Universidad Tecnológica Nacional

Facultad Regional Villa María

Ingeniería en Sistemas de Información

Proyecto Final

Villa María, _____ de _____ de _____

a) DENOMINACIÓN DEL PROYECTO

Proyecto « Solum »: Sistema de Gestión de Agroecosistemas con Predicción a Futuro

b) OBJETIVOS DEL PROYECTO Y TRABAJO DE INVESTIGACIÓN DESARROLLADO

Asistir a los ingenieros agrónomos brindándoles recomendaciones en forma de planes de siembra. Los mismos le ayudarán a mantener la sustentabilidad del suelo para su preservación futura, sin dejar de lado el rendimiento de sus cultivos. Los ingenieros agrónomos serán capaces de realizar seguimientos sobre cómo se desempeña su suelo, pudiendo controlar las variables del campo con información que le permitirá tomar decisiones que contribuyan al mantenimiento de la sustentabilidad de los suelos en el largo plazo.

Abstract: To assist agricultural engineers by providing them with recommendations as sowing plans. These will help them keep the soil balance for its future preservation without setting aside the yield of the crops. Agricultural engineers will be able to track how the soil performs, being able to control the variables of the field with information that will allow them to make decisions that contribute to the long-term maintenance of the soil sustainability.

c) GRUPO DE TRABAJO:

7532 – Barsce, Juan Cruz.
7827 – Cantoni, M. Luz.
7163 – Favro, Ignacio.
6547 – Girardi, Gonzalo.

CALIFICACIÓN: _____ _____

TRIBUNAL: _____ _____ _____

FIRMAS: _____ _____ _____

Universidad Tecnológica Nacional - Facultad Regional Villa María

Ingeniería en Sistemas de Información

Cátedra: Proyecto Final

Solum – Tomo I

Integrantes Grupo 1:

Barsce, Juan Cruz (Legajo Nº 7532)

Cantoni, M. Luz (Legajo Nº 7827)

Favro, Ignacio (Legajo N° 7163)

Girardi, Gonzalo (Legajo N° 6547)

Docentes:

Zohil, Julio

Villafañe, Christian

Fecha de regularización: 2011

Contenido

Capítulo I – Informe Preliminar

0. Introducción al Proyecto

El presente proyecto final se basa en la obtención de información sobre la sustentabilidad de los suelos, e intentar mitigar los efectos negativos de la actividad agropecuaria, basándose en la simulación de estados futuros y propuestas de actividades al agricultor.

Las propuestas de actividades tienen como fin generar opciones en el agricultor que permitan extender la vida útil de sus recursos terrestres, dichas propuestas son fundamentadas mediante la utilización de conceptos firmes y documentados de inteligencia artificial, más precisamente aprendizaje por refuerzo.

A grandes rasgos, el aprendizaje por refuerzo consiste en entrenar a un agente informático para capacitarlo en la toma de decisiones de un área específica, en este caso la agricultura y sustentabilidad de los suelos.

Parte I – ¿Quiénes Somos?

1.1. ¿Quién realiza el proyecto?

El grupo de trabajo está formado por cuatro estudiantes de Ingeniería en Sistemas de Información de la UTN Facultad Regional Villa María: Barsce Juan Cruz, Cantoni María Luz, Favro Ignacio y Girardi Gonzalo.

1.2. Motivación

El objetivo del grupo de trabajo es lograr experiencia y adquirir los conocimientos necesarios en el desarrollo de sistemas de información de importante envergadura, satisfaciendo algunas necesidades de información de "INTA", y facilitando la toma de decisiones de los terratenientes.

Conjuntamente con el propósito anterior, se incluye la integración y uso de conocimientos adquiridos durante todo el cursado de Ingeniería en Sistemas de Información; específicamente aquellos pertenecientes a áreas de mayor interés, como por ejemplo, Inteligencia Artificial.

Debido a la procedencia de los integrantes del grupo, y sus intereses personales, el proyecto se orientó hacia el agro, ya que es una de las principales actividades de la zona, y más específicamente a la sostenibilidad de agroecosistemas, como fin ecológico. .....

1.3. Introducción al Informe

El siguiente informe preliminar tiene por objeto dar a conocer características importantes del proyecto de sistema de información a realizar, denominado "Solum: Sistema de Gestión de Agroecosistemas con Predicción a Futuro", tales como: quién lo va a realizar, hacia quién va dirigido, su alcance, las metodologías a adoptar, etc.

Además, incluye una breve presentación de la institución que colabora con el proyecto, el Instituto Nacional de Tecnología Agropecuaria (INTA); y los problemas de información detectados en la misma.

Parte II – INTA

2.1. Presentación de la institución

INTA (Instituto Nacional de Tecnología Agropecuaria), es un organismo creado en 1956, con el propósito de "impulsar y vigorizar el desarrollo de la investigación y extensión agropecuarias y acelerar con los beneficios de estas funciones fundamentales: la tecnificación y el mejoramiento de la empresa agraria y de la vida rural".

Es un organismo descentralizado del Estado Nacional dependiente de la Secretaría de Agricultura, Ganadería, Pesca y Alimentos (SAGPyA), creado por el Decreto-Ley 21.680 /56. Por Ley 25.641/02 le fue restituida su autarquía y las facultades que tiene asignadas en su ley de creación, en la que se establecen la integración, atribuciones y deberes del Consejo Directivo y los órganos ejecutivos y sus funciones.

Prioriza entre sus acciones la generación de información y tecnologías para procesos y productos de este vasto sector, poniendo los mismos al servicio del productor rural.

2.2. Ubicación

El INTA cuenta con 15 Centros Regionales, con 47 Estaciones Experimentales Agropecuarias y más de 313 Unidades de Extensión que cubren toda la geografía del país y cuatro Centros de Investigación con 16 Institutos.

Las unidades involucradas en el proyecto son:

- Estación Experimental Agropecuaria (EEA) – INTA Marcos Juárez

  Dirección: Ruta 12 Km 3, Marcos Juárez, Córdoba

- Unidad de Extensión (UE) – INTA Villa María

  Dirección: Tucumán 1369, Villa María, Córdoba

2.3. Objetivos de INTA

- Competitividad:

  "Contribuir a la competitividad de las cadenas agroindustriales, al incremento continuo de las exportaciones, así como al acceso a nuevos mercados".

- Salud ambiental:

  "Contribuir a la salud ambiental y sostenibilidad de los principales sistemas productivos y agroecosistemas, manteniendo la potencialidad de los recursos naturales".

- Equidad social:

  "Fortalecer la inclusión social y el desarrollo territorial, integrando las economías regionales y locales a los mercados internos e internacionales, con generación de empleos e ingresos que disminuyan los niveles de pobreza rural-urbana".

2.4. Misión

"Realizar y promover acciones dirigidas a la innovación en el sector agropecuario, agroalimentario y agroindustrial para contribuir integralmente a la competitividad de las cadenas agroindustriales, salud ambiental y sostenibilidad de los sistemas productivos, la equidad social y el desarrollo territorial, mediante la investigación, desarrollo tecnológico y extensión".

2.5. Visión

Se aspira a que el INTA sea percibido por la sociedad como:

- La Institución pública de investigación y transferencia de tecnología al servicio del Sistema Agropecuario, Agroalimentario y Agroindustrial (SA).

- La Institución inspiradora y ejecutora de la política tecnológica agropecuaria y agroindustrial.

- Protagonista trascendente del Sistema Nacional de Innovación.

- Actor relevante en el desarrollo del SA a nivel territorial, regional y local.

- Entidad que orienta principalmente a los productores agropecuarios en el acceso a las tecnologías.

- Organización que se anticipa a las demandas futuras de la sociedad y los mercados facilitando el aprovechamiento de las oportunidades del SA.

2.6. Estructura organizacional de INTA

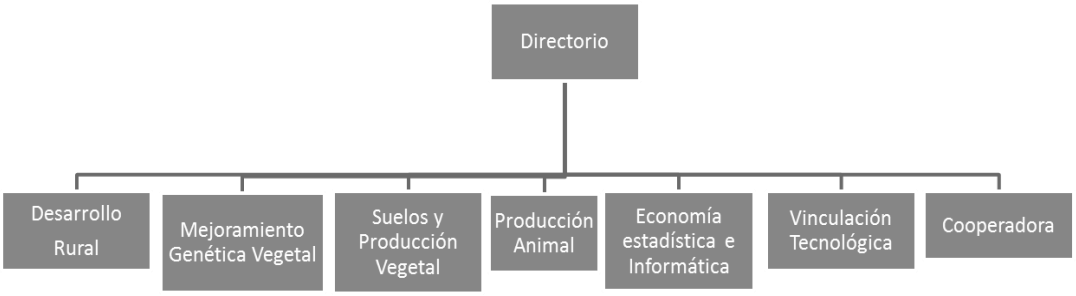
INTA cuenta con 15 Centros Regionales, con 47 Estaciones Experimentales Agropecuarias y más de 313 Unidades de Extensión que cubren toda la geografía del país y cuatro Centros de Investigación con 16 Institutos.

Las funciones principales de cada tipo de unidad son:

- Centros Regionales: Coordinar y dirigir las unidades operativas que tienen a cargo, y gestionar proyectos zonales y/o regionales.
- Estaciones experimentales agropecuarias: investigar y experimentar en terrenos propios, avances e innovaciones tanto agrícolas como ganaderas, para contribuir al desarrollo rural, el mejoramiento genético vegetal, la producción animal, conservación de suelos, etc.
- Unidades de extensión: son el último eslabón de la cadena antes del productor; establecen contacto con los mismos, los asesoran, desarrollan proyectos a nivel local como por ejemplo proHuerta, etc.

La planta de personal alcanza los 6.657 agentes. De ellos, el 44% son profesionales, el 31% son apoyo y el 25% son técnicos. Asimismo, la institución cuenta con 463 becarios (profesionales de reciente graduación).

2.7. Organigrama INTA Marcos Juárez



Descripción de cada área del organigrama

Directorio: área encargada de dirigir y coordinar y llevar a cabo la administración general de la unidad Marcos Juárez, y a su vez, representarla ante los centros regionales y otras instituciones.

Desarrollo rural: área encargada de llevar a cabo proyectos destinados directamente al productor, como por ejemplo ProHuerta, además es el enlace entre productores y la institución.

Mejoramiento genética vegetal: se divide en cuatro grupos de trabajo, dos de ellos encargados del mejoramiento de trigo y soja respectivamente, un tercero dedicado a la biotecnología, y el último es un laboratorio de calidad industrial de cereales y oleaginosas. Todos ellos se encargan de estudiar la composición interna de los distintos cereales, y mejorarla para que aumente su calidad.

Suelos y producción vegetal: área encargada de la investigación de calidad, tipos, componentes, y fertilización de suelos, teniendo en cuenta la gestión ambiental, y la agrometeorología. Para dichas tareas hay un laboratorio de química de suelos destinado

especialmente al área. Dentro de los involucrados se encuentran biólogos, auxiliares de campo, auxiliares de laboratorio, etc.

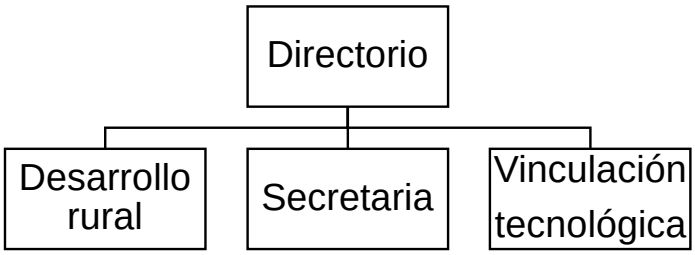
Producción animal: área encargada de evaluar la producción tanto bobina como porcina, su alimentación, la calidad de la carne, el uso de pasturas, cuestiones relativas a la bacteriología y serología, sanidad animal, etc.

Economía, estadística e informática: área encargada de realizar estudios estadísticos, e investigar cuestiones importantes para la economía agraria, la administración rural; y además realizar proyectos de sistemas informáticos relacionados con el agro y la geografía.

Vinculación tecnológica: área encargada de establecer comunicación con productores, y/o otras instituciones interesadas, con el fin de promover las innovaciones y lograr apoyo tanto técnico como económico para las mismas.

Cooperadora: área encargada de ayudar a las demás áreas productivas. Da contención personas encargadas de campo, de multiplicar la producción animal, etc.

Organigrama de INTA Villa María



Descripción de las áreas del organigrama

Las áreas que tienen en común ambos organigramas, fueron descriptas anteriormente, a continuación solamente se detallan aquellas áreas específicas del organigrama Villa María:

Secretaria: realiza tareas de administración, realización de informes necesarios, recepción de consultas de productores, etc. Puede decirse que es el área intermedia entre los productores y empresas privadas, y la organización.

2.8. Actividades desarrolladas en INTA

Desde 1956, la organización detecta necesidades, investiga, desarrolla y transfiriere tecnologías al sector agroindustrial en todo el país.

Cinco actividades son consideradas principales, y por lo tanto se describen a continuación:

- Investigación y desarrollo tecnológico

Se implementa a través de proyectos que integran las capacidades institucionales y se articulan con otras redes de actores públicos y privados del ámbito regional y nacional.

- Vinculación institucional y tecnológica

Tiene como objetivo formalizar las articulaciones sistemas nacionales y provinciales para incrementar las capacidades mutuas, activando un círculo virtuoso de formación de equipos de trabajo, equipamiento e infraestructura, al mismo tiempo que se cubren las demandas tecnológicas del SA.

- Extensión y desarrollo rural

Proceso de intercambio de información y conocimientos para el desarrollo de las capacidades de innovación de la comunidad rural. Para ello trabaja junto con otras instituciones, respondiendo a las demandas del sector agroindustrial.

- Relaciones institucionales

Tiene como objetivos: intervenir en la formalización de acuerdos interinstitucionales de cooperación científico-técnica; y promover e implementar acciones de cooperación internacional, tales como misiones técnicas, intercambio de profesionales y organización de cursos, establecidas en el marco de acuerdos internacionales.

- Servicios al productor y al sector

Brinda servicios en todo el país a través de los Institutos de Investigación, y las distintas unidades de INTA, elaborando normas, protocolos y sistemas de producción, participando en la certificación de calidad y desarrollando canales diferenciados de comercialización.

2.8.1. Descripción de actividades relacionadas con el proyecto

Dentro de las actividades descriptas anteriormente, el proyecto “Sistema de Gestión de agroecosistemas con predicción a futuro”, se relaciona directamente con las siguientes:

- **Investigación y desarrollo tecnológico**

  Se deduce la relación con esta área, ya que uno de los principales temas de interés para la misma, por su importancia estratégica, es *“La sostenibilidad como condición de competitividad. Manejo sostenible de la producción agrícola, ganadera y forestal, minimizando impactos negativos sobre los agroecosistemas”.*

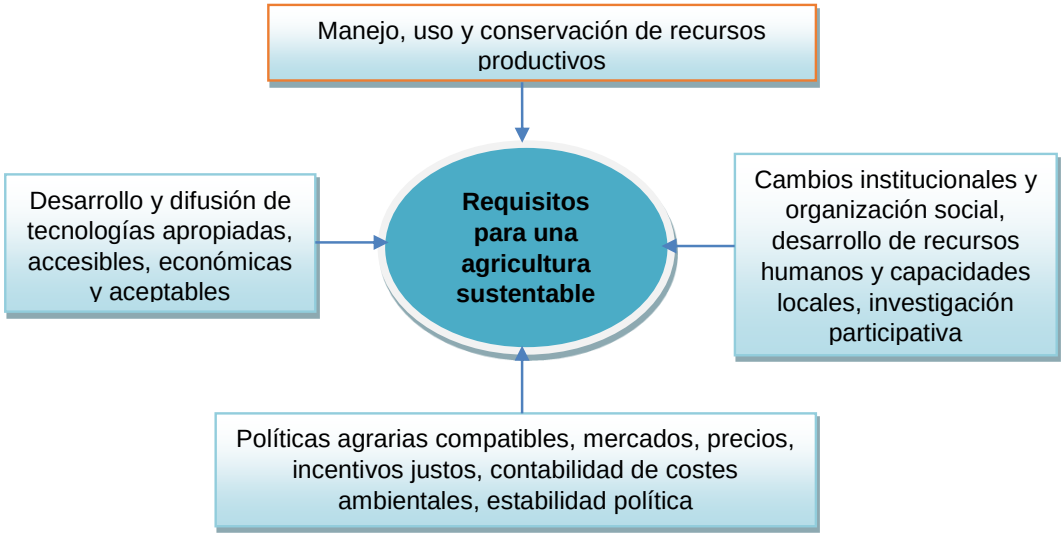
Este tema, promueve el incremento de la sustentabilidad, la competitividad y la inclusión social en los sistemas de producción.

Para una mejor comprensión, a continuación se va a definir sustentabilidad:

"el desarrollo es sustentable cuando satisface las necesidades de la presente generación sin comprometer la capacidad de las futuras generaciones para que satisfagan sus propias necesidades". (Comisión Brundtland – 1987)

Esta definición implica tener en cuenta tres dimensiones definidas: la económica, la social y la ambiental.

Los requisitos para una agricultura sustentable son los siguientes:



- **Servicios al productor y al sector**

Desde hace 15 años INTA viene implementando una política de asociación con el sector privado con el objetivo de potenciar las capacidades mutuas para la generación de innovaciones tecnológicas. Se han desarrollado los parques de innovación tecnológica (PITs) como ámbito de incubación de emprendimientos conjuntos.

Algunas de las capacidades de INTA se refieren a:

- Producción animal: análisis de fibras textiles, análisis de productos (carne, leche etc.), y análisis toxicológicos.
- Agroalimentos: certificación de productos, análisis de calidad, y análisis de residuos tóxicos.

Ingeniería: protección de cultivos y control ambiental, energía terramecánica, análisis de agua y suelos, interpretación de imágenes satelitales, y evaluación de impacto ambiental.

2.9. Unidades involucradas

Dentro de todas las unidades de INTA, el grupo de trabajo decidió pedir colaboración y apoyo de las siguientes:

- Unidad de Extensión Villa María
- Estación Experimental Agropecuaria Marcos Juárez

2.9.1. Descripción de UE INTA Villa María

Como todas las unidades de extensión, se encarga del intercambio de información y conocimientos para el desarrollo de las capacidades de innovación de la comunidad rural.

Trabaja junto a otras instituciones y respondiendo a las demandas del sector agroindustrial.

A su vez, colabora con el Programa Federal de Apoyo al Desarrollo Rural Sustentable (ProFeder), proponiendo diferentes estrategias de trabajo con distintos grupos de productores; esto último lo realiza mediante el desarrollo de subprogramas como por ejemplo, Pro-Huerta y Profam.

Cabe aclarar que INTA Villa María no realiza tareas experimentales ni de investigación, por lo cual fue indispensable para el correcto desarrollo del proyecto, la colaboración de otra unidad.

2.9.2. Recursos informáticos existentes

Actualmente el INTA Villa María no cuenta con una extensa variedad de recursos informáticos. Dispone únicamente de 5 PC de escritorio que utilizan regularmente para trabajos de oficina, además de contar con 3 impresoras y una conexión de internet comercial ADSL de 1Mb.

2.9.3. Descripción de UEE INTA Marcos Juárez

La Estación Experimental Agropecuaria Marcos Juárez posee laboratorios, gabinetes y campos experimentales que además de realizar los trabajos de investigación y/o transferencia de tecnología, disponen de capacidad para ofrecer servicios de rutina a profesionales, productores y empresas del medio.

Estos servicios se encuentran arancelados a los efectos de cubrir los costos operativos, más una retribución institucional que se destina al mantenimiento y modernización de la estructura y equipamiento de la Unidad.

La prestación de estos servicios tiene una prioridad secundaria frente a las actividades sustantivas de la Unidad, por lo tanto su realización está supeditada a la disponibilidad de tiempo y recursos que estos últimos permitan.

Otros servicios importantes brindados por INTA Marcos Juárez son:

- Evaluación de materiales genéticos

- ~ Evaluación de comportamiento a campo en ensayos de repeticiones y análisis estadísticos.
- ~ Cultivares de trigo
- ~ Cultivares de soja
- ~ Cultivares de maíz
- ~ Cultivares de forrajeras anuales

- Evaluación de agroquímicos
  - ~ Evaluación de productos químicos registrados o en vía de registración, en condiciones de campo sobre cultivos de trigo, soja, maíz o forrajeras en ensayos comparativos con repeticiones y análisis estadísticos
  - ~ Fertilizantes
  - ~ Insecticidas
  - ~ Fungicidas
- Estudios o análisis específicos
  - ~ Planificación y/o análisis económico de mediano-largo plazo
  - ~ Diseño y análisis estadísticos
  - ~ Consultas metodológicas
  - ~ Programación y/o aplicación de software para usos específicos
  - ~ Diagnóstico y planificación de emprendimientos productivos agrícolas y/o ganaderos

2.10. Problema a tratar en INTA

En las entrevistas realizadas a INTA Villa María y Marcos Juárez, se detectaron varios problemas en cuanto a manejo y deficiencia de información, pero debido al objetivo y orientación del proyecto, solamente se detalla el problema que intentará satisfacerse con el mismo:

Dentro del área de investigación y desarrollo, se encuentra un tema sumamente significativo debido a su importancia estratégica: la sostenibilidad como condición de competitividad, esto implica el manejo sostenible de la producción agrícola, ganadera y forestal, minimizando impactos negativos sobre los agroecosistemas.

El gran inconveniente involucra a:

- Productores: los cuales carecen de información del tema, y consecuentemente no toman conciencia ni realizan acciones para reparar, mantener y/o prevenir la sustentabilidad de sus tierras
- INTA: quien carece de registros relevantes al tema en cuestión, y por ende, es prácticamente imposible tomar decisiones acertadas a tiempo y concientizar a los productores y terratenientes.

Parte III – Propuesta de sistema de información

3.1. Objetivo

El proyecto tiene como objetivo principal brindar soporte a una de las principales actividades económicas del país: la agricultura. Para ello, considerando que evitar la degradación de los suelos es una de las prioridades de nuestro país y la gran cantidad de factores que influyen en la misma; se propone a Solum como un sistema que utilice técnicas de inteligencia artificial para asistir a los ingenieros del INTA (Instituto Nacional de Tecnología Agropecuaria) en la toma de decisiones en pos de preservar la sustentabilidad de los suelos en el largo plazo, sin dejar de lado la competitividad en el mercado.

3.2. A quién va dirigido

El documento va dirigido a todos los integrantes del grupo, como dirección de las actividades futuras a realizar y como documento que justifique las acciones desarrolladas. También presenta un marco referencial del porque se realiza el proyecto y cuáles son las herramientas disponibles para llevarlo a cabo.

Además contiene información de interés para los docentes de la cátedra, ya que es un compromiso plasmado en papel por los integrantes del grupo, donde se encuentran los compromisos y las directivas de funcionamiento del proyecto.

Por último, este documento permite a los usuarios y beneficiaros del proyecto estar al corriente de las actividades realizadas y tener un control sobre ellas, ya que al fin de cuentas van a influir sobre la organización.

3.3. Supuestos del proyecto

El proyecto supone por parte de los usuarios un manejo mínimo de informática, asumiendo que los mismos se encuentran capacitados para utilizar un navegador web.

Uno de los supuestos más importantes es que tanto el INTA, como productores agropecuarios, deben proveer al sistema de los datos requeridos e información fehaciente para la mejora en el entrenamiento y obtención de datos por parte del sistema.

Dicho proyecto supone además la capacitación mínima de los integrantes del equipo en el área agroeconómica y agropecuaria, debido a la estrecha relación del proyecto con las áreas mencionadas.

Finalmente, también se entiende que los usuarios tendrán disponibilidad de acceso a Internet.

3.4. Riesgos a tener en cuenta

El proyecto tiene algunos factores de riesgo importantes, ya que en gran parte se basa de agentes externos al mismo. Depende de una buena fuente de datos, que permitan al agente inteligente aprender de datos correctos, ya que un agente mal entrenado arrojará resultados incorrectos. También la simulación tiene plena dependencia de los datos históricos con los que el sistema pueda alimentarse.

Además, la calidad de las salidas está ligada al continuo aporte de datos. Tales deben ser cuidadosamente analizados, para evitar el ingreso de información incorrecta o malintencionada.

3.5. Descripción de la propuesta

La entidad seleccionada para llevar a cabo el proyecto planteó la necesidad de llevar un seguimiento del estado de suelo, es decir, en base a una serie de análisis de suelo, y a una variedad de minerales selectos, presentar informes estadísticos e históricos del estado del suelo, aprovechando la masividad de las comunicaciones informáticas para recopilar la información requerida. El grupo de trabajo, propuso en base al área de investigación seleccionada, simular estados futuros del suelo, y en base a esto, proponer mediante un componente de inteligencia artificial (IA) diversas acciones que mejoren la situación simulada, previniendo o minimizando la degradación de los recursos terrestres.

Las variables que influyen son muchas, y de ahí la necesidad de tener un control automatizado de las posibilidades que existen, y de esta manera simplificar la toma de decisiones, y es vital generar informes simples para motivar la participación de la comunidad, para generar una fuente de información confiable y amplia.

El proyecto brinda una herramienta de gran utilidad a la comunidad agrícola, base de la economía de nuestro país, y como tal, fuente principal de externalidades de la comunidad argentina en su totalidad.

Es importante destacar la plena factibilidad del proyecto, porque cuenta con factores como aprobación de los docentes de la cátedra, necesidad y plena cooperación por parte de los usuarios, y una extensa disposición de datos históricos como base del proyecto.

3.6. Metodologías a adoptar

Para el desarrollo de un producto que satisfaga las expectativas del cliente, resulta muy importante especificar cuál es el proceso de desarrollo de software adaptado en el presente proyecto. El seleccionado es el Proceso Unificado de Desarrollo (PUD), una sólida opción para construir el sistema.

La decisión de utilizar el PUD se justifica, en primera instancia, por el amplio conocimiento del proceso y de su herramienta central, UML, por parte del equipo, puesto que el mismo es el ampliamente enseñado en la carrera.

En segundo lugar, el PUD está fuertemente enfocado en documentar detalladamente cada una de las fases del proyecto, además de sus iteraciones y versiones, permitiendo al equipo enfocarse concretamente en cada una de las iteraciones, realizando en cada una las pertinentes mejoras de requerimientos, análisis, diseño, implementación, test y despliegue.

Adicionalmente, debido a la complejidad del dominio de los campos y de inteligencia artificial para poder procesarlos, el PUD permite que los requerimientos sean refinados a lo largo de todo el proyecto, lo que le otorga al equipo la flexibilidad de poder acudir a los expertos en inteligencia artificial y del INTA para validar constantemente los mismos.

A todo lo anterior es preciso agregar que UML recalca la utilización de componentes, los cuales deben respetar la alta cohesión y el bajo acoplamiento para poder ser reutilizados de forma exitosa. Esta característica es de sumo valor para el equipo pues para lograr un buen funcionamiento y una comprensión excelente los componentes deben respetar estas características, sobre todo el componente de inteligencia artificial.

Otra ventaja que proporciona el PUD a nuestro proyecto es que en cada decisión se involucra a todo el equipo, algo que encaja perfectamente con los deseos de los integrantes de nuestro grupo.

Si bien es conocido que el modelo del PUD es uno de los más complejos de los procesos de desarrollo, es el que más se apega a nuestras necesidades y por lo tanto es el optado por los integrantes del equipo.

3.7. Requerimientos detectados

Aclaración: Manipular incluye:

- Crear una entidad se entiende a la acción de registrar una nueva entidad en el sistema.
- Modificar una entidad se entiende a la acción de cambiar al menos el valor de un parámetro en la configuración de una determinada entidad.
- Dar de baja una entidad se entiende a la acción de registrar que la misma ya no volverá a ser tenida en cuenta por el sistema. Esto implica que se restringirá su acceso, y no necesariamente que sus datos serán eliminados del sistema.

3.7.1. Requerimientos funcionales

1- Administración de usuario: El sistema debe ser capaz de manipular usuarios.

2- Administración de roles: El sistema debe ser capaz de manipular los distintos roles que serán asignados a los usuarios.

3- Administración de tipos de cultivo: El sistema debe ser capaz de manipular los distintos tipos de cultivo (oleaginosa, por ejemplo) que serán asignados a los diferentes cultivos que se hallen en el mismo.

4- Administración de nutrientes: El sistema debe poder manipular los distintos tipos de nutrientes (macronutrientes primarios y secundarios, como el Nitrógeno, y micronutrientes como el Zinc) que pueda presentar la tierra del agroecosistema.

5- Administración de cultivos: El sistema debe poder manipular los distintos cultivos (soja, por ejemplo) pertenecientes a su respectivo tipo de cultivo registrados en el mismo.

6- Administración de fertilizantes: El sistema debe poder manipular los ingresos de los distintos fertilizantes que se aplican en los suelos en momentos determinados.

7- Administración de análisis de suelos: El sistema debe poder cargar los análisis de suelos realizados en un campo previamente registrado en el sistema, modificarlos y darlos de baja si fuera necesario. Además, debe poder calcular la evolución futura de la sustentabilidad del mismo y sugerir una acción preventiva previamente calculada para mantenerlo sustentable.

8- Administración de componentes del suelo: El sistema debe poder manipular los distintos componentes orgánicos e inorgánicos que yacen en el suelo.

9- Administración de campos: El sistema debe poder manipular los distintos campos cuya tierra será objeto del análisis del sistema.

10- Administración de País, Provincia y Localidad: El sistema debe poder manipular las distintas ubicaciones discriminadas por País, Provincia y Localidad.

11- Administración de sesiones de usuario: El sistema debe registrar y dar soporte a las sesiones de usuario, permitiendo que los usuarios autorizados inicien y cierren sesión.

12- Generar informe de estado de suelo a futuro:

- Realizar cálculos porcentuales.
- Detectar parámetros críticos.
- Generar gráficos.

El requerimiento engloba la necesidad de proceso de cálculos básicos para la generación de los gráficos que muestra el reporte, la simulación está contemplada.

13- Generar Informe de evolución de suelos: incluir un informe gráfico con información histórica de la evolución del suelo, en base a datos no simulados.

14- Generar listado de análisis de suelos disponibles: incluir listado de los análisis de suelos con los que cuenta el sistema para las distintas actividades. La posibilidad de ver los listados varía acorde al rol de usuario.

15- Generar mapa de situación de suelos: Se genera un mapa con referencias acorde a la zona y el estado del suelo.

16- Generar informe de actividades de usuarios: incluir las últimas actividades por los usuarios, como método de seguridad y fiabilidad de la información.

17- Generar listado de usuarios: Listado completo o aplicando filtros de los usuarios con los que cuenta el sistema.

3.7.2. Requerimientos no funcionales

1. La Interfaz Gráfica de Usuario debe ser lo más amigable posible, con íconos bien ubicados y nombres de acciones descriptivas.

2. El estado actual del suelo debe ser representado con gráficos que muestren en colores los niveles de nutrientes actuales, nivel mínimo de nutrientes, etcétera.

3. Los mapas deben presentar con colores diferentes aquellas zonas críticas y delimitaciones de regiones vecinas con características de suelo diferente.

Capítulo II – Work Breakdown Structure

1. Introducción

El presente documento tiene como objeto mostrar el alcance del proyecto. Para ello se utilizó una descripción jerárquica de las actividades utilizando como herramienta la Estructura de Descomposición de Trabajo (Work Breakdown Structure, WBS).

La última versión de la WBS es lo primero que se expone a continuación. Por motivos de espacio en la página, podrá verse hasta un sub-proyecto ampliado en cada una de ellas.

En el calendario se detallan las tareas a realizar, con sus duraciones y fechas de comienzo y fin, acorde a lo especificado en la WBS.

2. Planificación

2.1. Sub-proyecto: Sistema de gestión de agroecosistemas terminado

Entregable: Modelo de negocio finalizado
Actividades:

- Glosario de dominio: en esta actividad se listan y describen los términos desconocidos, o conocidos solamente por expertos del dominio, y sus definiciones. Para realizar lo dicho, se consulta a expertos y a toda persona necesaria para brindar información al respecto.
- Informe preliminar finalizado: Esta actividad consta de una investigación de la estructura formal e informal, definición de procesos críticos, problemas encontrados y recopilación de información relevante de la entidad destinataria del proyecto.

- **Entregable: Modelo de requerimientos finalizado**

Actividades:

- Realización de casos de uso de sistema finalizada: Esta actividad comprende la identificación de los casos de uso esenciales y de soporte, y sus actores.
- Descripción de CU de sistema finalizada: en esta instancia se identifican los casos de uso esenciales, y se realiza un listado con los mismos, y otro con los actores detectados.
- Confección de prototipos terminada: Los prototipos son una fuente útil de comunicación y permiten una mayor detección y corrección de requerimientos del producto. Se generarán prototipos que demuestren las funcionalidades del proyecto y que permitan al cliente, ajeno al área informática, validar si las necesidades fueron correctamente interpretadas por los desarrolladores. Se utilizarán métodos ágiles de prototipos, como bosquejos en papel, y para aquellas interfaces gráficas más importantes se pueden presentar algunos prototipos ya implementados en algún lenguaje pero sin funcionalidad.
- ERS terminada: Consiste en realizar el documento que se utilizará como contrato entre la organización y los desarrolladores. Esto implica, pero no se limita a, plasmar la definición y descripción de los requerimientos funcionales y no funcionales, restricciones, cuestiones legales y supuestos. Una vez confeccionado será validado por los responsables de la organización y corregido según sea necesario.
- Confección de prototipos terminada: Los prototipos son una fuente útil de comunicación y permiten una mayor detección y corrección de requerimientos del producto. Se generarán prototipos que demuestren las funcionalidades del proyecto y que permitan al cliente, ajeno al área informática, validar si las necesidades fueron correctamente interpretadas por los desarrolladores. Se utilizarán métodos ágiles de prototipos, como bosquejos en papel, y para aquellas interfaces gráficas más importantes se pueden presentar algunos prototipos ya implementados en algún lenguaje pero sin funcionalidad.

Entregable: Modelo de análisis.

Artefactos:

- **Diagramas estructurales terminados.**
  - Realización de diagramas de clases: Consiste en definir las distintas clases de interfaz, entidad y control que cimentarán la estructura del sistema. Esto incluye definir las clases, sus relaciones, atributos y sus responsabilidades.
  - Identificación de paquetes: En esta actividad se identificarán los distintos paquetes de clases que se incluirán en los distintos modelos y diagramas.

- **Diagramas comportamentales.**
  - Realización de diagramas de actividad (secuencia y comunicación): Consiste en confeccionar diagramas que expongan cómo se comportará el sistema ante distintas situaciones (que sean complejas o ameriten una profunda comprensión), en términos de actores, objetos y los mensajes enviados. Según sea necesario se enfatizará la relación temporal de los mensajes (por medio de diagramas de secuencia) o bien su organización (por medio de diagramas de comunicación).
  - Realización de diagramas de transición de estados: Se identificarán estados y sus respectivas relaciones para todas aquellas actividades que se comporten como máquinas de estados.

Entregable: Modelo de diseño

Actividades:

- Realización de diagramas de clases de diseño: Esta actividad consiste en extender los diagramas de análisis a sus correspondientes diagramas de diseño. Esto implica definir las clases de diseño a utilizar, definir los métodos, tipos de datos y parámetros de cada clase y sus relaciones con sus pares de diseño.
- Realización de mapeo a modelo relacional: Implica (sin limitarse a) definir las entidades creadas a partir de cada clase, sus relaciones, cardinalidad y claves primarias y foráneas.

Entregable: Modelos ejecutables

Actividades:

- Especificación de convenciones de programación: Se definirán todos aquellos términos, estándares, notaciones y estilos que se usarán para que cada uno de los componentes codificados por los distintos desarrolladores tengan una estructura uniforme.
- Definición de plataformas de trabajo: Se determinarán las plataformas sobre las cuales se realizará o se apoyará la codificación (esto es, lenguajes de programación, sistemas operativos, herramientas, etc.).
- Codificación: Se procederá a transformar los requerimientos en código comprensible para un lenguaje de programación.
- Confección de manuales de usuario: Consiste en realizar manuales que detallen la funcionalidad del sistema y al mismo tiempo sean lo suficientemente entendibles por quienes harán uso del sistema.
- División de trabajo: Consiste en dividir qué componentes se encargará de hacer cada desarrollador, especificando qué precondiciones y pos condiciones debe presentar cada uno en las entradas y salidas que reciba.

Entregable: Modulo inteligente integrado

Consiste en integrar el sistema de gestión con el módulo inteligente [ver Sub-proyecto: Componente inteligente finalizado].

Entregable: Modelo de despliegue terminado

Actividades:

- Identificar nodos y configuraciones de red: Consiste en identificar cada uno de los nodos y configuraciones de red en las que funcionará físicamente el sistema.
- Realizar esquemas con nodos identificados: Consiste en diagramar dichos nodos y configuraciones en diagramas que permitan ver su disposición en conjunto.

Entregable: Sistema completamente probado

Actividades:

- Especificación de casos de prueba: Se definirán las diferentes pruebas que se le efectuarán al sistema, con sus entradas y salidas esperadas, condiciones de ejecución y evaluación que se le efectuará.
- Realización de pruebas unitarias: Se efectuarán pruebas en cada uno de los distintos componentes del sistema en busca de errores.
- Realización de pruebas de integración: Se efectuarán pruebas entre distintos módulos para evaluar cómo funcionan en conjunto en busca de errores.
- Realización de pruebas del sistema: Se probará en distintas instancias el sistema completamente integrado con todos sus módulos para evaluar su funcionamiento en busca de incidencias.
- Confección de informe de pruebas: Se detallarán en un informe todos los detalles de las pruebas y correcciones realizadas.
- Corrección de incidencias: Se realizarán las acciones correspondientes para solucionar las incidencias detectadas en las pruebas anteriormente realizadas.

2.2. Sub-proyecto: Conclusiones investigación.
Actividades:

- Personal con conocimiento de opciones de modelado e implementación: incluye reuniones con expertos en inteligencia artificial, lectura de bibliografía recomendada acerca de las diferentes opciones disponibles para modelar e implementar el componente inteligente, con sus ventajas y desventajas. Esta actividad finaliza con la realización de un informe con dichas opciones descriptas.
- Personal capacitado para implementar componente inteligente: como primera medida se debe seleccionar la mejor opción, que va a ser la que va a implementarse y desarrollarse. Esta actividad también incluyen capacitaciones del grupo de trabajo para adquirir los conocimientos y habilidades necesarias para desarrollar el componente inteligente. Dentro de dicha capacitación, también debe leerse bibliografía referente, recomendada por expertos del tema.

2.3. Sub-proyecto: Componente inteligente finalizado.
Actividades:

- Diseñar abstracciones modeladas: Consiste en utilizar el conocimiento de expertos para diseñar el agente, la meta, el entorno, la recompensa y su representación de los estados.
- Transformar el conocimiento de expertos en unidades de conocimiento: Esta actividad consiste en plasmar el conocimiento y la experiencia de distintos profesionales especializados en la materia en modelos de razonamiento que le permitan al agente establecer políticas de decisión que serán utilizadas para resolver problemas futuros.
- Implementación del prototipo del agente: Consiste en crear un ejecutable inicial del módulo de aprendizaje por refuerzo, que permita efectuar simulaciones verificar la curva de aprendizaje del sistema y las decisiones que tomaría en cada situación.
- Ejecución de simulaciones: Consiste en efectuar pruebas (basadas en datos verídicos) del módulo de aprendizaje por refuerzo que le permitan adquirir entrenamiento en la toma de decisiones.
- Evaluación de resultados y corrección: Tras cada simulación se controlará el agente, sus políticas y sus decisiones tomadas, para efectuar mejoras en su proceso de aprendizaje.

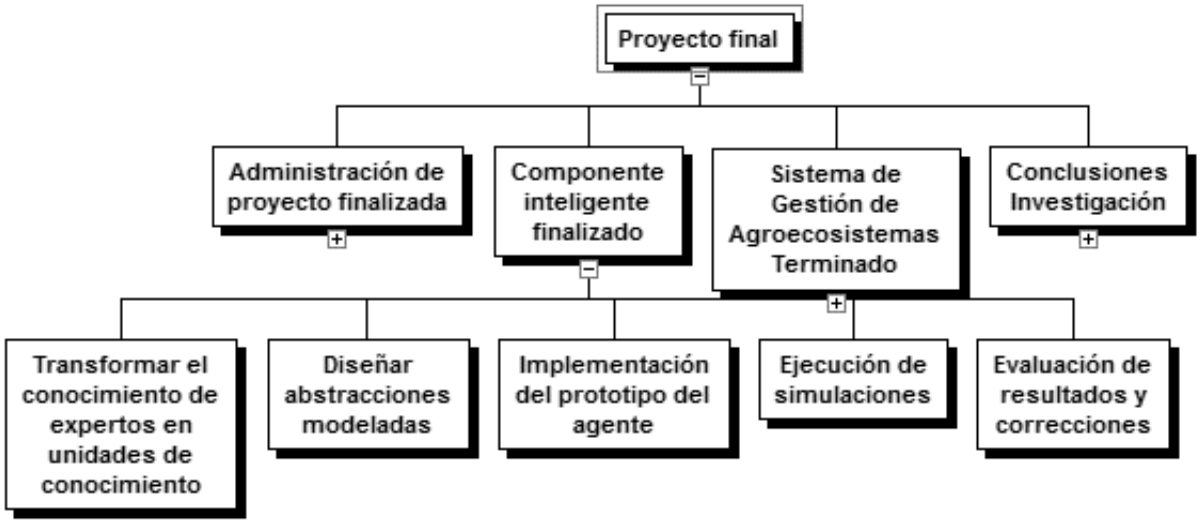
2.4. Sub-proyecto: Administración del proyecto finalizada.
Actividades:

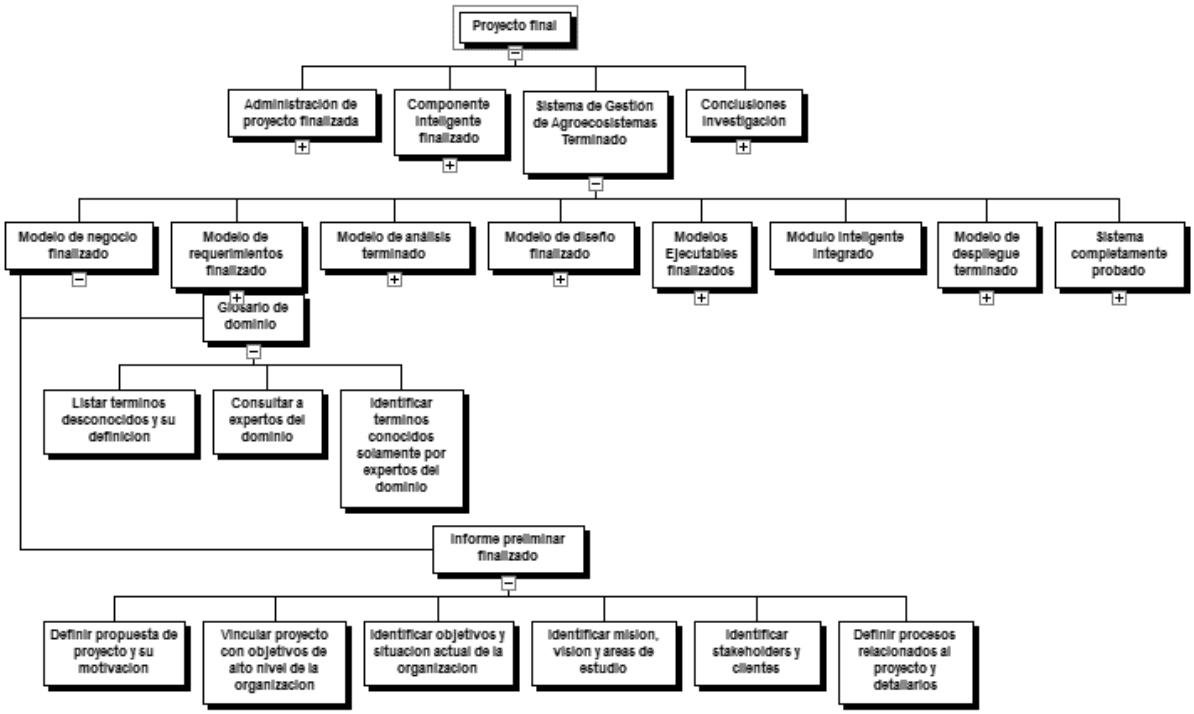
- Proyecto planificado: esta actividad incluye la realización de WBS con las tareas, actividades, entregables y sub-proyectos incluidos en el proyecto; además la definición de un calendario de proyecto y los recursos necesarios para su realización; y a modo de conclusión, la realización de un informe con toda la planificación.

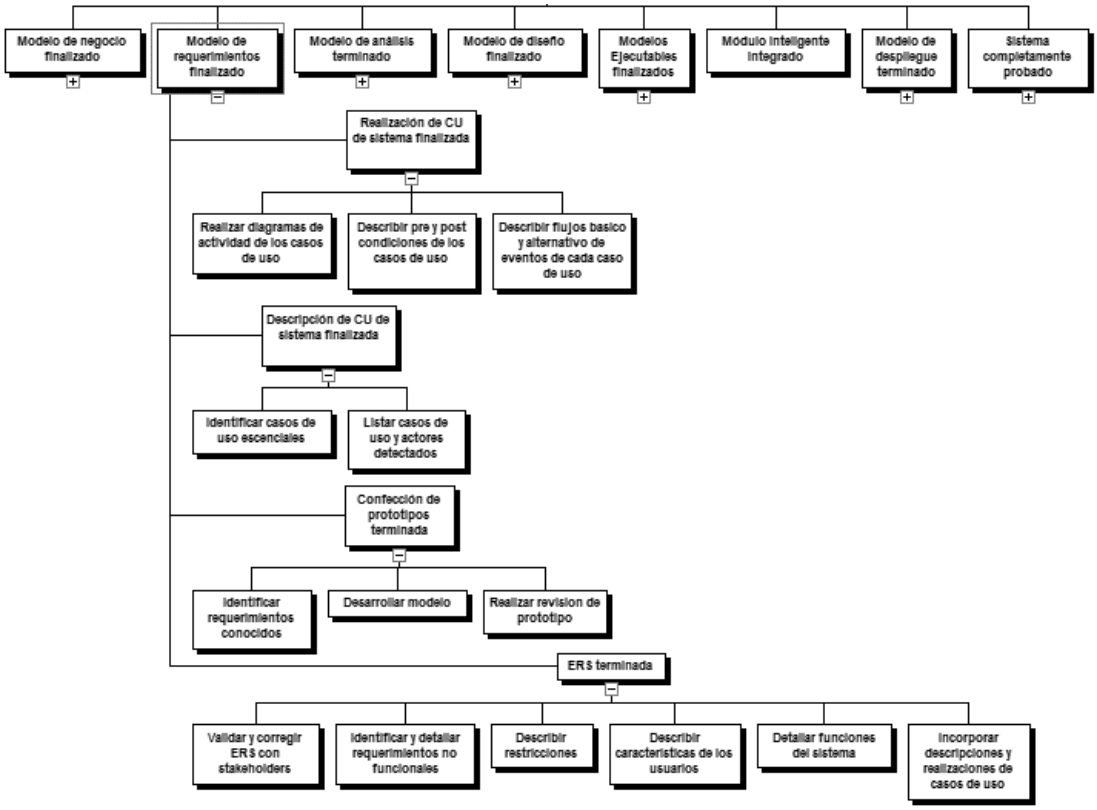
3. Work Breakdown Structure

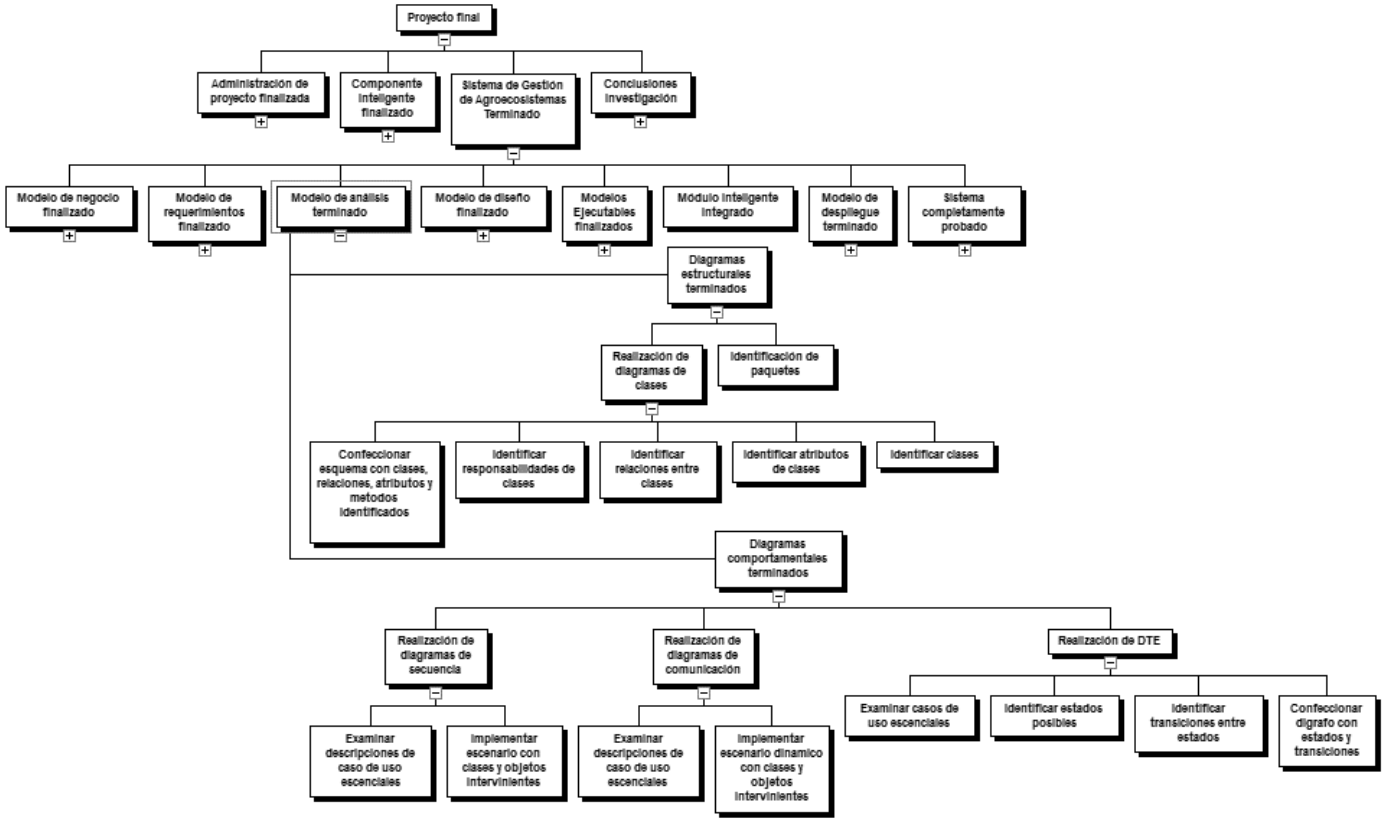
A continuación se muestra el esquema de WBS, ampliando cada sub-proyecto.

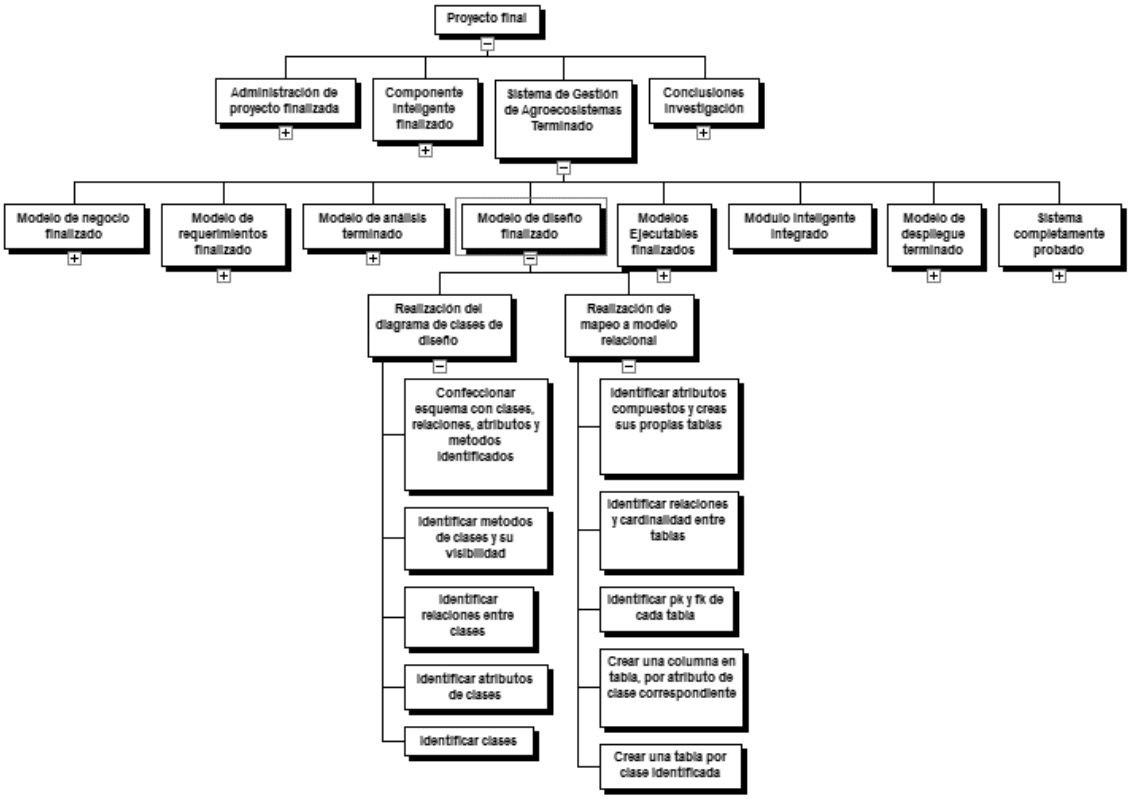

Proyecto final
Administración de proyecto finalizada
Componente inteligente finalizado
Sistema de Gestión de Agroecosistemas Terminado
Conclusiones Investigación
Proyecto planificado
Personal con conocimiento de opciones de modelado e implementación
Personal capacitado para implementar componente inteligente
Realizar informe planificación
Realizar WBS
Definir calendario y recursos de proyecto
Reunión con experto
Lectura de bibliografía
Confección de informe con opciones
Selección de opción a implementar
Capacitaciones
Lectura de bibliografía

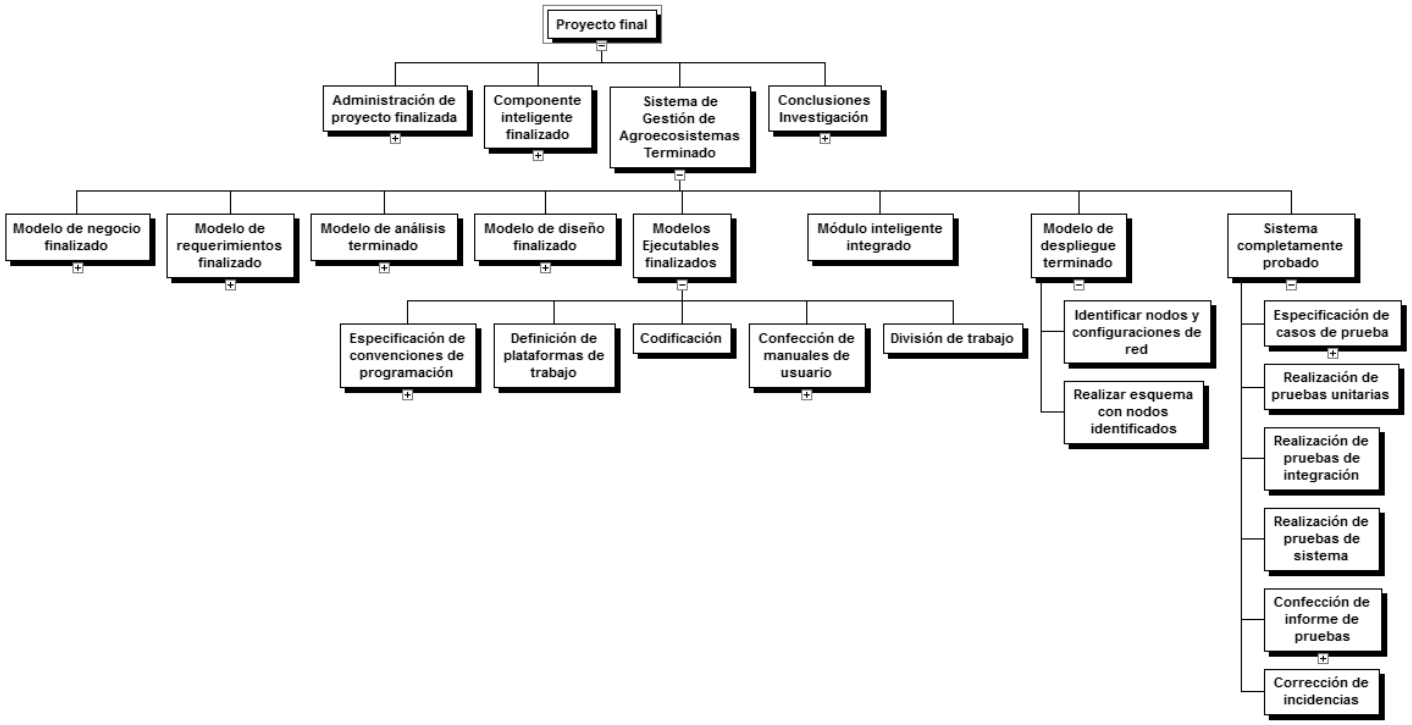

Proyecto final
Administración de proyecto finalizada
Componente inteligente finalizado
Sistema de Gestión de Agroecosistemas Terminado
Conclusiones Investigación
Transformar el conocimiento de expertos en unidades de conocimiento
Diseñar abstracciones modeladas
Implementación del prototipo del agente
Ejecución de simulaciones
Evaluación de resultados y correcciones

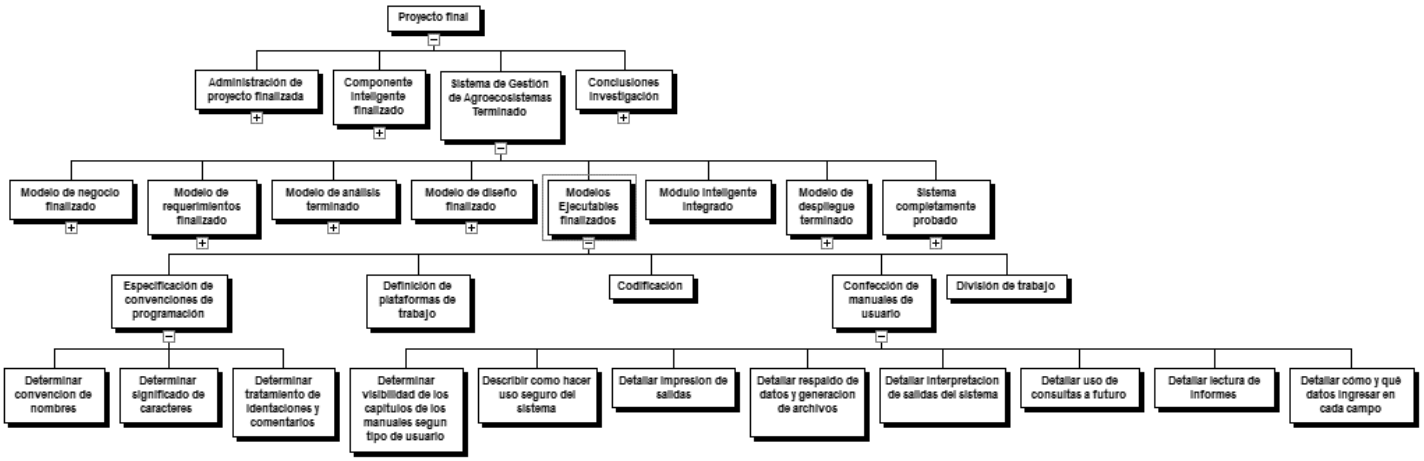

Proyecto final
Administración de proyecto finalizada
Componente inteligente finalizado
Sistema de Gestión de Agroecosistemas Terminado
Conclusiones investigación
Modelo de negocio finalizado
Modelo de requerimientos finalizado
Modelo de análisis terminado
Modelo de diseño finalizado
Modelos Ejecutables finalizados
Módulo inteligente integrado
Modelo de despliegue terminado
Sistema completamente probado
Glosario de dominio
Listar terminos desconocidos y su definicion
Consultar a expertos del dominio
Identificar terminos conocidos solamente por expertos del dominio
Informe preliminar finalizado
Definir propuesta de proyecto y su motivacion
Vincular proyecto con objetivos de alto nivel de la organizacion
Identificar objetivos y situacion actual de la organizacion
Identificar mision, vision y areas de estudio
Identificar stakeholders y clientes
Definir procesos relacionados al proyecto y detallarlos

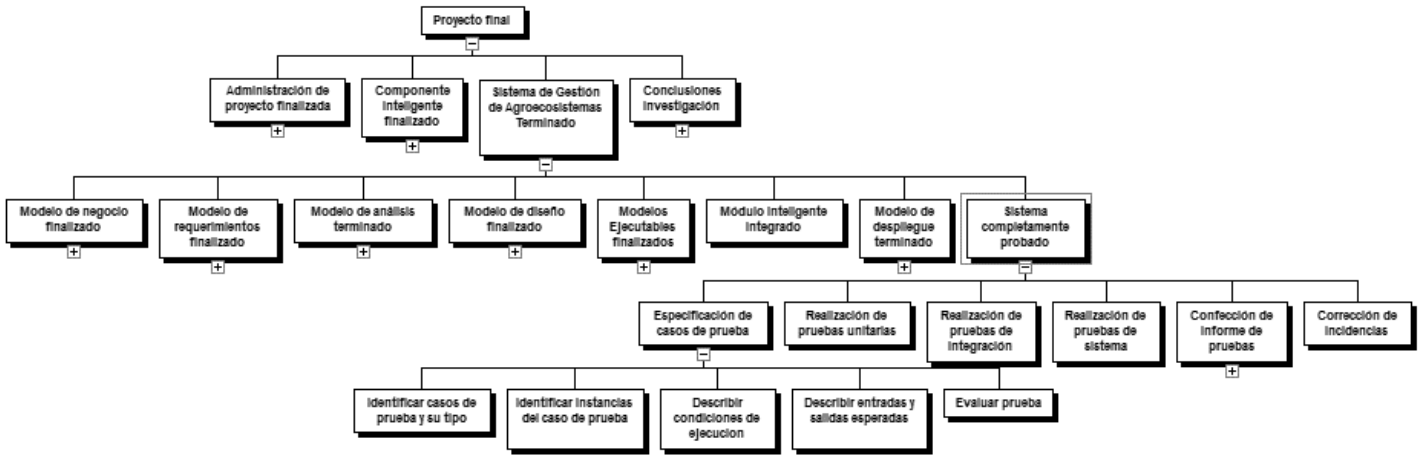

Modelo de negocio finalizado
Modelo de requerimientos finalizado
Modelo de análisis terminado
Modelo de diseño finalizado
Modelos Ejecutables finalizados
Modulo inteligente integrado
Modelo de despliegue terminado
Sistema completamente probado
Realización de CU de sistema finalizada
Realizar diagramas de actividad de los casos de uso
Describir pre y post condiciones de los casos de uso
Describir flujos basico y alternativo de eventos de cada caso de uso
Descripción de CU de sistema finalizada
Identificar casos de uso esenciales
Listar casos de uso y actores detectados
Confección de prototipos terminada
Identificar requerimientos conocidos
Desarrollar modelo
Realizar revision de prototipo
ERS terminada
Validar y corregir ERS con stakeholders
Identificar y detallar requerimientos no funcionales
Describir restricciones
Describir caracteristicas de los usuarios
Detallar funciones del sistema
Incorporar descripciones y realizaciones de casos de uso


Proyecto final
Administración de proyecto finalizada
Componente Inteligente finalizado
Sistema de Gestión de Agroecosistemas Terminado
Conclusiones investigación
Modelo de negocio finalizado
Modelo de requerimientos finalizado
Modelo de análisis terminado
Modelo de diseño finalizado
Modelos Ejecutables finalizados
Módulo Inteligente integrado
Modelo de despliegue terminado
Sistema completamente probado
Diagramas estructurales terminados
Realización de diagramas de clases
Identificación de paquetes
Confeccionar esquema con clases, relaciones, atributos y metodos identificados
Identificar responsabilidades de clases
Identificar relaciones entre clases
Identificar atributos de clases
Identificar clases
Diagramas comportamentales terminados
Realización de diagramas de secuencia
Realización de diagramas de comunicación
Realización de DTE
Examinar descripciones de caso de uso esenciales
Implementar escenario con clases y objetos intervinientes
Examinar descripciones de caso de uso esenciales
Implementar escenario dinamico con clases y objetos intervinientes
Examinar casos de uso esenciales
Identificar estados posibles
Identificar transiciones entre estados
Confeccionar digrafo con estados y transiciones


Proyecto final
Administración de proyecto finalizada
Componente inteligente finalizado
Sistema de Gestión de Agroecosistemas Terminado
Conclusiones investigación
Modelo de negocio finalizado
Modelo de requerimientos finalizado
Modelo de análisis terminado
Modelo de diseño finalizado
Modelos Ejecutables finalizados
Módulo inteligente integrado
Modelo de despliegue terminado
Sistema completamente probado
Realización del diagrama de clases de diseño
Realización de mapeo a modelo relacional
Confeccionar esquema con clases, relaciones, atributos y metodos identificados
Identificar metodos de clases y su visibilidad
Identificar relaciones entre clases
Identificar atributos de clases
Identificar clases
Identificar atributos compuestos y creas sus propias tablas
Identificar relaciones y cardinalidad entre tablas
Identificar pk y fk de cada tabla
Crear una columna en tabla, por atributo de clase correspondiente
Crear una tabla por clase identificada


Proyecto final
Administración de proyecto finalizada
Componente inteligente finalizado
Sistema de Gestión de Agroecosistemas Terminado
Conclusiones Investigación
Modelo de negocio finalizado
Modelo de requerimientos finalizado
Modelo de análisis terminado
Modelo de diseño finalizado
Modelos Ejecutables finalizados
Módulo inteligente integrado
Modelo de despliegue terminado
Sistema completamente probado
Especificación de convenciones de programación
Definición de plataformas de trabajo
Codificación
Confección de manuales de usuario
División de trabajo
Identificar nodos y configuraciones de red
Realizar esquema con nodos identificados
Especificación de casos de prueba
Realización de pruebas unitarias
Realización de pruebas de integración
Realización de pruebas de sistema
Confección de informe de pruebas
Corrección de incidencias


Proyecto final
Administración de proyecto finalizada
Componente inteligente finalizado
Sistema de Gestión de Agroecosistemas Terminado
Conclusiones investigación
Modelo de negocio finalizado
Modelo de requerimientos finalizado
Modelo de análisis terminado
Modelo de diseño finalizado
Modelos Ejecutables finalizados
Módulo inteligente integrado
Modelo de despliegue terminado
Sistema completamente probado
Especificación de convenciones de programación
Definición de plataformas de trabajo
Codificación
Confección de manuales de usuario
División de trabajo
Determinar convencion de nombres
Determinar significado de caracteres
Determinar tratamiento de identaciones y comentarios
Determinar visibilidad de los capitulos de los manuales segun tipo de usuario
Describir como hacer uso seguro del sistema
Detallar impresion de salidas
Detallar respaldo de datos y generacion de archivos
Detallar interpretacion de salidas del sistema
Detallar uso de consultas a futuro
Detallar lectura de informes
Detallar como y qué datos ingresar en cada campo


Proyecto final
Administración de proyecto finalizada
Componente inteligente finalizado
Sistema de Gestión de Agroecosistemas Terminado
Conclusiones investigación
Modelo de negocio finalizado
Modelo de requerimientos finalizado
Modelo de análisis terminado
Modelo de diseño finalizado
Modelos Ejecutables finalizados
Módulo inteligente integrado
Modelo de despliegue terminado
Sistema completamente probado
Especificación de casos de prueba
Realización de pruebas unitarias
Realización de pruebas de integración
Realización de pruebas de sistema
Confección de informe de pruebas
Corrección de incidencias
Identificar casos de prueba y su tipo
Identificar instancias del caso de prueba
Describir condiciones de ejecucion
Describir entradas y salidas esperadas
Evaluar prueba

4. Calendario

Las actualizaciones en la WBS, y en las prioridades actuales de las distintas tareas generaron como consecuencia grandes cambios en el calendario del proyecto. A continuación se expone la última calendarización realizada en Solum.

Nombre de tarea	Duración	Comienzo	Fin
Proyecto Final	**349 días**	**lun 28/03/11**	**jue 26/07/12**
Administración de proyecto finalizada	**22,5 días**	**lun 25/04/11**	**mié 25/05/11**
Proyecto planificado	**22,5 días**	**lun 25/04/11**	**mié 25/05/11**
Realizar WBS	15 días	lun 25/04/1	vie 13/05/11
Definir calendario y recursos de proyecto	15 días	mié 04/05/1	mié 25/05/1
Realizar informe planificación	5 días	mié 18/05/1	mié 25/05/1
Sistema de gestión de agroecosistema terminado	**349 días**	**lun 28/03/11**	**jue 26/07/12**
Modelo de negocio finalizado	**20 días**	**lun 28/03/11**	**vie 22/04/11**
Glosario de dominio	**20 días**	**lun 28/03/11**	**vie 22/04/11**
Identificar y describir terminos desconocidos y específicos del dominio	20 días	lun 28/03/1	vie 22/04/11
Consultar expertos del dominio e identificar terminos basicos	20 días	lun 28/03/1	vie 22/04/11
Informe preliminar finalizado	**19,5 días**	**lun 28/03/11**	**vie 22/04/11**
Definir propuesta de proyecto y su motivación	7 días	lun 28/03/1	mar 05/04/1
Identificar objetivos y situacion actual de la organizacion	11 días	mié 06/04/1	mié 20/04/1
Identificar mision, vision y areas de estudio	6 días	mié 06/04/1	mié 13/04/1
Definir procesos relacionados al proyecto y detallarlos	5,5 días	jue 14/04/1	jue 21/04/1
Identificar stakeholders y clientes	6 días	mié 06/04/1	mié 13/04/1
Validar y corregir informe preliminar	1 día	jue 21/04/1	vie 22/04/11
Modelo de requerimientos terminado	**44,5 días**	**jue 14/04/11**	**mié 15/06/11**
Realización de casos de uso de sistema finalizada	**14 días**	**mié 04/05/11**	**mar 24/05/11**
Realizar diagramas de actividad de los casos de uso	14 días	mié 04/05/1	mar 24/05/1
Describir pre y post condiciones de los casos de uso	13 días	mié 04/05/1	lun 23/05/1
Describir flujos basico y alternativo de eventos de cada caso de uso	14 días	mié 04/05/1	mar 24/05/1
Descripción de casos de uso de sistema finalizada	**19 días**	**vie 22/04/11**	**jue 19/05/11**
Identificar casos de uso esenciales	11 días	mié 04/05/1	jue 19/05/1
Listar casos de uso y actores detectados	15 días	vie 22/04/11	vie 13/05/11
Confección de prototipos terminada	**23,5 días**	**jue 14/04/11**	**mar 17/05/11**
Identificar requerimientos conocidos	15 días	jue 14/04/1	mié 04/05/1
Desarrollar modelo de prototipo	10 días	mar 03/05/1	mar 17/05/1
Realizar revision de prototipo	8 días	jue 05/05/1	lun 16/05/1
ERS terminada	**33 días**	**vie 22/04/11**	**mié 08/06/11**
Identificar y detallar requerimientos no funcionales	25 días	vie 22/04/11	vie 27/05/11
Describir restricciones	6 días	vie 22/04/11	lun 02/05/1
Describir caracteristicas de los usuarios	6 días	vie 22/04/11	lun 02/05/1
Detallar funciones del sistema	15 días	vie 22/04/11	vie 13/05/11
Incorporar descripciones y realizaciones de casos de uso	20 días	vie 22/04/11	vie 20/05/11
Validar y corregir ERS con stakeholders	25 días	mié 04/05/1	mié 08/06/1
Verificar y validar modelo	5 días	mié 08/06/1	mié 15/06/1
Modelo de análisis terminado	**77,5 días**	**mié 15/06/11**	**vie 30/09/11**
Diagramas estructurales terminados	**66,2 días**	**mié 15/06/11**	**jue 15/09/11**
Realización de diagramas de clases	**66,2 días**	**mié 15/06/11**	**jue 15/09/11**
Identificar clases de diseño	31 días	mié 15/06/1	jue 28/07/1
Identificar atributos de clases	26 días	jue 23/06/1	vie 29/07/11
Identificar relaciones entre clases	31 días	jue 23/06/1	vie 05/08/11
Identificar metodos de clases y su visibilidad	30 días	jue 23/06/1	jue 04/08/1
Confeccionar esquema con clases, relaciones, atributos y metodos ident	45 días	jue 23/06/1	jue 25/08/1
Verificar, validar y corregir diagramas	15 días	jue 25/08/1	jue 15/09/1
Identificación de paquetes	**10 días**	**jue 25/08/11**	**jue 08/09/11**
Identificar paquetes	10 días	jue 25/08/1	jue 08/09/1
Realizar diagrama de paquetes	10 días	jue 25/08/1	jue 08/09/1
Diagramas comportamentales terminados	**23 días**	**lun 22/08/11**	**mié 21/09/11**
Realización de diagramas de secuencia	**23 días**	**lun 22/08/11**	**mié 21/09/11**
Examinar descripciones de caso de uso esenciales	15 días	lun 22/08/1	vie 09/09/11
Implementar escenario con clases y objetos intervinientes	10 días	jue 08/09/1	mié 21/09/1
Realización de diagramas de comunicación	**23 días**	**lun 22/08/11**	**mié 21/09/11**
Examinar descripciones de caso de uso esenciales	15 días	lun 22/08/1	vie 09/09/11
Implementar escenario con clases y objetos intervinientes	10 días	jue 08/09/1	mié 21/09/1
Realización de diagrama de transición de estados	**10 días**	**mié 07/09/11**	**mar 20/09/11**
Identificar estados posibles	7 días	mié 07/09/1	jue 15/09/1
Identificar transiciones entre estados	7 días	mié 07/09/1	jue 15/09/1
Confeccionar digrafo con estados y transiciones	10 días	mié 07/09/1	mar 20/09/1
Examinar casos de uso esenciales	5 días	mié 07/09/1	mar 13/09/1
Verificar y validar modelo	7 días	jue 22/09/1	vie 30/09/11

Nombre de tarea	Duración	Comienzo	Fin
Modelo de diseño finalizado	**38 días**	**mié 31/08/11**	**vie 21/10/11**
Realización de diagrama de clases de diseño	**20 días**	**mié 31/08/11**	**mar 27/09/11**
Confeccionar esquema con clases, relaciones, atributos y metodos identificac	20 días	mié 31/08/1	mar 27/09/1
Identificar metodos de clases y su visibilidad	20 días	mié 31/08/1	mar 27/09/1
Identificar relaciones entre clases	20 días	mié 31/08/1	mar 27/09/1
Identificar atributos de clases	20 días	mié 31/08/1	mar 27/09/1
Identificar clases de diseño	20 días	mié 31/08/1	mar 27/09/1
Realización de mapeo a modelo relacional	**20 días**	**mié 21/09/11**	**mar 18/10/11**
Identificar atributos compuestos y creas sus propias tablas	20 días	mié 21/09/1	mar 18/10/1
Identificar relaciones y cardinalidad entre tablas	20 días	mié 21/09/1	mar 18/10/1
Identificar pk y fk de cada tabla	20 días	mié 21/09/1	mar 18/10/1
Crear una columna en tabla, por atributo de clase correspondiente	20 días	mié 21/09/1	mar 18/10/1
Crear una tabla por clase identificada	20 días	mié 21/09/1	mar 18/10/1
Verificar y validar modelo	3 días	mié 19/10/1	vie 21/10/11
Modelos ejecutables finalizados	**125 días**	**lun 24/10/11**	**vie 13/04/12**
Especificación de convenciones de programación	**2 días**	**lun 24/10/11**	**mar 25/10/11**
Determinar convencion de nombres	2 días	lun 24/10/1	mar 25/10/1
Determinar significado de caracteres	2 días	lun 24/10/1	mar 25/10/1
Determinar tratamiento de identaciones y comentarios	1 día	lun 24/10/1	lun 24/10/1
Definición de plataformas de trabajo	6 días	mié 26/10/1	mié 02/11/1
División de trabajo	3 días	mié 26/10/1	vie 28/10/11
Codificación	120 días	lun 31/10/1	vie 13/04/12
Verificar y validar modelo	5 días	lun 31/10/1	vie 04/11/11
Confección de manuales de usuario	**40 días**	**lun 31/10/11**	**vie 23/12/11**
Determinar visibilidad de los capitulos de los manuales segun tipo de usuario	8 días	lun 31/10/1	mié 09/11/1
Describir como hacer uso seguro del sistema	15 días	lun 31/10/1	vie 18/11/11
Detallar impresion de salidas	25 días	lun 31/10/1	vie 02/12/11
Detallar respaldo de datos y generacion de archivos	20 días	lun 31/10/1	vie 25/11/11
Detallar interpretacion de salidas del sistema	30 días	lun 31/10/1	vie 09/12/11
Detallar uso de consultas a futuro	35 días	lun 31/10/1	vie 16/12/11
Detallar lectura de informes	35 días	lun 31/10/1	vie 16/12/11
Detallar cómo y qué datos ingresar en cada campo	40 días	lun 31/10/1	vie 23/12/11
Verificar y validar manuales	7 días	lun 14/11/1	mar 22/11/1
Módulo inteligente integrado	30 días	mar 06/03/1	lun 16/04/12
Modelo de despliegue terminado	**22 días**	**lun 24/10/11**	**mar 22/11/11**
Identificacion de nodos y configuraciones de red	15 días	lun 24/10/1	vie 11/11/11
Realizar esquema con nodos identificados	15 días	lun 24/10/1	vie 11/11/11
Verificar y validar modelo	7 días	lun 14/11/1	mar 22/11/1
Sistema completamente probado	**199 días**	**lun 24/10/11**	**jue 26/07/12**
Especificación de casos de prueba	**22 días**	**lun 24/10/11**	**mar 22/11/11**
Identificar casos de prueba y su tipo	11 días	lun 24/10/1	lun 07/11/1
Identificar instancias del caso de prueba	11 días	lun 24/10/1	lun 07/11/1
Describir condiciones de ejecución	7 días	mar 08/11/1	mié 16/11/1
Describir entradas y salidas esperadas	11 días	mar 08/11/1	mar 22/11/1
Evaluar prueba	10 días	mar 08/11/1	lun 21/11/1
Realización de pruebas unitarias	40 días	lun 16/04/12	vie 08/06/12
Realización de pruebas de integración	40 días	vie 18/05/12	jue 12/07/12
Realización de pruebas del sistema	25 días	jue 21/06/12	mié 25/07/1
Confección de informe de pruebas	**16 días**	**jue 05/07/12**	**jue 26/07/12**
Plasmar especificación de casos de prueba	15 días	jue 05/07/12	mié 25/07/1
Describir casos de prueba ya realizados	11 días	mié 11/07/1	mié 25/07/1
Desarrollar conclusiones de cada caso de prueba ya realizado	11 días	mié 11/07/1	mié 25/07/1
Desarrollar conclusion general del sistema, una vez probado	11 días	mié 11/07/1	mié 25/07/1
Validar informe con stakeholders	3 días	mar 24/07/1	jue 26/07/12
Corrección de incidencias	25 días	jue 21/06/12	mié 25/07/1
Componente inteligente finalizado	**80 días**	**mar 15/11/11**	**lun 05/03/12**
Transformar conocimiento de expertos en unidades de conocimiento	20 días	mar 15/11/1	lun 12/12/1
Diseñar abstacciones modeladas	20 días	mar 13/12/1	lun 09/01/12
Implementación del prototipo del agente	20 días	mar 10/01/1	lun 06/02/12
Ejecucion de simulaciones	20 días	mar 07/02/1	lun 05/03/12
Evaluacion de resultados y correcciones	20 días	mar 07/02/1	lun 05/03/12
Conclusiones investigación	**85 días**	**mar 19/07/11**	**lun 14/11/11**
Personal con conocimiento de opciones de modelado e implementación	**30 días**	**mar 19/07/11**	**lun 29/08/11**
Reunión con experto	15 días	mar 19/07/1	lun 08/08/1
Lectura de bibliografía	15 días	lun 08/08/1	vie 26/08/11
Confección de informe con opciones	2 días	vie 26/08/11	lun 29/08/1
Personal capacitado para implementar componente inteligente	**55 días**	**mar 30/08/11**	**lun 14/11/11**
Selección de opción a implementar	5 días	mar 30/08/1	lun 05/09/1
Capacitaciones	40 días	mar 06/09/1	lun 31/10/1
Lectura de bibliografía	50 días	mar 06/09/1	lun 14/11/1

Capítulo III – Especificación de Requerimientos de Software

Parte I – Introducción

1.1. Preámbulo

El presente documento tiene como objetivo definir todos los requerimientos del Sistema de Gestión del Suelo de Agroecosistemas con Predicción a Futuro, “Solum”. Se incluyen descripciones generales de todo el Sistema en forma de listado de Casos de Uso (CU) y sus respectivas especificaciones. La presente aplicación tiene como destinatarios principales a la Institución Nacional de Tecnología Agropecuaria (INTA) y a la cátedra Proyecto Final correspondiente al quinto nivel de la carrera Ingeniería de Sistemas de Información, dictada en la Facultad Regional de Villa María como parte de la Universidad Tecnológica Nacional.

1.2. Audiencia

Los siguientes son los involucrados y destinatarios del presente documento:

Responsables de Confección: Barsce, Juan Cruz; Cantoni, María Luz; Favro, Ignacio; Girardi, Gonzalo

Responsables de Revisión: Ing. Zohil, Julio; Ing. Villafañe, Christian

Destinatarios: Ing. Zohil, Julio; Ing. Villafañe, Christian; Directivos y responsables del INTA Marcos Juárez e INTA Villa María.

1.3. Alcance

A partir del presente documento, su audiencia deberá ser capaz de comprender cuáles son las funciones y los servicios que brindará el Sistema de información a realizar.

