

# Acta de Reunión

Fecha: Lunes, 27 de octubre de 2025

Lugar: Sesión virtual (Teams)

Proyecto: Mapa de Vulnerabilidad Faunística – INVÍAS

Ecosistema y herramientas: Google Cloud Storage (bucket), endpoints/APIs, visor base con ESRI/ArcGIS, posible GeoServer (WMS), rack de procesamiento, integración Python–R (MaxEnt), Google Earth Engine, repositorio con wiki y carpetas de documentación

Duración aproximada: 1 h 28 min

## 1. Propósito de la reunión

Verificar el estado de integración técnica del flujo bucket → procesamiento → consumo, comprobar la ejecución del motor de modelación, revisar la estrategia de publicación (endpoints/servicios) y acordar organización del almacén de datos, cargas y responsabilidades de la semana, con miras a mostrar avances al cliente en el comité de fin de semana.

## 2. Asistentes

- Luis Esteban Gómez Cadavid
- Andrés Felipe Hernández Marulanda
- Jairo Iván Coy Coy
- Nelson Aníbal Miranda Ríos
- Jasmín Marín Perez

## 3. Desarrollo de la reunión (narrativa)

Se inició con la validación del flujo de conexión al bucket en Google Cloud Storage. Se mostró un endpoint funcional que permite descargar y visualizar directamente un archivo GeoJSON desde la nube; sin embargo, al intentar consumirlo desde la vista del visor local se presentan restricciones de permisos por tráfico cruzado (CORS). Se dejó claro que en un servidor de producción esto no debería ocurrir, pero en entorno local se interponen políticas que bloquean el acceso. Como plan alterno se evaluó encapsular la capa desde el back en un servicio tipo WMS (p. ej., con GeoServer) o montar una API intermedia para que el front consuma sin problemas.

Se evidenció la ejecución del motor de modelación: la función de MaxEnt adaptada corre en el entorno y genera la estructura de carpetas y logs; por ahora no produce resultados al carecer de insumos definitivos, pero la prueba confirmó integración y dependencias. El siguiente eslabón del pipeline es completar las etapas de reclasificación, vectorización y superposición en el eje vial para obtener la capa resultante por segmentos.

Se explicó que ya existe un principio de automatización: el sistema recorre el bucket para descubrir y listar capas disponibles, preparando filtros por directorio. Para consolidar este esquema se propuso segmentar el almacenamiento por fecha de actualización, región y teselas, permitiendo limpieza y rotación programada. Esta jerarquía facilitará distinguir datasets dinámicos, semiestáticos y estáticos, y soportará un proceso de mantenimiento que elimine información caducada y la reponga según las periodicidades definidas en el inventario maestro.

El equipo discutió el alcance de la interfaz: el compromiso contractual se centra en entregar servicios y capas consumibles en el ecosistema del cliente (ArcGIS/Hermes), no en un visor final. Se acordó que el visor interno es un demostrador para seguimiento y validación, mientras que la publicación oficial se hará vía servicios. No obstante, para el comité del viernes se intentará mostrar una visualización básica si se resuelven los permisos.

En gestión de datos se revisó la estrategia de buffers y teselado. Con el recorte a la red vial y buffers operativos, el tamaño de salidas se redujo sustancialmente: para Sentinel-2, el procesamiento y recorte por red vial resultó en pesos del orden de decenas de megabytes a escala nacional, lo que indica que la solicitud de 100 GB de almacenamiento en el rack es razonable. Se recomendó mantener métricas y respaldos, y acotar que los ráster e índices se generarán y almacenarán con criterios homogéneos.

Se asignaron roles para la semana: preprocesamiento (limpieza, alineación, preparación de insumos y reglas de mantenimiento) a cargo de Jasmin, Nelson y Luis Esteban; visualización y posprocesamiento (representación y pruebas de consumo desde el front) a cargo de Jairo; coordinación con el cliente sobre la necesidad real de una interfaz gráfica a cargo de Andrés, validando expectativas de servicios vs. visor. Se programó revisión intermedia el miércoles para consolidar avances.

## 4. Agenda (simplificada)

- Conexión al bucket y consumo por endpoints; diagnóstico CORS en entorno local.
- Ejecución del motor de modelación y adecuación del flujo de salida hacia la vía (reclasificar, vectorizar, superponer).
- Organización del almacén: jerarquía fecha → región → teselas; limpieza/rotación según periodicidades.
- Estrategia de publicación: servicios para el ecosistema del cliente; visor interno como demostrador.
- Optimización de tamaños: recortes por red vial y buffers; estimación de capacidad y confirmación de 100 GB en rack.
- Plan de trabajo semanal y revisión intermedia; definición de responsabilidades por frentes.

## 5. Revisión de compromisos pasados

- Integración Python-R operativa para ejecutar MaxEnt; pendientes insumos definitivos para producir salidas.
- Repositorio con wiki activo para registrar avances y documentación técnica.
- Conexión directa a bucket validada; endpoints funcionales desde back; fallas por CORS en consumo desde front local.

## 6. Próximos pasos y posibles fechas

- Subir al repositorio la carpeta Jackknife completa (insumos y estructura) y referenciarla en la wiki – inmediato.
- Definir lista de capas a mostrar al cliente y su organización en el bucket (fecha/región/teselas) – 28/10/2025.
- Resolver la restricción CORS con prueba en servidor o encapsular en WMS/GeoServer para consumo desde el front – 30/10/2025.
- Completar la etapa de salida: reclasificación, vectorización y superposición en vía; generar una capa de ejemplo – 30/10/2025.
- Confirmar con la entidad si requieren visor o solo servicios; cerrar expectativa en comité – 31/10/2025.
- Mantener corte semanal de avances en wiki y evidencias en repositorio (logs, capturas, capas de ejemplo) – semanal, desde 29/10/2025.

## 7. Hitos / Conclusiones

- Endpoint desde bucket funcional; bloqueo CORS identificado en consumo desde vista local.
- Motor MaxEnt ejecutándose con integración Python-R; lista la estructura de ejecución a falta de insumos definitivos.
- Criterio de almacén con jerarquía por fecha, región y teselas para limpieza/rotación automatizada.

- Confirmación de estrategia de publicación orientada a servicios para el ecosistema del cliente; visor interno como demostrador.

## **8. Análisis y recomendaciones**

Conectar el front mediante servicios (WMS/GeoServer) o una API intermedia mitigará las restricciones CORS y acelerará la demostración al cliente. La jerarquía fecha → región → teselas en el bucket permitirá balancear actualización y almacenamiento, y simplifica automatizaciones de limpieza. Mantener el visor como demostrador evita desvío del alcance, enfocando el esfuerzo en servicios consumibles. Se recomienda fijar checklist por corrida (buffer, grilla, SR, NoData), registro de periodicidades en el inventario maestro y evidencias mínimas en la wiki antes de cada comité.