring external expertise into the consortium. It is planned to install such a legal advisory board with experts from the consortium members' networks.

4 Relation from the architecture to the use cases

Within this section, the relation of the architecture to the concrete agricultural use cases covered in ATLAS is illustrated by two example use-cases. A more extensive list of use cases which are the drivers of the end-user requirements can be found within deliverable D3.1. In this respect, the "Fertilization Use Case" can be considered as an aggregate use case to demonstrate the relation of multiple features of the architecture, whereas the "Platform independent cross brand machine tracking" points out the specific aspects of the ATLAS interoperability architecture to interconnect multiple OEM data platforms.

4.1 Fertilization Use Case

In this scenario, two tractors collaborate in real time to optimally apply the amounts of Nitrogen, Phosphorus and Potassium (NPK). The first tractor dispenses just the right amount of slurry to reach the required amount for NPK, whichever is reached first. The missing amounts of P and K are transmitted via in-field communications to the second tractor that can then apply the adequate amounts of fertilizers to reach the prescription goal.

The scenario will involve a farmer, two tractors, an NIR sensor, the farmer's access to OEM data platforms, a Fertilization service with its real-time platoon app, OEM data platforms, and a meteorological service. Further services may be involved in the actual implementation, such as a satellite imagery service, a fertilization product database service, etc., but as they do not illustrate additional architecture features, they are left out from the scenario for the sake of simplicity.

4.1.1 Setup

The farmer owns two tractors with an AppEngine. The first is equipped with a slurry tank with a NIR sensor, while the second has both a front-mounted spreader and a trailer sprayer.

Pairing: the farmer establishes all necessary pairings with contributing ATLAS services, e.g. between his Fertilization service account and an ATLAS meteorological service, between his OEM data platform where he keeps information about his fields boundaries and optionally about field driving paths, between two OEM platform services, etc.

AppEngine pairing and app installation: the farmer creates an ATLAS AppCenter account to which he pairs both his tractors. The farmer can then locate the Fertilization service's companion app in the ATLAS AppCenter and install it on both his tractors.

4.1.2 Preparation

As the farmer prepares his task on his Fertilization web application, the Fertilization service retrieves information on field boundaries (and driving paths, if available) from the peered OEM data platform, meteorological data, etc.

The Fertilization service then determines the optimal amounts of NPK to apply on a fine field geo-position granularity, based on algorithms processing satellite images, soil analysis data, as well as regulatory constraints. This data also gets shared with the OEM data platform to visualise in this environment.

When ready, the farmer associates the two tractors he plans to use in the task in the Fertilization web application and triggers the export of the prescription and other necessary on-board data to the companion apps on the tractors.

4.1.3 In-field Operations

The first tractor follows a pre-defined path, received from companion app or recorded with the app in the field, around the field and sends the path information in real-time to the second tractor who follows in its tracks with some delay. The tractors positions are also recorded and streamed by proprietary means to the OEM data platform of the respective tractor; these may be consolidated into a service (via ATLAS service-to-service communication) where a farm's mixed-OEM fleet can be tracked in real-time.

The moment a tractor enters the field according to the prescribed path, the tractor stops and the operator is informed that the implement is about to be unfolded, with an option to suspend the operation.

The first tractor's NIR sensor adjusts the rates of slurry dispense so as to reach the required amount for NPK, whichever is reached first, in accordance with the prescription.

The geo-localised information on missing amounts (deltas) of phosphorus and potassium is sent via App2App to the sibling app on the other tractor (see Figure 14). As the second tractors advances, it uses the deltas received from the first tractor for its current location to deliver the remaining amounts of P and K, based on the known properties of the products loaded on its two tanks. The apps provide a consolidated field-wide visualisation of deltas to the operator as the operation progresses.

