Service Template Specification

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

The "irrigation_planning" Service Template

Abstract

This document describes the specifications for irrigation_planning services whose purpose is to provide irrigation plan, for a selected field, for the next few days. An irrigation plan is consisted of irrigation maps, with at least one map for each of the plans' days. Irrigation_planning services do not 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 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 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 BCP 14 [RFC 2119] [RFC8174], 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_planning" refers to "irrigation_planning service template". Instead, "irrigation_planning service" is equivalent to "a service implementing the irrigation_planning service template".

3 Irrigation Planning Usage Scenarios

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

3.1 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 planning 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 plan for a vineyard plantation. Based on the vines growing stage (linked to growing degree days after budburst), the soil's properties and previously implemented irrigations, the irrigation planning service determines the optimal irrigation amounts, for the next few days, that will maintain the vineyard plants under the proper water status (e.g., well irrigated or mildly stressed). The number of days that the service will run for, can be either passed as an argument during the initial request or can be pre-defined by the irrigation_planning service.

This irrigation plan is delivered on-demand to an irrigation management application vendor that is responsible (a) to choose among the planning days which will be the irrigation day (schedule irrigation); (b) converting the irrigation plan data to a format understandable by the equipment that will implement the suggested irrigation;(c) send the implemented irrigation plan to the field_data service.

4 Service Template Functions

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

Irrigation Planning Functions
- Prepare
- Get Plan Status
- Cancel

Error! Reference source not found.
- Get Plan Info

Error! Reference source not found.
- Get Irrigation Info
- Get Irrigation 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.

irrigation_planning 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.

4.1 Irrigation Planning Functions

These functions relate to the development of an irrigation plan. The planning process MAY be requested (a) to calculate the optimal water amount that should be applied on a field for the next few days; (b) determine the optimum number of irrigation events (within a day) through which this amount is RECOMMENTED to be applied.

4.1.1 Prepare Plan

This function allows for the preparation of the background information that MAY be necessary for the development of the irrigation plan. The preparation may take from some seconds to hours, depending on the complexity of the models integrated into the irrigation_planning service and the corresponding data requirements.

prepare_plan	
Inputs	field_urn, planning_days, notification URL
Outputs	planning info

planning_days is an integer parameter that corresponds to the number of days that the irrigation_planning service SHOULD run for. The number of the planning_days is RECOMMENDED to be defined by the irrigation management application vendor, who sent the initial request. If not defined by the irrigation management application vendor, the number of planning_days MUST be provided by the irrigation_planning service. In the case that the planning_days number defined by the irrigation management application vendor, is larger than that the irrigation_planning service can provide, the latter returns an irrigation plan for a number of days equal to its' maximum capability.

irrigation_planning services may retrieve any relevant information to initialize their planning from the field_data

service (e.g., boundaries, crop details, irrigation applications, etc.).

Clients may poll for the status of a planning process (see Get Plan Status) or be notified of completion (successful or unsuccessful) if they supplied a notification URL.

4.1.2 Get Plan Status

This function returns status information about a planning preparation process.

get_plan_status	
Inputs	plan_urn
Outputs	status

The plan status is one of "PREPARING", "READY", "CANCELED", "FAILED".

4.1.3 Cancel Plan

cancel_plan	
Inputs	plan_urn
Outputs	-

4.2 Irrigation Scheduling

Irrigation scheduling function purpose is to act as a decision support recourse for the irrigation management application vendor, helping him to decide the most suitable date for the irrigation implementation.

4.2.1 Get Plan Info

This function may only be performed on irrigation plans whose status is "READY". It returns (a) a download URL to the irrigation plan (GeoPackage file); (b) a date-time string which defines when the irrigation plan will expire; (c) brief irrigation info for each of the planning days, including the total_amount of water that is RECOMMENDED to be applied on the field (m^3), the number of irrigation events through which the total_amount is RECOMMENDED to be applied and the date (YYYY-MM-DD) that corresponds to the planning day. RECOMMENDED irrigation amount within irrigation plan (GeoPackage file) SHOULD be in millimetres of water (mm or $m^3/1000m^2$), while total_amount of water SHOULD be in cubic meters (m^3). Please refer to "5.1 Product

Application File Format" for detailed specification of the downloaded irrigation plan file format.