Para dicho fin, se presentan los requerimientos funcionales y no funcionales acordados con los usuarios. Además, para comprender la interacción de los actores del Sistema, se presentan diagramas de casos de uso en donde puede observarse a simple vista, que casos de uso se relacionan con cada actor.

En caso de añadir o modificar funcionalidad en el software, será necesario modificar este documento con el objeto de mantener el mismo consistente con los cambios realizados.

1.4. Definiciones, Acrónimos y Abreviaturas

Los siguientes conceptos son incluidos con el objeto de definir los términos utilizados por el Sistema:

Agente: Entidad capaz de percibir el estado de su entorno, procesar dichas percepciones y actuar, según su conocimiento, de manera racional con el objetivo de maximizar un resultado esperado en pos de alcanzar una determinada meta.

Agroecosistema: Subconjunto de ecosistema con foco principal en la actividad humana de la agricultura y sus impactos en el suelo, flujos de energía y balance neto de nutrientes. Todo agroecosistema presenta componentes bióticos como los animales, plantas, microorganismos e insectos, y abióticos tales como la radiación solar, el clima y el suelo. Los labores que desarrolla la gestión agraria son los apropiados para lograr una producción económicamente rentable dañando lo menor posible el medio ambiente.

Análisis de Suelo: Reporte que especifica el estado actual del suelo de un determinado campo. El análisis agroquímico del suelo se hace sobre una muestra homogénea de suelo que represente un continuo de suelo de un terreno. La muestra tomada desde el suelo pesa aproximadamente entre uno y dos kilogramos, y es almacenada en bolsas especiales para ser inmediatamente enviadas al laboratorio.

Aprendizaje por Refuerzos: Área del machine learning centrada en cómo los agentes se comportan para tomar acciones en un ambiente en pos de maximizar el reward acumulado, implementando una política de decisión que equilibre entre la utilización de aquellas acciones consideradas mejores (explotación) con aquellas acciones a priori no óptimas (según el conocimiento actual del agente) pero que sirven para encontrar mejores acciones para ser usadas en el futuro (exploración).

Base de Conocimiento: Tipo especial de base de datos orientada a representar persistentemente el conocimiento, de forma que el mismo sea legible por una computadora y le permita realizar razonamiento deductivo. En el mismo guardan las reglas aprendidas por el agente en forma de predicados de lógica de primer orden y, en el caso del agente Solum, en forma de triplas (estado, acción, recompensa).

Caso de Uso: Descripción de una secuencia de interacciones entre un usuario y un sistema de información que buscarán aportarle al primero una salida apropiada. Son especialmente útiles para describir la funcionalidad del sistema de información.

Complejidad Computacional: Rama de la teoría de la computación enfocada en clasificar los problemas que se le asignan a un algoritmo de acuerdo a su dificultad inherente (en términos de recursos computacionales requeridos para resolverlo). Aquellos problemas que presentan alta complejidad resultan difíciles de ser resueltos por computadoras en tiempos aceptables, por lo que puede ser deseable resignar la posibilidad de obtener solución óptima en pos de una satisfactoria a cambio de acelerar los tiempos de resolución.

CU: Abreviatura para “Caso de Uso”.

Ecosistema: Sistema natural que está formado por un conjunto de organismos vivos (biocenosis) y el medio físico donde se relacionan (biotopo). Representa una unidad compuesta de organismos interdependientes que comparten el mismo hábitat. Los ecosistemas suelen formar una serie de cadenas que muestran la interdependencia de los organismos dentro del sistema.

ERS: Abreviatura para “Especificación de Requisitos de Software”.

Especificación de Requisitos de Software: Documento que contiene una especificación completa del comportamiento del sistema que se va a desarrollar. Incluye un conjunto de CU, actores y relaciones que describen todas las interacciones que los usuarios tendrán con el software. Se incluyen también aquellos requisitos no funcionales recogidos durante la captura de requerimientos.

Entrenamiento: Instancia del sistema en donde se buscarán probar alternativas a priori sub-óptimas en combinación con la experiencia obtenida en pos de generar nuevo conocimiento del sistema. Normalmente esto se lleva a cabo cuando el conocimiento actual del sistema no se considera lo suficientemente grande como para confiar en la precisión de las salidas que pueda arrojar, hecho por el cual no serán tomadas en cuenta para la toma de decisiones. Esta instancia es activada o desactivada manual o automáticamente (según cómo está configurado) y no necesariamente se limita a la fase inicial de funcionamiento del sistema.

Ejemplo: una vez que está entrenado, para que la variable A nunca tome valores fuera del rango [0,1] y empiece a recibir datos de algún otro lugar donde A vale entre 5 y 7, es recomendable se active la fase de entrenamiento para dichos valores, de

modo que el sistema pueda aprender antes de tomar decisiones en base a los mismos.

Error Global: Diferencia total entre el valor predicho por la red neuronal en el entrenamiento y el valor esperado que se esperaba que la misma obtenga. Normalmente se busca que tal diferencia no exceda valores infinitesimales como 10-6.

Estancamiento: Hace referencia a aquella situación en el aprendizaje en la cual el agente que está intentando aprender encontró en su actual mejor solución un máximo local y no puede salir del mismo.

Experto: Responsable de controlar cómo se lleva a cabo el entrenamiento y desarrollo del agente, verificando la calidad de sus deducciones y decisiones, y corrigiendo y mejorando las políticas.

Fertilizante: Sustancia natural, sintética o artificial, de composición orgánica o inorgánica que permite añadirle al suelo nutrientes esenciales para su buen desempeño.

Machine Learning: Rama de la inteligencia artificial centrada específicamente en el desarrollo de técnicas que permitan a las computadoras realizar aprendizajes para resolver problemas (normalmente de alta complejidad computacional). Incluye una amplia gama de enfoques, que van desde el aprendizaje por inducción lógica (en el cual se genera nueva información como inferencia de otra información preexistente), hasta algoritmos genéticos (algoritmos que imitan el proceso de selección natural para generar una salida) o aprendizaje por refuerzos.

Nutriente: son sustancias importantes, con propiedades químicas que definen los distintos tipos de suelo; además son el sustento de los distintos cultivos. Existen dos tipos: macronutrientes (N, P, Ca, Mg, K, S) y micronutrientes (Fe, Mn, Co, 2n; B, MO, Cl).

Objetivo de Sustentabilidad: Meta del agente que consiste en alcanzar un cierto valor en los nutrientes del suelo. La simulación finaliza al poder alcanzar dichos valores, pudiendo opcionalmente incluir un intervalo máximo de tiempo.

Objetivo de Rendimiento: Meta del agente que busca maximizar el rendimiento en las cosechas en un período de tiempo determinado, teniendo como restricciones las escalas mínimas en los valores de los nutrientes. Dicha restricción pueda ser

blanda, donde el Sistema le aplica una penalización a aquellas simulaciones en donde los valores de los nutrientes lleguen a ser menores a ciertos valores mínimos predefinidos; o bien puede usarse una restricción dura, en donde aquellas simulaciones que en donde los valores de los nutrientes llegan a ser menores a los valores mínimos predefinidos son anuladas.

Penalización al estado actual: Acción que implica aplicarle al agente un descuento en la recompensa que obtendría normalmente las próximas veces que el mismo arribe a dicho estado por incumplir alguna condición crítica. Una penalización podría aplicarse, por ejemplo, cuando el agente obtenga un valor más bajo que el permitido para un determinado nutriente.

Plan de Siembra: Conjunto de acciones de siembra - cosecha a realizarse en un determinado intervalo de tiempo que buscarán alcanzar un objetivo en pos de mejorar o preservar la sustentabilidad de los suelos.

Política de Decisión: Una política de decisión es un conjunto de reglas que especifica qué hará el agente ante una determinada situación.

Política de Siembra: Secuencia de acciones de siembra que llevan al estado inicial del campo a aquel que cumpla el objetivo de la simulación según los criterios predefinidos.

Productor Agropecuario: Es quien se encarga de trabajar el campo, buscando obtener el mejor rendimiento posible y al mismo tiempo preservar y mejorar el balance de los suelos, cargando sus datos y generando y evaluando los distintos planes de siembra para tomar medidas pertinentes.

Recompensa: Número real que representa el estímulo que el agente recibe tras llegar a un determinado estado. Puede tomar valores positivos o negativos (en los cuales el estímulo suele referirse como castigo). Cada vez que el agente recibe una recompensa la suma a las que ya tenía, por lo cual buscará llegar a los estados que mayor recompensa le devuelvan.

Red Neuronal Artificial: Algoritmos de aprendizaje estadístico supervisado inspirados en las redes neuronales biológicas y utilizadas para estimar o aproximar funciones que dependen de una considerable cantidad de entradas y son muy complejas para estimarlas matemáticamente o bien son desconocidas.

RL: Reinforcement Learning, término en inglés para Aprendizaje por Refuerzo.

Reward: Ver Recompensa.

Suelo: Es la capa más superficial de la corteza terrestre, que resulta de la descomposición de las rocas por los cambios bruscos de temperatura y por la acción del agua, del viento y de los seres vivos. Los componentes del suelo se pueden clasificar en inorgánicos, como la arena, la arcilla, el agua y el aire; y orgánicos, como los restos de plantas y animales. Entre las propiedades de los suelos se encuentran: color, distribución del tamaño de las partículas, consistencia, textura, estructura, porosidad, atmósfera, humedad, densidad, pH, materia orgánica, capacidad de intercambio iónico, sales solubles y óxidos amorfos-sílice alúmina y óxidos de fierro libres.

Software: Suma total de los programas de cómputo, procedimientos, reglas, documentación y datos asociados que forman parte de las operaciones de un sistema de computación. Hay diversos tipos de software: de sistema, operativo y de programación.

Temporada: Período de tiempo que abarca desde la siembra de un cultivo hasta su cosecha. En Argentina las temporadas se dividen en primavera – verano y otoño – invierno.

Zona: Término general para referirse a cualquier ubicación geográfica desde Localidades hasta Países.

1.5. Referencias

Para realizar este documento se tomó como modelo a la estructura utilizada en la cátedra Diseño de Sistemas de la carrera Ingeniería en Sistemas de Información.

Esta elección fue realizada por el grupo de trabajo del proyecto, se fundamenta en la simplicidad de comprensión y la objetividad que presenta este modelo de ERS.

Parte II – Presentación del Producto

2.1. Objetivo

El Proyecto de Software Solum tiene como objetivos principales: brindar soporte a las decisiones de los productores rurales del país, ingenieros agrónomos e INTA y concientizar sobre el uso sustentable del suelo expresándolo como aumentos en el rendimiento a largo plazo y dando a su vez acciones de siembra recomendadas para mejorar la situación actual y futura.

2.2. Alcance

El alcance del producto puede definirse en base a sus requisitos y sus respectivas características mostradas en el siguiente cuadro.

Requisitos	Características
Simular el estado a futuro del suelo, basando el mismo en datos históricos, ofreciendo una mejor visión a largo plazo de los recursos.	Basando en variables previamente obtenidas y actividades descriptas por los interesados, el sistema realiza cálculos y estimaciones para brindar un estado simulado del suelo.
Generar propuestas a los productores para disminuir los efectos negativos de la actividad agropecuaria.	Por medio del aprendizaje continuo mediante técnicas de inteligencia artificial y la recopilación de datos, el sistema genera propuestas actualizadas sobre una eficiente administración del suelo.
Mantener una base de datos actualizada de la situación de los recursos terrestres de una población, dejando disponible dichos datos a quien lo requiera mediante una interfaz web.	Todos los datos ingresados para la obtención de las simulaciones y propuestas son incorporados y analizados para una mejora continua y brindar conocimiento a la población.
Permitirles a los productores calcular y obtener soluciones posibles respecto	Basándose en algunas variables previamente definidas y otras

a: saldo de nutrientes según un rendimiento y fertilización determinados, déficit hídrico, rendimiento medio posible y simulado, salinidad de los suelos, factores limitantes, etc.	incorporadas por el usuario, el sistema realiza cálculos y estimaciones para brindar los resultados solicitados, y recomendar acciones para evitar o disminuir las posibilidades de ocurrencia de la situación problemática.
Simplificar la comunicación de información y su disponibilidad.	Manteniendo de forma centralizada y simplificando los informes, se logra una comunicación más clara y efectiva de la información.

2.3. Supuestos

El proyecto supone por parte de los usuarios un manejo mínimo de informática como son el uso de Microsoft Windows y un navegador web.

Por parte de los productores agropecuarios y personal del INTA, el proyecto supone que los mismos deben proveer al sistema de los datos requeridos e información fehaciente para la mejora en el entrenamiento y obtención de datos por parte del sistema. Supone, además, la capacitación mínima de los integrantes del equipo en el área agroeconómica y agroquímica, debido a la estrecha relación del proyecto con las áreas mencionadas.

Por último, se asume disponibilidad en el acceso a internet por parte de los usuarios.

2.4. Riesgos

El proyecto tiene algunos factores de riesgo a considerar, ya que en gran parte se basa en agentes externos al mismo. Depende de una buena fuente de datos que permitan al agente inteligente aprender de información correcta, pues un agente entrenado con información apócrifa arrojará inevitablemente resultados incorrectos. También cabe aclarar que la simulación tiene plena dependencia de los datos históricos.

Además el software está sujeto al continuo aporte de datos que deben ser cuidadosamente analizados, para evitar el ingreso de información incorrecta o malintencionada.

Parte III – Descripción General de Sistema

3.1. Paquetes y su funcionalidad

El Sistema se compone de los presentes paquetes, formados a su vez por sus respectivos CU:

Paquete 01: Aprendizaje por Red Neuronal

Número de CU	Nombre
1	Entrenar Red Neuronal
2	Testear Red Neuronal de Predicción de Rendimiento
3	Configurar Parámetros Entrenamiento
4	Realizar Entrenamiento
5	Ejecutar Red Neuronal

Paquete 02: Gestión Ambiental

Número de CU	Nombre
6	Administrar Análisis de Suelo
11	Administrar Campos
16	Administrar Colores de Suelo
21	Administrar Cosechas.
26	Administrar Cultivos
31	Administrar Departamentos
36	Administrar Estados de Nutriente
41	Administrar Fertilizaciones
46	Administrar Fertilizantes
51	Administrar Localidades
56	Administrar Nutrientes
61	Administrar Objetivos
66	Administrar Opciones de Cultivos
71	Administrar Países
76	Administrar Planes de Siembra
81	Administrar Precipitaciones
86	Administrar Provincias
91	Administrar Reporte de Radiación Solar

96	Administrar Reportes de Precipitaciones
101	Administrar Siembras
106	Administrar Tipos de Suelo
111	Administrar Texturas de Suelo
116	Generar Alerta Visual por Nivel Insuficiente de Nutriente en el Suelo
121	Generar Alerta de Salinización de Suelos

Nota: luego de cada caso de uso “administrar” se incluyen los casos de uso de alta, baja, modificación y búsqueda numerados a continuación. Ejemplo: luego de 6. Administrar Análisis de Suelo, sigue 7. Alta Análisis de Suelo, 8. Baja Análisis de Suelo, 9. Modificar Análisis de Suelo y 10. Buscar Análisis de Suelo. Esto se repite para cada uno de los CU de “Administración”.

Paquete 03: Gestión de Reportes

Número de CU	Nombre
122	Generar Informe de Evolución Zonal de Nutrientes
123	Generar Informe de Actividad de Usuarios
124	Generar Informe de Errores
125	Generar Informe de Opinión de Usuarios
126	Generar Informe de Rendimientos
127	Generar Informe de Saldo de Nutrientes
128	Generar Informe de Visitas Realizadas por Zona
129	Generar Listado de Análisis de Suelo Disponibles
130	Generar Listado de Usuarios
131	Generar Mapa de Concentración de Nutrientes por Zonas
132	Generar Mapa de Situación de Suelos
133	Generar Reporte de Entrenamiento de Red Neuronal
134	Generar Reporte de Precisión de Red Neuronal
135	Generar Reporte de Seguimiento de Evolución de Suelos
136	Generar Reporte de Sesiones de Usuario Iniciadas

Paquete 04: Planificación de Cultivos

Número de CU	Nombre
137	Entrenar Agente Simulación Escenarios
138	Configurar Parámetros de Entrenamiento
139	Simular Plan
140	Generar Plan de Siembra

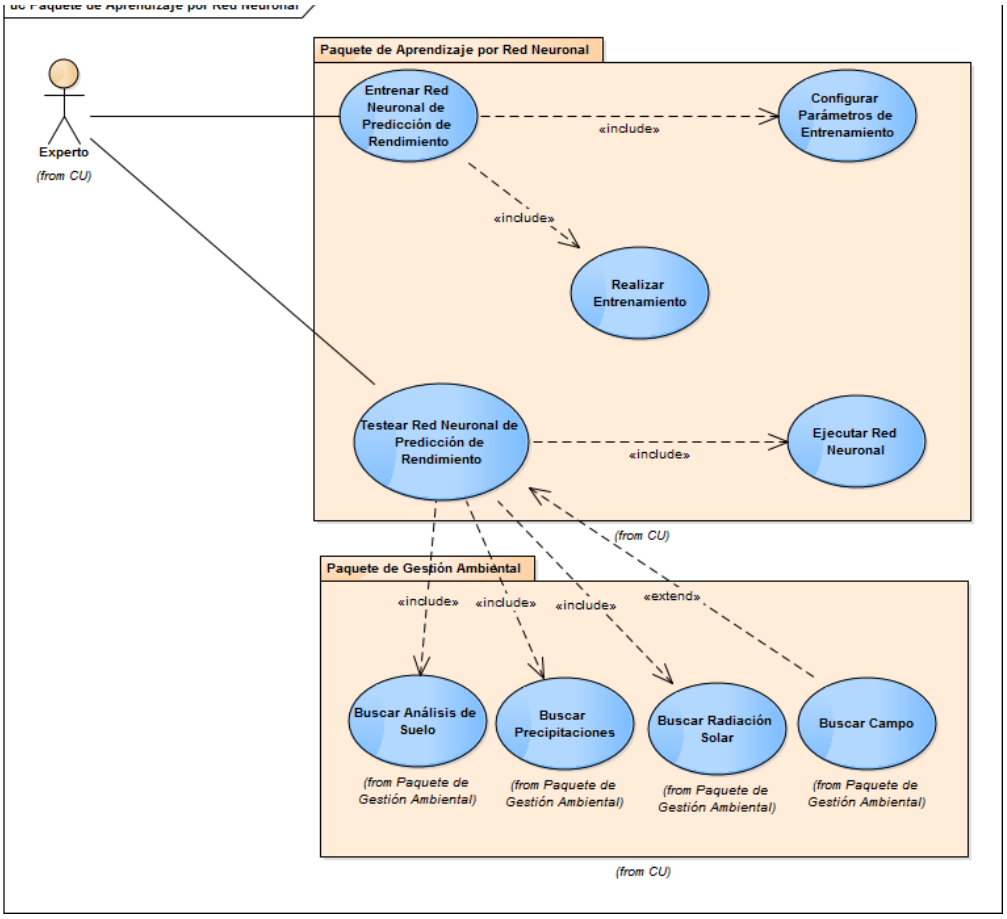
141	Seleccionar Política de Siembra
142	Buscar Acción de Siembra
143	Simular Transcurso de Tiempo
144	Iniciar Seguimiento de Plan de Siembra
145	Finalizar Seguimiento de Plan de Siembra
146	Determinar Fertilizante a Aplicar
147	Determinar Cultivo a Plantar

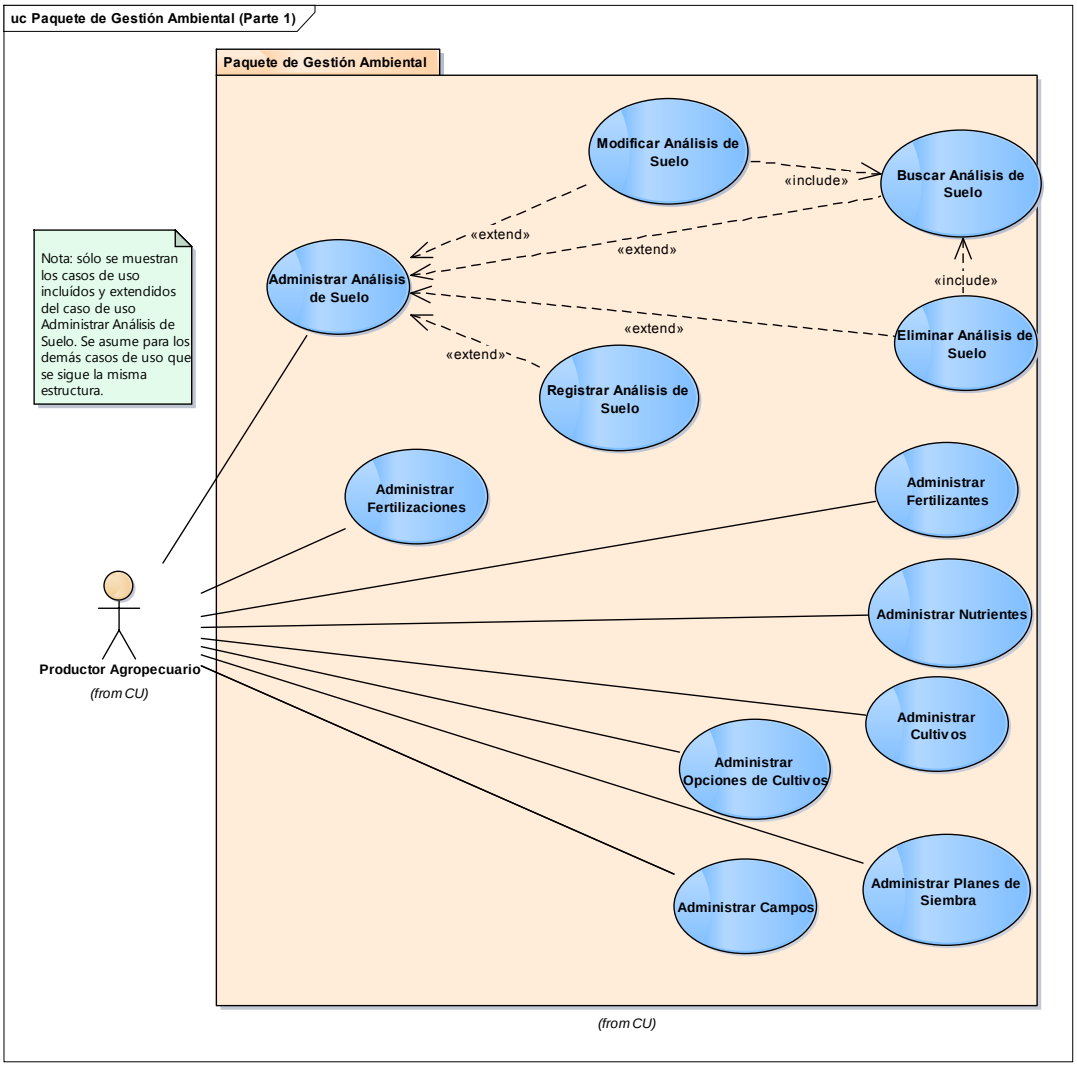
Paquete 05: Seguridad

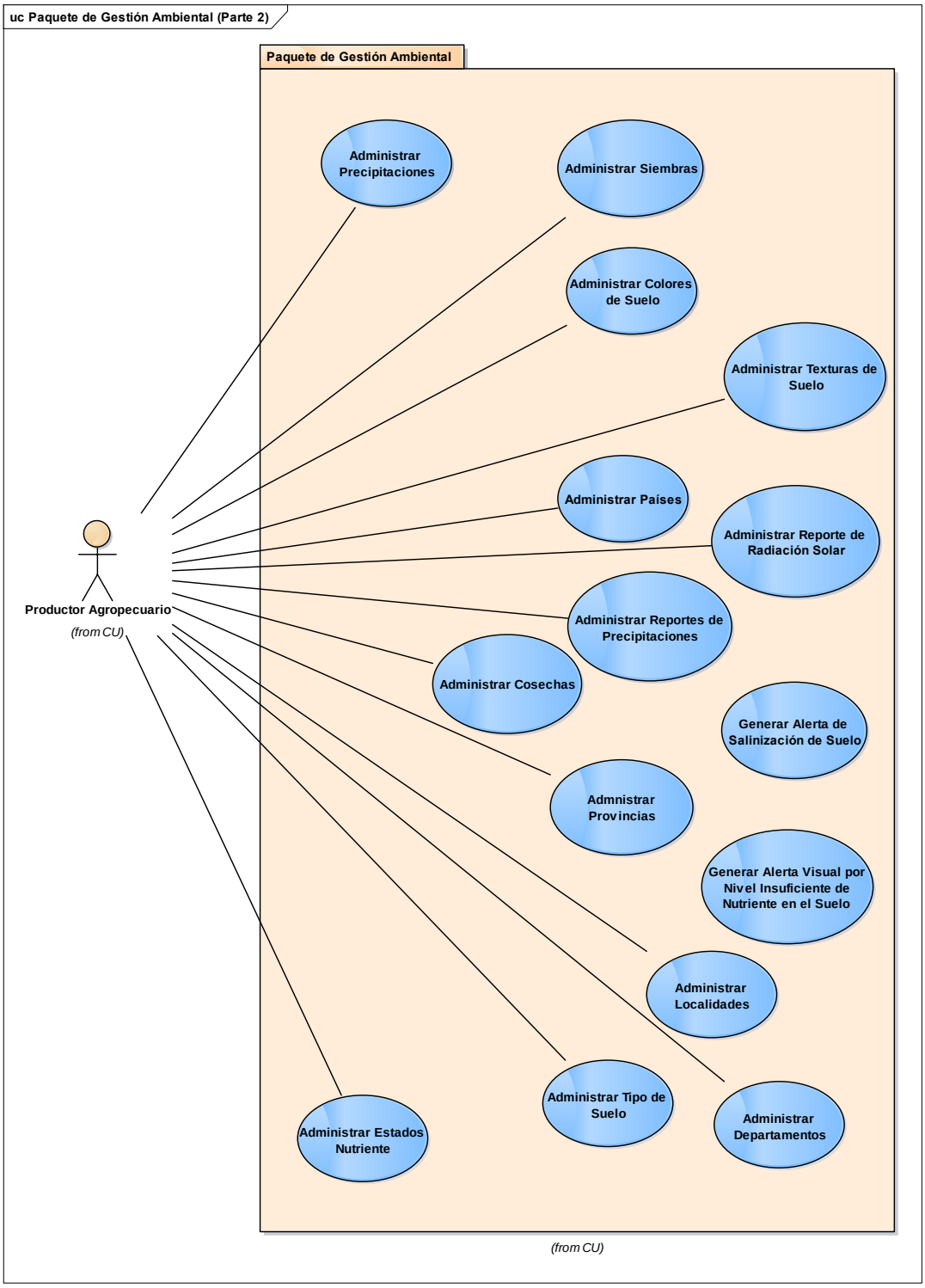
Número de CU	Nombre
148	Administrar Grupos de Usuario
153	Administrar Permisos
158	Administrar Roles de Usuario
163	Administrar Usuarios
165	Finalizar Sesión de Usuario
166	Iniciar Sesión de Usuario
167	Nueva Opinión de Usuario

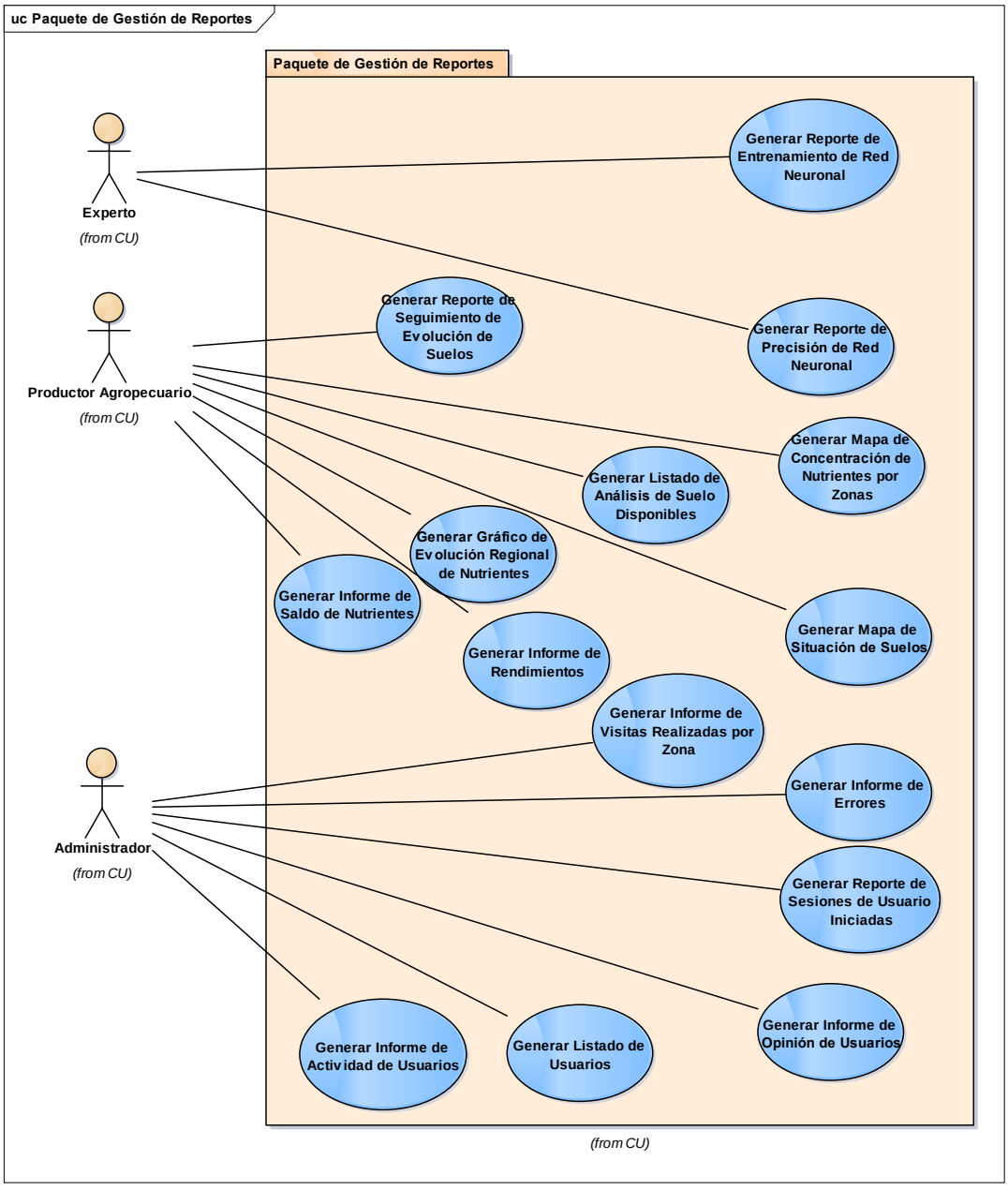
3.2. Actores

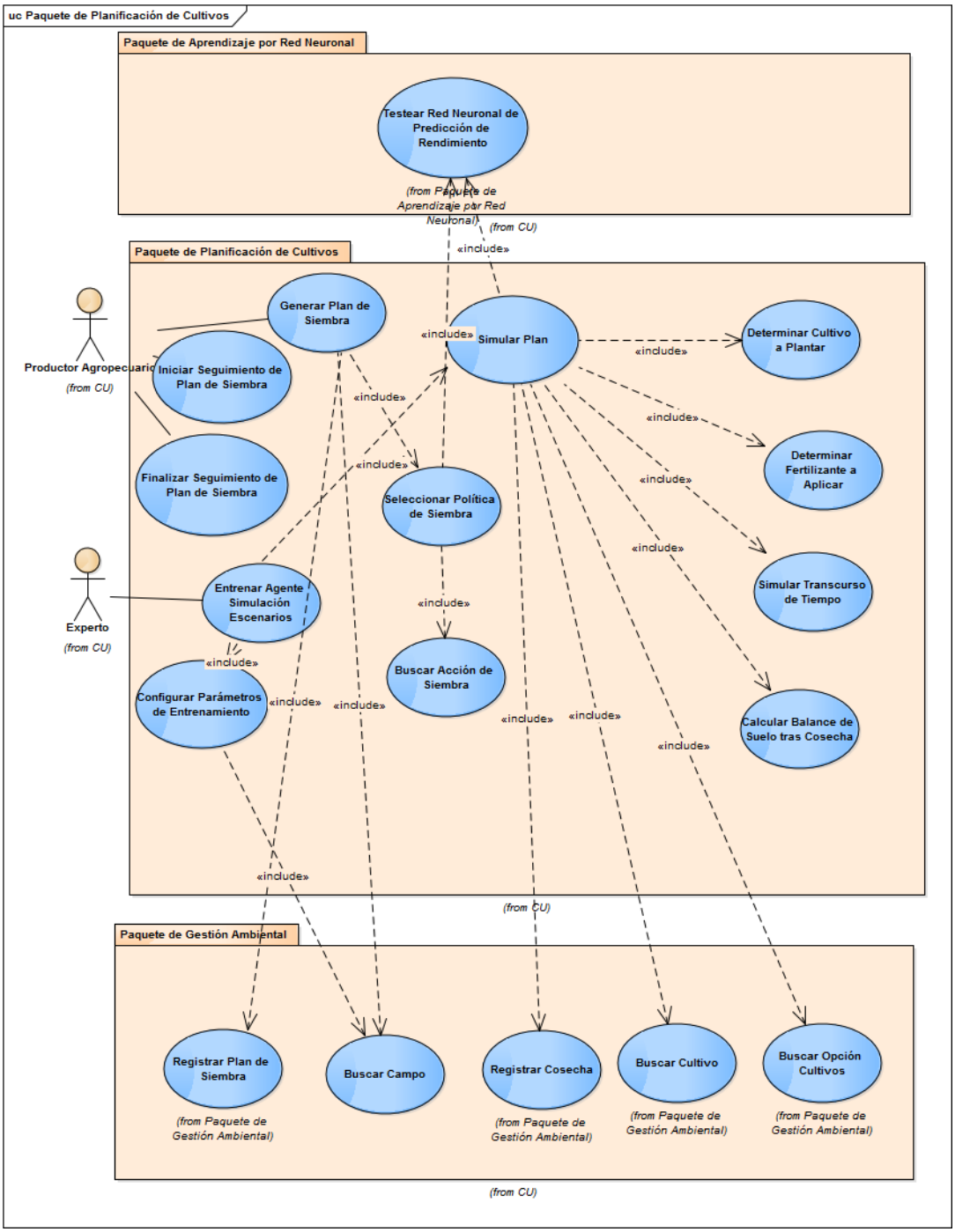
Nombre	Rol
Administrador	Persona que realiza la administración general del sistema, asegurando su correcto funcionamiento para sus usuarios y las actividades que ellos realizan.
Productor Agropecuario	Encargado de la administración y carga de datos del campo tales como las localidades, provincias y países; campos; componentes; fertilizantes; nutrientes; cultivos y tipos de cultivos; análisis de suelo, así como de la inicialización y finalización de los seguimientos del suelo y la generación de planes de siembra en base al conocimiento previamente aprendido.
Experto	Responsable de controlar cómo se lleva a cabo el entrenamiento y desarrollo del agente, verificando la calidad de sus deducciones y decisiones, y corrigiendo y mejorando las políticas.
Usuario del Sistema (nombre corto: Usuario)	Cualquier persona autorizada a acceder al sistema.

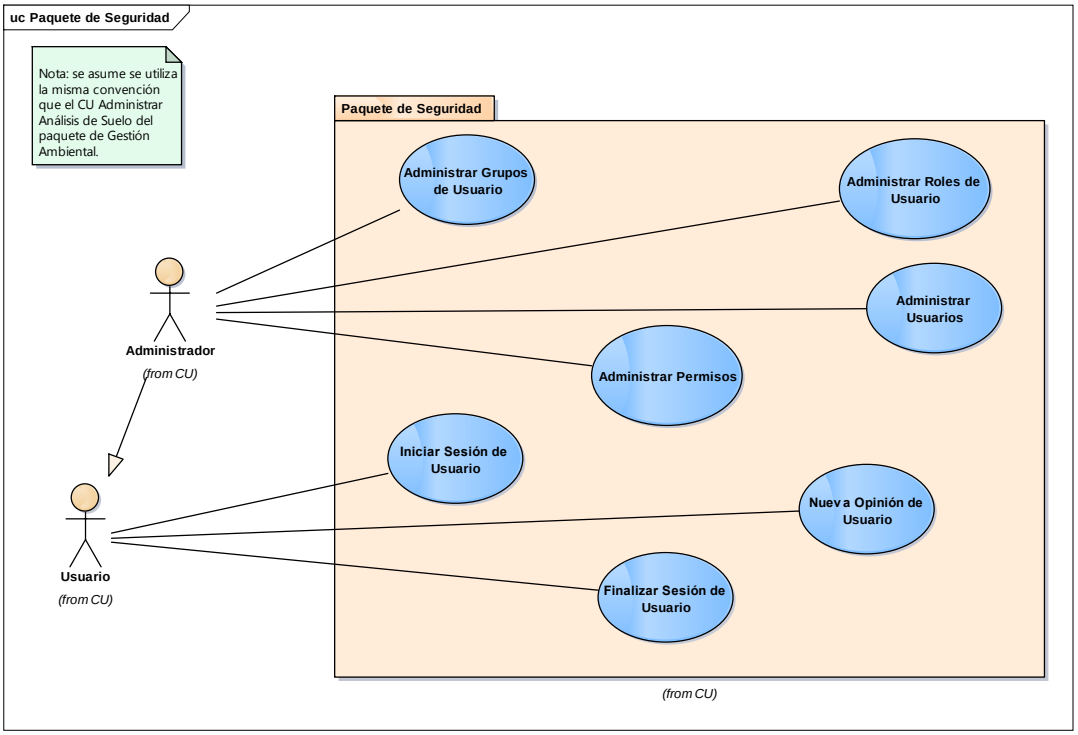

Paquete de Aprendizaje por Red Neuronal
Experto
(from CU)
Entrenar Red Neuronal de Predicción de Rendimiento
«include»
Configurar Parámetros de Entrenamiento
«include»
Realizar Entrenamiento
Testear Red Neuronal de Predicción de Rendimiento
«include»
Ejecutar Red Neuronal
(from CU)
Paquete de Gestión Ambiental
«include»
«include»
«include»
«extend»
Buscar Análisis de Suelo
Buscar Precipitaciones
Buscar Radiación Solar
Buscar Campo
(from Paquete de Gestión Ambiental)
(from Paquete de Gestión Ambiental)
(from Paquete de Gestión Ambiental)
(from Paquete de Gestión Ambiental)
(from CU)


uc Paquete de Gestión Ambiental (Parte 1)
Paquete de Gestión Ambiental
Modificar Análisis de Suelo
Buscar Análisis de Suelo
«include»
«extend»
«extend»
«include»
Administrar Análisis de Suelo
«extend»
Eliminar Análisis de Suelo
«extend»
Registrar Análisis de Suelo
Nota: sólo se muestran los casos de uso incluídos y extendidos del caso de uso Administrar Análisis de Suelo. Se asume para los demás casos de uso que se sigue la misma estructura.
Administrar Fertilizaciones
Administrar Fertilizantes
Administrar Nutrientes
Administrar Cultivos
Administrar Opciones de Cultivos
Administrar Planes de Siembra
Administrar Campos
Productor Agropecuario
(from CU)
(from CU)


uc Paquete de Gestión Ambiental (Parte 2)
Paquete de Gestión Ambiental
Administrar Precipitaciones
Administrar Siembras
Administrar Colores de Suelo
Administrar Texturas de Suelo
Administrar Países
Administrar Reporte de Radiación Solar
Administrar Reportes de Precipitaciones
Administrar Cosechas
Generar Alerta de Salinización de Suelo
Admnistrar Provincias
Generar Alerta Visual por Nivel Insuficiente de Nutriente en el Suelo
Administrar Localidades
Administrar Tipo de Suelo
Administrar Departamentos
Administrar Estados Nutriente
Productor Agropecuario
(from CU)
(from CU)


uc Paquete de Gestión de Reportes
Paquete de Gestión de Reportes
Experto
(from CU)
Productor Agropecuario
(from CU)
Administrador
(from CU)
Generar Reporte de Entrenamiento de Red Neuronal
Generar Reporte de Precisión de Red Neuronal
Generar Reporte de Seguimiento de Evolución de Suelos
Generar Mapa de Concentración de Nutrientes por Zonas
Generar Listado de Análisis de Suelo Disponibles
Generar Gráfico de Evolución Regional de Nutrientes
Generar Informe de Saldo de Nutrientes
Generar Mapa de Situación de Suelos
Generar Informe de Rendimientos
Generar Informe de Visitas Realizadas por Zona
Generar Informe de Errores
Generar Reporte de Sesiones de Usuario Iniciadas
Generar Informe de Opinión de Usuarios
Generar Informe de Actividad de Usuarios
Generar Listado de Usuarios
(from CU)


uc Paquete de Planificación de Cultivos
Paquete de Aprendizaje por Red Neuronal
Testear Red Neuronal de Predicción de Rendimiento
(from Paquete de Aprendizaje por Red Neuronal)
(from CU)
«include»
Paquete de Planificación de Cultivos
Productor Agropecuario
(from CU)
Generar Plan de Siembra
Iniciar Seguimiento de Plan de Siembra
Finalizar Seguimiento de Plan de Siembra
Simular Plan
Determinar Cultivo a Plantar
Determinar Fertilizante a Aplicar
Seleccionar Política de Siembra
Simular Transcurso de Tiempo
Calcular Balance de Suelo tras Cosecha
Experto
(from CU)
Entrenar Agente Simulación Escenarios
Configurar Parámetros de Entrenamiento
Buscar Acción de Siembra
(from CU)
Paquete de Gestión Ambiental
Registrar Plan de Siembra
(from Paquete de Gestión Ambiental)
Buscar Campo
Registrar Cosecha
(from Paquete de Gestión Ambiental)
Buscar Cultivo
(from Paquete de Gestión Ambiental)
Buscar Opción Cultivos
(from Paquete de Gestión Ambiental)
(from CU)



Nota: al igual que Administrador, todos los demás actores se presentan como una especialización de Usuario, asociándose con el mismo a través de una relación de herencia.

3.3. Perspectiva del Producto

La entidad seleccionada para llevar a cabo el proyecto planteó la necesidad de llevar un seguimiento del estado de suelo, es decir, en base a una serie de análisis de suelo y a una variedad de minerales selectos, presentar informes estadísticos e históricos del estado del suelo, aprovechando la masividad de las comunicaciones informáticas para recopilar la información requerida. El grupo de trabajo, propuso en base al área de investigación seleccionada, simular estados futuros del suelo, y en base a esto, proponer mediante un componente de inteligencia artificial (IA) diversas acciones que mejoren la situación simulada, previniendo o minimizando la degradación de los recursos terrestres.

Al existir una amplia cantidad de variables que influyen sobre el suelo, surge la necesidad de tener un control automatizado de las posibilidades que existen, y de esta manera simplificar la toma de decisiones, y es vital generar informes simples para motivar la participación de la comunidad, para generar una fuente de información confiable y amplia.

El proyecto brinda una herramienta de gran utilidad a la comunidad agrícola, base de la economía de nuestro país, y como tal, fuente principal de externalidades de la comunidad argentina en su totalidad.
Es importante destacar la plena factibilidad del proyecto, porque cuenta con factores como aprobación de los docentes de la cátedra, necesidad y plena cooperación por parte de los usuarios, y una extensa disposición de datos históricos como base del proyecto.

Parte IV – Descripción Detallada de Requerimientos

4.1. Aclaraciones

Se presenta a continuación la plantilla de Casos de Uso utilizada para especificar cada uno de los mismos.

<table>
<tr><td colspan="2">Nombre: Nombre del Caso de Uso</td><td>ID: Identificador del Caso de Uso</td></tr>
<tr><td colspan="2">Actor Principal: Nombre del Actor Principal involucrado en el Caso de Uso.</td><td>Actor Secundario: Nombre del actor secundario involucrado en el Caso de Uso.</td></tr>
<tr><td colspan="3">Prioridad: La prioridad que representa el Caso de Uso en el Sistema. Según su importancia se clasifican como Esenciales, Útiles o Deseables.

☐ Esencial ☐ Útil ☐ Deseable</td></tr>
<tr><td colspan="3">Complejidad: Indica qué tan complejo resulta llevar a cabo la ejecución del CU.

☐ Alta ☐ Media ☐ Baja</td></tr>
<tr><td colspan="3">Tipo de Caso de Uso: Indica si el CU es Concreto o Abstracto.

☐ Concreto ☐ Abstracto</td></tr>
<tr><td colspan="3">Objetivo: El objetivo que pretende alcanzar la ejecución del CU.</td></tr>
<tr><td colspan="3">Precondiciones: Hechos que deben cumplirse a priori para que el CU pueda ser ejecutado.</td></tr>
<tr><td rowspan="2">Post-Condiciones:</td><td colspan="2">Éxito: Hechos que se ocurren tras la correcta ejecución del CU.</td></tr>
<tr><td colspan="2">Fracaso: Hechos que ocurren tras la fallida ejecución del CU.</td></tr>
<tr><td colspan="2">Curso Normal</td><td>Alternativas</td></tr>
<tr><td colspan="2">Curso de acción adoptado por el Sistema en un caso típico.</td><td>Curso de acción adoptado por el Sistema ante situaciones menos frecuentes o inesperados.</td></tr>
<tr><td colspan="2">1.</td><td></td></tr>
<tr><td colspan="2">2.</td><td></td></tr>
<tr><td colspan="2">3.</td><td></td></tr>
<tr><td colspan="2">4.</td><td></td></tr>
<tr><td colspan="2">5.</td><td></td></tr>
<tr><td colspan="3">Requerimientos No funcionales Especiales: Requerimientos No Funcionales implicados en el presente CU.</td></tr>
<tr><td colspan="3">Observaciones: Consideraciones adicionales del CU.</td></tr>
</table>

4.2. Especificación de Casos de Uso

Paquete 01: Aprendizaje por Red Neuronal

Nombre: Entrenar Red Neuronal		ID: CU001
Actor Principal: Experto	Actor Secundario: No aplica	
Prioridad: ☒ Esencial ☐ Útil ☐ Deseable		
Complejidad: ☒ Alta ☐ Media ☐ Baja		
Tipo de Caso de Uso: ☒ Concreto ☐ Abstracto		
Objetivo: Realizar el entrenamiento de la red neuronal, buscando que la misma encuentre una función que mapee con el menor error posible los datos del suelo con los rendimientos esperados.		
Precondiciones: El usuario debe estar logueado en el Sistema con sus respectivos permisos.		
Post-Condiciones:	Éxito: Red neuronal entrenada.	
	Fracaso: La red neuronal no pudo converger a una solución que satisfaga el error global dentro del número máximo de iteraciones.	
Curso Normal		Alternativas
1. El CU comienza cuando el Experto selecciona “Entrenar Red Neuronal”.		
2. El Sistema muestra los parámetros básicos que requiere el entrenamiento: • Error global. • Número máximo de Iteraciones. • Conjunto de tuplas que definen el estado del campo (variables independientes). • Rendimiento obtenido para cada una de las tuplas (variables dependientes).		
3. El Experto selecciona “Configurar Parámetros Entrenamiento Red Neuronal”		
4. El Sistema invoca al CU Configurar Parámetros Entrenamiento Red Neuronal.		
5. El Sistema habilita la opción “Realizar Entrenamiento”.		
6. El Experto selecciona “Realizar Entrenamiento”.		

7. El Sistema invoca CU Realizar Entrenamiento.	7. A. La red neuronal no alcanzó la convergencia. 7. A.1. El Sistema muestra un mensaje informando la posibilidad de encontrar una función que mapee las variables independientes y las dependientes sin exceder el error mínimo. 7. A.2. El Sistema muestra la opción "Volver a Configurar Parámetros Entrenamiento Red Neuronal". 7. A.3. El Experto selecciona la opción "Volver a Configurar Parámetros Entrenamiento Red Neuronal". 7. A.4. El Sistema vuelve al paso 2. 7. A.3.A. El Experto selecciona Salir.
8. El Sistema muestra las siguientes estadísticas entre cada uno de los valores esperados y cada uno de los valores calculados de rendimiento: 8.1. Desviación estándar. 8.2. Desviación absoluta. 8.3. Desviación absoluta máxima.	
9. El Experto selecciona salir.	
Requerimientos No funcionales Especiales:	
Observaciones:	

Nombre: Testear Red Neuronal de Predicción de Rendimiento		ID: CU002
Actor Principal: Experto / Productor Agropecuario		Actor Secundario: No aplica
Prioridad: ☒ Esencial ☐ Útil ☐ Deseable		
Complejidad: ☐ Alta ☒ Media ☐ Baja		
Tipo de Caso de Uso: ☒ Concreto ☐ Abstracto		
Objetivo: Obtener el rendimiento de un cultivo plantado en un campo, en base a su estado.		
Precondiciones: El usuario debe estar logueado en el Sistema con sus respectivos permisos. Al menos debe existir un campo cargado en el Sistema.		
Post-Condiciones:	Éxito: La red neuronal ha sido exitosamente testeada.	
	Fracaso:	
Curso Normal		Alternativas
1. El CU inicia cuando es invocado desde el CU Simular Plan, el CU Seleccionar Política de Siembra, o bien cuando el Experto selecciona la opción "Testear Red Neuronal de Predicción de Rendimiento".		
2. El Sistema verifica que el estado del campo a calcular rendimiento fue pasado como parámetro al iniciar el CU.		2. A. El Sistema no cuenta con un estado del campo para calcular el rendimiento. 2. A.1. El Sistema invoca al CU Buscar Campo. 2. A.2. El Sistema solicita se seleccione un campo de la lista. 2. A.3. El Experto selecciona un campo de la lista. 2. A.4. El CU continúa en el paso 3.
3. El Sistema llama al CU Buscar Análisis de Suelo.		
4. El Sistema extrae del análisis de suelo más reciente los siguientes datos: 4.1. Cantidad de azufre presente en el suelo, mostrado en kilogramos por hectárea. 4.2. Cantidad de fósforo presente en el suelo, en Kg/Ha. 4.3. Cantidad de nitrógeno presente en el suelo, en Kg/Ha.		

4.4. pH del suelo.	
5. El Sistema llama al CU Buscar Precipitaciones.	
6. El Sistema sumariza la cantidad de precipitaciones de cada uno de los reportes de precipitaciones de la temporada actual, obteniendo la cantidad total de precipitaciones de la temporada.	
7. El Sistema llama al CU Buscar Radiación Solar.	
8. El Sistema sumariza la cantidad de radiación solar de cada uno de los reportes de radiación de la temporada actual, obteniendo la cantidad total de radiación solar que hubo en la temporada.	
9. El Sistema llama al CU Ejecutar Red Neuronal con la siguiente información pasada como parámetro: 9.1. Tipo de suelo. 9.2. Cantidad de fósforo en el suelo (en microgramos por gramo, ug / g). 9.3. Radiación solar	
10. El Sistema ejecuta la opción Salir, o bien el Experto selecciona la opción Salir.	
Requerimientos No funcionales Especiales:	
Observaciones:	

Paquete 02: Gestión Ambiental

<table>
<tr><td colspan="2">Nombre: Registrar Análisis de Suelo</td><td>ID: CU006</td></tr>
<tr><td colspan="2">Actor Principal: Productor Agropecuario</td><td>Actor Secundario: No aplica</td></tr>
<tr><td colspan="3">Prioridad: ☒ Esencial ☐ Útil ☐ Deseable</td></tr>
<tr><td colspan="3">Complejidad: ☐ Alta ☒ Media ☐ Baja</td></tr>
<tr><td colspan="3">Tipo de Caso de Uso: ☒ Concreto ☐ Abstracto</td></tr>
<tr><td colspan="3">Objetivo: Registrar en el Sistema un nuevo análisis de suelo.</td></tr>
<tr><td colspan="3">Precondiciones: El usuario debe tener una sesión iniciada en el Sistema con sus respectivos permisos.</td></tr>
<tr><td rowspan="2">Post-Condiciones:</td><td colspan="2">Éxito: El análisis se registró satisfactoriamente en el Sistema.</td></tr>
<tr><td colspan="2">Fracaso: El PA cancela el CU.</td></tr>
<tr><td colspan="2">Curso Normal</td><td>Alternativas</td></tr>
<tr><td colspan="2">1. El CU comienza cuando el Productor Agropecuario (PA) selecciona la opción “Registrar Análisis de Suelo”.</td><td></td></tr>
<tr><td colspan="2">2. El sistema solicita se ingrese:
a. Campo desde donde fue realizada la muestra (seleccionado a partir de una lista).
b. Ubicación geográfica en donde se realizó el análisis de suelo.
c. Fecha de realización del análisis.
d. Laboratorio en el cual se realizó el análisis.
e. Profundidad de la muestra, en metros (decimal mayor que cero).
f. Porcentaje de materia orgánica (decimal entre 0 y 100).
g. Conductividad del extracto (decimal entre 0 y 25).
h. Color (se elige uno entre los valores de una lista, como Amarillo, Rojo, Negro, entre otros).</td><td>2. A. El Sistema detecta que alguno de los valores no se encuentra en su rango correcto.
2. A.1 El Sistema solicita que se reingresen dichos datos.
2. A.2. El PA reingresa los datos.
2. A.2.A. El PA presiona “Cancelar”.
4. A.2.A.1 Se cancela el CU.</td></tr>
</table>

i. Nutrientes (se eligen todos los que se desea cargar a partir de los valores de una lista, debiendo como mínimo estar presentes el Nitrógeno (N), Fósforo (P) y Azufre (S)). Para cada nutriente se ingresa: su método de extracción (se selecciona uno de una lista), su forma iónica (se selecciona una de una lista) y su cantidad (decimal positivo). **j.** Porcentaje de humedad (decimal entre 0 y 100). **k.** Capacidad de intercambio catiónico (entero positivo). **l.** Textura (seleccionada de una lista). **m.** Tipo de suelo (seleccionado de una lista).	
3. El PA ingresa los datos requeridos.	
4. El Sistema muestra los valores para cada uno de los datos ingresados y solicita que se confirme el registro del análisis de suelo.	
5. El PA confirma el registro.	5. A. El PA presiona “Cancelar”. 5. A.1. Se cancela el CU.
6. El sistema registra un análisis de suelo en la fecha y coordenadas ingresadas, y con sus respectivos valores para cada uno de los datos cargados.	
7. El Sistema selecciona la opción “Salir”.	
Requerimientos No funcionales Especiales:	
Observaciones:	

<table>
<tr><td colspan="2">Nombre: Modificar Análisis de Suelo</td><td>ID: CU007</td></tr>
<tr><td colspan="2">Actor Principal: Productor Agropecuario</td><td>Actor Secundario: No aplica</td></tr>
<tr><td colspan="3">Prioridad: ☒ Esencial ☐ Útil ☐ Deseable</td></tr>
<tr><td colspan="3">Complejidad: ☐ Alta ☒ Media ☐ Baja</td></tr>
<tr><td colspan="3">Tipo de Caso de Uso: ☒ Concreto ☐ Abstracto</td></tr>
<tr><td colspan="3">Objetivo: Realizar la modificación de un análisis de suelo que fue buscado previamente.</td></tr>
<tr><td colspan="3">Precondiciones: El usuario debe estar logueado en el Sistema con sus respectivos permisos.</td></tr>
<tr><td rowspan="2">Post-Condiciones:</td><td colspan="2">Éxito: el análisis se modificó de forma satisfactoria</td></tr>
<tr><td colspan="2">Fracaso 1: El Sistema no encontró ningún análisis de suelo que cumpla los criterios de búsqueda establecidos por el PA,</td></tr>
<tr><td colspan="2">Curso Normal</td><td>Alternativas</td></tr>
<tr><td colspan="2">1. El CU comienza cuando el PA selecciona la opción “Modificar Análisis de Suelo”.</td><td></td></tr>
<tr><td colspan="2">2. El Sistema invoca al CU Buscar Análisis de Suelo.</td><td></td></tr>
<tr><td colspan="2">3. El sistema muestra los análisis de suelo encontrados y solicita se seleccione uno para su modificación.</td><td>3. A. El Sistema no encontró ningún análisis de suelo que cumpla los criterios de búsqueda dados por el PA.
3. A.1. El Sistema consulta si se desea volver a intentar la búsqueda de análisis de suelo, solicitando que el PA seleccione una opción.
3. A.1. El PA selecciona la opción “Reintentar búsqueda”.
3. A.2. El Sistema vuelve al paso 2.

3. A.1.A. El PA selecciona la opción “Cancelar”.
3. A.1.A.1. El Sistema selecciona “Salir”.</td></tr>
<tr><td colspan="2">4. El PA selecciona un análisis de suelo del conjunto de análisis encontrados en la búsqueda.</td><td></td></tr>
<tr><td colspan="2">5. Para el análisis de suelo encontrado, el sistema muestra:
a. Campo donde se realizó el análisis de suelo.</td><td></td></tr>
</table>

b. Ubicación geográfica en donde se realizó el análisis de suelo. **c.** Fecha de realización del análisis. **d.** Laboratorio. **e.** Profundidad de la muestra, en metros. **f.** Porcentaje de materia orgánica. **g.** Conductividad del extracto **h.** Color. **i.** Nutrientes, cada uno su método de extracción. **j.** Porcentaje de humedad [Si este dato fue cargado]. **k.** Capacidad de intercambio catiónico (entero positivo). **l.** Textura (seleccionada de una lista). **m.** Tipo de suelo (seleccionado de una lista).	
6. El Sistema habilita la modificación de los datos en los intervalos permitidos, de acuerdo como se estableció en el CU Agregar Análisis de Suelo.	
7. El Sistema muestra la opción “Guardar cambios”.	
8. El PA modifica los datos deseados.	
9. El PA presiona la opción “Guardar cambios”.	
10. El Sistema registra los cambios realizados.	
11. El Sistema selecciona la opción “Salir”.	
Requerimientos No funcionales Especiales:	
Observaciones:	

Nombre: Eliminar Análisis de Suelo		ID: CU008
Actor Principal: Productor Agropecuario	Actor Secundario: No aplica	
Prioridad: ☒ Esencial ☐ Útil ☐ Deseable		
Complejidad: ☐ Alta ☒ Media ☐ Baja		
Tipo de Caso de Uso: ☒ Concreto ☐ Abstracto		
Objetivo: Realizar la eliminación lógica de un análisis de suelo previamente buscado.		
Precondiciones: El usuario debe estar logueado en el Sistema con sus respectivos permisos.		
Post-Condiciones:	**Éxito:** el análisis se eliminó de forma satisfactoria.	
	Fracaso 1: el PA no ingresa otras coordenadas para realizar una búsqueda de análisis. **Fracaso 2:** el PA no reingresa el valor de pH.	
Curso Normal		Alternativas
1. El CU comienza cuando el PA selecciona la opción “Eliminar Análisis de Suelo”.		
2. El sistema invoca al CU Buscar Análisis de Suelo.		
3. El Sistema muestra todos los análisis de suelo resultantes de la búsqueda.		3. A. El Sistema no encontró ningún análisis de suelo que cumpla los criterios de búsqueda dados por el PA. 3. A.1. El Sistema consulta si se desea volver a intentar la búsqueda de análisis de suelo, solicitando que el PA seleccione una opción. 3. A.1. El PA selecciona la opción “Reintentar búsqueda”. 3. A.2. El Sistema vuelve al paso 2. 3. A.1.A. El PA selecciona la opción “Cancelar”. 3. A.1.A.1. El Sistema selecciona “Salir”.
4. El Sistema solicita se seleccionen los análisis de suelo que se quieren eliminar.		
5. El selecciona los análisis de suelo a eliminar.		
6. El sistema solicita se confirme la eliminación lógica.		

7. El PA confirma la eliminación.	7. A. El PA no confirma la eliminación. 7. A.1. El Sistema selecciona la opción “Salir”.
8. El Sistema actualiza el estado del análisis de suelo a “Eliminado”.	
9. El Sistema selecciona la opción “Salir”.	
Requerimientos No funcionales Especiales:	
Observaciones:	

<table>
<tr><td colspan="2">Nombre: Buscar Análisis de Suelo</td><td>ID: CU009</td></tr>
<tr><td colspan="2">Actor Principal: Productor Agropecuario</td><td>Actor Secundario: No aplica</td></tr>
<tr><td colspan="3">Prioridad: ☒ Esencial ☐ Útil ☐ Deseable</td></tr>
<tr><td colspan="3">Complejidad: ☐ Alta ☒ Media ☐ Baja</td></tr>
<tr><td colspan="3">Tipo de Caso de Uso: ☒ Concreto ☐ Abstracto</td></tr>
<tr><td colspan="3">Objetivo: Encontrar un análisis de suelo de entre los análisis de suelo cargados en el Sistema.</td></tr>
<tr><td colspan="3">Precondiciones: El usuario debe estar logueado en el Sistema con sus respectivos permisos.</td></tr>
<tr><td rowspan="2">Post-Condiciones:</td><td colspan="2">Éxito: El Análisis de Suelo buscado es encontrado y retornado al invocador</td></tr>
<tr><td colspan="2">Fracaso: El Análisis de Suelo no es encontrado</td></tr>
<tr><td colspan="2">Curso Normal</td><td>Alternativas</td></tr>
<tr><td colspan="2">1. El CU comienza cuando es invocado desde el CU Modificar Análisis de Suelo o desde el CU Eliminar Análisis de Suelo.</td><td></td></tr>
<tr><td colspan="2">2. El Sistema solicita se seleccione uno de los siguientes criterios de búsqueda, y se complete su información pertinente:
2.1. Campo donde se tomó el análisis de suelo.
2.2. Ubicación geográfica en donde se realizó el análisis de suelo.
2.3. Nombre del campo (cadena de longitud entre 3 y 255).
2.4. Fecha de carga del análisis (fecha anterior a la fecha actual).
2.5. Nombre de Provincia, Departamento o Localidad (selecciona una entre las previamente cargadas).</td><td></td></tr>
<tr><td colspan="2">3. El PA selecciona el criterio de búsqueda deseado.</td><td></td></tr>
<tr><td colspan="2">4. El Sistema muestra los resultados de búsqueda, ordenados primero por fecha y luego por orden alfabético.</td><td>13. A. El Sistema no encuentra ningún análisis de suelo que satisfaga los criterios de búsqueda.</td></tr>
</table>

	13. A.1. El Sistema informa la situación a través de un mensaje, en el cual concede la opción de reintentar la búsqueda cambiando los criterios, o bien de cancelar la búsqueda. 13. A.2. El PA selecciona la opción “Reintentar búsqueda”. 13. A.3. El Sistema vuelve al paso 2. 13. A.2.A. El PA selecciona la opción “Cancelar búsqueda”. 13. A.2.A.1. El Sistema selecciona “Salir”.
5. El Sistema devuelve al CU que realizó la invocación el análisis de suelo seleccionado.	
6. El Sistema selecciona “Salir”.	
Requerimientos No funcionales Especiales:	
Observaciones:	

Paquete 03: Gestión de Reportes

Nombre: Generar Alerta Visual por Nivel Insuficiente de Nutriente en el Suelo		ID: CU116
Actor Principal: No aplica	Actor Secundario: No aplica	
Prioridad: ☒ Esencial ☐ Útil ☐ Deseable		
Complejidad: ☐ Alta ☒ Media ☐ Baja		
Tipo de Caso de Uso: ☐ Concreto ☒ Abstracto		
Objetivo: Mostrar un semáforo visual a causa de un nivel inaceptable en la concentración de al menos un nutriente en el suelo.		
Precondiciones: El usuario debe estar logueado en el Sistema con sus respectivos permisos.		
Post-Condiciones:	Éxito: el semáforo se actualiza con éxito	
	Fracaso: el semáforo no se actualiza con éxito	
Curso Normal		Alternativas
1. El CU comienza cuando el Sistema detecta que al menos un nutriente se encuentra en los niveles “insuficiente” o “excesivo” en los últimos tres análisis de suelo.		1. A. El Sistema detecta que los nutrientes se encuentran en los niveles “adecuado” o “alto” para al menos tres de los últimos cinco análisis de suelo. 1. A.1. El Sistema mantiene el semáforo de estado aceptable de nutrientes en verde. 1. A.2. El Sistema finaliza el CU.
2. El Sistema modifica el color del semáforo del estado de los nutrientes al color rojo.		
3. El sistema modifica el titulo anterior del semáforo (estado aceptable de nutrientes), por el nombre del nutriente o nutrientes seguido de “cercano al límite aceptable”.		
4. El sistema muestra un mensaje informando que existen nutrientes cercanos a los límites de aceptación, indicando para cada uno: **a.** Nutriente afectado. **b.** Concentración actual. **c.** Límite mínimo y máximo aceptable de concentración. **d.** Causas		

e. Recomendaciones.	
5. El Sistema finaliza el CU.	
Requerimientos No funcionales Especiales:	
Observaciones:	

Nombre: Generar Informe de Evolución Zonal de Nutrientes		ID: CU122
Actor Principal: Productor Agropecuario	Actor Secundario: No aplica	
Prioridad: ☐ Esencial ☒ Útil ☐ Deseable		
Complejidad: ☐ Alta ☒ Media ☐ Baja		
Tipo de Caso de Uso: ☒ Concreto ☐ Abstracto		
Objetivo: Mostrar gráfico de líneas que indique la evolución temporal del nutriente o nutrientes especificados de determinada zona.		
Precondiciones: El usuario debe estar logueado en el Sistema con sus respectivos permisos.		
Post-Condiciones:	Éxito: el gráfico se muestra con éxito.	
	Fracaso: el gráfico no pudo mostrarse por falta de información.	
Curso Normal		Alternativas
1. El CU comienza cuando el Productor Agropecuario (PA) selecciona la opción “Generar Informe de Evolución Zonal de Nutrientes”.		
2. El Sistema solicita se seleccione la zona (Localidad, Departamento, Provincia o País) de la cual se desea obtener más información, y el rango temporal a ser considerado.		
3. El Sistema muestra una lista de los nutrientes, solicitando se seleccionen aquellos que serán mostrados en el gráfico.		
4. El PA ingresa los datos solicitados.		
5. El Sistema muestra un gráfico de curvas, donde cada una representa cada nutriente seleccionado. Dichas curvas se encuentran cortadas horizontalmente por líneas rectas		5. A. El Sistema no encuentra datos en la zona solicitada. 5. A.1. El Sistema informa tal situación. 5. A.1. El Sistema ejecuta la opción “Salir”.

que marcan los valores mínimos aceptables para cada nutriente.	
6. El sistema muestra las opciones “Exportar Informe de Evolución Zonal de Nutrientes” e “Imprimir Informe de Evolución Zonal de Nutrientes”.	
7. El PA selecciona la opción “Salir”.	7. A. El Administrador selecciona la opción “Exportar Informe”. 7. A.2. El Sistema procesa el informe y devuelve el mismo en formato Excel. 7. A.1. El Sistema solicita la ruta donde se almacenará el archivo. 7. A.2. El Administrador selecciona la ruta. 7. A.3. El Sistema procede a guardar el archivo en la ruta seleccionada. 7. A.4. El Sistema selecciona la opción “Salir”. 7. B. El administrador desea imprimir el informe. 7. B.1. El Sistema solicita que se seleccione una impresora. 7. B.2. El Administrador selecciona una impresora de la lista. 7. B.3. El Sistema realiza la impresión del informe. 7. B.4. El Sistema ejecuta la opción “Salir”.
Requerimientos No funcionales Especiales:	
Fecha Creación:	Fecha Ultima Modificación:
Observaciones:	

Nombre: Generar Informe de Actividad de Usuarios		ID: CU123
Actor Principal: Administrador	Actor Secundario: No aplica	
Prioridad: ☒ Esencial ☐ Útil ☐ Deseable		
Complejidad: ☐ Alta ☒ Media ☐ Baja		
Tipo de Caso de Uso: ☒ Concreto ☐ Abstracto		
Objetivo: Obtener un listado de las actividades realizadas por los usuarios en un período de tiempo determinado.		
Precondiciones: El usuario debe estar logueado en el Sistema con sus respectivos permisos.		
Post-Condiciones:	Éxito: El Administrador ha visualizado, impreso o exportado satisfactoriamente los datos requeridos.	
	Fracaso: El Administrador no pudo obtener el reporte requerido.	
Curso Normal		Alternativas
1. El CU comienza cuando el Administrador selecciona la opción “Generar Informe de Actividad de Usuarios.		
2. El Sistema solicita se seleccione al menos uno de los siguientes criterios para filtrar usuarios: 2.1. Nombre. 2.2. Apellido. 2.3. DNI. 2.4. Legajo. 2.5. Nombre de usuario.		
3. El Administrador selecciona el o los filtros correspondientes.		
4. El Sistema solicita se completen los campos para cada uno de los criterios seleccionados.		
5. El Administrador completa los respectivos campos.		
6. El Administrador selecciona la opción “Filtrar Usuarios”.		
7. El Sistema devuelve una lista con todos los usuarios que cumplen los criterios preestablecidos.		7. A. No se encuentran usuarios que cumplan con los criterios preseleccionados. 7. A.1. El Sistema muestra la opción “Editar Filtros de Búsqueda”.

	7. A.2. El Administrador selecciona la opción “Editar Filtros de Búsqueda”. 7. A.3. El Sistema vuelve al paso 2. 7. A.2.A. El Administrador selecciona la opción “Cancelar Búsqueda”. 7. A.2.A.1. El Sistema selecciona la opción “Salir”.
8. El Sistema solicita se seleccione uno de los usuarios de la lista.	
9. El Administrador selecciona uno de los usuarios de la lista.	
10. El Sistema solicita se ingrese un rango de fechas para mostrar las actividades realizadas.	10. A. No hay actividades almacenadas de los usuarios provistos. Fin del caso de uso
11. El Administrador ingresa un rango de fechas.	
12. El Sistema muestra un informe sobre las actividades de usuario encontradas.	12. A. El Sistema no encuentra actividades almacenadas para el usuario y rango de fechas seleccionados. 12. A.1. El Sistema muestra la opción “Editar Rango de Fechas”. 12. A.2. El Administrador selecciona la opción “Editar Rango de Fechas”. 12. A.3. El Sistema vuelve al paso 10. 12. A.2.A. El Administrador selecciona la opción “Cancelar Búsqueda”. 12. A.2.A.1. El Sistema selecciona la opción “Salir”.
13. El sistema muestra las opciones “Exportar Informe de Actividad” e “Imprimir Informe de Actividad”.	
14. El Administrador selecciona la opción “Salir”.	14. A. El Administrador selecciona la opción “Exportar Listado”. 14. A.2. El Sistema procesa el listado de actividades y devuelve el mismo en formato Excel.

	14. A.1. El Sistema solicita la ruta donde se almacenará el archivo. 14. A.2. El Administrador selecciona la ruta. 14. A.3. El Sistema procede a guardar el archivo en la ruta seleccionada. 14. A.4. El Sistema selecciona la opción “Salir”. 14. B. El administrador desea imprimir el listado. 14. B.1. El Sistema solicita que se seleccione una impresora. 14. B.2. El Administrador selecciona una impresora de la lista. 14. B.3. El Sistema realiza la impresión del informe. 14. B.4. El Sistema ejecuta la opción “Salir”.
Requerimientos No funcionales Especiales:	
Fecha Creación:	Fecha Ultima Modificación:
Observaciones:	

Nombre: Generar Informe de Visitas Realizadas por Zona		ID: CU128
Actor Principal: Administrador		Actor Secundario: No aplica
Prioridad: ☐ Esencial ☒ Útil ☐ Deseable		
Complejidad: ☐ Alta ☒ Media ☐ Baja		
Tipo de Caso de Uso: ☒ Concreto ☐ Abstracto		
Objetivo: Obtener un listado de las visitas realizadas por los usuarios por zona, en un período de tiempo determinado.		
Precondiciones: El usuario debe estar logueado en el Sistema con sus respectivos permisos.		
Post-Condiciones:	Éxito: el administrador obtiene un informe de las visitas realizadas por zona	
	Fracaso: el administrador no pudo obtener el informe	
Curso Normal		Alternativas
1. El CU comienza cuando el Administrador selecciona la opción “Visualizar listado de visitas realizadas”.		
2. El Sistema solicita se ingrese un rango de fechas.		2. A. El Sistema no encontró ninguna visita registrada. 2. A.1. El Sistema informa la situación. 2. A.2. El Sistema selecciona la opción “Salir”.
3. El Sistema solicita se ingrese la zona (Localidad, Departamento, Provincia o País) de la cual se desea obtener información.		
4. El Administrador ingresa los datos solicitados.		
5. El Sistema muestra las visitas encontradas para la zona y fecha ingresadas, mostrando un informe con la cantidad total y los nombres de los usuarios que ingresaron al software en cada una de las zonas.		5. A. El Sistema no encuentra visitas almacenadas para la zona y el rango de fechas provisto. 5. A.1. El Sistema informa tal situación. 5. A.2. El Sistema ejecuta la opción “Salir”.
6. El Administrador selecciona la opción “Salir”.		6. A. El Administrador selecciona la opción “Exportar Informe de Visitas”. 6. A.2. El Sistema procesa el informe y devuelve el mismo en formato Excel.

	6. A.1. El Sistema solicita la ruta donde se almacenará el archivo. 6. A.2. El Administrador selecciona la ruta. 6. A.3. El Sistema procede a guardar el archivo en la ruta seleccionada. 6. A.4. El Sistema selecciona la opción “Salir”. 6. B. El Administrador desea imprimir el informe de visitas. 6. B.1. El Sistema solicita que se seleccione una impresora. 6. B.2. El Administrador selecciona una impresora de la lista. 6. B.3. El Sistema realiza la impresión del informe. 6. B.4. El Sistema ejecuta la opción “Salir”.
Requerimientos No funcionales Especiales:	
Observaciones: Una visita desde una zona se define como una conexión que se establece entre el Sistema y un usuario ubicado en un determinado País / Provincia / Departamento o Localidad.	

Nombre: Generar Listado de Análisis de Suelos Disponibles		ID: CU129
Actor Principal: Productor Agropecuario	Actor Secundario: No aplica	
Prioridad: ☒ Esencial ☐ Útil ☐ Deseable		
Complejidad: ☐ Alta ☒ Media ☐ Baja		
Tipo de Caso de Uso: ☒ Concreto ☐ Abstracto		
Objetivo: Brindar un listado con análisis de suelo registrados en el sistema.		
Precondiciones: El usuario debe estar logueado en el Sistema con sus respectivos permisos.		
Post-Condiciones:	Éxito: El PA ha visualizado, impreso o exportado satisfactoriamente los datos requeridos.	
	Fracaso: El sistema no posee datos para los filtros deseados y el PA no desea reingresar los datos.	
Curso Normal		Alternativas
1. El CU comienza cuando el Productor Agropecuario (PA) selecciona la opción “Generar Listado de Análisis de Suelo Disponibles”.		
2. El Sistema solicita se ingresen: **a.** Rango temporal (fecha desde y hasta, donde fecha hasta es mayor a la fecha desde). **b.** País, Provincia, Departamento, Localidad. [Al menos uno de tales criterios debe seleccionarse, partiendo desde la mayor hacia la menor jerarquía] **c.** Filtro adicionales, de los siguientes [Opcional] **i.** Nivel mínimo de nutriente en el suelo. **ii.** Estado del nutriente objetivo.		
3. El PA ingresa los parámetros solicitados.		
4. El Sistema devuelve una lista con todos los análisis de suelo que cumplen con el filtro.		4. A. No se encuentran análisis de suelo que cumplan con los criterios preseleccionados.

	4. A.1. El Sistema muestra la opción “Editar Filtros de Búsqueda”. 4. A.2. El Administrador selecciona la opción “Editar Filtros de Búsqueda”. 4. A.3. El Sistema vuelve al paso 2. 4. A.2.A. El Administrador selecciona la opción “Cancelar Búsqueda”. 4. A.2.A.1. El Sistema selecciona la opción “Salir”.
5. El Sistema muestra las opciones “Exportar Listado de Análisis de Suelo” e “Imprimir Listado de Análisis de Suelo”.	
6. El Administrador selecciona la opción “Salir”.	6. A. El Administrador selecciona la opción “Exportar Listado”. 6. A.2. El Sistema procesa el listado de Análisis de Suelo y devuelve el mismo en formato Excel. 6. A.1. El Sistema solicita la ruta donde se almacenará el archivo. 6. A.2. El Administrador selecciona la ruta. 6. A.3. El Sistema procede a guardar el archivo en la ruta seleccionada. 6. A.4. El Sistema selecciona la opción “Salir”. 6. B. El administrador desea imprimir el listado. 6. B.1. El Sistema solicita que se seleccione una impresora. 6. B.2. El Administrador selecciona una impresora de la lista. 6. B.3. El Sistema realiza la impresión del informe. 6. B.4. El Sistema ejecuta la opción “Salir”.
Requerimientos No funcionales Especiales:	
Observaciones:	

Nombre: Generar Listado de Usuarios		ID: CU130
Actor Principal: Administrador	Actor Secundario: No aplica	
Prioridad: ☒ Esencial ☐ Útil ☐ Deseable		
Complejidad: ☐ Alta ☒ Media ☐ Baja		
Tipo de Caso de Uso: ☒ Concreto ☐ Abstracto		
Objetivo: Generar un listado de los usuarios basado en diferentes filtros a elección del solicitante		
Precondiciones: El usuario debe estar logueado en el Sistema con sus respectivos permisos.		
Post-Condiciones:	Éxito: Listado de usuarios generado correctamente.	
	Fracaso: No se encuentran usuarios que cumplan los criterios seleccionados de búsqueda.	
Curso Normal		Alternativas
1. El CU comienza cuando el Administrador selecciona la opción “Generar Listado de Usuarios”.		
2. El Sistema solicita se seleccione al menos uno de los siguientes criterios para filtrar usuarios: 2.1. Nombre. 2.2. Apellido. 2.3. Es activo (realizó alguna actividad en el último mes). 2.4. Fecha desde su última actividad. 2.5. Fecha hasta su última actividad.		
3. El Administrador selecciona el o los filtros correspondientes.		
4. El Sistema solicita se completen los campos para cada uno de los criterios seleccionados.		
5. El Administrador completa los respectivos campos.		
6. El Administrador selecciona la opción “Filtrar Usuarios”.		
7. El Sistema devuelve una lista con todos los usuarios que cumplen los criterios preestablecidos.		7. A. No se encuentran usuarios que cumplan con los criterios preseleccionados. 7. A.1. El Sistema muestra la opción “Editar Filtros de Búsqueda”.

	7. A.2. El Administrador selecciona la opción “Editar Filtros de Búsqueda”. 7. A.3. El Sistema vuelve al paso 2. 7. A.2.A. El Administrador selecciona la opción “Cancelar Búsqueda”. 7. A.2.A.1. El Sistema selecciona la opción “Salir”.
15. El Sistema muestra las opciones “Exportar Listado de Usuarios” e “Imprimir Informe de Actividad”.	
16. El Administrador selecciona la opción “Salir”.	16. A. El Administrador selecciona la opción “Exportar Listado”. 16. A.2. El Sistema procesa el listado de usuarios y devuelve el mismo en formato Excel. 16. A.1. El Sistema solicita la ruta donde se almacenará el archivo. 16. A.2. El Administrador selecciona la ruta. 16. A.3. El Sistema procede a guardar el archivo en la ruta seleccionada. 16. A.4. El Sistema selecciona la opción “Salir”. 16. B. El administrador desea imprimir el listado. 16. B.1. El Sistema solicita que se seleccione una impresora. 16. B.2. El Administrador selecciona una impresora de la lista. 16. B.3. El Sistema realiza la impresión del informe. 16. B.4. El Sistema ejecuta la opción “Salir”.
Requerimientos No funcionales Especiales:	
Observaciones:	

Nombre: Generar Mapa de Situación de Suelos		ID: CU132
Actor Principal: Productor Agropecuario	Actor Secundario: No aplica	
Prioridad: ☒ Esencial ☐ Útil ☐ Deseable		
Complejidad: ☐ Alta ☒ Media ☐ Baja		
Tipo de Caso de Uso: ☒ Concreto ☐ Abstracto		
Objetivo: Mostrar mapa zonal con la situación actual de los nutrientes del suelo.		
Precondiciones: El usuario debe estar logueado en el Sistema con sus respectivos permisos.		
Post-Condiciones:	Éxito: el mapa se visualiza con éxito	
	Fracaso: el mapa no puede mostrarse con éxito	
Curso Normal		Alternativas
1. El CU comienza cuando el Productor Agropecuario (PA) selecciona la opción “Visualizar mapa de suelos”.		
2. El PA muestra en pantalla un mapa satelital de la República Argentina, dividido en sus respectivas Provincias.		
3. El PA selecciona la provincia de la que desea obtener más información.		
4. El Sistema selecciona por defecto el nutriente Nitrógeno.		
5. El Sistema muestra en un cuadro la denominación del tipo de suelo, el color del suelo predominante y una lista de los análisis de suelo cargados para la zona seleccionada.		
6. El Sistema muestra por defecto la concentración del nutriente para la zona seleccionada, en base a los análisis de suelo allí cargados.		
7. El Sistema habilita el cambio de los nutrientes que serán mostrados a Nitrógeno, Fósforo y Azufre.		7. A. El PA cambia el nutriente a Nitrógeno, Fósforo o Azufre. 7. A.1. El Sistema cambia el nutriente seleccionado al nuevo nutriente. 7. A.1. El Sistema vuelve al paso 6.
8. El Sistema habilita la selección de localidades y el cambio de departamento o provincia seleccionada.		8. A. El PA selecciona una localidad o bien cambia el departamento o provincia.

