Service Template Specification

STS id crop\_protection\_advisor

Version 0.0.3

Status DRAFT

Author(s) M. Caballero (AgriCircle AG)

**The "crop\_protection\_advisor" Service Template**

Abstract

This document describes the specifications for crop\_protection\_advisor services whose purpose is to provide product application maps for selected compounds. crop\_protection\_advisor services are agnostic of tractors or other equipment; they typically use agronomical algorithms to provide advice on crop protection which would then be typically carried out by the ATLAS Equipment Centre.

Table of Contents

[1 Introduction 3](#_Toc105072774)

[2 Terminology 3](#_Toc105072775)

[3 Crop Protection Advisor Usage Scenarios 3](#_Toc105072776)

[3.1 NEED EXAMPLE 4](#_Toc105072777)

[4 Advisor Service Template API 4](#_Toc105072778)

[4.1 Disease Monitoring Endpoints 5](#_Toc105072779)

[4.1.1 Monitor field 5](#_Toc105072780)

[4.1.2 Get field monitoring status 6](#_Toc105072781)

[4.1.3 Unmonitor field 6](#_Toc105072782)

[4.2 Advice Endpoints 6](#_Toc105072783)

[4.2.1 Prepare Advice 6](#_Toc105072784)

[4.2.2 Get Advice Info 7](#_Toc105072785)

[4.2.3 Cancel Advice 7](#_Toc105072786)

[4.2.4 Get Application Info 7](#_Toc105072787)

[5 Data Formats 8](#_Toc105072788)

[5.1 Product Application File Format 8](#_Toc105072789)

[6 Access and Authentication 11](#_Toc105072790)

[7 Dynamic Behaviour 12](#_Toc105072791)

[7.1 Crop Protection Lifecycle 12](#_Toc105072792)

[7.2 Disease Monitoring 12](#_Toc105072793)

[7.2.1 Polling 13](#_Toc105072794)

[7.2.2 Notification 13](#_Toc105072795)

[7.3 Advice Preparation 13](#_Toc105072796)

[7.3.1 Polling 13](#_Toc105072797)

[7.3.2 Notification 14](#_Toc105072798)

[7.4 General Comments on Notifications 14](#_Toc105072799)

# Introduction

Crop protection is essential for crop yield. The crop protection requirements depend on TO BE COMPLETED.

crop\_protection\_advisor services determine product application maps to optimally satisfy specific crop protection requirements which can later be applied in one or more applications by the ATLAS Equipment Centre on a field.

Farmers may use different crop\_protection\_advisor services depending on the specific type of crop protection they require, their budget, the accuracy and the granularity of information they require, etc.

# Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC 2119](https://datatracker.ietf.org/doc/html/rfc2119) [[RFC2119](https://datatracker.ietf.org/doc/html/rfc2119)] and indicate requirement levels for compliant implementations.

The notation "[xxx]" (xxx in square brackets) is equivalent to "array of xxx".

When used alone, the terms "crop\_protection\_advisor" and "advisor" refer to "crop\_protection\_advisor service template". Instead, "crop\_protection\_advisor service and "advisor service" are equivalent to "a service implementing the crop\_protection\_advisor service template".

# Crop Protection Advisor Usage Scenarios

The following sections describe some aspects of the use of the crop\_protection\_advisor service template. The examples were chosen to illustrate the basic functions of applications using crop\_protection\_advisor, not to limit what crop\_protection\_advisor may be used for.