Irrigation management application vendor SHALL use this function to obtain the irrigation plan and use it in combination with info related to water availability constraints (e.g., water quotas), water supply system and irrigation system limitations and capacities, to choose the most suitable day to schedule the next irrigation implementation.

get_plan_info	
Inputs	plan_urn
Outputs	download URL, URL expiration date, planning days info[day1:{ total_amount, irrigation_events, date}]

4.3 Scheduling Implementation

These functions deal with re-estimating the field water needs and thus the irrigation plan for a specific date-time defined by the irrigation management application vendor. These functions can be used only an hour (or less) before the irrigation implementation.

4.3.1 Scheduled Irrigation

This function informs irrigation_planning services about the date-time that the irrigation management application vendor decided to implement irrigation, so they can take advantage of the up-to-date meteorological, forecast, implemented irrigations and soil moisture data to optimise the irrigation plan accordingly.

scheduled_irrigation	
Inputs	plan_urn, implementation_datetime
Outputs	up-to-date_plan_urn, status

implementation_datetime is a date-time string in ISO8601 format, that MUST be provided by the irrigation management application vendor.

The scheduled_irrigation status is one of "PREPARING", "READY", "CANCELED", "FAILED".

4.3.2 Get Irrigation Info

This function may only be performed on an up-to-date irrigation plan whose status is "READY". It returns (a) the implementation_datetime; (b) a download URL to the up-to-date irrigation plan (GeoPackage file); (c) the date-time that the up-to-date plan is going to expire; (d) the total_amount of water that is required to implement the irrigation plan (m^3); and (e) brief info about the recommended irrigation events, such as the number of irrigation event, the corresponding total_amount of water that is recommended to be applied (m^3) on the specific irrigation event and the suggested start date-time. Recommended irrigation amount within irrigation plan (GeoPackage file) SHOULD be in millimetres of water (mm or $m^3/1000m^2$), while total_amount of water SHOULD be in cubic meters (m^3). Please refer to "5.1 Product Application File Format" for detailed specification of the downloaded irrigation plan file format.

get_irrigation_info	
Inputs	up-to-date plan urn
Outputs	implementation_datetime, download URL, URL expiration date-time, irrigation plan total_amount, irrigation events info[{ number, total_amount, date-time}]

5 Data Formats

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

5.1 Product Application File Format

Irrigation Plans 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		
ROW	COLUMN	VALUE
	type	"irrigation_planning"
	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_planning 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".

5.1.1 Irrigation Plan File Format

An initial GeoPackage Irrigation Plan MUST contain the following additional tables:

TABLE gpkg_contents		
ROW	COLUMN	VALUE
	…	
	table_name	"day_1"
	data_type	"2d-gridded-coverage|features"
	min_x	"min_longitude"
	min_y	"min_latitude"
	max_x	"max_longitude"

	max_y	"max_latitude"
	srs_id	"4326"
	…	
	table_name	"day_n"
	data_type	"2d-gridded-coverage|features"
	min_x	"min_longitude"
	min_y	"min_latitude"
	max_x	"max_longitude"
	max_y	"max_latitude"
	srs_id	"4326"
	table_name	"planning_info"
	data_type	"attributes"

TABLE planning_info

ROW	COLUMN	VALUE
	day_1_date	"ISO 8601 date (STRING)"
	day_1_total	"total_amount(m^3)(INTEGER)"
	day_1_events	"number of irri. events(INTEGER)"
	day_1_start	"events start date-times (STRING)"
	…	
	day_n_date	"ISO 8601 date (STRING)"
	day_n_total	"total_amount(m^3)(INTEGER)"
	day_n_events	"number of irri. events(INTEGER)"
	day_n_start	"events start date-times (STRING)"

5.1.2 up-to-date Irrigation Plan File Format

A GeoPackage for an up-to-date Irrigation Plan MUST contain the following additional tables:

TABLE gpkg_contents

ROW	COLUMN	VALUE
	…	
	table_name	"day"
	data_type	"2d-gridded-coverage|features"
	min_x	"min_longitude"
	min_y	"min_latitude"
	max_x	"max_longitude"
	max_y	"max_latitude"
	srs_id	"4326"
	table_name	"planning_info"
	data_type	"attributes"

TABLE planning_info		
ROW	COLUMN	VALUE
	events	"total number of events (INTEGER)"
	amount	"total_amount(m^3) (INTEGER)"
	start	"events start date-times (STRING)"

5.1.3 Features and Tiles data_type tables

Every unique, features or tiles data_type table, within the gpkg_contents table, MUST be accompanied by a user-defined features or tile pyramid data table. The name of the latter table(s) SHOULD match with the corresponding table_name(s) within gpkg_contents table. For example, in the case of the Irrigation Plan GeoPackage file ("5.1.1 Irrigation Plan File Format"), the features or tile data tables MAY take the names day_1, day_2, …, day_n etc.

