PRESTAR SERVICIOS PROFESIONALES PARA ATENDER TEMAS TECNICOS RELACIONADOS PARA LAELABORACIÓNDEDOCUMENTOSDELINEAMIENTOS TÉCNICOS, DE LOS PROGRAMAS, PROYECTOS, CONTRATOS Y/O CONVENIOS A CARGODELINVIAS

INVIAS
INSTITUTO NACIONAL DE VIAS

Arquitectura cadena general de proceso: Preprocesamiento-procesamiento

INSTITUTO NACIONAL DE VÍAS TERRITORIAL ANTIOQUIA
PROYECTO: MAPA DE VULNERABILIDAD FAUNÍSTICA

Avance del tercer mes de trabajo sobre integración de toda la cadena de proceso del sistema actualizado del proyecto del Mapa de vulnerabilidad Faunística

Ing. Jairo I. Coy
Equipo de DevOps

ARQUITECTURA DE TODA LA CADENA DE PROCESOS

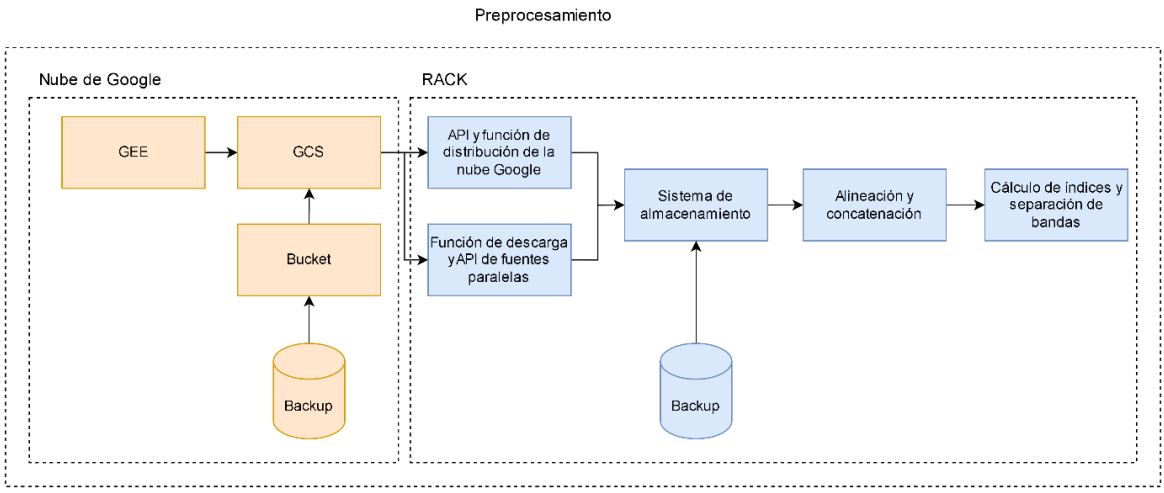
El documento muestra los componentes y arquitectura que forma toda la cadena de procesos para la generación de las nuevas capas del mapa de vulnerabilidad faunística.

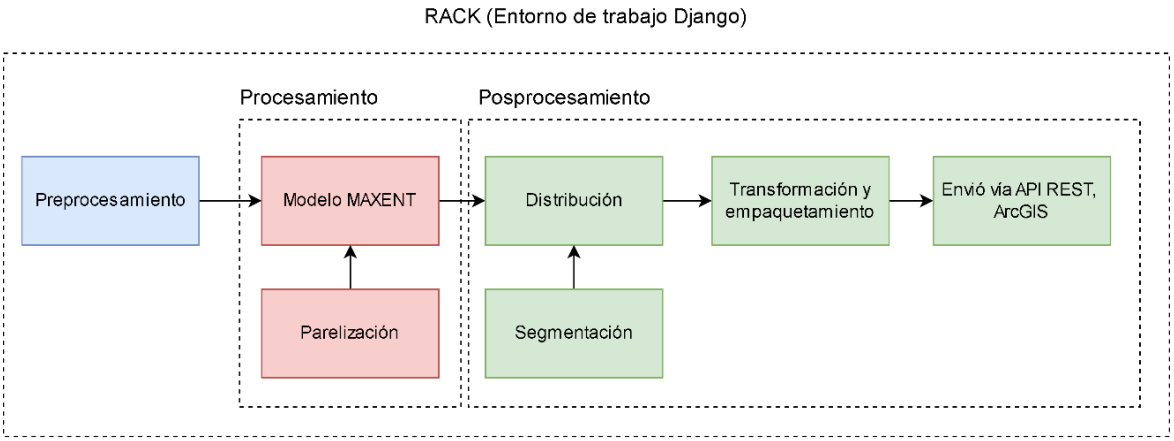
El sistema se encuentra formado por una componente en la nube y dos en servidores remotos. Se toma la nube de Google como generador de capas y almacenador y los servidores como elementos de procesamiento, almacenamiento y renderización (visualización de resultados).

