Service Template Specification

STS id	irrigation_advisor
Version	0.1.0
Status	PUBLISHED
Author(s)	I.D. Tsakmakis, V. Pisinaras, C. Brogi (LRI,FZJ)

The "irrigation_advisor" Service Template

Abstract

This document describes the specifications for irrigation_advisor services whose purpose is to provide an irrigation map, for a selected field and a specific application date. irrigation_advisor services are not obligated to take into consideration the water supply system infrastructure (main and lateral pipelines or canals etc.) or the equipment that will implement the suggested irrigation (drip, sprinkler, micro-sprinkler etc.). They are based on soil water balance equations and/or crop models of varying complexity levels (from empirical to mechanistic), that are fed with data related to plant-soil-atmosphere-water continuum and derive estimations of the required net irrigation amount. The latter can be used successively by an irrigation management application vendor.

Table of Contents

1 Introduction

Shortage of plant available water may strongly reduce crop yield or cause crop production failure. Thus, in certain regions and for certain cultivations, irrigation is essential to achieve optimum crop yield. However, the “when” and “how much” to irrigate may vary significantly depending on crop variety, soil characteristics, local weather conditions, and on the existing soil water amount available for the plant.

An irrigation_advisor service determines the amount of water that is required to maintain the water levels within the rooting zone profile in satisfactory levels. This amount should not be miss-interpreted as the water needed to refill rooting zone back to field capacity. The satisfactory levels may vary significantly depending on crop type, crop different growing stages, or the implemented irrigation strategies’ scope: "no-water-stress", "regulated-mild water stress", "regulated-severe water stress", "sustained water stress" etc.

Farmers may use different irrigation services/options depending on the accuracy they want to achieve, their budget, or the availability of ancillary data.

2 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], when, and only when, they appear in all capitals, as shown here and indicate requirement levels for compliant implementations.

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

When used alone, the term "irrigation_advisor" refers to "irrigation_advisor service template". Instead, "irrigation_advisor service" is equivalent to "a service implementing the irrigation_advisor service template".

3 Pre-requisites

A thorough understanding in the following is required for both service consumers or service provider implementors:

- GeoPackage specifications (https://www.geopackage.org/)

4 irrigation_advisor Usage Scenarios

The following sections describe some aspects of the use of the irrigation_advisor service template. The examples were chosen to illustrate the basic functions of irrigation application using irrigation_advisor, not to limit what irrigation_advisor may be used for.

4.1 Vineyard Irrigation

Wine grape vines are a delicate crop, that can produce new vegetation and ripening grapes at the same time, with substantially different water needs during its' various growing stages.

An irrigation_advisor service can be designed to promote regulated deficit irrigation over full irrigation, in certain growing stages (e.g., veraison stage), aiming to control the vegetation growth and improve grapes quality characteristics (e.g., sugar, pH, etc.).

The service receives initially a request to prepare an irrigation application map for a vineyard plantation. Based on the vines growing stage (linked to growing degree days after budburst), the soil's properties and previously applied irrigations, the irrigation_advisor service determines the optimal irrigation amount, for a given application day, that will maintain the vineyard plants under the proper water status (e.g., well irrigated or mildly stressed).

The irrigation map is delivered on an irrigation management application vendor that is responsible (a) to convert the irrigation plan data to a format understandable by the equipment that will perform the irrigation application;(b) send the implemented irrigation application map to the field_data service.

5 Service Template API Overview

This section provides a very high-level summary of the irrigation_advisor API:

Irrigation Endpoints
    Σφάλμα! Το αρχείο προέλευσης της αναφοράς δεν βρέθηκε.
field
    Σφάλμα! Το αρχείο προέλευσης της αναφοράς δεν βρέθηκε.
monitoring info
    Σφάλμα! Το αρχείο προέλευσης της αναφοράς δεν βρέθηκε.
field
Advice Endpoints
    Prepare Advice
    Get Advice
    Cancel Advice
    Get Application Info

Implementations of irrigation_advisor may require more parameters that are not included in the API. Such implementations MAY provide end-user configuration and management tools in a proprietary user interface.

Within this section, functions are summarized with simple tables:

logical operation 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.

irrigation_advisor services are not required to handle intense traffic from a single client, such as the one that may result from being directly invoked on user interface interactions in an FMIS, for instance. Implementors MAY generate a 429 TOO MANY REQUESTS error response if the rate of calls exceed some pre-defined quota.

