Service Template Specification

STS id ec\_extension

Version 0.0.1

Status draft-WIP

Author(s) M. Caballero (AgriCircle AG)

**The "ec\_extension" Service Template**

Abstract

This document describes the specifications for ec\_extension services whose purpose is to dynamically extend the task processing of the ATLAS Equipment Centre.

Table of Contents

[1 Introduction 3](#_Toc103338392)

[2 Terminology 3](#_Toc103338393)

[3 Pre-requisites 3](#_Toc103338394)

[4 ec\_extension Use Scenarios 3](#_Toc103338395)

[4.1 Precision Crop Protection 3](#_Toc103338396)

[5 Service Template Functions 4](#_Toc103338397)

[5.1 Configuration Functions 5](#_Toc103338398)

[5.1.1 Get Capabilities 5](#_Toc103338399)

[5.2 Task Management Functions 5](#_Toc103338400)

[5.2.1 Is Task Processable 5](#_Toc103338401)

[5.2.2 Pre-process Application Map 5](#_Toc103338402)

[5.2.3 Post-process Application Result 6](#_Toc103338403)

[6 Data Formats 6](#_Toc103338404)

[6.1 Application Map File Formats 6](#_Toc103338405)

[6.2 Application Results File Formats 6](#_Toc103338406)

[7 Dynamic Behaviour 6](#_Toc103338407)

[7.1 ec\_service with Companion App 6](#_Toc103338408)

# Introduction

The ATLAS Equipment Centre handles transfer of application maps to tractor, retrieves the as applied results and updates them on the farmer’s field\_data service. This processing is very generic and is not able to deal with advanced use cases.

The ec\_extension service template defines a mechanism that enables dynamically configured ec\_extension services to participate and enhance the ATLAS Equipment Centre task processing. Notably, the ec\_extension service template introduces support for custom ATLAS AppEngine app that can further customize the task processing in near-real time.

# 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) 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 "ec\_extension" refers to "ec\_extension service template". Instead, "ec\_extension service" is equivalent to "a service implementing the ec\_extension service template".

# Pre-requisites

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

* GeoJSON specifications (<https://geojson.org/>)
* GeoPackage specifications (<https://www.geopackage.org/>)
* The ATLAS Equipment Centre
* The ATLAS AppEngine and AppEngine SDK

# ec\_extension Use Scenarios

The following sections describe some aspects of the use of ec\_extension. The examples were chosen to illustrate the basic operation of applications using ec\_extension, not to limit what ec\_extension may be used for.

## Precision Crop Protection

The standard process for crop protection follows the “advisor pattern” described in the crop\_protection service template: an advice for a field is prepared (e.g. based on remote sensing) and then a task (advice, equipment) is submitted to the ATLAS Equipment Centre to be carried out on the field.

Consider now the use case where a vendor would like to develop a device with a lidar to scan trees as it drives through an orchard and adjust the prescribed amount of crop protection accordingly to tree volume variations in near-real time. In addition, the vendor would like to offer a feature that allows farmers to visualize a 3D representation of their orchard on their web application.

The ATLAS AppEngine technology enables the integration on the tractor side, and the ec\_extension service template provides the means to provide the data integration with the ATLAS world. For more technical details, see 7 Dynamic Behaviour.

# Service Template Functions

This section provides a very high-level summary of the ec\_extension functions:

Configuration Functions

Get Capabilities

Task Management Functions

Is Task Processable

Pre-process Application Map

Post-process Application Result

Within this section, operations are summarised 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.

ec\_extension 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. Clients requiring more intense field information are advised to implement a mirroring mechanism using the subscription functions, as outlined in **Error! Reference source not found.** **Error! Reference source not found.**.

## Configuration Functions

### Get Capabilities

Via this function, the ec\_service indicates which types of tasks it is able to process and provides information about an optional companion app (download url, permissions, …)

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

| get\_capabilities |

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

| Inputs | - |

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

| Outputs | processable task types, app |

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

## Task Management Functions

### Is Task Processable

The ATLAS Equipment Centre will determine that an ec\_service is applicable for processing a task first based on the task types it supports, and second on the results of this endpoint

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

| can\_process\_task |

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

| Inputs | advice\_urn |

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

| Outputs | yes|no |

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

If the service indicates it cannot process the task, then the ATLAS Equipment Centre will continue processing the task with the default built-in behaviour. Otherwise, the following functions will be invoked as described in

### Pre-process Application Map

Through this function, an ec\_service has the opportunity of adding proprietary information to an application map that may be of use to its companion app running in an AppEngine.

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

| pre\_process\_map |

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

| Inputs | gpkg |

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

| Outputs | gpkg |

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

Even if no customization is performed, the service is expected to return the same application map it received.

### Post-process Application Result

Through this function, an ec\_service has the opportunity to extract proprietary information from an as applied map that may have been produced by its companion app running in an AppEngine.

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

| post\_process\_results |

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

| Inputs | field urn, gpkg |

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

| Outputs | gpkg |

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

Even if no customization is performed, the service is expected to return the same application map it received.

# Data Formats

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

## Application Map File Formats

The application maps received in and returned by the pre\_process\_map endpoint are expected to be in GeoPackage (gpkg) format, compliant with the conventions detailed in the respective advisor service templates.

## Application Results File Formats

The resulting as applied maps received in and returned by the post\_process\_results endpoint are expected to be in GeoPackage (gpkg) format, compliant with the conventions detailed in the field\_data service template.

# 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 diagram.

## ec\_service with Companion App

Diagram

Description automatically generated