The typical use case scenarios with a crop\_protection\_advisor service typically an FMIS that orchestrates operations with a field\_data and the ATLAS Equipment Centre as can be seen below:

A screenshot of a computer

Description automatically generated with medium confidence

## NEED EXAMPLE

~~Most crops have nitrogen (N), phosphorous (P) and potassium (K) nutrient requirements.~~

~~A fertilisation service can be designed so as to focus on NPK nutrients and privilege the use of organic over chemical fertilisers.~~

~~The service receives a request to prepare an NPK fertilization for a corn plantation using slurry, and two other products, one with high P dosage and another with high K dosage. Based on the corn NPK requirements, the soil's properties and previously applied products, the fertilization service determines the optimal amounts of each product's needs. These application maps are delivered on-demand to another service that is responsible for converting the data to a format understandable by the equipment that will perform the tasks on the field.~~

# Advisor Service Template API

This section provides a very high-level summary of the advisor service template functions:

Disease Monitoring Endpoints

Monitor field

Get field monitoring status

Unmonitor field

Advice Endpoints

Prepare Advice

Get Advice Info

Cancel Advice

Get Application Info

Within this section, functions are summarized with simple tables:

+--------------------------------------------+

| <logical function name> |

+-------------+------------------------------+

| Inputs | <URL parameters or |

| | request body attributes |

+-------------+------------------------------+

| Outputs | <response body attributes> |

+-------------+------------------------------+

Only the most meaningful parameters are discussed in this document. Please refer to the OpenAPI specifications for full details.

Advisor services are not required to handle intense traffic from a single client, especially for GET functions. Implementors MAY generate a 429 TOO MANY REQUESTS error response if the rate of calls exceed some pre-defined quota.

## Disease Monitoring Endpoints

These functions relate to registering, and unregistering fields for which alerts are desired. Advisor implementation may provide the means to actively monitor fields and generate alerts on disease risks.

### Monitor field

This endpoint registers a field to be monitored for one or more diseases by the crop\_protection\_advisor.

+--------------------------------------------+

| monitor\_field |

+-------------+------------------------------+

| Inputs | field urn, diseases, |

| | notification url |

+-------------+------------------------------+

| Outputs | |

+-------------+------------------------------+

The standard for disease identifiers is taken from the European and Mediterranean Plant Protection Organization (EPPO); more specifically, from the EPPO Code for a given disease (<https://www.eppo.int/RESOURCES/eppo_databases/eppo_codes>).

See 7.2 Disease Monitoring for details on the monitoring notification process.

### Get field monitoring status

This endpoint returns the monitoring info for a field.

+--------------------------------------------+

| monitoring\_info |

+-------------+------------------------------+

| Inputs | field urn |

+-------------+------------------------------+

| Outputs | diseases, status, alerts |

+-------------+------------------------------+

The status may be one of IN\_PROGRESS, CONFIGURATION\_REQUIRED, READY, FAILED. In this context, the IN\_PROGRESS status indicates that some background processing is still being carried out and that the service is not yet ready to perform active monitoring. The CONFIGURATION\_STATUS indicates that the end-user is required to perform some manual configuration on the service’s proprietary UI. FAILED indicates that monitoring is not possible. 0 or more alerts may be present only when the status is READY.

### Unmonitor field

This endpoint cancels monitoring of a field.

+--------------------------------------------+

| unmonitor\_field |

+-------------+------------------------------+

| Inputs | field urn |

+-------------+------------------------------+

| Outputs | |

+-------------+------------------------------+

## Advice Endpoints

These endpoints relate to the creation of crop protection advices.

### Prepare Advice

This endpoint is used to requests an advice to be prepared for specified diseases and the products that are at disposition.

+--------------------------------------------+

| prepare\_advice |

+-------------+------------------------------+

| Inputs | field urn, substances, |

| | notification URL |

+-------------+------------------------------+

| Outputs | advice info |

+-------------+------------------------------+

crop\_protection\_advisor services may retrieve any relevant information to prepare their advices from the field\_data service (e.g., boundaries, current crop, previous applications, etc.). Advices should be prepared by taking the current conditions into account. Advisors have the opportunity to refine their internal recommendations at the actual time the application for the advice is requested (see Get Application Info).

Advice preparation may be a lengthy process. Clients may either poll or request to be notified to determine that the advice is READY (see 7.3 Advice Preparation).