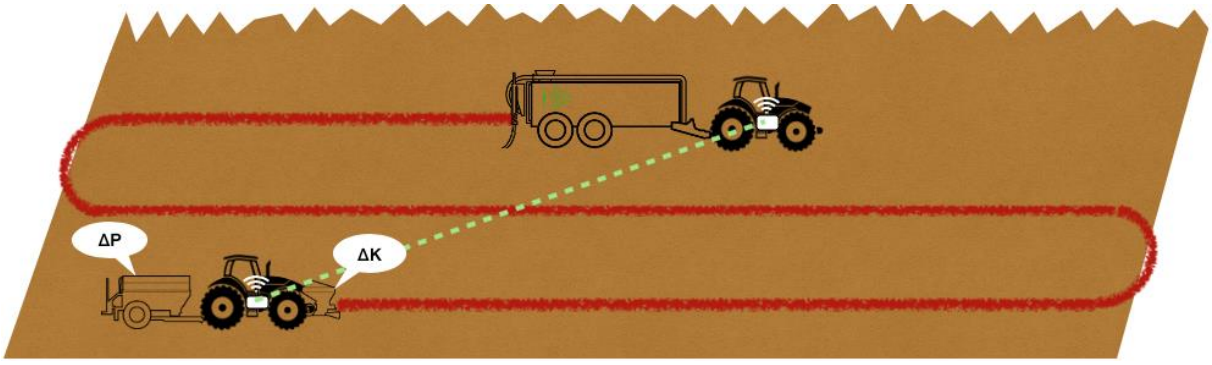

Figure 14: Multi-tractor Operation with in-field communication

As soon as a tractor finishes its task and is about to leave the field, the tractor is stopped and the operator is informed that the implement is about to be folded, with an option to suspend the operation.

All actually applied quantities are automatically recorded by the tractors' built-in system as well as by the apps. The tractor OEM transmits its data by proprietary means to their data platform, while the app sends the consolidated values to their companion Fertilization service. The retrieved data may be synchronized from the Fertilization service or the OEM data platform to further ATLAS services, but this is not in the scope of the present use case.

Table 6: Architecture components involved in the fertilization use case

Architecture Feature	**Use case mapping**
Platform data exchange (see section 3.2)	• Preparation data sharing • Path sharing • As applied value sharing • Retrieval of field boundaries in data platform by Fertilization service, etc. • Tractor positions between OEM clouds
Service pairing (see section 3.2.4)	• Between farmer's Fertilization service and his OEM data platform account, etc.
ATLAS Registry and Certification (see sections 3.4.1 and 3.4.3)	• Fertilization service • OEM data platform service • Meteorological data service • Etc.
ATLAS AppCenter (see section 3.4.2)	• Account creation • Search for app • Pairing of AppEngines • Installation of app on AppEngines
AppEngine (see section 3.3)	• Pairing • App installation
App and Companion service (see section 3.3.3)	• Real-time platoon Fertilization app • On-field app to App communication • Prescription data download • Consolidated as-applied upload • Improved automation
Safety (see section 3.3.5)	• Tractor speed control • Implement management (boom folding / unfolding)

4.2 Platform independent cross brand machine tracking

This use case covers the development of cross brand tracking services to enable the equipment owners to track their equipment in systems of their choice, not related to any brand or equipment manufacturer solution unlike today where owners are not able to see even the simplest information like the geographical position of the multi coloured fleet in one place. The setup of this use case includes a farmer with at least two equipment, starting with self-propelled harvesting machines or tractors of different brands, all telemetry enabled and visible in a tracking system (OEM or third party).