	8. A.1. El Sistema cambia al departamento, localidad o provincia seleccionado. 8. A.2. El Sistema vuelve al paso 5.
9. El PA selecciona la opción “Salir”.	9. A. El PA selecciona la opción “Exportar Mapa”. 9. A.2. El Sistema procesa el mapa y devuelve el mismo en formato PDF. 9. A.1. El Sistema solicita la ruta donde se almacenará el archivo. 9. A.2. El PA selecciona la ruta. 9. A.3. El Sistema procede a guardar el archivo en la ruta seleccionada. 9. A.4. El Sistema selecciona la opción “Salir”. 9. B. El PA desea imprimir el informe del mapa de situación. 9. B.1. El Sistema solicita que se seleccione una impresora. 9. B.2. El PA selecciona una impresora de la lista. 9. B.3. El Sistema realiza la impresión del mapa. 9. B.4. El Sistema ejecuta la opción “Salir”.
Requerimientos No funcionales Especiales:	
Observaciones:	

Nombre: Generar Reporte de Entrenamiento de Red Neuronal	ID: CU133
Actor Principal: Experto	Actor Secundario: No aplica
Prioridad: ☒ Esencial ☐ Útil ☐ Deseable	
Complejidad: ☐ Alta ☒ Media ☐ Baja	
Tipo de Caso de Uso: ☒ Concreto ☐ Abstracto	
Objetivo: Realizar un informe que reporte progreso del entrenamiento de la red neuronal.	
Precondiciones: El usuario debe estar logueado en el Sistema con sus respectivos permisos.	
Post-Condiciones:	Éxito: El CU ha finalizado correctamente.
	Fracaso:

Curso Normal	Alternativas
1. El CU comienza cuando el Experto selecciona la opción “Generar informe de entrenamiento de red neuronal”.	
2. El Sistema solicita se ingrese el rango de fechas en donde se tomarán en cuenta los entrenamientos realizados.	
3. El Experto ingresa los rangos de fechas.	
4. El Experto selecciona la opción “Generar informe”.	
5. El Sistema muestra una lista con todos los entrenamientos realizados en el rango de fechas especificado.	
6. El Sistema calcula, para cada entrenamiento realizado: **a.** La desviación promedio entre el valor de rendimiento obtenido y el valor esperado. **b.** La desviación absoluta promedio entre el valor obtenido y esperado. **c.** La desviación absoluta máxima entre el valor obtenido y esperado.	
7. El Sistema calcula los siguientes parámetros generales: **a.** La desviación promedio, absoluta y absoluta máxima	

entre todos los entrenamientos realizados. **b.** La variación de las desviaciones promedios, absolutas y máximas entre todos los entrenamientos realizados.	
8. El Sistema muestra un gráfico con el progreso de las desviaciones absolutas máximas de todos los entrenamientos. En tal gráfico muestra dos curvas: la primera con los valores de las desviaciones absolutas máximas, mientras que en la segunda muestra dichos valores promediados por la cantidad de entrenamientos que fueron realizados.	
9. El Sistema muestra un gráfico con el progreso de las desviaciones absolutas de todos los entrenamientos. En tal gráfico muestra dos curvas: la primera con los valores de las desviaciones absolutas mientras que en la segunda muestra los valores promediados por la cantidad de entrenamientos que fueron realizados.	
10. El Sistema muestra los cálculos realizados para cada uno de los entrenamientos y aquellos cálculos generales.	
11. El Sistema muestra advertencias para los siguientes valores: **a.** Desviaciones absolutas máximas que están por encima de: 50% (advertencia crítica), 30% (advertencia moderada) y 20% (advertencia leve). **b.** Desviaciones absolutas promedio por encima de: 30% (advertencia crítica), 20%	

(advertencia moderada), 10% (advertencia leve).	
12. El Experto selecciona la opción “Salir”.	12. A. El Experto selecciona la opción “Exportar Informe”. 12. A.2. El Sistema procesa el informe y devuelve el mismo en formato Excel. 12. A.1. El Sistema solicita la ruta donde se almacenará el archivo. 12. A.2. El Experto selecciona la ruta. 12. A.3. El Sistema procede a guardar el archivo en la ruta seleccionada. 12. A.4. El Sistema selecciona la opción “Salir”. 12. B. El Experto desea imprimir el listado. 12. B.1. El Sistema solicita que se seleccione una impresora. 12. B.2. El Experto selecciona una impresora de la lista. 12. B.3. El Sistema realiza la impresión del informe. 12. B.4. El Sistema ejecuta la opción “Salir”.
Requerimientos No funcionales Especiales:	
Observaciones:	

<table>
<tr><td colspan="2">Nombre: Generar Reporte de Seguimiento de Evolución de Suelos</td><td>ID: CU135</td></tr>
<tr><td colspan="2">Actor Principal: Productor Agropecuario</td><td>Actor Secundario: No aplica</td></tr>
<tr><td colspan="3">Prioridad: ☒ Esencial ☐ Útil ☐ Deseable</td></tr>
<tr><td colspan="3">Complejidad: ☐ Alta ☒ Media ☐ Baja</td></tr>
<tr><td colspan="3">Tipo de Caso de Uso: ☒ Concreto ☐ Abstracto</td></tr>
<tr><td colspan="3">Objetivo: Confeccionar un informe que detalle la evolución en los nutrientes del suelo, en base a un seguimiento.</td></tr>
<tr><td colspan="3">Precondiciones: El usuario debe estar logueado en el Sistema con sus respectivos permisos.</td></tr>
<tr><td rowspan="2">Post-Condiciones:</td><td colspan="2">Éxito: El PA ha visualizado y/o impreso satisfactoriamente los datos requeridos.</td></tr>
<tr><td colspan="2">Fracaso:</td></tr>
<tr><td colspan="2">Curso Normal</td><td>Alternativas</td></tr>
<tr><td colspan="2">1. El CU comienza cuando el Productor Agropecuario (PA) selecciona la opción “Generar Reporte de Seguimiento de Evolución de Suelos”.</td><td></td></tr>
<tr><td colspan="2">2. El Sistema muestra una lista con los planes de siembra actuales, su fecha de inicio de seguimiento y sus respectivos campos.</td><td>2. A. No existen planes de siembra en los cuales se esté efectuando seguimiento.
2. A.1. El Sistema informa tal situación.
2. A.1. El Sistema ejecuta la opción “Salir”.</td></tr>
<tr><td colspan="2">3. El Sistema solicita se seleccione un plan de siembra de la lista.</td><td></td></tr>
<tr><td colspan="2">4. El PA selecciona un plan de siembra.</td><td></td></tr>
<tr><td colspan="2">5. El PA selecciona la opción “Generar Reporte de Seguimiento”.</td><td></td></tr>
<tr><td colspan="2">6. El Sistema muestra, para cada análisis de suelo transcurrido desde la fecha de inicio del seguimiento:
a. Para cada uno de los nutrientes que se están siguiendo, un gráfico con dos curvas: una curva que muestra el valor predicho del nutriente otra curva muestra el valor esperado del nutriente.</td><td></td></tr>
</table>

b. Para cada uno de los nutrientes que se están siguiendo, el porcentaje de cambios, y entre cada valor predicho y esperado muestra las desviaciones estándar, el porcentaje de desviaciones estándar absoluto, y el porcentaje de desviaciones estándar absoluto máximo. **c.** Para cada una de las propiedades del suelo que aparezcan en todos los análisis del seguimiento, el porcentaje de cambios.	
7. El PA selecciona la opción “Salir”.	7. A. El PA selecciona la opción “Exportar Informe”. 7. A.2. El Sistema procesa el informe y devuelve el mismo en formato Excel. 7. A.1. El Sistema solicita la ruta donde se almacenará el archivo. 7. A.2. El PA selecciona la ruta. 7. A.3. El Sistema procede a guardar el archivo en la ruta seleccionada. 7. A.4. El Sistema selecciona la opción “Salir”. 7. B. El PA desea imprimir el listado. 7. B.1. El Sistema solicita que se seleccione una impresora. 7. B.2. El PA selecciona una impresora de la lista. 7. B.3. El Sistema realiza la impresión del informe. 7. B.4. El Sistema ejecuta la opción “Salir”.
Requerimientos No funcionales Especiales:	
Observaciones:	

Paquete 04: Planificación de Cultivos

<table>
<tr><td colspan="3">Nombre: Entrenar Agente Simulación Escenarios</td><td>ID: CU137</td></tr>
<tr><td colspan="2">Actor Principal: Experto</td><td colspan="2">Actor Secundario: No aplica</td></tr>
<tr><td colspan="4">Prioridad: ☒ Esencial ☐ Útil ☐ Deseable</td></tr>
<tr><td colspan="4">Complejidad: ☐ Alta ☒ Media ☐ Baja</td></tr>
<tr><td colspan="4">Tipo de Caso de Uso: ☒ Concreto ☐ Abstracto</td></tr>
<tr><td colspan="4">Objetivo: Realizar el entrenamiento del agente de simulación de escenarios, incluyendo el proceso de aprendizaje para cada objetivo de simulación.</td></tr>
<tr><td colspan="4">Precondiciones: El usuario debe estar logueado en el Sistema con sus respectivos permisos.</td></tr>
<tr><td rowspan="2">Post-Condiciones:</td><td colspan="3">Éxito: Agente entrenado.</td></tr>
<tr><td colspan="3">Fracaso: Agente no entrenado correctamente.</td></tr>
<tr><td colspan="2">Curso Normal</td><td colspan="2">Alternativas</td></tr>
<tr><td colspan="2">1. El CU comienza cuando el Experto selecciona "Entrenar Agente de Simulación de Escenarios".</td><td colspan="2"></td></tr>
<tr><td colspan="2">2. El Sistema muestra los parámetros requeridos para el entrenamiento.</td><td colspan="2"></td></tr>
<tr><td colspan="2">3. El Sistema invoca al CU Configurar Parámetros de Entrenamiento.</td><td colspan="2"></td></tr>
<tr><td colspan="2">4. El Sistema muestra la opción "Ejecutar Simulación".</td><td colspan="2"></td></tr>
<tr><td colspan="2">5. El Experto selecciona "Ejecutar Simulación".</td><td colspan="2"></td></tr>
<tr><td colspan="2">6. El Sistema invoca al CU Simular Plan.</td><td colspan="2"></td></tr>
<tr><td colspan="2">7. El Sistema muestra el conjunto de acciones planificadas.</td><td colspan="2"></td></tr>
<tr><td colspan="2">8. El Experto selecciona “Salir”.</td><td colspan="2"></td></tr>
<tr><td colspan="4">Requerimientos No funcionales Especiales:</td></tr>
<tr><td colspan="4">Observaciones:</td></tr>
</table>

Nombre: Configurar Parámetros de Entrenamiento		ID: CU138
Actor Principal: Experto	Actor Secundario: No aplica	
Prioridad: ☒ Esencial ☐ Útil ☐ Deseable		
Complejidad: ☐ Alta ☒ Media ☐ Baja		
Tipo de Caso de Uso: ☒ Concreto ☐ Abstracto		
Objetivo: Definir bajo qué configuración se realizará la simulación de escenarios.		
Precondiciones: El usuario debe estar logueado en el Sistema con sus respectivos permisos.		
Post-Condiciones:	Éxito:	
	Fracaso:	
Curso Normal		Alternativas
1. El CU inicia cuando el Experto selecciona la opción "Configurar Parámetros de Entrenamiento" desde el CU Entrenar Agente Simulación Escenarios.		
2. El Sistema llama al CU Buscar Campo.		
3. El Sistema muestra los parámetros necesarios para llevar a cabo la simulación.		
4. El Sistema solicitando se ingrese: 4.1. Objetivo (uno de los siguientes): **4.1.1.** Opción 1: Sustentabilidad. **4.1.1.1.** Fecha límite para alcanzar la sustentabilidad (fecha mayor que la fecha actual). **4.1.1.2.** Nutrientes objetivo. **4.1.1.3.** Valores a alcanzar en los mismos. **4.1.2.** Opción 2: Rendimiento. **4.1.2.1.** Intervalo de tiempo considerado. **4.1.2.2.** Nutrientes a cuidar. **4.1.2.3.** Valores mínimos de los nutrientes a cuidar. **4.1.2.4.** Tipo de restricción utilizada (agresiva o conservadora – si se		

utiliza esta última, los valores mínimos de los nutrientes a cuidar no pueden ser mayores a los valores actuales del suelo). 4.2. Campo: **4.2.1.** Campo donde transcurrirá la simulación, en base a los campos previamente cargados. 4.3. Análisis de suelo: **4.3.1.** El análisis de suelo que será considerado como estado inicial de los nutrientes del campo. 4.4. Políticas de aprendizaje del agente: **4.4.1.** α: alpha (número real, varía entre 0 y 1, valor por defecto: 0,1). **4.4.2.** γ: gamma (número real, varía entre 0 y 1, valor por defecto: 0,85). **4.4.3.** λ: lambda (número real, varía entre 0 y 1, valor por defecto: 0,5). **4.4.4.** ε: épsilon (número real, varía entre 0 y 1, valor por defecto: 0,2). 4.5. Simulación: **4.5.1.** Cantidad de episodios de entrenamiento (entero positivo, valor por defecto: 200). **4.5.2.** Cantidad máxima de pasos por episodio (entero positivo, valor por defecto: 500). **4.5.3.** Cantidad máxima de finalizaciones anormales de episodios (entero positivo, valor por defecto: 10.000).	

5. El Experto ingresa los datos solicitados.	
6. El Sistema registra la configuración.	
7. El Experto selecciona salir.	
Requerimientos No funcionales Especiales:	
Observaciones:	

Nombre: Simular Plan		ID: CU139
Actor Principal: Experto	Actor Secundario: No aplica	
Prioridad: ☒ Esencial ☐ Útil ☐ Deseable		
Complejidad: ☒ Alta ☐ Media ☐ Baja		
Tipo de Caso de Uso: ☒ Concreto ☐ Abstracto		
Objetivo: Realizar la simulación de un plan de siembra.		
Precondiciones: Los parámetros de la simulación deben estar previamente cargados y la sesión de usuario debe estar iniciada con los permisos correspondientes.		
Post-Condiciones:	Éxito:	
	Fracaso:	
Curso Normal		Alternativas
1. El CU inicia cuando el Experto selecciona la opción "Ejecutar Simulación" invocado desde el CU Entrenar Agente Simulación Escenarios.		
2. El Sistema asigna como fecha de la simulación a la fecha actual del sistema.		
3. El Sistema invoca al CU Buscar Cultivo.		
4. El Sistema no encuentra ningún cultivo actualmente plantado en el campo.		4. A. Existe un cultivo actualmente plantado en el campo. 4. A.1. El sistema continúa con el paso 9.
5. El Sistema llama al CU Buscar Opción de Cultivo.		
6. El Sistema invoca al CU Determinar Cultivo a Plantar.		
7. El Sistema invoca al CU Determinar Fertilizante a Aplicar.		
8. El Sistema registra la siembra del cultivo seleccionado y la dosis de fertilizante aplicada.		
9. El Sistema invoca al CU Simular Transcurso de Tiempo.		
10. El Sistema actualiza la nueva fecha de simulación.		
11. El Sistema invoca al CU Testear Red Neuronal de Predicción de		

Rendimiento pasando como parámetro el estado actual del campo.	
12. El Sistema calcula, de acuerdo a su base de conocimiento, las adiciones y sustracciones que recibirán los nutrientes del suelo tras la cosecha.	
13. El Sistema invoca al CU Registrar Cosecha.	
14. El Sistema llama al CU Calcular Balance de Suelo tras Cosecha.	
15. El Sistema actualiza los valores del suelo en el estado de la simulación del campo.	
16. El Sistema actualiza su base de conocimiento con las reglas aprendidas.	
17. El Sistema verifica el estado actual de los nutrientes del suelo contra el objetivo de la simulación.	17. A. El objetivo de la simulación es la Sustentabilidad 17. A.1. El Sistema compara la fecha límite con la fecha actual de la simulación, y la primera es mayor o igual que la segunda. 17. A.2. Para cada uno de los valores objetivo: El Sistema compara el valor actual simulado del nutriente con el valor objetivo. 17. A.2.A. El Sistema compara la fecha límite con la fecha actual de la simulación, y la última es mayor que la primera. 17. A.2.A.1. El Sistema registra imposibilidad de alcanzar sustentabilidad en el proceso de aprendizaje del episodio actual. 17. A.2.A.2. El Sistema reestablece el estado actual de la simulación al estado inicialmente configurado 17. A.2.A.3. El Sistema vuelve al paso 3. 17. B. El objetivo de la simulación es el Rendimiento

	17. B.1. Para cada uno de los valores objetivo: El Sistema compara el valor actual simulado con el valor mínimo permitido del nutriente 17. B.2. El Sistema verifica que el primer valor es mayor que el último. 17. B.2.A. El Sistema verifica que el primer valor es menor que el último, siendo el criterio de restricción preestablecido la “restricción no estricta” 17. B.2.A.1. El Sistema le aplica una penalización al estado actual. 17. B.2.B. El Sistema verifica que el primer valor es menor que el último, siendo el criterio de restricción preestablecido la “restricción estricta” 17. B.2.B.1. El Sistema le aplica una penalización al estado actual. 17. B.2.B.2. El Sistema reestablece el estado actual. 18. B.1.B.4. El Sistema vuelve al paso 3.
18. El Sistema determina que el estado objetivo ha sido alcanzado.	18. A. El estado objetivo no ha sido alcanzado para el objetivo de simulación Sustentabilidad 18. A.1. El Sistema consulta si se alcanzó la cantidad de pasos máxima por episodio y dicha cantidad no fue alcanzada. 18. A.2. El Sistema vuelve al paso 3. 18. A.1.A. El Sistema consulta si se alcanzó la cantidad de pasos máxima por episodio y dicha cantidad fue alcanzada 18. A.1.A.1. El Sistema asume estancamiento en el proceso de aprendizaje del episodio actual. 18. A.1.A.2. El Sistema reestablece el estado actual de la simulación al estado inicialmente configurado.

	18. A.1.A.3. El Sistema vuelve al paso 3. 18. B. El estado objetivo no ha sido alcanzado para el objetivo de simulación Rendimiento 18. B.1. El Sistema vuelve al paso 3.
19. El Sistema registra la acción realizada en el paso de simulación.	
20. El Sistema reestablece el estado actual de la simulación al estado inicialmente configurado.	
21. El Sistema verifica que el episodio finalizado haya sido el último.	21. A. El Sistema verifica que el episodio finalizado no es el último 21. A.1. El Sistema vuelve al paso 3.
22. El Sistema seleccionar Salir.	
	* En cualquier momento: *.A. El Sistema comprueba que la simulación ha alcanzado el número máximo de finalizaciones anormales de episodios (finalizaciones por cantidad excesiva de pasos, por ejemplo) *.A.1. El Sistema asume la imposibilidad de converger en la solución en las condiciones actuales y procede a detener la simulación. *.A.2. El Sistema informa que la simulación se detuvo por imposibilidad de convergencia y selecciona Salir.
Requerimientos No funcionales Especiales:	
Observaciones: “Dosis de fertilizante” se entiende por la cantidad de Nitrógeno, Fósforo y Azufre que debe llevar el fertilizante que se aplicará en la siembra del cultivo.	

Nombre: Generar Plan de Siembra		ID: CU140
Actor Principal: Productor Agropecuario	Actor Secundario: No aplica	
Prioridad: ☒ Esencial ☐ Útil ☐ Deseable		
Complejidad: ☐ Alta ☒ Media ☐ Baja		
Tipo de Caso de Uso: ☒ Concreto ☐ Abstracto		
Objetivo: Generar un plan de siembra a futuro en base a las reglas aprendidas y a un objetivo determinado.		
Precondiciones: El usuario debe estar logueado en el Sistema con sus respectivos permisos.		
Post-Condiciones:	Éxito: Plan de siembra generado exitosamente.	
	Fracaso:	
Curso Normal		Alternativas
1. El Sistema llama al CU Buscar Campo.		
2. El Sistema solicita: 2.1. Objetivo de planificación (uno de los siguientes): **2.1.1.** Sustentabilidad. **2.1.1.1.** Fecha límite para alcanzar la sustentabilidad (fecha mayor que la fecha actual). **2.1.1.2.** Nutrientes objetivo. **2.1.1.3.** Valores a alcanzar en los mismos. **2.1.2.** Rendimiento (uno de los siguientes): **2.1.2.1.** Restricción: **2.1.2.1.1.** Estricta. **2.1.2.1.2.** No estricta. **2.1.2.2.** Intervalo de la simulación (número entero positivo, en años, valor por defecto: 5). **2.1.2.3.** Nutrientes a cuidar. **2.1.2.4.** Valor mínimo que pueden tomar dichos nutrientes.		

2.2. Número máximo de pasos por episodio (entero positivo, valor por defecto: 500). 2.3. Campo donde transcurrirá la simulación, en base a los campos previamente cargados en el sistema. 2.4. Análisis de suelo que será considerado como estado inicial de los nutrientes del campo.	
3. El Productor Agropecuario ingresa los datos solicitados.	
4. El Sistema habilita la opción “Efectuar Planificación”.	
5. El Sistema invoca al CU Seleccionar Política de Siembra.	
6. El Sistema muestra el plan de siembra resultante.	
7. Para cada acción de siembra aplicada 7.1. El Sistema muestra: **7.1.1.** Temporada (mes de año). **7.1.2.** Nombre del cultivo a sembrar.	
8. El Sistema muestra la opción “Guardar Plan de Siembra”.	
9. El Productor Agropecuario selecciona la opción “Guardar Plan de Siembra”.	9. A. El Productor Agropecuario no desea guardar el plan de siembra. 9. A.1. El Productor Agropecuario presiona “Salir”.
10. El Sistema invoca al CU Alta Plan de Siembra.	
11. El Productor Agropecuario presiona “Salir”.	
Requerimientos No funcionales Especiales:	
Observaciones:	

Nombre: Seleccionar Política de Siembra		ID: CU141
Actor Principal: Productor Agropecuario	Actor Secundario: No aplica	
Prioridad: ☒ Esencial ☐ Útil ☐ Deseable		
Complejidad: ☐ Alta ☒ Media ☐ Baja		
Tipo de Caso de Uso: ☒ Concreto ☐ Abstracto		
Objetivo: Obtener una secuencia de acciones de siembra que lleven al campo desde el estado inicial hacia el estado final donde cumple con el objetivo preestablecido.		
Precondiciones: El usuario debe estar logueado en el Sistema con sus respectivos permisos.		
Post-Condiciones:	Éxito: Política de siembra registrada exitosamente.	
	Fracaso:	
Curso Normal		Alternativas
1. El CU comienza cuando es invocado desde CU Generar Plan de Siembra.		
2. El Sistema asigna como estado inicial al estado inicial del campo.		
3. Mientras no se haya alcanzado el estado final o el número máximo de pasos previamente configurados: 3.1. El Sistema registra el estado actual. 3.2. El Sistema invoca al CU Buscar Acción de Siembra. 3.3. El Sistema selecciona la acción de siembra que devuelva mayor recompensa. 3.4. El Sistema invoca al CU Testear Red Neuronal de Predicción de Rendimiento pasando como parámetro el estado actual del campo. 3.5. El Sistema le aplica la acción seleccionada al estado. 3.6. El Sistema obtiene un nuevo estado. 3.7. El Sistema obtiene la recompensa relacionada con la aplicación de la acción para el estado dado. 3.8. El Sistema registra: **3.8.1.** Acción de siembra.		3.3. A. El Sistema encuentra acciones de siembra con igual recompensa. 3.3. A.1. El Sistema selecciona aleatoriamente una acción del conjunto de acciones que devuelvan igual recompensa. 3.3. A.2. El Sistema continúa en el paso 3.4.

3.8.2. Recompensa. **3.8.3.** Estado.	
4. El Sistema ejecuta la opción “Salir”.	
Requerimientos No funcionales Especiales:	
Observaciones: “Acción de siembra” se considera a toda acción que involucra la siembra, fertilización y cosecha de un determinado cultivo en una determinada temporada.	

Nombre: Buscar Acción de Siembra		ID: CU142
Actor Principal: Productor Agropecuario	Actor Secundario: No aplica	
Prioridad: ☒ Esencial ☐ Útil ☐ Deseable		
Complejidad: ☐ Alta ☒ Media ☐ Baja		
Tipo de Caso de Uso: ☒ Concreto ☐ Abstracto		
Objetivo: Buscar acciones de siembra para el estado actual del campo en base al aprendizaje previamente realizado.		
Precondiciones: El usuario debe estar logueado en el Sistema con sus respectivos permisos.		
Post-Condiciones:	Éxito: Acción de siembra exitosamente encontrada.	
	Fracaso:	
Curso Normal		Alternativas
1. El CU comienza cuando es invocado desde CU Seleccionar Política de Siembra.		
2. El Sistema asigna como criterio de búsqueda al estado actual del campo (incluyendo el objetivo).		
3. El Sistema busca en su base de conocimiento las acciones de siembra para el estado dado.		
4. Para cada acción de siembra encontrada, el Sistema agrega la acción a una lista junto con su valor Q resultante de aplicarla.		4. A. El Sistema no encuentra acciones de siembra para el estado actual del campo 4. A.1. El Sistema agrega a la lista una acción por defecto para ese estado.
5. El Sistema ejecuta la opción “Salir”.		
Requerimientos No funcionales Especiales:		
Observaciones:		

Paquete 05: Seguridad

Nombre: Nueva Opinión de Usuario		ID: CU167
Actor Principal: Usuario	Actor Secundario: No aplica	
Prioridad: ☐ Esencial ☒ Útil ☐ Deseable		
Complejidad: ☐ Alta ☒ Media ☐ Baja		
Tipo de Caso de Uso: ☒ Concreto ☐ Abstracto		
Objetivo: Registrar una opinión, sugerencia, comentario o consulta de un usuario logueado en el sistema.		
Precondiciones: El usuario debe estar logueado en el Sistema con sus respectivos permisos.		
Post-Condiciones:	Éxito: la opinión se registró con éxito.	
	Fracaso:	
Curso Normal		Alternativas
1. El CU comienza cuando el Usuario selecciona la opción “Registrar opinión”.		
2. El Sistema muestra un área de texto para escribir.		
3. El Usuario escribe su opinión en el área especificada para tal efecto.		
4. El Usuario selecciona la opción “Registrar”.		
5. El Sistema verifica que la opinión no excede los 500 caracteres.		5. A. El Sistema verifica que la opinión excede los 500 caracteres. 5. A.1. El Sistema informa tal situación. 5. A.2. El Sistema borra los caracteres excedentes.
6. El Sistema registra la opinión.		
7. El Sistema ejecuta la opción “Salir”.		
Requerimientos No funcionales Especiales:		
Observaciones:		

Parte V – Requerimientos Adicionales

5.1. Requerimientos No Funcionales

5.1.1. La interfaz gráfica de usuario debe ser intuitiva, amigable, sin exceso de botones ni de información, similar a la de otro software utilizado por los usuarios y utilizar vocabulario común para los productores.

5.1.2. El estado actual del suelo debe ser representado con gráficos que muestren en colores los niveles de nutrientes actuales, nivel mínimo de nutrientes, etcétera.

5.1.3. Los Mapas deben ser representados con colores diferentes aquellas zonas críticas y delimitaciones de regiones vecinas con características de suelo diferentes.

5.1.4. El software debe ser una aplicación web, debiendo poder visualizarse en cualquier navegador web para alentar al uso masivo de la misma.

5.1.5. El sistema deberá poder guardar informes generados en formato .docx, .xlsx y .pdf.

5.2. Requerimientos de Licencia

5.2.1. Concesión de Licencia de Uso del Servicio

La aplicación de gestión de agroecosistemas será concedida al Instituto Nacional de Tecnología Agropecuaria (INTA). Dicha institución estará habilitada a utilizar el sistema tal como lo detalla la funcionalidad descrita en el presente documento, por tiempo ilimitado y sin costo alguno.

Esta licencia además habilita a la institución a permitir el uso del sistema a todos los usuarios que considere necesarios e involucrados.

En caso que la institución desee incorporar funcionalidades, módulos, o conexiones al sistema, deberá contactarse con el grupo de desarrollo Solum para negociar los costos y tiempos que implicarán dichos cambios.

5.2.2 Limitaciones de la Licencia

Todo usuario del sistema acepta que bajo ninguna circunstancia:

a. Modificará o provocará la modificación de los archivos generados por el sistema de manera automática.

b. Facilitará, creará ni mantendrá cualquier tipo de conexión no autorizada con el sistema.

c. Alterará ni colaborará en la alteración de ningún componente interno del sistema, o parte del mismo.

d. Explotará el sistema o alguna parte de él con fines de lucro.

e. Ingresará deliberadamente datos erróneos o intentará corromper la información manejada por el sistema.

Parte VI – Anexo: Cambios en Requerimientos

A modo de anexo, se incluye un listado con los cambios de requerimientos que fueron efectuados en el presente documento desde la finalización del cursado hasta la fecha. Tal listado se divide entre cambios en Casos de Uso y cambios en Actores. En cuanto a los primeros, los Casos de Uso utilizan los nombres y numeraciones que se les había asignado anteriormente. En cuanto al Estado, el mismo define qué cambios se realizaron en el Caso de Uso, si es que hubo alguno.

Referencias:

- Sin cambios. Ningún cambio ha sido efectuado en el Caso de Uso, aparte de algún cambio mínimo de formato o redacción.
- Renombrado. El Caso de Uso tiene la misma funcionalidad que antes, pero su nombre fue cambiado por otro considerado más apropiado con la notación utilizada.
- Reemplazado por otro caso de uso. La funcionalidad básica que tenía el Caso de Uso pasa a formar parte de otro caso de uso.
- Reemplazado. Aplicable a los Casos de Uso que hacían uso de algún mecanismo Inteligencia Artificial. Aquellos con tal leyenda han sido reemplazados por otros Casos de Uso que no tienen una relación biyectiva con los nuevos. Como ejemplo, considere un nuevo Caso de Uso llamado Testear Red Neuronal de Predicción de Rendimiento. El mismo no puede relacionarse con Casos de Uso anteriores, ya que anteriormente no estaba contemplado el uso de Redes Neuronales Artificiales.
- No mantenido. Se ha decidido no mantener el Caso de Uso para la presente Especificación de Requerimientos. Esto puede deberse a una serie de razones, como por ejemplo que un Caso de Uso fue reevaluado y no se encontró el valor agregado que aportaba su preservación, o bien que un Caso de Uso excedía el alcance del proyecto.

N° CU	Nombre de CU	Estado
36	Activar fase de entrenamiento en clúster	Reemplazado
33	Administrar análisis de suelo	Sin cambios

12	Administrar campo	Sin cambios
50	Administrar caso de experto	Reemplazado
15	Administrar componente de suelo	Reemplazado por Administrar Colores de Suelo, Administrar Texturas de Suelo
21	Administrar cultivo	Sin cambios
81	Administrar departamento	Sin cambios
60	Administrar estación climatológica	No mantenido (tal información no va a ser considerada por el momento, se cargarán directamente las precipitaciones sobre la temporada)
90	Administrar fenómeno climatológico	No mantenido (tal funcionalidad se encuentra cubierta al considerar precipitaciones y radiaciones solares)
18	Administrar fertilizante	Sin cambios
69	Administrar laboratorio	Sin cambios
9	Administrar localidad	Sin cambios
24	Administrar nutriente	Sin cambios
59	Administrar opinión de usuarios	Sin cambios
6	Administrar país	Sin cambios
75	Administrar perfil usuario	Renombrado a Administrar Grupos de Usuario
53	Administrar política de decisión	Reemplazado
93	Administrar propiedad de suelo	No mantenido (dicha funcionalidad ya se encuentra especificada en clases como Textura y Color)
3	Administrar provincia	Sin cambios

63	Administrar reporte clima	Renombrado a Administrar Precipitaciones
66	Administrar reporte radiación solar	Sin cambios
84	Administrar rol usuario	Sin cambios
78	Administrar tipo año	No mantenido (tal información no va a ser considerada por el momento)
72	Administrar tipo clima	No mantenido (tal información no va a ser considerada por el momento)
27	Administrar tipo de cultivo	Sin cambios
87	Administrar tipo suelo	Sin cambios
30	Administrar usuario	Sin cambios
96	Administrar zona de carta de suelo	No mantenido (tal funcionalidad se encuentra cubierta por los análisis de suelos)
107	Calcular cambios en variantes de suelo	Reemplazado
103	Calcular clústers	Reemplazado
109	Calcular déficit hídrico por tiempo y zona	No mantenido (tal funcionalidad requiere conocimiento adicional complejo, el cual no será considerado por el momento)
104	Calcular políticas de decisión	Reemplazado
106	Calcular predicción de estado de suelo a futuro	Reemplazado
108	Calcular recomendaciones de mejora	Reemplazado
110	Calcular rendimiento medio, posible, simulado	Reemplazado

114	Calcular riesgo salinidad	Reemplazado por Generar Alerta de Salinización de Suelo
112	Calcular saldo de nutrientes según rendimiento y fertilización	Reemplazado
105	Calcular simulación	Reemplazado
115	Enviar mail de información al usuario	No mantenido. No queda claro qué información de utilidad le sería enviada al usuario.
117	Enviar sms de información al usuario	No mantenido. No queda claro qué información de utilidad le sería enviada al usuario.
39	Evaluar plan de acción	Renombrado a Generar Reporte de Seguimiento de Evolución del Suelo
40	Finalizar fase de entrenamiento en clúster	Reemplazado
2	Finalizar sesión de usuario	Sin cambios
56	Generar alerta visual por cantidad de un nutriente del suelo cercano al límite de aceptación	Renombrado a Generar Alerta Visual por Nivel Inadecuado de Nutriente en el Suelo
58	Generar gráfico de evolución temporal de nutrientes en una zona determinada	Renombrado a Generar Gráfico de Evolución Regional de Nutrientes
37	Generar informe de actividad de usuarios	Sin cambios
41	Generar informe de clúster	Reemplazado
42	Generar informe de decisiones	Reemplazado
43	Generar informe de entrenamiento	Reemplazado por CU Generar Reporte de

		Entrenamiento de Red Neuronal
101	Generar informe de errores	Sin cambios
44	Generar informe de evolución de suelos	Renombrado a Generar Reporte de Seguimiento de Evolución de Suelos
100	Generar informe de opinión de usuarios	Sin cambios
99	Generar informe de precisión de entrenamiento	Reemplazado por Generar Reporte de Entrenamiento de Red Neuronal
111	Generar informe de rendimientos	Sin cambios
115	Generar informe de riesgo salinidad	Sin cambios
113	Generar informe de saldo de nutrientes	Sin cambios
57	Generar informe de visitas realizadas por zona en un intervalo de tiempo determinado	Sin cambios
45	Generar listado de análisis de suelo disponibles	Sin cambios
38	Generar listado de usuarios	Sin cambios
49	Generar mapa de concentración de un nutriente por zonas	Sin cambios
48	Generar mapa de situación de suelos	Sin cambios
46	Generar plan de acción	Reemplazado
1	Iniciar sesión de usuario	Sin cambios
102	Recalcular plan de acción	Reemplazado
47	Simulación de escenarios	Reemplazado

Actores

Nombre	Rol	Estado

Administrador	Persona que realiza la administración del sistema, asegurando su correcto funcionamiento para sus usuarios y las actividades que ellos realizan.	Sin cambios
Administrador de Negocio	Encargado de la administración y carga de datos tales como: localidades, provincias y países; campos; componentes; fertilizantes; nutrientes; cultivos y tipos de cultivos.	Renombrado a Productor Agropecuario, ya que de lo contrario su nombre es demasiado abstracto.
Responsable de Análisis de Suelo	Es quien se encarga de realizar la carga de los datos incluidos en los análisis de suelos.	Combinado con Productor Agropecuario, ya que su función es totalmente realizable por el mismo.
Supervisor de Entrenamiento	Es el responsable de controlar cómo se lleva a cabo el entrenamiento y desarrollo del agente, verificando la calidad de sus deducciones y decisiones, y corrigiendo y mejorando las políticas.	Renombrado a Experto. Desde ahora se considerará a dicha persona como quien se encargue de toda la parte de entrenamiento del agente.
Supervisor de Escenarios	Es quien realiza las simulaciones basadas en distintos escenarios personalizados, de modo que pueda comprenderse cómo reaccionarían los suelos ante diferentes situaciones (y no estén limitados solamente a los escenarios actual y futuro).	No mantenido, ya que dicha funcionalidad la realizan tanto el Experto como el Productor Agropecuario.
Supervisor de Sustentabilidad	Es quien se encarga de preservar y mejorar el	No mantenido, pues dicha funcionalidad la

	balance de los suelos, generando y evaluando los distintos planes de acción para tomar medidas pertinentes.	puede hacer directamente el productor agropecuario.
Usuario del Sistema (nombre corto: Usuario)	Cualquier persona autorizada a acceder al sistema.	Sin cambios

Capítulo IV – Modelo de Análisis

1. Introducción

En el presente documento se exhibe el Modelo de Análisis del proyecto “Gestión de agroecosistemas con predicción a futuro”.

En el mismo se exponen las Clases de Análisis y el Diagrama de Clases de Análisis principal del Sistema, que muestra las principales clases necesarias y sus relaciones con las que se modela el dominio de los campos y dan soporte para la simulación.

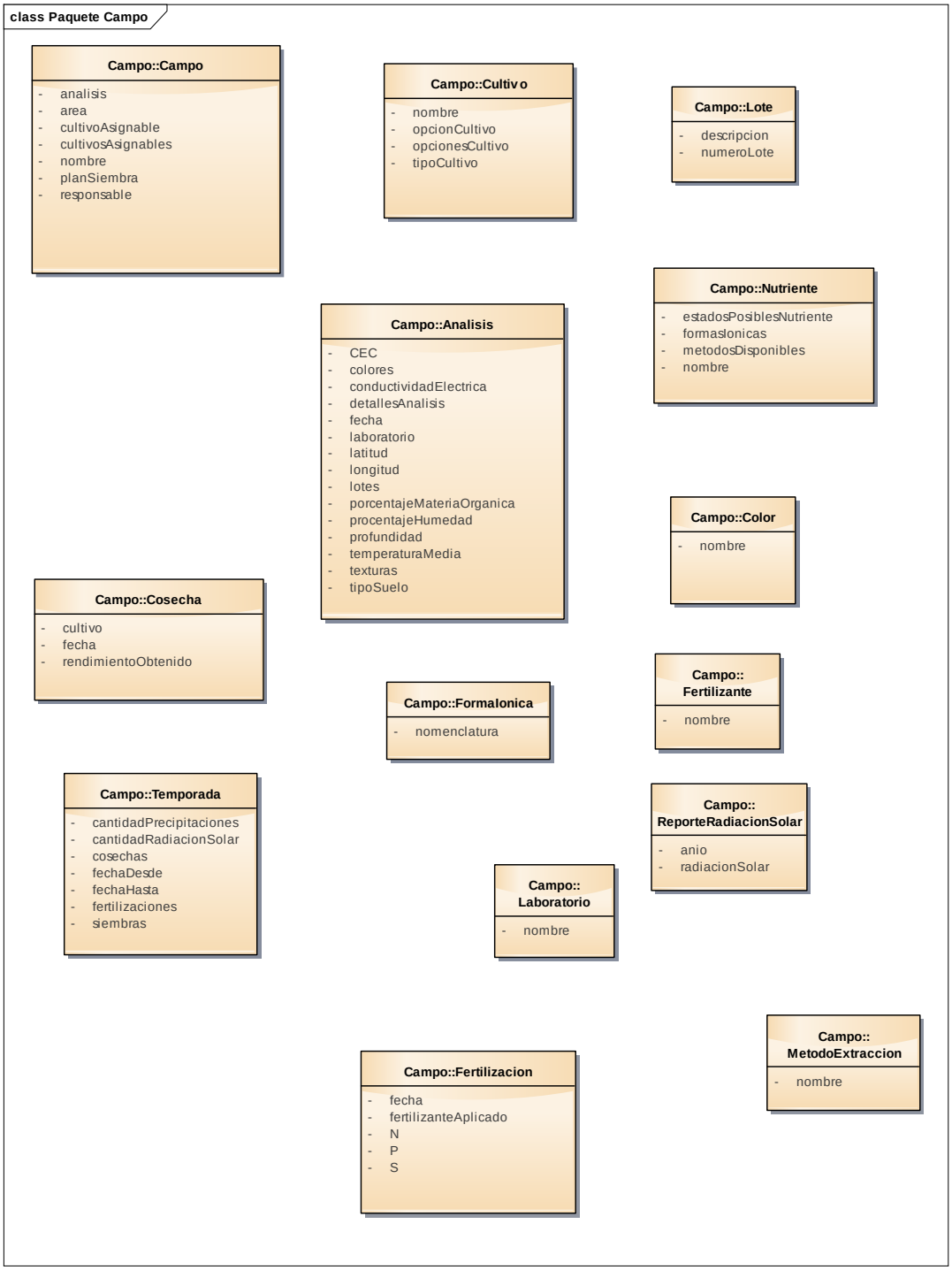
Adicionalmente, se muestran los principales Diagramas de Secuencia (desde el punto de vista del análisis) del sistema.

2. Clases de Análisis

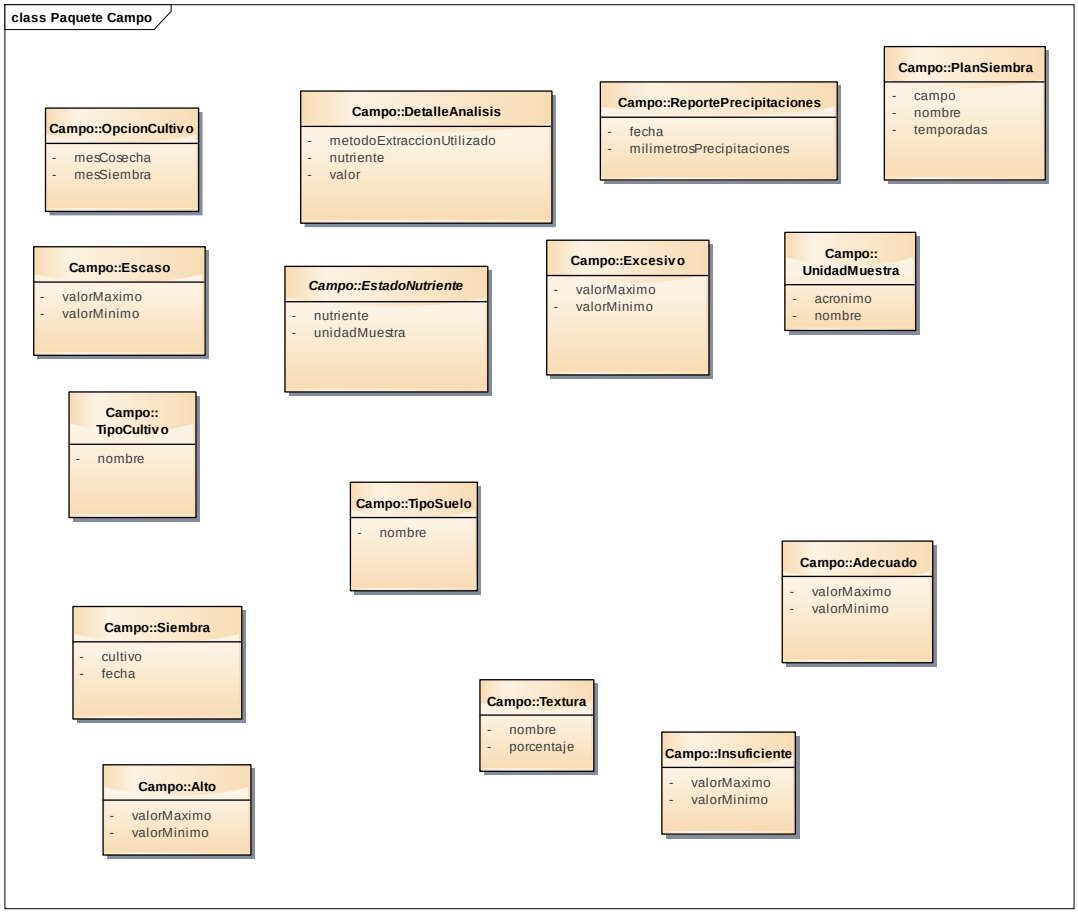
2.1. Introducción

En los presentes diagramas se muestra cada una de las clases de análisis que componen los respectivos paquetes.

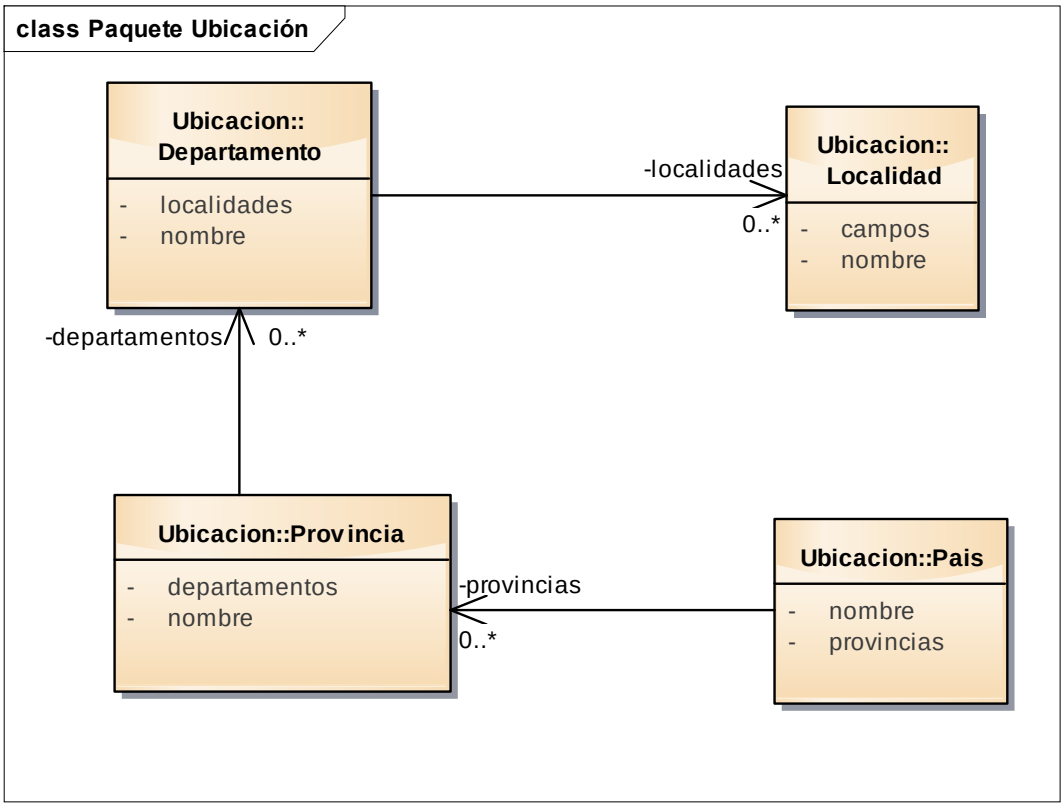
2.2. Paquete Campo (1)



2.3. Paquete Campo (2)

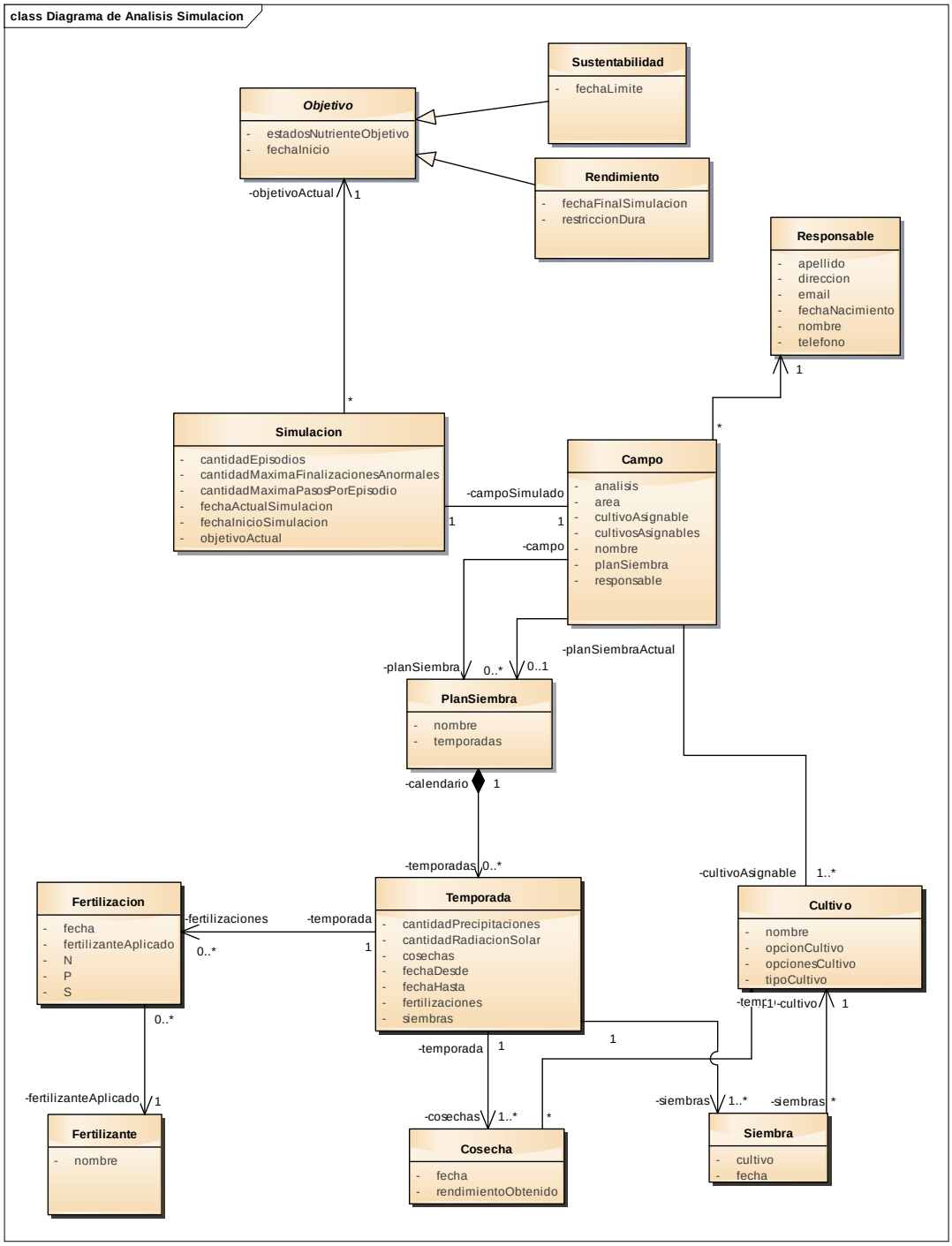


2.4. Paquete Ubicación

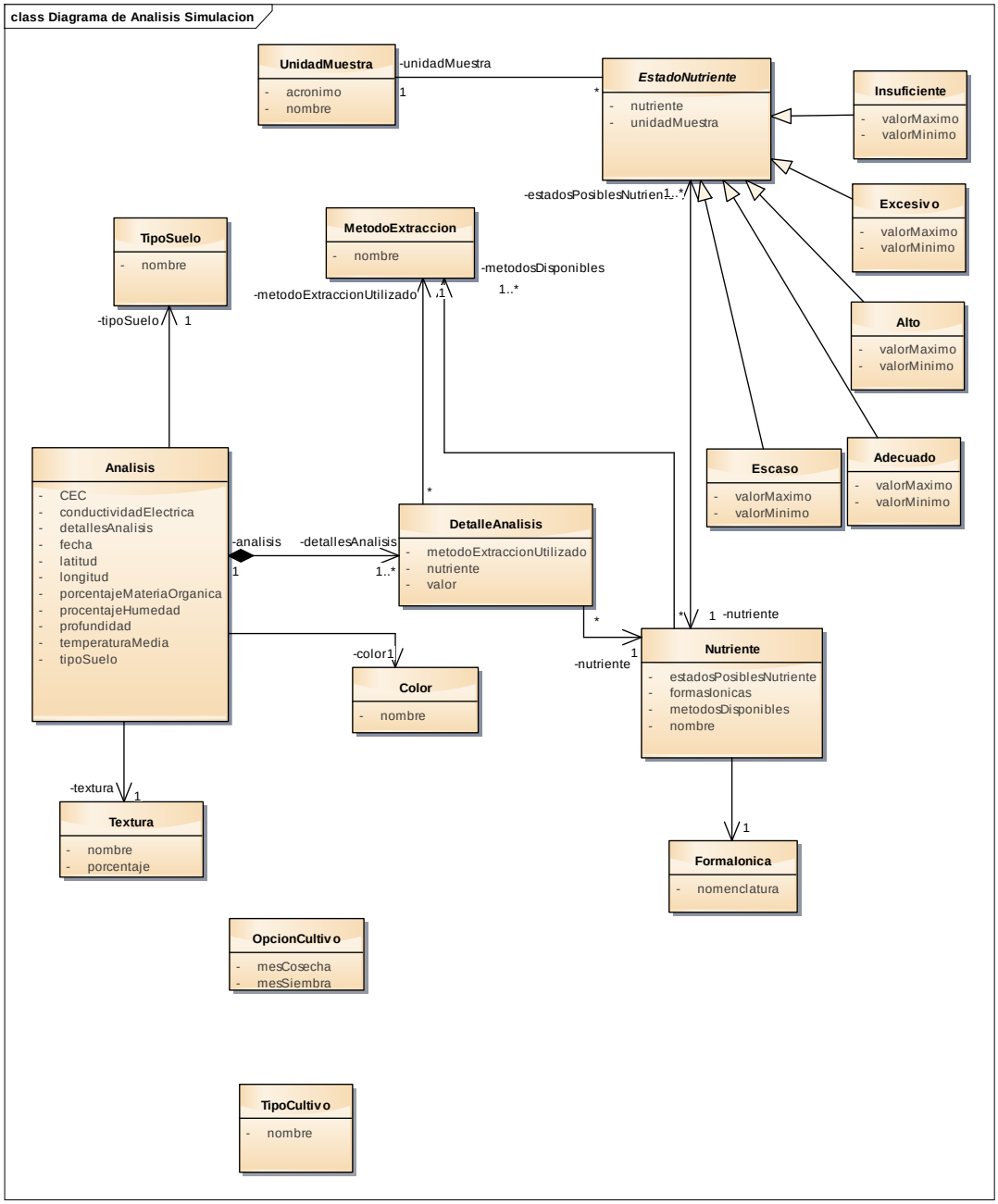


3. Diagramas de Clases de Análisis

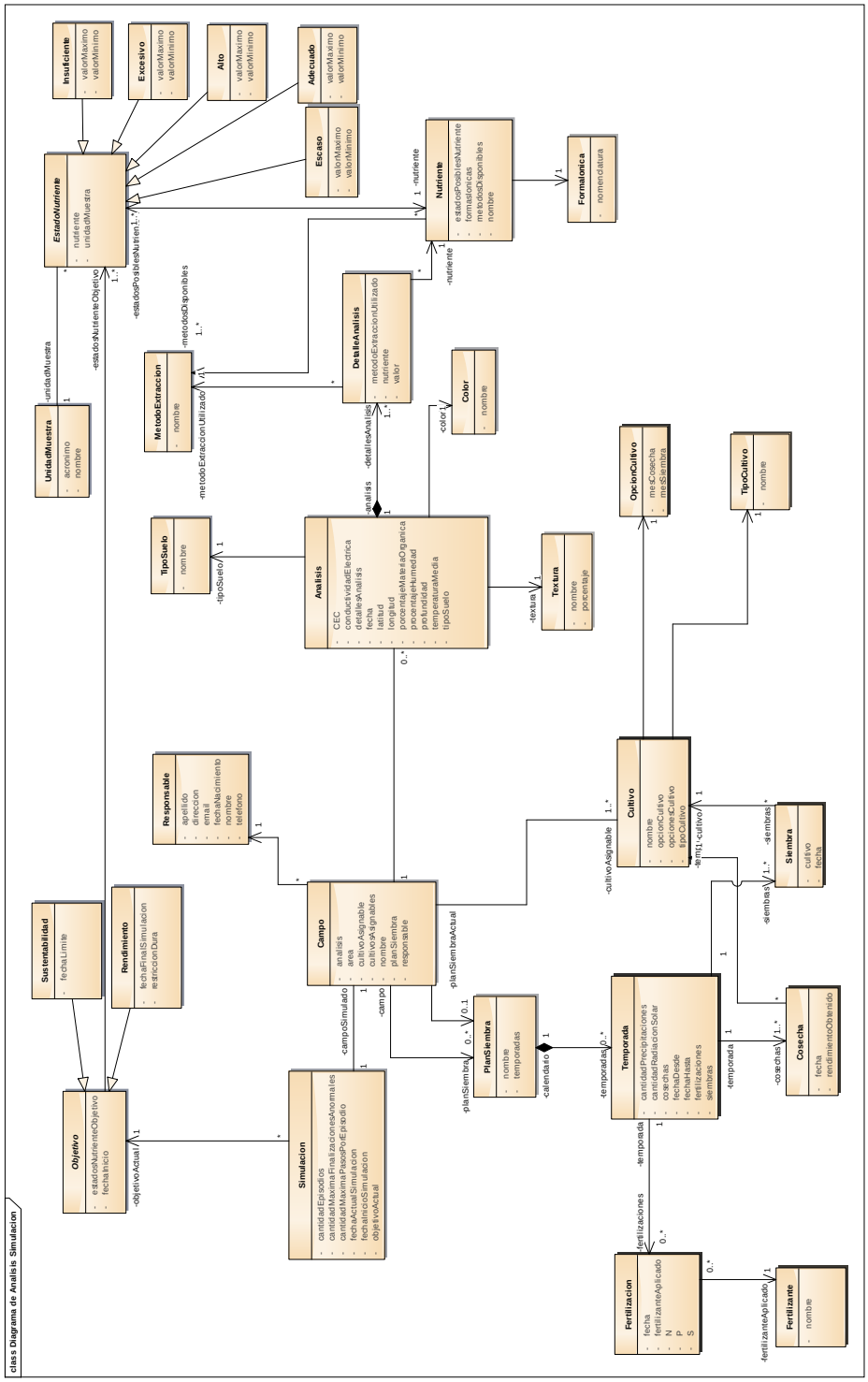
3.1. Primera Parte


class Diagrama de Analisis Simulacion
Sustentabilidad
- fechaLimite
Objetivo
- estadosNutrienteObjetivo
- fechaInicio
Rendimiento
- fechaFinalSimulacion
- restriccionDura
-objetivoActual
1
Responsable
- apellido
- direccion
- email
- fechaNacimiento
- nombre
- telefono
1
*
*
Simulacion
- cantidadEpisodios
- cantidadMaximaFinalizacionesAnormales
- cantidadMaximaPasosPorEpisodio
- fechaActualSimulacion
- fechaInicioSimulacion
- objetivoActual
-campoSimulado
1
1
Campo
- analisis
- area
- cultivoAsignable
- cultivosAsignables
- nombre
- planSiembra
- responsable
-campo
-planSiembra
0..*
0..1
-planSiembraActual
PlanSiembra
- nombre
- temporadas
-calendario
1
-temporadas
0..*
-cultivoAsignable
1..*
Fertilizacion
- fecha
- fertilizanteAplicado
- N
- P
- S
-fertilizaciones
0..*
-temporada
1
Temporada
- cantidadPrecipitaciones
- cantidadRadiacionSolar
- cosechas
- fechaDesde
- fechaHasta
- fertilizaciones
- siembras
Cultivo
- nombre
- opcionCultivo
- opcionesCultivo
- tipoCultivo
-temp
1
-cultivo
1
0..*
-temporada
1
1
-fertilizanteAplicado
1
Fertilizante
- nombre
-cosechas
1..*
*
-siembras
1..*
-siembras
*
Cosecha
- fecha
- rendimientoObtenido
Siembra
- cultivo
- fecha

3.2. Segunda Parte

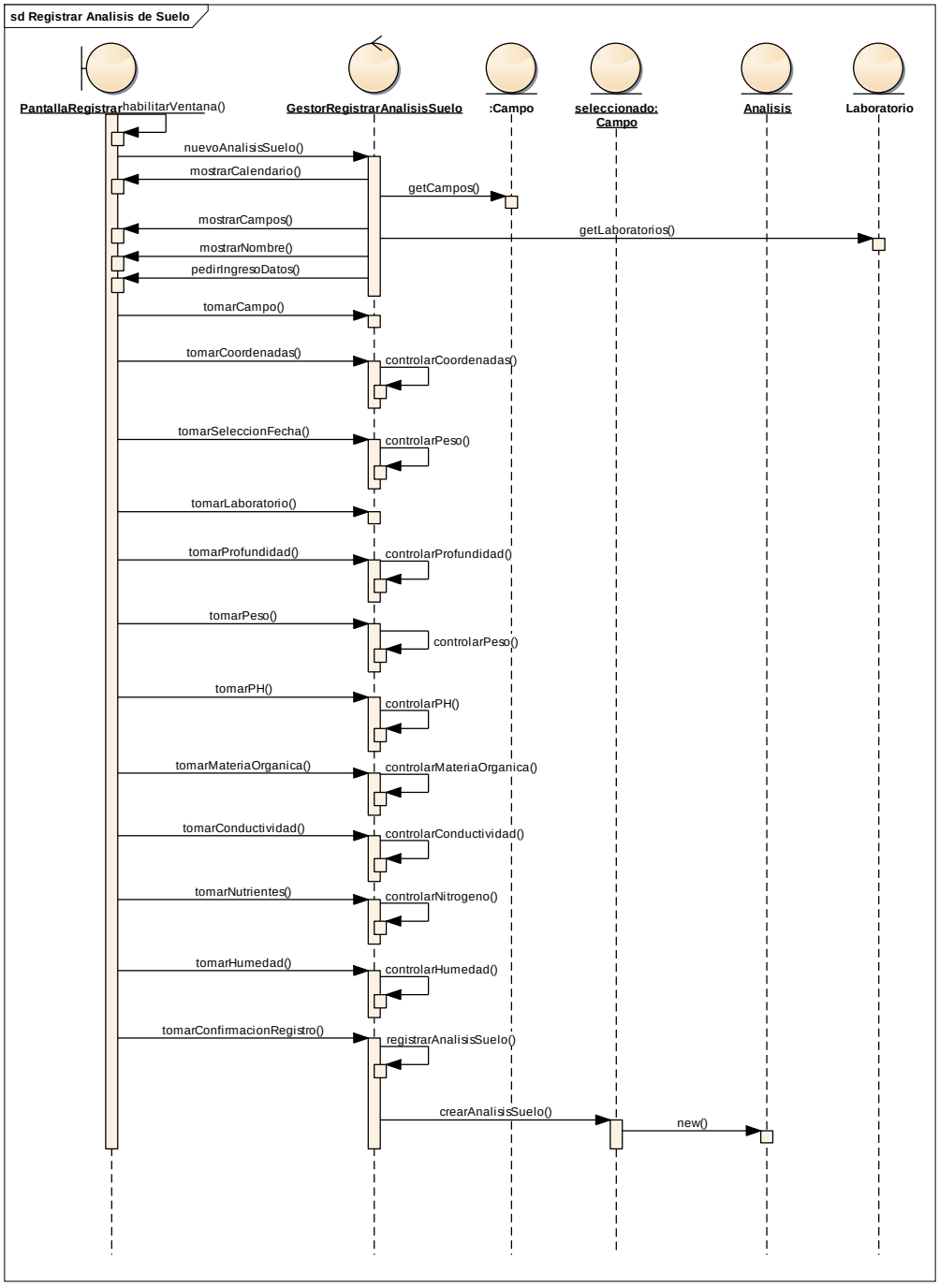


3.3. Vista Completa

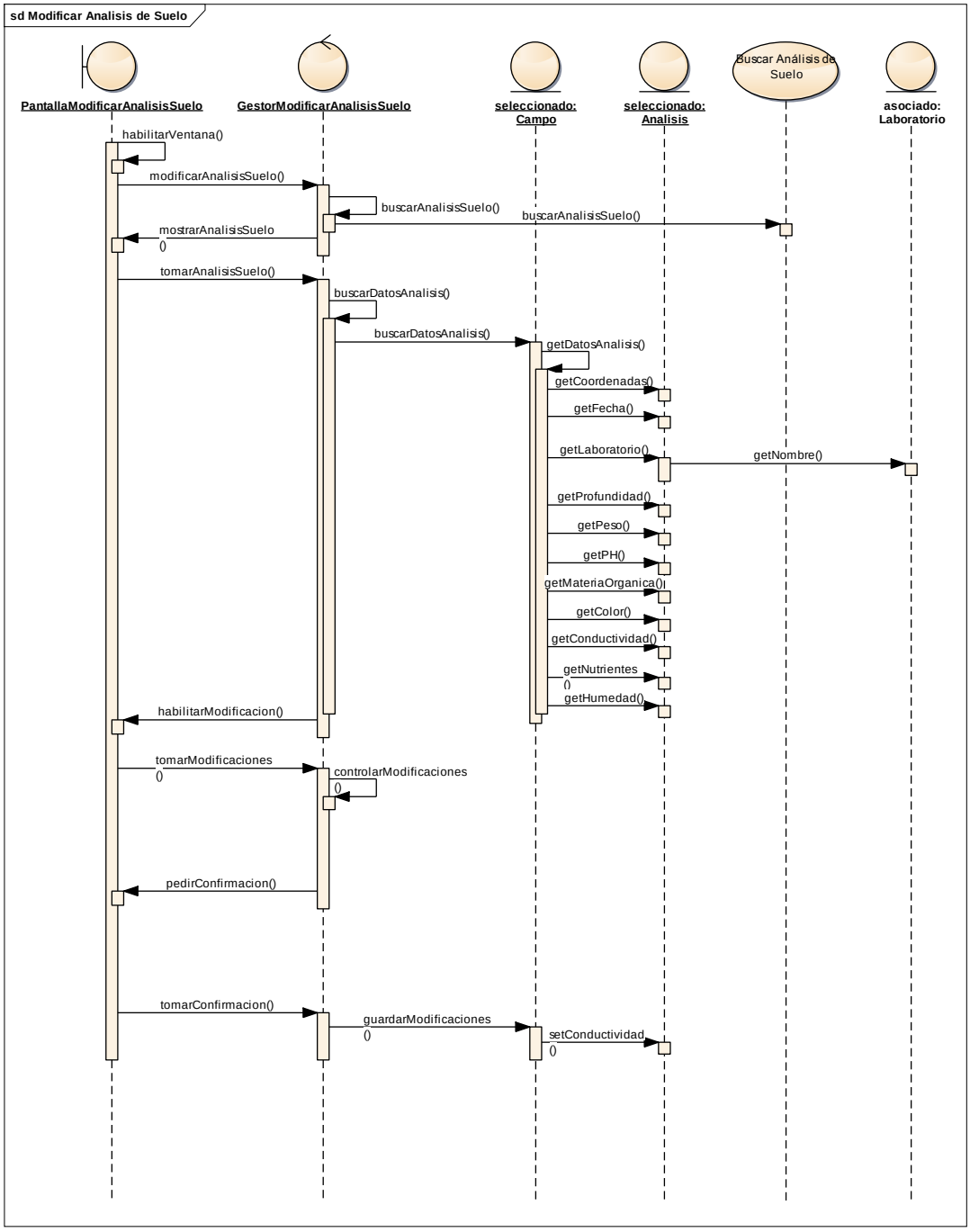

class Diagrama de Analisis Simulacion
Insuficiente
- valorMaximo
- valorMinimo
Excesivo
- valorMaximo
- valorMinimo
Alto
- valorMaximo
- valorMinimo
Adecuado
- valorMaximo
- valorMinimo
Escaso
- valorMaximo
- valorMinimo
EstadoNutriente
- nutriente
- unidadMuestra
Nutriente
- estadosPosiblesNutriente
- formasIonicas
- metodosDisponibles
- nombre
Formaionica
- nomenclatura
UnidadMuestra
- acronimo
- nombre
MetodoExtraccion
- nombre
DetalleAnalisis
- metodoExtraccionUtilizado
- nutriente
- valor
Color
- nombre
TipoSuelo
- nombre
Analisis
- CEC
- conductividadElectrica
- detalleAnalisis
- fecha
- latitud
- longitud
- porcentajeMateriaOrganica
- porcentajeHumedad
- profundidad
- temperaturaMedia
- tipoSuelo
Textura
- nombre
- porcentaje
OpcionCultivo
- mesCosecha
- mesSiembra
TipoCultivo
- nombre
Responsable
- apellido
- direccion
- email
- fechaNacimiento
- nombre
- telefono
Cultivo
- nombre
- opcionCultivo
- opcionesCultivo
- tipoCultivo
Siembra
- cultivo
- fecha
Sustentabilidad
- fechaLimite
Rendimiento
- fechaFinalSimulacion
- restriccionDura
Campo
- analisis
- area
- cultivoAsignable
- cultivosAsignables
- nombre
- plantSiembra
- responsable
PlanSiembra
- nombre
- temporadas
Temporada
- cantidadPrecipitaciones
- cantidadRadiacionSolar
- cosechas
- fechaDesde
- fechaHasta
- fertilizaciones
- siembras
Cosecha
- fecha
- rendimientoObtenido
Objetivo
- estadosNutrienteObjetivo
- fechaInicio
Simulacion
- cantidadEpisodios
- cantidadMaximaFinalizacionesAnormales
- cantidadMaximaPasosPorEpisodio
- fechaDeSimulacion
- fechaInicioSimulacion
- objetivoActual
Fertilizacion
- fecha
- fertilizanteAplicado
- N
- P
- S
Fertilizante
- nombre
1 -nutriente
-estadosPosiblesNutrien 1..*
-estadosNutrienteObjetivo
-unidadMuestra
-metodosDisponibles 1..*
-metodoExtraccionUtilizado
-detallesAnalisis 1..*
-nutriente
-color
-analisis
-tipoSuelo
-textura
-cultivoAsignable 1..*
-temp1 -cultivo
-siembras 1..*
-planSiembraActual
-campoSimulado
-campo
-planSiembra 0..*
-calendario
-temporadas 0..*
-temporada
-cosechas 1..*
-objetivoActual
fertilizaciones
0..*
-fertilizanteAplicado

4. Diagramas de Secuencia (Vista de Análisis)

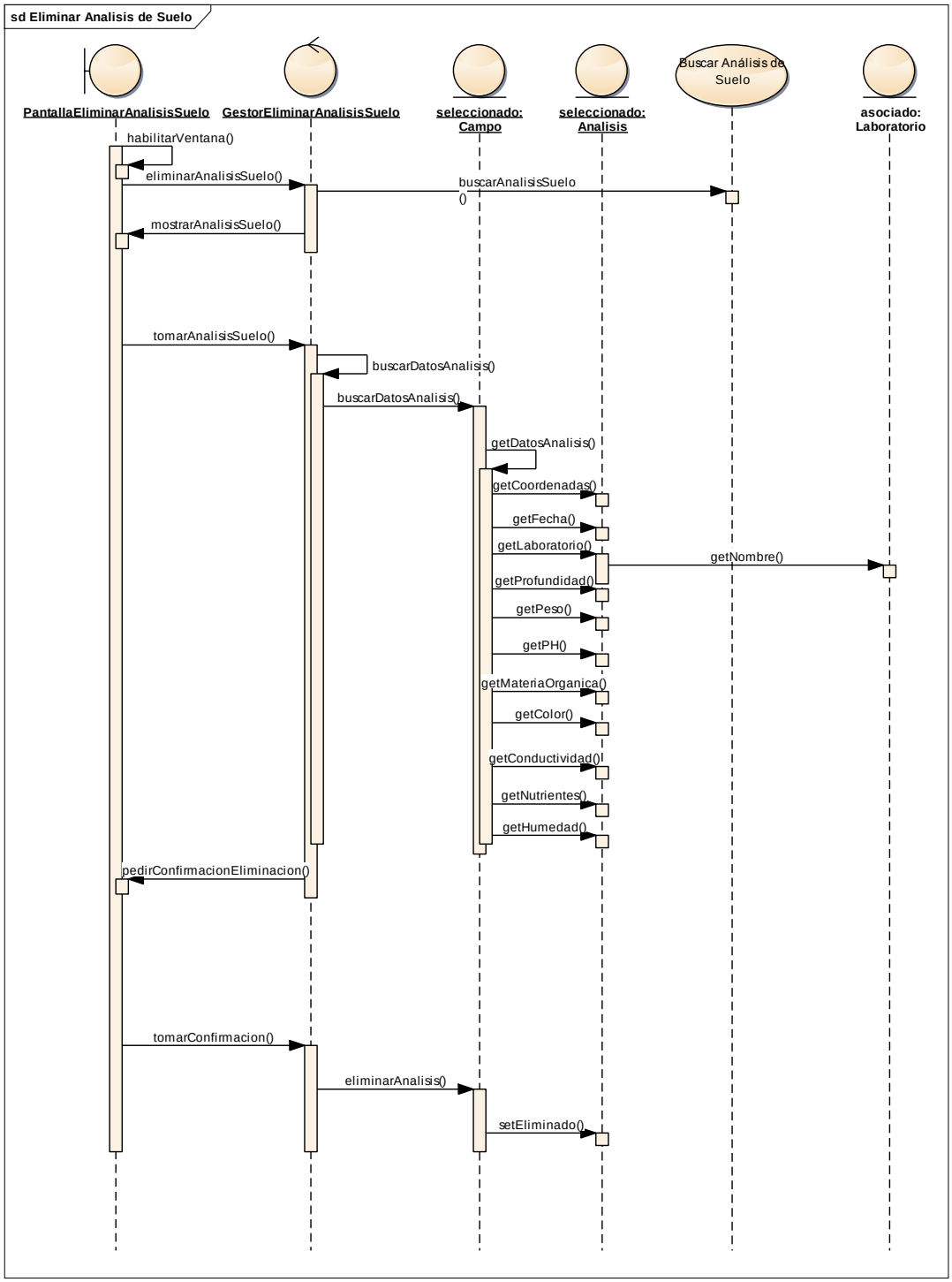
4.1. Registrar Análisis de Suelo


sd Registrar Analisis de Suelo
PantallaRegistrar
GestorRegistrarAnalisisSuelo
:Campo
seleccionado: Campo
Analisis
Laboratorio
habilitarVentana()
nuevoAnalisisSuelo()
mostrarCalendario()
getCampos()
mostrarCampos()
getLaboratorios()
mostrarNombre()
pedirIngresoDatos()
tomarCampo()
tomarCoordenadas()
controlarCoordenadas()
tomarSeleccionFecha()
controlarPeso()
tomarLaboratorio()
tomarProfundidad()
controlarProfundidad()
tomarPeso()
controlarPeso()
tomarPH()
controlarPH()
tomarMateriaOrganica()
controlarMateriaOrganica()
tomarConductividad()
controlarConductividad()
tomarNutrientes()
controlarNitrogeno()
tomarHumedad()
controlarHumedad()
tomarConfirmacionRegistro()
registrarAnalisisSuelo()
crearAnalisisSuelo()
new()

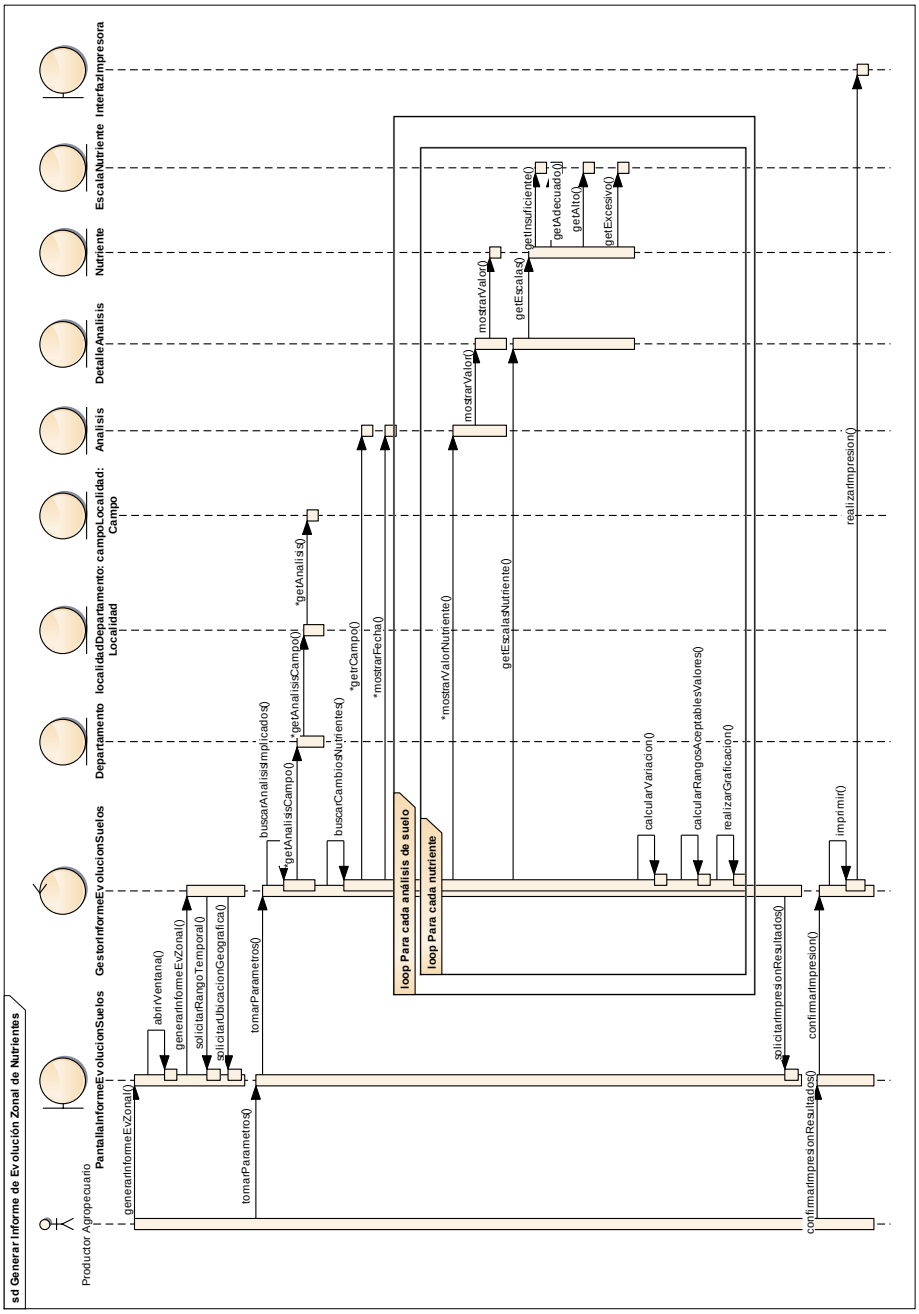
4.2. Modificar Análisis de Suelo



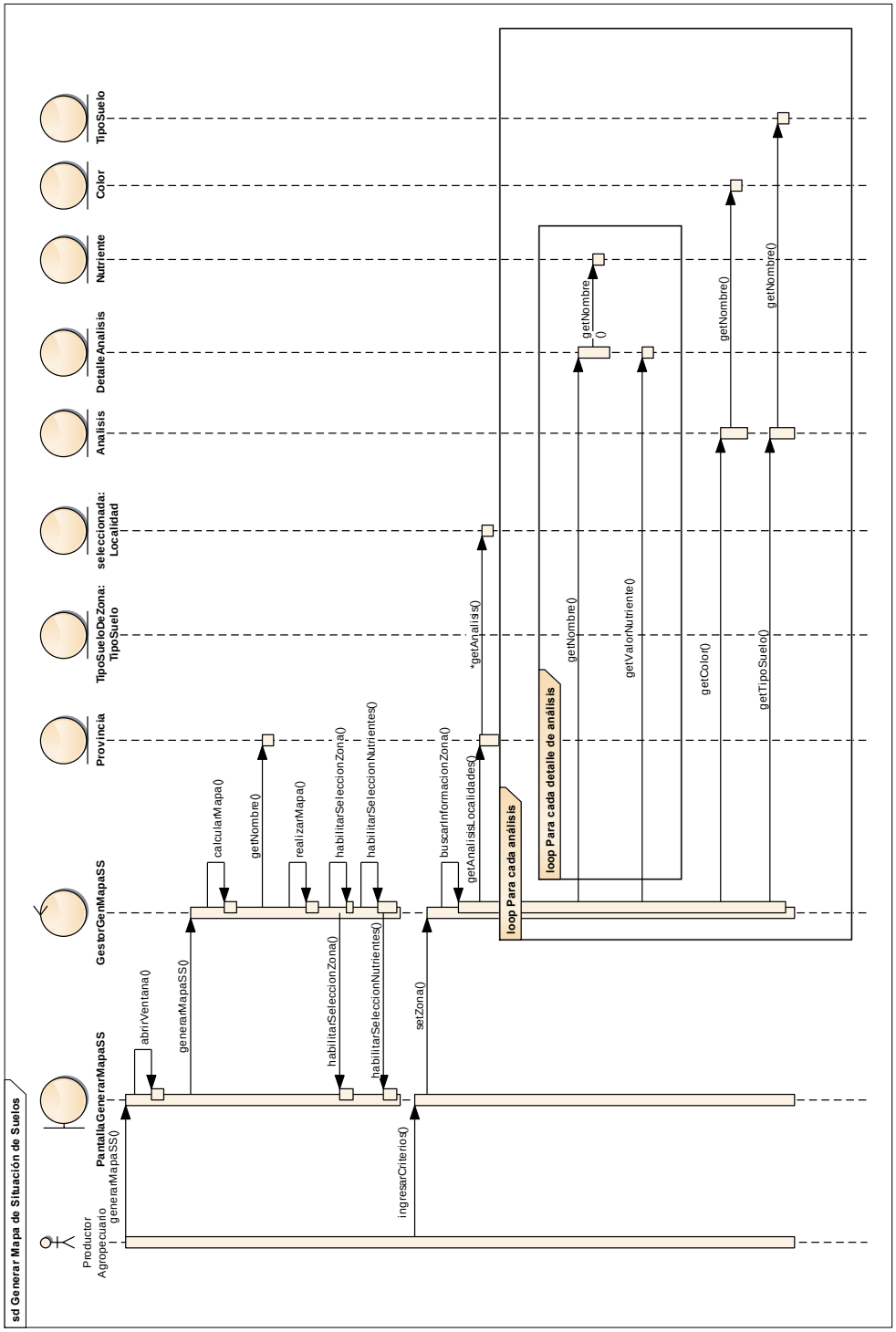
4.3. Eliminar Análisis de Suelo


sd Eliminar Analisis de Suelo
PantallaEliminarAnalisisSuelo
GestorEliminarAnalisisSuelo
seleccionado: Campo
seleccionado: Analisis
Buscar Análisis de Suelo
asociado: Laboratorio
habilitarVentana()
eliminarAnalisisSuelo()
buscarAnalisisSuelo()
mostrarAnalisisSuelo()
tomarAnalisisSuelo()
buscarDatosAnalisis()
buscarDatosAnalisis()
getDatosAnalisis()
getCoordenadas()
getFecha()
getLaboratorio()
getNombre()
getProfundidad()
getPeso()
getPH()
getMateriaOrganica()
getColor()
getConductividad()
getNutrientes()
getHumedad()
pedirConfirmacionEliminacion()
tomarConfirmacion()
eliminarAnalisis()
setEliminado()

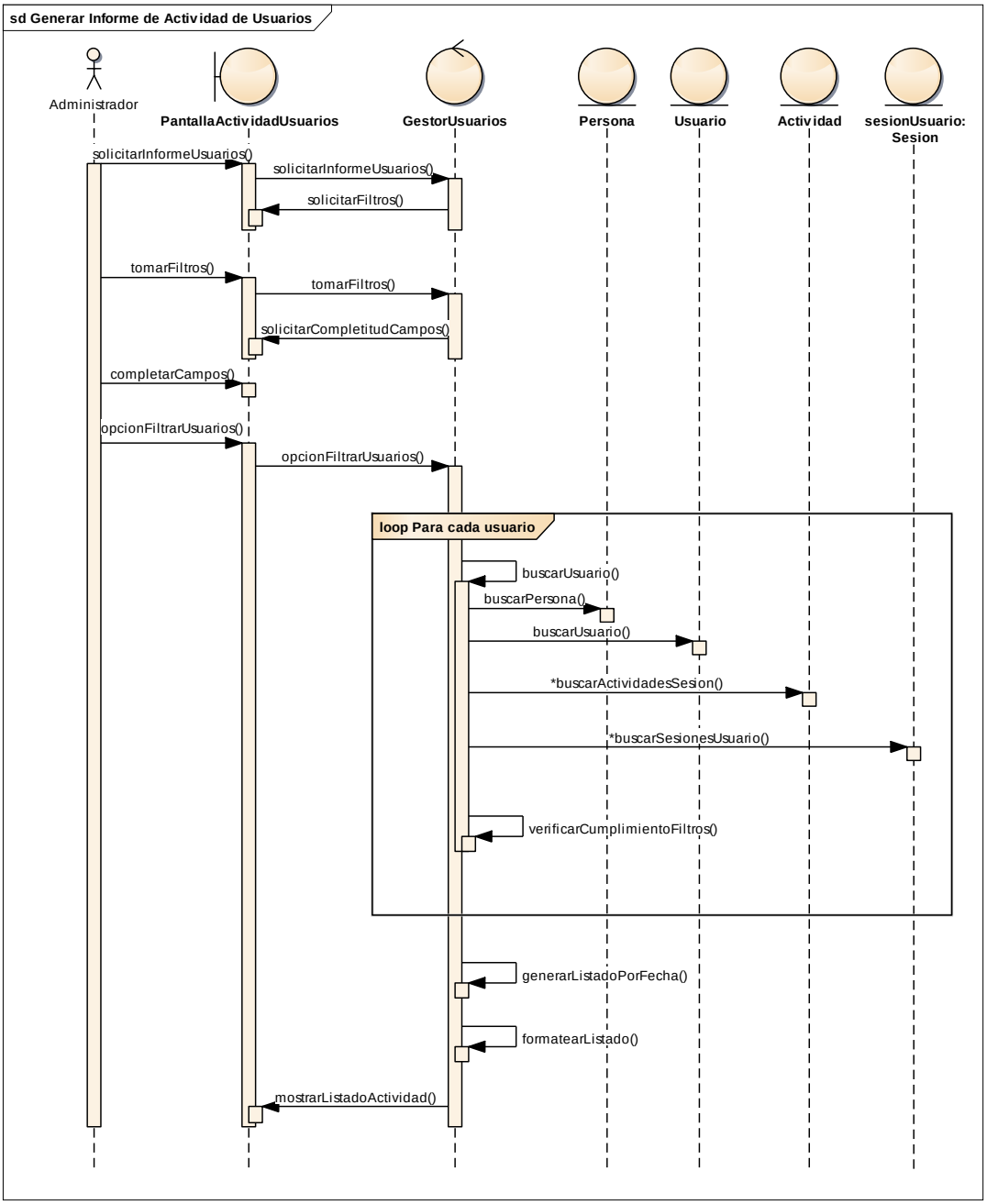
4.4. Generar Informe de Evolución Zonal de Nutrientes



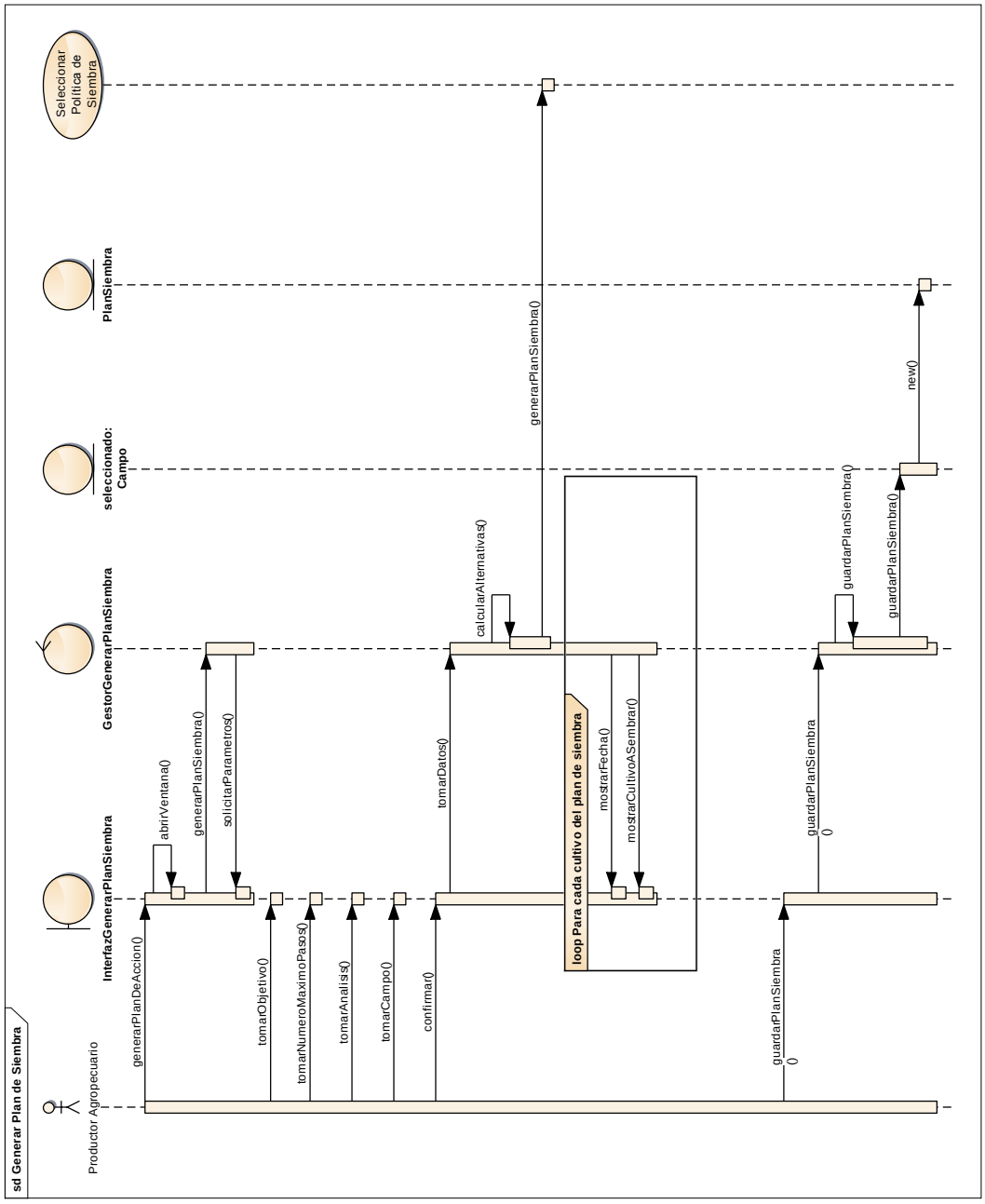
4.5. Generar Mapa de Situación de Suelos



4.6. Generar Informe de Actividad de Usuarios



4.7. Generar Plan de Siembra



Capítulo V – Modelo de Diseño

1. Introducción

El presente documento muestra las principales vistas del Modelo de Diseño del Sistema Solum.

