Writing Material Changes

This section describes the API used to submit material changes to the tracking layer.

In general, the API will use a POST REST call that involves:

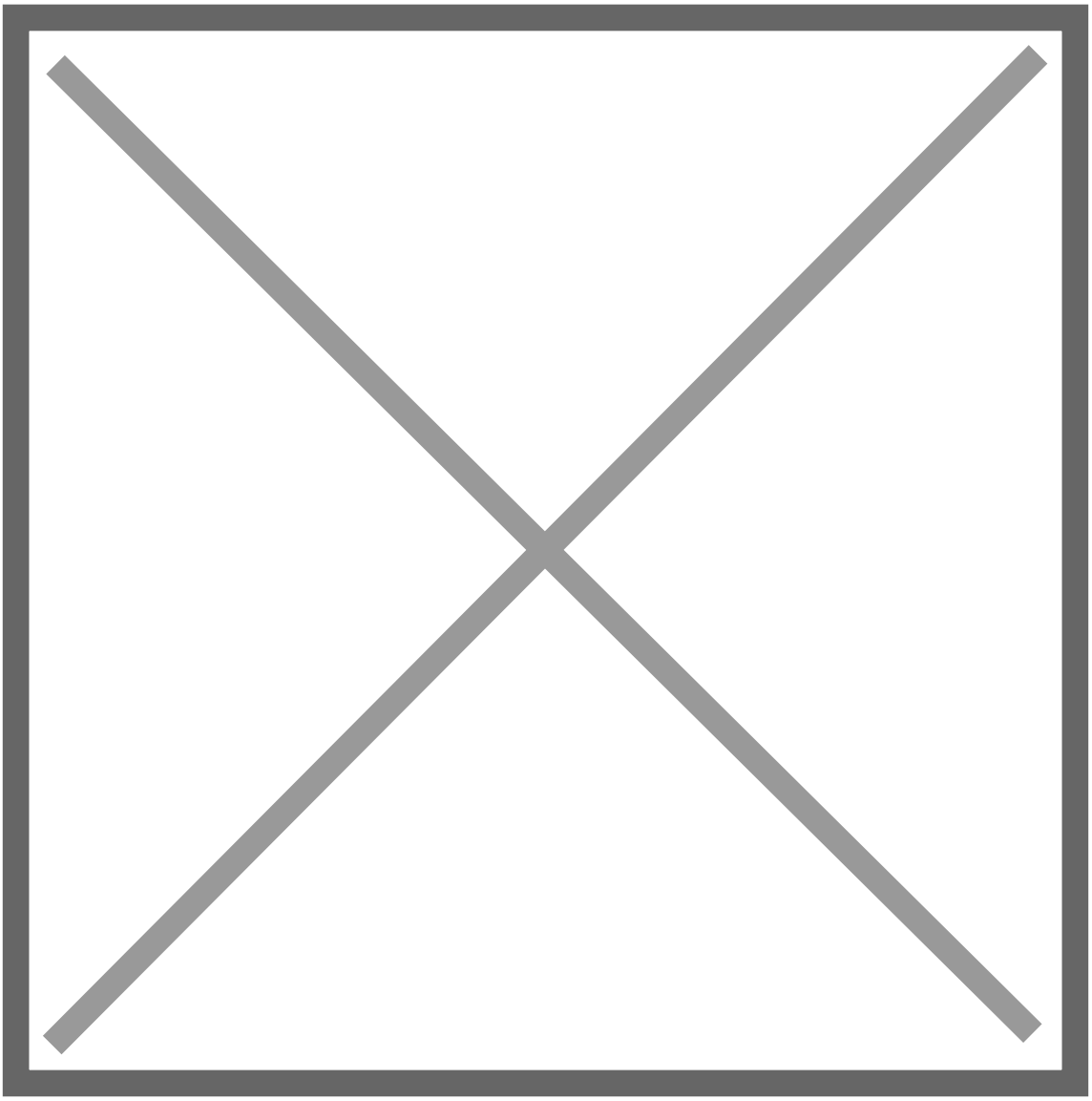
1. An event type, which can be UPDATE, SIM_START or SIM_END
2. An entity Id to identify which material tracking layer will be used
3. An HTTP callback address that will be used to notify all changes have been committed
4. A CSV file that contains a list of material operations to be performed. This file is called a “Material Operation Table” for the remainder of this document.

