Acta de reunión - INVIAS (Mapa de Vulnerabilidad Faunística)

Fecha: miércoles, 24 de septiembre de 2025

Lugar: Sala de conferencias (virtual/presencial mixta)

Proyecto: Desarrollo de plataforma y modelo analítico para Mapa de Vulnerabilidad Faunística - INVIAS

1. Propósito de la reunión

Alinear la arquitectura técnica y el flujo metodológico del proyecto (ingestión, modelado y publicación), asignar permisos/roles en repositorios, y preparar un micro-entregable verificable para el comité de INVIAS el viernes.

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

Durante la sesión se acordó ordenar el trabajo del proyecto en torno a una arquitectura única (monorepo) con tres repositorios funcionales: (1) backing/servicios y gestión de datos, (2) modelos analíticos (incluyendo MaxEnt y scripts de preprocesamiento), y (3) interfaz/visualización. Se enfatizo que el foco inmediato no es construir una gran interfaz, sino un motor de distribución de datos que entregue resultados trazables, verificables y livianos para el cliente.

Se reviso la organización en equipos y permisos sobre los repositorios (clonar, aportar por ramas y control de cambios). Se definió que las observaciones sobre los diagramas de flujo se realizaran con permisos de comentario para la mayoría, y permisos de edición centralizados (Andrés, Luis Esteban y Jairo; se habilita edición a Nelson para el módulo correspondiente). Se acordó documentar en el repositorio una wiki con: librerías instaladas, versiones, endpoints, políticas de nomenclatura, y estructura de carpetas por modulo.

En metodología de procesamiento de datos, se precisó el flujo en tres etapas para las variables dinámicas desde Google Earth Engine (GEE): (a) preparar y validar el procesamiento en GEE; (b) ejecutar por Python para persistir rasteres en almacenamiento local/servidor; (c) publicar resultados mediante tile caching con GDAL (tiled pyramids/overviews) para evitar sobrecargas de Postgres cuando se consulten grandes

volúmenes raster. Para insumos estáticos (o de baja frecuencia) - p. ej., drenajes/cuerpos de agua, RUNAP/áreas protegidas, pendientes y elevaciones (DEM), coberturas/deforestación (Observatorio de Bosques, IDEAM) - se acordó mantenerlos versionados en repositorios dedicados y actualizarlos manualmente con periodicidad (usualmente anual), dejando el procedimiento en el manual de administración.

Respecto al modelo MaxEnt, se ratificó la entrada de dos familias de variables: (i) índices espectrales y derivados de teledetección (S2 u otras colecciones, p. ej. NDVI, NDBI, etc.), y (ii) derivados estáticos o cuasi estáticos convertidos a continuos vía distancias (p. ej., distancia a cuerpos de agua y drenajes, a áreas protegidas RUNAP, a focos de ganancia/perdida de bosque por año). Los hotspots de atropellamiento serán generados con técnicas de densidad/agrupamiento (KDE/DBSCAN u otra técnica robusta) y sirven como semilla espacial del entrenamiento. MaxEnt producirá un raster de probabilidades que, para visualización en el visor de INVIAS, se trasvasara a una capa vectorial de las vías, transfiriendo el valor de probabilidad por segmento o subtramo (buffer +-100 m, según validación).

Se estableció como criterio metodológico critico que todas las capas de entrada al modelo trabajaran a resolución homogénea de 10 m (acorde a Sentinel-2) y en WGS84, realizando resampling/re proyección previos para evitar distorsiones por mezcla de resoluciones (escala mínima cartografiable coherente con 10 m). Asimismo, se diferenció la temporalidad: variables dinámicas (actualización automática acotada por ventana móvil de ~365 días) vs. variables de actualización anual o multianual (actualización manual y controlada).

En cuanto al entregable visible para INVIAS, se aclaró que inicialmente el visor mostrara únicamente la capa vectorial final coloreada por probabilidad (mapa de vulnerabilidad faunística). Exponer, además, todas las capas raster insumo en el front no es requisito y sobrecargaría el servidor; por lo tanto, dichas capas quedaran para auditoria/trazabilidad en el back y como descarga de informes (p. ej., reporte de MaxEnt). Se dejo abierta la posibilidad de habilitar descargas especificas o endpoints para capas agregadas/azulejadas cuando la infraestructura lo permita.