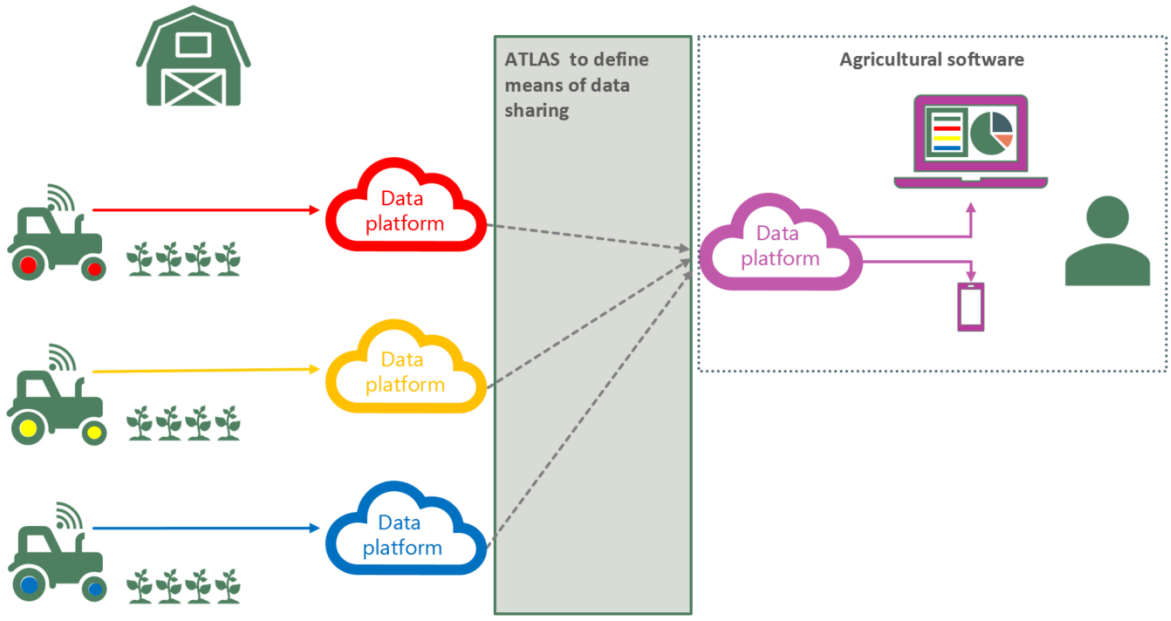

Figure 15: Use case visual

The farmer needs to interconnect all tracking systems by a pairing mechanism that enables him to decide in which of the (or a third one) tracking systems all tracking information of the fleet shall be made visible.

This farmer should be able to discover which of the systems can be connected to each other through a discovery function in the system the farmer actually wants to use for the task of fleet tracking. After the discovery the farmer can enable the connection between the tracking systems by using the credentials given out for any of the systems to be connected (OAuth and system account connection).

After that the equipment needs to be running and working to actually produce data (starting with a very small set of information) which then is transported to the coupled other systems.

4.2.1 In field operations

To actually get the use case running the participating equipment needs to be run in the field to measure positioning and equipment data which then is transported to the tracking systems and from there on to the tracking system of choice of the farmer.

The tracks of the equipment then are shown in all participating systems with a specified delay which is acceptable for the farmer (a delay will occur in any case).

Table 7: Architecture components involved in the machine tracking use case

Architecture Feature	**Use case mapping**
Data platform data exchange (see section 3.2)	• Preparation data sharing • Self -propelled machine and tractor positions between OEM data platforms
Data platform pairing (see section 3.2.4)	• Between farmer's tracking systems
ATLAS Registry and Certification (see sections 3.4.1 and 3.4.3)	• Tracking services • Etc.

5 Conclusion

In this deliverable document, we described the high-level reference architecture which will be implemented to achieve one of the main goals of ATLAS: the interoperability of sensors, agricultural machines and data platforms. The architecture was designed with concrete use cases in mind. In this respect, two basic concepts are foreseen: data platform based data exchange and processing, and on-board computing and processing capabilities through a self-contained computing platform. With these complementary solutions, the implementation of complex use cases with real-time requirements and challenging in-field conditions will be possible, and a significant contribution to the digitalization of agriculture will be made.

The whole architecture was designed to require only a minimum of basic, central components: the ATLAS Registry, the AppCenter and the AppEngine are sufficient to build a whole ecosystem of applications.

As an explanatory aspect it must be added that the central components will need a strong regulation regarding safety and security aspects through an independent authority. This authority will need to fulfill tasks of certification and fostering of the ATLAS Network to ensure that the quality of the services offered in the network comply to a defined level of quality and reliability so that the network itself offers a significant benefit when joining.

Levels of certification for safety and security are to be defined over the course of the project and not upfront to not design levels which later on cannot be fulfilled by joining participants.

The intention of this document is to provide the high-level blueprint of the ATLAS interoperability ecosystem. The concrete implementation of the concepts presented will be conducted within the course of the project and the technical details of all components will be worked out in this respect.