These REST calls only trigger the processing of material changes, which occurs asynchronously. The REST calls return immediately, with an indication of whether the request was accepted or not. The REST call includes a callback that will be used to notify the caller after the changes were actually written into the material tracking layer.

The write REST API is used in two main scenarios:

1. Committing a new list of material operations
2. Synchronizing simulations and committing their results

The following sequence diagram shows scenario (1):



The agents at play here are:

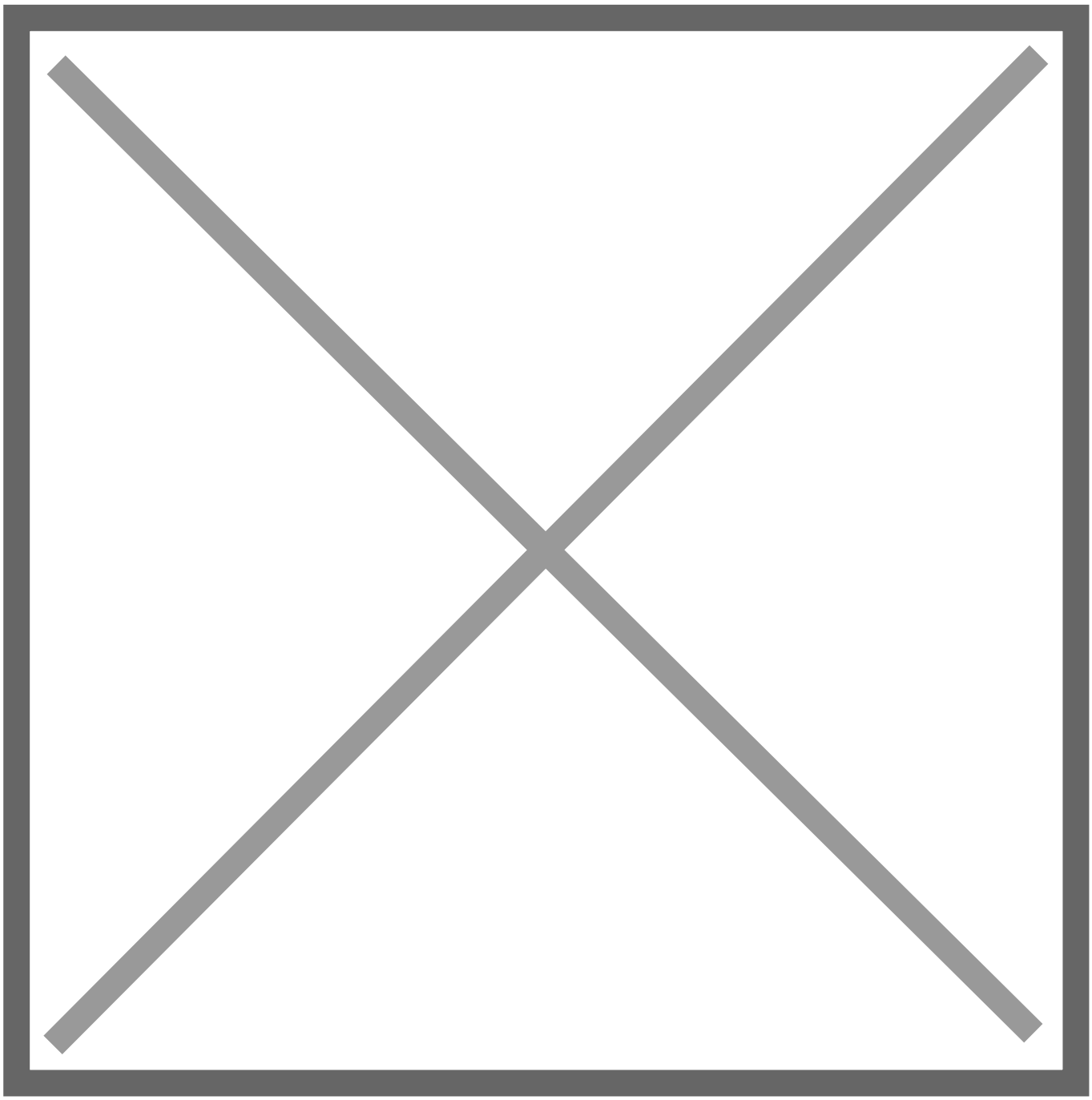
- FMS – The source of raw fleet events, and similar raw event sources
- Event Manager – This entity transforms raw events into Material Tracking events, which translate directly to calls to the material tracking REST API (SpatialDB REST in the diagram)
- SpatialDB REST – This is the front-end to the write API
- Operation Processor – This is a service that asynchronously processes material changes