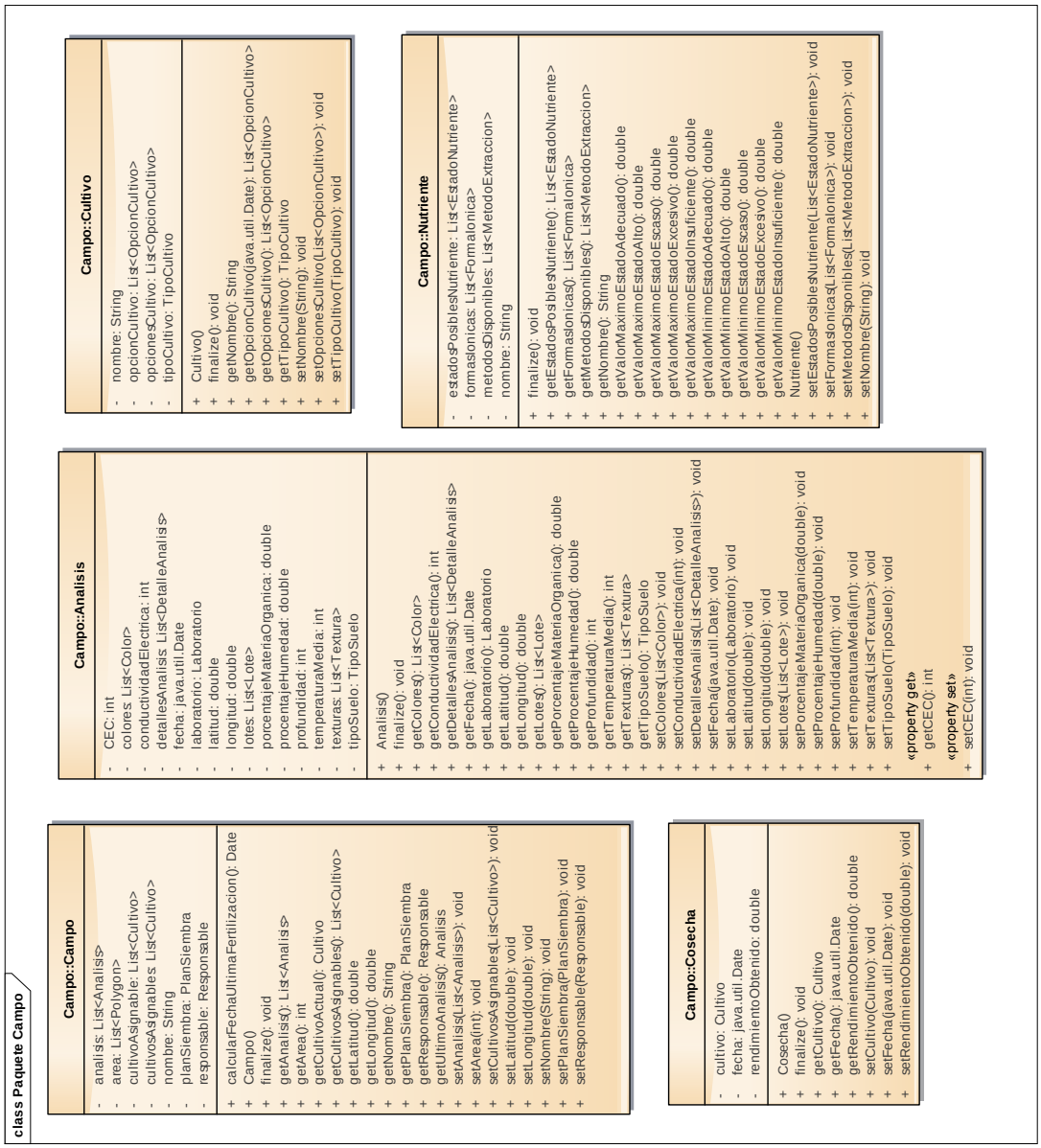
Es oportuno mencionar que en el presente workflow no se encuentra incluido el Diagrama de Entidad Relación (DER), debido a que el mismo es reemplazado por Mapeos Objeto-Relacional (ORM) generados automáticamente por el framework de aplicaciones web Grails.

2. Clases de Diseño

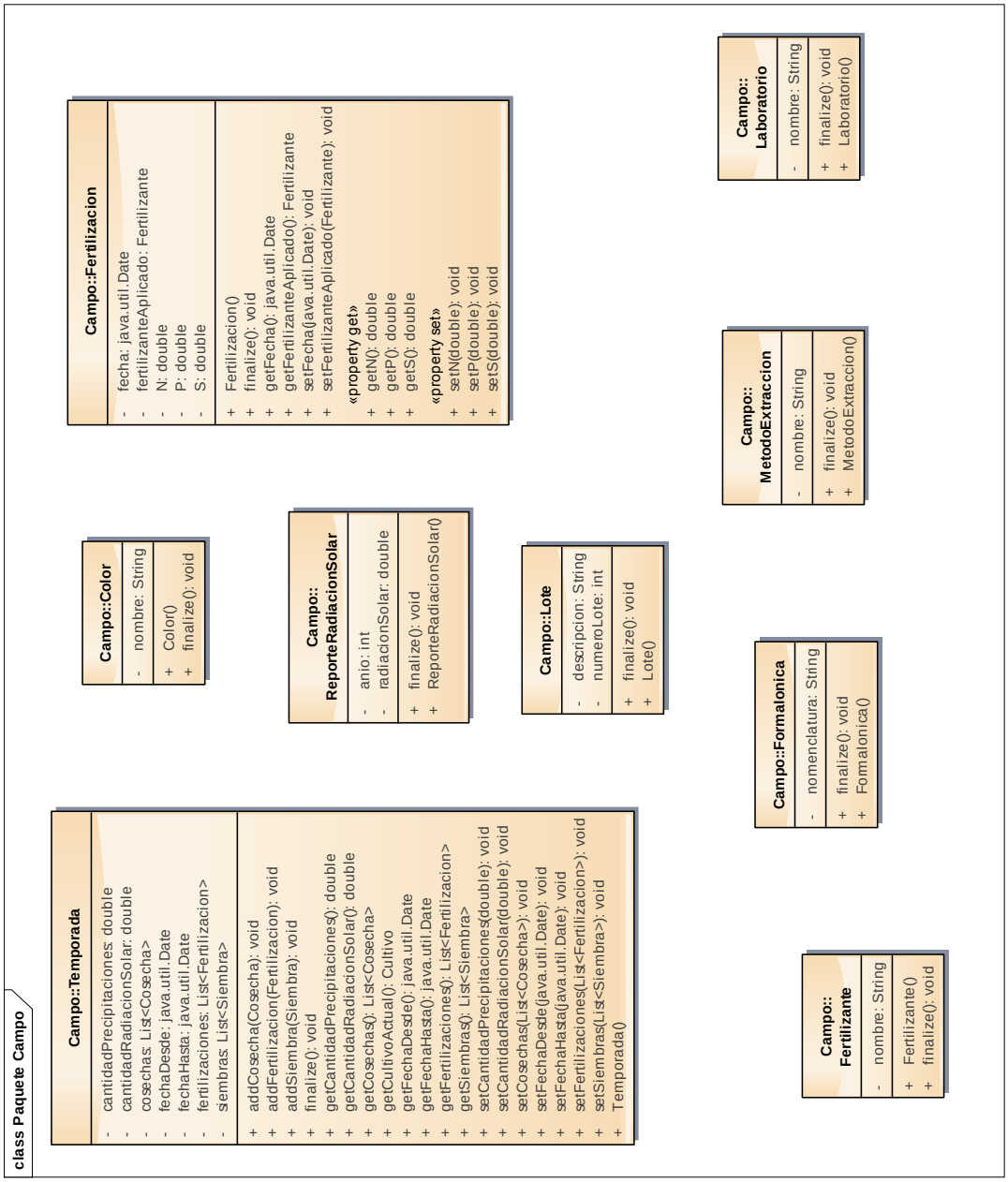
2.1. Introducción

En los presentes diagramas se muestra cada una de las clases de diseño que componen los respectivos paquetes.

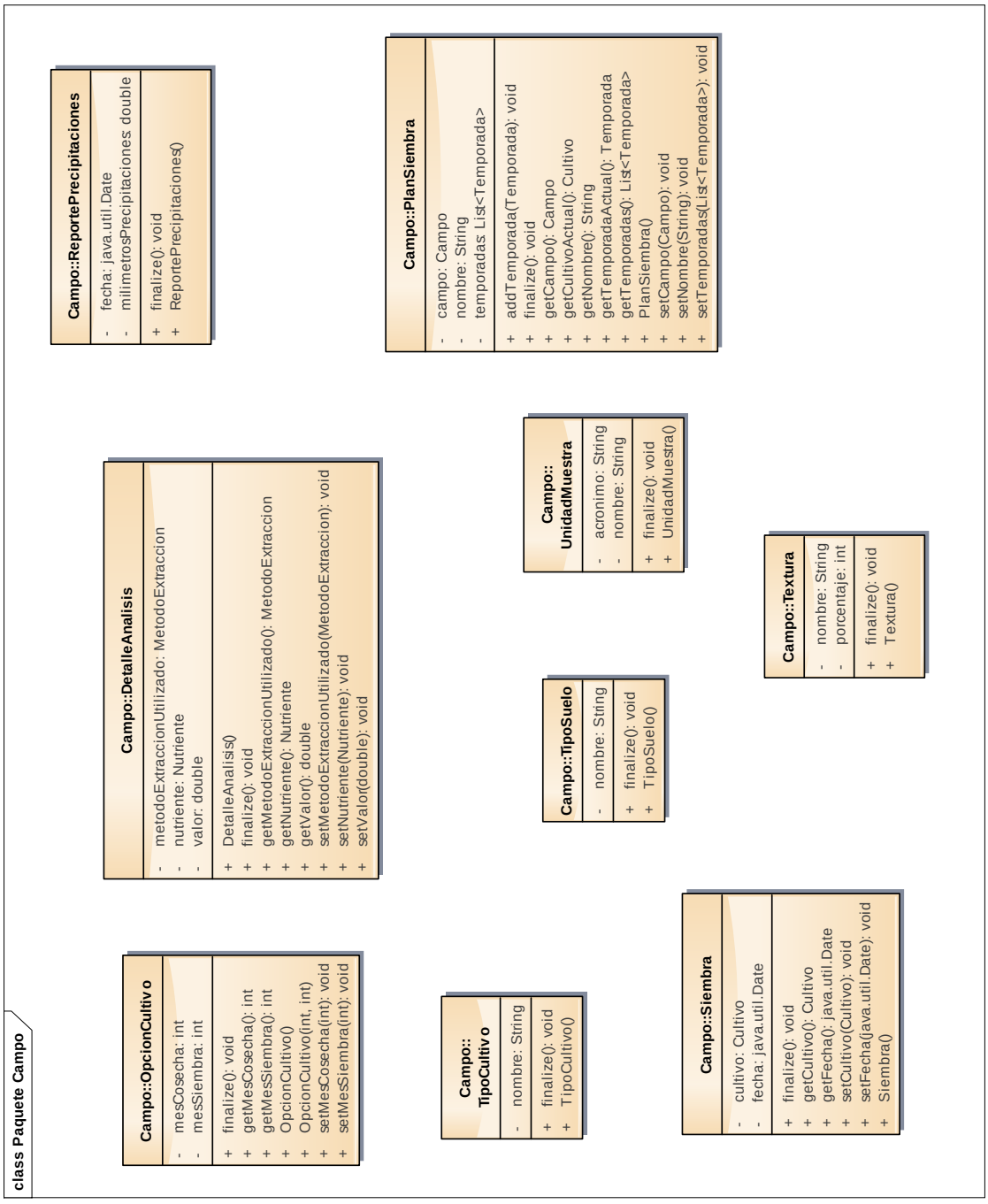
2.1. Paquete Campo (1)



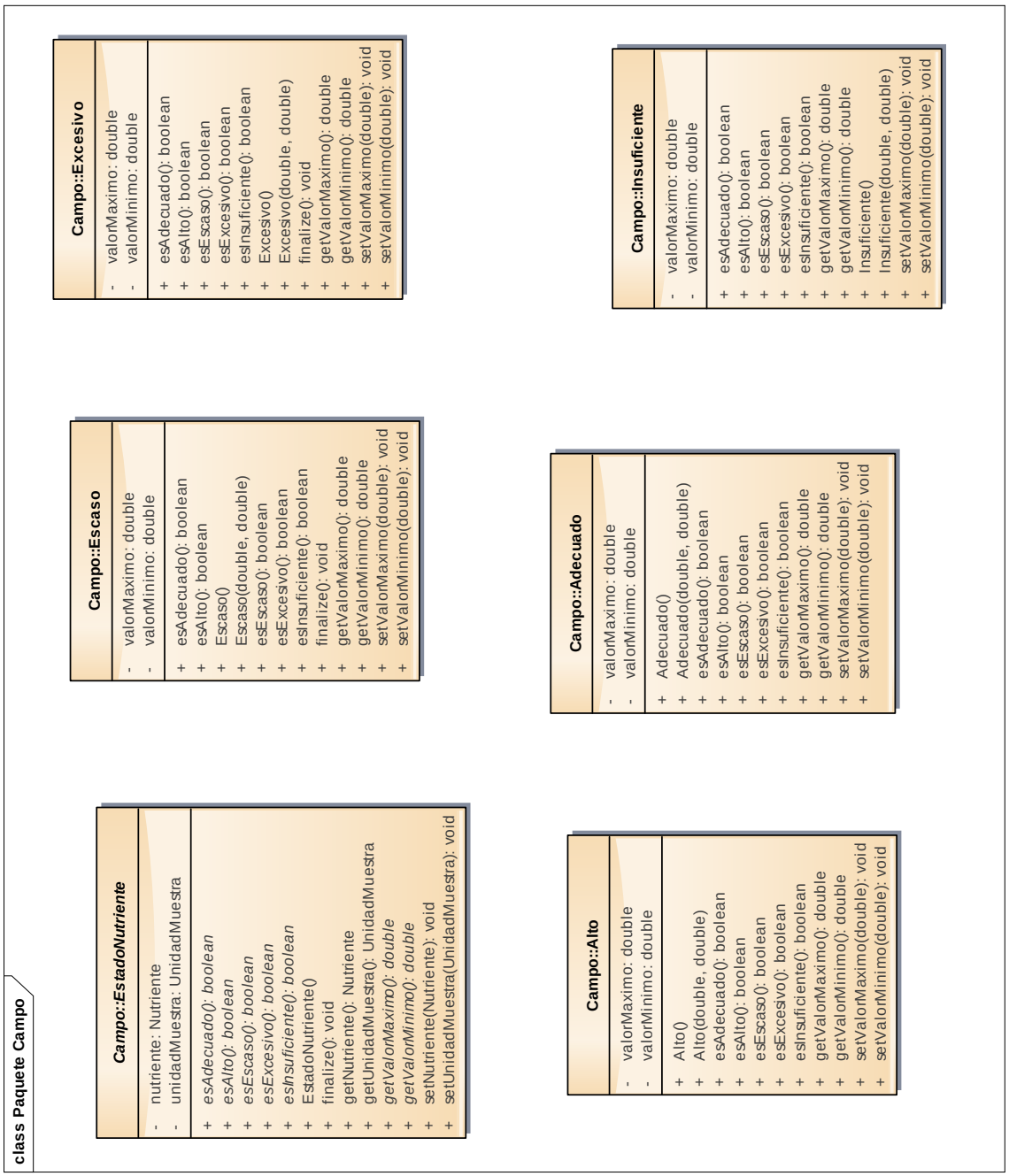
2.2. Paquete Campo (2)



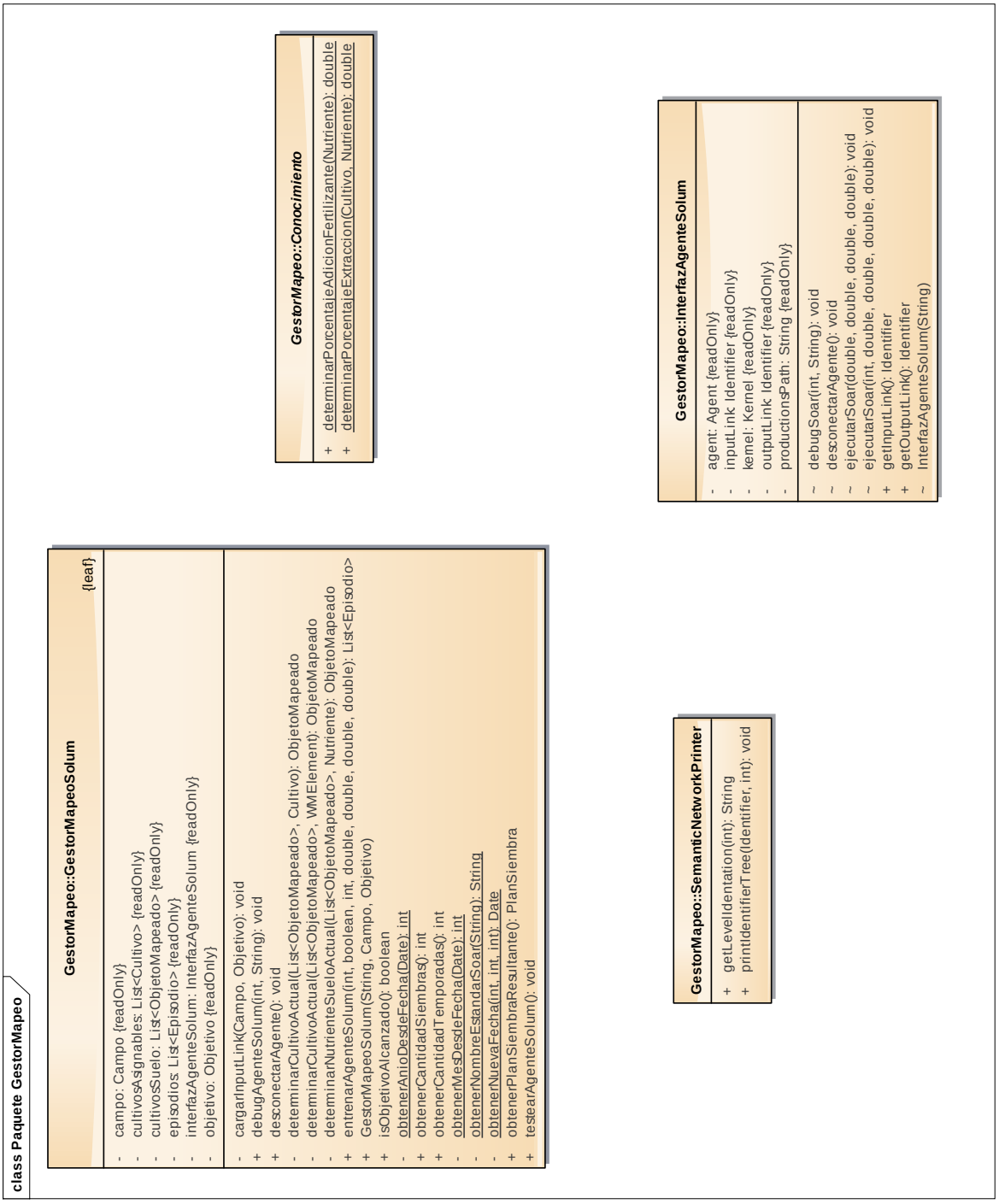
2.3. Paquete Campo (3)



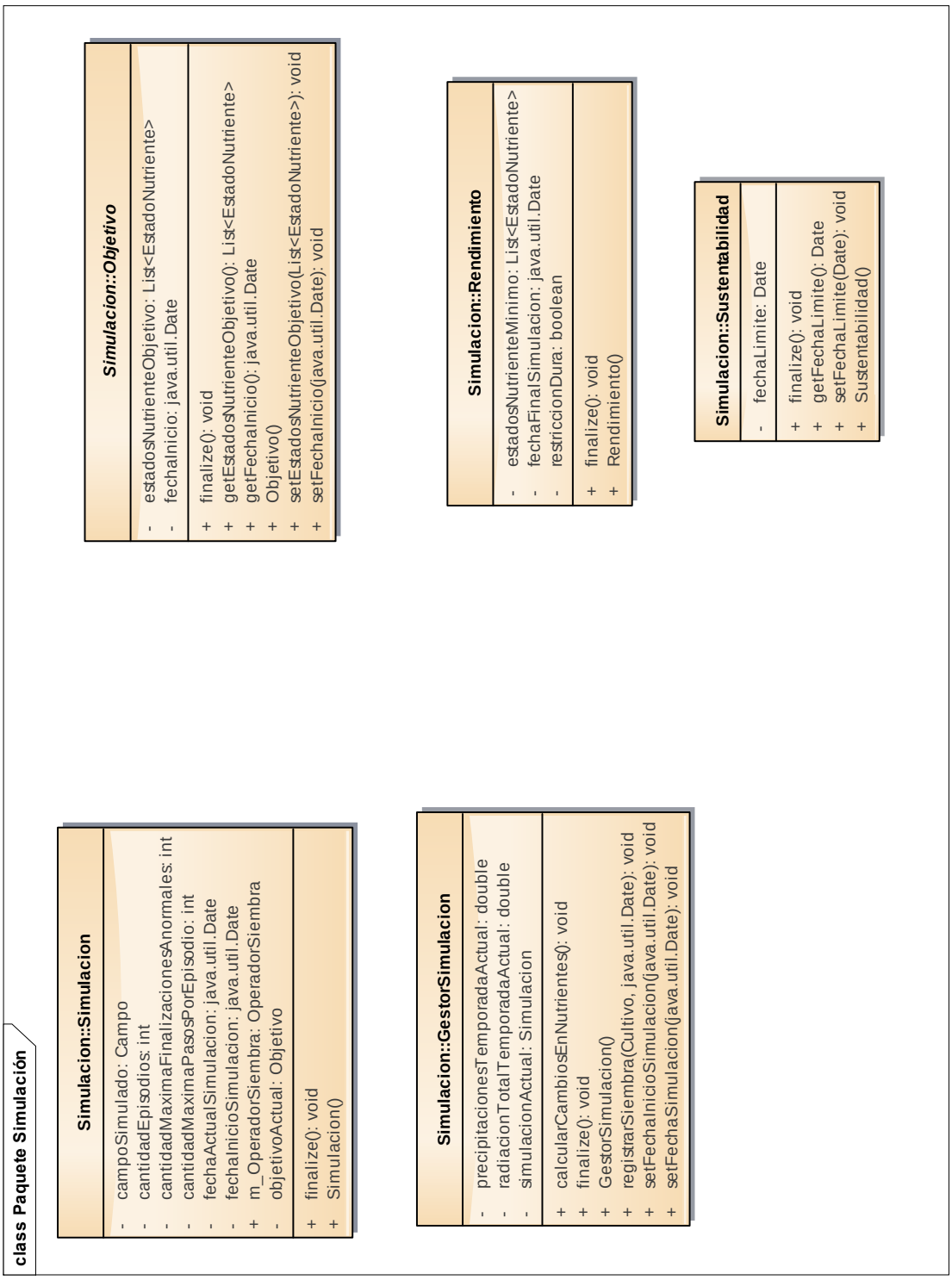
2.4. Paquete Campo (4)



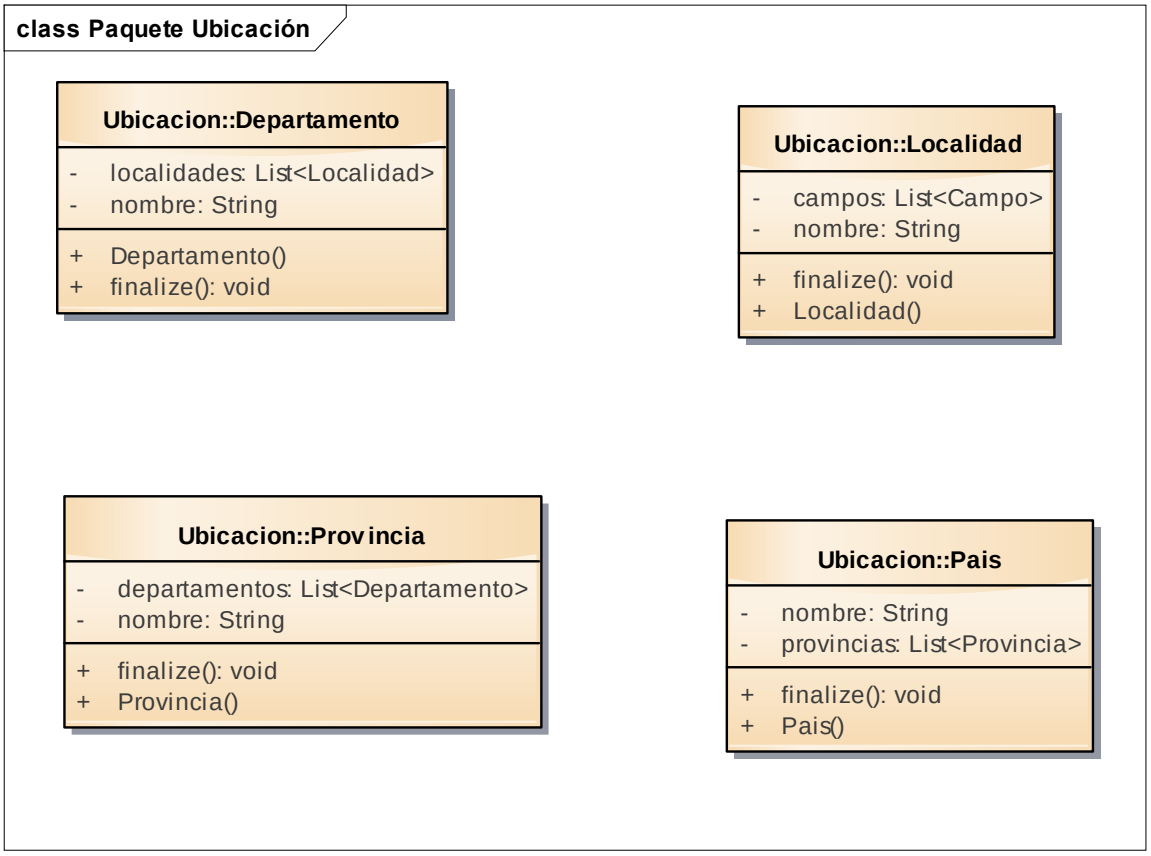
2.5. Paquete GestorMapeo



2.6. Paquete Simulación


class Paquete Simulación
Simulacion::Simulacion
- campoSimulado: Campo
- cantidadEpisodios: int
- cantidadMaximaFinalizacionesAnormales: int
- cantidadMaximaPasosPorEpisodio: int
- fechaActualSimulacion: java.util.Date
- fechaInicioSimulacion: java.util.Date
+ m_OperadorSiembra: OperadorSiembra
- objetivoActual: Objetivo
+ finalize(): void
+ Simulacion()
Simulacion::Objetivo
- estadosNutrienteObjetivo: List<EstadoNutriente>
- fechaInicio: java.util.Date
+ finalize(): void
+ getEstadosNutrienteObjetivo(): List<EstadoNutriente>
+ getFechaInicio(): java.util.Date
+ Objetivo()
+ setEstadosNutrienteObjetivo(List<EstadoNutriente>): void
+ setFechaInicio(java.util.Date): void
Simulacion::GestorSimulacion
- precipitacionesTemporadaActual: double
- radiacionTotalTemporadaActual: double
- simulacionActual: Simulacion
+ calcularCambiosEnNutrientes(): void
+ finalize(): void
+ GestorSimulacion()
+ registrarSiembra(Cultivo, java.util.Date): void
+ setFechaInicioSimulacion(java.util.Date): void
+ setFechaSimulacion(java.util.Date): void
Simulacion::Rendimiento
- estadosNutrienteMinimo: List<EstadoNutriente>
- fechaFinalSimulacion: java.util.Date
- restriccionDura: boolean
+ finalize(): void
+ Rendimiento()
Simulacion::Sustentabilidad
- fechaLimite: Date
+ finalize(): void
+ getFechaLimite(): Date
+ setFechaLimite(Date): void
+ Sustentabilidad()

2.7. Paquete Ubicación



3. Diagrama de Clases de Diseño Planificación

3.1. Introducción

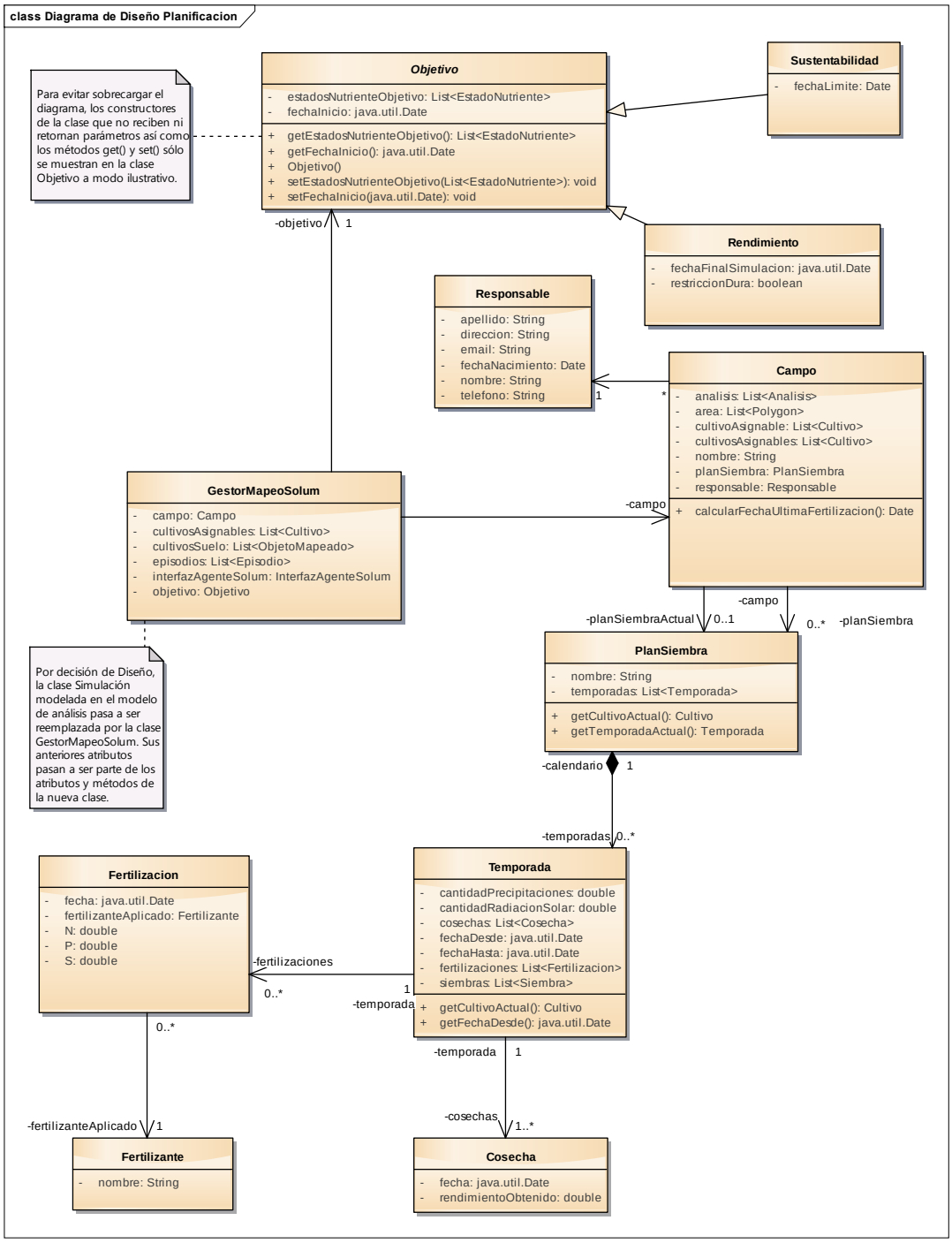
A continuación se muestra el principal diagrama de clases de diseño de Solum. Tal diagrama se distingue principalmente de su par de análisis en que el primero incluye un mayor nivel de detalle puesto que se lo asocia directamente con la tecnología en la que se implementa (Java), mientras que el segundo tiene como objetivo mostrar las entidades del dominio a un nivel mayormente conceptual. En particular, los siguientes son algunos de los elementos que se incluyen en los presentes Diagramas de Clases de Diseño:

1. Se incluyen los sentidos de las asociaciones, en pos de enfatizar la navegabilidad del diagrama.
2. Se incluyen los tipos de datos que utilizan cada uno de los atributos.
3. Se incluyen los métodos de cada clase junto a su signatura.

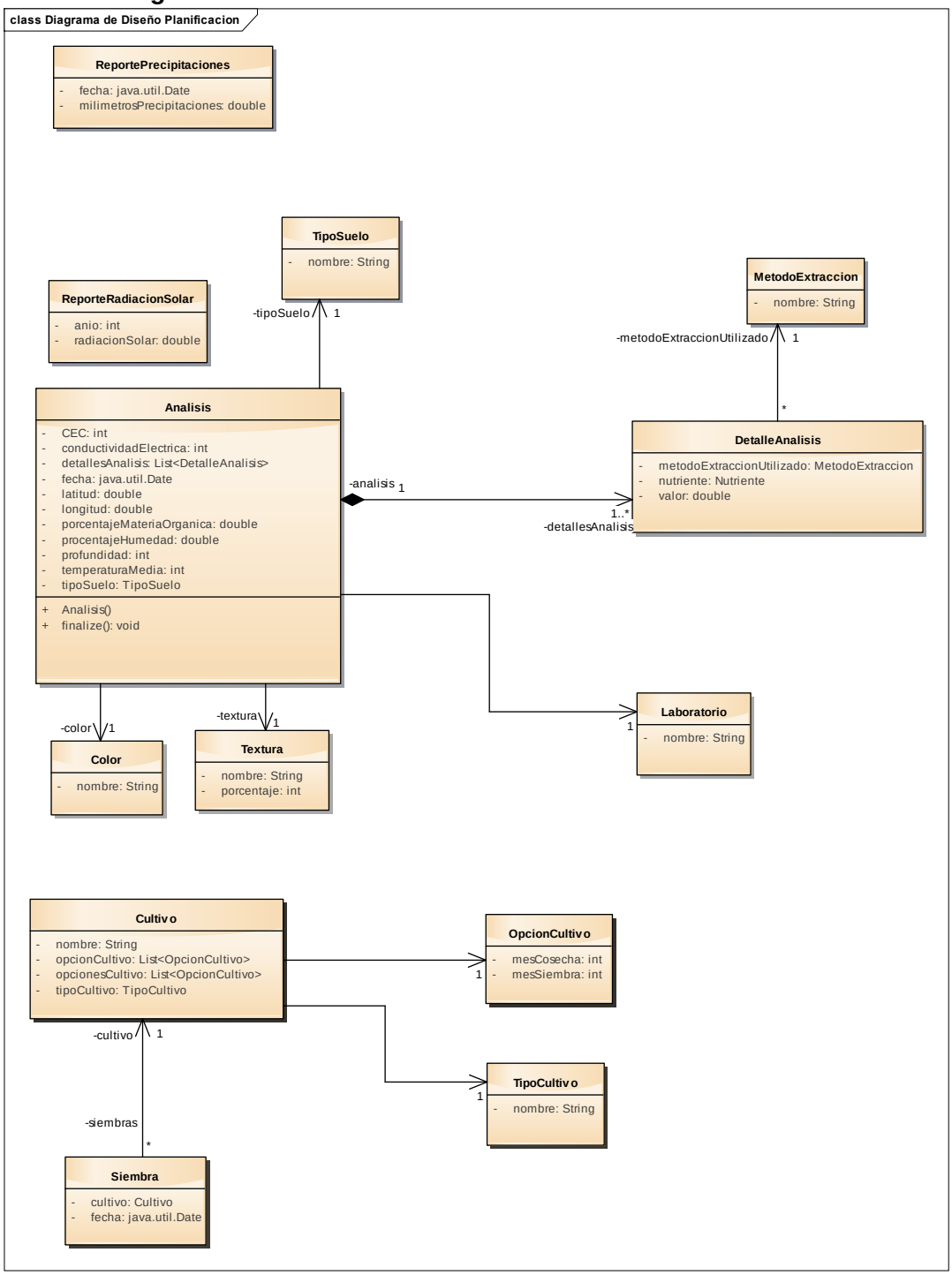
La planificación se distingue por dos objetivos, Sustentabilidad y Rendimiento. El objetivo de Sustentabilidad consiste en alcanzar un cierto valor en los nutrientes del suelo. La simulación finaliza al poder alcanzar dichos valores.

Por su parte, Rendimiento busca maximizar el rendimiento en las cosechas en un período de tiempo determinado, teniendo como restricciones las escalas mínimas en los valores de los nutrientes. Considerar que dicha restricción pueda ser blanda (es decir que el sistema no anule simulaciones que arrojen valores menores a las escalas mínimas, pero que les aplique un castigo a los mismos), o bien que dichas restricciones sean duras, en donde el sistema directamente anule aquellas simulaciones que no cumplan con las restricciones.

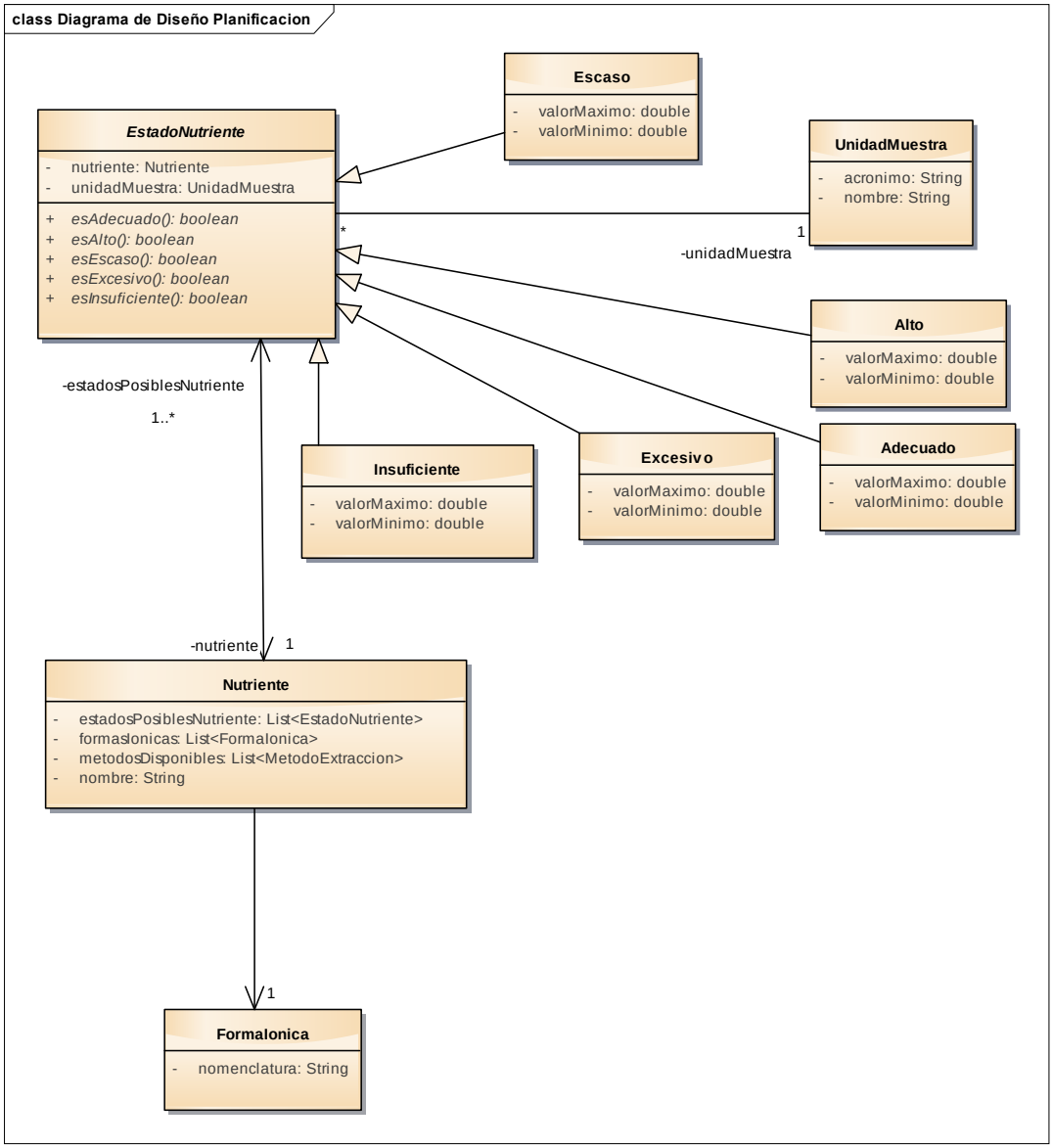
3.2. Primera Parte



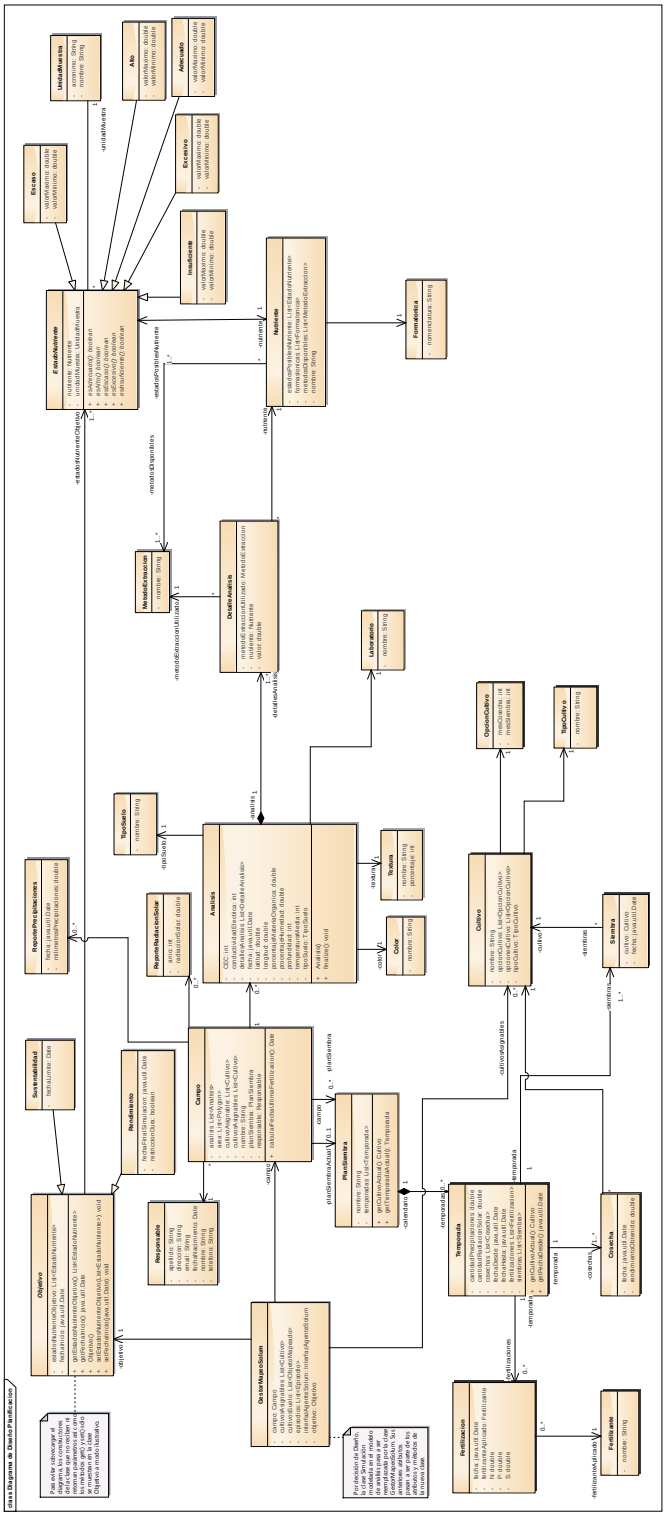
3.3. Segunda Parte



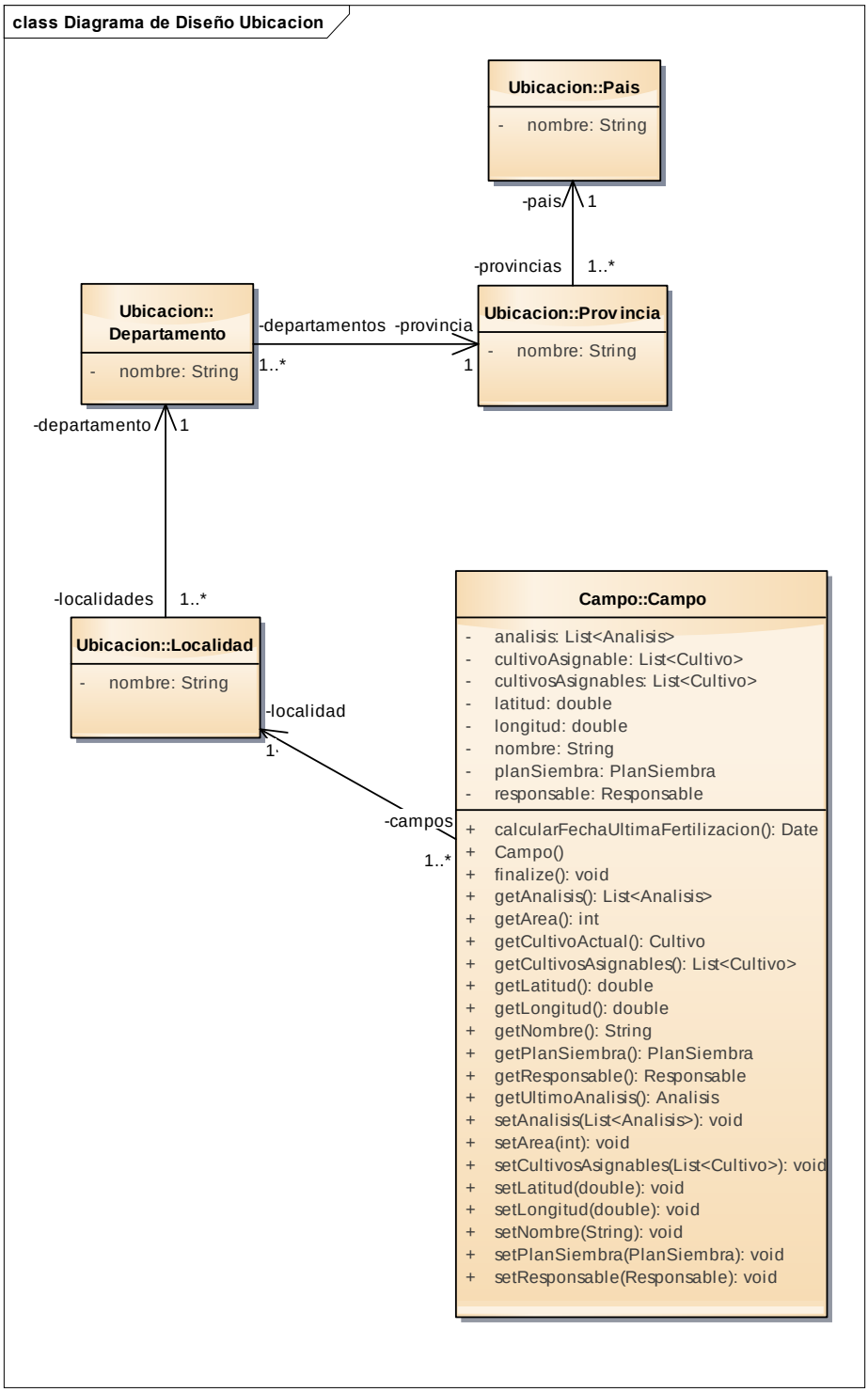
3.4. Tercera Parte



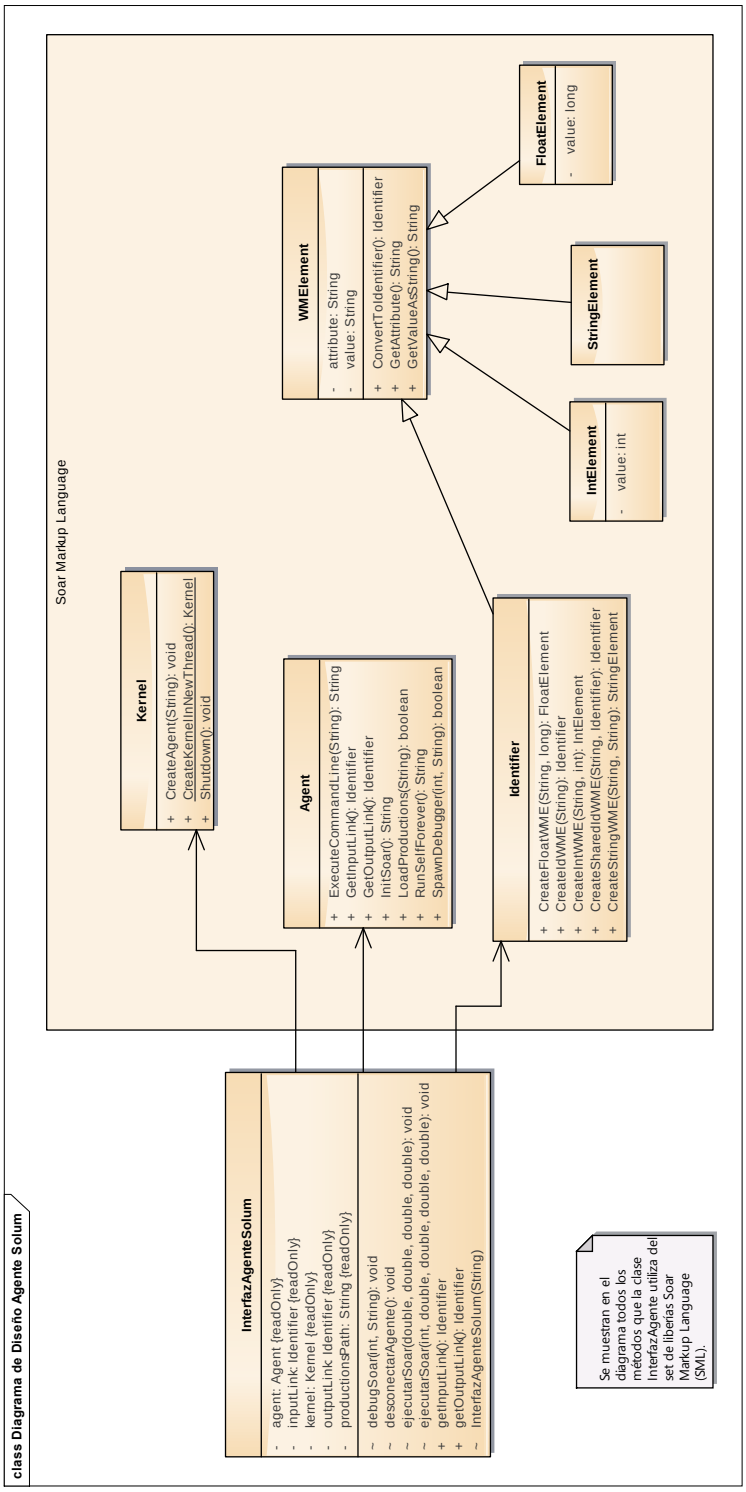
3.5. Vista Completa



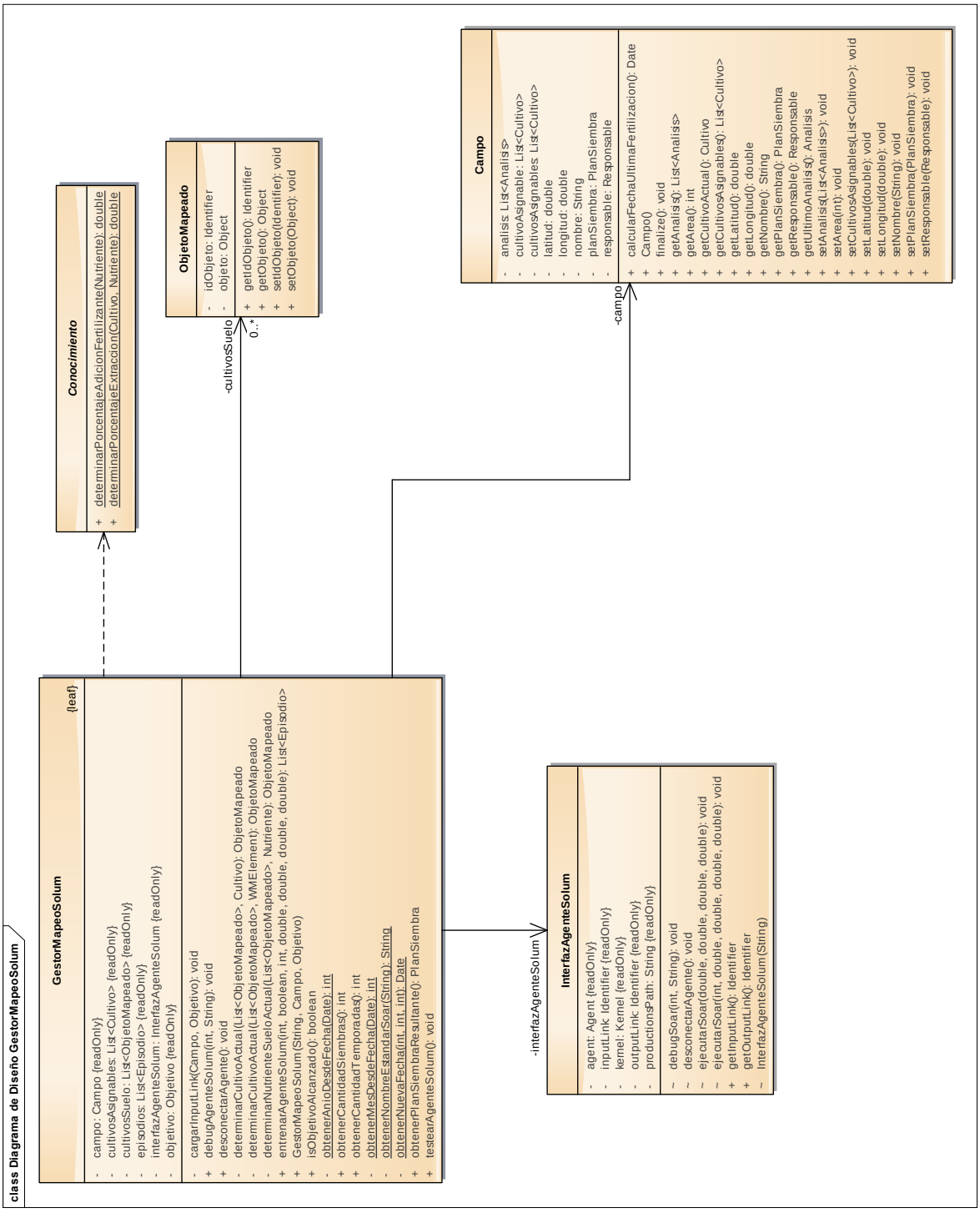
4. Diagrama de Clases de Diseño Ubicación


class Diagrama de Diseño Ubicacion
Ubicacion::Pais
- nombre: String
-pais
1
-provincias
1..*
Ubicacion::
Departamento
- nombre: String
-departamentos
1..*
-provincia
1
Ubicacion::Provincia
- nombre: String
-departamento
1
-localidades
1..*
Ubicacion::Localidad
- nombre: String
-localidad
1
-campos
1..*
Campo::Campo
- analisis: List<Analisis>
- cultivoAsignable: List<Cultivo>
- cultivosAsignables: List<Cultivo>
- latitud: double
- longitud: double
- nombre: String
- planSiembra: PlanSiembra
- responsable: Responsable
+ calcularFechaUltimaFertilizacion(): Date
+ Campo()
+ finalize(): void
+ getAnalisis(): List<Analisis>
+ getArea(): int
+ getCultivoActual(): Cultivo
+ getCultivosAsignables(): List<Cultivo>
+ getLatitud(): double
+ getLongitud(): double
+ getNombre(): String
+ getPlanSiembra(): PlanSiembra
+ getResponsable(): Responsable
+ getUltimoAnalisis(): Analisis
+ setAnalisis(List<Analisis>): void
+ setArea(int): void
+ setCultivosAsignables(List<Cultivo>): void
+ setLatitud(double): void
+ setLongitud(double): void
+ setNombre(String): void
+ setPlanSiembra(PlanSiembra): void
+ setResponsable(Responsable): void

4.1. Diagrama de Diseño AgenteSolum


class Diagrama de Diseño Agente Solum
Soar Markup Language
Kernel
+ CreateAgent(String): void
+ CreateKernelInNewThread(): Kernel
+ Shutdown(): void
Agent
+ ExecuteCommandLine(String): String
+ GetInputLink(): Identifier
+ GetOutputLink(): Identifier
+ InitSoar(): String
+ LoadProductions(String): boolean
+ RunSelfForever(): String
+ SpawnDebugger(int, String): boolean
Identifier
+ CreateFloatWME(String, long): FloatElement
+ CreateIdWME(String): Identifier
+ CreateIntWME(String, int): IntElement
+ CreateSharedIdWME(String, Identifier): Identifier
+ CreateStringWME(String, String): StringElement
WMElement
- attribute: String
- value: String
+ ConvertToIdentifier(): Identifer
+ GetAttribute(): String
+ GetValueAsString(): String
FloatElement
- value: long
StringElement
IntElement
- value: int
InterfazAgenteSolum
- agent: Agent {readOnly}
- inputLink: Identifier {readOnly}
- kernel: Kernel {readOnly}
- outputLink: Identifier {readOnly}
- productionsPath: String {readOnly}
~ debugSoar(int, String): void
~ desconectarAgente(): void
~ ejecutarSoar(double, double, double, double): void
~ ejecutarSoar(int, double, double, double, double): void
+ getInputLink(): Identifier
+ getOutputLink(): Identifier
~ InterfazAgenteSolum(String)
Se muestran en el diagrama todos los métodos que la clase InterfazAgente utiliza del set de liberias Soar Markup Language (SML).

4.2. Diagrama de Diseño GestorMapeoSolum



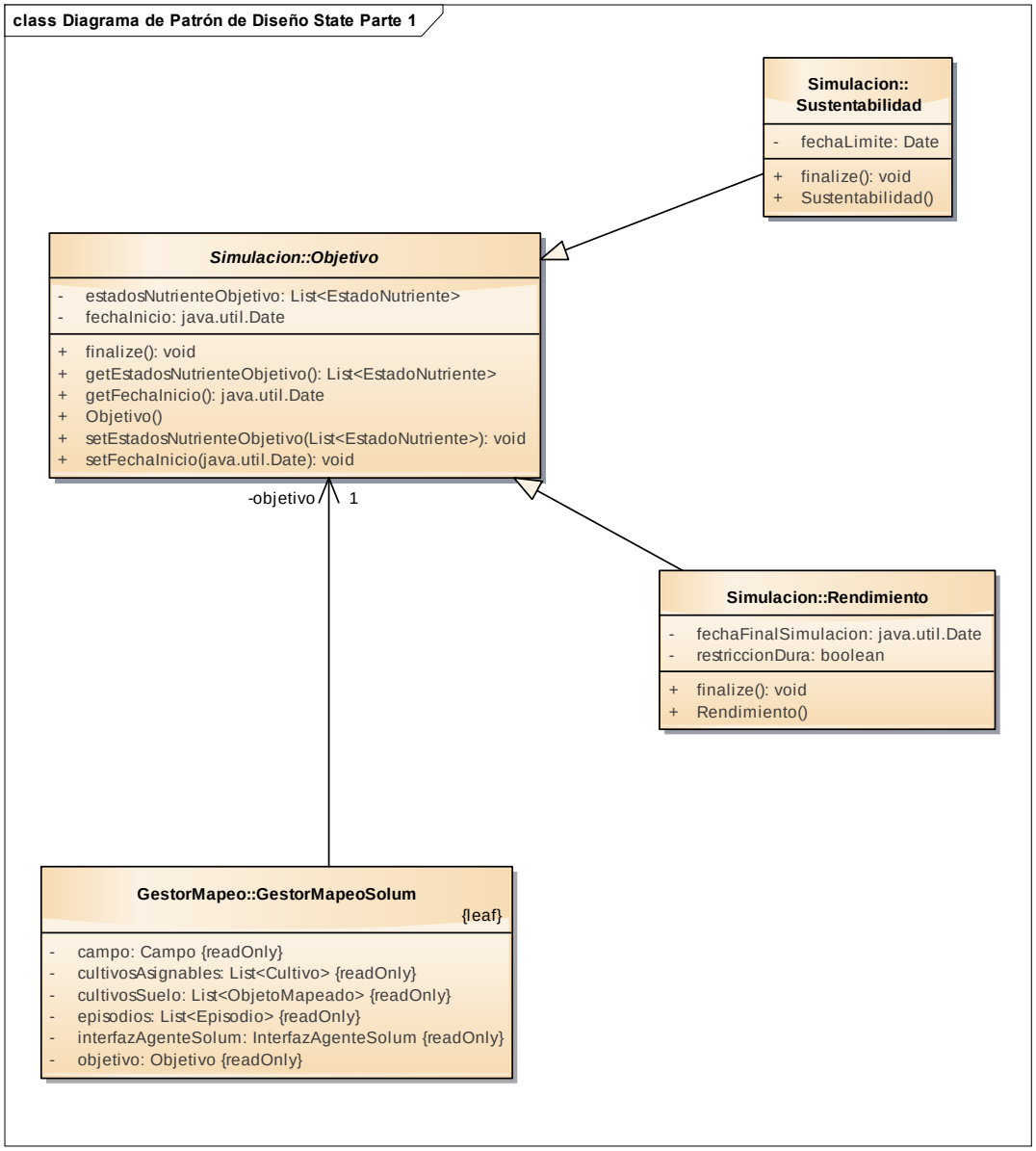
5.1. Patrones GoF Utilizados

Introducción

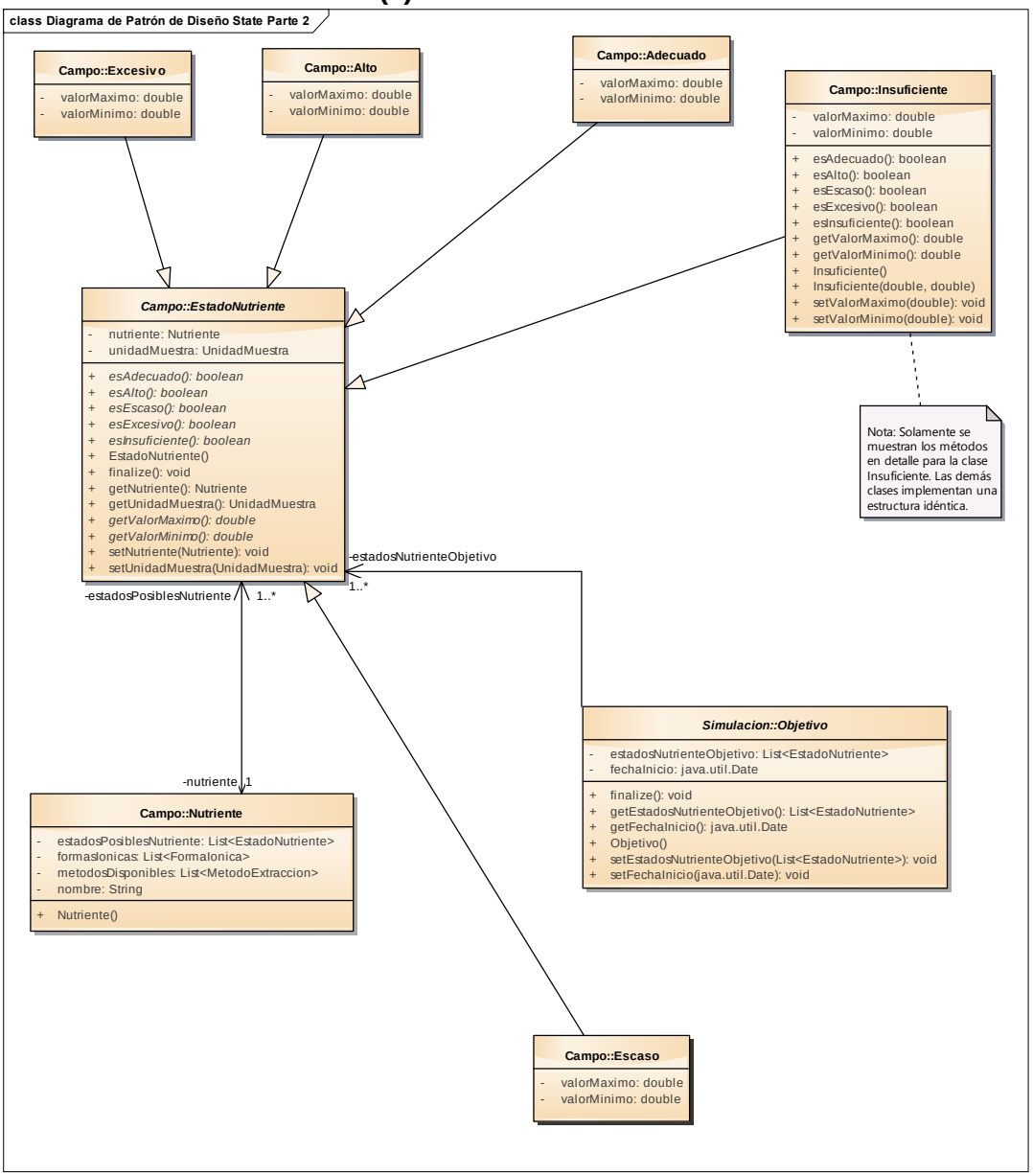
Se muestran a continuación los dos Patrones de Diseño del *Gang of Four* (Gamma, Helm, Johnson y Vlissides, 1995) que se utilizaron en el diseño del sistema. Específicamente, ambos patrones utilizados fueron los patrones de comportamiento State. En el primer caso, se trata de un patrón ubicando en torno a la clase abstracta Objetivo, que asume comportamiento según el objetivo correspondiente. Tal comportamiento cambia internamente la ejecución que se realiza en el agente Solum. De esta manera, la simulación tendrá un comportamiento bien marcado, conservador cuando se trata del objetivo Sustentabilidad o agresivo cuando se trata del objetivo Rendimiento.

Por su parte, en el segundo caso puede observarse que el patrón State se ubica en torno a la clase EstadoNutriente, en donde un Objetivo o un Nutriente tienen una relación de asociación con objetos de tipo EstadoNutriente sin conocer directamente de qué estado concretamente se trata. Esto resulta particularmente útil para definir el comportamiento que los mismos tendrán al realizar el agente Solum la simulación, puesto que sus reglas de inferencia se clasifican según los estados nutrientes de los que se trate.

Patrón GoF de Diseño State (1)

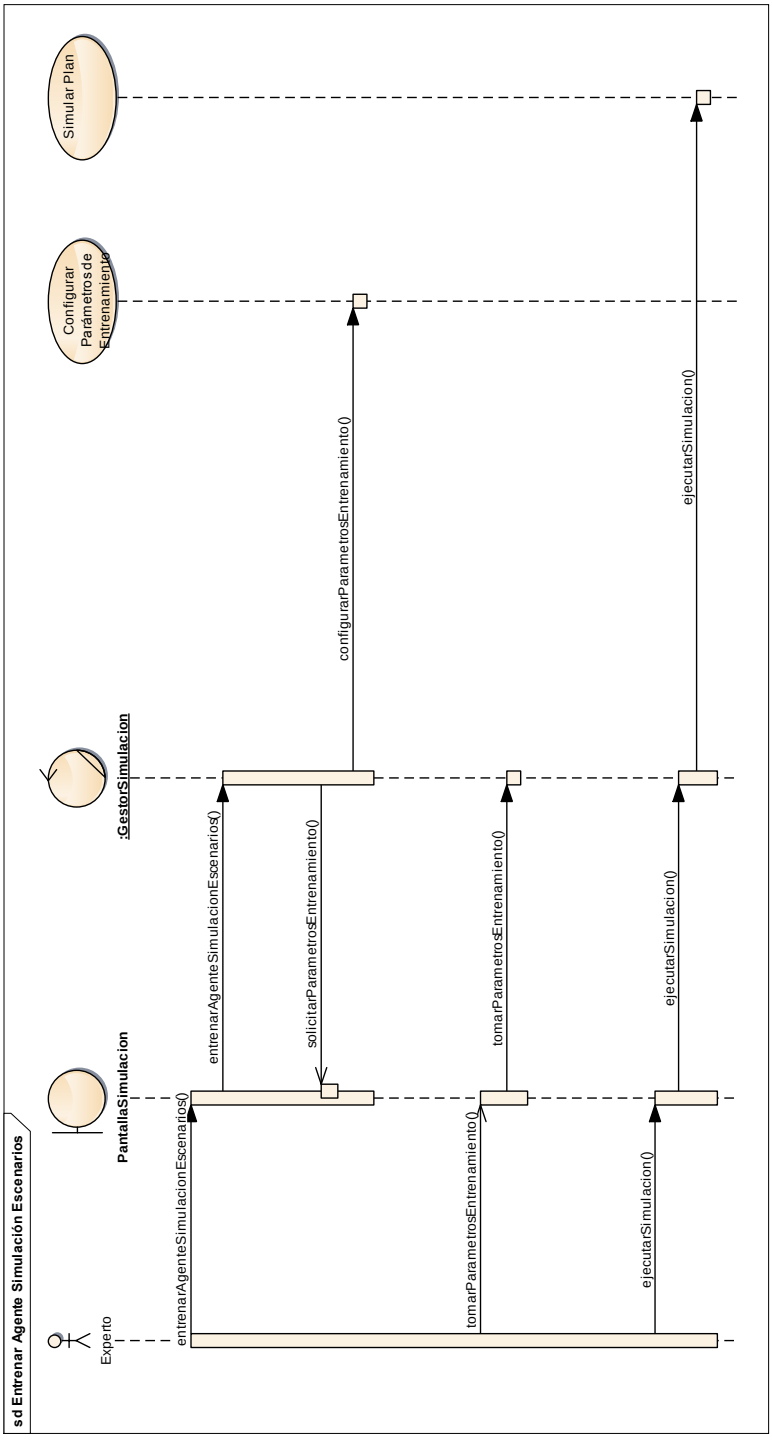


Patrón GoF de Diseño State (2)

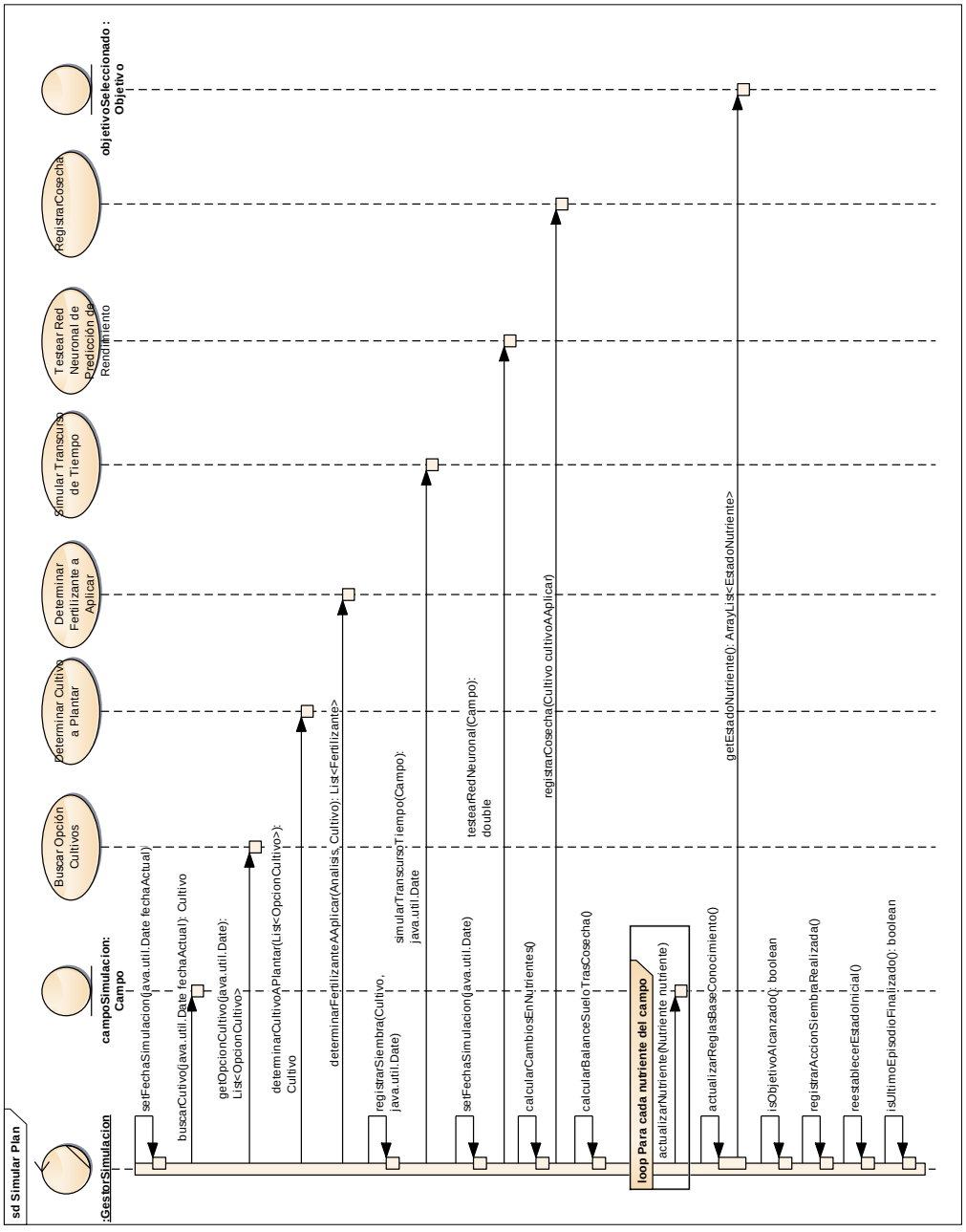


6. Diagramas de Secuencia (Vista de Diseño)

6.1. Entrenar Agente de Simulación de Escenarios


sd Entrenar Agente Simulación Escenarios
Experto
PantallaSimulacion
:GestorSimulacion
Configurar Parámetros de Entrenamiento
Simular Plan
entrenarAgenteSimulacionEscenarios()
entrenarAgenteSimulacionEscenarios()
solicitarParametrosEntrenamiento()
configurarParametrosEntrenamiento()
tomarParametrosEntrenamiento()
tomarParametrosEntrenamiento()
ejecutarSimulacion()
ejecutarSimulacion()
ejecutarSimulacion()

6.2. Simular Plan



7. Modelado Formal

7.1. Introducción

En la presente sección se muestran aquellos diagramas que muestran el modelado interno del Agente Solum. Los mismos reflejan cómo el agente almacena internamente la información que utilizará para realizar el aprendizaje sobre el entorno para poder llevar a cabo la generación de los planes de siembra. Para comprender con mejor detalle los fundamentos que explican cómo se almacena internamente el conocimiento del agente, sentando las bases sobre la necesidad de tales diagramas, puede consultarse la Documentación de Inteligencia Artificial y la Documentación del Dominio Campos.