5.1 Irrigation Monitoring Endpoints

These functions relate to registering, and unregistering fields for which alerts are desired. irrigation_advisor services may provide the means to actively monitor fields and generate alerts when an irrigation application is recommended.

5.1.1 Monitor Field

This endpoint registers a field to be monitored by the irrigation_advisor. On a successful registration, irrigation_advisor will monitor the depletion levels within plants' root zone and proactively suggest an irrigation application when a certain depletion threshold is about to be reached.

monitor_field	
Inputs	field_urn, notification URL
Outputs	(see monitoring_info)

NOTE: an irrigation_advisor service MAY not support field monitoring. In that case, the implementation MUST return an error code 501 (Not Supported).

See Irrigation Monitoring for details on the monitoring notification process.

5.1.2 Get monitoring recommendations

This endpoint returns the monitoring recommendations for a field, that is whether a irrigation_advisor proactively recommends an irrigation to be applied on a monitored field.

monitoring_info	
Inputs	field_urn
Outputs	status, [recommendations]

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_REQUIRED 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.

5.1.3 Unmonitor field

This endpoint cancels monitoring for a field.

unmonitor_field	
Inputs	field_urn
Outputs	-

5.2 Advice Endpoints

These endpoints relate to the creation of irrigation advices. An advice may be requested to plan and optimize irrigation application for a specific date.

5.2.1 Prepare Advice

This endpoint is used to request an advice to be prepared for a specific application date.

prepare_advice	
Inputs	field_urn, application
Outputs	advice urn

irrigation_advisor services may retrieve any relevant information for preparing 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 Advice Preparation).

5.2.2 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_REQUIRED 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.

5.2.3 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

5.2.4 Get Application Info

This endpoint may only be performed on advices whose status is READY. It returns a download URL to the application map, the total estimated amount of water needed to implement the irrigation and the number of applications that this amount is suggested to be applied.

get_application_info

Inputs	advice id
Outputs	download URL, irrigation amount, number of applications

All values are in the corresponding predefined units. Please refer to "Irrigation File Format" for detailed specification of the downloaded product irrigation map file format.

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_REQUIRED 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.

For maximum accuracy, the irrigation 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.

6 Data Formats

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

6.1 Product Application File Format

Irrigation 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
1	type	"irrigation"
	participant	"<atlas participant id>"
	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 irrigation_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 fertilisation application GeoPackage MUST contain the following additional tables:

TABLE gpkg_contents		

ROW	COLUMN	VALUE
	...	
	table_name	"irrigation"
	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 irrigation

ROW	COLUMN	VALUE
	id	"INTEGER"
	geometry	"GeoPackage Geometry"
	amount	"total water amount (m^3) (INTEGER)"
	applications	"nub. of applications (INTEGER)"

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

TABLE irrigation

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 (http://docs.opengeospatial.org/is/17-066r1/17-066r1.html#_storage_formats_and_grid_cell_values).

TABLE application_info

COLUMN	TYPE

id	INTEGER
type	TEXT
date	DATE
info	TEXT

and a single row:

TABLE application_info		
ROW	COLUMN	VALUE
1	type	"irrigation"
	date	"yyyy-MM-dd"
	info	"<application_info_json>"

Irrigation "application_info.info" is a JSON object serialized as a string, such as:

```
+--------------------------------------------------+
| {                                                |
|   "applications": [{                             |
|    "id": "e.g. app_01",                          |
|    "unit": "mm",                                 |
|    "amount": <amount_in_application_unit>,       |
|    "date_time": "yyyy-MM-ddTHH:mm:ssZ"           |
|   },                                             |
|   {                                              |
|    "id": "e.g. app_02",                          |
|    "unit": "mm",                                 |
|    "amount": <amount_in_application_unit>,       |
|    "date_time": "yyyy-MM-ddTHH:mm:ssZ"           |
|   }]                                             |
| }                                                |
+--------------------------------------------------+
```

7 Access and Authentication

Farmers MUST have an account setup on an 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 advisor services may be considered 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.

8 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.

8.1 Irrigation Lifecycle

The "irrigation lifecycle" in ATLAS typically operates in a larger context involving at least five services and an end-user such as an irrigation management application vendor which, in collaboration, provide an end-to-end solution from algorithmically derived irrigation plans to actual irrigation applications performed on a field.