TABLE day_1 (features)	
ROW	COLUMN	VALUE
	id	"INTEGER"
	geometry	"GeoPackage Geometry"

		irrigation	"required amount (mm)(INTEGER)"
		event	"number of irri. event (INTEGER)"

TABLE day_1 (tiles)		
ROW	COLUMN	VALUE
	id	"INTEGER"
	zoom_level	"INTEGER"
	tile_column	"INTEGER"
	tile_row	"INTEGER"
	tile_data	"BLOB (PNG/JPEG/TIFF)"

Tiles data_type table SHOULD have a number of rows equal to the number of the proposed irrigation events. Column "id" elements MUST match with the irrigation event number, e.g. an id equal to 2 MUST correspond to the second irrigation event.

Each tile of the tile_data BLOB column, MUST have one layer/band named "irrigation", with pixel values corresponding to an integer number (int32) that SHOULD be equal to the required irrigation amount (mm).

Moreover, in the case of features or tiles data_type tables, the gpkg_contents table MUST be accompanied by the gpkg_spatial_ref_sys and gpkg_geometry_columns (in the case of features data_type) or gpkg_tile_matrix (in the case of tile data_type) tables, as they described in the relative GeoPackage standard documentation (https://www.geopackage.org/)

6 Access and Authentication

Irrigation management application vendors MUST have an account setup on a irrigation_planning 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).

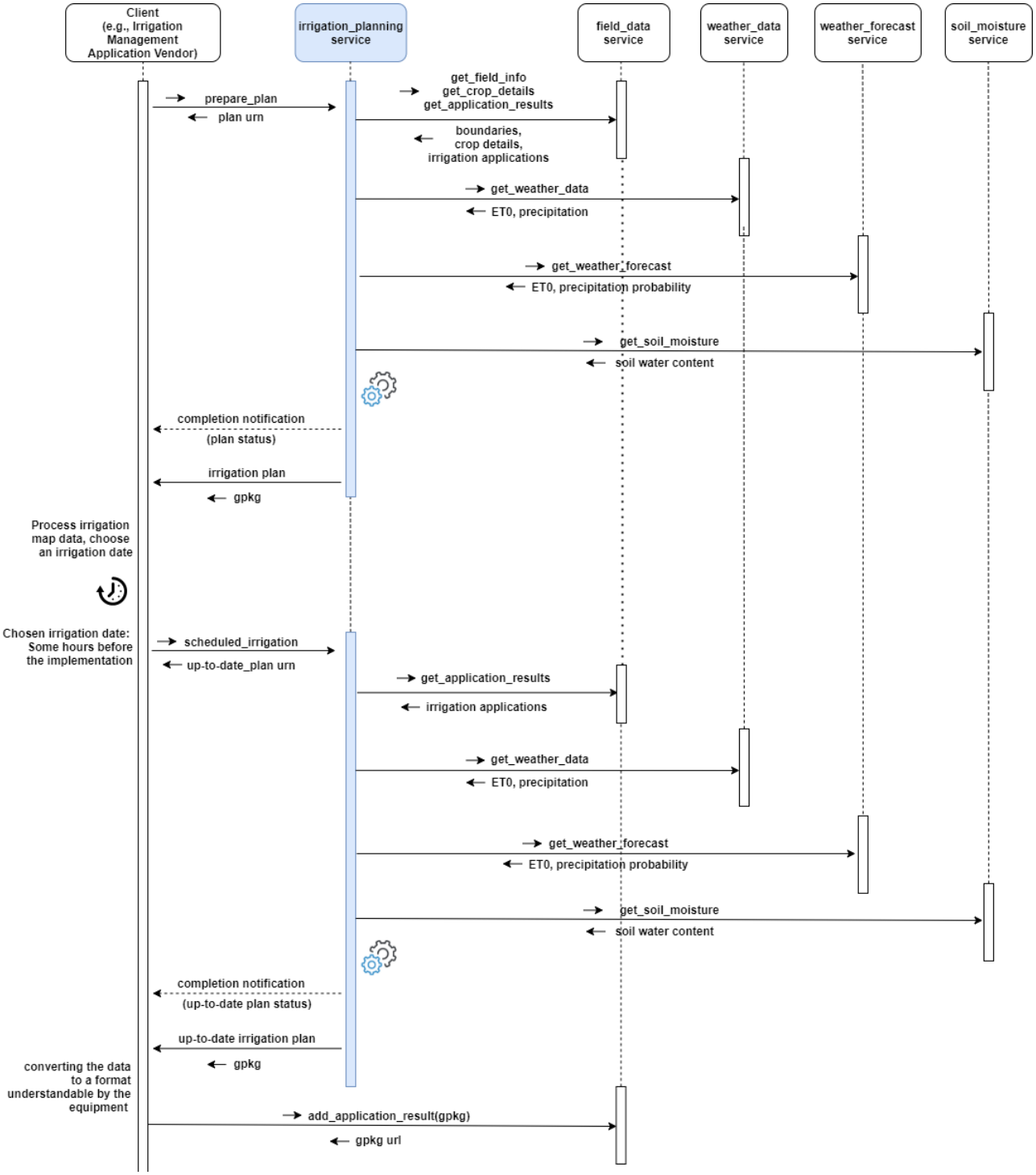
Some of the information held by irrigation_planning services may be considered to be sensitive from a GDPR perspective. The service's authorization server SHOULD request the client's end-user consent at service pairing time in order to deliver an access token.