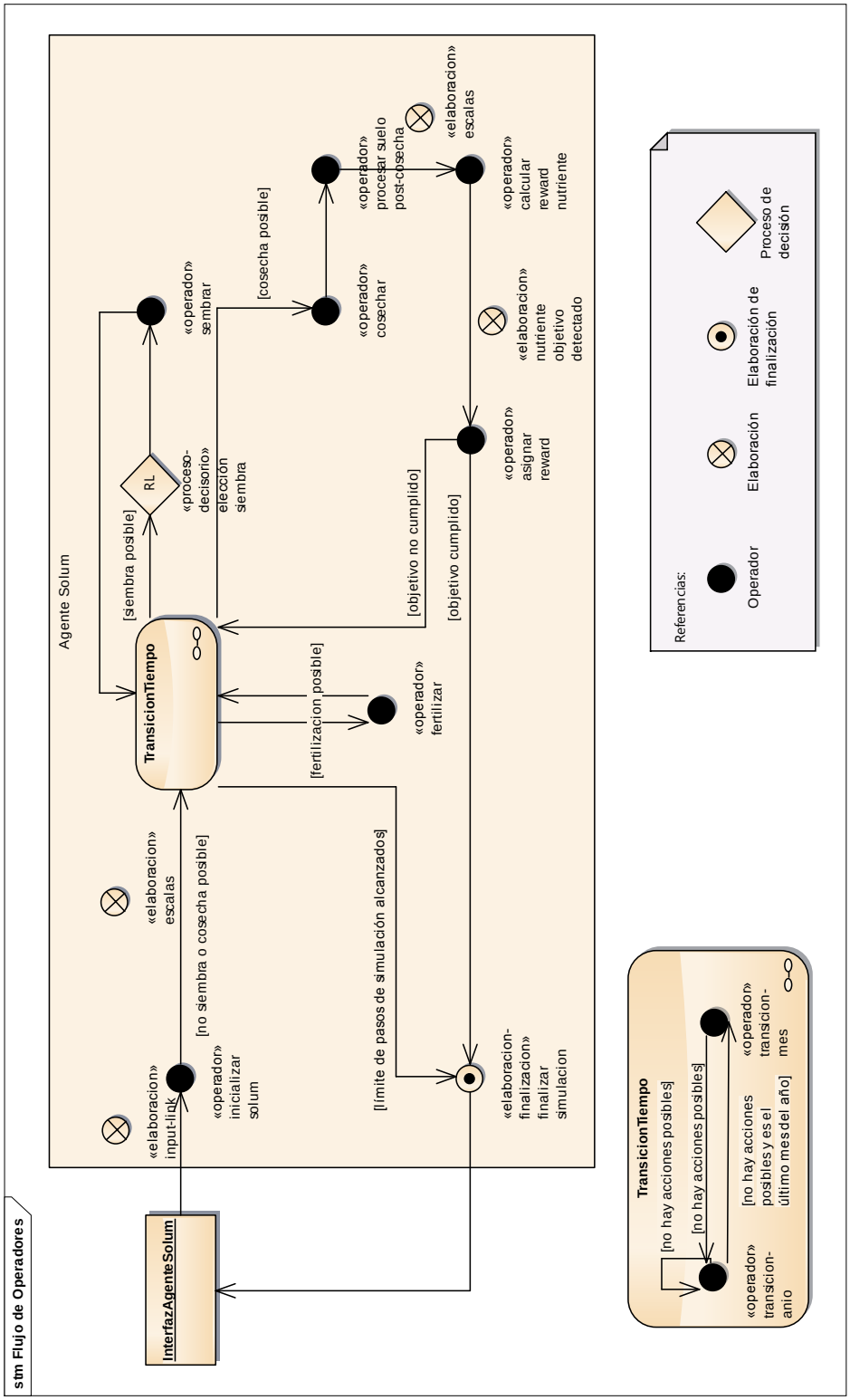
7.2. Diagrama de Flujo de Operadores

7.2.1. Introducción

El presente diagrama muestra cómo se organiza internamente cada ciclo de ejecución del Agente Solum, implementado bajo la Arquitectura Cognitiva Soar. Un ciclo de ejecución se compone de ejecuciones de una cierta cantidad de operadores y elaboraciones hasta que se alcanza algún objetivo preestablecido o bien se detiene el agente de forma explícita desde la arquitectura. A grandes rasgos, un operador es una función que se aplica en base a estado de la memoria de trabajo del agente, dividida en dos fases definidas como proposición y aplicación. En la primera fase de cada operador se enuncian las condiciones que el estado del agente debe cumplir para que el operador pueda ser propuesto para modificar el entorno. Por otra parte, en la fase de aplicación se detallan cuáles serán los cambios que el operador, en el caso de ser seleccionado, realizará sobre la memoria de trabajo. Cada operador, a su vez, se divide en dos partes: la primer parte (o Left Hand Side, LHS). En la misma, se presentan las condiciones que la memoria de trabajo debe cumplir para que el operador ejecute su segunda parte: la segunda parte (o Right Hand Side, RHS). La misma realiza cambios directamente sobre la memoria de trabajo. Entre medio de ambas fases existe una fase llamada fase de decisión, la cual toma todos aquellos operadores que han sido propuestos y selecciona uno, que será aplicado. Dicha selección puede realizarse por algún criterio explícito (especificando que un operador siempre debe seleccionarse antes que otro), indiferente (donde el operador seleccionado se determina aleatoriamente) o bien por medio de aprendizaje por refuerzos o el uso de sub-estados.

Por otra parte, las elaboraciones se presentan como funciones más simples, incluyendo solamente un LHS y un RHS. Cumplidas las condiciones del LHS, pasará a ejecutarse el RHS. Debe observarse que, a diferencia de los operadores, las elaboraciones presentan un comportamiento particular: las modificaciones que realizan están ligadas a las condiciones que las encadenaron inicialmente. Esto significa que si alguna de las condiciones del LHS deja de ser cumplida, la elaboración será retractada, quitando de la memoria de trabajo a lo que se generó desde el RHS de la elaboración.

7.2.2. Diagrama

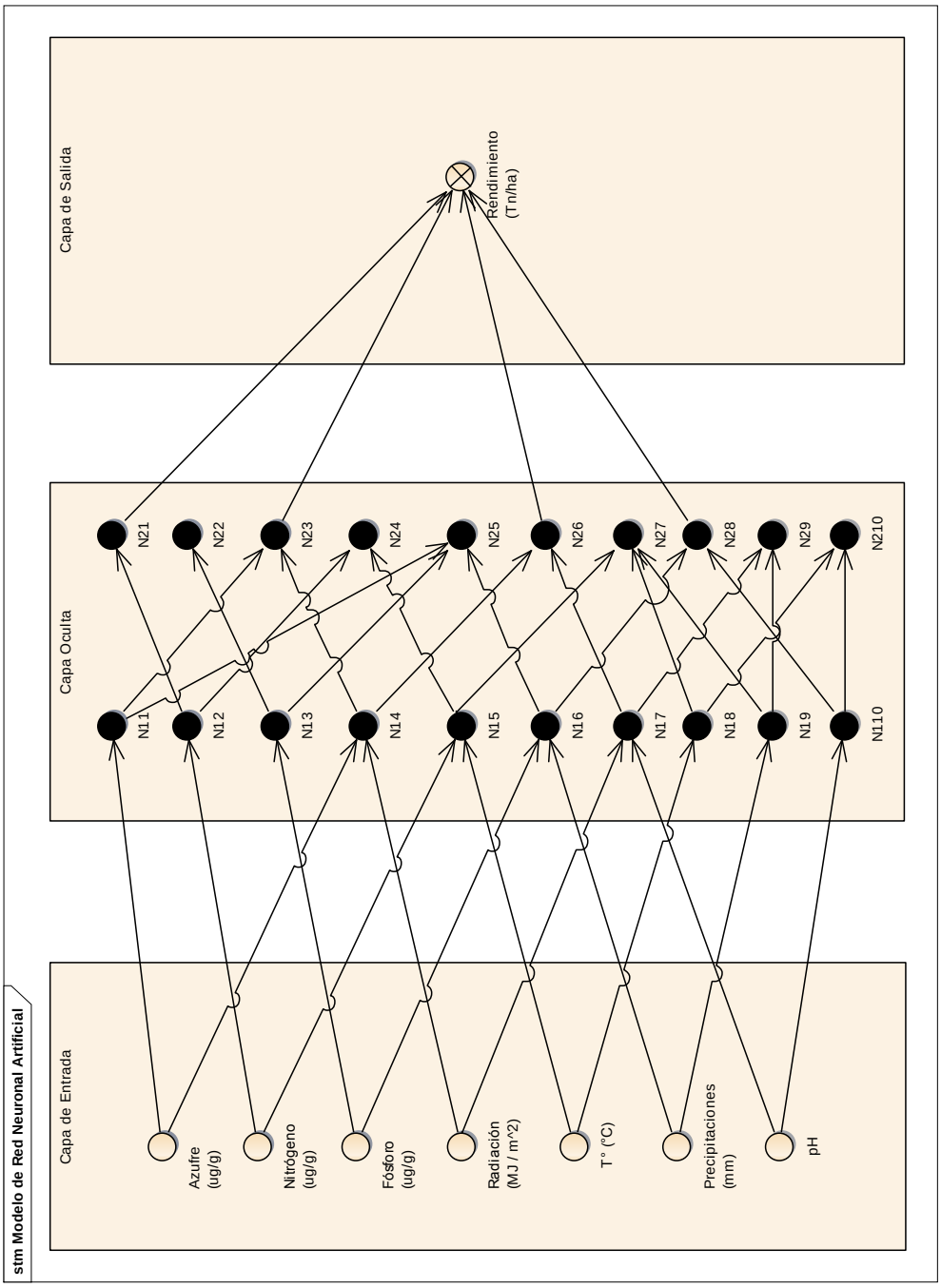

stm Flujo de Operadores
InterfazAgenteSolum
Agente Solum
«elaboracion» input-link
«operador» inicializar solum
«elaboracion» escalas
[no siembra o cosecha posible]
TransicionTiempo
[límite de pasos de simulación alcanzados]
«elaboracion-finalizacion» finalizar simulacion
[siembra posible]
RL
«proceso-decisorio» eleccion siembra
«operador» sembrar
[fertilizacion posible]
«operador» fertilizar
[cosecha posible]
«operador» cosechar
«operador» procesar suelo post-cosecha
«elaboracion» escalas
«operador» calcular reward nutriente
«elaboracion» nutriente objetivo detectado
[objetivo no cumplido]
[objetivo cumplido]
«operador» asignar reward
Referencias:
Operador
Elaboración
Elaboración de finalización
Proceso de decisión
TransicionTiempo
[no hay acciones posibles]
[no hay acciones posibles]
«operador» transicion-anio
[no hay acciones posibles y es el último mes del año]
«operador» transicion-mes

7.3. Diagrama de Red Neuronal

7.3.1. Introducción

El presente diagrama muestra el diseño que se efectuó para la Red Neuronal Artificial de Rendimiento de Cultivos implementada en Solum. En la misma puede observarse la presencia de los nodos para cada una de las capas de la red. Cabe destacar que, por motivos de simplicidad visual, han sido omitidas varias de las relaciones de los nodos de la capa oculta.

7.3.2. Diagrama



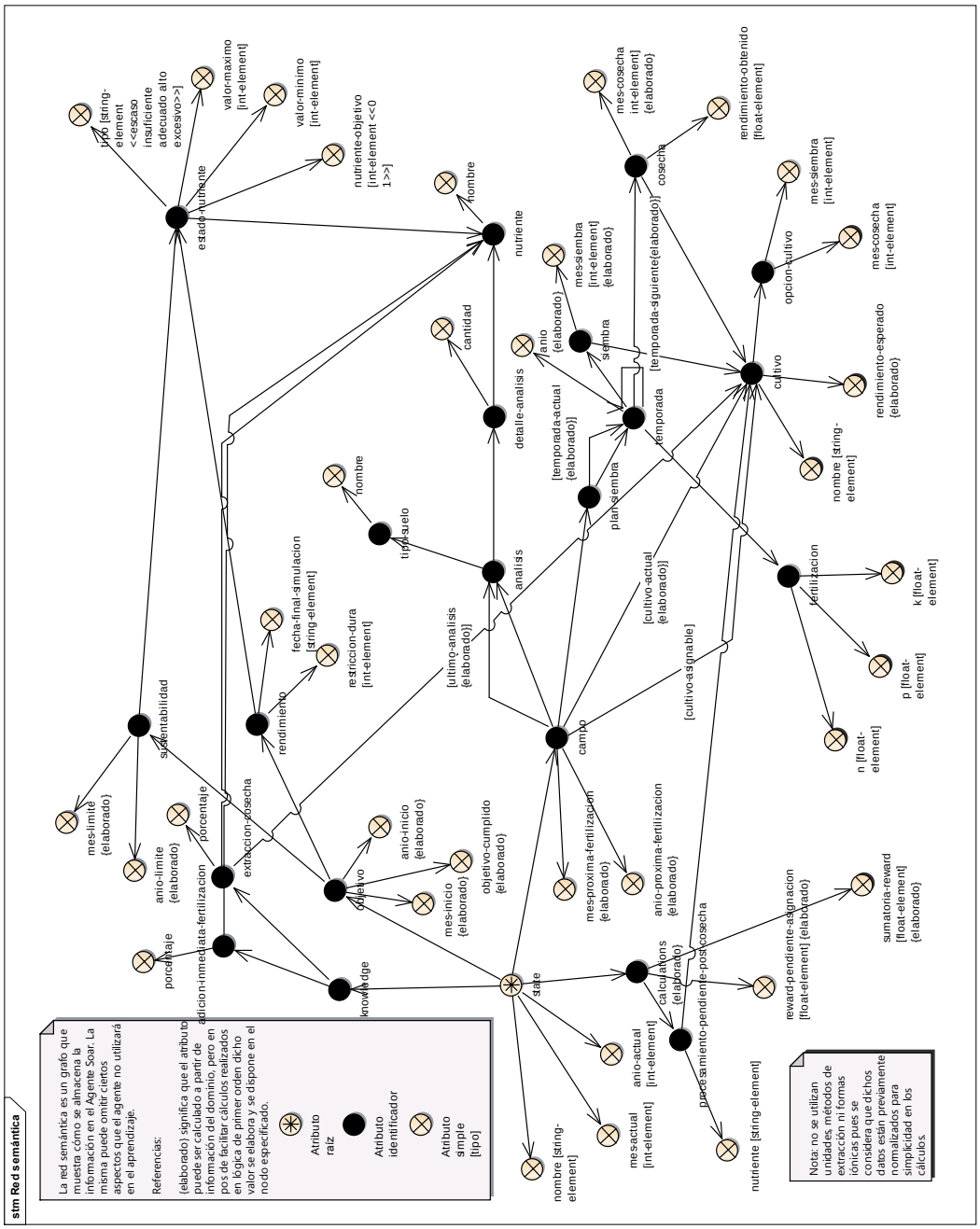
7.4. Diagrama de Red Semántica

7.4.1. Introducción

El presente diagrama muestra cómo el conocimiento del dominio es almacenado internamente en el agente que efectuará la simulación. Tal diagrama fue elaborado con el objeto de destacar cómo el agente se diferencia del diagrama de clases de diseño en ciertos puntos, a saber:

1. Parte de la información del dominio no es considerada dentro del conocimiento del agente. Esto es establecido de tal manera para limitar la cantidad de reglas que el agente generará automáticamente al realizar su aprendizaje, en pos de agilizar la convergencia del agente.
2. Se muestran aquellas variables de cálculo internas del agente. Las mismas son necesarias para llevar a cabo la simulación, representando variables tales como banderas que monitorean el cálculo de la recompensa.
3. Tres atributos distintos fueron incorporados en la red semántica. Por una parte, el atributo de tipo raíz que es el que marca el inicio de la red semántica. Similarmente, un atributo identificador es aquel que representa un atributo compuesto de la red semántica, es decir aquel atributo que puede a su vez contener otros atributos. Por último se incluyen los atributos simples, aquellos que contienen algún valor entero (int-element), de punto flotante (float-element) o de cadena (string-element) concretos..
4. Los nombres de los atributos identificadores utilizan la convención utilizada en Soar. En la misma, una clase llamada `EjemploClase` se escribe como `ejemplo-clase`; mientras que un atributo llamdo `atributoUno` se escribe como `atributo-uno`.

7.4.2. Diagrama



Capítulo VI – Ambiente de Implementación

1. Introducción

El presente documento exhibe el ambiente utilizado para realizar la implementación del Sistema de Información Solum.

2. Hardware

El Hardware utilizado para la implementación consistió en

- Computadora Core I5.
- 4 Gb de memoria RAM.
- Sistemas operativos Microsoft Windows 8 y Debian 8 Jessie.

3. Software y tecnologías utilizadas

3.1. Front-end

- Postgres v9.4.1 junto con el plugin Postgis, para realizar el soporte de la base de datos geoespacial.
- GeoServer, para montar el Servidor de Mapas v2.7.0.
- OpenStreetMap, para obtener datos sobre mapas.
- Leaflet Frontend Javascript Framework, para obtener mapas interactivos.
- AngularJS Frontend Javascript Framework v1.3.15.
- Bootstrap Frontend CSS Framework v3.3.4.
- W3af, para llevar a cabo análisis de vulnerabilidades Web.
- sqlmap, para llevar a cabo análisis de vulnerabilidades SQL.
- OpenVAS, para realizar análisis de vulnerabilidades en infraestructura.

3.2. Back-end

- Grails v2.3.8, para generación del framework web.
- Groovy/Grails Tool Suite 3.6.4.
- JDK 1.7.0_79.
- Groovy 2.3.10.
- Tomcat 7.0.52.

3.3. Agente Solum

- Java v7 para implementar conexión con el agente Solum.
- JUnit 4.10, con fines de realizar el testing del código implementado en Java.
- VisualSoar para implementar el agente Solum.
- Mathlab para implementar redes neuronales.
- SoarJavaDebugger para realizar el debug paso a paso del agente implementado.

3.4. Android

- Cordova v5.0.0.
- Ionic v1.5.

3.5. Gestión de versiones de código fuente

- Subversion v1.7.20.

4. Infraestructura

4.1. Introducción

Dadas las características de Solum, en donde el sistema está orientado a resolver consultas no sólo a la comunidad de especialistas agropecuarios, sino también a productores o gente vinculada por distintos motivos a la actividad como podrían ser instituciones educativas u organismos de diversos fines; la alta disponibilidad es prioritaria en Solum, con lo cual el equipo de ingeniería detrás del proyecto plantea un despliegue sobre una infraestructura resistente a fallos de hardware mediante la duplicidad de los datos y de equipamiento. Para llevar a cabo el despliegue de infraestructura propuesto, el equipo de Solum realizó un relevamiento de los productos de software existentes que se ajusten a las tecnologías de desarrollo seleccionadas.

Para llevar a cabo la alta disponibilidad de Solum, se requerirá de dos equipos que en lo posible presenten características idénticas.

4.2. Replicación de PostgreSQL

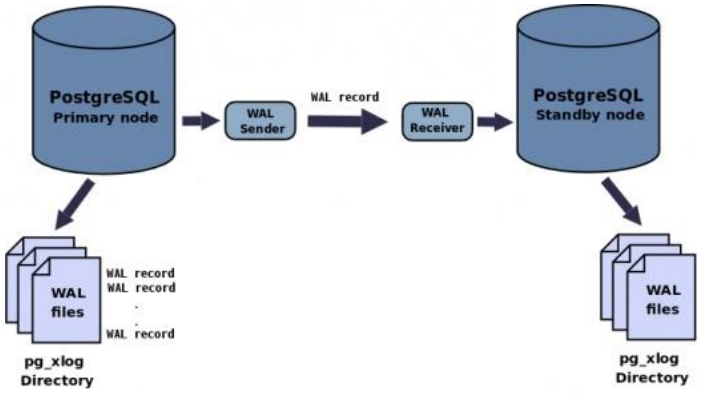
Solum utiliza PostgreSQL como motor de base de datos, el cual provee la capacidad de implementar réplicas de bases de datos de forma nativa y de manera activa/pasiva, contando con un servidor maestro que permite realizar operaciones de lectura/escritura, y otro servidor secundario se encuentra en espera, el cual se activa en el caso de detectar un fallo en el servidor maestro. Para ello se requiere una conexión punto a punto entre los servidores que permitan mantener la consistencia entre las bases de datos. Tomando en cuenta las capacidades de transferencia de datos existentes en la actualidad y las necesidades de Solum, se seleccionó un modelo de réplica de información denominado "Envío de Log de Transacciones", el cual puede ser implementado de tres maneras, basado en el envío del archivo de log de transacciones: por réplica, por streaming o por una combinación entre estas dos técnicas. Para los fines de este proyecto y las características del mismo se utilizará la réplica por streaming, debido a la baja sobrecarga que genera además de su simpleza y efectividad. Si bien existe una ventana de tiempo donde puede haber pérdida de información debido al rendimiento de la red punto a punto planteada, esta posibilidad es casi despreciable.

4.3. Consistencia de los datos

PostgreSQL utiliza los denominados ficheros WAL (Write Ahead Log / REDO) para guardar toda la información sobre las transacciones y cambios realizados en la base de datos. Los ficheros WAL se utilizan para garantizar la integridad de los datos grabados en la base de datos. También se utilizan para reparar automáticamente posibles inconsistencias en la base de datos después de una caída súbita del servidor.

La réplica mediante streaming funciona mediante la transferencia de registros WAL sobre la marcha entre servidores de bases de datos. Esta funcionalidad permite transferir asincrónicamente registros WAL sobre la marcha entre un servidor maestro y uno/varios esclavos.

En la práctica, un proceso denominado receptor WAL (WAL receiver) en el servidor esclavo se conecta mediante una conexión TCP/IP al servidor maestro. Por su parte, en el servidor maestro existe otro proceso denominado remitente WAL (WAL sender), que es el encargado de mandar los registros WAL sobre la marcha al servidor esclavo.



4.4. Desacoplamiento de punto de acceso a la base de datos

Es necesario que los servicios de aplicación se mantengan desacoplados del funcionamiento de la base de datos, con lo cual es requisito fundamental que se apunte a un destino único el origen de los datos, es decir una única dirección IP. A fines de implementar este requerimiento, se utilizará Heartbeat, el cual permite que un servidor secundario tome el direccionamiento del servidor primario (y, por lo

tanto, su rol dentro de la infraestructura de red) si se detecta un fallo en el dispositivo.

Heartbeat debe correr en ambos servidores, y su funcionamiento reside estableciendo una interfaz de monitoreo entre los dispositivos espejo, y otra interfaz que permanece a la espera de cualquier conflicto con el servidor primario y que se activa si se detecta un fallo.

A modo de ejemplo, considérense dos servidores "A" y "B", donde "A" es el servidor primario y "B" es el servidor secundario. El Servidor "A" tiene la IP de monitoreo 192.168.1.2/29 mientras que "B" tiene 192.168.1.3/29. Luego existe un IP de acceso al servicio, que para nuestro ejemplo será 192.168.1.1, esta dirección estará en ambos servidores. Tal IP se la conoce como IP de clusterización, sólo que en uno de los servidores (el secundario) permanecerá inactiva hasta que Heartbeat detecte un fallo en el servidor primario. Ocurrido esto, Heartbeat activará la interfaz con esta IP en el servidor secundario y este comenzará a responder a las consultas tanto de escritura como de lectura de los clientes.

En este caso de ejemplo el servidor podría estar en la IP 192.168.1.3/29 y conectarse entonces a la Base de Datos en 192.168.1.1/29 siempre, sin importar cuál de los dos equipos servidores de base de datos se encuentre activo. La infraestructura descripta en el ejemplo puede visualizarse en la siguiente imagen.

5. Base de Datos Geoespacial

5.1. Servidor de mapas y base de datos geoespacial

Solum gestiona información sobre Campos, siendo ésta una de las clases de dominio de mayor relevancia dentro del modelo de análisis y diseño planteado. Como parte íntegra de los campos y sus atributos más importantes se destaca la ubicación geográfica dentro del mundo, ya que a partir de ella se pueden determinar cualidades que van desde el clima, la composición típica del suelo o los cultivos posibles de desarrollo hasta determinadas cuestiones geopolíticas. Es por ello que una parte nuclear de Solum reside en la capacidad de administrar datos Geoespaciales y poder realizar operaciones sobre ellos, tales como determinar si un polígono contiene un punto, el área de un polígono o la distancia entre dos puntos.

Sumado a esto, es un requerimiento altamente deseable administrar estos datos de una manera sencilla para los humanos. Por ejemplo, al momento de definir los puntos que conforman el polígono límite de un campo, no es amigable ingresar un conjunto de coordenadas como

Punto 1: [-63.381500244140625, -32.42460139046622]

Punto 2: [-63.351287841796875, -32.430976720547015]

Punto 3: [-63.35678100585937,- 32.447782248455646]

Punto 4: [-63.386306762695305, -32.44198759273829]

ya que es casi imposible de recordar o entender a simple vista. En su lugar, utilizar un mapa georreferenciado le permite al usuario seleccionar un conjunto de puntos y poder generar por medio de éstos el polígono límite deseado.

5.2. PostgreSQL con extensión PostGIS

PostgreSQL no tiene soporte por sí mismo para almacenar datos geoespaciales ni realizar operaciones con ellos como las mencionadas en el párrafo anterior. Para dar solución a esta necesidad, Solum incorpora a PostgreSQL la extensión PostGIS, que añade estas capacidades a cualquier motor de base de datos. Entre las funcionalidades más destacadas se encuentra la posibilidad de definir atributos en las tablas del tipo geográficos, como puntos, multipuntos, líneas, multilíneas, polígonos, multipolígonos o colecciones geométricas; y también realizar

operaciones de mesura o comparación entre estos objetos entre las cuales se pueden mencionar el área, longitud o distancia de estos atributos, o también si un objeto geométrico contiene, interseca o tiene puntos en común con otro.

5.3. Origen de datos para referencia mediante OpenStreetMap

Con motivo de simplificar la lectura de un mapa, es importante contar con puntos de referencia como caminos rurales o rutas, ríos, edificaciones y ciudades entre otros posibles datos, de modo que la localización de un punto de importancia en Solum se haga no sólo mediante coordenadas sino que además permita identificar el mismo en base a las mencionadas referencias. En consecuencia, en Solum se optó por inicializar las tablas en la base de datos que almacenen puntos de referencia importantes (caminos rurales, rutas, ciudades, ríos y edificios, entre otros), de modo que al momento de gestionar información geoespacial se cuente con aquellos puntos importantes de referencia precargados. Estos datos podrían ser relevados, pero sería una enorme e innecesaria tarea tomando en cuenta que existe un proyecto comunitario de cartografía abierto y de libre utilización llamado OpenStreetMap que dispone tanto de servidores de mapas como de datos para geoespaciales para ser utilizados en bases de datos propias. Por lo tanto el equipo de desarrollo de Solum optó por inicializar la base de datos de puntos de referencia con los datos provistos por OpenStreetMap.

5.4. Servidor de Mapas GeoServer

Contar con los datos es parte de la solución, luego es necesario poder proveer a las aplicaciones de éstos mediante alguna interfaz estándar. Una de las existentes y más sencillas en su uso es la interfaz estándar WMS (Web Map Service), con lo cual Solum internamente solicita estos datos mediante esta interfaz.

Para servir esta información utilizando la interfaz WMS es necesario un Servidor de Mapas que, utilizando la base de datos Geoespacial PostgreSQL/PostGIS con estos puntos de referencia como origen de datos, reciba las consultas procese las mismas y retorne la información solicitada. GeoServer es una aplicación desarrollada en Java (mismo lenguaje de desarrollo de Solum), que es de libre utilización e implementa WMS entre otras interfaces estándar de consulta.

GeoServer permite definir Capas y Grupos de Capas acorde a un origen de datos como podría ser una tabla que contenga rutas, ríos, edificios o uso de tierras, entre

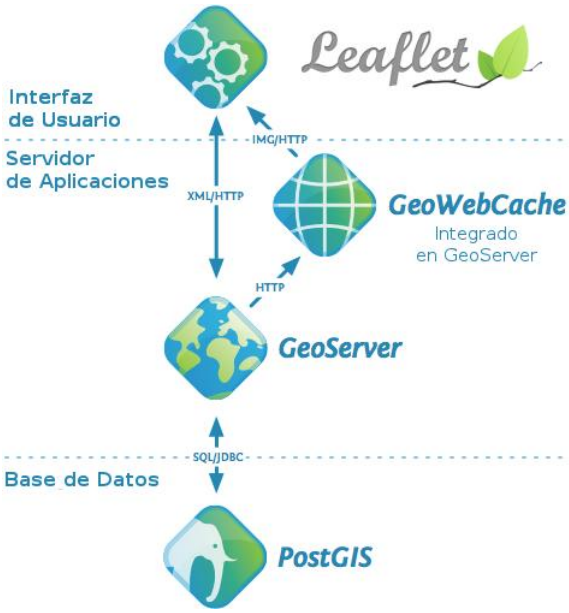
otros, y luego proveer estas capas renderizando en diversos formatos que el cliente solicite (PNG, JPG entre otros) a una aplicación que realice consultas de manera estándar, siendo así una capa intermedia entre la capa datos y la capa de vista.

Para el caso de Solum, GeoServer proveerá las capas Rutas, Uso de Tierras, Cursos de Agua, Ciudades y Edificaciones; no siendo esto un limitante para añadir posteriormente nuevas capas. Es importante destacar que los campos serán servidos por Solum en formato de texto que permita manipular con mayor precisión esta información.

Por último, GeoServer viene integrado con un caché en disco para agilizar el renderizado de las Capas, ya sea acorde a su utilización (es decir las capas más utilizadas se cachean para proveerlas más rápido posteriormente) o se permite definir zonas de mayor uso de forma previa para que GeoServer tenga precargadas las mismas, a este cache se lo llama GeoWebCache. Esta utilidad es muy potente y su utilización depende del espacio en Disco disponible, con lo cual se adaptará acorde a los recursos disponibles.

5.5. Gestión de mapas e implementación web mediante LeafletJS

Para poder administrar los mapas mediante una interfaz de usuario web, existe una librería muy sencilla, pero a la vez potente llamada LeafletJS, implementada en JavaScript, lenguaje dominante en el desarrollo de aplicaciones de cliente Web.



LeafletJS permite obtener las capas provistas por GeoServer, sobreponerlas, mostrarlas u ocultarlas de forma dinámica como también crear y mostrar nuevas capas. Como funcionalidad más importante se destaca que LeafletJS mantiene la georeferenciación de las capas provistas por GeoServer y logra que el usuario pueda obtener y administrar información geográfica de una manera mucho más sencilla.

Solum integra PostgreSQL/PostGIS, GeoServer y LeafletJS en la aplicación, logrando así un uso simple y de una capacidad de aprendizaje veloz por parte de los usuarios, lo que permitirá una mayor aceptación en su utilización.

6. Representational State Transfer (REST)

6.1. Introducción

Debido a que el proyecto funciona en un entorno web, el equipo de desarrollo tomó la determinación de inclinar la interfaz de comunicación entre el cliente y la capa de servicio de aplicaciones hacia un estilo de arquitectura de software REST (Representational State Transfer). La misma consiste en una guía de buenas prácticas para crear servicios web escalables mediante un protocolo de comunicación que respeta un conjunto de restricciones. Normalmente, los sistemas RESTful se comunican sobre el protocolo HTTP (Hyper Text Transfer Protocol) con los mismos verbos utilizados por los navegadores web para obtener y enviar datos como son el GET, POST, PUT y DELETE. Las interfaces REST usualmente involucran colecciones de recursos con identificadores, un ejemplo podría ser `/campo/pedro/analisis/20150724` siendo esto `/campo/{productor}/analisis/{fecha}`, el cual puede ser operado utilizando verbos estándar como DELETE para llevar a cabo la eliminación de dicho análisis de suelo en el campo de Pedro realizado con fecha 24 de julio de 2015.

La selección de este protocolo de comunicación cliente-servidor tiene sustento en las ventajas que presenta, como se pueden mencionar:

- Rendimiento: entre la interacción de los distintos componentes el rendimiento es un factor dominante que impacta directamente en la percepción del usuario y la eficiencia de la red. Para el caso de nuestro proyecto y dentro de las incumbencias de nuestra profesión, al ser un equipo interdisciplinario se realizó una división tareas específicas que requerían de una interfaz de comunicación simple y que permitiera interactuar a los distintos lenguajes y métodos de desarrollo.
- Escalabilidad: Soporta gran cantidad de componentes e interacción entre estos. Roy Fielding, uno de los principales autores de la especificación de HTTP, describe a la escalabilidad como:

"La separación cliente-servidor de REST consiste en la simplificación de implementación de componentes, reduce la complejidad de la semántica de los conectores, mejora la efectividad de los ajustes del rendimiento, y aumenta la escalabilidad de componentes de servicio puros. Los requerimientos de sistemas

en capas, permiten a los intermediarios – proxies, puertas de enlaces y firewalls – ser introducidos en varios puntos de la comunicación sin realizar cambios a las interfaces de los componentes, logrando así asistir en la traducción de la comunicación o mejorar la performance a larga escala. REST permite el procesamiento mediante mensajes con restricciones autodescriptivas, siendo estas interacciones carentes de estado entre las distintas consultas."

Solum está compuesto por una diversa cantidad de componentes tales como aplicaciones de servicio, aplicaciones clientes y aplicaciones de inteligencia artificial, que desde su concepción son considerablemente diferentes y requieren que su comunicación sea lo mayormente flexible y óptima posible. Para la comunicación de cada uno de sus componentes se prioriza en particular:

- Simplicidad en las interfaces.
- Permitir modificación de componentes acorde a la necesidad de cambios y que esto no tenga un impacto en otros componentes del sistema.

Si bien REST presenta otras ventajas como visibilidad de la comunicación, portabilidad de componentes y resistencia a fallos, para los fines del proyecto fueron determinantes y de bastos motivos para su utilización las arriba mencionadas.

6.2. Restricciones de Protocolo

Modelo cliente-servidor: Una interfaz uniforme separa al cliente del servidor y esta separación consiste en, por ejemplo, que los clientes no tienen conocimiento de cómo se almacenan ni qué tecnología se utiliza (SQL, LDAP, archivos, entre otros) para realizar dicho almacenamiento de datos, o en nuestro caso como se realiza el procesamiento de inteligencia artificial que permite mediante el envío de un grupo de análisis de suelo generar las actividades que logren un objetivo, sino simplemente solicitar a través de la interfaz propuesta se resuelvan los requerimientos y esperar la devolución de una respuesta estándar.

Protocolo carente de estado: La comunicación cliente-servidor no requiere que se almacenen datos de contexto del cliente en el servidor entre consultas. Cada consulta del cliente hacia el servidor es contiene toda la información requerida se envía en conjunto con la consulta, mientras que el estado del cliente se mantiene en el cliente.

Cacheable: El protocolo permite el cacheo de ciertas consultas, optimizando la interacción entre cliente y servidor.

Sistema en capas: El cliente no sabe si está conectado directamente al servidor o a un sistema intermedio que pueda permitir por ejemplo balance de carga o reforzar características como la seguridad permitiendo insertar políticas de seguridad o cache de consultas.

Interfaz uniforme: Esta es la característica fundamental de REST. El uso de interfaces uniformes simplifica y desacopla la arquitectura, lo que permite a cada parte desarrollarse independientemente de las demás.

6.3. Especificaciones de la implementación REST en Solum:

Verbos de comunicación: Se utilizarán los verbos estándar de HTTP para la comunicación cliente-servidor de la siguiente manera:

- GET permitirá obtener un objeto o una lista de objetos.
- POST llevar a cabo la creación de nuevos objetos.
- PUT la modificación de objetos.
- DELETE la eliminación de objetos.

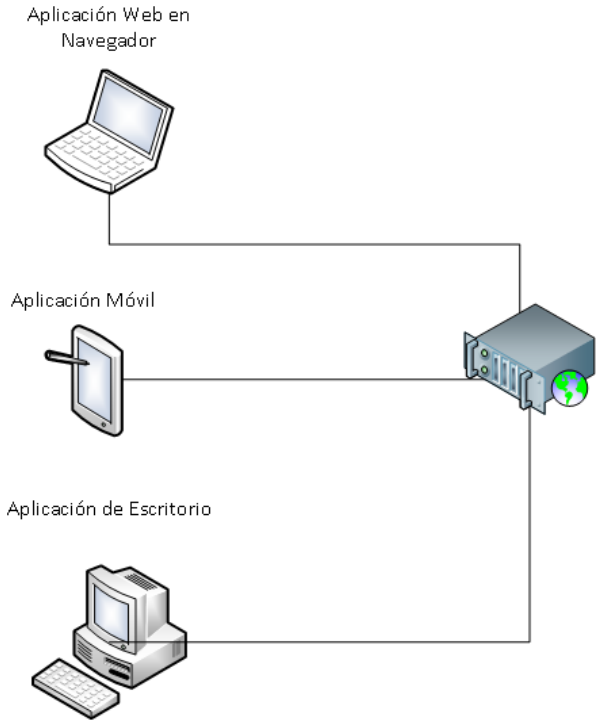
Lenguaje de descripción de objetos: Para la descripción de los objetos el equipo de desarrollo definió utilizar JSON, por su amplia aceptación por diferentes tecnologías de desarrollo, disponibilidad de librerías, simpleza de uso, la capacidad auto descriptiva del lenguaje y lo claro que resulta su entendimiento.

Se muestra a continuación un ejemplo de implementación y funcionamiento de REST en Solum, la creación de un análisis de suelo, donde el cliente debe simplemente enviar una solicitud con el verbo POST mediante protocolo HTTP al recurso `/campo/pedro/analisis` con los datos propios del análisis como podrían ser los siguientes:

```
{"analisis":
     {
          "CEC": 12.16,
          "fecha": "24/07/2015",
          "color": "marrón",
          "conductividad_electrica": 12,
          "latitud": -32.41102,
          "longitud": -63.20755,
          "detalle_analisis": [
               {
                    "metodo_extraccion": "pasta
                    saturada",
                    "nutriente": "potasio",
                    "valor": 45
               },
               {
                    "metodo_extraccion": "metodo 2",
```

```
                    "nutriente": "fosforo",
                    "valor": 10
               },
               {
                    "metodo_extraccion": "metodo 2",
                    "nutriente": "azufre",
                    "valor": 150
               }
          ],
          "lote": {
               "descripcion": "Extracción del campo de
Pedro Fuentes en las                    inmediaciones de
Villa María el día 24 de julio de 2015",
               "numero_lote": 2
          },
          "porcentaje_materia_organica": 0.4,
          "porcentaje_humedad": 0.8,
          "temperatura_media": 15,
          "textura":[
               {
                    "nombre": "arena",
                    "porcentaje": 0.2},
               {
                    "nombre": "limo",
                    "porcentaje": 0.5
               },
               {
                    "nombre": "arcilla",
                    "porcentaje": 0.3
               },
          ],
          "tipo_suelo": {
               "nombre": "molisol"
          }
     }
}
```

De esta manera se puede lograr una buena interacción con la aplicación desde distintos tipos de dispositivos independientemente de la tecnología de desarrollo que se utilice en el cliente, obteniendo una arquitectura que responde al siguiente modelo de despliegue.



Capítulo VII – Convenciones de Implementación

0. Introducción

El presente documento incluye todas las convenciones de programación acordadas hasta el momento para la implementación del sistema Solum. El mismo se divide en dos partes: la primera incluye las convenciones acerca de las prácticas de programación que se aplicarán para generar un código con la mayor calidad posible. La segunda parte, por otro lado, incluye todo lo acordado hasta el momento acerca de los diseños de interfaz.

Parte I – Prácticas de programación

1.1. Reglas de legibilidad

Poder entender fácilmente el código es un requisito fundamental para que el mismo pueda ser sometido a mantenimiento por cualquier persona del equipo, aún si dicha persona no ha sido quien efectuó esa codificación. Para ello se enuncian a continuación una serie de buenas prácticas que darán soporte a esto.

- Uso intensivo de comentarios. Los comentarios añaden una semántica y un contexto a las líneas de código, resultando fundamentales para entenderlo rápidamente. Se usarán comentarios en las siguientes partes del código.
    - Al inicio de cada clase o interfaz, usando la plantilla "/** <Comentario> */".
    - Al inicio de cada método, usando la plantilla "/** <Comentario> */". En cada método que así lo requiera se definirá también.
    - Al definir atributos importantes.
    - Al definir partes de código complejas, como estructuras anidadas o llamadas recursivas.
    - En cualquier otra parte en la que se quiera clarificar algún aspecto.
- Uso intensivo de sangrías y espacios. Para aumentar la legibilidad, se utilizarán considerablemente los espacios para separar las líneas de código y las sangrías para enfatizar su contexto.
- Nomenclatura. Las reglas de nomenclatura son las siguientes:
    - Para las clases y atributos se utilizarán nombres en sentido gramatical (nunca verbos).
    - Para los métodos se utilizarán nombres que describan a primera vista todo lo que el método hace.
    - Para los métodos sin valor de retorno se utilizarán normalmente nombres conformados por un verbo seguido por un objeto al que afecte el verbo (por ejemplo: imprimirFactura).
    - Para los métodos con valor de retorno se utilizará normalmente una descripción del valor devuelto.
    - Se evitará el usar nombres con verbos aplicables a todo contexto, como por ejemplo procesar.

  - Se evitará el uso de nombres genéricos o ambiguos.
  - En el caso de que existan grupos de métodos que realicen operaciones similares se establecerá un sistema de creación de nombres coherente.
  - Se evitarán los nombres sin sentido de las variables temporales.
  - Para las variables booleanas se utilizarán nombres positivos que sugieran una respuesta del tipo “Si” o “No”.

- Siempre que sea posible, se intentará que un método no ocupe más de 100 líneas (incluyendo comentarios, sangrías, etc.).
- Nunca un método ocupará más de 200 líneas.
- Nomenclatura capitalizada. En cada método y en cada clase, la primera letra de cada palabra de su identificador se escribirá en mayúsculas. En los atributos se seguirá el mismo criterio con la excepción del primer carácter.
- Cuando una parte de un método se vuelva obsoleta, será eliminada del código (en lugar de dejarla comentada para tener un resguardo).
- En lo posible, un método no debería superar los 7 parámetros.
- Se evitará el uso de significados “ocultos” en las variables. Un ejemplo de esto sería si tomar la variable porcentajeHumedad y le asignarle -1 cuando se desconoce.
- Se evitará el uso de los valores directos en el programa. En su lugar se utilizarán constantes (excepto para cierto tipo de valores, como “true”).
- Al crear cualquier variable, parámetro o valor de retorno que en algún momento dado pueda tomar el valor “null” se debe explicar de alguna forma qué significa el mismo.
- En estructuras del tipo if … then … else …; la condición positiva debe ser colocada primero.
- Las condiciones iniciales y de terminación del bucle deben ser claras por su propia declaración, sin necesidad de inspeccionar el código (siempre que sea posible y manejando baja complejidad).
- Es recomendable que un método no tenga una complejidad (cantidad de if, while, do, for, …) mayor a 7; y en ninguna situación mayor a 15.
- La longitud de una línea de código no debería superar el ancho de la pantalla. Si es realmente necesario que esto suceda, la línea debería partirse de forma que quede claro que se continuará en la siguiente.

- Una línea de código tendrá sólo una sentencia.
- Se evitarán las declaraciones múltiples de variables del mismo tipo.

1.2. Reglas de interfaz y métodos

Se incluyen las reglas que regulan las relaciones entre las distintas partes del sistema y la codificación de métodos.

- Todo método debe ser utilizado por alguien.
- Todo método debe definir claramente sus precondiciones y postcondiciones; pero no necesariamente verificarlos.
- Todos los métodos que se utilizan para acceder al sistema deben verificar sus precondiciones. Se debe hacer énfasis en:
    - Datos provenientes del usuario.
    - Datos originados por otros programas.
    - Datos provenientes de archivos generados por otro programa.
    - Datos que procedan de algún canal inseguro.
    - Datos compartidos entre aplicaciones.
- Los siguientes métodos deberán siempre verificar sus pre y post condiciones:
    - Aquellos que trabajen con excepciones.
    - Aquellos que afecten a la computadora, sistema operativo u otros programas.
    - Aquellos que accedan a la base de datos.
- Un método que sobrescribe a otro nunca debe ampliar su conjunto de precondiciones.
- Un método que sobrescribe a otro nunca debe reducir su conjunto de postcondiciones.
- Un método no puede disminuir sus invariantes.
- Un método que sobrescribe a otro debe tener excepciones que representen un subconjunto (no necesariamente propio) del mismo.
- Si varios métodos usan parámetros similares, el orden de los mismos debe ser coherente.
- Un método debe usar todos los parámetros pasados.

- Para evitar excepciones innecesarias, todos los métodos que devuelvan estructuras de datos de cualquier tipo devolverán una estructura vacía para representar la ausencia de datos (en lugar de un valor “null”).

1.3. Reglas de alta cohesión

La cohesión es aquella propiedad que determina el nivel de relación entre las distintas partes de un módulo. Para conseguir que Solum alcance un alto nivel de reutilización y sea fácil de mantener, los esfuerzos de codificación se concentrarán en intentar mantener la cohesión en un alto nivel. Para ello se utilizarán las siguientes reglas:

- Cada método debe tener un contrato totalmente definido, de modo que quienes dependan de él siempre sepan qué servicios brindará el mismo más allá de su implementación. El mismo incluye:
    - Signatura del método (parámetros aceptados y devueltos).
    - Precondiciones.
    - Postcondiciones.
    - Excepciones que puede generar.

- Al codificar, siempre se utilizará alguno de los tres tipos de cohesión, siempre que no haya alguna restricción o aspecto de diseño que lo impida:
    - Funcional: donde cada método utiliza una única operación.
    - Secuencial: cuando el método contiene elementos que deben ser realizados en un orden específico o bien comparten datos.
    - Comunicativa: cuando la única relación entre los pasos de un método es la información que comparten.

- Cuando se desee generar una dependencia entre dos objetos sin relación, se priorizará el uso de patrones para crear una entidad intermediaria y así evitar generar una dependencia directa.
- Se aislarán en métodos los siguientes aspectos de un programa:
    - Áreas de cambio probable o frecuente.
    - Operaciones con datos complejos.
    - Segmentos de código con algoritmos y lógica compleja.

- Todo aquel código que haga uso de operaciones particulares de la computadora, lenguaje o sistema operativo.

1.4. Reglas de bajo acoplamiento

El acoplamiento mide la ligadura entre los distintos componentes del sistema. Para desarrollar un sistema fácilmente mantenible y reutilizable, los esfuerzos se centrarán en mantener bajo el acoplamiento, de modo que se incluyan solamente las dependencias necesarias. Para ello se utilizarán las siguientes reglas:

- Se optará por alguno de los siguientes niveles de acoplamiento, siempre que no haya algún motivo de diseño que lo impida:
  - Acoplamiento por datos simples: ocurre cuando el único acoplamiento de información que existe entre dos partes está dado por una lista de parámetros.
  - Acoplamiento completo por datos no estructurados. Igual que el caso anterior, pero con el agregado de que la casi todos los parámetros de la lista se usan.
  - Acoplamiento simple por objetos. Se dice que un subsistema está acoplado de forma simple a un objeto si el mismo crea instancias de ese objeto.
- Se aplicará, siempre que sea posible y conveniente, la ley de Demeter: Para toda clase C con un conjunto M de métodos, todos los objetos con los cuales interactúa M deben cumplir alguna de las siguientes propiedades:
  - Ser instancias de las clases a las que pertenecen los argumentos de M.
  - Instancias de C.
  - Objetos globales.
  - Objetos directamente creados o invocados por M.
- Se minimizarán las clases estáticas.
- Se declararán los campos como no heredables.
- Se utilizarán siempre valores por referencia.

- Se utilizará la clonación (siempre que sea posible) en métodos donde se realice una manipulación compleja de algún objeto.
- Se encapsularán los atributos y métodos de cada clase que no sean utilizados por otras.
- Se minimizará el uso de la herencia siempre que sea posible. La herencia, a pesar de su sencillez y facilidad de modificación en términos de código, presenta una serie de inconvenientes. Entre ellos se destaca la imposibilidad de cambiar las implementaciones heredadas de las clases en tiempo de ejecución y la limitación de la flexibilidad y reutilización generadas al romper el encapsulamiento. Para desarrollar un sistema que sea lo suficientemente mantenible, se tratará de minimizar dichas relaciones, dándole lugar a las interfaces y relaciones de composición.

1.5. Otras reglas

Se incluyen a continuación otras reglas de buenas prácticas de programación que no pertenecen a las categorías anteriores.

- Evitar el uso del tipo de datos “float”.
- Al utilizar variables, se debe procurar que su tiempo de vida sea pequeño.
- En los bloques condicionales, se evitará utilizar un nivel de anidación superior a los cuatro niveles.
- Se evitará el uso de cláusulas if … then <acción1> else <acción2> con <acción1> vacía.
- Siempre se utilizarán evaluaciones booleanas de circuito corto.
- Se evitará la anidación del operador “?”.
- El interior de un bucle debería ser tratado como si fuera un método.

Parte II – Interfaz gráfica

2.1. Aspectos básicos

- Notificación de errores.
    - Cuando una excepción es propagada hacia la capa de interfaz y deba ser mostrada, debe ofrecerse al usuario alguna de la siguiente información:
        - Qué acaba de suceder.
        - Porqué se está mostrando el mensaje de error.
        - Cómo se ve afectado el usuario por el mismo.
        - Qué acciones son posibles a partir de ahora.
        - Qué se puede hacer para evitar el error, en caso de que sea evitable.
        - A quién debería dirigirse para solucionar el problema.
    - Siempre debe poder guardarse toda la información técnica del error.
- Consistencia. El software debe ser homogéneo en la forma en que las funciones de la interfaz trabajan, el lenguaje del usuario y los estándares vigentes. Por ejemplo, si vemos dos íconos iguales en distintas partes del programa, es razonable pensar que harán lo mismo.
- Control. El usuario debe sentir control sobre lo que está haciendo. Esto se consigue con información que actúe como guía.
- Sensibilidad. En procesos largos el usuario debe notar un progreso.
- Personalización. El usuario debe poder, en la medida de lo posible, personalizar los elementos en pantalla y su experiencia con el sistema.
- Claridad. La información que se muestra en pantalla debe ser inmediatamente comprensible y no presentar ambigüedades.
- Confirmación de acciones peligrosas o críticas. El usuario debe ser advertido cuando esté por realizar una acción de este tipo.

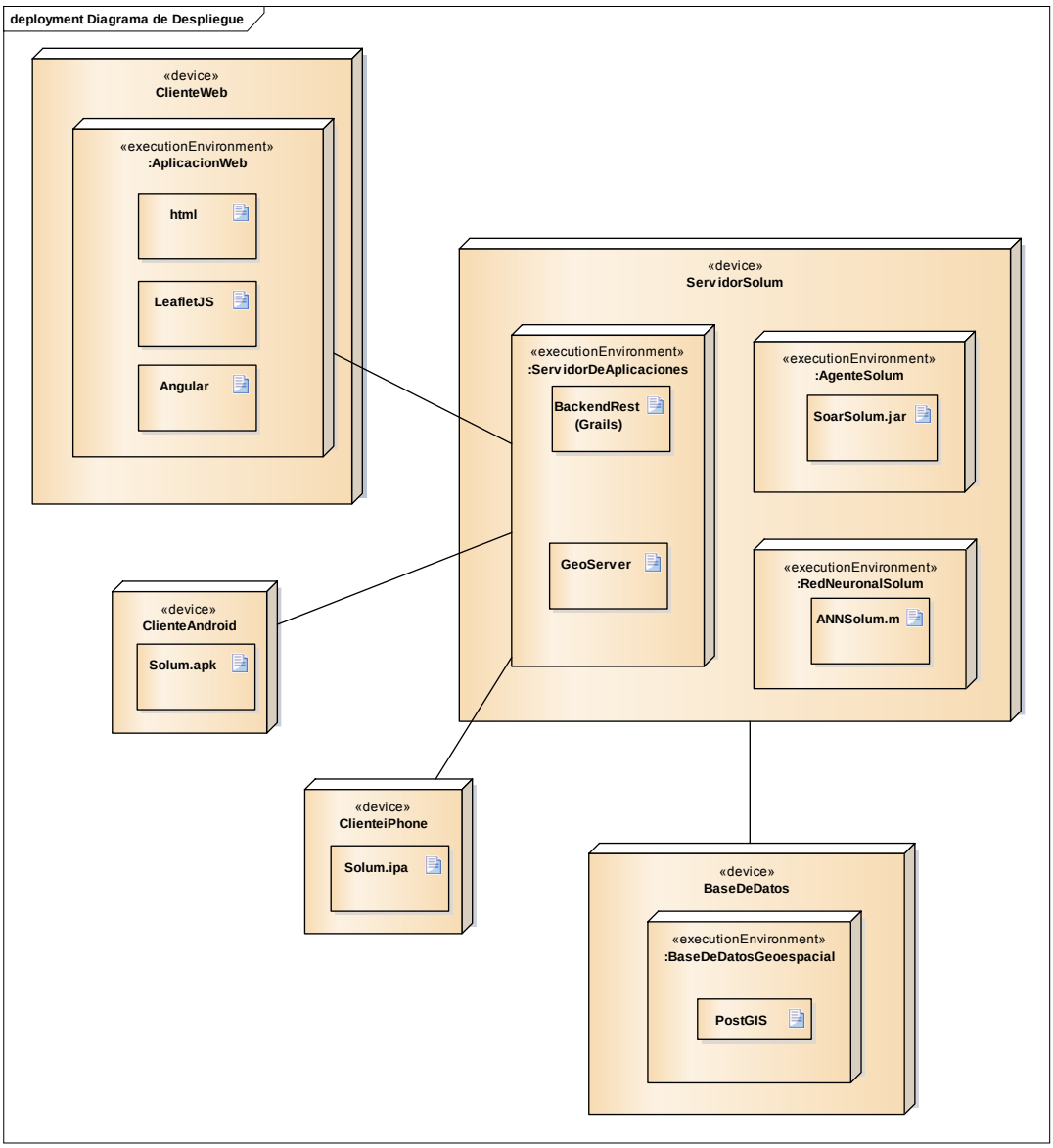
- Equilibrar la información y la cantidad de ventanas

• Capítulo VIII – Modelo de Despliegue

• 1. Introducción

• El presente documento tiene como objetivo mostrar el Modelo de Despliegue utilizado en Solum. El mismo presenta cómo se modeló al Sistema a través de sus distintos nodos, ambientes de ejecución y artefactos.

- **2. Diagrama de Despliegue**



Capítulo IX – Plan de Testing

1. Introducción

El presente documento tiene como propósito especificar las estrategias necesarias para definir el modo en el que debe ser probado y evaluado – en parámetros de calidad y satisfacción del cliente – el software Solum.

El plan de pruebas especificado en la presente sección se basa en los siguientes objetivos:

- Identificar los elementos que pueden ser objetivo de pruebas y el alcance de estas.
- Determinar el cumplimiento de los requerimientos
- Evaluar la calidad del sistema Solum teniendo en cuenta las necesidades de los clientes y usuarios finales.
- Describir las estrategias que serán usadas para el testeo correspondiente.
- Definir la clasificación para los defectos encontrados.
- Definir los criterios de entradas y salidas para los test realizados.

1.1. Alcance

Este documento describe el plan de testing del sistema de información Solum, cuyo objetivo principal es brindar soporte a INTA, productores particulares e ingenieros agrónomos, acerca del desarrollo de prácticas de agricultura. Ofreciéndoles una base de información, conocimiento y asesoría a la toma de decisiones orientada hacia la conservación y rentabilidad del suelo.

El presente Plan de Testing se muestra como una especificación de alto nivel de los requerimientos funcionales y de calidad que serán probados, del ambiente de testing, de la estrategia de testing, como así también de las responsabilidades y criterios de éxito.

El comportamiento que exhiba el Sistema Solum será contrastado contra la Especificación de Requerimientos de Software. Los casos de prueba como así también los criterios de éxito serán derivados del mismo.

1.2. Propósito

Solum brinda las siguientes funcionalidades generales:

- Seguimiento del estado de suelo, es decir, en base a una serie de análisis de suelo, y a una variedad de minerales selectos
- Presentar informes estadísticos e históricos del estado del suelo.
- Simular estados futuros del suelo.
- Proponer mediante un componente de inteligencia artificial (IA) diversas acciones que aspiran a mejorar la situación simulada, previniendo o minimizando la degradación de los recursos terrestres.
- Generar informes simples para motivar la participación de la comunidad, para generar una fuente de información confiable y amplia.

1.3. Criterios de Entrada

Para poder llevar a cabo las pruebas del Sistema, deben cumplirse las siguientes precondiciones.

- Los test unitarios deben encontrarse realizados y completos para cada uno de los componentes del sistema.
- El sistema debe estar completamente integrado con todos sus módulos y dependencias incluidas.

1.4. Criterios de Salida

Para que el test del Sistema pueda considerarse aceptable, los siguientes criterios deben ser cumplidos al finalizar el mismo:

- El agente inteligente debe presentar, tras el entrenamiento realizado, una curva que demuestre su convergencia para cada uno de los objetivos de simulación. Dicha curva probará la convergencia del agente mostrando una tendencia decreciente entre el promedio de siembras realizadas por cada episodio de entrenamiento.
- El sistema debe poder funcionar de forma estable (sin errores de nivel Medio, Alto o Crítico) la funcionalidad descripta en los Casos de Uso.

2. Test del Sistema

El sistema a probar se define como un sistema de información de soporte a la toma de decisiones, con un módulo adicional que implementa inteligencia artificial y permite realizar simulaciones a futuro, plantear escenarios actuales (reales o no) y obtener resultados a futuro, brindando al usuario recomendaciones o consejos sobre acciones a realizar para mejorar o mantener su tierra.

2.1. Estrategia de Testing

El enfoque seleccionado para la etapa de testing es una fusión de técnicas analíticas –basándose en las áreas de mayor riesgo del sistema- y dinámicas –centrado en pruebas exploratorias reactivas a eventos donde ejecución y evaluación son realizadas en paralelo-.

Este enfoque se debe a la naturaleza del sistema: contiene un área sumamente crítica: agente inteligente; tiempos de planificación sumamente cortos y una gran necesidad de adaptar las pruebas a las reacciones y cambios del sistema. Finalmente la experiencia de los responsables de la ejecución de pruebas, lo que brinda cierto grado de confianza y efectividad a las pruebas exploratorias frente a técnicas estáticas de testing.

Las etapas definidas a desarrollar son:

- Testing unitario y de componentes (llevado a cabo por los desarrolladores del sistema basándose en revisiones cruzadas de código)
- Pruebas de integración de componentes (llevado a cabo por los desarrolladores del sistema)
- Testing sistémico (llevado a cabo por el responsable de testing) Esta etapa incluirá ejecución de test cases funcionales y técnicas de testing exploratorio en sesiones.
- Testing de aceptación (llevado a cabo por el responsable de testing en conjunto con los clientes)
- Testing de documentación

2.2. Actividades de la Etapa de Pruebas

A continuación se mencionan las actividades básicas a realizar en esta etapa en el orden en el que deben ser realizadas:

- Revisar base de pruebas (requerimientos, integridad del software –nivel de riesgo-, arquitectura y especificaciones de interfaz).
- Identificar y priorizar las condiciones de prueba en base al análisis de los elementos de prueba, la especificación y estructura del software.
- Diseñar casos de prueba de alto nivel
- Evaluar cobertura de los módulos del sistema con los casos de prueba diseñados.
- Identificar y diseñar escenarios básicos a incluirse en el testing exploratorio.
- Crear datos de prueba.
- Ejecutar casos de prueba diseñados.
- Ejecutar testing exploratorio.
- Registrar resultados de los ciclos de pruebas.
- Comparar resultados reales con los esperados.
- Reportar incidencias a los responsables, ya sea por escrito o de manera informal.
- Evaluar la necesidad de realizar un nuevo ciclo de prueba según los resultados obtenidos.
- Analizar lecciones aprendidas para determinar cambios futuros necesarios.

2.3. Casos de prueba y errores

Los casos de prueba son registrados en planillas divididos por área del sistema relacionada con los mismos. Los resultados de las ejecuciones incluyen los defectos encontrados con la debida información.

Un caso de prueba puede encontrarse en los siguientes estados:

- Borrador: el caso de prueba se está planificando y diseñando, como así también su ambiente, condiciones iniciales, etc.
- Pendiente de ejecución: el caso de prueba está diseñado, pendiente de ser ejecutado por primera vez.

- En ejecución: el caso fue ejecutado por lo menos una vez, y es necesario utilizarlo nuevamente, o está transitando por sus primeras ejecuciones.
- Finalizado: este estado es arribado una vez que el caso termina de utilizarse las veces que sean necesarias. Puede finalizarse con o sin éxito, según los criterios definidos anteriormente en este punto.
- Descartado: el equipo desea desechar un caso porque resulta obsoleto al momento de su ejecución, o esta desactualizado, o simplemente se considera innecesario.

A su vez, los resultados que arroja un caso de prueba Finalizado con éxito pueden ser clasificados y encontrarse en un estado asociado según lo siguiente:

- Abierto: el error necesita ser tratado para evaluar posibles cambios o formas de eliminarlo, criticidad, entre otros.
- Buscando soluciones: el error está siendo discutido con el resto del equipo, para determinar su posible solución.
- Pendiente de corrección: las correcciones ya han sido discutidas, acordadas y aprobadas por todos los intervinientes, restando solamente su implementación.
- Solucionando: las correcciones están siendo realizadas a fin de eliminar o reducir notablemente el error.
- Probando soluciones: una vez realizadas las correcciones, se procede a ejecutar nuevamente el caso de prueba que localizo el error y se espera no aparezca nuevamente o se repite el ciclo.
- No reproducible: el error reportado no ha podido ser nuevamente reproducido.

Por su parte, los distintos defectos encontrados pueden ser clasificados como se lista a continuación.

- Defecto Cosmético: La información se muestra correctamente pero el aspecto es incorrecto. Esto ocurre por hechos tales como palabras mal escritas, fuente o formato incorrectos, etc.
- Bajo: Un defecto que afecta un requisito no primordial para el cual existe una solución alternativa.

- Medio: Una defecto que afecta un requisito no primordial para el cual no existe una solución alternativa. La funcionalidad no puede ser utilizada.
- Alto: Un defecto que afecta de forma directa a un requisito primordial para el cual existe una solución alternativa. El uso o la prueba del sistema puede continuar de modo degradado.
- Crítico: Problema de seguridad o que afecta a un requisito primordial para el cual no existe una solución alternativa. El mismo impide el uso o la prueba del sistema.

2.4. Entregables

Se realizarán entregas periódicas que incluirán casos de prueba ejecutados, escenarios básicos cubiertos en test exploratorios y los respectivos informes de salida o errores encontrados.

3. Configuración del Testing

La presente sección establece los componentes que definen el ambiente de testing.

3.1. Hardware

- Computadora Cliente
- Computadora Core I3.
- 2 Gb de memoria RAM.

3.2. Software

- Sistemas operativos Microsoft Windows 8 y Debian 8 Jessie para las computadoras que ejecuten los servicios de back-end.
- Navegadores instalados en cada una de las computadoras clientes. Preferentemente Chrome y Firefox actualizados. Otros navegadores pueden ser utilizados teniendo en cuenta que pueden existir degradaciones en la interfaz y performance del sistema.

3.3. Otros

- Conexión a internet preestablecida.
- SML.jar precargado en la librería del proyecto Java.
- Soar.Dll ubicado en Windows/SysWoW64.
- SML_Java_Interface ubicado en Windows/SysWoW64.

4. Responsabilidades

4.1. Responsabilidades de los desarrolladores

- Ejecutar las pruebas unitarias en cada uno de sus módulos.
- Ejecutar pruebas de integración de componentes
- Realizar revisiones de código de otros desarrolladores informando el resultado de las mismas y las sugerencias de cambios.
- Corregir los problemas reportados.

4.2. Responsabilidades del encargado de testing

- Planificar las pruebas del sistema.
- Configurar el ambiente de prueba.
- Diseñar y organizar casos de prueba y escenarios básicos para guiar testing exploratorio.
- Ejecutar las pruebas del sistema.
- Escribir el reporte de testing.

5. Anexos

En el presente Anexo se incluyen los escenarios básicos a incluir dentro de los ciclos de testing exploratorio y también los Casos de Prueba a ejecutar organizados por área del sistema.

Solum	
Proyecto:	**Solum**
Área:	**IA**
Escenario #	Descripción
IA_001	Ingresar análisis de suelo que se encuentra fuera del promedio de datos ya ingresados
IA_002	Variar parámetros de ingreso dentro de los límites normales
IA_003	Ejecutar sólo una vez al módulo IA y validar las variaciones entre aprendizaje previo y posterior a la ejecución

Solum	
Proyecto:	**Solum**
Área:	**Usuarios**
Escenario #	Descripción
US_001	Ingresar usuarios inválidos
US_002	Limpiar cache d navegador y volver a cargar la página para volver a página de autenticación
US_003	Consultar usuario con algunos datos personales vacíos
US_004	Consultar usuario sin imagen de perfil

Solum	
Proyecto:	**Solum**
Área:	**Campos**
Escenario #	Descripción
CA_001	Registrar un campo con algunos campos vacíos y seleccionar Guardar
CA_002	Ingresar todos los datos del campo a registrar y seleccionar Cancelar
CA_003	Ingresar la información requerida del campo a registrar y seleccionar establecer límites sin establecer puntos en mapa
CA_004	Registrar un campo con sólo 2 puntos en el mapa
CA_005	Registrar un campo con más de 5 puntos en el mapa
CA_006	Guardar un campo a registrar sin establecer puntos en el mapa
CA_007	Registrar un campo exitosamente y refrescar navegador

Solum	
Proyecto:	**Solum**
Área:	**Inicio**
Escenario #	Descripción
INI_001	Ingresar URL de una página de Solum que no sea /Inicio (por ejemplo, usuarios) sin haberse registrado previamente
INI_002	Seleccionar Ingresar sin introducir usuario ni password
INI_003	Navegar entre páginas utilizando el Back del navegador
INI_004	Seleccionar zoom in/out muchas veces seguidas hasta reducir/ampliar la escala al extremo
INI_005	Quitar todas las capas del mapa y verificar campos registrados
INI_006	Mostrar solo una capa del mapa y verificar campos registrados
INI_007	Minimizar/cambiar tamaño del navegador y validar comportamiento de componentes
INI_008	Seleccionar opción Salir y volver a loguearse con el mismo usuario

Solum	
Proyecto:	**Solum**
Área:	**Análisis de Suelo**
Escenario #	Descripción
AN_001	Registrar un análisis de suelo con algunos campos vacíos y seleccionar Guardar
AN_002	Ingresar todos los datos del análisis de suelo a registrar y seleccionar Cancelar
AN_003	Registrar un análisis de suelo exitosamente y refrescar navegador
AN_004	Seleccionar punto de análisis fuera del polígono del campo
AN_005	Seleccionar un punto de análisis dentro del polígono y luego seleccionar otro fuera
AN_006	Seleccionar un punto de análisis dentro del polígono y luego seleccionar otro (dentro del área)

Solum	
Proyecto:	**Solum**
Área:	**Puntos de referencia**
Escenario #	Descripción
PREF_001	Configurar usuario o servidor inválido y guardar la configuración
PREF_002	Realizar zoom in/out en la página de inicio de Solum Mobile
PREF_003	Registrar nuevo punto teniendo localización deshabilitada
PREF_004	Agregar mas de tres puntos de referencia para un mismo campo.
PREF_005	Eliminar todos los puntos de referencia para un campo y seleccionar Enviar a Solum
PREF_006	Modificar fecha de análisis de suelo
PREF_007	Seleccionar enviar a Solum y luego Cancelar envío
PREF_008	Minimizar aplicación y luego volver a abrirla
PREF_009	Registrar análisis de suelo en Solum Web a partir de puntos generados desde la aplicación mobile sin agregar nuevos puntos via web
PREF_010	Registrar campo en Solum Web a partir de puntos generados desde la aplicación mobile sin agregar nuevos puntos via web

Solum				Proyecto:	Solum
				Área:	Inicio
Pre-condición	URL de Solum es conocida y está disponible				
Prioridad	Alta				
Caso de Prueba #	**Título**	**Descripción**	**Pasos**	**Resultados Esperados**	**Estado**
Solum_Inicio_001	Acceder a Solum desde un navegador	Solum puede accederse desde cualquier navegador, preferentemente Chrome o Firefox actualizado. En caso de utilizar otro, es posible que se experimenten degradaciones mínimas de interfaz y/o performance.	1. Ingresar URL de Solum en el navegador	La página de autenticación de Solum es mostrada con los campos Usuario y Password y el botón Ingresar	Finalizado
Solum_Inicio_002	Ingresar al sistema con usuario valido	El ingreso al sistema es permitido luego de ingresar un usuario y contraseña válidos.	1. Ingresar usuario y contraseña válidos	La información ingresada es mostrada	Finalizado
			2. Seleccionar opción Ingresar	Página de inicio de Solum es mostrada (default) con todos sus componentes.	
Solum_Inicio_003	Mostrar componentes de página de inicio	La página de inicio de Solum es mostrada correctamente.	1. Autenticarse en el sistema	Ingreso autorizado.	Finalizado

			2. Verificar tarjetas de accesos directos son mostradas	4 tarjetas son mostradas en el panel superior de la pantalla: Campos, Análisis de Suelo, módulo IA y Usuarios	
			3. Verificar mapa con campos registrados es mostrado con capas y zoom in/out	Mapa con las capas seleccionadas y los campos registrados es mostrado. La opción zoom in/out también es mostrada dentro del área del mapa	
			4. Verificar barra de herramientas con los módulos disponibles es mostrada	Barra de herramientas es mostrada con el logo de Solum y las siguientes opciones: Inicio, Análisis de Suelo, Puntos de Referencia, Agente Inteligente, Gestión de Campos, Gestión de Usuarios y opción Salir	
Solum_Inicio_004	Acceder a Gestión de Usuarios desde acceso directo	Redireccionar a la página Gestión de Usuarios desde Inicio	1. Autenticarse en el sistema	Ingreso autorizado.	Finalizado

			2. Seleccionar opción Ver Detalles en la tarjeta de Usuarios	Página Gestión de Usuarios es mostrada con toda la información	
			3. Validar cantidad de usuarios mostrados en inicio	La cantidad de usuarios mostrados en la tarjeta de la página de inicio coincide con la cantidad de usuarios registrados	
Solum_Inicio_005	Acceder a Gestión de Campos desde acceso directo	Redireccionar a la página Gestión de Campos desde Inicio	1. Autenticarse en el sistema	Ingreso autorizado.	Finalizado
			2. Seleccionar opción Ver Detalles en la tarjeta de Campos	Página Gestión de Campos es mostrada con toda la información	
			3. Validar cantidad de Campos Registrados mostrados en inicio	La cantidad de Campos Registrados mostrados en la tarjeta de la página de inicio coincide con la cantidad de Campos registrados en el sistema	
Solum_Inicio_006	Acceder a Análisis desde acceso directo	Redireccionar a la página Análisis desde Inicio	1. Autenticarse en el sistema	Ingreso autorizado.	Finalizado

			2. Seleccionar opción Ver Detalles en la tarjeta de Análisis	Página Análisis es mostrada con toda la información	
			3. Validar cantidad de Análisis Registrados mostrados en inicio	La cantidad de Análisis Registrados mostrados en la tarjeta de la página de inicio coincide con la cantidad de Análisis registrados en el sistema	
Solum_Inicio_007	Acceder a módulo IA desde acceso directo	Redireccionar al módulo IA desde Inicio	1. Autenticarse en el sistema	Ingreso autorizado.	Finalizado
			2. Seleccionar opción Ver Detalles en la tarjeta de IA - reglas generadas	Página IA es mostrada con toda la información	
			3. Validar cantidad de reglas generadas mostradas en inicio	La cantidad de reglas generadas mostradas en la tarjeta de la página de inicio coincide con la cantidad de reglas generadas en el sistema	
Solum_Inicio_008	Visualizar mapa de modulo inicio	Mapa es mostrado con las capas seleccionadas y el zoom=100%	1. Autenticarse en el sistema	Ingreso autorizado.	Finalizado

			2. Seleccionar zoom in/out (opciones + y -) dentro del área del mapa	Mapa es mostrado con la información de todas las capas y los campos registrados en la escala seleccionada (ampliada o reducida)	
			3. Marcar/desmarcar algunas capas del menú desplegable Capas en el mapa	Mapa es mostrado con la información de las capas seleccionadas y los campos registrados	

Solum				Proyecto:	Solum
				Área:	Campos
Pre-condición	Usuario válido esta logueado en el sistema. La página Gestión de Campos fue seleccionada desde Inicio				
Prioridad	Media				
Caso de Prueba #	**Título**	**Descripción**	**Pasos**	**Resultados Esperados**	**Estado**
Solum_Campo_001	Registrar nuevo campo	Registrar un nuevo campo en el sistema ingresando puntos en el mapa y datos específicos.	1. Seleccionar opción Agregar Campo (+)	Página de registro de nuevos campos es mostrada con los siguientes componentes: nombre, lista desplegable de responsables, opciones de cultivos asignables, opción establecer límites, guardar, cancelar y mapa para ingresar área del campo a registrar	Finalizado
			2. Ingresar nombre/alias del campo a ingresar, responsable y cultivo/s asignable/s	Información ingresada es mostrada en la página	

			3. Marcar al menos 3 puntos en el mapa que representen los vértices del área del campo. (para posicionar el mapa rápidamente pueden utilizarse los parámetros altitud y longitud debajo del mapa)	Puntos sombreados son mostrados por cada click realizado en el mapa	
			4. Seleccionar opción Establecer Límites	Los puntos se unen y el área del campo es sombreada.	
			5. Seleccionar opción Guardar	El campo es registrado y el usuario es redireccionado a la página Campos, donde el campo recientemente ingresado es incluido en la lista y en el mapa.	
Solum_Campo_002	Buscar campo registrado	Consultar un campo ya registrado desde el campo Buscar o desde la lista de registros	1. Ingresar caracteres que no están incluidos en ningún nombre de campo	La lista de campos registrados es filtrada mientras se ingresan caracteres mostrándose finalmente vacía ya que el campo no existe	Finalizado

			2. Ingresar el nombre de campo ya registrado	La lista de campos registrados es filtrada mientras se ingresan caracteres mostrando finalmente un solo resultado correspondiente al campo buscado	
			3. Borrar el nombre del campo búsqueda	Todos los campos registrados son mostrados nuevamente en la lista	
Solum_Campo_003	Localizar campo en mapa	Localizar campos registrados en el mapa utilizando la opción Vista de la lista de Campos Registrados o bien seleccionando uno directamente desde el mapa	1. Seleccionar la opción Vista de un campo registrado (icono: ojo)	El mapa es relocalizado mostrando el campo seleccionado con un área sombreada y la información del mismo	Finalizado
			2. Cerrar el recuadro de información del campo	El componente que muestra información es cerrado pero el mapa continúa mostrando el campo	

			3. Seleccionar otro campo desde el mapa haciendo click en un área sombreada	El mapa es relocalizado mostrando el campo seleccionado con un área sombreada y la información del mismo	

Solum				**Proyecto:**	Solum
				Área:	Usuarios
Pre-condición	Usuario válido esta logueado en el sistema. La página Usuarios fue seleccionada desde Inicio				
Prioridad	Baja				
Caso de Prueba #	**Título**	**Descripción**	**Pasos**	**Resultados Esperados**	**Estado**
Solum_Usuarios_001	Consultar usuarios registrados	Página Usuarios es mostrada con toda la información	1. Abrir módulo Usuarios desde Inicio	Página Usuarios es mostrada con los siguientes componentes: campo Buscar, lista de los usuarios registrados (con imagen de perfil, nombre y apellido, usuario y opción Vista de cada uno de ellos)	Finalizado

			2. Validar lista de usuarios registrados es mostrada	Todos los usuarios registrados son mostrados con la correspondiente información	
			3. Seleccionar opción Vista de un usuario particular	La lista de usuarios es desplazada hacia abajo y en primer lugar se muestra: la imagen ampliada del usuario seleccionado, nombre de usuario, password, nombre y apellido, email, dirección y teléfono.	
Solum_Usuarios_002	Buscar un usuario registrado	Consultar un usuario ya registrado desde el campo Buscar	1. Ingresar nombre de usuario en el campo búsqueda	La lista de usuarios es filtrada mientras se ingresan los caracteres de búsqueda mostrando finalmente un solo resultado correspondiente al usuario buscado	Finalizado

			2. Borrar el nombre ingresado en el campo búsqueda	Todos los usuarios registrados son mostrados con la correspondiente información	
			3. Ingresar un nombre inválido de usuario en el campo búsqueda	La lista de usuarios es filtrada mientras se ingresan los caracteres de búsqueda mostrando finalmente cero resultados ya que el usuario buscado no existe	

Solum				Proyecto:	Solum
				Área:	IA
Pre-condición	Usuario válido esta logueado en el sistema. La página IA fue seleccionada desde Inicio				
Prioridad	Alta				
Caso de Prueba #	**Título**	**Descripción**	**Pasos**	**Resultados Esperados**	**Estado**
Solum_IA_001	Interfaz Agente Solum	Interfaz Agente Solum - Test Unitario/Integración	1. Inicializar Kernel Soar (dependencias requeridas previamente cargadas)	Kernel Inicializado correctamente	Finalizado
			2. Inicializar Agente Soar (path donde se encuentran las producciones Soar)	Agente iniciado correctamente	
			3. Cargar input link (campo que incluye un análisis de suelo con valores de azufre, fósforo y nitrógeno. Se persigue el objetivo de Sustentabilidad)	Input link correctamente cargado	
			4. Ejecutar debugger (path donde se encuentra el debugger de Soar)	Debugger inicializado correctamente	
			5. Verificar consistencia de red semántica manualmente	Red semántica correctamente cargada	
			6. Ejecutar agente en modo debug	Operadores de siembra correctamente propuestos y aplicados	

			7. Verificar manualmente el contenido de output link	Output link produce consistentemente un plan de siembra que alcanza el objetivo	
			8. Repetir pasos 2 a 7	Output link produce consistentemente un plan de siembra que alcanza el objetivo	
			9. Realizar ejecución total iterativa del Agente Solum	Se alcanzó el objetivo de Sustentabilidad o bien se llegó a la máxima cantidad posible de pasos. Se deben obtener un total de 100 planes de siembra que hayan alcanzado el objetivo.	
			10. Verificar aprendizaje	Las ejecuciones iterativas del agente muestran convergencia hacia una curva de aprendizaje.	

Solum_IA_002	Gestor Mapeo Solum	Gestor Mapeo Solum - Prueba Unitaria/Integración	1. Instanciar objetos de dominio (campo que incluye un análisis de suelo con valores de azufre, fósforo y nitrógeno. Se persigue el objetivo de Sustentabilidad)	Objetos de dominio inicializados correctamente	Finalizado
			2. Transformar objetos de dominio a identificadores y elementos de la memoria de trabajo (WME)	Elementos mapeados correctamente	
			3. Cargar input link	Input link correctamente cargado	
			4. Ejecutar debugger (path donde se encuentra el debugger de Soar)	Debugger inicializado correctamente	
			5. Verificar consistencia de red semántica manualmente	Red semántica correctamente cargada	
			6. Ejecutar agente en modo debug	Operadores de siembra correctamente propuestos y aplicados	
			7. Realizar ejecución de elaboraciones internas del agente Soar	Los elementos del input link se elaboran correctamente en una estructura de elementos de memoria de trabajo	

			8. Verificar manualmente el contenido del output link	Output link produce consistentemente un plan de siembra que alcanza el objetivo	
			9. Inicializar Agente Soar	Agente iniciado correctamente	
			10. Repetir pasos 3, 4, 5, 6 y 8	Output link produce consistentemente un plan de siembra que alcanza el objetivo	
			11. Realizar ejecución total iterativa del Agente Solum	Se alcanzó el objetivo de Sustentabilidad o bien se llegó a la máxima cantidad posible de pasos. Se deben obtener un total de 100 planes de siembra que hayan alcanzado el objetivo.	
			12. Verificar aprendizaje	Las ejecuciones iterativas del agente muestran convergencia hacia una curva de aprendizaje.	

			13. Realizar ejecución del agente en modo test	El agente no realiza aprendizaje en esta ejecución. Obtiene uno de los mejores planes de siembra que alcanzaron el objetivo.	
Solum_IA_003	Validar resumen de entrenamiento	Página IA contiene la información resumida de las ejecuciones del módulo IA	1. Abrir página IA desde Inicio	Página IA es mostrada con los siguientes submódulos: Agente Inteligente, Red Neuronal, Estado de los Campos, Estado de los nutrientes	Finalizado
			2. Validar información mostrada del Agente Inteligente	Submódulo Agente Inteligente muestra: último entrenamiento del agente, tiempo de la última ejecución, cantidad de reglas generadas, ritmo de evolución de los nutrientes y gráfico de reglas generadas por ejecución	

			3. Validar información mostrada de la Red Neuronal	Submódulo Red Neuronal muestra: último entrenamiento de la red, error en el entrenamiento, tiempo de ejecución y gráfico de desviaciones	
			4. Validar información mostrada de estados de los campos	Submódulo Estado de los Campos muestra: adhesión a los planes de siembra, salinidad y rendimiento promedio	
			5. Validar información mostrada de estados de nutrientes	Submódulo Estado de Nutrientes muestra: nitrógeno, fósforo y azufre	

Solum				Proyecto:	Solum
				Módulo:	Análisis
Pre-condición	Usuario válido esta logueado en el sistema. La página Gestión de Análisis de suelo fue seleccionada desde Inicio				
Prioridad	Alta				
Caso de Prueba #	**Título**	**Descripción**	**Pasos**	**Resultados Esperados**	**Estado**
Solum_Analisis_001	Registrar nuevo análisis de suelo	Registrar un nuevo análisis de suelo en el sistema ingresando datos requeridos.	1. Seleccionar opción Agregar Análisis de suelo (+)	Página de registro de nuevos análisis de suelos es mostrada con los siguientes componentes: campo, latitud, longitud, cec, conductividad, fecha, porcentaje materia orgánica, porcentaje de humedad, laboratorio, color, texturas, profundidad, temperatura media, tipo de suelo y nutrientes. Mapa para ingresar punto de análisis es mostrado.	Finalizado

			2. Ingresar campo.	Información ingresada es mostrada en la página. Mapa es localizado en el área del campo seleccionado, mostrando al mismo sombreado.	
			3. Seleccionar punto donde se realizó el análisis en el mapa.	Punto seleccionado es marcado en el mapa. Latitud y altitud son autocompletados con datos del punto seleccionado.	