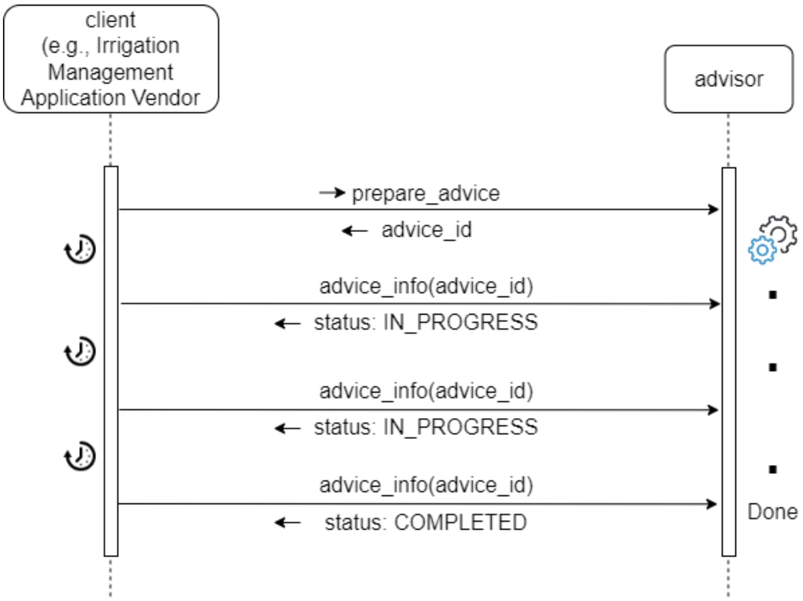
8.2 Advice Preparation

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

8.2.1 Polling

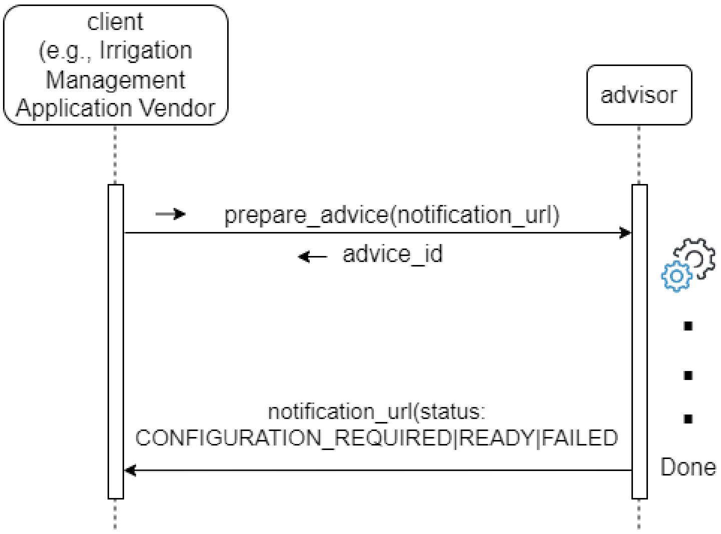
After requesting an advice preparation, the client polls the advice_info endpoint at regular intervals until the returned status is READY or FAILED.

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 irrigation management application vendor originating the request to display an appropriate message to the end-user with a clickable link to the provided URL.



8.2.2 Notification

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



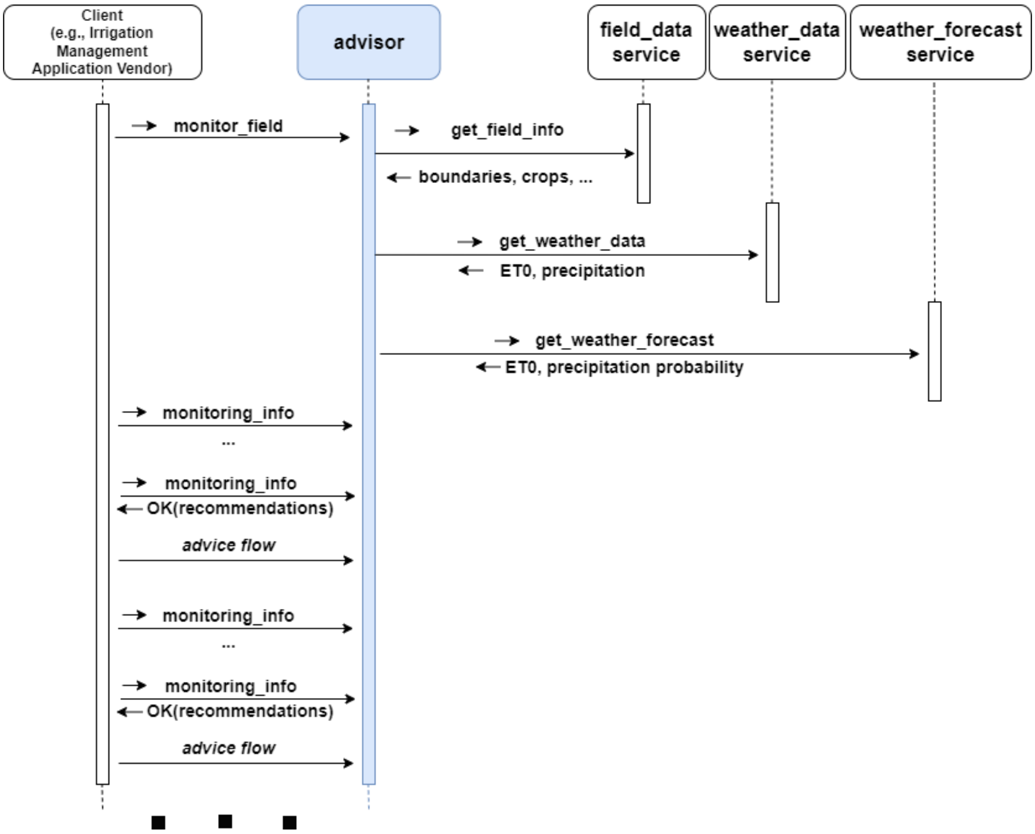
Advisors MUST invoke the notification URL supplied by the client with an HTTPS POST command. The payload will be identical to the one that would be returned by the advice_info request.