The scenario shows the outcome of a single fleet event, which describes an excavator loading a bucket of material into a truck. The Event Manager uses the location of the bucket to compile a list of material changes that were produced by this event into a CSV file, where each row describes one material change operation. The Event Manager then issues a POST request to the SpatialDB REST endpoint, using the UPDATE event, asking to update the material model following the changes described in the attached CSV operation table. If the tracking system is in a determined state, meaning there are no simulations pending, a message to process and commit the operation table is lodged in the SpatialDB internal processing queue. This event is eventually picked up by the material tracking processor, which executes the requested changes. When the changes are

committed, the processor services issues a callback.

If there was a pending simulation (not shown in the diagram), the SpatialDB REST would have returned a special code that signals the Event Manager must put this request in a local queue. Then, upon receiving the callback, the Event Manager can get the next pending request from the local queue and issue the REST request again.

The second main usage scenario for the REST API is to perform simulations. Unlike updates to the material tracking, simulations are performed by agents foreign to the material tracking layer. The API includes a protocol to synchronize the external simulations with the material tracking processor.



The agents involved in the simulation scenario are the same as in the previous sequence graph, with one addition:

- Simulation Agent – An external service that can compute material changes asynchronously and submit them to the system

The scenario starts when the Event Manager receives an event from Fleet Management about a truck dumping material. Using the location of the truck, and the registered stockpile polygons, the Event Manager finds a Stockpile simulation agent that is associated with the region.

The Event Manager then makes a POST to the SpatialDB REST interface containing the START_SIM event. The POST payload includes a CSV file that specifies a Material Operation Table. The operations in this table are considered pre-requisites for the simulation to start.

The START_SIM request is then handled by the SpatialDB REST interface. If the system is not performing any simulation for the provided Material Tracking Layer, the processing of the precondition operations is queued. When submitting the job, the Event Manager can specify a callback that will be received by the Simulation Agent. Once the system completes committing the precondition operations, the callback is triggered. At this point, the Simulation Agent knows the system meets the preconditions and can start the simulation process.

The callback contains a token that allows identifying the event that originated the simulation, which the Simulation Agent can use to retrieve additional information for the event. For instance, using a truck’s singularity ID to load the voxel contents of the truck.

When the simulation completes, the Simulation Agent sends another POST request to the SpatialDB. This time, it uses the END_SIM event, and the post contains a CSV file with a Material Operation Table for the results of the simulation.

If the END_SIM event is never received within an application-defined timeout, the system assumes the simulation produced no results.

Revision #1
Created 17 March 2025 20:43:14 by admin
Updated 17 March 2025 20:44:00 by admin