INTERNATIONAL TELECOMMUNICATION UNION

TELECOMMUNICATION STANDARDIZATION SECTOR

STUDY PERIOD 2017-2020

FG-AI4A-I-#

Focus Group on AI and IoT for Digital Agriculture

Original: English

WG(s):	Working group or N/A	Los Altos, 20 APRIL 2022

INPUT DOCUMENT

Source:	HPE for Linux Foundation AgStack
Title:	Anonymized Geographical Entity Registration System
Purpose:	Discussion

Contact:	Ted Dunning HPE United States	Tel: Fax: Email:	+1-858-414-0013 none ted.dunning@hpe.com

Abstract: An anonymized geographical entity registration system that allows anybody to register a geo-polygon or to search for polygons that overlap or include a search entity. If a proposed polygon is substantially identical, except for small discrepancy, a previously registered polygon, then the identifier for the previously registered polygon will be returned that has the best match to the proposed polygon. Search for an existing polygon can be done based on best match or maximum overlap.

This service is important for organizing data in agricultural services. By not attaching any personal, private or confidential information to the polygons, the system inherently protects such data but allows a common frame of reference for user interfaces or data systems that do handle such non-public data.

1 Purpose

The service described here is intended to provide a common point of reference for geographical regions embodied as an opaque identifier. Such an identifier can be retained and used to refer to a geographical region in other services.

As an example of a use of these identifiers, a farmer can use this service to establish identifiers for plots of land that she cultivates or is interested in. These identifiers can be retained for easy reference to these plots of land or passed to services to as a key to store data.

As another example of a use for these identifiers, an identifier can be used to express a subscription interest in data that is associated with geographical points or regions. As new volumes of data are produced, individual data elements that match a particular polygon can be selected efficiently. Such a matching could be used, for instance, to allow subscription to micro-meteorological data for an area of land.

2 Service access

The service will be accessed using HTTPS using a well-known set of relative URLs relative to a base URL for the overall service.

3 Data Objects

Geo-point – a latitude, longitude pair expressed in decimal degrees with a resolution of 1µ° to allow approximately 10cm resolution. The projection of these latitude/longitude pairs to actual positions of the earth is not specified.

Geo-bound – a closed polygon on the surface of the earth. This polygon expressed as a sequence of geo-points that represent the boundary of the polygon in clockwise order. The edges between geo-points are assumed to follow great circle paths and should be non-intersecting. The polygon can be non-convex. Areas of such polygons is computed using a idealized spherical geoid which can lead to small errors which are ignored in returning results.

Polygon identifier – an opaque 96-bit unsigned integer identifier that represents a polygon. Polygon identifiers are rendered textually as 24 hexits. It will almost always be the case that different identifiers will always refer to different polygons, but it is possible for different identifiers to refer to identical polygons. In such cases, both identifiers will always be returned if one is returned.

Hexit – a hexadecimal digit, that is, one of “0123456789abcdef”.

4 Service actions

4.1 Find regions

The find region operation has the following arguments:

Argument name	***Type***	***Required?***	***Description***
bound	Geo-bound	yes	The polygon to be identified
inclusion	Decimal number in the range 0 to 1	no	The area of the intersection divided by the area of the result polygon will be at least this large. Default is 0.9
overlap	Decimal number in the range from 0 to 1	no	The area of the intersection of the result and query divided by the area of the query polygon will be at least this large. Default is 0.9
create	0 or 1	yes	If no matching polygon is found, a new polygon will be registered if create=1
accessKey	string	yes	The identifier of the workload making the request
signature	string	yes	The hash of the arguments excluding the signature keyed with the private key associated with the access key.

This single operation can be used for multiple purposes:

- To define a new polygon or find a very closely matching polygon (omit inclusion and overlap, create=1)
- To find an existing polygon, but not create a new one (omit inclusion and overlap, create=0)
- To find existing polygons that completely contain the query polygon with only minor exceptions, but which are no more than 100 times larger than the query (set overlap near 1, set inclusion to 0.01, create=0)

- To find existing polygons entire contained within the query polygon with only minor exceptions but which are no smaller than 1% of the query polygon in size (set overlap to 0.01, set inclusion to near 1, create=0)

The accessKey and signature are included to guarantee the integrity of the request and to allow metering of requests. An accessKey can be provisioned without tying it to any personally identifiable information and new accessKey’s can be generated frequently to prevent long-term pattern analysis.