8.3 Field Monitoring

Advanced advisors have the ability to monitor conditions on fields in the background and proactively generate crop irrigation recommendations. There are two methods for determining whether recommendations are available: by polling the monitoring_info endpoint or by notification.

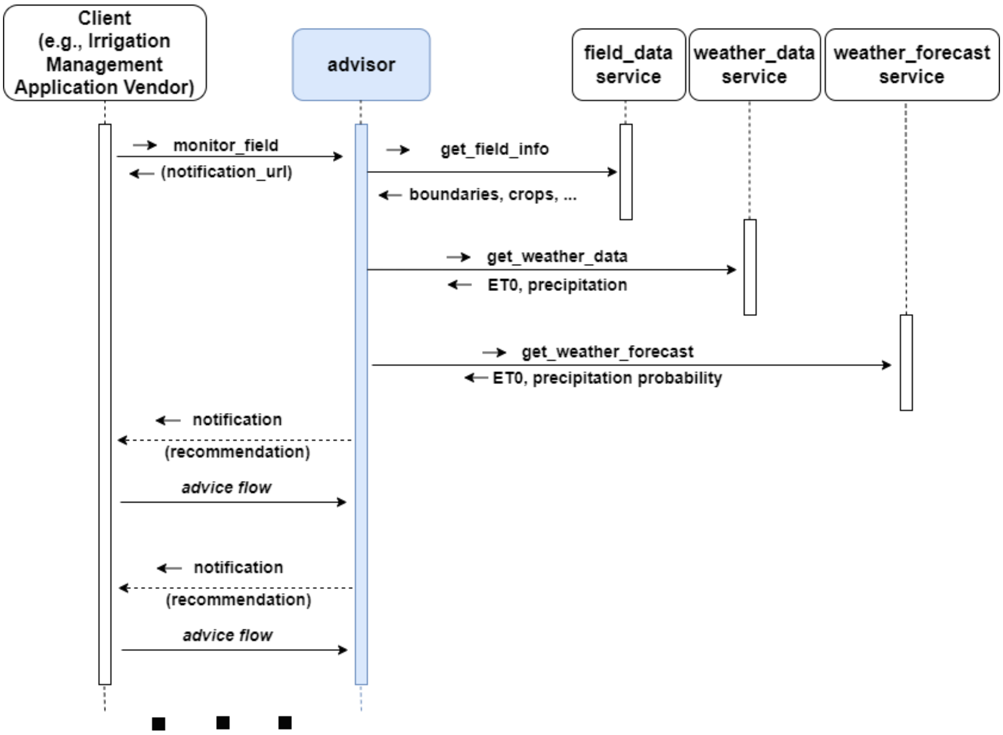
8.3.1 Polling

After requesting a field to be monitored, the client polls monitoring_info endpoint at regular intervals to check whether any 'recommendations' are available.



8.3.2 Notification

If a notification URL was supplied on the monitor_field endpoint, it will be invoked by the advisor when a new recommendation becomes available.



Advisors MUST invoke the notification URL supplied by the client with an HTTPS POST command. The payload will be as defined in the OpenAPI 'Recommendation' specifications.

8.4 Genera 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.