			3. Ingresar capacidad de intercambio catiónico, conductividad eléctrica, fecha, porcentaje de materia orgánica, porcentaje de humedad, laboratorio donde se realizó el análisis, color, texturas (arcilla, arena, limo), profundidad, temperatura media, tipo de suelo, nutrientes (fósforo, nitrógeno y azufre con sus cantidades y métodos de extracción)	Información ingresada es mostrada en la página.	
			5. Seleccionar opción Guardar	El análisis de suelo es guardado correctamente y agregado a la lista de registrados.	

Solum				Proyecto:	Solum
				Módulo:	Puntos de Referencia
Pre-condición	La aplicación Solum ha sido instalada en el dispositivo. Usuario válido esta logueado en la aplicación.				
Prioridad	Media				
Caso de Prueba #	**Título**	**Descripción**	**Pasos**	**Resultados Esperados**	
Pto_Ref_001	Configurar usuario	Configurar otro usuario en la aplicación mobile.	1. Seleccionar opción Configuración en la aplicación	Sección Configuración es mostrada con usuario, contraseña y servidor por defecto	Finalizado
			2. Cambiar usuario y contraseña	Información ingresada es mostrada en la página.	
			3. Seleccionar Guardar	El usuario es guardado con éxito	
			4. Ir a otra sección de la aplicación y regresar a Configuración	La configuración reciente persiste	
			5. Cerrar la aplicación y volver a abrirla, luego ir a Configuración.	La configuración reciente persiste	
Pto_Ref_002	Generar punto de referencia para Análisis de suelo	Generar un punto de referencia para registrar un análisis de suelo desde la web luego de enviar dicha información desde el dispositivo.	1. Seleccionar opción Análisis de Suelo en la aplicación	Sección Análisis de Suelo es mostrada con fecha (no modificable), descripción, latitud, longitud, opción	Finalizado

				calcular posición y enviar a Solum.	
			2. Seleccionar opción Calcular Posición	Icono de carga es mostrado hasta que la posición es calculada y mostrada en latitud y longitud.	
			3. Seleccionar Enviar a Solum	Pop-up es mostrado para confirmar el envio de la información.	
			4. Seleccionar Aceptar	Los datos son enviados a Solum Web.	
			5. Loguearse en Solum web con el mismo usuario y verificar que el punto enviado desde el dispositivo es mostrado en la sección Puntos de Referencia > Análisis d suelos	El punto fue exitosamente enviado a Solum Web y es mostrado en la sección Puntos de Referencia con la opción de vista, registrar análisis de suelo y eliminar.	

Pto_Ref_003	Generar puntos de referencia para Campo	Generar puntos de referencia para registrar un campo desde la web luego de enviar dicha información desde el dispositivo.	1. Seleccionar opción Campo en la aplicación	Sección Coordenadas de Campo es mostrada con: descripción y opciones Nuevo Punto y Enviar a Solum.	Finalizado
			2. Seleccionar opción Nuevo punto	Icono de carga es mostrado hasta que la posición es calculada y mostrada en una nueva página con las opciones Agregar y Cancelar y los datos latitud y longitud.	
			3. Seleccionar opción Agregar	Página Coordenadas de Campo es mostrada nuevamente con el punto registrado mostrado debajo de la descripción del campo.	

			4. Repetir pasos 2 y 3 tres veces.	Los 4 puntos registrados son mostrados en la página Coordenadas de Campó con sus coordenadas y las opciones de Eliminar puntos.	
			5. Seleccionar Enviar a Solum	Los datos son enviados a Solum Web	
			5. Loguearse en Solum web con el mismo usuario y verificar que los puntos enviados desde el dispositivo son mostrados en la sección Puntos de Referencia > Campos	Los puntos fueron exitosamente enviados a Solum Web y son mostrados en la sección Puntos de Referencia con la opción de vista, registrar campo y eliminar.	

Pto_Ref_004	Registrar Análisis de suelo en base a punto de referencia	Registrar análisis de suelo en Solum Web en base a un punto de referencia enviado desde un dispositivo.	1. Ingresar a sección Puntos de referencia (un punto de referencia de análisis de suelo ya fue enviado a la web)	El punto ingresado previamente es mostrado en la lista.	Finalizado
			2. Seleccionar opción Registrar Análisis	El usuario es redireccionado a la página de registro de análisis de suelo donde se muestran todos los valores a ingresar y la localización ya cargada (punto de referencia)	
			3. Ingresar datos requeridos y seleccionar Guardar	El análisis de suelo es correctamente registrado en el sistema.	

Pto_Ref_005	Registrar Campo en base a puntos de referencia	Registrar campo en Solum Web en base a puntos de referencia enviados desde un dispositivo.	1. Ingresar a sección Puntos de referencia (almenos tres puntos de referencia de campo ya fueron enviados a la web)	Los puntos ingresados previamente son mostrados en la lista.	Finalizado
			2. Seleccionar opción Registrar Campo	El usuario es redireccionado a la página de registro de campos donde se muestran todos los valores a ingresar y los puntos del polígono ya ingresados (puntos de referencia)	
			3. Ingresar datos requeridos, establecer límites y seleccionar guardar.	El campo es generado correctamente y registrado en el sistema.	

A continuación se listan las ejecuciones de test cases realizadas durante el desarrollo de Solum.

Solum	
Fecha	15/02/2015
Cantidad TC ejecutados	4
Tiempo testing exploratorio	2 horas
Áreas involucradas	Usuarios
Revisión #	12
Responsable de resultados	Luz Cantoni
Responsable de ejecución	Luz Cantoni

Área	Total TC	Exitosos	Fallados	Bloqueados	No ejecutados	Porcentaje éxito por área
Inicio	8	2	0	0	6	25%
IA	3	0	0	0	3	-
Análisis de suelo	1	0	0	0	1	-
Campos	3	0	0	0	3	-
Usuarios	2	0	0	0	2	-
				Porcentaje éxito de versión		100%

Solum	
Fecha	07/03/2015
Cantidad TC ejecutados	2
Tiempo testing exploratorio	3 horas
Áreas involucradas	IA
Revisión #	18
Responsable de resultados	Luz Cantoni
Responsable de ejecución	Luz Cantoni

Área	Total TC	Exitosos	Fallados	Bloqueados	No ejecutados	Porcentaje éxito por área
Inicio	8	0	0	0	8	-
IA	3	1	1	0	1	33%
Análisis de suelo	1	0	0	0	1	-
Campos	3	0	0	0	3	-
Usuarios	2	0	0	0	2	-
				Porcentaje éxito de versión		50%

Solum	
Fecha	15/03/2015
Cantidad TC ejecutados	10
Tiempo testing exploratorio	4 horas
Áreas involucradas	IA, Campos, Usuarios, Inicio
Revisión #	20
Responsable de resultados	Luz Cantoni
Responsable de ejecución	Luz Cantoni

Área	Total TC	Exitosos	Fallados	Bloqueados	No ejecutados	Porcentaje éxito por área
Inicio	8	3	0	0	5	38%
IA	3	2	1	0	0	67%
Análisis de suelo	1	0	0	0	1	-
Campos	3	1	1	0	1	33%
Usuarios	2	1	1	0	0	50%
				Porcentaje éxito de versión		70%

Solum	
Fecha	02/04/2015
Cantidad TC ejecutados	13
Tiempo testing exploratorio	2 horas
Áreas involucradas	Campos, Usuarios, Inicio
Revisión #	27
Responsable de resultados	Luz Cantoni
Responsable de ejecución	Luz Cantoni

Área	Total TC	Exitosos	Fallados	Bloqueados	No ejecutados	Porcentaje éxito por área
Inicio	8	5	3	0	0	63%
IA	3	0	0	0	3	-
Análisis de suelo	1	0	0	0	1	-
Campos	3	2	1	0	0	67%
Usuarios	2	1	1	0	0	50%
				Porcentaje éxito de versión		62%

Solum	
Fecha	21/04/2015
Cantidad TC ejecutados	17
Tiempo testing exploratorio	7 horas
Áreas involucradas	Campos, Usuarios, Inicio, IA, Análisis de suelo
Revisión #	35
Responsable de resultados	Luz Cantoni
Responsable de ejecución	Luz Cantoni

Área	Total TC	Exitosos	Fallados	Bloqueados	No ejecutados	Porcentaje éxito por área
Inicio	8	3	5	0	0	38%
IA	3	2	1	0	0	67%
Análisis de suelo	1	0	1	0	0	0%
Campos	3	1	2	0	0	33%
Usuarios	2	2	0	0	0	100%
				Porcentaje éxito de versión		47%

Solum	
Fecha	15/05/2015
Cantidad TC ejecutados	17
Tiempo testing exploratorio	7 horas
Áreas involucradas	Campos, Usuarios, Inicio, IA, Análisis de suelo
Revisión #	38
Responsable de resultados	Luz Cantoni
Responsable de ejecución	Luz Cantoni

Área	Total TC	Exitosos	Fallados	Bloqueados	No ejecutados	Porcentaje éxito por área
Inicio	8	5	3	0	0	63%
IA	3	1	2	0	0	33%
Análisis de suelo	1	0	1	0	0	0%
Campos	3	2	1	0	0	67%
Usuarios	2	2	0	0	0	100%
				Porcentaje éxito de versión		59%

Solum	
Fecha	23/05/2015
Cantidad TC ejecutados	17
Tiempo testing exploratorio	5 horas
Áreas involucradas	Campos, Usuarios, Inicio, IA, Análisis de suelo
Revisión #	40
Responsable de resultados	Luz Cantoni
Responsable de ejecución	Luz Cantoni

Área	Total TC	Exitosos	Fallados	Bloqueados	No ejecutados	Porcentaje éxito por área
Inicio	8	6	2	0	0	75%
IA	3	1	2	0	0	33%
Análisis de suelo	1	1	0	0	0	100%
Campos	3	2	1	0	0	67%
Usuarios	2	2	0	0	0	100%
				Porcentaje éxito de versión		71%

Solum	
Fecha	02/06/2015
Cantidad TC ejecutados	17
Tiempo testing exploratorio	9 horas
Áreas involucradas	Campos, Usuarios, Inicio, IA, Análisis de suelo
Revisión #	42
Responsable de resultados	Luz Cantoni
Responsable de ejecución	Luz Cantoni

Área	Total TC	Exitosos	Fallados	Bloqueados	No ejecutados	Porcentaje éxito por área
Inicio	8	6	2	0	0	75%
IA	3	2	1	0	0	67%
Análisis de suelo	1	1	0	0	0	100%
Puntos de referencia	5	0	0	0	5	-
Campos	3	3	0	0	0	100%
Usuarios	2	1	1	0	0	50%
				Porcentaje éxito de versión		76%

Solum	
Fecha	28/06/2015
Cantidad TC ejecutados	17
Tiempo testing exploratorio	7 horas
Áreas involucradas	Campos, Usuarios, Inicio, IA, Análisis de suelo
Revisión #	43
Responsable de resultados	Luz Cantoni
Responsable de ejecución	Luz Cantoni

Área	Total TC	Exitosos	Fallados	Bloqueados	No ejecutados	Porcentaje éxito por área
Inicio	8	6	2	0	0	75%
IA	3	3	0	0	0	100%
Análisis de suelo	1	0	1	0	0	0%
Puntos de referencia	5	3	2	0	0	60%
Campos	3	3	0	0	0	100%
Usuarios	2	1	1	0	0	50%
				Porcentaje éxito de versión		73%

SOLUM	
Fecha	01/07/2015
Cantidad TC ejecutados	17
Tiempo testing exploratorio	10 horas
Áreas involucradas	Campos, Usuarios, Inicio, IA, Análisis de suelo
Revisión #	44
Responsable de resultados	Luz Cantoni
Responsable de ejecución	Luz Cantoni

Área	Total TC	Exitosos	Fallados	Bloqueados	No ejecutados	Porcentaje éxito por área
Inicio	8	7	1	0	0	88%
IA	3	3	0	0	0	100%
Análisis de suelo	1	1	0	0	0	100%
Puntos de referencia	5	4	1	0	0	80%
Campos	3	3	0	0	0	100%
Usuarios	2	2	0	0	0	100%
				Porcentaje éxito de versión		91%

Solum	
Fecha	02/07/2015
Cantidad TC ejecutados	17
Tiempo testing exploratorio	3 horas
Áreas involucradas	Campos, Usuarios, Inicio, IA, Análisis de suelo
Revisión #	45
Responsable de resultados	Luz Cantoni
Responsable de ejecución	Luz Cantoni

Área	Total TC	Exitosos	Fallados	Bloqueados	No ejecutados	Porcentaje éxito por área
Inicio	8	8	0	0	0	100%
IA	3	3	0	0	0	100%
Análisis de suelo	1	1	0	0	0	100%
Puntos de referencia	5	5	0	0	0	100%
Campos	3	3	0	0	0	100%
Usuarios	2	2	0	0	0	100%
				Porcentaje éxito de versión		100%

Nota: Nuevos casos de prueba serán agregados en la próxima ejecución para contemplar nuevas funcionalidades: plan de siembra y aplicación Mobile.

A continuación se listan algunos de los defectos encontrados durante el desarrollo del sistema.

Fecha	Defecto	Estado	Reportado por	Resuelto por
09-mar	Error en Right-Hand-Side en operador apply*cosecha	Cerrado Resuelto	Juan Barsce	Juan Barsce
11-mar	Elaboraciones infinitas en Soar tras inicializar el estado. Agente atrapado en un bucle infinito	Cerrado Resuelto	Juan Barsce	Juan Barsce
23-mar	_firstload no es cargado correctamente en el Agente	Cerrado Resuelto	Juan Barsce	Juan Barsce
25-mar	Operator-tie, Soar no puede decidir entre sus alternativas de siembra	Cerrado Resuelto	Luz Cantoni	Juan Barsce
01-abr	Dependencias incorrectamente cargadas	Cerrado Resuelto	Gonzalo Girardi	Ignacio Favro
14-abr	Parsing incorrecto de fechas entre Soar y Java	Cerrado Resuelto	Juan Barsce	Gonzalo Girardi
14-abr	Nombre del proyecto incompatible con el nombre de la carpeta	Cerrado Resuelto	Ignacio Favro	Gonzalo Girardi
14-abr	Puntos seleccionados en mapa para registrar campo no son guardados	Cerrado Resuelto	Luz Cantoni	Ignacio Favro
19-abr	Error en las escalas de Soar, existe un rango de valores que no está en ninguna escala	Cerrado Resuelto	Juan Barsce	Ignacio Favro
19-abr	No se muestra latitud y longitud para registrar campos y análisis de suelo	Cerrado Resuelto	Luz Cantoni	Ignacio Favro
19-abr	Operador fertilizar agrega fertilizante adicional a aquellos nutrientes aun estando en nivel alto	Cerrado Resuelto	Luz Cantoni	Gonzalo Girardi
27-abr	Operador calcular-reward-nutriente ocasionalmente asigna reward antes que los cálculos de reward de todos los nutrientes estén finalizados	Cerrado Resuelto	Juan Barsce	Juan Barsce

04-may	Error en la compilación por diferencia de versión de Java	Cerrado Resuelto	Luz Cantoni	Gonzalo Girardi
04-may	Campo inicio no se localiza según campo seleccionado.	Cerrado Resuelto	Luz Cantoni	Ignacio Favro
15-may	Faltan datos de entrada en el registro de análisis de suelo	Cerrado Resuelto	Luz Cantoni	Ignacio Favro
26-may	Componentes de Inicio solapados	Cerrado Resuelto	Luz Cantoni	Ignacio Favro
01-jul	Puntos de referencia no son enviados a Solum Web	Cerrado Resuelto	Luz Cantoni	Ignacio Favro

Universidad Tecnológica Nacional - Facultad Regional Villa María

Ingeniería en Sistemas de Información

Cátedra: Proyecto Final

Solum – Tomo II

Integrantes Grupo 1:

Barsce, Juan Cruz (Legajo Nº 7532)

Cantoni, M. Luz (Legajo Nº 7827)

Favro, Ignacio (Legajo N° 7163)

Girardi, Gonzalo (Legajo N° 6547)

Docentes:

Zohil, Julio

Villafañe, Christian

Fecha de regularización: 2011

Contenido

Capítulo I – Definiciones del Dominio de los Suelos

1. Introducción

El presente proyecto está centrado en los campos que realizan agricultura, particularmente en la fertilización y el mantenimiento de los suelos. El presente documento, por su parte, sirve a modo de introducción sobre el complejo dominio de los campos, para poder realizar el modelado a partir de él. A continuación se definen los términos relativos al mencionado dominio.

1.1. Campo

Un campo es un espacio abierto rural utilizado para la agricultura, ganadería y otros fines agropecuarios. El mismo se encuentra inmerso en un entorno, el cual se compone de variados elementos, entre ellos, el suelo.

1.2. Suelo

Es la capa más superficial de la corteza terrestre, que resulta de la descomposición de las rocas por los cambios bruscos de temperatura y por la acción del agua, del viento y de los seres vivos.

El proceso mediante el cual los fragmentos de roca se hacen cada vez más pequeños, se disuelven o van a formar nuevos compuestos, se conoce con el nombre de meteorización.

Los productos rocosos de la meteorización se mezclan con el aire, agua y restos orgánicos provenientes de plantas y animales para formar suelos. Luego el suelo puede ser considerado como el producto de la interacción entre la litosfera, la atmósfera, la hidrosfera y la biosfera. Este proceso tarda muchos años, razón por la cual los suelos son considerados recursos naturales no renovables.

1.3. Parámetros del suelo

El suelo presenta los siguientes parámetros principales:

- Salinidad. Es el contenido salino del suelo. La sal es un elemento natural de los suelos y agua. El proceso de incremento del contenido de sal se conoce como salinización, el cual puede ocurrir naturalmente por eventos como la acción capilar de napas con sal, o artificialmente a través del uso de potasio como fertilizante. A medida que la salinidad se incrementa, la misma puede producir una degradación del suelo y de la vegetación. La salinidad de un suelo se mide haciendo un análisis de conductividad. Su unidad es el deci-Siemens por metro (dS/m). En base a las mediciones efectuadas, los suelos se clasifican de la siguiente manera:

Clase de salinidad del suelo	Conductividad del Extracto Saturado (dS/m)	Efecto en los cultivos
No salino	0 – 2	Depreciable
Levemente salino	2 – 4	Pueden decrementar los rendimientos en cultivos sensibles
Moderadamente salino	4 – 8	Puede decrementar los rendimientos en varios cultivos
Altamente salino	8 – 16	Sólo aquellos cultivos tolerantes rinden aceptablemente
Muy altamente salino	> 16	Sólo unos pocos cultivos tolerantes rinden aceptablemente

- Humedad. Es la cantidad de agua contenida en el suelo. Cuando el suelo se encuentra demasiado seco, la transpiración de la planta es reducida para poder conservar mayor cantidad de agua. Por detrás del punto de marchitez permanente, la planta pierde la capacidad de extraer agua. Esto es lo que se busca evitar a través del riego. El estancamiento del agua acumulada por encima de la capacidad de absorción del suelo puede causar problemas de oxigenación y distribución de nutrientes, y dejar el suelo más propenso a enfermedades y malezas.
  La humedad se expresa en un ratio que va de 0 (completamente seco) hasta el valor de la porosidad del material al saturarse.

- Temperatura. La temperatura del suelo regula la germinación de semillas, el crecimiento de las raíces y plantas, la absorción del agua y la disponibilidad de nutrientes. Depende del ratio entre la energía absorbida y la energía perdida. Varía de -20ºC hasta 60º, con fluctuaciones considerables en los niveles de menor profundidad del suelo. Entre los aspectos que afectan la temperatura del suelo se encuentran el color del suelo (suelos blancos absorben más calor que los suelos oscuros) o incrementos en el contenido de agua.

- Nutrientes. Un total de 16 nutrientes son esenciales para el crecimiento y la reproducción de las plantas: Macronutrientes: carbono, hidrógeno, oxígeno, nitrógeno, fósforo, potasio, azufre, calcio y magnesio. Micronutrientes: hierro, boro, magnesio, cobre, zinc, molibdeno y cloro.
  El hidrógeno, junto con el carbono y el oxígeno, son suministrados por el agua y dióxido de carbono, mientras que los restantes derivan de los componentes minerales (que pasan a formar parte de la materia orgánica, también llamada humus) del suelo.
  El elemento de mayor importancia que las plantas obtienen del suelo es el nitrógeno (N). El mismo raramente escasea en el suelo, pero no puede ser tomado por las raíces en su forma cruda, puesto que antes debe ser descompuesto por microorganismos. Es uno de los elementos más fluctuantes en la composición del suelo.
  Otro de los elementos más críticos del suelo es el fósforo (P). El cual proviene mayormente de la materia orgánica. Se encuentra mayormente disponible para su absorción en pH de 6.5. Debe notarse que la aplicación de fertilizantes solubles (con P) tenderá a generar deficiencias en zinc, mientas que la aplicación de fertilizantes con zinc puede inmovilizar el fósforo, limitando su absorción.

- Materia orgánica. Es un compuesto de carbón situado en la superficie del suelo que incluye todo el material orgánico o inorgánico de animales o vegetales. Su descomposición se conoce como humus. Una buena composición de materia orgánica es crucial para el buen crecimiento de las plantas, por lo que resulta vital reponer nutrientes una vez realizada cada cosecha. La producción de materia orgánica dependen altamente del clima. Las bajas temperaturas son ideales para la acumulación de materia orgánica, ya que las mismas no resultan propensas para las bacterias que realizan la descomposición. Inversamente, altas temperaturas y excesivas lluvias

resultan ideales para los microbios, lo que puede resultar en una merma muy considerable del humus. La materia orgánica se mide como un porcentaje de la muestra de suelo analizada.

- Textura. La textura es la proporción en la que se distribuyen las partículas que componen el suelo. Permite conocer cuánta agua puede absorber el suelo y cuán fácil resulta para las plantas acceder a ella. Según su tamaño, porosidad y absorción de agua, se clasifican en limo, arcilla y arena. La medida que determina esto es la dimensión de la partícula elemental (en mm). Ver (5) para más información.

- pH. Es una medida de la acidez o alcalinidad de un suelo. pH se define como el logaritmo negativo (en base 10) de la actividad de los iones de hidronio en una solución.
  Entre los factores que contribuyen a la acidez del suelo se encuentran las lluvias, el uso de fertilizantes con amonio (los cuales contienen altas cantidades de nitrógeno) o la descomposición de materia orgánica. Aquellas plantas que crecen en suelos ácidos son altamente propensas a la intoxicación. Una forma de incrementar el pH es por medio del uso de un tipo particular de cal.
  Por su parte, los suelos con alta alcalinidad presentan baja capacidad de infiltración (proceso en el cual el agua entra en los suelos), haciendo que el agua de lluvia sea propensa a estancarse, limitando la agricultura a cultivos tolerantes al anegamiento (como el arroz). Una forma de reducir el pH es utilizando fertilizantes con azufre.
  Concretamente, los valores se clasifican de la siguiente forma:

Denominación	Rango de pH
Ultra ácido	< 3,5
Extremadamente ácido	3,5 – 4,4
Muy fuertemente ácido	4,5 – 5,0
Fuertemente ácido	5,1 – 5,5
Moderadamente ácido	5,6 – 6,0
Levemente ácido	6,1 – 6,5
Neutral	6,6 – 7,3
Levemente alcalino	7,4 – 7,8
Moderadamente alcalino	7,9 – 8,4
Fuertemente alcalino	8,5 – 9,0
Muy fuertemente alcalino	> 9,0

La tolerancia individual de los distintos cultivos puede observarse en (6).

- Color. El color del suelo está determinado por sus minerales y materia orgánica. Entre los colores posibles se encuentran el Amarillo, Rojo, Marrón oscuro, Amarillo rojizo, Rojo oscuro, Gris oscuro, Negro, Negro metálico, Amarillo pálido, Marrón muy pálido y Gris claro.
- Capacidad de intercambio catiónico (CEC). Es la máxima cantidad de cationes de cualquier clase que un suelo puede almacenar dado unos pH, disponibles para ser intercambiados con la solución del suelo. Los mismos representan una medida de la fertilidad y capacidad de retención de nutrientes del suelo; además de la capacidad de proteger el agua de la contaminación de cationes.
  Un catión es una partícula positivamente cargada (una partícula negativamente cargada se denomina anión). Existen dos tipos: los cationes formadores de ácidos y los cationes formadores de bases. Un suelo con altos niveles de cationes de aluminio e hidrógeno es ácido, con bajos niveles de pH. Por otra parte, un suelo con altos niveles de calcio, magnesio, potasio y sodio es alcalino, puesto que los mismos son formadores de bases.
  Tanto la materia orgánica como las partículas de arcilla tienen partes negativamente cargadas, de modo que atraen y fijan los elementos positivamente cargados.
  Los nutrientes positivamente cargados más importantes son el calcio, el magnesio y el potasio.

Otros factores que inciden sobre el suelo son los siguientes, enunciados brevemente (debido a su baja consideración en la simulación del agente inteligente en el sistema):

- Aire.
  - Temperatura.
  - Humedad.
- Agua.
  - Temperatura.
  - Salinidad.
- Lluvia.
  - Precipitaciones.
- Solar.

    - Energía solar.
- Viento.
    - Velocidad.
    - Dirección.

2. Etapas de la agricultura

2.1. Labranza

Es la agitación mecánica del suelo para prepararlo para la cultivación. Su objetivo es remover y airear el suelo, haciendo más fácil la plantación del cultivo, ayudando a la mezcla de materia orgánica y destruyendo malezas. Debe notarse que, dependiendo de la intensidad de la labranza, la misma puede secar el suelo y generar pérdida de nutrientes, decrementando el ratio de infiltración de agua del suelo (pudiendo compactarlo) y pudiendo dejarlo en estado vulnerable a las enfermedades. Existen tres tipos de labranza:

- Reducida, que busca dejar entre un 15 y 30% de cobertura de residuos en el suelo por hectárea.
- Intensiva, que deja menos de 15% de residuos.
- De conservación, que deja al menos un 30% de residuos en la superficie del suelo. Los mismos disminuyen el movimiento del agua, lo que reduce la erosión del suelo. Adicionalmente reduce la compactación y el uso de combustible utilizado para realizar la labranza. Debe notarse que este tipo de labranza generará un retraso en el calentamiento del suelo debido a la exposición reducida de la tierra oscura a la luz del sol, lo que puede producir un retraso en la plantación de algunas semillas.

2.2. Siembra

Es el proceso de colocar semillas, con el objetivo de que germinen y desarrollen plantas. Se realiza normalmente luego de realizar una labranza en el suelo. El tipo de siembra más comúnmente utilizado es la siembra a campo abierto, donde los campos son preparados y se los deja abiertos antes de ser sembrados. La semilla se deposita en el terreno sin ser cubierta durante la germinación, quedando expuesta a las condiciones climáticas. La fecha y condiciones de siembra son altamente dependientes de cada cultivo.

2.3. Fertilización

Es la forma más directa de mejorar el rendimiento de los campos. Se realiza aproximadamente una vez cada dos años, normalmente en algún momento entre la primavera y el verano, preferentemente cuando no haya cultivos. Un fertilizante es un compuesto de origen natural o sintético que se le agrega al suelo para suministrarle uno o más nutrientes esenciales que las plantas necesitan para su crecimiento. Los fertilizantes típicamente proveen:

- 6 macronutrientes: nitrógeno (N), fósforo (P), potasio (K), calcio (Ca), magnesio (Mg) y azufre (S).

- 8 micronutrientes: boro (B), cloro (Cl), cobre (Cu), hierro (Fe), magnesio (Mn), molibdeno (Mo), zinc (Zn) y níquel (Ni).

Los macronutrientes son consumidos por la planta en grandes cantidades, estando presentes en altos porcentajes de la estructura de la planta (de 1,5 a 6%). Por su parte, los micronutrientes son consumidos en bajas cantidades y están presentes en el tejido de la planta en el orden de partes por millón (ppm). Debe notarse que no todas las plantas requieren los mismos nutrientes ni las mismas cantidades.

Los fertilizantes se componen de materiales. Para definir su composición se utiliza el índice NPK, el cual representa el acrónimo de la relación entre los elementos nitrógeno (N), fósforo (P) y potasio (K) que son comúnmente utilizados en los fertilizantes. El valor de N representa el porcentaje de nitrógeno elemental por peso del fertilizante, mientras los valores de P y K representan el porcentaje de óxido en la forma P2O5 (56,4% de oxígeno y 46,3% de fósforo, es decir que el fósforo total P = 0,463 * P2O5) y K2O (17% de oxígeno y 83% de potasio, el cálculo de K total es análogo al de P) que estaría presente en el fertilizante si todo el fósforo elemental fuera oxidado en esas formas. Así, por ejemplo, un fertilizante etiquetado como 22-10-6 indica que el mismo fertilizante tiene un 22% de nitrógeno, 10% de potasio si pasara a la forma P2O5 y 6% de fósforo si pasara a la forma K2O.

Los siguientes son los fertilizantes más comunes (8):

Material	N	P_2O_5	K_2O	MgO	S
Ammonium Nitrate	35	0	0	0	0
Ammonium Sulfate	21	0	0	0	24
Calcium Nitrate	15.5	0	0	0	0
Diammonium Phosphate	18	46	0	0	0
Monoammonium phosphate	11	52	0	0	0
Muriate of Potash	0	0	60	0	0
Potassium Nitrate	13.5	0	44	0	0
SKMG or SULPOMAG	0	0	22	18	22
Sulphate of Potash	0	0	50	0	18
Single Super Phosphate	0	22	0	0	14
Triple Super Phosphate	0	46	0	0	0
Urea	46	0	0	0	0

2.4. Cosecha

Es el proceso de recolección de cultivos maduro de los campos. Marca el final del ciclo de crecimiento de un cultivo particular, así como también de la temporada de crecimiento. En campos pequeños sin mecanización, la cosecha es la labor más intensiva de la temporada; mientras en campos grandes se utilizan máquinas altamente sofisticadas a tal efecto.

3. Descripción de los nutrientes principales

3.1. Nitrógeno

Es el nutriente más importante de las plantas, conformando aproximadamente un 60% de las mismas. Es principalmente responsable del crecimiento de las hojas y de la vegetación. Una exitosa gestión del nitrógeno permite optimizar los rendimientos de los cultivos, aumentando su rentabilidad y reduciendo al mínimo su pérdida de nitrógeno. Sin embargo, la gestión del nitrógeno resulta única, debido al complicado comportamiento del nutriente en el suelo y la planta.

Por un lado, una gestión deficiente de nitrógeno puede derivar en un crecimiento detenido, hojas con clorofila insuficiente (cloróticas) y rendimiento significativamente reducido. Por otro lado, un exceso de nitrógeno puede resultar en un pobre sistema radicular, tejido blando, plantas débiles, bajos rendimientos y mayor susceptibilidad a las enfermedades y plagas.

En la naturaleza, el nitrógeno se encuentra principalmente en el aire y en el suelo. El nitrógeno atmosférico sólo puede ser usado por las plantas leguminosas (como el girasol). Por otro lado, la mayor parte del nitrógeno del suelo está contenida por la materia orgánica, que es relativamente estable y no está directamente disponible para la absorción de las plantas. Las plantas solamente pueden absorber el nitrógeno en sus formas inorgánicas NO3 (nitrato) y NH4 (amonio), mientras que sólo el 2-3% por año del nitrógeno contenido en la materia orgánica se deja disponible para las plantas (este proceso se llama mineralización).

El nitrógeno en el suelo se pierde de tres formas: lixiviación, donde el nitrógeno en forma de NO3 se mueve hacia abajo junto con el agua debido a la baja retención del suelo; volatilización, donde el nitrógeno se pierde en forma de gas amoníaco (NH3), normalmente debido a la fertilización superficial con urea (fertilizante con altas cantidades de nitrógeno); y por último la desnitrificación, donde el nitrógeno se pierde en forma de nitrógeno nítrico (N-NO3) es transformado nuevamente en gas por las bacterias (las cuales consumen más rápidamente en suelos saturados o húmedos) y vuelve a la atmósfera.

Para realizar la gestión del nitrógeno se deben considerar dos aspectos. El primero es cuándo aplicar el nitrógeno. En cultivos como los cereales y granos en los que se realizan sólo unas pocas aplicaciones de fertilizantes, debe regularse de forma que no se aplique demasiado temprano a la siembra (puesto que las lluvias pueden hacer que el mismo se pierda por lixiviación antes de que el mismo pueda ser aprovechado por el cultivo), mientras que tratar de aplicarlo en el momento justo no siempre es algo que pueda cumplirse debido a los posibles imprevistos climáticos o logísticos. La práctica más común consiste en dividir la aplicación de modo que la mayor dosis de nitrógeno sea aplicada antes de la etapa de mayor absorción de nitrógeno del cultivo.

3.2. Potasio

Es considerado como el segundo nutriente esencial después del nitrógeno. Regula el movimiento del agua y los nutrientes en las células de la planta, promoviendo el florecimiento, fructificación y dureza de la planta. Tiene un impacto considerable en la forma, tamaño, color y sabor.

Un nivel deficiente de potasio en las plantas puede causar clorosis (quemaduras marginales en las hojas medias y bajas de la planta, representadas por un color amarillento), crecimiento lento o retrasado y una baja en la tolerancia a los cambios de temperatura, estrés hídrico y plagas. Si la deficiencia persiste, las plantas pueden llegar a perder sus hojas mucho antes de lo que deberían.

3.3. Fósforo

Se considera como el tercer nutriente esencial luego del potasio y el nitrógeno. Participa en los distintos procesos metabólicos como la fotosíntesis, transferencia de energía y en la síntesis y degradación de los carbohidratos. Promueve el crecimiento de la raíz y el tallo.

Los tipos de compuestos de fósforo que existen en el suelo son principalmente determinados por el pH y por el tipo y cantidad de minerales, donde el punto óptimo se encuentra en un pH entre 6.0 y 7.0. Otro factor fundamental que contribuye a la disponibilidad de fósforo es la materia orgánica disponible en el suelo y los residuos de los cultivos.

La movilidad del fósforo en el suelo resulta muy limitada, por lo que las raíces sólo pueden absorber fósforo solamente en su entorno inmediato. La absorción del fósforo es un proceso intenso en energía, así que las condiciones que inhiben la actividad de las raíces como las bajas temperaturas, exceso de agua, entre otros, inhiben la absorción del fósforo.

Un suelo con una baja cantidad de fósforo retrasa el crecimiento de la planta, de las raíces y su florecimiento, generando además una coloración púrpura en las hojas más viejas de la planta. Estos síntomas se hacen visibles cuando la concentración del fósforo en las hojas desciende del 0,2%.

Por otra parte, un suelo con una alta cantidad de fósforo ocasiona la absorción de mayor cantidad de elementos en el suelo, tales como el hierro, manganeso o zinc.

En cuanto a los valores mostrados en los análisis de suelo, el fósforo presenta un comportamiento muy particular, debido a que el mismo no es móvil en el suelo, por lo tanto la cantidad de fósforo disponible para las plantas es mucho menor que la cantidad total. Así, la cantidad de fósforo en el suelo resulta siendo un índice utilizado para ayudar a predecir los requerimientos de los fertilizantes fosfatados de los cultivos. En ese sentido, las recomendaciones para la aplicación de fertilizantes se determinan sobre la base de los ensayos de campo en muchos suelos y cultivos. Diferentes métodos de análisis de suelos derivan en resultados diferentes, que deben ser interpretados con cautela. Este es un tema de amplio debate, donde ocurren casos en los que distintos laboratorios que utilizan distintos

métodos pueden tener una interpretación distinta de los mismos resultados. Como lineamientos generales, se recomienda en primer lugar tomar muestras con cierta profundidad para poder estimar niveles más aproximados al valor de fósforo disponible para las plantas (puesto que los valores de fósforo en el suelo son siempre mayores que los encontrados en el subsuelo, donde la planta puede tomarlo), y en segundo lugar que la cantidad de muestras sea considerable, ya que la mayor parte del fósforo fertilizado se encuentra a una distancia de 3 – 5 cm del punto donde se aplicó el fertilizante en la capa del suelo, por lo que la ubicación exacta de las muestras de suelo puede afectar considerablemente el resultado.

4. Ejemplo específico de cultivo: glycine max (semilla de soja)

La cultivación de la semilla de soja resulta ideal en climas con cálidos veranos, idealmente con temperaturas de 20 a 30º. Temperaturas por debajo de 10 o encima de 40º pueden impactar seriamente en el crecimiento de la planta. El pH ideal de la soja va de 6,0 a 7,5; siendo 6,5 el óptimo. Luego de ser plantado, el cultivo tarda entre 80 y 120 días en alcanzar la madurez para poder ser cosechado. El rendimiento promedio de la soja es de 2,5 toneladas por hectárea. Una de las rotaciones más comunes es soja – maíz. La fertilización del cultivo de soja puede verse en (9).

5. Referencias

1. **Fajar, Nakanishi, Tagashira, Fukuda.** http://www.f.ait.kyushu-u.ac.jp/~fajar/IntroducingSPL.pdf. *Introducing Software Product Line Development for Wireless Sensor/Actuator Network Based Agriculture Systems.* [En línea]

2. **Brouwer.** http://www.fao.org/docrep/r4082e/r4082e00.htm. *Irrigation Water Management: Training Manual No. 1 - Introduction to Irrigation.* [En línea]

3. **Wikipedia.** Soil - Wikipedia, the free encyclopledia. *https://en.wikipedia.org/wiki/Soil.* [En línea]

4. **Oosterbaan.** Lecture notes on: water and salt balances of the soil, drainage and soil salinity. *http://www.waterlog.info/pdf/balances.pdf.* [En línea]

5. **Brown.** http://ufdc.ufl.edu/IR00003107/00001. *Soil texture.* [En línea] University of Florida, Institute of Food and Agricultural Sciences.

6. **Unknown.** http://www.luv2garden.com/ph.html. *Soil pH how to measure.* [En línea]

7. **Lynn, Pearson.** http://web.archive.org/web/20071027060221/http://soils.usda.gov/education/resources/k_12/lessons/color/. *The Color of Soil.* [En línea] United States Department of Agriculture.

8. http://www.agroservicesinternational.com/Education/Fert2.html. *What are fertilizers?* [En línea]

6. Links de interés

Análisis de suelo

- Análisis de suelo: información detallada sobre su composición y sus nutrientes, entre otros http://www.smart-fertilizer.com/articulos
- Análisis de suelos http://www.infoagro.com/abonos/analisis_suelos.htm
- Carta de suelos de la Provincia de Córdoba http://inta.gob.ar/documentos/publicaciones-sobre-cartas-de-suelos-de-la-provincia-de-cordoba/
- Understanding your Soil Analysis Report http://www.lovesgardens.com/Attachments/SoilAnalysisBooklet1.pdf
- Interpreting the Report http://www.agroservicesinternational.com/Soil%20Analysis/Report.html
- Ejemplo de análisis de suelo #1 http://soiltest.cfans.umn.edu/files/2012/08/Ex_rept_lawn.pdf
- Ejemplo de análisis de suelo #2 http://www.loganlabs.com/doc/Soil-Report-Sample.pdf

Ciclo de cultivo

- En Argentina http://soybeansandcorn.com/Argentina-Crop-Cycles

Fertilización

- Soja http://www.extension.umn.edu/agriculture/soybean/fertilizing-soybeans/

Programa relacionado (trial)

- Smart-fertilizer http://www.smart-fertilizer.com

Capítulo II – Inteligencia Artificial

0. Prólogo

El presente documento pretende mostrar las alternativas de investigación seleccionadas para desarrollar el sistema de información Solum. En el mismo se detalla la metodología utilizada, la arquitectura cognitiva Soar, especificando las estructuras que la componen y su ciclo de ejecución, seguido por su aplicación en Solum. A modo general, la elección de Soar como arquitectura cognitiva en la que se basa el agente Solum se fundamenta en la flexibilidad que la misma aporta tanto para brindar estructuras que almacenen conocimiento para modelar el dominio de los campos, como así también para realizar el aprendizaje a través de la experiencia para poder desempeñarse mañana mejor de lo que lo hace hoy. Para explicar estos conceptos, se detallará en qué consiste la arquitectura cognitiva Soar, junto con los aspectos más importantes de los que hace uso el agente Solum, a saber: Conceptos generales de la Inteligencia Artificial y Arquitecturas Cognitivas, Fundamentos de las Arquitecturas Cognitivas, Proceso de Desición de Markov (Markov Decision Process, PDM), Aprendizaje por Refuerzos (Reinforcement Learning, RL) y Redes Neuronales.

Parte I – Introducción a la Inteligencia Artificial

Uno de los estudios que desde los inicios de la computación ha generado debate es la posibilidad de crear entidades que se consideren inteligentes. Esto ha causado debates desde todo punto de vista, partiendo desde el cognitivo hasta el matemático, sobre qué criterios se deben tomar para considerar a una entidad, un agente, inteligente. Este cuestionamiento dio lugar, poco después de la Segunda Guerra Mundial, al campo de estudio académico conocido como Inteligencia Artificial (AI). El mismo se enfoca en investigar cómo crear computadoras que se presenten como inteligentes, de modo que se pueda asistir a los expertos en el proceso de toma de decisiones. Existen divergencias entre los académicos sobre la definición de inteligencia artificial (Bellman, 1978; Winston, 1992; Kurzweil, 1990; Nillsson, 1998), las cuales se describen a continuación.

Un enfoque considera inteligente a todo sistema que actúa como lo hace un ser humano. Para tal caso fue propuesto un test (Turing, 1950), que establecía que, para que una computadora pueda ser considerada como inteligente, un interrogador humano debía hacerle preguntas y ser incapaz de distinguir si sus respuestas provienen de una computadora u otro humano. Tal prueba establece que, como mínimo, para que una computadora sea capaz de pasar el test debía ser competente de 1) procesar de forma oral o escrita el lenguaje natural, 2) ser capaz de representar el conocimiento de alguna forma, para poder guardar lo que lee, oye y conoce y modelarlo, 3) poseer capacidades de razonamiento automático, para así ser capaz de usar la información almacenada para deducir la respuesta y derivar conclusiones y, por último, 4) adaptarse a nuevas circunstancias y detectar y extrapolar patrones. Turing estableció explícitamente que el interrogador debía evitar el contacto físico con la computadora, pero si el test incluyese estímulo físico, la computadora además debiera poder tener 5) visión para poder percibir y 6) capacidad de manipular físicamente objetos. Cada uno de los requisitos constituye en la actualidad varias de las grandes áreas especializadas de la inteligencia artificial.

Por otra parte, otro enfoque considera que un sistema puede considerarse a sí mismo inteligente cuando el mismo piensa como lo hace el ser humano. El pensamiento de una persona puede verse a partir de observar nuestros propios pensamientos (introspección), desde el punto de vista de cómo lo hace el cerebro (caja blanca), o bien con experimentos que observen el proceso de pensamiento de una persona (caja negra). Para esto poder ser traducido como conocimiento, debe ser posible modelar cada uno de estos aspectos no sólo desde el punto de vista computacional sino también desde el biológico o psicológico.

Abstrayéndose del ser humano, otra perspectiva se presenta como más general y considera que lo que define a la inteligencia no es su parecido al ser humano sino una de las capacidades fundamentales del mismo, la racionalidad. En ese sentido, una condición

necesaria para considerar a un sistema como inteligente es que el mismo pueda pensar racionalmente, esto es, pudiendo partir desde premisas y, a través de un proceso de razonamiento lógicamente irrefutable, llegar a producir conclusiones. Adicionalmente, cuando un agente es considerado como parte de un ambiente que no conoce totalmente y situado en la búsqueda de un objetivo, debe confrontar situaciones en las que no existe una acción que a priori se considere correcta, pero debe tomar una decisión. El entender cuál es la mejor acción conlleva a entender cómo actúa el entorno, lo que le otorgará al agente experiencia para poder desenvolverse mejor en el futuro de lo que lo hace en la actualidad. Tanto la experiencia como la posibilidad de razonar le confieren al agente la capacidad de pensar y actuar racionalmente, lo que representa su inteligencia.

Desde el punto de vista cognitivo, aunque la investigación en AI está en evolución desde hace más de 50 años, todavía se considera que se está lejos de alcanzar una inteligencia artificial a nivel humano. Para resolver una tarea, como humanos recurrimos casi sin realizar esfuerzo a una amplia variedad de capacidades cognitivas. En contraste, muchos sistemas de AI están diseñados para resolver problemas de un tipo tomando un enfoque específico. Mientras que dichos sistemas se desempeñan excepcionalmente en resolver dichos problemas específicos, no son capaces de integrar las distintas capacidades que asociamos con la inteligencia humana.

Teniendo en cuenta esos fundamentos, desde sus principios en 1983, un equipo de investigadores encabezado por John Laird ha puesto su foco en la construcción de una arquitectura cognitiva que integre el razonamiento basado en conocimiento intensivo, ejecución reactiva, razonamiento jerárquico, junto con las capacidades de planificación y aprendizaje (Laird 1991a, 2008; Laird y Newell 1983; Laird, Newell y Rosenbloom 1987; Laird y Rosenbloom 1996; Newell 1990; Rosenbloom y col. 1991). Esta arquitectura recibe el nombre de *State Operator Action* and *Result*, o más sintéticamente *Soar*. Se distingue por su habilidad de utilizar una amplia variedad de tipos y niveles de conocimiento para resolver problemas y subproblemas, de modo que el comportamiento del agente inteligente emerja de la combinación dinámica del conocimiento del que dispone, sea que el mismo fue programado o aprendido de la experiencia.

Parte II – Arquitecturas Cognitivas

2.1. Introducción

Uno de los desafíos fundamentales en el desarrollo de agentes con nivel de razonamiento humano es definir las estructuras computacionales primitivas que guardan, recuperan y procesan conocimiento; como así también su organización.

Un sistema computacional puede verse a través de múltiples niveles de abstracción (Newell, 1990). La capa más baja es la física, la cual consta de capas de silicio, circuitos digitales, arquitectura computacional y software que provee una serie de estructuras fijas para dar soporte a la computación general.

Por encima se encuentra la capa cognitiva, la cual provee los procesos fijos, memorias y sus algoritmos y estructuras de datos asociados para adquirir, representar y procesar conocimiento sobre el ambiente y realizar tareas para el razonamiento momento a momento, resolución de problemas y comportamiento orientado a la resolución de problemas. Esto deriva en la ecuación principal: comportamiento = arquitectura + conocimiento. El conocimiento que se almacena en la arquitectura incluye conocimiento general (como capacidades de procesamiento del lenguaje, planificación y razonamiento en retrospectiva) y conocimiento particular de las tareas (que es específico al dominio del problema).

En el nivel más alto y por encima de la capa cognitiva se encuentra la capa de conocimiento, la cual se abstrae completamente del procesamiento realizado en la capa cognitiva. En lugar de describir un agente a partir de estructuras de datos, representaciones de conocimiento y algoritmos, lo describe usando el contenido del conocimiento y el principio de racionalidad (Newell 1982, 1990), en el cual el agente selecciona las acciones en pos de alcanzar un objetivo a partir del conocimiento del que dispone. Alcanzar la capa de conocimiento requiere un perfecto nivel de racionalidad, el cual es computacionalmente inviable excepto que el agente cuente con conocimiento muy limitado o en la resolución de problemas muy viables.

Es importante notar que el papel de la arquitectura cognitiva es el de facilitar el comportamiento del agente, pero el mismo necesita además ser determinado por el conocimiento. Es por ello que el principal desafío de las arquitecturas cognitivas es el de proveer estructuras para aproximarse al nivel de conocimiento bajo la restricción de estar físicamente limitados por las capacidades de los recursos computacionales. Una de las hipótesis que en las que se basa la investigación de las arquitecturas cognitivas es que las aproximaciones al comportamiento racional emergen de la combinación de grandes cuerpos de conocimiento y del conjunto definido de procesos que manipulan los mismos. Dichos procesos se caracterizan porque proveen una estructura de tal forma que sólo se necesita introducir información sobre la tarea (tanto general como específica).

Al diseñar una arquitectura cognitiva se debe hacer frente a varias dificultades, tales como el diseño del agente, la integración de un buen mecanismo de aprendizaje que sea eficiente en consumo de recursos y la decisión de cómo representar el conocimiento, de forma que pueda ser consultado de forma efectiva y eficiente. Al utilizar una arquitectura, el diseñador del agente evita tener que considerar cómo implementar todas estas complejas

capacidades cognitivas desde cero en un lenguaje de programación como Java; en cambio se enfocará directamente en la codificación del conocimiento.

Los sistemas basados en arquitecturas tienen las siguientes ventajas:

- Se implementan muy eficientemente, ya que no requieren el ciclo adicional de recuperación e interpretación de conocimiento para producir un comportamiento en el agente.
- La arquitectura se encuentra disponible para todas las tareas desde el principio de la existencia del agente.
- Se puede acceder a información que no estaría disponible en los agentes con conocimiento predefinido. Por ejemplo, la arquitectura puede mantener una memoria de las decisiones anteriores del agente para encontrar patrones de comportamiento ocultos.
- Un sistema basado en arquitectura es estable, de tal forma que los diseños de otros mecanismos arquitectónicos pueden estar basados en los detalles de su implementación.
- Un sistema basado en arquitectura no interviene con el procesamiento de tareas del agente: el agente puede realizar sus tareas con total normalidad sin necesidad de realizar interrupciones para realizar su aprendizaje.

2.2. Arquitectura Cognitiva Soar

La arquitectura Soar se basa en objetivos, espacios de problema, estados y operadores. La fig. 1 presenta una visión estructural de Soar, con sus memorias primitivas representadas por bloques cuadrados, sus procesos con bloques con borde circular y sus conexiones con flechas dirigidas. La entrada viene a través de un módulo de percepción y es guardada en la memoria perceptual de corto plazo. Las estructuras simbólicas son extraídas de la memoria perceptual de corto plazo y añadidas a la memoria de trabajo de Soar. La memoria de trabajo actúa como una memoria a corto plazo global que efectúa la recuperación del conocimiento de la memoria a largo plazo simbólica de Soar, sirviendo además como base para inicializar acciones. Las tres memorias simbólicas a largo plazo son independientes entre sí, con mecanismos de aprendizaje separados.

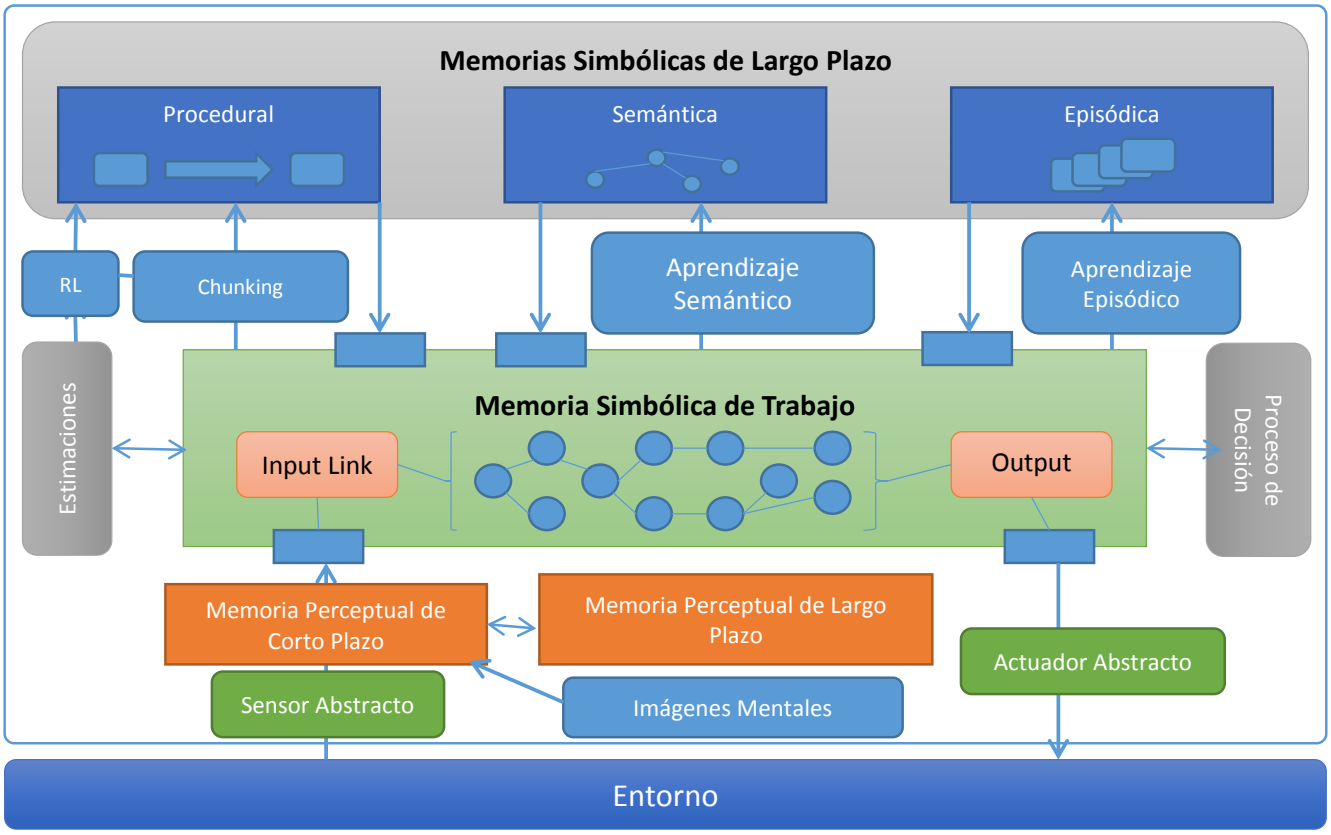


Fig. 1. Diagrama de bloques de la arquitectura cognitiva Soar.

La memoria procedural es responsable de recuperar el conocimiento que controla el procesamiento. El conocimiento de Soar se presenta como reglas de producción, las cuales realizan un *matching* de condiciones con los contenidos de la memoria de trabajo y realizan sus acciones en paralelo.

Las reglas de producción pueden modificar la memoria de trabajo. Generan preferencias para controlar el comportamiento, las cuales son usadas por el procedimiento de decisión para seleccionar un operador. Una vez que un operador es seleccionado el mismo es aplicado, causando cambios permanentes en la memoria de trabajo. La memoria de trabajo, por su parte, tiene áreas reservadas que son constantemente monitoreadas por otras memorias y procesos, de modo que un cambio en la memoria de trabajo puede disparar llamadas desde la memoria semántica o episódica, o bien podría generar acciones desde el ambiente.

Soar posee dos mecanismos de aprendizaje asociados con la memoria procedural: chunking y Aprendizaje por Refuerzos. El Aprendizaje por Refuerzos es el mecanismo por el cual Soar determina la mejor acción ante igualdad de preferencias en los operadores. El mismo será tratado en detalle posteriormente. Por su parte, el *chunking* es el mecanismo por el cual la arquitectura lleva a cabo el proceso de aprendizaje generando nuevas reglas

de producción a partir de que el proceso de generación de planes de siembra es realizado como un procedimiento automático que convierte la búsqueda en el espacio de estados del problema partiendo desde un estado inicial en conocimiento lógico que puede ser accedido posteriormente mediante la activación de las reglas que lo constituyen en función del estado actual del campo.

Por otra parte, la memoria semántica almacena hechos generales, mientras que la memoria episódica guarda instantáneas de la memoria de trabajo. Ambas memorias son accedidas por la creación de cues (señales) en la memoria de trabajo.

El conocimiento almacenado en estos módulos trabaja en conjunto para generar comportamiento, con el conocimiento procedural para brindarle control al agente y con la memoria de trabajo que actúa como un espacio de trabajo global.

2.3. Modelo Computacional de Espacio de Problema

El modelo computacional de espacio de problema (PSCM, por sus siglas en inglés *Problem-Space Computational Model*) es la teoría general de toma de decisión y resolución de problemas en la que se basa Soar. PSCM es lo que permite que se organice el conocimiento del agente, permitiendo que emerja su comportamiento como una serie de decisiones en base a su conocimiento disponible para seleccionar acciones en pos de completar sus metas.

En el PSCM, un agente tiene una meta, es decir un estado ideal del entorno al que está intentando llegar, debiendo ser capaz de cambiarlo mediante algún mecanismo actuador, y de percibirlo mediante algún sensor. El entorno, por su parte, puede ser simulado, real o digital. El agente se dice que en todo momento se encuentra inmerso en un estado, donde sus acciones posibles son los operadores. Una vez que selecciona un operador, el mismo es aplicado realizando cambios permanentes en el entorno, los cuales a su vez dan pie a las proposiciones de nuevos operadores.

Un espacio de problema es un conjunto de estados por el cual el agente puede moverse usando sus operadores disponibles. El “problema” consiste en un estado inicial y un conjunto de estados que cumplen la meta del agente y que el mismo pretende alcanzar.

Las acciones que el agente tiene a su disposición se conocen como operadores. Un operador es una función que consta de dos partes: la primera es la parte de proposición, donde se testean que los distintos aspectos del estado cumplan una serie de requisitos definidos por el operador llamados pre-condiciones. Si las mismas se cumplen, se ejecutan las post-condiciones del operador, donde se realiza la proposición efectiva del operador. Una vez que el agente selecciona uno de los operadores propuestos, se procede con la fase de aplicación, donde el operador realiza los cambios efectivos del entorno. La aplicación de un operador puede involucrar múltiples acciones, a criterio del diseñador del

agente. Las condiciones para la proposición de operadores restringen el espacio de operadores para considerarlos sólo cuando los mismos sean relevantes, y así definir el espacio posible de estados que podrían ser considerados para cumplir con los objetivos.

Un operador puede buscar resolver un problema interno, cambiando la memoria de trabajo, o externo, cambiando efectivamente el ambiente. Para el caso que el problema a resolver sea interno, existen dos enfoques generales para su aplicación. El primer enfoque consiste en que el operador modifique destructivamente las estructuras del estado actual, removiendo y añadiendo estructuras. En el segundo enfoque se crea un nuevo estado, sin crear ni destruir estructuras del estado actual; las nuevas producciones se añaden de forma simultánea.

```
Ejemplo 1: Proposición del operador siembra

sp {propose*siembra

#Precondiciones

   (state <s> ^name solum

              ^campo <campo>

              ^mes-actual <mes-actual>)

   (<campo> -^cultivo-actual

            -^procesamiento-pendiente-post-cosecha

             ^cultivo-asignable <cultivo>)

   (<cultivo> ^opcion-cultivo <opcion-cultivo>)

   (<opcion-cultivo> ^mes-siembra <mes-actual>

                     ^mes-cosecha <mes-cosecha>)

-->

#Postcondiciones

   (<s> ^operator <op> +)

   (<op> ^nombre siembra

         ^cultivo <cultivo>

         ^mes-siembra <mes-actual>

         ^mes-cosecha <mes-cosecha>)

}
```

Sólo un operador puede ser seleccionado y aplicado a la vez. No obstante, si su aplicación realiza cambios sobre las diferentes estructuras de estado al mismo tiempo, los mismos se ejecutarán en paralelo (desde luego, desde el punto de vista técnico esto depende de la capacidad de procesamiento del sistema, pero para evitar errores inesperados un diseñador siempre debe asumir el paralelismo).

Un ejemplo de proposición de operador se muestra en el Ejemplo 1. El mismo muestra las precondiciones y post-condiciones necesarias para la proposición del operador siembra-soja. Como referencia: el texto encerrado en <> representa una variable simbólica, el texto que le sigue al símbolo ^ representa el nombre de los atributos, el texto que le sigue al nombre de los atributos pero no se encuentra encerrado en <> representa valores concretos, la palabra reservada sp representa el inicio de la *Soar Production*, y, por último, el símbolo --> representa la separación entre las precondiciones y post-condiciones.

Así, por ejemplo, en la proposición las precondiciones pueden interpretarse como sigue: "si existe un estado <s>, que contiene un atributo nombre cuyo nombre es solum, un atributo campo con valor <campo>, y un atributo mes-actual con valor <mes-actual>; donde <campo> no contiene un atributo cultivo-actual, no contiene un atributo de tipo procesamiento-pendiente-post-cosecha y contiene un atributo cultivo asignable con valor <cultivo-asignable>; dicho atributo a su vez tiene un atributo opción-cultivo con valor <opcion-cultivo>, que a su vez tiene un atributo mes-siembra cuyo atributo coincide con el de <mes-actual> y un atributo mes-cosecha cuyo valor es <mes-cosecha>".

Por su parte, las post-condiciones de la aplicación, disparadas solamente cuando el estado cumple todas las precondiciones arriba citadas, pueden ser interpretadas como siguen: "crear un nuevo operador <op> en el estado <s>, con prioridad de elección + (aceptable). Asignar el atributo name a <op> y el valor siembra-soja. Asignar además a <op> el atributo mes-siembra cuyo valor es el anteriormente definido por <mes-siembra>, y finalmente el atributo mes-cosecha cuyo valor será el mismo que se definió anteriormente por <mes-cosecha>".

La estructura de los operadores que utiliza lógica de primer orden permite una gran flexibilidad en comparación con la programación clásica. En primer lugar, ni las estructuras de datos ni sus tipos son declarados de antemano, sino que se añaden en tiempo de ejecución en forma de atributos con los valores o tipos pertinentes. Es tarea del diseñador elegir qué atributos son pertinentes para cada objeto. En segundo lugar, no se utiliza el concepto de listas como se lo conoce convencionalmente en los lenguajes de programación, puesto que asignarle más de una vez un valor o variable a un atributo resultará en el atributo teniendo múltiples valores (en Soar no existe la asignación de

variables de forma convencional, para eliminar el valor de una variable debe llamarse a dicho valor o dicha variable de forma explícita). Por ejemplo, si al atributo mes-siembra se le agrega el valor octubre y posteriormente se le agrega el valor noviembre, al consultar sobre el valor de mes-siembra el sistema devolverá

```
<m> ^mes-siembra octubre ^mes-siembra noviembre
```

Es decir que mes-siembra tendrá dos valores: octubre y noviembre. Esto resulta en una ventaja de la arquitectura, puesto que si el campo no contiene un cultivo y el cultivo de soja puede ser sembrado tanto en octubre como en noviembre, se propondrán ambas alternativas al mismo tiempo como dos operadores distintos, quedando a juicio del agente determinar cuál de las mismas es la mejor opción y sin necesidad de implementar mecanismos adicionales para recorrer los distintos valores. Esto deriva en otra ventaja adicional: pueden realizarse acciones para todos los operadores que cumplan la condición, es decir que si las pre-condiciones se cumplen para un conjunto de estructuras, entonces las post-condiciones serán aplicadas a todas ellas. En otras palabras, las variables pueden utilizarse como cuantificadores universales. Esto incrementa sustancialmente la flexibilidad y facilidad de mantenimiento del lenguaje, de modo que el diseñador del agente puede abstraerse de ciertos detalles en la programación y enfocarse directamente en el comportamiento del agente.

Como complemento a los operadores Soar se define la función de elaboración. Una elaboración es una función que, a diferencia de los operadores, se aplica siempre que se cumplen las precondiciones establecidas (se deduce, en consecuencia, que cuando dichas condiciones dejan de cumplirse los valores que las mismas habían asignado serán retractados). Es por ello que las mismas utilizan una única fase y se aplican directamente al comprobarse el cumplimiento de los requisitos. Aunque las elaboraciones también alteran el estado, se distinguen en que sólo pueden volver a ser llamadas cuando se produzca un cambio en los requisitos, para evitar que las mismas sean llamadas infinitamente si el estado que cambiaron no alteró el cumplimiento de las precondiciones. Las elaboraciones sirven como soporte a los operadores realizando tareas rutinarias que deben hacerse siempre que se cumplan algunas pautas. Como ejemplo para ilustrar cuándo usar elaboraciones, considérese una variable que almacena el valor numérico máximo que otra variable ha tomado durante la ejecución del programa. Una elaboración podría encargarse de detectar cuándo ha cambiado el máximo y actualizar inmediatamente la variable de control con el nuevo valor.

2.4. Ciclo de Procesamiento en Soar

La arquitectura cognitiva Soar cuenta con un ciclo de procesamiento que establece el orden de la ejecución del programa, tal como se muestra en la Figura 2.

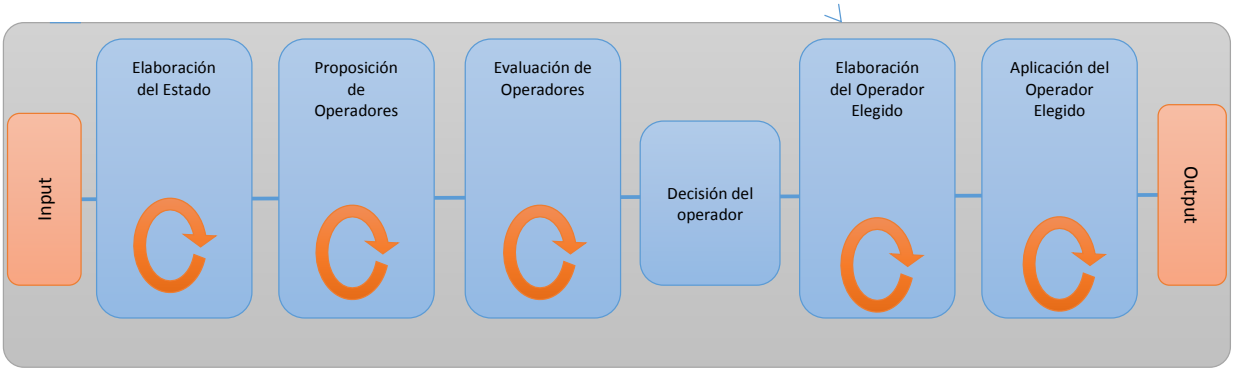


Fig. 2. Ciclo de procesamiento de la arquitectura Soar.

Cada ciclo es aplicado en fases. En la fase de entrada, la memoria de trabajo obtiene nuevos datos provenientes de sensores. En la fase de elaboración se disparan nuevas reglas de producción (y se retractan otras) para interpretar los nuevos datos y generar conocimientos derivados. En la fase de proposición la arquitectura propone operadores para el estado actual utilizando preferencias, para luego ser comparados en la fase de evaluación. Todas las producciones que hagan *matching* se disparan en paralelo (al igual que todas las retracciones) hasta que no haya más reglas que hagan matching completamente o haya retracciones de reglas de producción (estado conocido como estancamiento o *quiescence*). Por lo tanto, un proceso de decisión selecciona un nuevo operador basado en las preferencias numéricas provistas por las reglas de aprendizaje por refuerzo. Una vez que un operador de siembra es seleccionado, ejecuta su fase de aplicación y se disparan sus reglas de aplicación específicas. Las acciones de estas producciones dan lugar a más matches o retractaciones; tal como en la fase de proposición, las producciones se disparan y retractan en paralelo hasta encontrarse con el estado de quiescence. Por último, los comandos de salida son enviados al actuador, el cual realizará directamente los cambios requeridos en el entorno. El ciclo continúa hasta que el programa recibe una orden de detención desde el programa Soar (como la acción de una regla de producción).

Parte III – Aprendizaje por Refuerzos

3.1. Introducción

El término Aprendizaje por Refuerzos o Aprendizaje por Interacción (Reinforcement Learning, RL) se ha derivado de las ciencias sociales y naturales, y se encuentra relacionado a la forma en que los seres vivos aprenden incrementalmente a partir de la experiencia. La capacidad de aprender y adaptarse al entorno es considerada como uno de los atributos distintivos del ser humano (y de otras especies) y ha sido una de las principales áreas de investigación de la Inteligencia Artificial desde sus inicios.

De esta manera, la idea de "aprender" mediante la interacción con el entorno es probablemente el principal concepto que surge cuando se piensa acerca de la naturaleza de cualquier proceso de aprendizaje que involucre la habilidad y la capacidad de un individuo para adaptarse a su entorno, y mejorar su comportamiento a partir de su exposición a un problema o una determinada situación, como es habitual en la naturaleza. En particular, los seres humanos, como muchas otras especies, construyen su "conocimiento" interactuando con el medio que los rodea, es decir, "haciendo", en base a un vínculo sensorial establecido con el medio ambiente en el que se encuentran situados que les permite experimentar a partir de secuencias tipo causa-efecto. De esa manera, a partir de la evaluación de las consecuencias de las acciones y reacciones en una secuencia, es posible llevar a cabo un proceso de aprendizaje progresivo que se transforma en habilidad y destreza para realizar determinadas tareas y alcanzar metas u objetivos (Canavesio, 2007a y b).

Durante tal proceso de interacción con el entorno en que se encuentra situado, el individuo o agente cognitivo toma decisiones basadas en la experiencia y que están sujetas al mismo tiempo a la incertidumbre propia del entorno, procurando alcanzar una meta u objetivo. Si los resultados de ejecutar en el entorno las acciones derivadas de las decisiones tomadas fueron satisfactorios, el individuo aumenta el "valor" de la asociación entre la decisión tomada y las consecuencias provocadas por la misma. Entonces, las decisiones dan lugar a las acciones, y la secuencia de dichas acciones define el comportamiento del agente cognitivo en función de los resultados derivados de las decisiones tomadas. Además, es importante destacar que como en todo proceso decisorio, las mismas no solo conllevan consecuencias inmediatas sobre la situación, sino que además generan repercusiones a largo plazo, esto es, los efectos combinados de las acciones deben ser comprendidos entiendo la naturaleza secuencial de las mismas.

Conceptualmente es posible entender el Aprendizaje por Refuerzos como una formalización del "condicionamiento instrumental", teoría según la cual una acción tomada por un organismo en respuesta a un estímulo tiende a ser más frecuente según el carácter

agradable (asociado con el placer) o desagradable (asociado con el dolor) de la consecuencia producida en el entorno a partir de tal acción (Castrillón Mazo et al., 2004). Eventualmente, la idea principal es que un agente situado en un entorno 'asocie' el placer recibido al ejecutar una acción dada en un determinado contexto o entorno como un "premio" a su buen comportamiento, y el dolor recibido al ejecutar otra acción en el mismo contexto como un "castigo" o penalidad. Mediante un proceso de prueba y error, el agente sujeto del aprendizaje terminará por adoptar el comportamiento que le reporta la mayor acumulación en cantidad y/o calidad de premios recibidos.

Un punto fundamental es en que en ningún momento al agente se le especifica de manera alguna qué acción debe tomar, como sucede en la mayoría de los métodos tradicionales de "machine learning" (aprendizaje maquinal), sino que el mismo debe descubrir por sí mismo cuáles son las acciones que le proveerán una mayor retribución, a partir del proceso de interacción con el entorno. En la mayoría de los casos prácticos, la acción tomada en un determinado momento de interacción no sólo afecta la recompensa inmediata, sino también el próximo estado del entorno y recursivamente, todas las recompensas subsiguientes.

Formalmente, el Aprendizaje por Refuerzos puede definirse como el proceso mediante el cual un agente encuentra iterativamente una política (define como comportarse en cada situación) o regla decisoria que soluciona de manera óptima un problema descripto como un Proceso de Decisión de Markov (Sutton y Barto, 1998), el mismo se convierte en la opción natural a la hora de implementar computacionalmente un mecanismo para resolver un problema de toma de decisiones secuenciales en ausencia de un modelo del entorno, tal como ocurre en el problema de generar un plan de siembra. En lugar de dotar al agente cognitivo con el conocimiento disponible al momento de su programación, sólo es necesario diseñar de manera apropiada una función que premie o castigue las decisiones que éste tome a medida que interactúa con un entorno simulado, de manera tal que adquiera su conocimiento a partir de una secuencia de pruebas y errores (decisiones y sus efectos). Así mismo, el aprendizaje por prueba y error puede ser resumido en dos aspectos fundamentales definidos en la Ley del Efecto (Fernández Rebollo, 2003). El primero de ellos establece que el aprendizaje es selectivo, dando a entender que involucra la selección de entre varias alternativas, y la elección de la mejor en función de sus consecuencias a largo plazo. El segundo es que es asociativo, en el sentido de que las alternativas encontradas se asocian a las situaciones o contextos particulares en las que se tomaron. Estas dos características son clave y estarán presentes en la aplicación particular del Aprendizaje por Refuerzos que se integra con Soar.

3.2. Fundamentos

El Aprendizaje por Refuerzos como técnica se encuentra a mitad de camino entre el aprendizaje supervisado y el aprendizaje no supervisado (Russell y Norvig, 2010). Si bien requiere un conocimiento estructurado del dominio para operar, se basa en la experimentación para lograr que el agente cognitivo adquiera habilidades y aprenda comportamientos. En el tipo de problemas que pueden resolverse empleando RL se dispone de información sobre lo bien o mal que se alcanza un objetivo en su conjunto, pero no de lo buenos o malos que fueron los pasos concretos que se llevaron a cabo para realizarlo. Así, lo que el agente cognitivo aprende es un determinado comportamiento (mapeo entre situaciones o estados y acciones) a partir de la interacción con un ambiente intentando maximizar, a largo plazo, una señal de refuerzo numérica (Sutton y Barto, 1998). Tal interacción es del tipo "prueba y error": diferentes acciones se prueban en el mismo estado en diferentes etapas del aprendizaje, intentando determinar la mejor a partir de la realimentación recibida. Estos problemas pueden formalizarse de manera básica como sigue:

- Dado un agente que debe resolver un problema en un determinado entorno.
- Dado un conjunto de valores $V = \{v_1, v_2, \ldots, v_m\}$ de experiencias que relacionan estados *s* en los cuales se ha encontrado el agente, acciones *a* que puede llevar a cabo, y refuerzos inmediatos *r* que recibe del entorno,
- Asociar el nuevo estado con la acción que maximice el conjunto de refuerzos que se espera recibir a lo largo del tiempo.

Lo que se desea en este tipo de formulación es que el agente cognitivo aprenda a seleccionar las acciones que permitan obtener los mejores resultados en el tiempo. Los algoritmos que constituyen el tema central de Solum se basan en esta forma de aprendizaje, que es la que será detallada a continuación.

De manera general, la idea de realizar aprendizaje a partir de la interacción se remonta a los comienzos de la Inteligencia Artificial como disciplina. El término "Aprendizaje por Refuerzos" fue introducido en los primeros trabajos de Minsky (Minsky, 1954). Sin embargo, esta área de conocimiento ha sido estudiada en mayor profundidad recién en los últimos 20 años. En ese tiempo, se ha convertido en uno de los campos de investigación más atractivos y activos dentro de *machine learning*, en parte debido al hecho de que en los paradigmas tradicionales para llevar a cabo el proceso de aprendizaje se parte del supuesto de que el agente posee una base de conocimiento que fundamenta sus decisiones *a priori*. Sin embargo, las necesidades requeridas por las aplicaciones prácticas del mundo real han demostrado que es difícil contar con cierta información a priori, como por ejemplo modelos de comportamiento del entorno o sobre los efectos de las acciones, y es por ello que

adquiere relevancia decisiva el aprendizaje a partir de la interacción continua entre el agente y el entorno en que se encuentra situado.

Durante la interacción con su entorno, el agente desarrolla una política de actuación de manera incremental. Es por ello que el RL se plantea como estrategia de solución iterativa para los Procesos de Decisión de Markov en general, más allá de la técnica de representación de conocimiento empleada. A nivel conceptual, un MDP modela procesos de decisión secuencial bajo incertidumbre y toman en cuenta no solo la decisión actual sino también las consecuencias de decisiones futuras (Seip, 2007). Esta característica hace que los MDP sean utilizados como marco de referencia adecuado para analizar la performance y convergencia de los algoritmos que permiten implementar computacionalmente el Aprendizaje por Refuerzos como solución de los mismos, y como mecanismo de aprendizaje embebido en sistemas cognitivos de gran porte.

Formalmente, un Proceso de Decisión de Markov (MDP) puede definirse como una 5-upla

$$(S, A, P.(.,.), R.(.,.), \gamma)$$

donde S es un conjunto finito de estados, A es un conjunto finito de acciones (en particular, A_s es el conjunto finito de acciones disponibles desde el estado S). Por otra parte, $P_a(s, s') = \Pr(s_{t+1} = s' | s_t = s, a_t = a)$ es la probabilidad de que la ejecución de la acción a en el estado s en tiempo t lleve al estado s' en el tiempo $t + 1$. Tal igualdad cumple con la Propiedad de Markov, ya que, formalmente

$$P_a(s, s') = \Pr(s_{t+1} = s' \mid s_t = s, a_t = a, s_{t-2} = s_i, a_{t-2} = a_i, \dots, s_{t-n} = s_0, a_{t-n} = a_0)$$
$$= \Pr(s_{t+1} = s' | s_t = s, a_t = a), \forall s \in S, \forall a \in A$$

Esto significa que la probabilidad de transicionar hacia s' estando en el estado s_t y ejecutando la acción a_t depende solamente de s_t y a_t y no del historial anterior de acciones ejecutadas que derivaron el en estado actualmente considerado. Expresado en otros términos, esto significa que cada estado s resume completamente la información relevante del entorno para poder tomar una decisión. Obsérvese como ejemplo el caso del ajedrez, donde el estado actual se encuentra definido por la posición de las piezas en el tablero, siendo esta información suficiente como para tomar una decisión sobre la siguiente jugada, sin importar qué movimientos derivaron en esa situación.

Adicionalmente, $R_a(s, s')$ es la recompensa inmediata recibida luego de transicionar desde el estado s al estado s', mientras que $\gamma \in [0,1]$ es el factor de descuento, que representa cuán importantes serán las recompensas actuales frente a las recompensas futuras, lo que determinará el balance de un agente entre decisiones que favorecen las recompensas a corto plazo frente a aquellas que favorecen las recompensas a largo plazo.

3.3. Componentes del Aprendizaje por Refuerzos

En la formulación estándar del RL, un agente interactúa con su entorno para alcanzar una meta. La idea básica del modelo es que en cada interacción, el agente debe ser capaz de sensar y capturar los elementos más relevantes acerca del estado o situación del entorno (s_t) y, basándose en sus percepciones y en el conocimiento que haya acumulado, elegir una acción a ejecutar en dicho entorno (a_t). Las acciones generan un impacto y modifican el entorno de alguna manera, y este cambio es comunicado al agente a través de una función de premio o castigo (r_t). El agente debe aprender progresivamente a elegir aquellas acciones que le proporcionan un aumento en el valor acumulado de la señal de refuerzos r_t. En este contexto, puede entenderse al problema de generación de planes de siembra como un agente que percibe un estado del ambiente (estado del campo), el cual va impactando directamente a través de acciones de siembra (operadores), que deben ser seleccionadas en base al conocimiento que haya acumulado a partir de decisiones anteriores para mantener una medida de rendimiento y alcanzar mejor su objetivo a largo plazo.

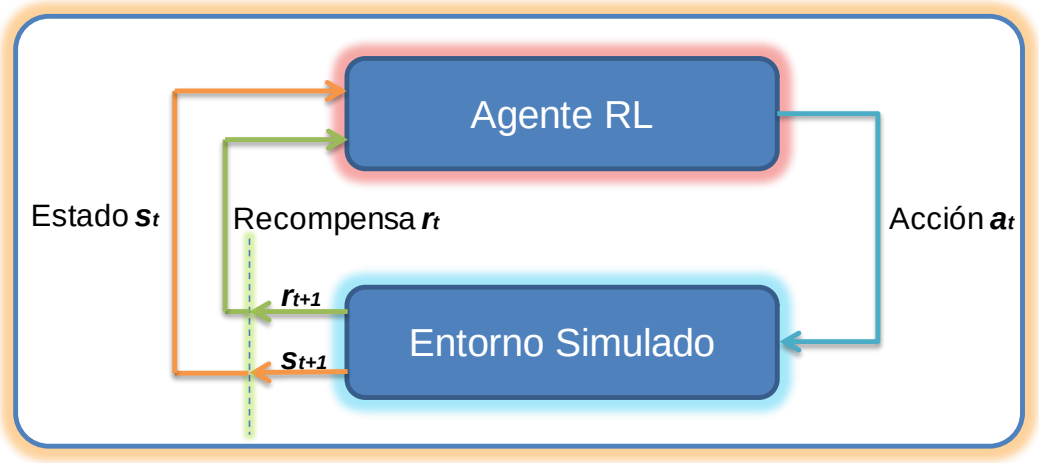


Fig. 3. Interacción Agente-Entorno en Aprendizaje por Refuerzos

A partir de lo anterior, puede establecerse que un sistema que emplea Aprendizaje por Refuerzos posee principalmente los siguientes componentes: un *Entorno*, donde el proceso a ser estudiado tiene lugar y es el encargado de otorgar las *Recompensas* a las acciones tomadas; un *Agente*, el cual debe ser capaz de aprender a controlar dicho proceso y una *Política*, que indica el comportamiento del agente en un momento dado (qué acciones tomar en qué estados), el cual se encuentra basado en el *Conocimiento* que haya adquirido.

3.3.1. Entorno

Se puede concebir al "entorno" esencialmente como el "problema" que es objeto de acciones de control propuestas por el agente, las cuales representan pasos secuenciales

para arribar a una “solución”. Debido a las características que presenta la interacción agente-entorno, las bases del Aprendizaje por Refuerzos se encuentran ligadas al formalismo de los PDM. A nivel conceptual, un PDM es un modelo para la toma de decisiones secuenciales en sistemas dinámicos bajo incertidumbre (Sutton y Barto, 1998; Hing, 2006). Desde el punto de vista de la teoría de probabilidad, un proceso estocástico tiene la propiedad de Markov cuando la probabilidad condicional de futuros estados del proceso, dado el estado actual y el historial de interacciones pasadas formado por pares *estado-acción* que llevaron al agente estado actual, depende solo del estado presente sin importar el historial mencionado (Sutton & Barto, 1998; Seip, 2007). Llevado al contexto de procesos de decisión esto significa que si el estado actual del MDP al tiempo t es conocido, las transiciones al nuevo estado $t + 1$ son independientes a todos los pares *estado-acción* previos a t.