El sistema de procesamiento se desarrolla bajo lenguaja Python y se trabaja con un entorno de trabajo Django. El sistema de renderización corresponde al sistema central de geo portales del INVIAS (HERMES).

Con el propósito de reducir la carga, optimizar recursos y favorecer en tiempos de respuesta, se modifica la estructura del flujo de información, trabajo de forma anidad y no independiente, como inicialmente se había diseñado.

Las actividades de procesamiento se originarán desde la nube de Google y cerrarán sus procesos en el servidor de procesamiento. Todo los relacionado a la generación de capas se distribuirá entre Google Earth Engine (GEE) y las API's alojadas en el entorno de trabajo (Django). Las capas originarias de GEE se almacenarán en un bucket de Google Cloud Storage (GCS) y serán consumidas por el sistema central de procesamiento.





El sistema central de procesamiento contiene una serie de funciones que permiten distribuir la información, segmentar los datos, realiza procesos de concatenación y ejecuta cálculos, estableciendo una separación de bandas de las capas recolectadas. Todas estas actividades, tiene el propósito de alistar la información para ser aceptada por el modelo de procesamiento y análisis central MAXENT, corresponden a la etapa de preprocesamiento.

Finalizada la acción del modelo, este genera una serie de resultados en las que se encuentra un RASTER (.tif), acorde al diseño propuesto el análisis se realizará en toda la región como conjunto y no por segmentos (una sola capa).

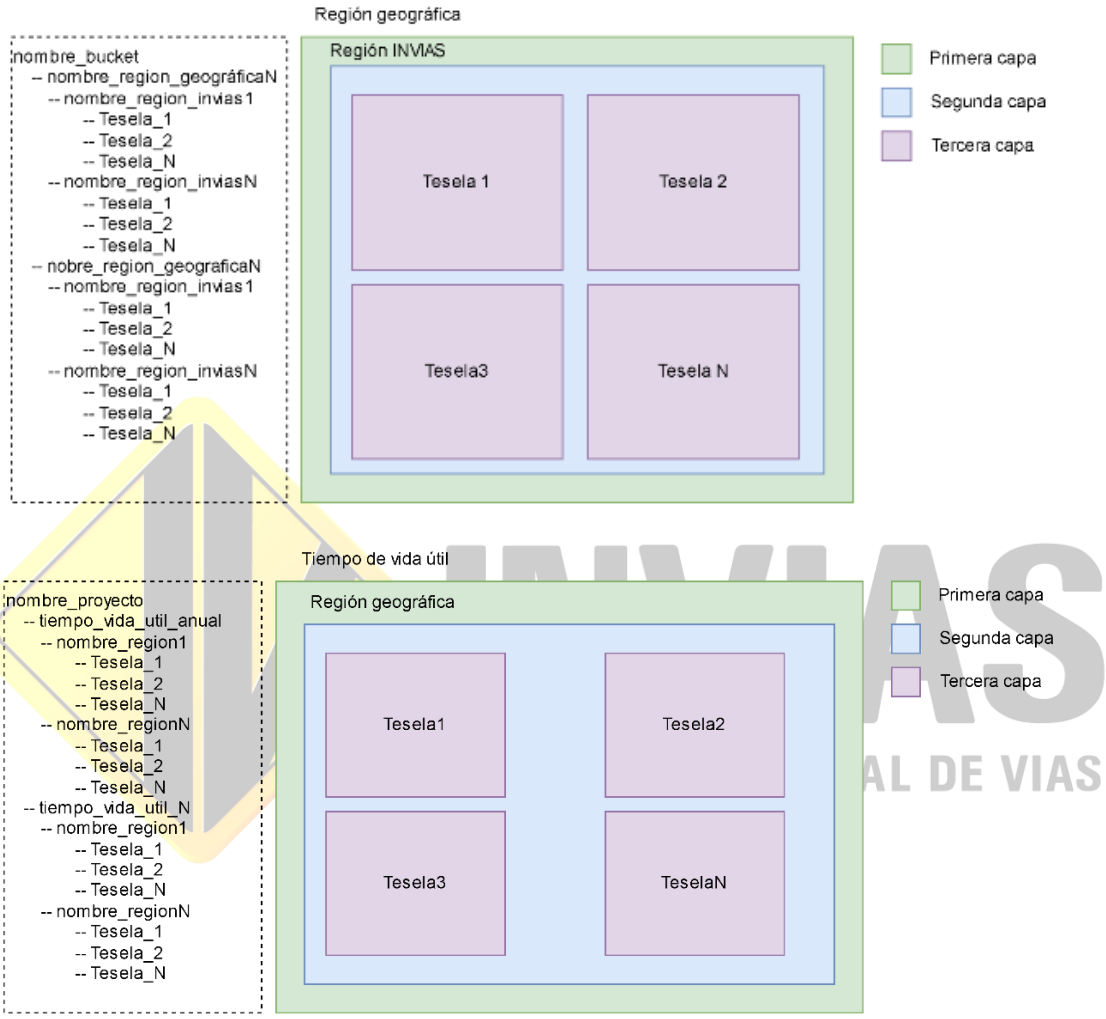
Este archivo, tendrá las nubes de puntos resultado, correspondiente a las probabilidades de riesgo animal en cada región del país en las vías registradas dentro del sistema de INVIAS en el territorio colombiano.