This operation will return a list of elements containing the following values:

Element name	***Type***	***Description?***
region-id	24 hexit string	The identifier for an overlapping region
intersection	Decimal number in the range 0 to 1	The area intersection divided by the area of the union or result and query polygons
inclusion	Decimal number in the range 0 to 1	The area of the intersection divided by the area of the result polygon
overlap	Decimal number in the range from 0 to 1	The area intersection divided by the area of query polygon

If the original query polygon had non-zero area and has a create flag with a value of 1, then the result will always have at least one element. If create is 0, then this list may be empty. If create has a value of 1 and the result list would otherwise be empty, a new identifier should be created for the query polygon and both identifier and query polygon should be retained to be returned in subsequent queries.

4.2 Get region

The get region operation has the following arguments:

Argument name	***Type***	***Required?***	***Description***
region-id	24 hexit string	yes	The identifier for an overlapping region
accessKey	string	yes	The identifier of the workload making the request
signature	string	yes	The hash of the arguments excluding the signature keyed with the private key associated with the access key.

This operation is used to retrieve the bounds for the specified polygon.

The accessKey and signature are included to guarantee the integrity of the request and to allow metering of requests. An accessKey can be provisioned without tying it to any personally identifiable information and new accessKey’s can be generated frequently to prevent long-term pattern analysis.

This operation will return a list of elements containing the following values:

Element name	***Type***	***Description?***

region-id	24 hexit string	The identifier for an overlapping region
bounds	List of geo-points	The polygon

5 Query consistency

It should be possible to scale the service to a global scale by instantiating multiple instances of the service in multiple locations. Such globally distributed instances must return consistent results such that if no new polygons are registered with the system, the same query should produce the same results for all instances of the service.

Globally distributed instances should in all network conditions generate unique identifiers for all retained polygons. If it is possible for duplicate identifiers to be generated, the probability of such a collision should be negligible even if billions of identifiers are generated.

Furthermore, queries to a single instance of the service must exhibit read-after-write consistency in that if a first query causes retention of a polygon a second query that is issued after the first will return that retained polygon if it matches the query constraints. This must be true even if the instance is composed of a cluster of servers and the two queries are sent to different servers in the same cluster.

If two such queries are sent to different instances of the service, then the retained polygon must be returned if the second query is issued at least 30 seconds after the first and if all network links between the instances of the service are up and the system has stabilized from any previous network disturbances. In the event of some level of network partition the system should restore the consistency of retained polygons as quickly as possible.

During a network partition, different instances of the service may retain identical polygons under different identifiers. After network partitions are healed and the system has restored consistency, all such identical retained polygons should be returned in all subsequent searches if they meet the query constraints.

6 Testing and validation

The service can be tested by generating large and small polygons with known levels of overlap to verify that polygons are retained after being created and that the query qualifiers function as desired to retrieve previously generated polygons. Testing should be done at scales that range from polygons that range in scale from roughly a meter to thousands of kilometers. Test should be one with arbitrary rotations of the test polygons to increase the probability of detecting boundary conditions.

Testing should verify that polygons consisting of a single point will not be retained nor will return any results. Similarly, degenerate polygons with zero area should not be retained nor produce any query results.

7 Considerations for privacy

The accessKey and signature are included to guarantee the integrity of the request and to allow metering of requests. An accessKey can be provisioned without tying it to any personally identifiable information and new accessKey can be generated frequently to prevent long-term pattern analysis.

In addition, the server can mask the accessKeys before logging any requests or results. If desired, the key used to do this masking can be rotated frequently. These measures will make it difficult to correlate queries back to users while still allowing the system to be monitored and faults diagnosed.