### Get Advice Info

This function returns status information about an advice.

+--------------------------------------------+

| advice\_info |

+-------------+------------------------------+

| Inputs | advice urn |

+-------------+------------------------------+

| Outputs | advice info |

+-------------+------------------------------+

In addition to the information submitted when preparing the advice, the advice info also contains a status about the advice preparation which is one of IN\_PROGRESS, CONFIGURATION\_REQUIRED, READY, FAILED. In this context, the IN\_PROGRESS status indicates that some background processing is still being carried out and that the advice is not yet. The CONFIGURATION\_STATUS indicates that the end-user is required to perform some manual configuration on the service’s proprietary UI in order to enable the completion of the requested advice.

### Cancel Advice

+--------------------------------------------+

| cancel\_advice |

+-------------+------------------------------+

| Inputs | advice urn |

+-------------+------------------------------+

| Outputs | - |

+-------------+------------------------------+

Invoking this endpoint causes the advice status to become FAILED. If an advice that is not yet in READY state is cancelled and if a notification URL was provided in the prepare\_advice request, a notification MUST be dispatched

### Get Application Info

This endpoint may only be performed on advices whose status is READY. It returns a download URL to the application map and the total estimated amount of product needed carry out the application.

+--------------------------------------------+

| get\_application\_info |

+-------------+------------------------------+

| Inputs | advice id, application ref |

+-------------+------------------------------+

| Outputs | download URL, product amount |

+-------------+------------------------------+

All values are in the corresponding product units. Please refer to "5.1 Product Application File Format" for detailed specification of the downloaded product application map file format.

For maximum accuracy, the application map is requested in a "just-in-time" fashion by the ATLAS Equipment Centre (or any other consumer) to give an opportunity to advisors to make adjustments to their pre-computed preparation. However, since this information is usually requested just as farmers are about to go out on the field to perform a task, it is important that it is computed in a short time (maximum, few seconds). Any potentially time-consuming processing must be pre-computed and cached internally during the registration and/or preparation stages.

# Data Formats

This section focuses on the description of binary (file) data formats. Please refer to the crop\_protection\_advisor OpenAPI specifications for details on all other payload and parameter descriptions.

## Product Application File Format

Nutrient Application data MUST be in GeoPackage (<https://www.geopackage.org/>) format version 1.2 or newer.

+=========================================================+

| TABLE gpkg\_contents |

+=====+=============+=====================================+

| ROW | COLUMN | VALUE |

+-----+-------------+-------------------------------------+

| | table\_name | "atlas" |

| +-------------+-------------------------------------+

| | data\_type | "attributes" |

+-----+-------------+-------------------------------------+

All Atlas GeoPackage files MUST contain an attributes table named "atlas" with a single row and the following structure:

+============================+

| TABLE atlas |

+================+===========+

| COLUMN | TYPE |

+----------------+-----------+

| id | INTEGER |

| type | TEXT |

| participant | TEXT |

| format\_version | TEXT |

+----------------+-----------+

and a single row:

+=========================================================+

| TABLE atlas |

+=====+================+==================================+

| ROW | COLUMN | VALUE |

+-----+----------------+----------------------------------+

| | type | "crop\_protection\_application" |

| +----------------+----------------------------------+

| | participant | "<atlas participant id>" |

| 1 +----------------+----------------------------------+

| | application | "…" |

| +----------------+----------------------------------+

| | format\_version | "MAJOR.MINOR" |

+-----+----------------+----------------------------------+

Services SHALL validate that the "atlas.participant" field matches the information attached to the authentication context in which the file is uploaded.

If the GeoPackage was generated and uploaded by an Atlas service, then "atlas.application" SHOULD be "<service name>-<service version>", otherwise it MAY contain "<application name>-<application version>". In either case, no validation will be performed.

The "atlas.format\_version" field MUST be the version of the crop\_protection\_advisor template that is targeted by the client uploading the file. GeoPackage files of a given type are guaranteed to be compatible (no breaking changes) for a same MAJOR version of "atlas.format\_version".