La creación de este archivo será detectada por el sistema, lo cual continuará hacia la ejecución de segmentación, transformación y empaquetamiento para así ser entregados al servidor donde se realizará la renderización por medio de una petición REST ya sea tipo POST o GET.

Durante esta etapa, se le asignaran valores y niveles, permitiendo de esta forma identificar las diferencias que tiene cada una de una forma gráfica, recurriendo a la integración de mapas de colores y/o coropletas.

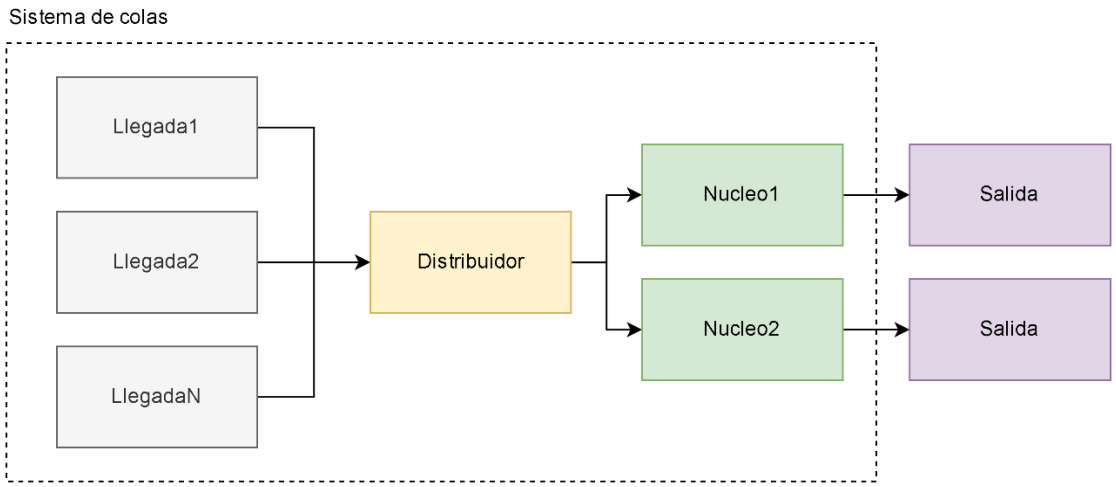
Como elementos de seguridad, cada sistema tiene sus componentes de almacenamiento, para el caso de GEE se encuentra el bucket, para el caso de Django se tiene una base de datos PostgreSQL. En cuanto a el sistema de renderización HERMES se trabaja con una base de datos empresarial PostgreSQL soportada por ArcGIS Enterprise.

El sistema de almacenamiento tiene dos flujos y arquitecturas, dado que las fuentes que se trabajan son diferentes y requieren de dos bloques de procesos. Para el caso de la información del bucket, su diseño se expone arriba, y la de fuentes paralelas se expone abajo.



Para la descarga de información, se integra una metodología por hilos (paralelización de procesos), con lo que se logra reducir el tiempo de descarga, respecto a los procesos tradicionales.

Anticipando posibles fallos debido a la saturación (escalabilidad) y pariendo la arquitectura del servidor, se integra una dinámica ejecución de actividades por sistema de colas. Su diseño se expone.



Como mecanismo de soporte se implementa un sistema de supervisión que reactive el sistema en caso de presentarse una problemática seria y se interrumpa su actividad.

El sistema en su totalidad se encuentra constituido por múltiples sistemas de seguridad. La nube como tal tiene sus mecanismos evitando que se presenten fugas de información. Django ya tiene integrado también sus mecanismos de seguridad, como también el servidor.

El sistema se diseño para evitar la intervención humana, no obstante, cuenta con su interfaz de administración, al igual que hay fuentes que no se encuentran completamente automatizadas por su origen.