La solución a un problema representado como un PDM consiste en encontrar una regla prescriptiva o política que permita elegir una acción para un estado dado en cada momento de decisión, de manera tal que cierto criterio de performance sea optimizado. Dicha política de actuación puede obtenerse de manera óptima a partir de programación dinámica si se cuenta con un modelo probabilístico del entorno o bien aproximarse por interacción a partir del Aprendizaje por Refuerzos cuando no se dispone de dicha información, situación que ocurre en la mayoría de los problemas reales.

3.3.2. Agente

El Aprendizaje por Refuerzos puede ser visto como un método incremental para resolver un PDM sin conocer a priori la dinámica del entorno, en base a la recepción de información acerca de los estados y las respuestas del entorno a las acciones ejecutadas sobre el mismo (Mainegra Hing, 2006). Tales acciones son controladas por un agente, por lo cual a nivel conceptual es posible definirlo como una entidad que, a través de una serie de sensores, percibe su entorno y actúa a través de las acciones que ejecuta en el mismo (Russell y Norvig, 2010). Formalmente, un agente es una función f, que toma como argumento el historial de interacción pasado y devuelve una acción a tomar en el estado actual. Una manera conveniente para representar esta idea del agente es emplear una medida de probabilidad sobre el conjunto A de acciones en base a un historial de interacción. Por lo tanto, la probabilidad de ejecutar a en el tercer ciclo, dado el historial actual, se define como

$$f(a_3|s_1r_1a_1, s_1r_2a_2, s_3r_3)$$

Durante el proceso de aprendizaje, el agente debe modificar su política de actuación de manera tal de adecuar la misma a las condiciones cambiantes del entorno, y de esta manera mejorar su comportamiento a largo plazo.

En base a lo anteriormente descripto, los siguientes son los elementos que caracterizan a un agente (Sutton & Barto, 1998; Mainegra Hing, 2006):

- Meta. Objetivo que busca alcanzar el agente. Define su criterio de performance, permitiendo mesurar cuán , el cual debería ser mejorado a partir de su comportamiento. En general, el criterio empleado es maximizar la recompensa acumulada en el tiempo.
- Comportamiento. Forma en que el agente interactúa con su entorno con el objeto de alcanzar la meta establecida, y es conocida como la política del agente. Una política es un mapeo funcional del estado percibido del entorno a una acción que debería ser tomada en dicho estado. Específicamente, para el problema de la gestión de los agroecosistemas, la política define cual debería ser la acción de siembra ejecutada en cada estado del suelo y momento dado.
- Conocimiento. Resultado obtenido y almacenado en base al procesamiento de la información que el agente recibe como retroalimentación por parte del entorno, a partir de las acciones que ejecuta secuencialmente. El mismo constituye la información que emplea el agente para definir su comportamiento, y se resume en lo que se denomina *función de valor*.
- Método de aprendizaje. Es el mecanismo reflejado en un algoritmo mediante el cual el agente procesa y actualiza su base de conocimiento.

3.3.2.1. Metas, Recompensas y Retornos

El objetivo de un agente enfrentado a la solución de un PDM es obtener el conocimiento que le permita comportarse de tal manera de obtener el máximo valor de recompensa a largo plazo. En cada paso de la interacción agente-entorno, el premio o castigo obtenido es un simple número $r \in \Re$. Es por ello que la recompensa no es un indicador absoluto de la conveniencia o no de la ejecución de una acción en un estado, ya que sólo informa acerca del beneficio a corto plazo que reporta dicha elección. La ejecución de una acción en un estado que reporte al agente una recompensa positiva alta (premio), pero que lo lleve a estados en los cuales reciba poca recompensa o recompensa negativa (castigo), será menos conveniente que la ejecución de una acción que reporte una recompensa menor, o incluso negativa en el corto plazo, pero que a largo plazo lleve a estados donde, eventualmente, se recibirán recompensas acumuladas más elevadas.

La señal de refuerzo o recompensa es la única información proveniente del entorno que posee el agente para realizar su aprendizaje, y lo que aprende es a ejecutar la secuencia de acciones más convenientes que maximizarán la suma de los premios recibidos cuando comienza en un determinado estado inicial y procede hasta alcanzar la meta establecida.

Si la secuencia de recompensas recibida por el agente después del instante de tiempo t está indicada por $r_{t+1}, r_{t+2}, r_{t+3}$ ..., lo que se busca maximizar es el retorno esperado (Sutton y Barto, 1998). Un retorno se define como una función de una secuencia de recompensas. La forma más sencilla del retorno es la suma de las recompensas

$$R_t = r_{t+1} + r_{t+2} + r_{t+3} + r_T$$

donde *T* es el tiempo final. Este enfoque tiene sentido en problemas con una definición clara de tiempo final (horizonte finito), por ejemplo, cuando la actuación del agente puede descomponerse de manera natural en episodios, como las rondas de un juego, algunas tareas de robots y otro tipo de interacciones repetidas, donde cada episodio termina con un estado especial llamado estado terminal. Los problemas continuos, por su parte, presentan mayores problemas para definir la función de recompensa, puesto que $T \rightarrow \infty$ (horizonte infinito). Un concepto adicional necesario para definir el retorno en estos casos es el de descuento. Esto es, el agente debe intentar maximizar la suma de las recompensas descontadas que recibe en el futuro. La formulación para el retorno descontado se define como

$$R_t = r_{t+1} + \gamma\, r_{t+2} + \gamma^2 r_{t+3} + \ldots = \sum_{k=0}^{\infty} \gamma^k r_{t+k+1}$$

donde el parámetro $0 < \gamma < 1$ se denomina *tasa de descuento* y representa el valor presente de las recompensas futuras. Si $\gamma = 0$, el agente sólo aprenderá a partir de la recompensa inmediata. En la medida en que $\gamma \rightarrow 1$, las recompensas futuras tienen más influencia en el comportamiento del agente.

3.3.2.2. Comportamiento

Define la manera en que el agente escoge las acciones en función de la percepción del estado o situación del entorno. En un sentido práctico, la política define las acciones (operadores de siembra) que serán ejecutadas en secuencia por el agente para obtener un plan de siembra. En este sentido, el mismo debe resolver el dilema o *trade-off* entre exploración vs explotación, balanceando adecuadamente una exploración del espacio de estados del entorno con una explotación del conocimiento adquirido en las interacciones previas. El conflicto o dilema de exploración-explotación está presente en todo problema de Aprendizaje por Refuerzos. Explotar significa utilizar soluciones conocidas, es decir sacar provecho del conocimiento acumulado hasta el presente en base a lo aprendido en el pasado (seleccionando una acción *greedy*, que es la óptima según lo conocido hasta el momento) para continuar obteniendo altas recompensas. Exploración, por otro lado, significa buscar nuevas soluciones (seleccionar otra acción a la greedy) que le permita al agente adquirir nuevos conocimientos, o detectar cambios en el entorno que pueden producir que una acción que en el pasado proporcionaba recompensas altas, ya no sea óptima. Si siempre se selecciona la acción greedy (explotación absoluta), resulta casi imposible conocer el valor de las otras acciones, que podrían resultar, a la larga, mejores que la acción greedy. Por el contrario, si nunca se toma la acción greedy, la estimación de la función de valor de acción no tendría utilidad, ya que no se estaría utilizando el conocimiento acumulado para mejorar la política. Por ello, es necesario realizar un balance adecuado entre exploración y explotación, de manera tal que el agente explote su conocimiento para obtener buenas recompensas a la vez que explora el ambiente para realizar una mejor selección de acciones en el futuro y detectar posibles cambios en el entorno.

En base a esto, un balance eficiente es fundamental para el mecanismo de Aprendizaje por Refuerzos, pues demasiada exploración puede causar un comportamiento aleatorio, mientras que poca exploración puede resultar en una política que converge muy lentamente, muy alejada de la óptima. Al respecto, existen dos estrategias básicas para resolver de manera eficiente este dilema. La primera consiste en elegir aleatoriamente una acción exploratoria con una determinada probabilidad, y el resto de las veces, elegir aquella que el agente considere la acción óptima según los valores estimados. Esta técnica se conoce como estrategia ***ε-greedy***, donde ε es la proporción de veces que se elige una acción exploratoria. El parámetro $0 \leq \varepsilon < 1$ es el que permite realizar el balance entre exploración-explotación. Cuando $\varepsilon \rightarrow 1$, mayor es el grado de exploración. Una mejora que puede aplicarse a este método consiste en ir disminuyendo el valor de ε a lo largo del tiempo, de manera que el agente comience explorando agresivamente el espacio de

estados, pero reducir ε a medida que el agente vaya comprendiendo el efecto de las acciones en el entorno, aumentando la explotación.

Otra técnica que permite resolver el dilema planteado es la conocida como *Softmax*, que se plantea como mejora de *ε-greedy* puesto que la misma tiene la desventaja de que, al momento de explorar, todas las acciones tienen la misma probabilidad de ser elegidas. Softmax, en cambio, intenta otorgar diferentes probabilidades a cada acción, en base al conocimiento adquirido sobre la recompensa que provee cada una de ellas a futuro. Para ello utiliza la distribución de Boltzmann, en virtud de la cual le asigna una probabilidad $\pi(a_t)$ a cada acción a_t correspondiente al estado s_t, tal como lo muestra la siguiente ecuación.

$$\pi(a_t) = \frac{Q(s,b)/\tau}{\sum_{b=1}^{m} e^{Q(s,b)/\tau}}$$

En la ecuación, τ es la 'temperatura computacional' y determina el grado de exploración de la política. Altas temperaturas hacen que las probabilidades de todas las acciones sean similares; por el contrario, bajas temperaturas hacen que la política se aproxime a la greedy. El método Softmax con distribución de Boltzmann es adecuado siempre y cuando la diferencia entre el valor de la mejor acción y el de las otras acciones sea grande, pero es poco efectiva por cuestiones obvias si dichos valores son muy semejantes.

3.3.2.3. Función de Valor

En términos generales, la función de valor resume el conocimiento adquirido por el agente a partir de su interacción con el entorno. La mayoría de los algoritmos de RL se basan en estimar una función de valor V, esto es, funciones que calculan qué tan bueno es para un agente estar en un cierto estado (o par estado-acción). La noción de "qué tan bueno" se encuentra definida por la recompensa futura que el agente espera obtener, en términos del retorno esperado. La recompensa o refuerzo mide numéricamente el impacto inmediato de tomar una determinada acción, en tanto la función de valor mide además el impacto de tomar una serie de acciones en el futuro. Además, es importante notar que mientras la recompensa depende de las acciones del agente, la función de valor se define con respecto a políticas de actuación particulares.

Se define como valor $V\pi(s)$ de un estado s bajo una política π como el valor esperado del retorno total resultante de la interacción con el entorno en el estado comenzando en el estado s y siguiendo la política π en adelante. Matemáticamente, la *función de valor de estado* se expresa como:

$$V^{\pi}(s) = E_{\pi}\{R_t | s_t = s\} = E_{\pi}\left\{\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} | s_t = s\right\}$$

donde $E\pi\{*\}$ denota el valor esperado a partir de que el agente sigue la política π. Informalmente, la función de valor de estado $V\pi(s)$ indica "qué tan bueno" es el estado *s* en términos de la cantidad de recompensa que el agente puede esperar obtener interactuando a partir de él y siguiendo la política elegida. El objetivo perseguido es encontrar una política óptima π^*, tal que

$$V\pi^*(s) \geq V\pi'(s) \; \forall \; s \in S, \forall \; \pi$$

Es decir, para resolver el problema es necesario encontrar la secuencia de acciones que maximiza la función de valor. Esta política óptima puede obtenerse a partir de alguno de los métodos tradicionales de programación dinámica probabilística (Bellman, 1957), en el caso de que se disponga de información completa acerca de las probabilidades de transición entre los estados acerca del modelo (Puterman, 1994). Tales métodos se encuentran basados mayormente en la búsqueda de una solución que satisface la ecuación de Bellman, en la cual se asume que son conocidos todos los detalles acerca de las probabilidades mencionadas y las funciones de probabilidad que describen los premios y castigos asociados con tales transiciones. La Ecuación de Bellman puede ser derivada como sigue (Sutton y Barto, 1998):

$$V^{\pi}(s) = E_{\pi}\{R_t | s_t = s\}$$

$$= E_{\pi}\left\{r_{t+1} + \gamma \sum_{k=0}^{\infty} \gamma^k r_{t+k+2} \, | s_t = s\right\}$$

$$= \sum_{a} \omega(s,a) \sum_{s'} P_{ss'}^{a} \, [R_{ss'}^{a} + \gamma V^{\pi}(s')]$$

donde s' denota el estado siguiente y $\omega(s,a)$ es la probabilidad de ejecutar la acción a en el estado *s* de acuerdo con la política π. Esta ecuación es conocida como Ecuación de Bellman (Bellman, 1957) y establece la relación existente entre el valor de un estado y los valores de los posibles estados sucesores para $V\pi(s)$. El inconveniente que se presenta en este tipo de planteo es que los métodos tradicionales para encontrar la política óptima a partir de la resolución de la ecuación de Bellman requieren un conocimiento completo acerca de los parámetros del problema (por ejemplo probabilidades de transición $P_{ss'}^{a}$ para todos los estados, y recompensas $R_{ss'}^{a}$, etc). En determinados problemas y entornos resulta factible conocer los parámetros del modelo; sin embargo, en el problema de generación de planes de siembra esto implicaría conocer de antemano todos el espacio de estados posibles (todos los estados del campo que son posibles de generarse en base al vocabulario Γ, lo cual es posible pero por la cantidad de variables que gobiernan sobre los suelos resulta computacionalmente intratable, entonces surge la necesidad de emplear generalización y abstracción de estados), así como también todas las posibles transiciones

entre estados como consecuencia de la aplicación de los operadores de siembra. Por lo tanto, a partir de trabajos previos en el área de Inteligencia Artificial relativos al desarrollo de políticas de decisión óptimas, es importante destacar un concepto que ha surgido como el más ventajoso en relación a lo anteriormente especificado el cual tiene que ver el aprendizaje de una función de valor estado-acción $Q\pi(s,a)$ (Watkins y Dayan, 1992). La función de valor $Q\pi(s,a)$ de ejecutar una acción a en un estado s bajo una política π se define como el valor esperado del retorno obtenido por el agente comenzando la interacción en el estado s, ejecutando la acción a y siguiendo en adelante la política π (Sutton y Barto, 1998):

$$Q^{\pi}(s,a) = E_{\pi}\{R_t | s_t = s, a_t = a\} = E_{\pi}\left\{\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} | s_t = s, a_t = a\right\}$$

Esta función de valor para las acciones $Q\pi(s,a)$ indica "qué tan bueno" es tomar la acción a en el estado s, en términos de la cantidad de recompensa que el agente puede esperar obtener luego de ejecutar la mencionada acción si a partir del estado resultante se aplica la política π.

3.3.2.3.1. Funciones de Valor Óptimas

El RL como mecanismo de solución para un PDM consiste en encontrar iterativamente una política que logre la mayor acumulación de recompensa posible a largo plazo. En ese sentido, el objetivo esencial de las funciones de valor es el de establecer un orden parcial y permitir la comparación entre diferentes políticas, funcionando como medidas de performance. Así, en base a la función de valor, una política π será mejor o igual a otra política π' si y sólo si

$$V\pi(s) \geq V\pi'(s) \; \forall \, s \in S$$

Esto significa que el retorno esperado siguiendo la política π es mayor o igual al retorno esperado siguiendo la política π' para cada estado. La *política óptima* π^* se define como aquella para la cual

$$V\pi^*(s) = V^*(s) \geq V\pi(s) \; \forall \, \pi, \forall \, s \in S$$

Por su parte, la función de valor de estado de la política óptima π^* es, entonces

$$V^*(s) = max_{\pi} V^{\pi}(s)$$

A partir de este concepto, puede observarse que la ecuación de Bellman para la función de valor de la política óptima $V^*(s)$, a diferencia de la ecuación de Bellman para funciones de valor de otras políticas, no necesita ser escrita en términos de sus probabilidades $\omega(s,a)$ (Sutton y Barto, 1998):

$$V^*(s) = max_a \sum_{s'} P^a_{ss'} [R^a_{ss'} + \gamma V^*(s')]$$

Esta ecuación se conoce como la ecuación de optimalidad de Bellman, que en realidad consiste en un sistema de ecuaciones, una por cada estado posible del sistema. Si la dinámica del entorno es conocida (es decir, se conocen las probabilidades mencionadas anteriormente), entonces en principio es posible resolver el sistema de ecuaciones para explicitar V^*. Una vez que se obtiene V^* se puede diseñar una política greedy con respecto a la función de valor V^*. Una de las mayores dificultades en el aprendizaje de una política amplia para la generación de planes de siembra se fundamenta en la imposibilidad de obtener un modelo del entorno que defina de manera exacta todas las probabilidades de transición posibles entre estados dadas las acciones de siembra y la ocurrencia de eventos disruptivos dada la incertidumbre que rige el problema. Afortunadamente, tales transiciones de estados pueden ser simuladas y puede aprenderse una política óptima de siembras para una gran variedad de los estados posibles.

Por similitud, la función de valor estado-acción óptima puede escribirse:

$$Q^*(s,a) = max_\pi Q^\pi(s,a)$$

La función Q en términos de la ecuación de optimalidad de Bellman resulta entonces en (Sutton y Barto, 1998):

$$Q^*(s,a) = \sum_{s'} P^a_{ss'} [R^a_{ss'} + \gamma max_{a'} Q^*(s',a')]$$

La importancia de la función $Q^*(s,a)$ se basa en que la misma puede solucionar la insuficiencia que presenta la función de valor $V^*(s)$ por un lado, para descubrir la política óptima cuando no se dispone de un modelo probabilístico completo del problema en cuestión; por otro lado, para seleccionar acciones a partir de la política óptima. Es decir, si el agente conoce *Q*,* para determinar qué acción seleccionar en el estado actual, sólo debe efectuar una búsqueda de un paso en el espacio de estados, esto es, para cualquier estado *s* lo único que necesita es encontrar una acción *a* que maximiza $Q^*(s,a)$. Si bien el costo computacional de representar esta función es mayor que para $V^*(s)$, la primera permite seleccionar acciones óptimas sin tener conocimient sobre estados sucesores posibles o sus valores, esto es, sin conocer las dinámicas que rigen el comportamiento del entorno del agente. Puede entonces deducirse que:

$$V^*(s) = max_a Q^*(s,a)$$

lo que implica que es posible obtener V^* a partir de la definición de Q^*. Como consecuencia, se deduce que una política óptima para la función de valor estado-acción puede obtenerse directamente:

$$\pi^*(s) = argmax_a Q^*(s,a)$$

Es decir que si el agente conoce la función Q^*, puede derivar de manera directa la política y la función de valor V^*. En problemas con información incompleta e incertidumbre presente en el entorno, la función $V^*(s)$ no es útil porque no discrimina el conocimiento aprendido a partir de cada acción sino que lo resume en el valor del estado, constituyendo esta situación una clara desventaja en cuanto a la formulación de algoritmos de aprendizaje.

Otra cuestión que resulta esencial en aplicaciones prácticas es que, si bien es posible aprender políticas óptimas a nivel teórico, esto rara vez sucede, en particular debido al extremo costo computacional que reviste resolver las ecuaciones planteadas en espacios de estado de gran tamaño, tarea que constituye un ideal que puede ser alcanzado en distintos grados en función del agente implementado o la representación de estados y acciones empleada. Por lo tanto, en la implementación computacional del Aprendizaje por Refuerzos como solución a problemas planteados como un PDM se recurre al aprendizaje de la función de valor $Q^*(s,a)$ a partir de enfrentar al agente a un entorno simulado, en el cual pueden incluirse, si se dispone de ellos a priori, datos reales.

A partir de lo anterior, se discutirán dos aspectos importantes: el primero, se encuentra relacionado a la manera en que un agente aprende a aproximar computacionalmente la función $Q^*(s,a)$ sin necesidad de recurrir a una formulación probabilística de la dinámica del entorno y de la función de premios y castigos. Una vez que se obtiene una aproximación aceptable de la función de valor, la correspondiente política óptima se obtiene a partir de la ecuación anteriormente enunciada. Sin embargo, es necesario contemplar el hecho de que los métodos algorítmicos existentes para realizar la mencionada aproximación de manera iterativa requieren grandes cantidades de memoria y espacio de almacenamiento para representar, actualizar y recuperar el conocimiento aprendido. En problemas que poseen un espacio de estados pequeño y finito, es posible construir la aproximación mencionada empleando estructuras tipo array o matriz con una entrada para cada estado concreto (imaginar, por ejemplo, un juego de Ta-Te-Ti). No obstante, para el tipo de problema que se resuelve en Solum, y a partir de que se emplea una representación lógico-relacional de los estados, las acciones de siembra, y el conocimiento adquirido por el agente, es necesario desarrollar métodos de abstracción y generalización del espacio de estados y el conocimiento adquirido.

3.3.2.3.2. Aproximación de la Función de Valor mediante Q-Learning

Los mecanismos de RL se dividen en dos clases fundamentales, que se encuentran diferenciadas en relación al conocimiento que se tiene sobre modelos del dominio a tratar por el agente (Kaelbling, 1996). Es decir, si se posee un conocimiento completo del problema modelado vía PDM, como sucede con los conjuntos de estados y acciones y la dinámica del entorno representada por la probabilidad de transición entre los mismos más la recompensa asociada a cada estado, se pueden aplicar de manera directa técnicas matemáticas basadas en programación dinámica, muy desarrolladas y ampliamente utilizadas en problemas de optimización en los que se dispone de toda la información necesaria (Puterman, 1994). Por el contrario, cuando no se dispone del mencionado conocimiento sobre el problema a tratar, se suelen seguir dos aproximaciones. Una de ellas, la solución consiste en que el agente aprenda primero el modelo del entorno, luego se apliquen las técnicas de programación dinámica definidas anteriormente. Dichas aproximaciones constituyen los denominados *métodos basados en modelos* (Kumar & Varaiya, 1986). En el segundo caso, para aproximar funciones de valor, se emplean técnicas alternativas que puedan aplicarse sin incorporar un conocimiento "a priori" del comportamiento del entorno. Estos son los que se conocen como *métodos libres de modelo* (Watkins, 1989).

Las técnicas de programación dinámica, tales como Iteración sobre Política e Iteración sobre Valor, pueden solucionar PDMs (es decir, encontrar una política óptima) en tiempo polinomial (Powell, 2011; Bertsekas & Tsitsiklis, 1996). Esto las hace aplicables frente a métodos de búsqueda heurística directa que necesitan un tiempo exponencial para hacerlo. Sin embargo, tales métodos presentan una considerable desventaja: requieren el conocimiento previo de $P_{ss'}^{a}$, y $R_{ss'}^{a}$,, es decir, un modelo completo del entorno (dinámica y función de premios/castigos). El problema es que tal como sucede en la mayoría de las aplicaciones de interés real, es imposible o muy difícil contar con dicho modelo ya que el mismo implicaría conocer acabadamente la dinámica del sistema productivo.

Por otra parte, los métodos libres de modelo solucionan el PDM sin la necesidad de un modelo del ambiente. Sólo requieren experiencia (real o simulada), entendida como secuencias sucesivas de estados, acciones y recompensas muestreadas del ambiente a través de un proceso de interacción del agente con éste, asumiéndose un desconocimiento completo de la dinámica del entorno. En otras palabras, el agente va conociendo los estados a medida que los visita, y las acciones disponibles para ser ejecutadas en cada una de ellos, pero no cuáles son los efectos de las mismas en el entorno hasta no poderlo experimentar. De esa manera, algoritmos se basan sólo en la experiencia que obtiene el agente vía interacción con el entorno. Dentro de los métodos libres de modelos, uno de los que mayor relevancia ha adquirido es el *Q-learning* (C. Watkins, 1989). Si bien a partir de

entonces se han formulado muchas variantes, el aquí descripto es el algoritmo básico basado en una representación tabular de la función de valor Q, para facilitar la comprensión del mecanismo de aprendizaje.

La regla de aprendizaje básica de *Q-learning* se encuentra basada en estimaciones de la función de valor *Q* que son actualizadas después de cada interacción agente-entorno. El agente comienza con alguna estimación de la función de valor, la cual puede ser inicializada de manera arbitraria o bien empleando conocimiento previo que se encuentre disponible. En cada momento de decisión *t*, en el cual el entorno se encuentra en el estado s_t, el agente escoge una acción a_t de acuerdo a su política. A partir de la acción ejecutada por el agente, el entorno reacciona devolviendo una recompensa r_{t+1} y realizando una transición a un nuevo estado s_{t+1}. Con esta nueva información, el agente actualiza los valores-*Q* y escoge nuevamente una acción a_{t+1} para el estado s_{t+1} a partir del cual el proceso se repite. La regla de actualización de conocimiento en este método, se define como (Watkins, 1989; Sutton y Barto, 1998):

$$Q(s_t, a_t) = Q(s_t, a_t) + \alpha\left[r_{t+1} + \gamma max_{a_{t+1}} Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)\right]$$

donde γ es el factor de descuento y α es la velocidad de aprendizaje, asumiendo ambos valores reales entre 0 y 1 inclusive. Mediante este método, se aproxima directamente la función de valor de acción óptima $Q^*(s, a)$. El algoritmo se muestra a continuación.

1. Repetir
    b. Inicializar s
    c. Repetir
        i. Elegir a de s usando una política derivada de Q (por ejemplo, ε-greedy)
        ii. Ejecutar acción a. Observar r, s′
        iii. Actualizar Q(s,a) usando

$$Q(s, a) = Q(s, a) + \alpha[r + \gamma max_{a'} Q(s', a') - Q(s, a)]$$

        iv. Actualizar s = s′; a = a′
    d. Hasta que s = estado objetivo
2. Hasta que no queden episodios de entrenamiento restantes

Una de las ventajas principales derivadas de integrar en el agente cognitivo algoritmos de aprendizaje como *Q*-learning proviene del hecho de que el mismo trabaja de manera *on-line*, es decir, permite al agente aprender mientras interactúa con su entorno, el cual puede

ser implementado como una simulación, y de manera totalmente incremental. En general, el concepto de Aprendizaje por Refuerzos puede dividirse en dos fases: la primera, en la cual el agente interactúa en un entorno controlado, le permite aprender una política de actuación inicial. Una característica distintiva consiste en su capacidad de continuar aprendiendo a partir de su interacción con el mundo real, de manera tal de adaptar su política a las condiciones planteadas en el entorno. Un detalle no menor que debe ser mencionado es que, durante la fase de aprendizaje, las "pérdidas" por decisiones mal tomadas se producen en el ambiente simulado.

Por otra parte, el algoritmo *Q*-learning permite al agente aprender aunque la consecución del objetivo ocurra luego de la ejecución de varias acciones de forma consecutiva. Esto es lo que se conoce como una asignación retardada de las recompensas *(delayed reward)*, similar a lo que ocurre en el problema de planificación de siembras, en el cual las consecuencias de las acciones que se toman no solo tienen impacto en el futuro inmediato sino que se trasladan en el tiempo. Así, estos beneficios del Aprendizaje por Refuerzos son aspectos fundamentales que lo hacen apropiado para su utilización en Solum.

Adicionalmente, puede observarse que la especificación del algoritmo Q-learning es independiente del tipo de representación empleada para s, a y $Q(s,a)$. Es decir que únicamente establece cómo el agente procesa el resultado derivado de la interacción con el entorno de manera iterativa, sin considerar cómo están representados tales estados o acciones. Si tal representación es la representación lógico-relacional, resultará necesario contar con mecanismos de abstracción del espacio de estados del problema, agrupando computacionalmente estados similares (tales mecanismos se explican en la descripción de la arquitectura cognitiva Soar), dado que al utilizarse un lenguaje simbólico en la definición de los estados del campo, podría ocurrir que dos estados que difieran únicamente en un símbolo (por ejemplo, un estado que en el conjunto de hechos que lo define contiene un nivel de nitrógeno de 50,4 ppm, mientras otro exactamente igual excepto que contiene un nivel de nitrógeno de 50,405 ppm) sean tratados como dos estados distintos, cuando muy probablemente deberían tratarse como el mismo estado a los efectos de decidir qué cultivo sembrar.

3.3.2.3.3. Aproximación de la Función de Valor mediante SARSA (λ)

En la presente sección se detalla el algoritmo finalmente utilizado en Solum mediante la arquitectura cognitiva Soar, junto con los fundamentos que justifican su inclusión como mejora de Q-Learning (Sutton y Barto, 1998). Q-Learning sirve como un excelente algoritmo para sentar las bases teóricas del aprendizaje por refuerzo y la convergencia en el aprendizaje. Sin embargo, en casos prácticos puede observarse que el mismo, si bien teóricamente respeta su convergencia, termina presentando ciertas dificultades en cuanto a la celeridad en la que realiza el aprendizaje y se adapta a los cambios en el ambiente,

siendo propenso a quedar atrapados en máximos locales. Para evitar esta situación, se propone un nuevo mecanismo que actúa directamente sobre la función Q: *las trazas de elegibilidad*.

Las trazas de elegibilidad se proponen como una variable adicional de memoria que se asocia a cada estado. La traza de elegibilidad para el estado s en el tiempo t se define como

$$Z_t(s) = \begin{cases} \gamma\lambda Z_{t-1}(s) & si\ s \neq s_t \\ \gamma\lambda Z_{t-1}(s) + 1 & si\ s = s_t \end{cases}$$

donde s es un estado no final, γ es el factor de descuento y λ es el parámetro de *decay* de las trazas. Este tipo de traza de elegibilidad se conoce como "traza acumulativa" debido a que la traza de cada estado acumula valores cada vez que el mismo es visitado, para luego decaer gradualmente es su valor mientras más tiempo transcurra desde su última visita. Informalmente, la idea de las trazas de elegibilidad es que, al recibir el agente una recompensa (o castigo) desde el entorno, todos aquellos estados que han sido visitados compartan una parte del crédito (o culpa) proporcional a su aporte en el feedback recibido desde el ambiente. Tal adición será determinada por el valor que se le asigne a λ. Un $\lambda = 0$ hará que todas las trazas en un instante de tiempo t sean 0 excepto la correspondiente al estado S_t. En la práctica, esto equivaldrá a no contar con trazas para los estados, debido a que la acumulación de valor estará limitada por el instante de tiempo en el que se visitó el estado. Para valores de λ en el rango $0 < \lambda < 1$ se considerarán los estados anteriores, aunque con una ponderación cada vez menor conforme transurra el tiempo desde su última visita. Para valores de $\lambda = 1$, las trazas pasarán a decaer solamente por el valor del factor de descuento γ.

Con la idea de extender las trazas de elegibilidad a los pares estado-acción, surge el algoritmo SARSA (λ). SARSA (λ), adicionalmente se presenta como una mejora del algoritmo State Action Reward State Action (SARSA), el cual se plantea una modificación del algoritmo Q-Learning al tener un aprendizaje *on-policy*, en el cual la acción que el agente incorpora en su aprendizaje será determinada a partir de su política y será la misma acción que el agente seguirá en el instante de tiempo inmediatamente posterior; en lugar de ejecutar la acción mejor valorada en el momento y luego reutilizar su política para decidir su próxima acción.

Sea $Z_t(s,a)$ la notación usada para denotar la traza de elegibilidad del par estado-acción (s,a). El algorimo SARSA (λ) propone incorporar, por cada transición que realice el agente, la presente actualización.

$$Q_{t+1}(s,a) = Q_t(s,a) + \alpha\delta_t Z_t(s,a)\ \forall s,a$$

donde

$$\delta_t = R_{t+1} + \gamma Q_t(s_{t+1}, a_{t+1}) - Q_t(s_t, a_t)$$

y, además, para todo (s, a)

$$Z_t(s) = \begin{cases} \gamma\lambda Z_{t-1}(s,a) + 1 & si\ s = s_t\ y\ a = a_t \\ \gamma\lambda Z_{t-1}(s,a) & en\ caso\ contrario \end{cases}$$

Así, el algoritmo SARSA (λ) quedaría como sigue.

1. Repetir
    b. Inicializar s, a
    c. Inicializar $Z(s,a) = 0\ \forall s \in S, \forall a \in A$
    d. Elegir a a partir de s usando una política derivada de Q (por ejemplo, ε-greedy)
    e. Repetir
        i. Efectuar acción a, obtener R y s'
        ii. Elegir a' a partir de s' usando una política derivada de Q (por ejemplo, ε-greedy)
        iii. Hacer $\delta = R + \gamma Q(s', a') - Q(s, a)$
        iv. Hacer $Z(s,a) = Z(s,a) + 1$
        v. Para cada $s \in S, a \in A$, hacer

$$Q(s,a) = Q(s,a) + \alpha\delta Z(s,a)$$

$$Z(s,a) = \gamma\lambda Z(s,a)$$

        vi. Fin para
        vii. Actualizar s = s'; a = a'
    f. Hasta que s = estado objetivo
2. Hasta que no queden episodios de entrenamiento restantes

Como puede observarse, la actualización que antes era realizada explícitamente en el algoritmo Q-Learning mediante

$$Q(s_t, a_t) = Q(s_t, a_t) + \alpha\left[r_{t+1} + \gamma max_{a_{t+1}} Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)\right]$$

Ahora es realizada al asignar el valor a δ y actualizar el par $Q(s,a)$ correspondiente al estado recién visitado.

La mejora que plantea SARSA (λ) permite que el agente actualice todo su $Q(s,a)$ proporcionalmente al alcanzar un estado que arroje recompensa, permitiendo que el agente pueda comenzar a hacer uso de su política de explotación (en el caso de usar ε-greedy, por ejemplo) desde la primera vez que el mismo encontró recompensa. Esto significa que, a partir del inicio del segundo episodio, el agente, desde su estado inicial, ya contará con estados y acciones que serán más pasibles de ser elegidos que otros, evitando tener que pasar por una alta cantidad de estados sub-optimos hasta que la política pueda ser propagada desde el estado final hacia el estado inicial. Esta es una notable mejora del algoritmo Q-Learning, puesto que el mismo actualizaba solamente los valores $Q(s,a)$ que se encontraran próximos a un Q ya actualizado, sin importar cómo el agente había podido llegar hacia el estado final. Para entornos complejos esto significaba una considerable ralentización en el entrenamiento, puesto que el agente podría llegar a necesitar tiempos muy altos hasta que la actualización de Q llegara al estado inicial.

SARSA (λ) permite que el agente alcance la convergencia con un considerable incremento en velocidad, reduciendo significativamente la relación entre el tamaño del entorno frente a la complejidad, y permitiendo que el agente se adapte con mayor velocidad a los cambios en el entorno.

Nota: Este capítulo es una adaptación de Barsce, Palombarini y Martínez, CACIC 2013; Palombarini, Barsce y Martínez, JAIIO 2014 y Palombarini 2014.

Parte IV – Redes Neuronales Artificiales

4.1. Introducción

Uno de los enfoques optados en el estudio de la inteligencia artificial considera a toda entidad o agente como “inteligente” cuando el mismo piensa tal como lo hace un ser humano. Dentro de la imitación del pensamiento que hace el humano se pueden realizar tres distinciones. Una de las distinciones establece que el pensamiento puede ser analizado mediante introspección, realizando un monitoreo de la observación del estado mental de un ser humano. Otra de las distinciones sugiere que el pensamiento puede ser analizado desde el punto de vista de cómo lo hace el cerebro. Esta teoría sostiene que si fuese posible imitar físicamente el proceso que llevan adelante las neuronas que componen el cerebro humano, resultará, en consecuencia, viable imitar el flujo de pensamiento. Esta corriente fue una de las que a mediados de siglo pasado sentó una de las bases de la inteligencia artificial, consistente en imitar el funcionamiento de las neuronas en los humanos, potenciando el mismo con la capacidad de cálculo de las computadoras. En la actualidad, se define como red neuronal artificial a un sistema de neuronas creadas digitalmente e interconectadas de tal forma que, en base a los valores recibidos como entradas, realicen el mapeo de una función que intentará aproximarse a una salida esperada.

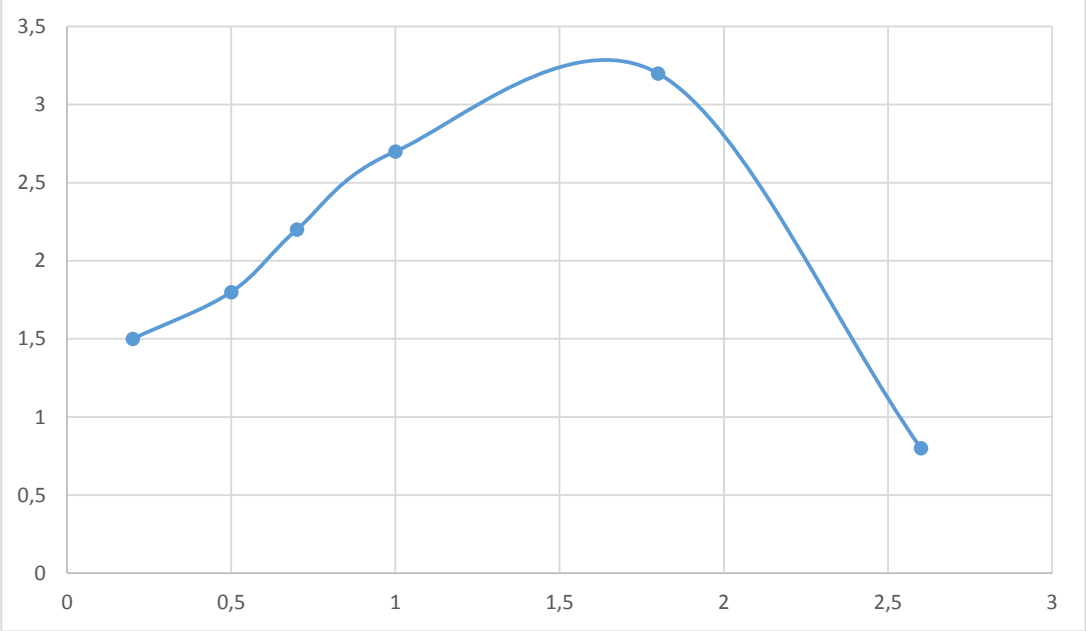


Fig. 4. Ejemplo de función aproximada en base a pares (x,y) previamente conocidos.

Las redes neuronales pueden realizar su aprendizaje de forma supervisada, en donde se conocen de antemano pares de ejemplo $(x, y), x \in X, y \in Y$ para encontrar la función $f: X \rightarrow Y$. Otros enfoques involucran el aprendizaje no supervisado, en el cual se cuenta con algunos valores de x y se busca minimizar una función de costo; y el del aprendizaje

por refuerzo, en el cual los valores de x son generados a partir de las interacciones del agente con su entorno, y capturados a partir del sensor que posee el agente.

El enfoque utilizado en Solum es el del aprendizaje supervisado, en donde el agente debe consultar una base de datos con valores históricos para poder realizar sus respectivas predicciones. Esta decisión se fundamenta en el hecho de que es posible conseguir datos de ejemplo en el dominio de los suelos con relativa facilidad, siendo factible obtener una suficiente cantidad de pares de ejemplo con un cierto grado de pluralidad. Los valores recibidos como entradas normalmente son listas de números reales (pudiendo también ser valores de texto), en un orden específico dispuesto al diseñar la red, para generar una lista formada por números reales o valores de texto como salidas. Se distinguen en las redes neuronales la fase de entrenamiento, en la cual la red aprende a mapear una función en base a ejemplos provistos, y la fase de testeo, en donde la red usa los datos que aprendió para poder predecir valores en base en un entorno real.

4.2. Arquitectura de una Red Neuronal

Previo al uso de la red neuronal, la misma debe ser diseñada de la forma que se considere o pruebe que es la mejor que se adapta al problema en cuestión. Entornos simples requerirán redes neuronales con baja cantidad de nodos (neuronas), mientras que entornos con comportamiento complejo requerirán redes neuronales con configuraciones o métodos de entrenamiento más avanzados.

A grandes rasgos, una red neuronal se compone normalmente de tres capas: la capa de entrada, la capa oculta y la capa de salida. La capa de entrada es la primera capa de la red, compuesta de un número específico de neuronas, donde por cada neurona se ingresa un valor numérico o cadena, según cómo el problema lo resuelva. Dichos valores son predefinidos por el usuario antes de comenzar con el entrenamiento.

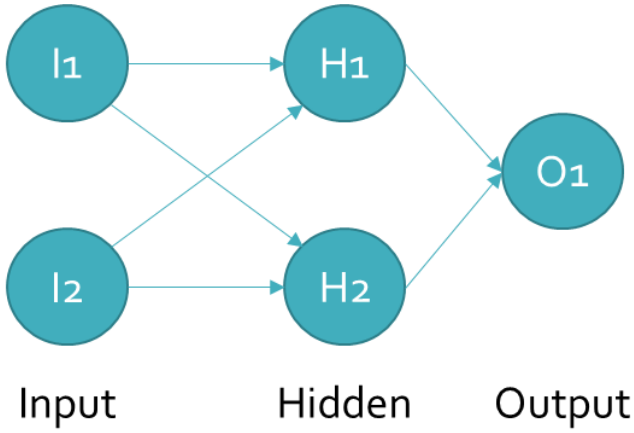


Fig. 5. Ejemplo de estructura de una Red Neuronal.

La capa que procesa los datos de entrada para transformarlos en resultados se conoce como capa intermedia o capa oculta. Su propósito permitirle a la red generar la mejor salida posible en pos de ganar precisión en la función aproximada. Su denominación parte de que la misma funciona como una caja negra, en donde en el diseño solamente se enfoca en la definición de su estructura, y sus aspectos internos sólo son modificados por un algoritmo de entrenamiento y nunca en forma manual. Esta capa es crucial para alcanzar una buena aproximación, ya que aquí es donde se guarda el aprendizaje que realiza la red en forma de pesos que le asignan a la entrada una ponderación sobre la salida. Su estructura varía según la complejidad del problema en cuestión, siendo su estructura final comúnmente obtenida mediante prueba y error para muchos problemas.

La capa que en la red produce los resultados se conoce como capa de salida. La misma muestra el procesamiento final que recibieron los datos de la capa de entrada efectuada por la capa oculta.

4.3. Entrenamiento

Para poder predecir correctamente una salida, una red neuronal debe ser entrenada, es decir, llevar a cabo la realización de simulaciones para poder ajustar sus pesos a la función que se desee mapear. Tal ajuste tiene varios componentes, listados a continuación.

4.3.1. Algoritmos de Entrenamiento

Un algoritmo de entrenamiento tiene como función iterar una cierta cantidad de veces hasta poder alcanzar una predicción lo suficientemente confiable. El algoritmo utiliza una función de activación con el objeto de definir las salidas de sus nodos, dada una serie de entradas. Cada iteración consiste en ajustar las salidas de cada uno de los nodos, probando aleatoriamente diferentes ponderaciones en busca de aproximarse lo más posible a la importancia real que debería recibir el nodo. Tras cada iteración, el algoritmo compara el resultado predicho por la red con el resultado real, usando ello como realimentación para la próxima iteración, hasta poder alcanzar un valor de predicción satisfactorio.

4.3.2. Ejemplos de Entrenamiento

Los ejemplos permiten que cada iteración del algoritmo de entrenamiento oriente los pesos de cada uno de los nodos hacia una predicción y mesure qué tan lejos se encuentra la red llegar al valor deseado del entrenamiento. La calidad de los ejemplos es crítica en el sentido que los mismos deben ser lo suficientemente heterogéneos para que representen una muestra fiel de la función a la que la red intentará acercarse, en lo posible representando una amplia cantidad de valores que incluyan los rangos límites de valores que puede tomar la función como así también fluctuaciones que tome la curva.

4.3.3. Backpropagation

Tras recibir la red una entrada como estímulo, el algoritmo de aprendizaje debe contar con algún método para propagar las entradas desde su capa hasta las capas posteriores de la red, teniendo en cuenta cada una de las ponderaciones previamente calculadas, hasta poder generar una salida. Al haberla generado, mediante la propagación hacia atrás o backpropagation, se propagan proporcionalmente los errores desde la capa de salida hasta llegar a la capa oculta y capa de entrada, donde cada uno de los nodos recibirá una fracción del error ponderada por sus pesos, que indica cuál fue su contribución en el error total de predicción. Esto permite que, a medida que la red neuronal es entrenada, los nodos de la capa oculta son organizados entre sí mismos, de modo de entender mejor las características de la capa de entrada. La salida generada por un nodo o neurona de una capa oculta o capa de salida se define formalmente como

$$n = \sum_{i=1}^{i=q} (w_i * p_i) + b$$

donde q es la cantidad de nodos cuya salida sirve de entrada a la neurona, p es el valor de entrada de la i-ésima neurona, w es el peso que la neurona le asigna a la misma y b es un valor constante.

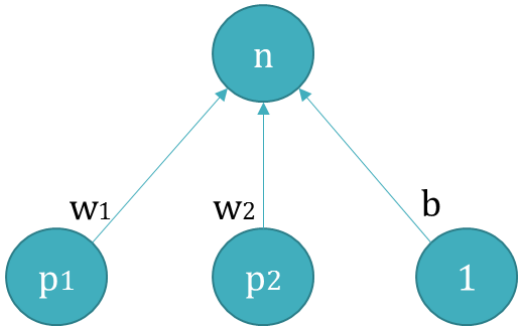


Fig. 6. Ejemplo gráfico de salida generada por una neurona.

La salida de la neurona luego es modificada por una función de activación predefinida en el algoritmo de aprendizaje, por ejemplo la función escalón de Heavyside, quedando

$$a = f(n) = f\left(\sum_{i=1}^{i=q}(w_i * p_i) + b\right)$$

4.4. Calidad de una Red Neuronal

Obtenida la función de salida, existen métricas para corroborar si la red neuronal es apta para ser utilizada en entornos reales. La métrica utilizada para saber si la red se encuentra entrenada es el error cuadrático medio, el cual representa el promedio de la diferencia entre el valor estimado por la red neuronal y el valor provisto por ejemplos, elevado al cuadrado. Formalmente:

$$MSE(\theta') = \sum_{i=1}^{n} \frac{[(\theta' - \theta)^2]}{n}$$

donde n es la cantidad de pares (x, y) suministrados como ejemplo, θ' es el valor conocido de y dado un x, mientras que θ es el valor calculado de y a partir de la red neuronal dado un x.

Tal valor suele establecerse en una cantidad menor al 0.001%, siendo la primer restricción que debe cumplir una red neuronal para considerarse apta para predecir valores reales es la de no superar dicho valor para ninguno de los casos de ejemplo. Una vez cumplida esta restricción, deben analizarse los datos devueltos en el testeo de los valores de predicción, lo que consiste en probar qué valores predice la red entrenada dados x distintos a aquellos que fueron utilizados para el entrenamiento, pero a los que se conoce de antemano los valores de y que a los que la misma debería aproximarse. En base a dichas predicciones, se determinan los siguientes valores:

$$Desviación\ total\ media = \frac{\sum_{i=1}^{n} \frac{(y'_i - y_i)}{y'_i}}{n}$$

$$Desviación\ absoluta\ total\ media = \frac{\sum_{i=1}^{n} abs\left[\frac{(y'_i - y_i)}{y'_i}\right]}{n}$$

$$Desviación\ absoluta\ máxima = máx_i \left\{abs\left[\frac{(y'_i - y_i)}{y'_i}\right]\right\}$$

donde n es la cantidad de valores x aproximados, y_i es el valor aproximado por la función generada por la red neuronal para un x_i dado, mientras que y'_i es el valor que se esperaba dé como resultado la aproximación. En base a dichos valores, el paso siguiente es realizar un análisis de cuán significativos son la desviación absoluta total media y la máxima desviación absoluta, puesto que ambos valores son que definirán cuánto es el error que puede esperarse para cualquier punto arbitrario, y cuál es el error máximo que puede llegar esperarse en la red neuronal. En base a los mismos, quien diseñe la red neuronal debe decidir si tales parámetros son tolerables o no. En tal caso, se deberá optar por conseguir mayor cantidad de datos para realizar el entrenamiento, realizar una mayor cantidad de

episodios de entrenamiento o bien optar por cambiar el diseño de la capa oculta o capa de entrada de la red.

4.5. Redes Neuronales en Solum

La implementación de redes neuronales en Solum se utiliza para generar una serie de funciones de predicción que complementan a los operadores del Agente Solum. Las mismas son llamadas tras una cierta condición del agente, como por ejemplo el realizar una fertilización. La información relevante para realizar la predicción es pasada por el agente a través de su estructura de output-link. Teniendo tal información se realiza el testeo de la red neuronal, en donde se obtienen los valores estimados, los cuales son devueltos al agente a través de su input-link, para luego proceder a la reanudación de su ciclo normal de ejecución. Se detallan a continuación la red neuronal más importante diseñada para el Agente Solum, la red neuronal de predicción de cultivos, y se describen brevemente aquellas otras que alimentan a la misma.

4.5.1. Predicción de rendimiento de cultivos

Estima, en base a los nutrientes más importantes del suelo (azufre, fósforo y nitrógeno), el cultivo sembrado, el tipo de suelo, la radiación solar anual recibida (en $\frac{MJ}{m^2}$) y la temperatura media, el rendimiento en toneladas por hectárea que obtendrán los cultivos tras su cosecha. La estructura utilizada para el entrenamiento de la red consistió en una capa de entrada comprendida por 7 neuronas, dos capas ocultas compuestas por 10 neuronas cada una, y la capa de salida con una neurona que representa el rendimiento del cultivo en toneladas por hectárea. Como función de activación se utiliza la función sigmoide para las capas de entrada y ocultas, y una función de transformación lineal para la capa de salida, tal como lo muestran las figuras 7 y 8, respectivamente.

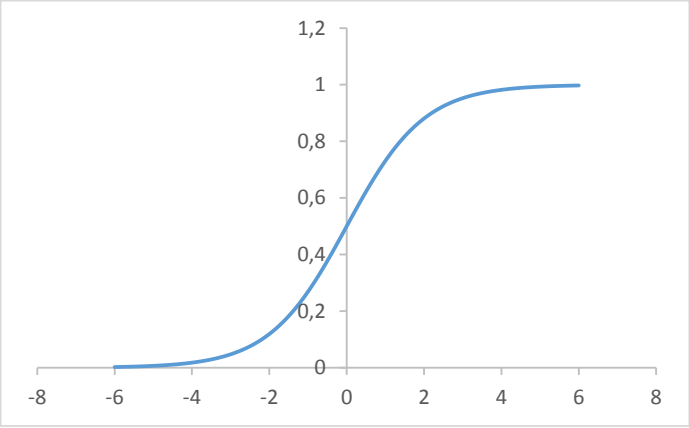


Fig. 7. Función sigmoide de activación, cuya ecuación es $S(t) = \frac{1}{1+e^{-t}}$

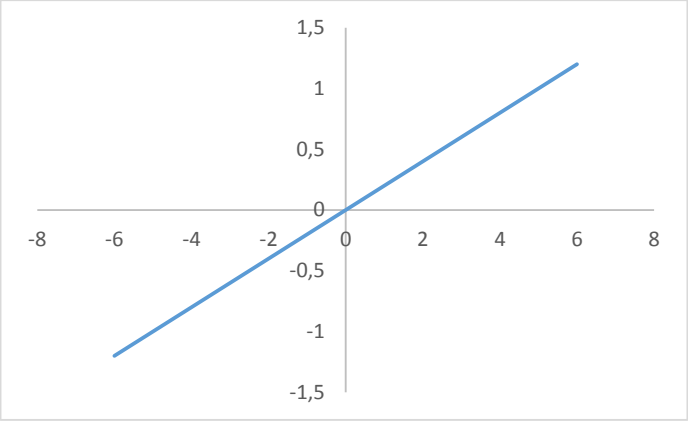


Fig. 8. Función lineal de activación.

En cuanto al error global objetivo, el mismo se definió como 10-6 con una cantidad máxima de iteraciones de 1000, considerando que el aprendizaje de la red neuronal debe ser alcanzado antes de tal cantidad. Se eligió normalizar los datos de entrada con valores entre -1 y +1. Así, antes de llevar a cabo las iteraciones, se procede a realizar el proceso de normalización de valores con una función provista a tal efecto. Los nuevos valores serán luego denormalizados de la neurona de salida para poder transformarse en valores relevantes al problema que se busca resolver. Como algoritmo de entrenamiento se optó por el algoritmo Levenberg–Marquardt (Kenneth Levenberg, 1944).

4.5.2. Predicción de extracción de nutrientes por cosecha

Se encarga de predecir cómo fluctuarán los nutrientes tras haber realizado una cosecha. Tal fluctuación es una de las más importantes a tener en cuenta para cumplir el objetivo de sustentabilidad, puesto que el agente tendrá que decidir qué cultivos le producen la mejor extracción posible a los nutrientes del suelo para poder alcanzar un nivel aceptable de sustentabilidad.

4.5.3. Predicción de adición de nutrientes por fertilizaciones

Estima cuánto fluctuarán positivamente los nutrientes en el tiempo tras haber realizado una fertilización. Las variables de entrada consideradas para dicha red son el fertilizante aplicado, su dosis de azufre, fósforo y nitrógeno y los nutrientes actuales de suelo (azufre, fósforo y nitrógeno).

4.5.4. Predicción de precipitaciones anuales

Estima la cantidad de milímetros de precipitaciones en un año dado, que determinará las condiciones particulares del suelo de los nutrientes y en las que crecerán los cultivos.

4.5.5. Predicción de radiación solar anual

Estima la cantidad de luz que recibirán los suelos en un año dado. Esta información se utiliza para alimentar directamente la red neuronal de rendimiento de cultivos.

4.5.6. Predicción de temperatura

Estima la temperatura promedio, en grados Celsius, que se predice hará en un determinado período de tiempo, como por ejemplo, el período en donde se siembra el cultivo soja. Esta información sirve como entrada esencial para la red neuronal de rendimiento de cultivos, puesto que determinará cómo serán las condiciones para en el crecimiento de los cultivos.

4.5.7. Resumen

En la presente sección se mostró un enfoque que permite llevar a cabo la generación de funciones de aproximación para la predicción de funciones que van desde el rendimiento de los cultivos hasta la temperatura y lluvias. Las mismas le servirán como complemento al agente Solum para poder calcular de la forma más aproximada posible el plan de siembra que más beneficios le aportará a un campo de acuerdo a sus características principales y su objetivo a largo plazo. Las redes neuronales se presentan como una alternativa apropiada para la predicción de las variables del campo por su rápida adaptación a las distintas condiciones de los suelos, lo que las convierte en una técnica muy apropiada para entender un dominio altamente complejo como el de los suelos. Las redes diseñadas le permitirán al agente conocer en detalle y con precisión cada una de las distintas consecuencias de las siembras que aplique sobre un suelo dado, de modo que pueda descubrir qué cultivos contribuyen mejor a cada uno de sus objetivos.

Como ventajas de las redes neuronales pueden destacarse su capacidad para funcionar en entornos no lineales y con datos de ejemplo dispersos, particularmente para dominios complejos en los que se conocen los rangos de valores que pueden tomar las salidas. Un aspecto crucial que facilitan las redes neuronales es que las mismas se abstraen de tener que conocer cómo la información es generada en el dominio, limitándose a analizar las entradas y salidas sin tener que realizar modelado previo.

Como contrapartida, debe notarse que las redes no son ideales para dominios en los cuales no se conocen los rangos mínimos o máximos de los datos de salida, o bien aquellos dominios susceptibles a los cambios. Además debe notarse que el diseño de la red neuronal es un proceso que lleva tiempo ya que debe realizarse una considerable cantidad de prueba y error debido a que las redes no poseen mecanismos para explicar las razones por las cuales se produjo cada salida.

Parte V – Integración Solum – Agente Soar

5.1. Introducción

Para ilustrar el enfoque Soar, se mostrará cómo el mismo es integra en el sistema Solum para brindar asistencia. En el PSCM diseñado para Solum, el agente puede perseguir dos objetivos de simulación: Sustentabilidad o Rentabilidad. En el primero, el agente parte desde un estado sub-óptimo del suelo con el objetivo de alcanzar un nivel adecuado de nutrientes en un intervalo determinado de tiempo, mientras que, en el caso de la rentabilidad, el objetivo del agente es maximizar la rentabilidad obtenida de las siembras en un intervalo de tiempo predefinido. En este enfoque, el espacio del problema se compone por todas las posibles combinaciones de recomendaciones de acción que pueden ser generados durante el intervalo especificado a partir de un estado inicial definido por las variables iniciales del suelo definidas inicialmente. De esta manera, un estado objetivo es todo aquel que cumple con los niveles requeridos de nutrientes en el intervalo especificado. En cada paso, el agente intentará realizar la transición desde estado actual hacia uno que espera se acerque a su objetivo. El elemento central que define la transición de un estado a otro son las siembras que el agente decida efectuar en cada uno de los momentos. El conjunto de siembras que llevan al agente desde un estado inicial hasta un estado objetivo de Sustentabilidad o Rentabilidad se conoce como plan de siembra. Una de las propiedades que definen el estado en cada momento y, por tanto, aquellos cultivos pasibles de ser sembrados, es el mes actual de la simulación. El mismo da lugar a que se propongan los distintos operadores de siembra tales como siembra-trigo-invernal o siembra-maiz.

5.2. Operadores

A los fines de alcanzar su objetivo, el agente Solum dispone de un conjunto de operadores con los que puede afectar el estado de forma directa, diseñados para que el mismo pueda alcanzar su objetivo. Los operadores brindan *O-support* al agente en la simulación, lo cual implica que todo cambio que realicen los mismos sobre el estado será de carácter permanente, sin importar si las condiciones que llevaron a los mismos a ejecutarse ya no son aplicables y pudiendo solamente ser revertidos por la aplicación de otro operador. Los mismos son detallados a continuación.

5.2.1. Inicializar-solum

Operador que se dispara sólo una vez al comenzar la ejecución del agente. Su función es la de establecer todos los elementos internos de la memoria de trabajo en su estado inicial para que, junto con el contenido proveniente desde el entorno, el agente pueda dar comienzo a la simulación y generación de un plan de siembra.

5.2.2. Transición-mes

Se dispara cuando no existe ninguna otra acción posible sobre el estado actual, provocando que la simulación transicione hacia el próximo mes. Esta situación suele alcanzarse cuando 1) existe un cultivo plantado en el campo, pero todavía no se encuentra listo para ser cosechado; o bien, 2) no existen cultivos pasibles de ser plantados por el momento en el campo.

5.2.3. Transición-año

Tal como sucede en el operador Transición-mes, Transición-año se dispara cuando no existe ninguna otra acción sobre el estado actual, con la diferencia de que la simulación se encuentra en el último mes del año. Tal distinción se realiza para que el año quede marcado como listo para luego ser guardado en el plan de siembra final.

5.2.4. Fertilización

Operador que produce una adición en los niveles actuales de los nutrientes, en base al conocimiento generado por una red neuronal. Se ejecuta una vez cada un cierto tiempo preestablecido en algún momento en el que no exista ningún cultivo sembrado en el campo. En el caso que algunos de los nutrientes del campo tengan sus valores en un nivel elevado (definido por las escalas de nutrientes de la simulación), el operador evitará agregar fertilizante en tales nutrientes.

5.2.5. Siembra

Operador que altera el estado actual del agente agregando un cultivo en el campo simulado. Una vez que el cultivo ha sido plantado, el mismo demorará una determinada cantidad de meses en quedar listo para su cosecha, impidiendo en ese tiempo que otro cultivo sea

sembrado en el campo. El operador de Siembra es uno de los más importantes de la simulación, ya que en cada ocasión al agente se le presentarán diferentes opciones de siembra en donde el mismo deberá usar Aprendizaje por Refuerzos para poder tomar su decisión. Tras la aplicación del operador de siembra y hasta su nueva proposición, el agente entra en un estado conocido como gap, el cual detiene el uso del RL con el fin de no añadir procesamiento adicional para aquellos operadores que no usarán aprendizaje por refuerzo (como por ejemplo, aquellos que poseen una preferencia *indifferent* o bien aquellos establecidos como *best* o *worst* para un caso en particular), hasta que vuelva a aplicarse un operador que precise del uso de tales métodos.

5.2.6. Cosecha

Alcanzada la cantidad de tiempo necesaria para que el cultivo quede listo para su cosecha, el operador Cosecha realiza la extracción del mismo en el campo, dejando la simulación preparada para que se efectúe la predicción de los nutrientes tras la cosecha.

5.2.7. Procesar-suelo-post-cosecha

Una vez realizada la cosecha, el operador Procesar-suelo-post-cosecha actualiza los valores del suelo de acuerdo a los nutrientes con el que el mismo contaba tras la siembra y al cultivo que se acaba de cosechar, teniendo en cuenta que existen cultivos que extraen una mayor cantidad de nutrientes que otros debido a la cantidad de restos que los mismos dejan en el suelo.

5.2.8. Calcular-reward-nutriente

Tras realizarse el procesamiento post-cosecha del suelo, el operador Calcular-reward-nutriente determina el reward total que le será asignado al agente por haber efectuado la siembra en el estado actual de la simulación. Tal reward se obtiene sumando el reward individual de cada nutriente, en base a los siguientes criterios:

Dados $V, V'min\ y\ V'max$, se define al reward asignado en el estado $s \in S$, R_s, como sigue

$$R_s = \sum_{i=1}^{\aleph} R_{s,N_i} \mid R_{s,N_i} = \begin{cases} V_{s,N_i} - V'_{s,N_i} \min\ donde\ V < V'_{s,N_i}min \\ V'_{s,N_i} \max - V_{s,N_i}\ donde\ V \geq V'_{s,N_i}min \end{cases}$$

Donde $V_{s,N_i}, V'_{s,N_i}min\ y\ V'_{s,N_i}max$ representan el valor actual de un nutriente N_i, el menor valor de tal nutriente que cumple con el objetivo de la simulación y el máximo valor objetivo del nutriente, respectivamente. Por su parte, N_i es el i-ésimo nutriente $N_i \in \aleph$.

5.2.9. Asignar-reward

Tras haber calculado el reward correspondiente al estado s, el presente operador es quien establece formalmente el valor de reward para el estado s, vinculándolo con la acción de

siembra a. Para realizar esto, utiliza un elemento primitivo de la memoria de trabajo conocido como reward-link, a quien le envía directamente el valor de reward total.

5.3. Elaboraciones

Como complemento a los operadores, el agente cuenta con un conjunto de elaboraciones que le asisten en cálculos y tareas de rutinas para poder utilizar de forma más natural sus operadores y usar ciertos tipos de funcionalidad. Cabe destacar que aquellos cambios generados por las elaboraciones cuentan con *I-support*, lo cual implica que si alguno de los requisitos para realizar una elaboración dejan de estar presentes en algún momento posterior a su elaboración, todas aquellas modificaciones realizadas por la elaboración pasan a ser retractadas, dejando de estar presentes en la memoria de trabajo. Esto hace que las mismas requieran un diseño cuidadoso, puesto que aquellas elaboraciones retractadas pueden generar una retracción de información en cadena si se utilizaron otras elaboraciones a partir de las mismas. Cabe mencionar que aquellas elaboraciones que son utilizadas como requisitos o complementos de operadores sí serán mantenidas permanentemente, debido a que la nueva información generada contará con O-support. Las elaboraciones usadas en Solum son detalladas a continuación.

5.3.1. Elaboraciones de input-link

Representan un conjunto de elaboraciones que trasladan la información proveniente de la Interfaz Solum hacia la memoria de trabajo del Agente Solum. Para ello utilizan de forma explícita la estructura del input-link provista por la Arquitectura Soar creando, por cada elemento proveniente de dicha estructura, un nuevo elemento de la memoria de trabajo. Esto se realiza mediante una elaboración y no un operador debido a que los datos del input-link no son directamente modificables desde el agente, debido a que, por definición, el input-link es la estructura que almacena la información provista de forma explícita desde el entorno del agente, por lo que un operador será incapaz de modificar tal estructura de forma explícita, necesitando de una estructura distinta a tal efecto. Entre las elaboraciones de input-link se encuentran elaboraciones tales como elaborar*campo-input-link, la cual copia los contenidos del input-link que hacen referencia al campo en una nueva estructura que se halla directamente en la memoria de trabajo.

5.3.2. Elaboraciones de finalización

Representan a aquellas elaboraciones que tienen potestad para detener el agente, si las mismas detectan que las condiciones de parada del agente han sido cumplidas. Entre las condiciones de parada se encuentran aquellas de éxito, tales como la situación en la que el agente ha alcanzado satisfactoriamente su objetivo de simulación, y aquellas de fracaso, tales como la situación en la que el agente ha superado la cantidad máxima de siembras permitidas o bien superó el tiempo permitido real o virtual de simulación.

5.3.3. Elaboraciones de escalas de nutriente

Conjunto de elaboraciones que determinan en qué nivel (escaso, insuficiente, adecuado, alto o excesivo) se encuentran los nutrientes en cada momento. Tales elaboraciones simplifican el cálculo de reward y la verificación del cumplimiento parcial o total de los objetivos de simulación por parte del agente.

5.3.4. Elaboración de objetivo parcial

Utilizando las escalas de nutriente, la presente elaboración corrobora si el objetivo de la simulación se está cumpliendo para alguno de los nutrientes del suelo. Esta situación se da cuando un nutriente tiene su valor entre el máximo y el mínimo de la escala de nutriente adecuada para el mismo.

5.4. Aprendizaje por Refuerzos en Solum

Ante la situación en la cual existan varios operadores de siembra con igual preferencia (situación común debido a no se ha establecido de antemano una jerarquía entre ellos), el agente Solum hace uso del algoritmo de aprendizaje por refuerzo SARSA(λ) (Sutton y Barto 1998) para que el agente pueda asociar un reward a cada operador de siembra para cada estado. SARSA(λ) utiliza la bien conocida fórmula mostrada en la siguiente ecuación

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha\,[r_{t+1} + \gamma Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)] * e(s_t, a_t)$$

donde $Q(s,ro)$ es el valor de aplicar el operador de siembra *ro* en el estado *s* del campo, mientras que α y γ son los parámetros del algoritmo, *r* es el valor de reward mientras que $e(s,ro)$ es el valor de la traza de elegibilidad para el operador *ro* en el estado s. Para generar la política se utiliza el algoritmo ***ε-greedy***, con un valor lineal de ***ε***. Los siguientes parámetros son los usados por defecto para llevar a cabo el aprendizaje:

$$\gamma = 0{,}9 \mid \varepsilon = 0{,}1 \mid \lambda = 0{,}1 ; \alpha = 0{,}1$$

Debido a que el espacio del problema puede ser extremadamente largo y los valores de Q son guardados en reglas de producción que no pueden ser preestablecidas, debe definirse un template de generación de reglas de aprendizaje por refuerzo (Nason, Laird 2005) para poder generar preferencias numéricas actualizables que sigan las especificaciones Soar y desarrollar el procedimiento de aprendizaje cada vez que se visiten estados por medio de aplicaciones de operadores. La siguiente es la template que se utiliza para la inicialización de cada regla de aprendizaje en el agente Solum.

```
sp {solum*rl*rules
   :template
   (state <s> ^name solum
              ^campo <campo>
```

```
                ^operator <o> +)
    (<campo> ^escala-nutriente <escala-nutriente>)
    (<escala-nutriente> ^tipo <tipo>
                        ^nutriente <nutriente>)
    (<o> ^nombre siembra)
-->
    (<s> ^operator <o> = 0)
}
```

La misma puede leerse como sigue: “Dado el estado `<s>`, el cual tiene a `solum` como nombre, contiene un `<campo>`, que contiene a su vez a una <`escala`-nutriente> con tipo <`tipo`> y para el nutriente <`nutriente`>, teniendo además un operador <`o`> de preferencia aceptable y cuyo nombre es `siembra`, entonces inicializar en `0` el operador <`o`>”. De esta forma, el template creará una regla por cada una de las distintas ocurrencias de esta plantilla, permitiéndole al agente tener un conjunto expresivo de estados por sobre los cuales realizar el aprendizaje. Cabe mencionar que tal plantilla puede ser modificada en caso de querer mejorar la convergencia o bien cuando .

Utilizar SARSA(λ) le permite al agente determinar cuándo es mejor realizar una acción por sobre otra, de acuerdo el reward asociado (valor devuelto por el entorno al alcanzar un estado objetivo) que va consiguiendo con cada ejecución. De esta manera, ante dos operadores de igual preferencia como en la siembra de soja o la siembra del maíz, el agente tiene dos opciones. Una es utilizar su conocimiento ya adquirido en experiencias anteriores para seleccionar la alternativa que mejores resultados le ha dado hasta el momento. Esto se conoce como explotación y es la medida más conservadora, puesto que el resultado más común de la misma es el ya conocido. No obstante, la explotación debe tener dos consideraciones: primero, en entornos inciertos como en Solum, es altamente probable que la mejor alternativa que el agente conoce hasta el momento sea un máximo local, existiendo opciones que devuelven mayor cantidad de reward que no pueden ser alcanzadas por las acciones que el agente cree que son las mejores hasta el momento. En segundo lugar, aún si el agente hubiera encontrado la política para alcanzar el máximo absoluto, el agente debe estar preparado para que el entorno presente un comportamiento cambiante, pudiendo ocurrir que en el caso de algún imprevisto que dicha política pase a ser sub-óptima. Para buscar evitar esto, el agente realiza, con menor frecuencia, una segunda opción a la explotación llamada exploración. En la misma, con ayuda de las trazas de elegibilidad que le ayudan a saber qué estados son los que hace tiempo no se visitan, el agente busca realizar acciones que según su conocimiento no son óptimas para detectar cambios en el

entorno y buscar mejorar su rendimiento maximizando el reward. En Solum el agente está programado para realizar acciones de explotación un 80% del tiempo, mientras que el restante 20% lo destinará a la exploración.

5.5. Convergencia en el aprendizaje del Agente Solum

Para demostrar que el agente aprovecha los beneficios del Aprendizaje por Refuerzos, se ha ejecutado un experimento simple a tal efecto. En el mismo, el agente debía decidir si optar por la siembra de un cultivo que apenas realiza extracción de los nutrientes del suelo, la cual arrojará un balance positivo de nutrientes para su preservación a largo plazo, o bien optar por un cultivo que extrae fuertemente el contenido de los nutrientes del suelo, el cual arrojará un balance negativo a largo plazo en el nivel de los nutrientes realizando fertilizaciones periódicas. El experimento pretendió demostrar que el agente puede realizar la convergencia hacia una política de siembra a largo plazo que favorezca el cultivo del primer nutriente, bajo el objetivo de la sustentabilidad. El experimento muestra que el agente es capaz de alcanzar la convergencia hacia su objetivo, en una cantidad cercana a los 20 episodios de simulación, tal como lo muestra la Figura 9.

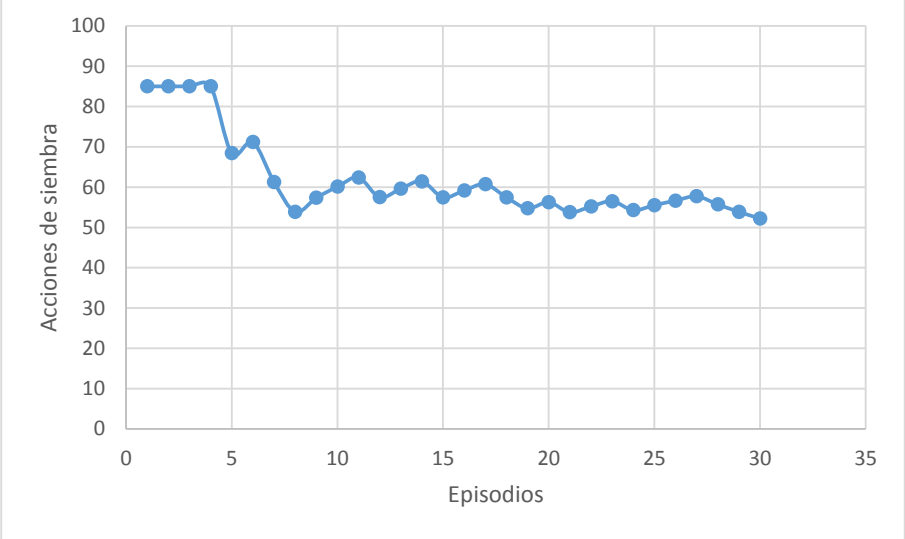


Fig. 9. Curva de Aprendizaje del Agente Solum.

Cabe mencionar que aquellos episodios en los que el agente no pudo converger hacia una política que alcance el objetivo en una cantidad menor a las 100 acciones de siembra fueron finalizados en dicho número de acciones. Los mismos, no obstante, son integrados en la curva de aprendizaje para mostrar cómo el agente aprende a evitar tales episodios. Tal como puede apreciarse, alcanzar la sustentabilidad en los primeros episodios le toma al agente cerca de unas 85 acciones de siembra. En el séptimo episodio el agente puede alcanzar la política de siembra óptima, mientras que alrededor del episodio 20 pasa a tomar conocimiento de las mejores alternativas para converger en menos de 60 acciones de

siembra. La fluctuación observada en los últimos episodios se da a causa de la frecuencia con la que el agente toma acciones exploratorias en cada episodio y, siendo que la misma es aleatoria, en algunos casos puede presentarse con más frecuencia que en otros.

5.6. Entorno de implementación y pruebas del Agente Solum

Para realizar la implementación del agente Solum en Soar, se hace uso de los entornos Visual Soar v4.6.1 y Soar Debugger 9.3.2. Visual Soar es un entorno de programación que se utiliza para realizar el diseño y la implementación de los agentes junto con el conocimiento necesario para la proposición, aplicación y elaboración de los operadores. Para definir la estructura del estado se utiliza el Datamap de Soar, el cual permite especificar las semánticas y relaciones entre los identificadores, atributos y valores. Esta es una característica complementaria de Soar, puesto que le permite al diseñador revisar que las reglas sean consistentes y realizar la detección de aquellos atributos que no son revisados ni modificados por ninguna regla de producción, resultando en una herramienta muy útil para mantener la coherencia interna de los hechos lógicos definidos por la elaboración y la estructura de proposición y aplicación de operadores.

5.7. Conclusiones

Se presentó un enfoque novedoso para el aprendizaje basado en el uso de la arquitectura cognitiva Soar para la simulación de una política basada en reglas, para tratar el problema de alcanzar un objetivo definido en la agricultura. La política de recomendaciones le permite al agente generar automáticamente una secuencia de operadores que buscará maximizar el reward obtenido en pos de obtener los mejores planes de siembra en los campos haciendo énfasis en la preservación de los suelos. Al basarse en un apropiado conjunto de templates de reglas, el enfoque permite la generación automática de nuevas reglas de representación del conocimiento del dominio a través del aprendizaje por refuerzo y el chunking de heurísticas que pueden ser naturalmente entendidas por el usuario final.

Capítulo III – Política de Seguridad de Solum

1. Introducción

La información en la actualidad es un recurso que, como el resto de los activos, tiene valor para las organizaciones y su funcionamiento, por consiguiente debe ser debidamente protegida, garantizando la continuidad de los sistemas de información, minimizando los riesgos de daño y contribuyendo de esta manera, a una mejor gestión.

Para que estos principios de la Política de Seguridad de la Información sean efectivos, resulta necesaria la implementación de una Política de Seguridad de la Información que forme parte de la cultura organizacional, lo que implica que debe contarse con el manifiesto compromiso de todos las personas que son parte o se ven afectadas por la organización, para contribuir a la difusión, consolidación y cumplimiento.

Tomando en cuenta que Solum es un producto que funciona sobre un entorno Web, el presente documento se encuentra basado en la propuesta del SANS Institute (www.sans.org) de Políticas de Seguridad para aplicaciones de este tipo, sumado a las recomendaciones y ranking de vulnerabilidades frecuentes generado por el Open Web Application Security Project (OWASP – www.owasp.org).

2. Términos y Definiciones

2.1. Seguridad de la Información: La seguridad de la información se entiende como la preservación de las siguientes características:

- Confidencialidad: se garantiza que la información sea accesible sólo a aquellas personas autorizadas a tener acceso a la misma.
- Integridad: se salvaguarda la exactitud y totalidad de la información y los métodos de procesamiento.
- Disponibilidad: se garantiza que los usuarios autorizados tengan acceso a la información y a los recursos relacionados con la misma, toda vez que lo requieran.

Adicionalmente, deberán considerarse los conceptos de:

- Autenticidad: busca asegurar la validez de la información en tiempo, forma y distribución. Asimismo, se garantiza el origen de la información, validando el emisor para evitar suplantación de identidades.
- Auditabilidad: define que todos los eventos de un sistema deben poder ser registrados para su control posterior.
- Protección a la duplicación: consiste en asegurar que una transacción sólo se realiza una vez, a menos que se especifique lo contrario. Impedir que se grabe una transacción para luego reproducirla, con el objeto de simular múltiples peticiones del mismo remitente original.
- No repudio: se refiere a evitar que una entidad que haya enviado o recibido información alegue ante terceros que no la envió o recibió.
- Legalidad: referido al cumplimiento de las leyes, normas, reglamentaciones o disposiciones a las que está sujeto el Organismo.
- Confiabilidad de la Información: es decir, que la información generada sea adecuada para sustentar la toma de decisiones y la ejecución de las misiones y funciones.

2.2. Información: Se refiere a toda comunicación o representación de conocimiento como datos, en cualquier forma, con inclusión de formas textuales, numéricas, gráficas, cartográficas, narrativas o audiovisuales, y en cualquier medio, ya sea magnético, en papel, en pantallas de computadoras, audiovisual u otro.

2.3. Sistema de Información: Se refiere a un conjunto independiente de recursos de información organizados para la recopilación, procesamiento, mantenimiento, transmisión y difusión de información según determinados procedimientos, tanto automatizados como manuales.

2.4. Tecnología de la Información: Se refiere al hardware y software operados por el Organismo o por un tercero que procese información en su nombre, para llevar a cabo una función propia de la Organización, sin tener en cuenta la tecnología utilizada, ya se trate de computación de datos, telecomunicaciones u otro tipo.

2.5. Comité de Seguridad de la Información: El Comité de Seguridad de la Información es un cuerpo integrado por representantes de todas las áreas sustantivas del Organismo, destinado a garantizar el apoyo manifiesto de las autoridades a las iniciativas de seguridad.

2.6. Responsable de Seguridad Informática: Es la persona que cumple la función de supervisar el cumplimiento de la presente Política y de asesorar en materia de seguridad de la información a los integrantes del Organismo que así lo requieran.

2.7. Release (Software): Entrega de una versión del producto de Software ya sea correctiva, de modificación, ampliación o mejora.

2.8. Vulnerabilidad: Es una falla o debilidad en el diseño de un sistema, implementación, u operación y administración que puede ser explotada para violar la política de seguridad del sistema.

2.9. Amenaza: Una amenaza es un ataque potencial que, explotando una vulnerabilidad, puede dañar los activos propios de una aplicación (recursos valiosos, como una tabla de una base de datos o en los archivos del sistema).

2.10. Test: Es una acción que tiende a demostrar una vulnerabilidad en una aplicación.

2.11. Exploit: Es una pieza de software, porción de datos, o secuencia de comandos que toma ventaja de una vulnerabilidad.

2.12. Inyección SQL (SQLi): Vulnerabilidad mediante la ejecución de código SQL y en casos de sistema operativo, mediante entradas de usuario no parametrizadas o no tratadas debidamente. Puede permitir desde la obtención, alteración o destrucción de información, como así también denegación de servicios. Cabe destacar que este tipo de vulnerabilidad no se aplica solo a sistemas de entorno web.

2.13. Cross Site Request Forgery (CSRF): El CSRF es un tipo de exploit malicioso de un sitio web en donde comandos no autorizados son transmitidos por un usuario en el cual el sitio web confía debido a cookies o información de sesión previamente ingresada.

2.14. Cross Site Scripting (XSS): Es un tipo de vulnerabilidad típico de las aplicaciones Web, que permite a una tercera parte inyectar en páginas web vistas por el usuario código generalmente JavaScript, evitando medidas de control como la Política del mismo origen.

3. Propósito del Documento

El propósito del presente documento es definir las evaluaciones de seguridad de la aplicación web en conjunto con INTA Marcos Juárez. Las evaluaciones de la aplicación web se realizan para identificar potenciales o ya existentes debilidades como resultado de un inadvertido error de configuración, autenticación débil, insuficiente manejo de errores, filtración de información sensible, entre otras. El descubrimiento y la subsecuente mitigación de estos inconvenientes, limitarán la superficie de ataque de los servicios disponibles tanto de forma interna como externa a la organización, como así también satisfacer de manera conforme a cualquier política relevante existente del lugar.

4. Alcance

Esta política cubre todas las evaluaciones de seguridad en la aplicación web solicitadas ya sea por individuos, grupos o departamentos, con el propósito de mantener la postura de seguridad, conformidad, manejo de riesgos y control de cambios de tecnología que se encuentren en uso en INTA Marcos Juárez.

Todas las evaluaciones de seguridad en la aplicación web serán realizadas por un delegado de seguridad dentro de Solum y con la posibilidad de participación de un integrante de INTA Marcos Juárez. Todos los descubrimientos serán considerados confidenciales y distribuidos a las personas en un documento formal de “Necesidad de saber”. La distribución de los descubrimientos fuera de la organización se encuentra estrictamente prohibida, a menos que la misma sea aprobada por el Jefe de Tecnologías de Información.

Cualquier relación con las aplicaciones multicapa encontradas durante la etapa de determinación de alcances será incluida en las evaluaciones, a menos que sean explícitamente limitadas. Las limitaciones y su justificación serán documentadas previo al inicio de la evaluación.

5. Política

5.1. La aplicación web está sujeta a evaluaciones de seguridad acorde a los siguientes criterios:

a) Nuevos Releases o grandes cambios en el producto serán sujetos a evaluaciones completas quedando luego vinculadas a los requerimientos de la política.

b) Terceras partes o aplicaciones web adquiridas serán sujetos a evaluaciones completas quedando luego vinculadas a los requerimientos de la política.

c) Puntos de Release serán sujetos a niveles de prueba apropiados a los riesgos de los cambios en la funcionalidad o arquitectura de la aplicación.

d) Los Releases correctivos serán sujetos a niveles de prueba apropiados a los riesgos de los cambios en la funcionalidad o arquitectura de la aplicación.

e) En Releases de emergencia se permitirá renunciar a las evaluaciones de seguridad correspondientes, asumiendo el riesgo que esto conlleva hasta que una adecuada evaluación pueda llevarse a cabo. Un Release de emergencia será designado como tal sólo por el Jefe de Tecnologías de Información o por quien sea delegado por esta autoridad.