A crop protection application GeoPackage MUST contain the following additional tables:

+=========================================================+

| TABLE gpkg\_contents |

+=====+=============+=====================================+

| ROW | COLUMN | VALUE |

+-----+-------------+-------------------------------------+

| … |

+-----+-------------+-------------------------------------+

| | table\_name | "product" |

| +-------------+-------------------------------------+

| | data\_type | "2d-gridded-coverage|features" |

+-----+-------------+-------------------------------------+

| | table\_name | "application\_info" |

| +-------------+-------------------------------------+

| | data\_type | "attributes" |

+-----+-------------+-------------------------------------+

If the data\_type of the product table is "features", then the columns must be as follows:

+=========================================================+

| TABLE product |

+=====+=================+=================================+

| ROW | COLUMN | VALUE |

+-----+-----------------+---------------------------------+

| | id | "INTEGER" |

| +-----------------+---------------------------------+

| | geometry | "GeoPackage Geometry" |

| +-----------------+---------------------------------+

| | amount | "amount in product units (REAL)"|

|-----+-----------------+---------------------------------+

Instead, if the data\_type of the product table is "2d-gridded-coverage" then it must be defined as follows:

+=========================================================+

| TABLE product |

+=====+=================+=================================+

| ROW | COLUMN | VALUE |

+-----+-----------------+---------------------------------+

| | id | "INTEGER" |

| +-----------------+---------------------------------+

| | zoom\_level | "INTEGER" |

| +-----------------+---------------------------------+

| | tile\_column | "INTEGER" |

| +-----------------+---------------------------------+

| | tile\_row | "INTEGER" |

| +-----------------+---------------------------------+

| | tile\_data | "BLOB (TIFF)" |

|-----+-----------------+---------------------------------+

Where tile\_data is a tiff blob holding Float32 values as per GeoPackage specifications.

In either case, the product values are in the units specified in "application\_info.info.product.unit".

By convention, application map GeoPackage files MUST contain an attributes table named "application\_info" with the following structure:

+============================+

| TABLE application\_info |

+================+===========+

| COLUMN | TYPE |

+----------------+-----------+

| id | INTEGER |

| type | TEXT |

| date | DATETIME |

| info | TEXT |

+----------------+-----------+

and a single row:

+=========================================================+

| TABLE application\_info |

+=====+=============+=====================================+

| ROW | COLUMN | VALUE |

+-----+-------------+-------------------------------------+

| | type | "crop\_protection" |

| 1 +-------------+-------------------------------------+

| | date | "yyyy-MM-ddTHH:mm:ssZ" |

| +-------------+-------------------------------------+

| | info | "<application\_info\_json>" |

+-----+-------------+-------------------------------------+

Crop protection "application\_info.info" is a JSON object serialized as a string, such as:

+-----------------------------------------------+

| { |

| "product": { |

| "name": "optional name", |

| "unit": "kg|l", |

| "amount": "amount\_in\_product\_unit", |

| "substances": { |

| "80-56-8": "amount\_in\_grams\_per\_unit",|

| "50-00-0": "amount\_in\_grams\_per\_unit",|

| … |

| } |

| } |

| } |

+-----------------------------------------------+

The standard for Crop protection substances identifiers is taken from the Pesticide Properties DataBase (PPDB, <http://sitem.herts.ac.uk/aeru/ppdb/en/index.htm>); more specifically, the "CAS RN" (Chemical Abstracts Service Registry Number) from a PPDB entry.

# Access and Authentication

Farmers MUST have an account setup on a crop\_protection\_advisor service in order to authenticate and access API endpoints. The service implementor is responsible for the creation of accounts; it is not covered in the service template specifications.

Unless specifically documented in the OpenAPI specifications, all API calls must include credentials in form of Bearer authentication (also called token authentication). Clients can obtain an access token on behalf of their user from the service's authorization server (see ATLAS service pairing).