Se identifico la necesidad de automatizar la lectura de la base de datos SUKUBUN. INVIAS reporta flujos manuales con ediciones de columnas (p. ej., especie observada), lo cual introduce inconsistencias. El equipo revisara la estructura para detectar parámetros de cambio/flags que permitan un proceso al menos semiautomatizado (p. ej., bit de actualizado), reduciendo ambigüedades entre registros modificados y no modificados. Se acordó solicitar y analizar el diccionario de datos/estructura remitida por INVIAS y diseñar filtros de ingestión acordes.

Finalmente, se acordó reportar avances semanales por micro-entregables integrados (adquisición -> procesamiento -> visualización), evitando listados atomizados por persona. Se preparará un informe para el viernes con evidencias (capturas/enlace al

visor/repositorios), y una lámina que resuma porcentaje de avance por ítem contractual (capacitación, arquitectura, automatización, modelación, etc.).

4. Agenda (simplificada)

- Ordenamiento del trabajo en monorepo y permisos por roles
- Flujo de datos: GEE -> Python (persistencia) -> Tiles GDAL -> Publicación
- Variables del modelo MaxEnt (dinámicas vs. estáticas) y hotspots
- Criterios técnicos: resolución 10 m, WGS84, resampling/reproyección
- Estrategia de visualización: solo capa vectorial de vulnerabilidad en front
- revisión y semi automatización de ingestión desde SUKUBUN (estructura/flags)
- Plan de micro-entregables semanales e informe para comité/INVIAS

5. Revisión de compromisos pasados

- Se socializo el esquema de flujo de trabajo (diagramas) y se dejo a revisión con comentarios
- Se consolido la lista de librerías y se inicio su instalación y registro en la wiki del repositorio
- Se avanzo en la prueba de publicación con repositorio conectado a Colab; pendiente integración a Django
- Se solicito a INVIAS la estructura de la base de datos para automatización; se recibió un insumo inicial por correo para revisión

6. Próximos pasos y posibles fechas

- Unificar resolución (10 m) y sistema de referencia (WGS84) en todas las capas de entrada
- Implementar pipeline GEE->Python para almacenamiento y tile caching con GDAL (pirámides/azulejado)
- Estandarizar esquema de carpetas/endpoints del back y registrar librerías/versiones en wiki
- Analizar la estructura de SUKUBUN y diseñar flags/validaciones para ingestión semiautomatizada
- Generar capa vectorial de salida (probabilidad por segmento) desde el raster MaxEnt (buffer +-100 m)
- Preparar informe para viernes con: micro-entregable visible, enlace/preview y tablero de % avance por ítem

Fecha clave: viernes 26 de septiembre de 2025 - presentación de micro-entregable e informe de avance.

7. Hitos / Conclusiones

- Se adopta arquitectura monorepo con tres repositorios funcionales y wiki vinculada

- Criterio firme: visualización inicial solo de la capa final vectorial coloreada por probabilidad
- Estandarización técnica: 10 m de resolución y WGS84 para el modelado; resampling previo obligatorio
- Se separan temporalidades: dinámicas (ventana móvil ~365 días) vs. anuales (actualización manual documentada)
- Se prioriza tile caching (GDAL) para servir rasteres y evitar sobrecarga de Postgres
- Se corrige denominación oficial del aplicativo: SUKUBUN

8. Análisis y recomendaciones

La reunión fue productiva y enfocada en resolver cuellos de botella metodológicos. La decisión de separar computo pesado (rasteres y entrenamiento MaxEnt) del front de visualización mitigara riesgos de performance y mejora la trazabilidad. El énfasis en resolución homogénea y SRID único reduce sesgos por mezcla de escalas.

Riesgos y mitigaciones: (1) Inconsistencias de SUKUBUN: diseñar validaciones de ingestión con flags y catálogos controlados (listas de especies, sinónimos), más un registro de cambios (last_update). (2) Carga del servidor por rasteres: mantener estrictamente el tile caching y políticas de TTL/expiración de cache. (3) Derivas metodológicas por capas heterogéneas: checklist de pre-flight (resolución, SRID, extent, nulls).

Tips de mejora para siguientes reuniones: llevar demo corta de micro-entregable (gif/captura o link al visor temporal), tablero de control con KPIs por ítem contractual, y un change-log en la wiki. Sugerir un Manual de administración con cadencias de actualización (dinámicas vs. anuales), y una matriz RACI por modulo (ingestión, modelado, publicación).