5.2. Todos los inconvenientes de seguridad que sean descubiertos durante las evaluaciones deben ser mitigados acorde a los siguientes niveles de riesgo. Los mismos están basados en la Metodología de Clasificación de Riesgos del OWASP. Se requerirán pruebas de validación y corrección a fines de validar, corregir y/o plantear estrategias de mitigación para todos los riesgos de nivel medio o superior.

a) Alto – Todos los inconvenientes de alto riesgo deben ser corregidos inmediatamente o deberán ponerse en marcha las estrategias de mitigación a fines de limitar la exposición previa al despliegue de una corrección. Aquellas aplicaciones con alto riesgo no deben desplegarse en el entorno de ejecución.

b) Medio – Los inconvenientes de riesgo medio deben ser revisados para determinar qué debe ser mitigado. Aquellas aplicaciones con nivel de riesgo medio no deberían desplegarse en el entorno de producción, basado en el número de inconvenientes y en aquellos casos donde múltiples inconvenientes incrementan el riesgo a niveles inaceptables. Los inconvenientes deberían ser corregidos en un punto de entrega a menos que otras estrategias de mitigación limiten la exposición.

c) Bajo – Los inconvenientes de bajo riesgo deben ser revisados con motivo de determinar qué debe realizarse para corregir el riesgo y programar su ejecución.

5.4. En base a un análisis del equipo de test de seguridad las siguientes herramientas son las autorizadas para realizar las evaluaciones de la aplicación tomando en cuenta características como código abierto y reconocimiento de la comunidad:

- w3af: Escáner de seguridad en aplicaciones web.

- sqlmap: Herramienta automatizada de inyección SQL.
- OpenVAS: Sistema de evaluación de vulnerabilidades en sistemas informáticos.

5.5. Límites:

- Se realizarán análisis exhaustivos de la aplicación web Solum, de sus releases y de las aplicaciones de terceros utilizadas por Solum.
- No se analizarán aplicaciones existentes en INTA ni la dependencia de datos existentes de INTA requeridos por Solum.
- Se realizará un análisis de la infraestructura informática utilizada por Solum sólo al momento de su implementación, no se realizarán análisis periódicos de infraestructura debido a que esto ya excede la responsabilidad y desarrollo del producto Solum. Para dicho análisis se utilizará la herramienta automatizada OpenVAS.

5.6. Políticas de implementación para mitigación de riesgos: Se aplicarán las siguientes políticas de implementación basadas en los riesgos existentes y orden de criticidad según el orden de importancia de la OWASP.

Riesgo	Mitigación
Inyección SQL	En caso de no utilizar un ORM y de requerir realizar consultas directamente en la base de datos, se parametrizarán las consultas evitando concatenar sentencias SQL con variables de entrada de usuario.
Pérdida de autenticación y gestión de sesiones	Las URL no deben involucrar información sensible (ID de sesión, nombres de usuario, contraseñas). Se establecerá un tiempo de expiración de sesión de 20 minutos de inactividad.
Secuencia de Comandos en Sitios Cruzados (XSS)	Las entradas de usuario son tomadas como inseguras en todos los casos, con lo cual todos aquellos valores que deban ser mostrados en distintos sectores del resultado HTML, CSS y/o Javascript deben ser escapados y pre procesados.
Referencia directa insegura a objetos	Todas las referencias a objetos serán obtenidas en base a la sesión de usuario iniciada y no en base a entradas de usuario.
Configuración de seguridad incorrecta	Cada implementación de tercero o servicios de infraestructura serán sometidos, en primera instancia, a pruebas automáticas para verificar vulnerabilidades conocidas o

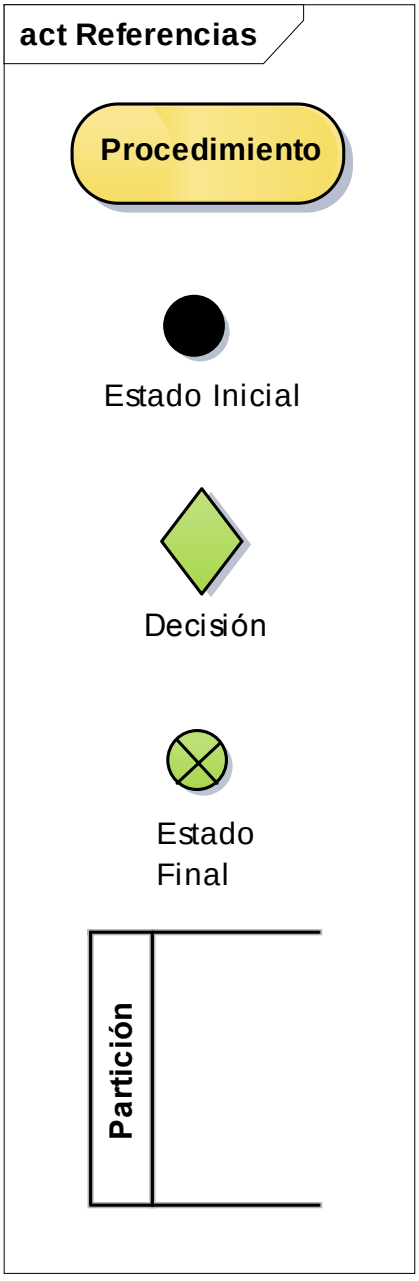
	configuraciones por defecto inseguras. En el caso de frameworks de desarrollo se deshabilitarán o eliminarán, de ser posible, aquellos ejemplos incluidos o cuentas creadas automáticamente y, de existir una guía de puesta en producción del framework, la misma deberá seguirse paso a paso, dejando registro de las tareas realizadas para dicha mitigación del riesgo.
Exposición de datos sensibles	Todos los recursos que involucren datos sensibles deberán transmitirse cifrados. La encriptación de datos debe ser asimétrica o de clave pública, de modo que sólo el backend tenga la posibilidad de desencriptar la información mediante la clave privada.
Inexistente control de acceso a nivel de funcionalidades	Se deberán controlar los niveles de acceso de usuario para cada recurso disponible, no sólo evitando mostrar links directos a los recursos que requieran un nivel de acceso distinto.
Falsificación de Peticiones en Sitios Cruzados (CSRF)	Se utilizarán tokens que permitan validar que la información proviene de un origen autenticado y corresponden a un formulario o método de entrada generado por la aplicación.
Uso de componentes con vulnerabilidades conocidas	Se realizará un listado de los componentes de terceros utilizados, de este modo se podrán analizar vulnerabilidades existentes. Se utilizará la herramienta OpenVAS periódicamente a fines de tener un rápido estado de situación sobre nuevas vulnerabilidades.
Redirección y reenvíos no validos	Las redirecciones a sitios nunca serán originadas a partir de entradas de usuarios.

Capítulo IV – Manual de Procedimientos

1. Introducción

El presente documento detalla los pasos necesarios para la realización de los procedimientos en el sistema de información Solum. Como nota, debe tenerse en cuenta que aquellos procedimientos relacionados con el agente de planificación y la red neuronal son mostrados a modo de ejemplo general, el procedimiento a seguir en cada situación depende de las características de la misma.

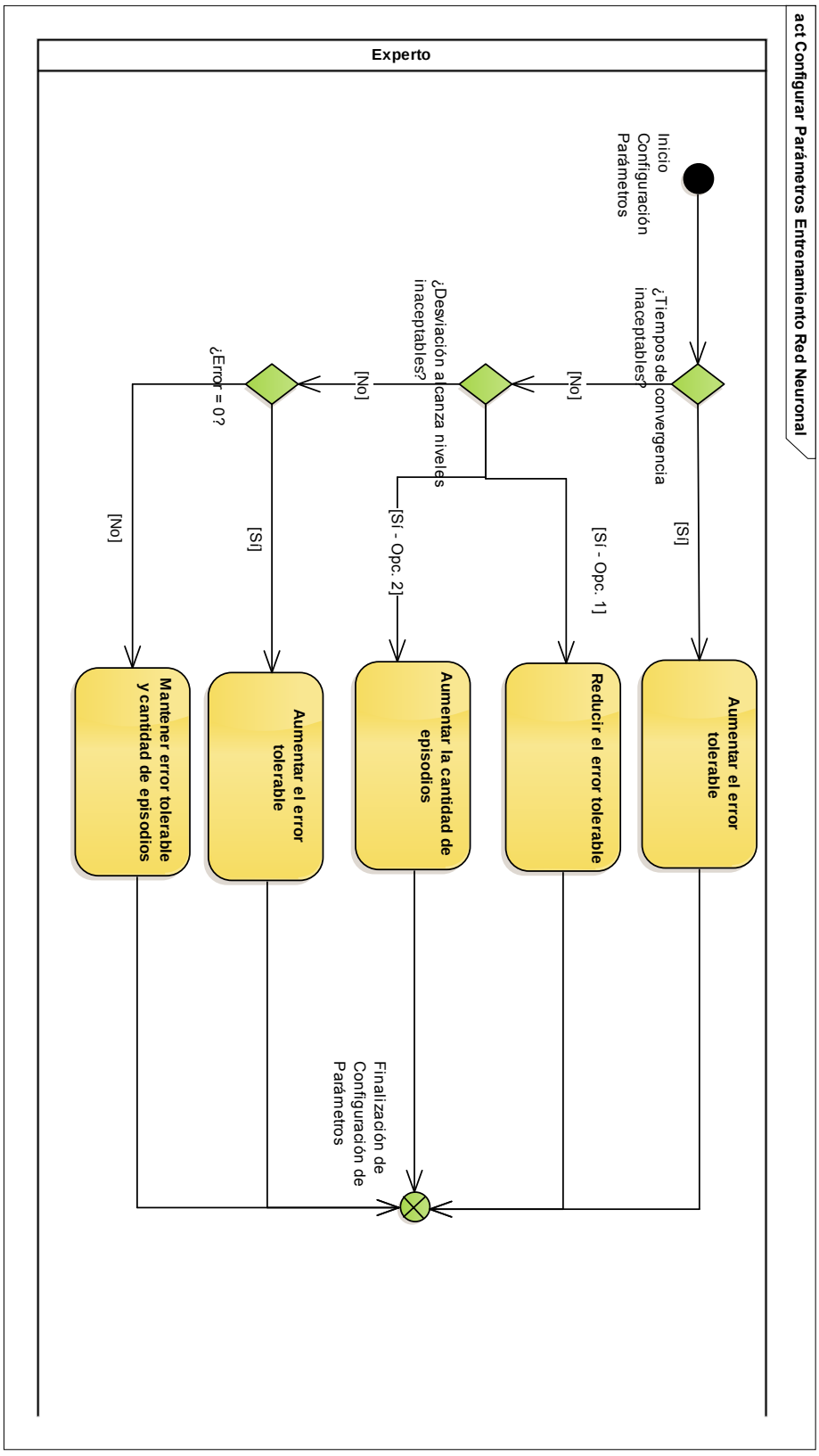
2. Referencias



3. Procedimientos

3.1. Configurar Parámetros de Entrenamiento de Red Neuronal

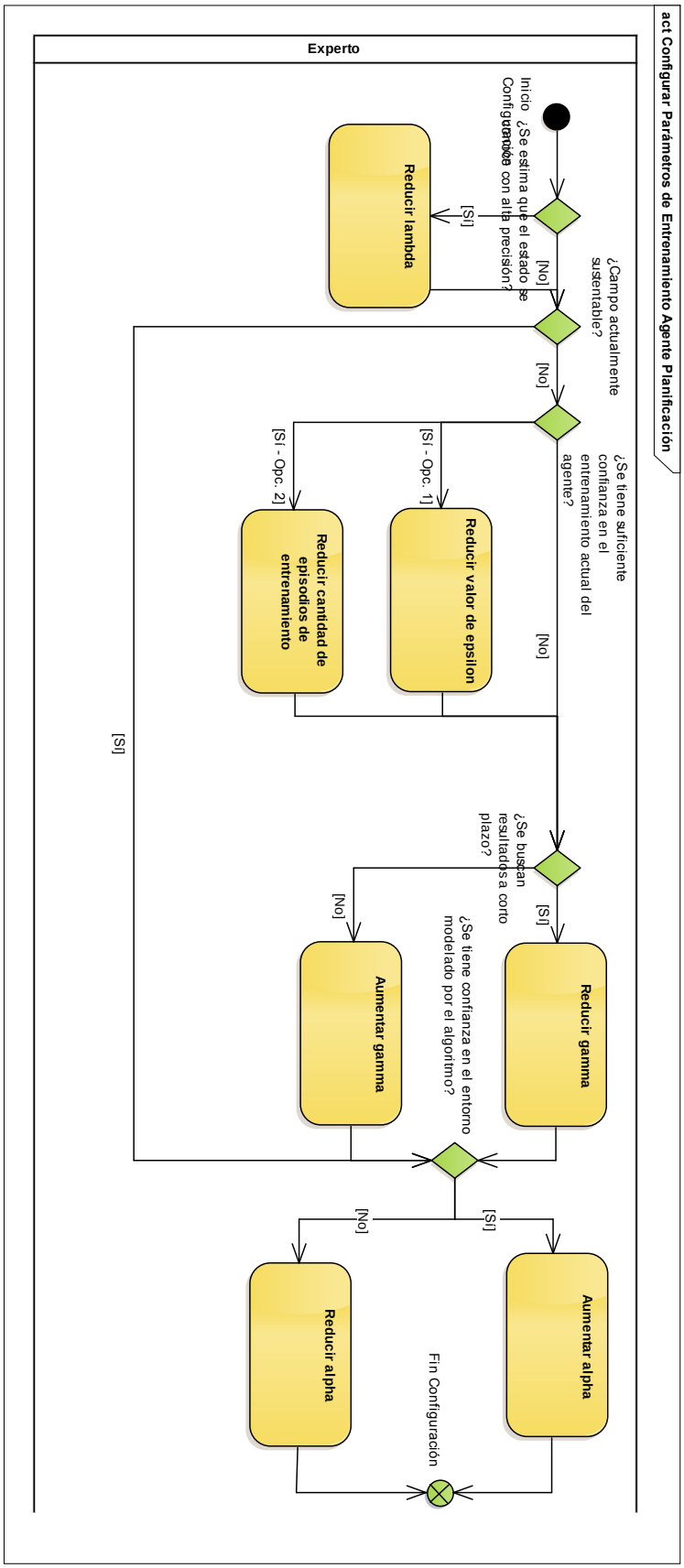
En el presente diagrama se muestran aquellos factores que debe tener en cuenta el experto al seleccionar las variables que establecerán cómo será el entrenamiento de la red neuronal. Las variables más importantes a definir son el error tolerable (que debería partir de un valor muy bajo, por ejemplo 10^{-6}) y la cantidad de episodios de entrenamiento (que debería ser la mayor posible, dependiendo del tiempo que demande la red en ser entrenada). Deben tomarse en cuenta aquellos casos en los que el error cuadrático medio obtenido en el entrenamiento de la red sea cero, ya que eso muy probablemente signifique que la red ha imitado la curva de datos de entrada, arrojando valores altos de error para cualquier consulta fuera de los datos de entrada. Para definir cada uno de los parámetros deben tenerse en cuenta aspectos tales como la satisfacción del Experto con respecto a los tiempos de convergencia, la desviación estándar obtenida al testear la red y la desviación máxima alcanzada, entre otros.


act Configurar Parámetros Entrenamiento Red Neuronal
Experto
Inicio Configuración Parámetros
¿Tiempos de convergencia inaceptables?
[No]
[Si]
¿Desviación alcanza niveles inaceptables?
[No]
[Si - Opc. 1]
[Si - Opc. 2]
¿Error = 0?
[No]
[Si]
Aumentar el error tolerable
Reducir el error tolerable
Aumentar la cantidad de episodios
Aumentar el error tolerable
Mantener error tolerable y cantidad de episodios
Finalización de Configuración de Parámetros

3.2. Configurar Parámetros de Entrenamiento Agente de Planificación

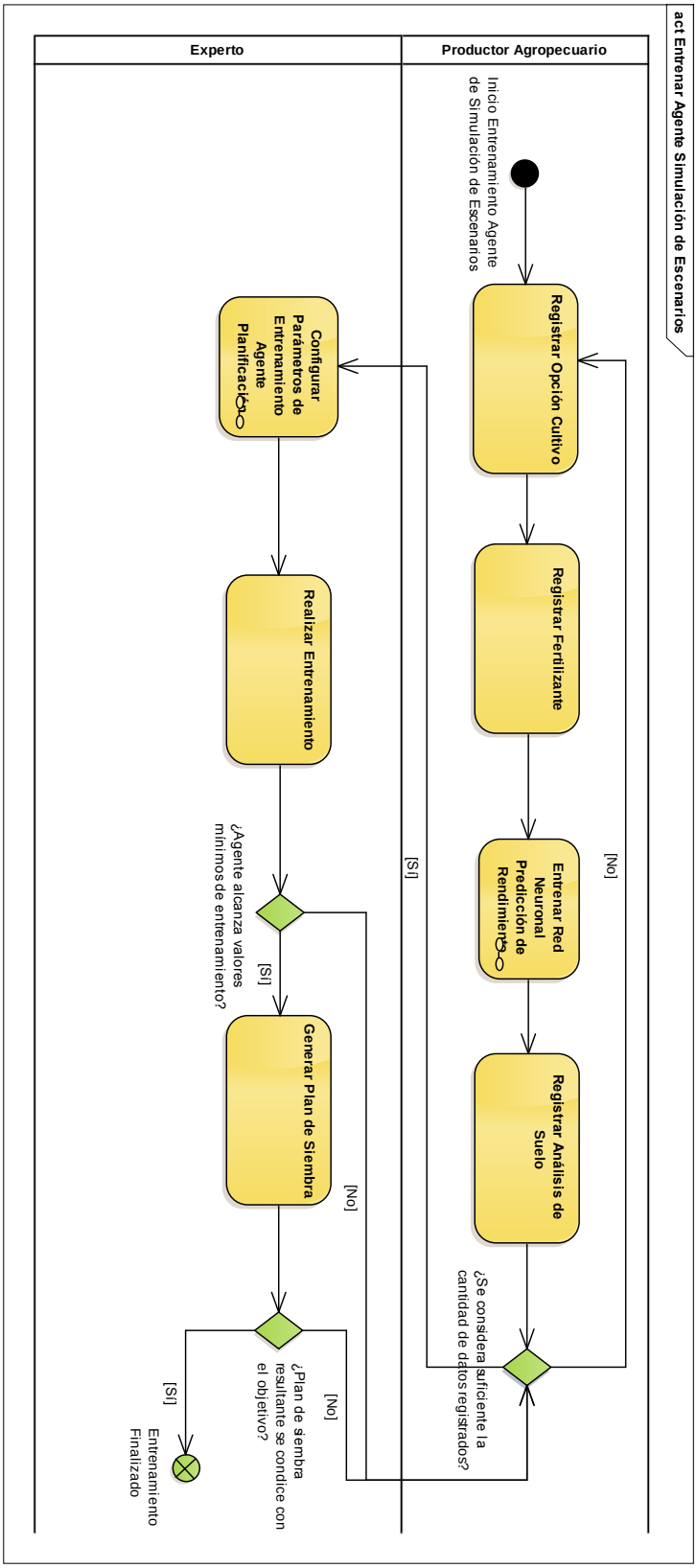
El presente diagrama muestra aquellas alternativas que debe considerar el experto para seleccionar las variables que establecerán cómo será el entrenamiento del agente de planificación. Entre las mismas el experto debe considerar parámetros tales como el factor de descuento (*discount-rate parameter*, γ), el ratio de aprendizaje (*learning rate*, α), el ratio de decaimiento de trazas (*decay rate*, λ) existen consideraciones tales como la satisfacción del Experto con respecto a los tiempos de convergencia, desviación estándar, entre otros aspectos.

El enfoque de modificación de parámetros de entrenamiento que se propone se basa en el cambio de a un parámetro a la vez, realizar reentrenamiento y posterior cambio de parámetros en caso que el problema persista.


act Configurar Parámetros de Entrenamiento Agente Planificación
Experto
Inicio Configuración
¿Se estima que el estado se configuración con alta precisión?
[Sí]
Reducir lambda
[No]
¿Campo actualmente sustentable?
[No]
¿Se tiene suficiente confianza en el entrenamiento actual del agente?
[Sí - Opc. 1]
Reducir valor de epsilon
[Sí - Opc. 2]
Reducir cantidad de episodios de entrenamiento
[No]
[Sí]
¿Se buscan resultados a corto plazo?
[Sí]
Reducir gamma
[No]
Aumentar gamma
¿Se tiene confianza en el entorno modelado por el algoritmo?
[Sí]
Aumentar alpha
[No]
Reducir alpha
Fin Configuración

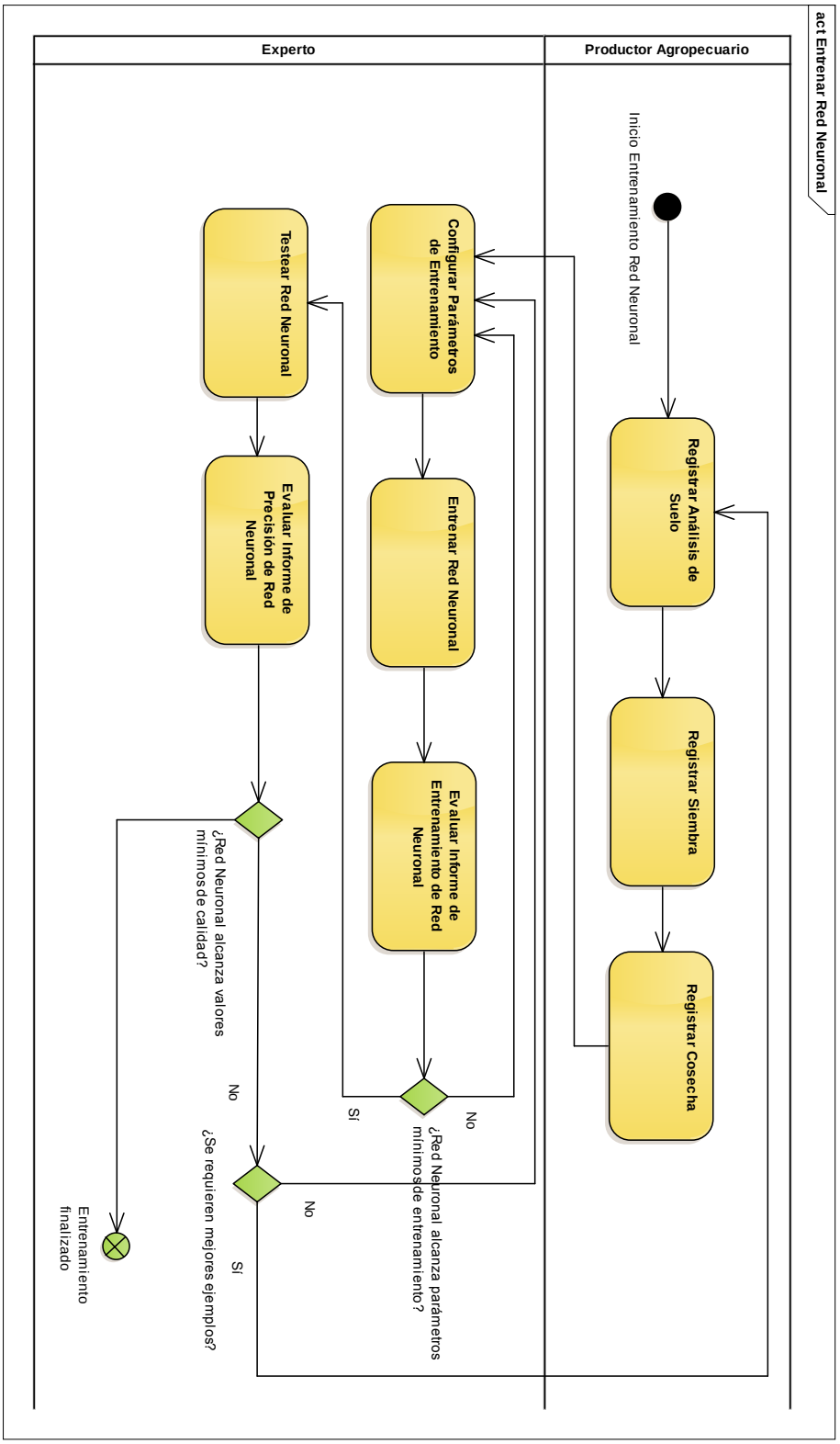
3.3. Entrenar Agente de Simulación de Escenarios

El presente diagrama muestra los procedimientos y alternativas que el experto debe considerar al elegir los parámetros de entrenamiento del agente de simulación de escenarios. Entre los mismos se encuentran, por un lado, la cantidad de datos registrados en el sistema que serán utilizados por el entrenamiento, en donde el experto debe decidir si, de acuerdo con el entrenamiento actual del agente, los mismos se consideran suficientes para poder realizar un buen entrenamiento. Por otro lado, el experto debe juzgar si el agente es capaz de, una vez entrenado, alcanzar valores mínimos de calidad que él considera aceptables.


act Entrenar Agente Simulación de Escenarios
Productor Agropecuario
Experto
Inicio Entrenamiento Agente de Simulación de Escenarios
Registrar Opción Cultivo
Registrar Fertilizante
Entrenar Red Neuronal Predicción de Rendimiento
Registrar Análisis de Suelo
¿Se considera suficiente la cantidad de datos registrados?
[No]
[Si]
Configurar Parámetros de Entrenamiento Agente Planificación
Realizar Entrenamiento
¿Agente alcanza valores mínimos de entrenamiento?
[Si]
[No]
Generar Plan de Siembra
¿Plan de siembra resultante se condice con el objetivo?
[No]
[Si]
Entrenamiento Finalizado

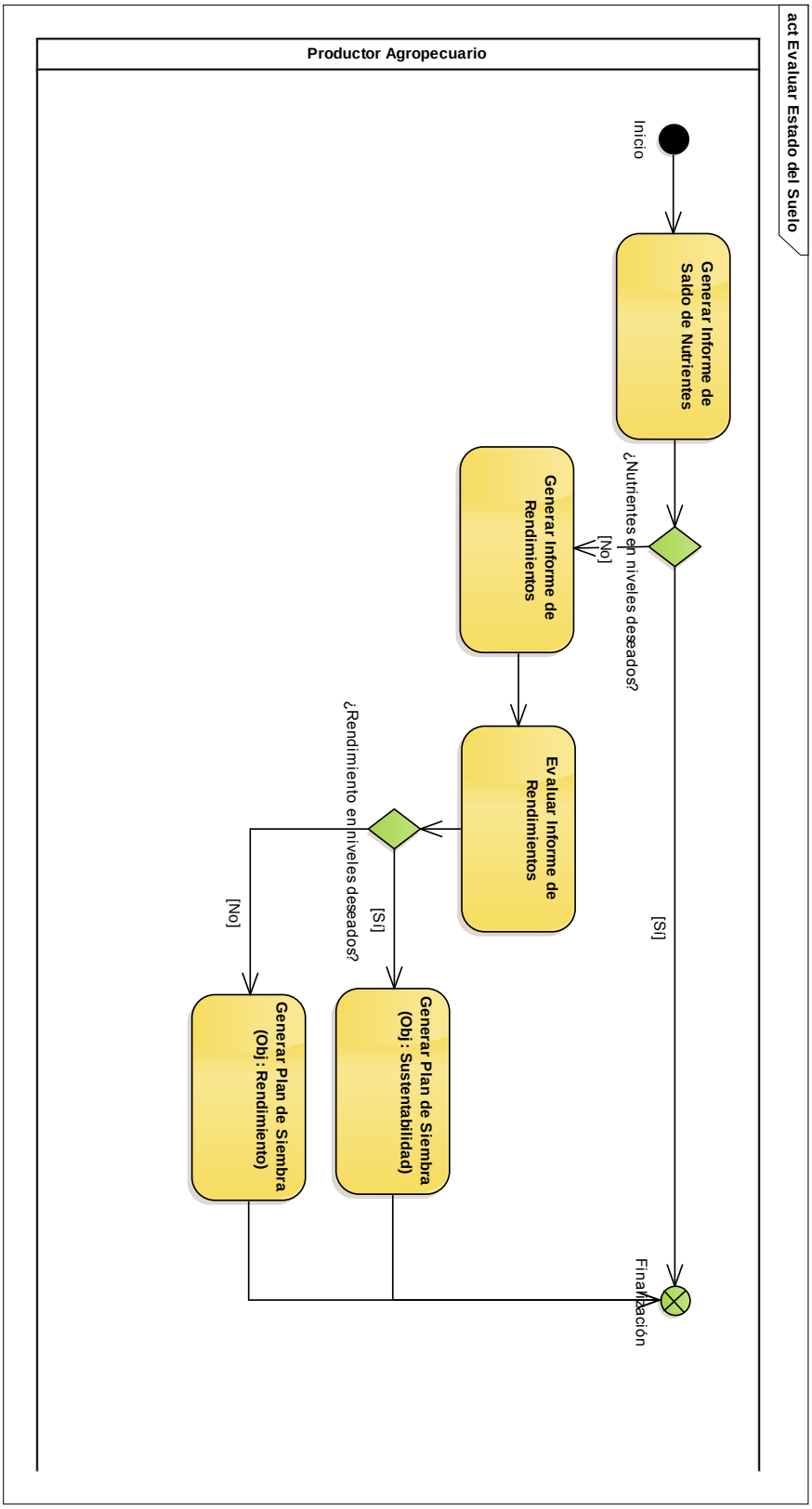
3.4. Entrenar Red Neuronal de Predicción de Rendimiento

El presente diagrama muestra los procedimientos que se siguen para realizar el entrenamiento de una red neuronal (en este caso, la Red Neuronal de Predicción de Rendimiento). A modo general, el diagrama puede dividirse en tres partes: carga de datos, entrenamiento de la red y testing de la misma.


act Entrenar Red Neuronal
Experto
Productor Agropecuario
Inicio Entrenamiento Red Neuronal
Registrar Análisis de Suelo
Registrar Siembra
Registrar Cosecha
Configurar Parámetros de Entrenamiento
Entrenar Red Neuronal
Evaluar Informe de Entrenamiento de Red Neuronal
¿Red Neuronal alcanza parámetros mínimos de entrenamiento?
No
Sí
Testear Red Neuronal
Evaluar Informe de Precisión de Red Neuronal
¿Red Neuronal alcanza valores mínimos de calidad?
No
¿Se requieren mejores ejemplos?
No
Sí
Entrenamiento finalizado

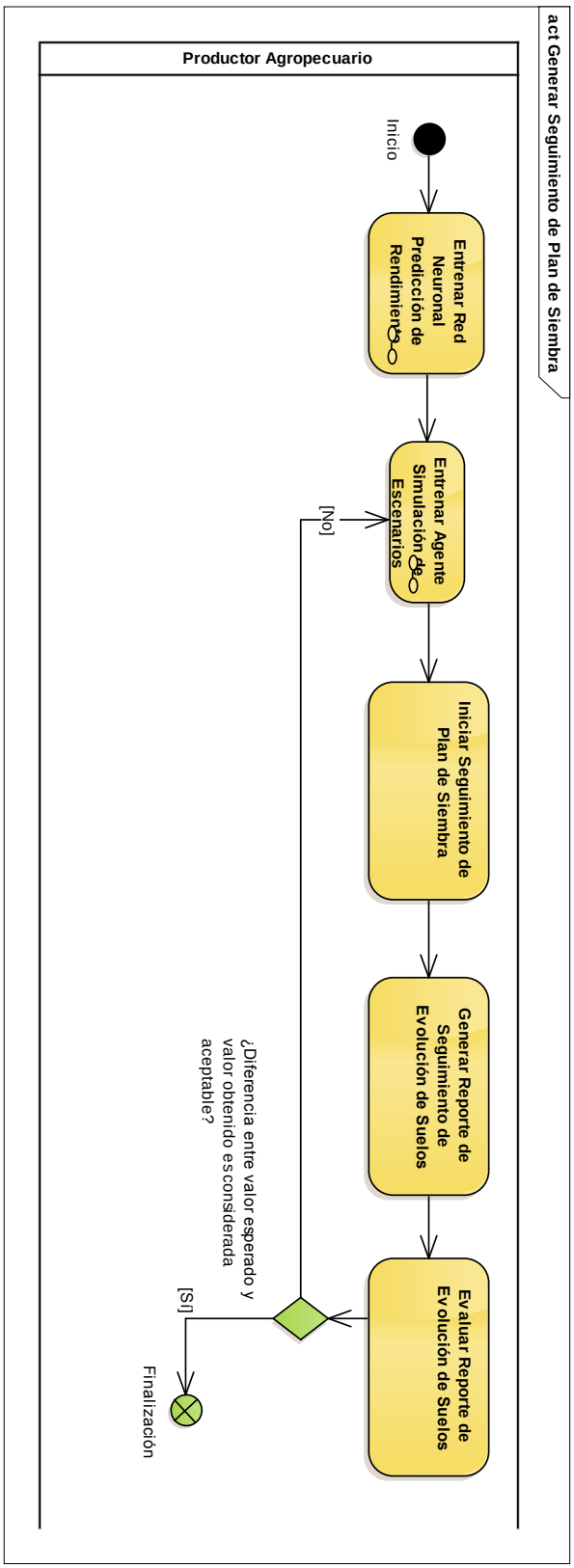
3.5. Evaluar Estado del Suelo

El presente diagrama muestra los procedimientos utilizados para llevar a cabo la evaluación de los nutrientes del suelo en un determinado momento, de acuerdo con los niveles actuales de nutrientes y rendimiento que presente el campo.


act Evaluar Estado del Suelo
Productor Agropecuario
Inicio
Generar Informe de Saldo de Nutrientes
¿Nutrientes en niveles deseados?
[No]
Generar Informe de Rendimientos
Ev aluar Informe de Rendimientos
¿Rendimiento en niveles deseados?
[Si]
[No]
Generar Plan de Siembra (Obj : Sustentabilidad)
Generar Plan de Siembra (Obj : Rendimiento)
[Si]
Finalización

3.6. Generar Seguimiento de Plan de Siembra

El presente diagrama muestra los procedimientos llevados a cabo para generar el seguimiento de un plan de siembra que puede derivar en un reajuste del agente de simulación de escenarios en el caso que el mismo no produzca los resultados esperados.


act Generar Seguimiento de Plan de Siembra
Productor Agropecuario
Inicio
Entrenar Red Neuronal Predicción de Rendimiento
Entrenar Agente Simulación de Escenarios
Iniciar Seguimiento de Plan de Siembra
Generar Reporte de Seguimiento de Evolución de Suelos
Evaluar Reporte de Evolución de Suelos
¿Diferencia entre valor esperado y valor obtenido es considerada aceptable?
[No]
[Si]
Finalización

Capítulo V – Manual de Usuario

1. Introducción

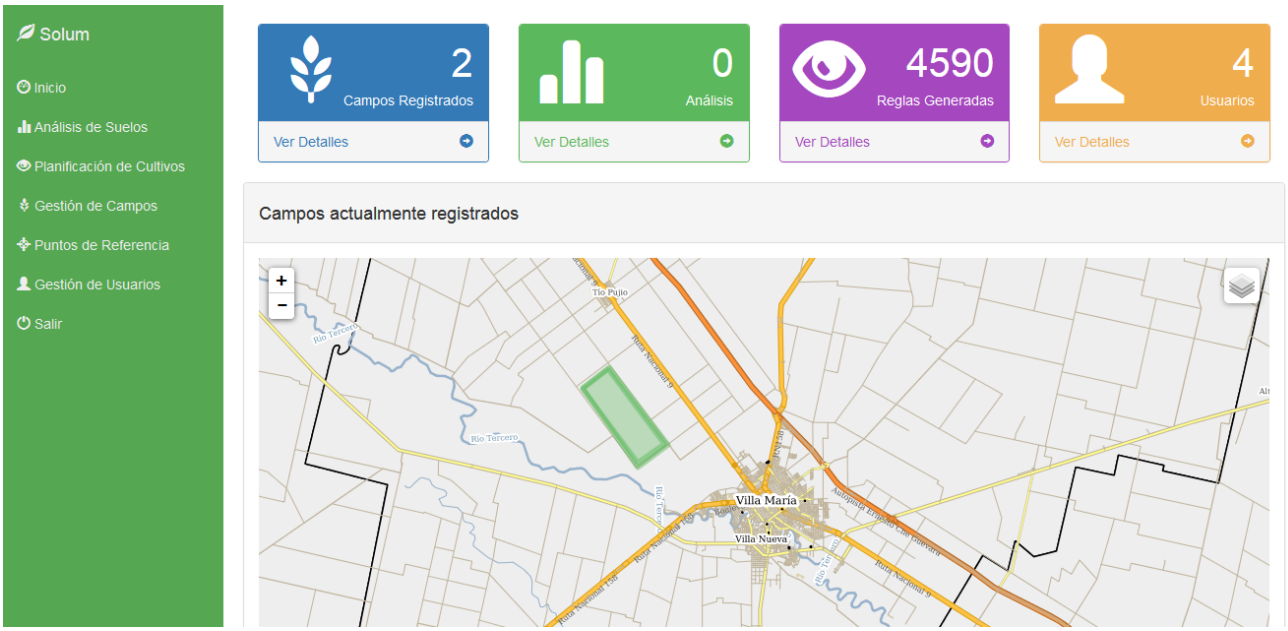
El presente documento tiene como objetivo brindarle asistencia al usuario en el uso del sistema de información Solum. A tal efecto, se muestran indicaciones paso a paso de las operaciones más comunes provistas por Solum.

2. Inicio de Sesión

La pantalla Inicio de Sesión es la primera interfaz que se le presenta a todo usuario. En la misma, el usuario debe ingresar su información de inicio de sesión para poder acceder al Sistema. Adicionalmente, puede optar por que sus datos de inicio de sesión sean recordados la próxima vez que acceda al Sistema.

2. Menú Principal

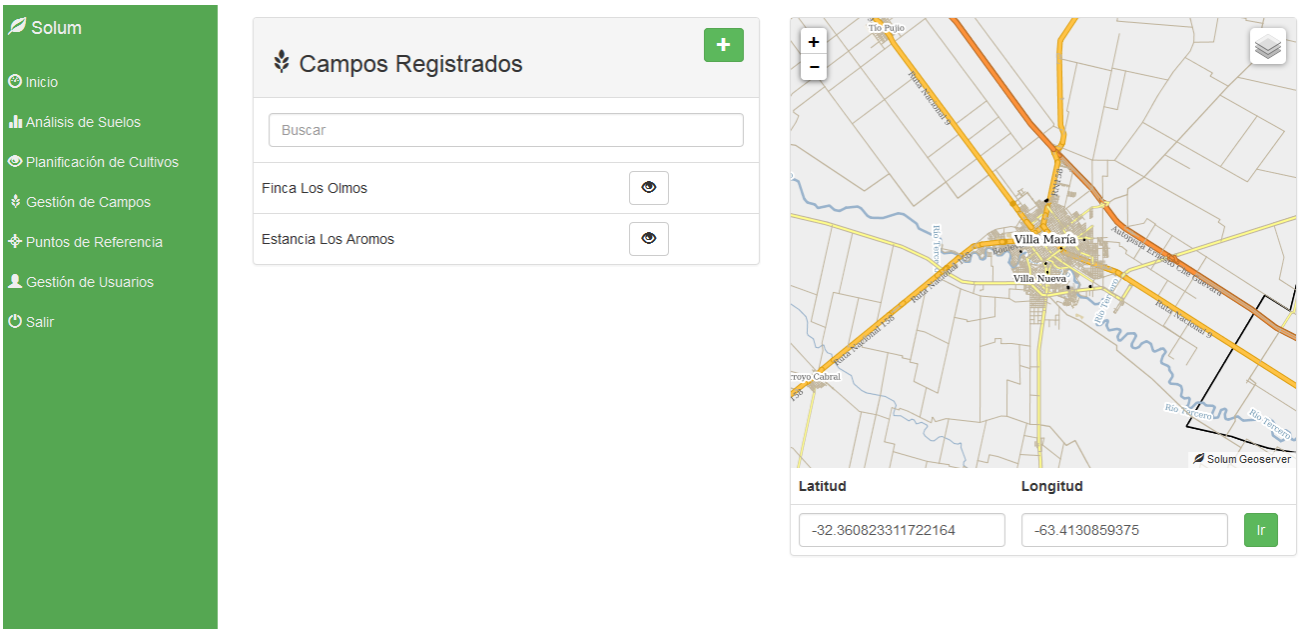
El menú principal muestra una vista general de los campos registrados en el Sistema, mostrando su ubicación en un mapa. Por encima del mapa se muestra un resumen de la información más relevante del Sistema: la cantidad de campos registrados, la cantidad de análisis de suelos registrados, la cantidad de reglas generadas por el Agente de Planificación de Cultivos y la cantidad de usuarios habilitados a usar el Sistema. El usuario puede hacer clic en “Ver Detalles” para más información sobre el aspecto deseado. Por otra parte, a la izquierda puede verse un panel que proporciona links para acceder a cualquier parte deseada del Sistema.



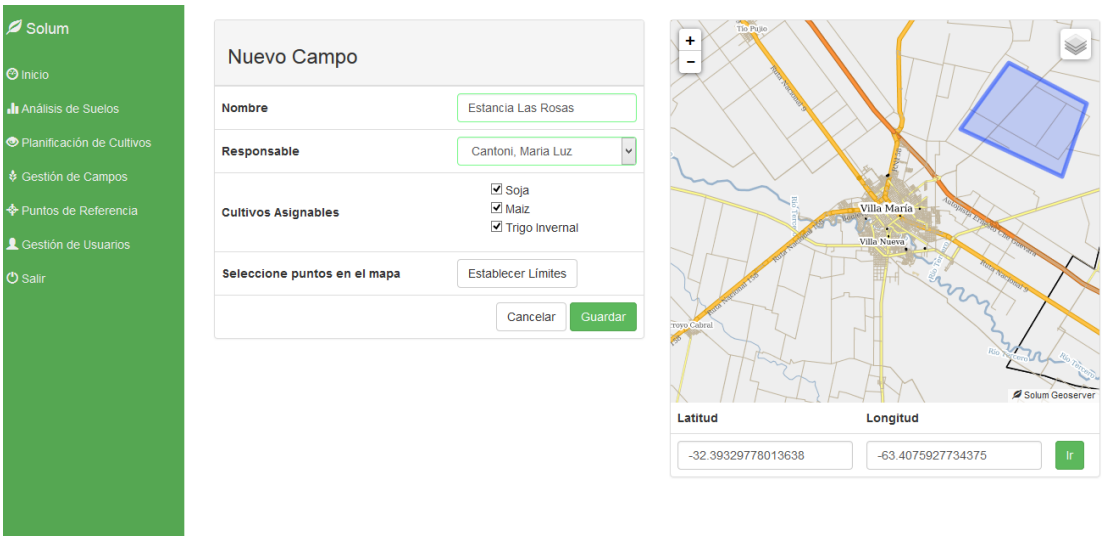
3. Gestión Ambiental

3.1. Gestión de Campos

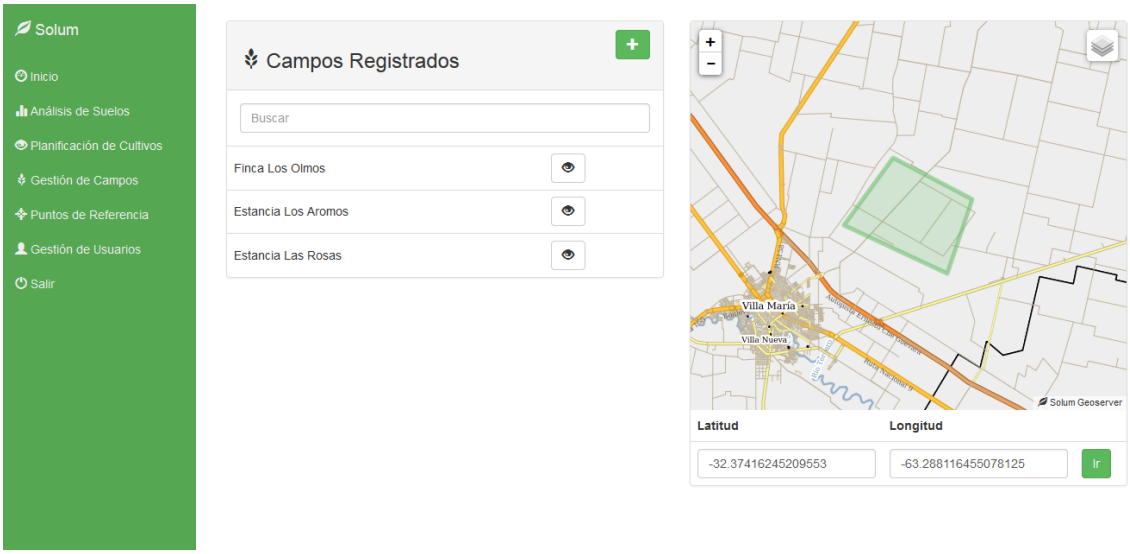
En la pantalla Gestión de Campos se muestra un resumen de todos los campos registrados en el Sistema, junto con un mapa que puede utilizarse como referencia para ver los demás campos cargados y establecer límites a los nuevos. Por debajo del mapa es posible solicitarle al mismo que se centre en una determinada latitud y longitud, escribiendo tales valores de forma manual y haciendo clic en el botón “Ir” ubicado a su izquierda. Para filtrar el nombre de los campos registrados con algún nombre deseado, sólo basta con escribir en el campo “Buscar”. Al ingresar cada carácter, automáticamente aparecerán los nombres de los campos que coincidan con la misma. Para realizar la carga de un nuevo campo debe hacerse clic en el botón “+” ubicado a la derecha de “Campos Registrados”.



Tras ello, el Sistema muestra la información necesaria para la realizar la creación de un nuevo campo. El usuario debe completar los campos “Nombre” y “Responsable”, junto con la selección de los cultivos asignables al campo. Adicionalmente, el usuario debe trazar cada uno de los puntos del polígono que conforma el área de su campo. Una vez que terminó con los mismos, presionar el botón “Establecer Límites” hará que los límites del campo queden establecidos, formando un polígono de puntos, tal como el rectángulo azul que puede verse en el mapa.

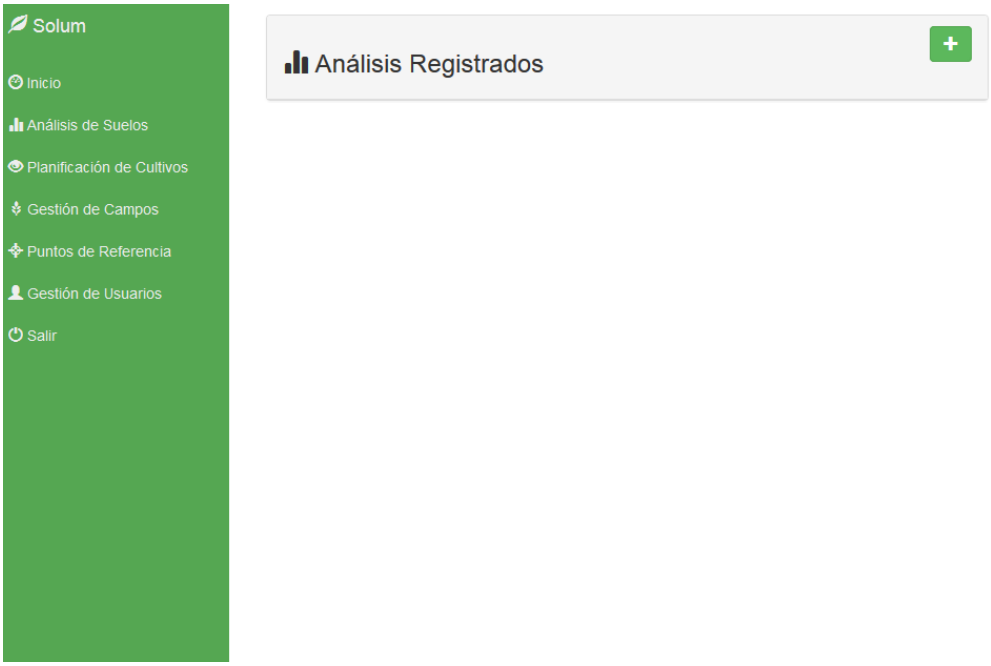


Una vez que el usuario presiona “Guardar”, el campo queda guardado, agregándose en la lista, tal como se muestra en la imagen a continuación. Si el usuario deseara cancelar la carga del campo, en cualquier momento puede presionar el botón “Cancelar” para descartar todos los cambios y volver a Gestión de Campos.

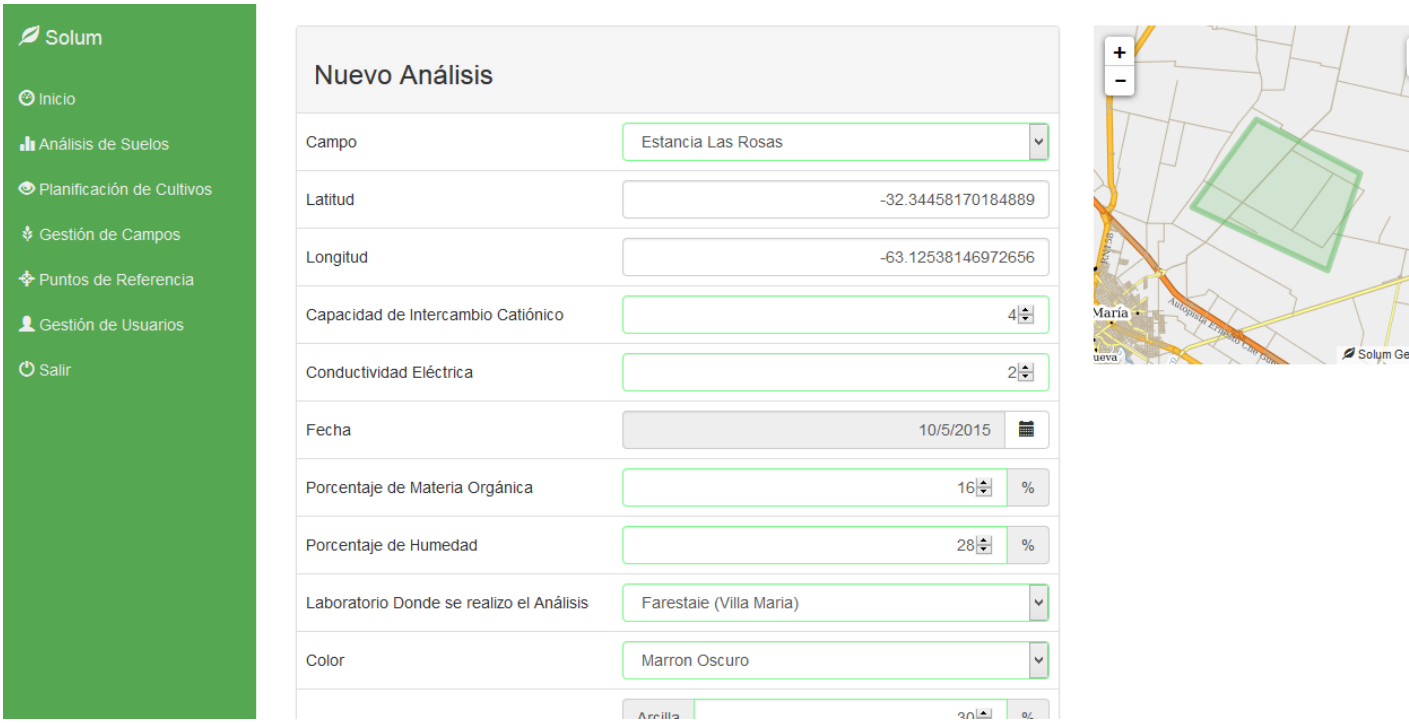


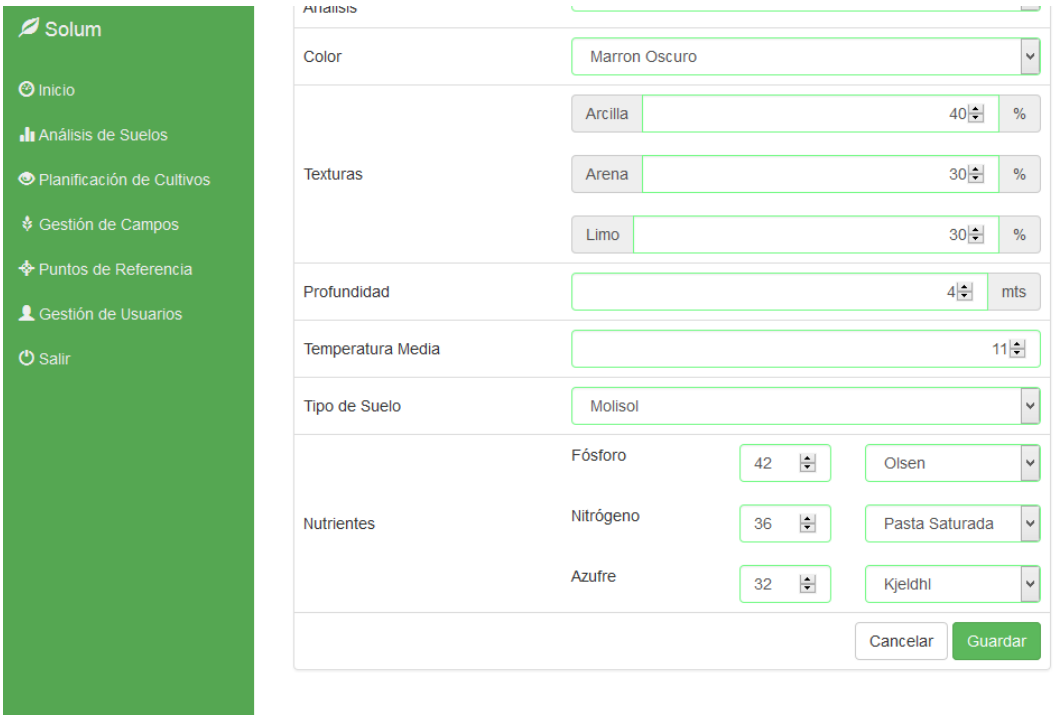
3.2. Gestión de Análisis de Suelos

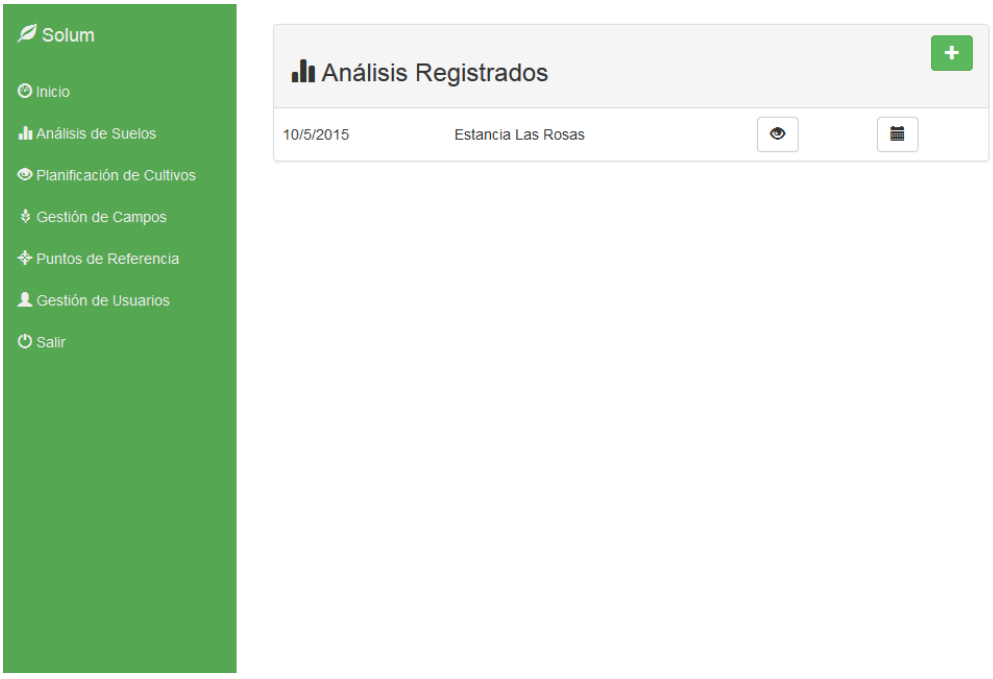
En la pantalla Gestión de Análisis de Suelos pueden verse todos los análisis de suelo disponibles en el Sistema (de existir alguno). Para agregar un análisis de suelo, sólo basta con hacer clic en el botón “+” ubicado a la derecha de “Análisis Registrados”.



El usuario debe completar cada una de las distintas características del análisis de suelo. Una vez que el Campo quede seleccionado, el mismo pasará a ser mostrado por el mapa. Para establecer la latitud y longitud, el usuario sólo tiene que elegir un punto en el mapa dentro del área del campo que seleccionó. Una vez completados todos los campos presiona el botón “Guardar”.



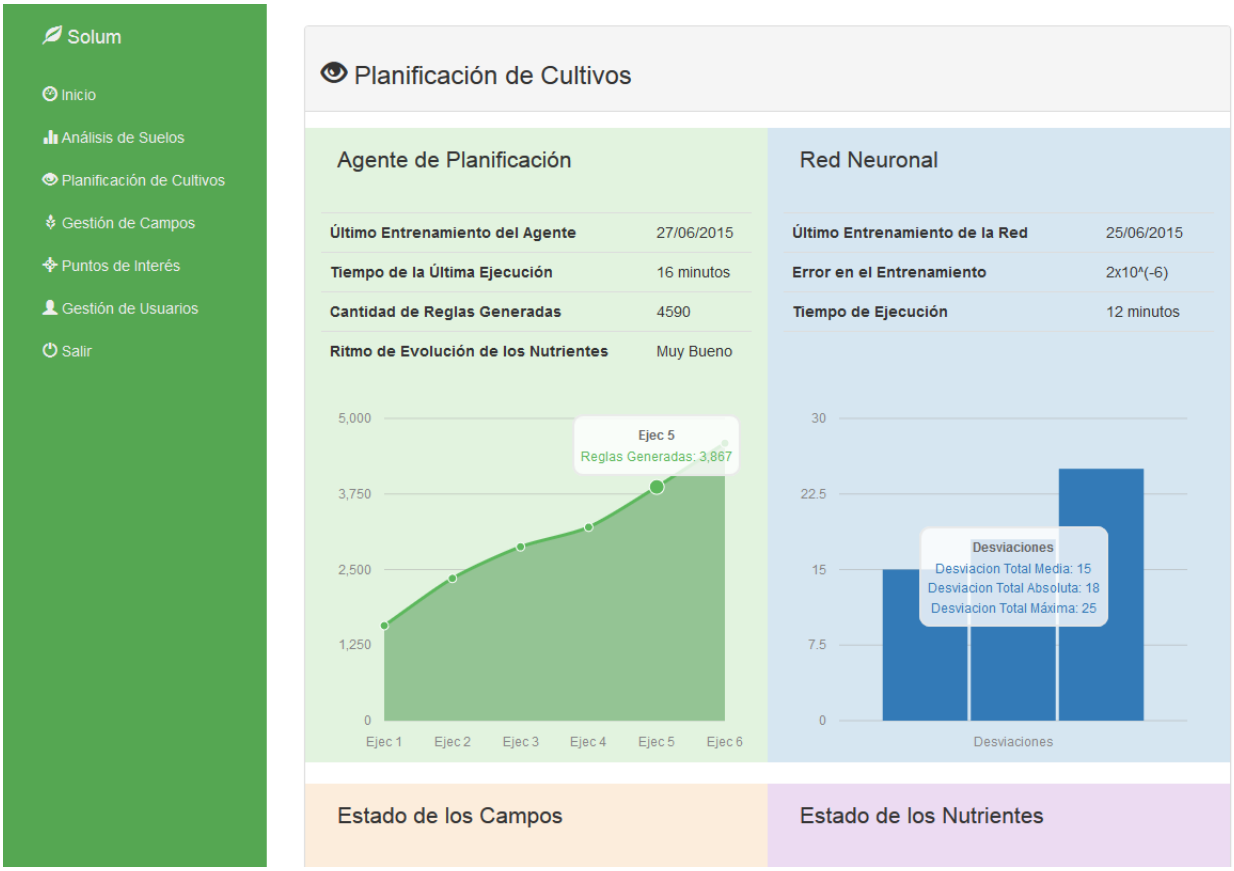

Solum
Inicio
Análisis de Suelos
Planificación de Cultivos
Gestión de Campos
Puntos de Referencia
Gestión de Usuarios
Salir
Color
Marron Oscuro
Texturas
Arcilla
40
%
Arena
30
%
Limo
30
%
Profundidad
4
mts
Temperatura Media
11
Tipo de Suelo
Molisol
Nutrientes
Fósforo
42
Olsen
Nitrógeno
36
Pasta Saturada
Azufre
32
Kjeldhl
Cancelar
Guardar


Solum
Inicio
Análisis de Suelos
Planificación de Cultivos
Gestión de Campos
Puntos de Referencia
Gestión de Usuarios
Salir
Análisis Registrados
10/5/2015
Estancia Las Rosas

4. Planificación de Cultivos

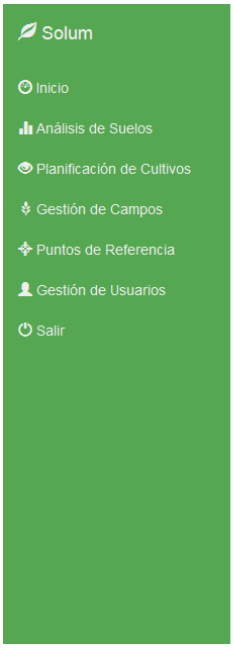
4.1. Vista General

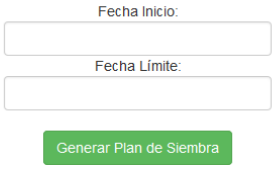
En la presente pantalla se muestra un resumen del estado general de los suelos que permite al usuario tomar decisiones sobre cómo llevar a cabo la planificación de cultivos. Se destacan cuatro paneles: “Agente de Planificación”, que detalla el estado actual del agente Soar. Por otra parte, “Red Neuronal”, detalla el estado actual de entrenamiento de la red neuronal mientras que “Estado de los Campos” muestra el rendimiento que en promedio alcanzan los campos, así como el nivel de adhesión a los planes de siembra que presentan los mismos junto con el nivel de salinidad. Por último, “Estado de los Nutrientes” muestra aquella información referida al estado de los nutrientes más importantes de los campos.



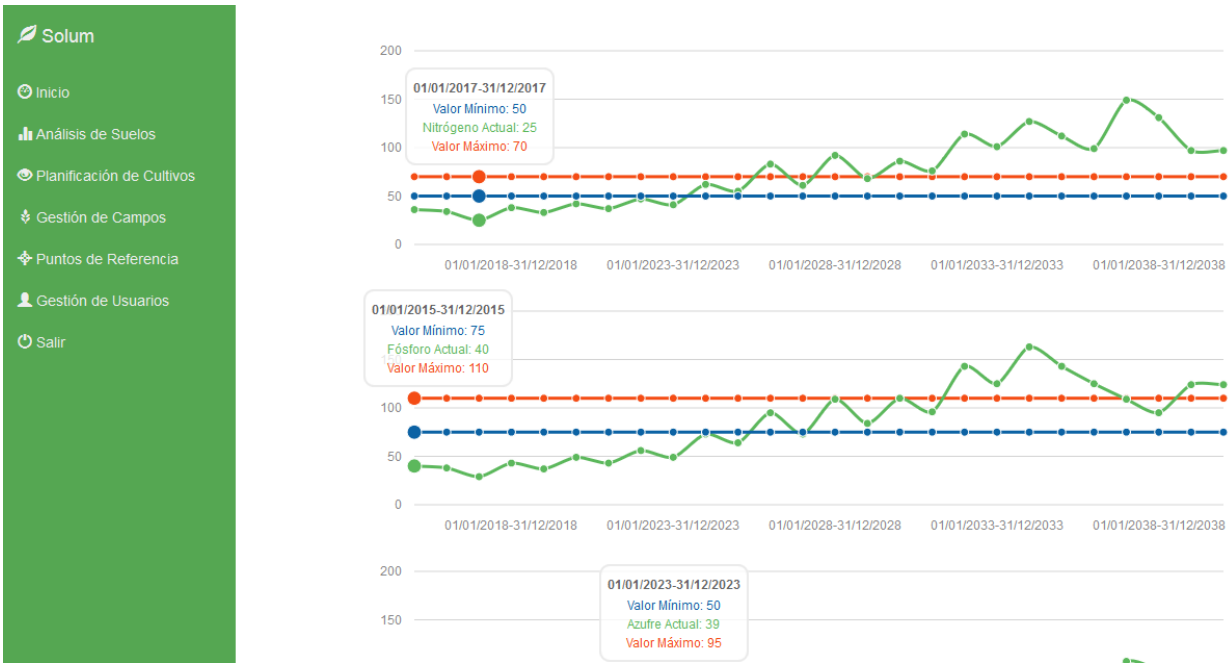
4.2. Obtener Plan de Siembra

A partir de la actual pantalla es posible obtener un plan de siembra. Para ello debe especificarse cuál es la fecha inicial de la simulación, junto con el intervalo máximo en el que la misma debe alcanzar su objetivo. Definidas ambas fechas, ya puede iniciarse el cálculo del Plan de Siembra mediante el botón “Obtener Plan de Siembra”.





Tras calcularse el plan de siembra, la siguiente pantalla muestra el resultado del mismo, en donde puede verse en detalle cada una de las siembras y cosechas realizadas durante el transcurso de tiempo que el agente se toma en alcanzar su objetivo. Tras cada cosecha, puede verse el estado esperado que tendrán los nutrientes del suelo según el agente de replanificación. Por otra parte, puede observarse gráficamente cuáles son los distintos valores que cada uno de los nutrientes tomará tras las respectivas siembras, cosechas y fertilizaciones luego realizadas por año.

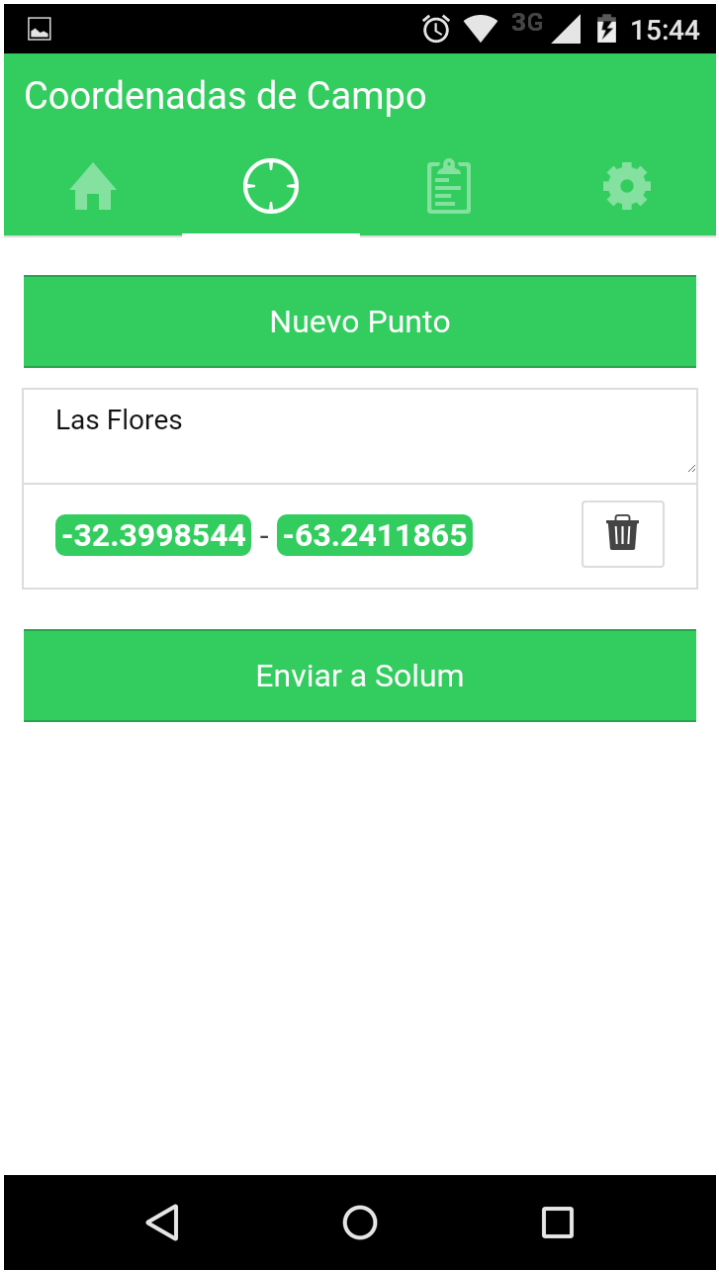


5. Aplicación Móvil

5.1. Vista General

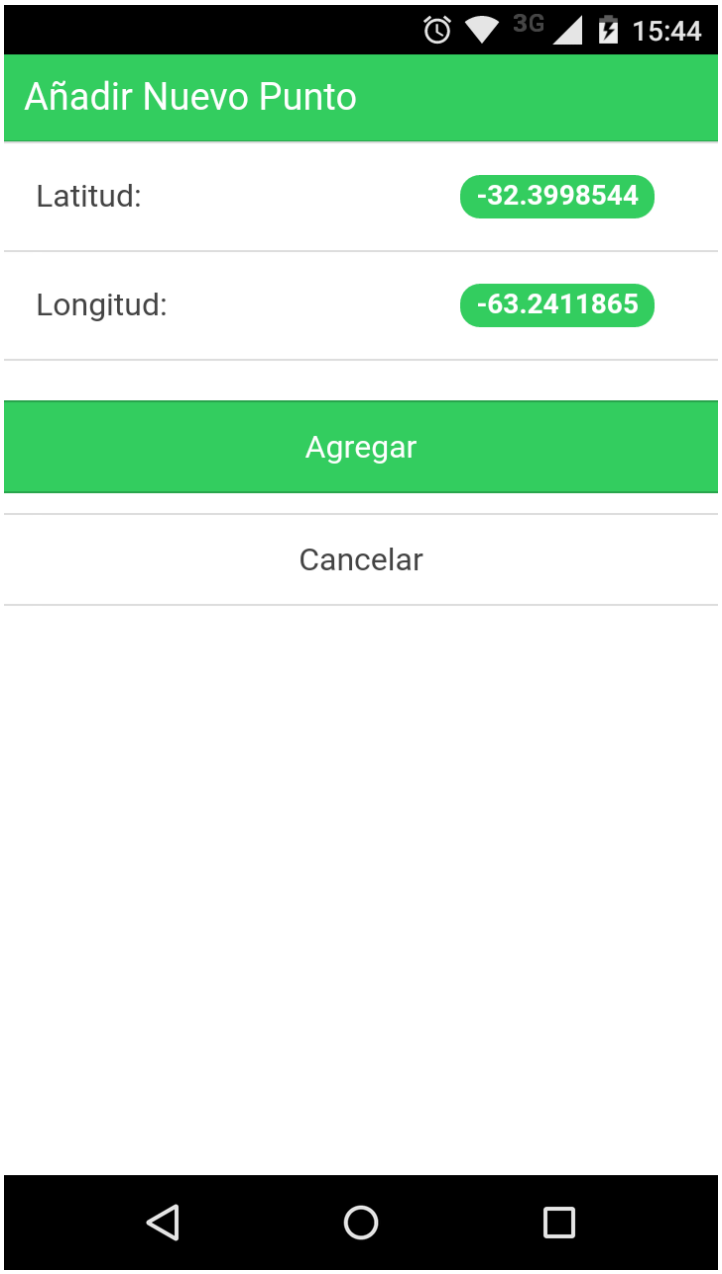
La Aplicación Móvil de Solum le brinda al ingeniero agrónomo o productor una forma práctica de interactuar directamente con el Sistema desde su campo utilizando un Smartphone.

Una de las funciones destacadas en la aplicación consiste en guardar aquellos puntos que conforman el polígono del campo, para poder llevar a cabo la carga del mismo en el Sistema. Para realizar la carga de cada uno de los puntos, sólo basta con ingresar a la pestaña “Coordenadas de Campo” y marcar cada uno de los puntos que conforman los límites del campo a través del botón “Nuevo Punto”.



Al presionar la opción “Nuevo Punto”, el Sistema procederá a utilizar el GPS para calcular las coordenadas, las cuales son mostradas a continuación. Si el punto es satisfactorio, el botón “Agregar” guardará temporalmente la latitud y longitud

obtenidas. Presionar “Enviar a Solum” llevará a cabo el envío de las mismas a los servidores de Solum, para que desde ellas se cree un nuevo campo, si es que así se lo desea.

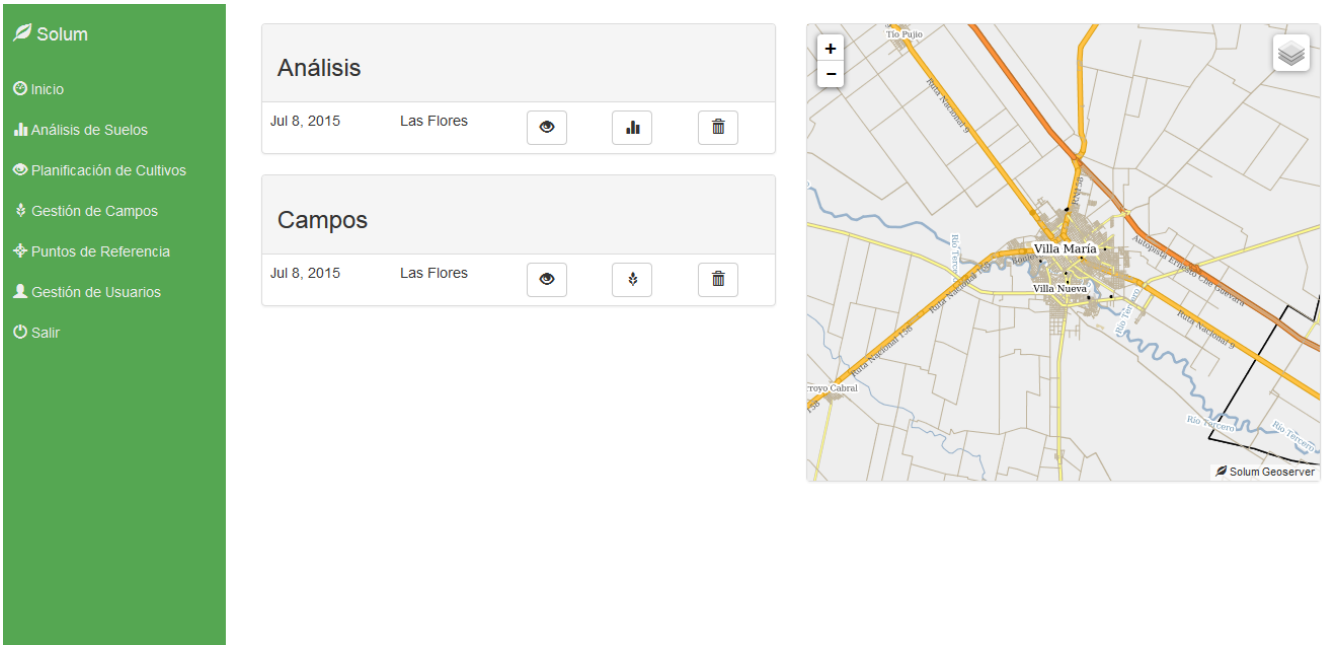


Otra función importante que brinda la aplicación es la posibilidad de marcar las coordenadas en donde ha sido extraída la muestra del análisis de suelo. Esta función se ubica en la tercera pestaña de la aplicación, “Punto del Análisis”. Al igual que como sucede con “Punto de Campo”, al presionar “Calcular Posición”, la aplicación usará el GPS para determinar la localización en latitud y longitud. Agregadas las mismas, el botón “Enviar a Solum” lleva a cabo su envío al servidor Solum.

La cuarta pestaña de la aplicación permite configurar las opciones de inicio de sesión y la dirección web en donde se encuentra el servidor Solum. En el caso que las mismas deban modificarse, basta con introducir un nuevo usuario y contraseña y/o dirección web, para luego guardar tales opciones seleccionando “Guardar Configuración”.

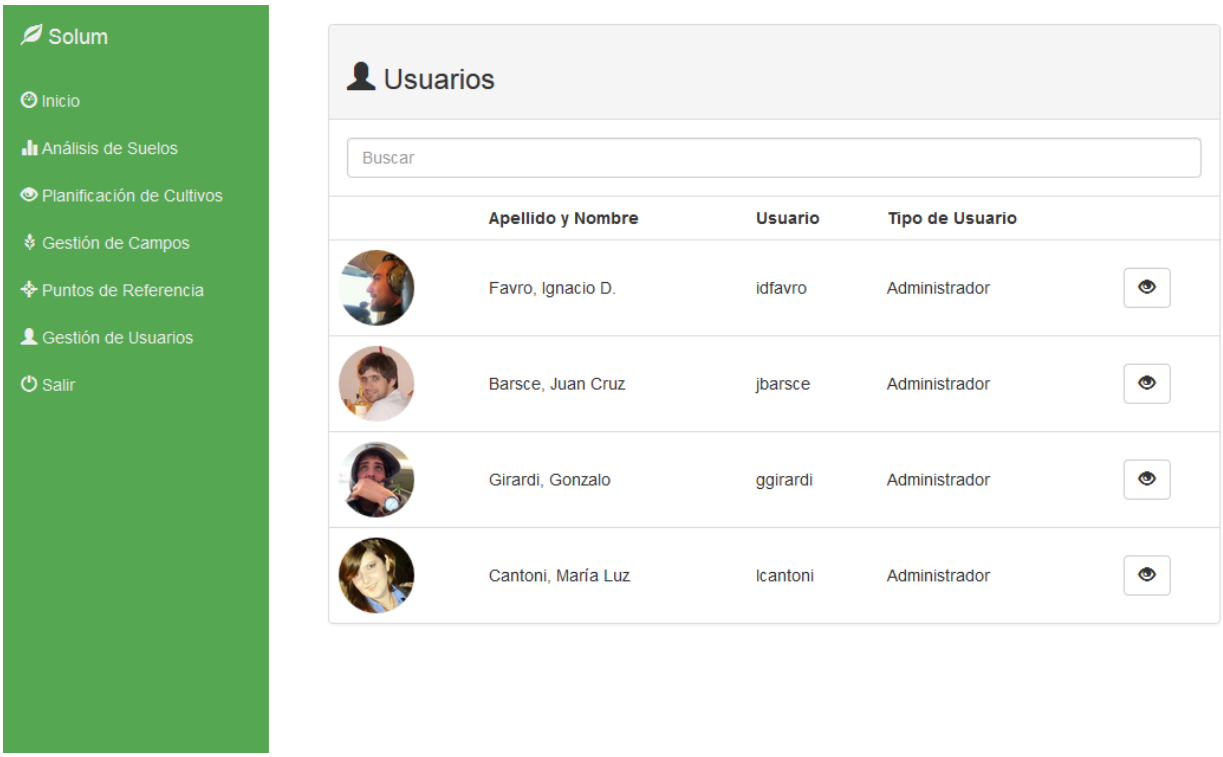
5.2. Puntos de Referencia

En la pantalla Puntos de Referencia se listan todos los puntos guardados desde la aplicación móvil. Si el usuario lo desea, puede ver desde qué coordenadas los mismos fueron creados. Adicionalmente, puede crear un análisis de suelo o un campo a partir de los puntos de análisis y campos guardados, respectivamente.



6. Gestión de Usuarios

La pantalla Gestión de Usuarios muestra el listado de cada uno de los usuarios del sistema, junto con su respectivo tipo de usuario.



Con el objetivo de prestar un conjunto de servicios apropiado para cada necesidad, el sitio clasifica a los distintos usuarios en tres tipos, listados a continuación.

- Administrador. Es quien que realiza la administración general del sistema, asegurando su correcto funcionamiento para sus usuarios y las actividades que ellos realizan. Posee acceso completo a toda la funcionalidad existente en el sistema, incluyendo la administración de los demás usuarios.
- Productor Agropecuario. Aquellos usuarios que se encargan de todo aspecto relativo de la administración y carga de datos relacionados con el campo. tales como las localidades, provincias y países; campos; componentes; fertilizantes; nutrientes; cultivos y tipos de cultivos; análisis de suelo, así como de la inicialización y finalización de los seguimientos del suelo y la generación de planes de siembra en base al conocimiento previamente aprendido.
- Experto. Responsable de controlar cómo se lleva a cabo el entrenamiento y desarrollo del agente, verificando la calidad de sus deducciones y

decisiones, y haciendo uso del sistema para corregir y mejorar las políticas de decisión.

7. Preguntas Frecuentes

¿El uso de técnicas de inteligencia artificial (IA) para generar planes de siembra reemplaza a la opinión de un experto?

El uso de técnicas de IA sirve como complemento para asistir a la toma de decisiones de un experto. El mismo debe verificar que las decisiones que sugiere el sistema cuenten con un nivel considerable de precisión vayan en concordancia con su plan a largo plazo para el campo.

¿Qué determina la calidad de la generación de planes de siembra?

La calidad en la predicción de los planes de siembra está ligada a la cantidad, representatividad y calidad de los ejemplos con los que se entrene al agente. A grandes rasgos, aportarán una mayor calidad la inclusión de una gran cantidad de ejemplos heterogéneos, que muestren cómo el agente realiza predicciones a partir de ejemplos de todo tipo de campos de la región, desde aquellos que tienen un pobre balance de nutrientes en el suelo hasta aquellos con buenos balances.

¿Qué determina la calidad de la predicción de las redes neuronales?

Al igual que sucede con los planes de siembra, la calidad de la predicción en las redes neuronales, especialmente para la redes de rendimiento y de extracción depende principalmente de la calidad de los ejemplos utilizados. Se recomienda usar una amplia gama de ejemplos que muestren el comportamiento de varios fertilizantes en suelos que presenten variaciones en sus rangos de nutrientes. Adicionalmente, se recomienda entrenar el sistema para que se enfoque en ejemplos provenientes de un tipo de suelo particular, ya que por cada tipo de suelo que se añada a los ejemplos se requerirá de ejemplos de fluctuaciones de nutrientes y rendimientos particulares a ese tipo de suelo.

¿Cuándo puede considerarse que la red neuronal ha finalizado su entrenamiento?

La decisión debe ser tomada por el experto. Él será quien compare los resultados de predicción esperados contra los resultados obtenidos y determine si los mismos tienen un error tolerable (error cuadrático medio, desviación estándar máxima, etc.) para poder ser tenidos en cuenta, o si la red requiere de entrenamiento adicional.