Some of the information held by crop\_protection\_advisor services may be considered to be sensitive from a GDPR perspective. The service's authorization server SHOULD request the client's end-user consent at service pairing time in order to deliver an access token.

# Dynamic Behaviour

The purpose of the diagrams in this section is to illustrate communication patterns, more complex than plain request/response API calls, that involve several interactions and/or asynchronous behaviour. Even though a sequence diagram representation is used, the diagrams are by no means to be interpreted as UML Sequence Diagrams. Specifically, in the spirit of focusing on functional behaviour and readability, error handling is deliberately not covered in the diagrams.

## Crop Protection Lifecycle

The "Crop protection lifecycle" in ATLAS typically operates in a larger context involving three services and an end-user application such as an FMIS which, in collaboration, provide an end-to-end solution from algorithmically computed advices to actual operations performed on a field.

A picture containing chart

Description automatically generated

## Disease Monitoring

There are two methods for determining whether a monitoring request has completed (successfully or unsuccessfully): by polling the monitoring\_info endpoint or by notification.

### Polling

After requesting a field to be monitored, the client polls monitoring\_info endpoint at regular intervals until the returned status is CONFIGURATION\_REQUIRED, READY or FAILED.

A CONFIGURATION\_REQUIRED status indicates that a user action is required on the proprietary user interface of the advisor. By convention, in case of CONFIGURATION\_REQUIRED status, the advisor MAY return a browser URL to the location where a user can provide additional parameters (as required for a specific service implementation). It is the user interface of the client application originating the request to display an appropriate message to the end-user with a clickable link to the provided URL.

### Notification

If a notification URL was supplied on the monitor\_field endpoint, it will be invoked by the advisor when the monitoring status changes.

Advanced advisors have the ability to monitor conditions on fields in the background and proactively generate disease alerts on the notification URL provided at field registration.

Advisors MUST invoke the notification URL supplied by the client with an HTTPS POST command. The payload will be identical to the one that would be returned by a request to the monitoring\_info endpoint.

## Advice Preparation

There are two methods for determining whether an advice preparation has completed (successfully or unsuccessfully): by polling the advice\_info endpoint or by notification.

### Polling

After requesting an advice preparation, the client polls the advice\_info endpoints at regular intervals until the returned status is READY or FAILED.

The status may be CONFIGURATION\_REQUIRED which indicates that a user action is required on the user interface of the advisor. By convention, in case of CONFIGURATION\_REQUIRED status, the service SHOULD return a browser URL to the location where a user can provide additional parameters (as required for a specific service implementation). It is the user interface of the ADIS originating the request to display an appropriate message to the end-user with a clickable link to the provided URL.

A screenshot of a computer

Description automatically generated with low confidence

### Notification

If a notification URL was supplied on the prepare\_advice endpoint, it will be invoked by the advisor when the preparation status changes.

A screenshot of a computer

Description automatically generated with low confidence

Advisors MUST invoke the notification URL supplied by the client with an HTTPS POST command. The payload will be identical to the one that would be returned by the advice\_info request.

## General Comments on Notifications

Services must provide best efforts to deliver notifications. A notification is considered successful if the target returns an http result code 2XX.

Errors may occur during notification delivery. Depending on the type of error, services must react in different ways:

i) Network error - the connection to the client's host (from notification URL) cannot be established. The service MUST retry a certain number of times. The number of retries and possible backoff strategy is left at the discretion of the service implementer.

ii) Server errors (5XX result code) – these errors are potentially transient. The same strategy as for Network errors SHOULD be applied.

iii) Client errors (4XX result code) – typically when the notification URL is invalid, or the authentication is invalid/expired. 4XX errors should never be sent for transient client-side conditions and therefore services SHOULD NOT attempt retries.

Upon an excessive number of errors, services MAY give up further notification attempts. In that case, clients can only retrieve completion information via polling.