7 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.

7.1 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.

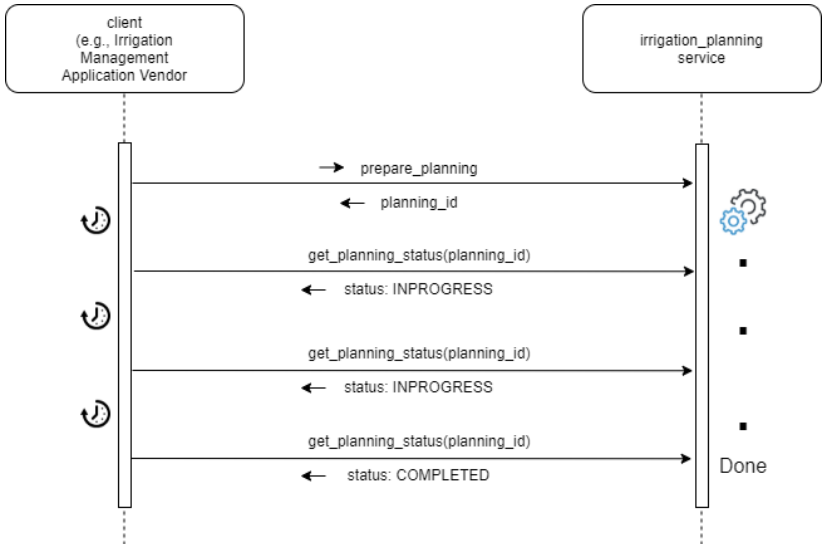


7.2 Planning Completion

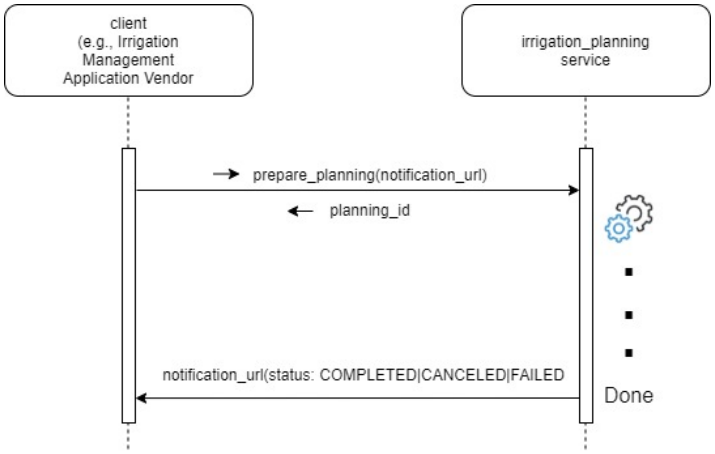
There are two methods for determining whether a plan preparation has been completed (successfully or unsuccessfully): by polling the get_plan_status or by notification.

7.2.1 Polling

After requesting a plan preparation, the client polls get_plan_status ,at regular intervals, until the returned status is "READY", "CANCELED", or "FAILED".



7.2.2 Notification



If a notification URL was supplied on the prepare_plan or scheduled_irrigation functions, it will be invoked by the irrigation_planning service when the preparation status of (a) the irrigation plan or (b) the up-to-date irrigation plan changes.

Service 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 get_plan_status or scheduled_irrigation requests.

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.