Sistema de Simulación para la Gestión Coordinada de la Producción de Asociaciones Piscícolas Pequeñas y Medianas en Colombia

Juan Sebastián Mantilla Serrano y Hammer Ronaldo Muñoz Hernández

Trabajo de Grado para Optar al Título de Ingeniero de Sistemas e Informática

Director

Hugo Hernando Andrade Sosa

Magíster en Informática

Universidad Industrial de Santander

Facultad de Ingenierías Físico-Mecánicas

Escuela de Ingeniería de Sistemas e Informática

Ingeniería de Sistemas

Bucaramanga

2026

Agradecimientos

Queremos expresar nuestra más sincera gratitud a nuestro director, el profesor Hugo Hernando Andrade Sosa, cuyo acompañamiento, rigor y visión fueron determinantes en cada etapa de este trabajo. Hacemos extensivo este reconocimiento al grupo de investigación SIMON, en cuya línea de investigación se enmarca este proyecto y sobre cuyo trabajo previo se cimienta el nuestro.

Agradecemos también a la Universidad Industrial de Santander y a la Escuela de Ingeniería de Sistemas e Informática por la formación que hizo posible llegar hasta aquí, así como a los profesores y jurados cuyas observaciones contribuyeron a fortalecer este documento.

A título personal, agradezco a mi tía por su invaluable apoyo durante el desarrollo de este trabajo de grado, así como por la guía y el acompañamiento que ha brindado a mi formación tanto personal como intelectual. A mis padres, por su amor incondicional, su respaldo constante y por acompañarme en cada etapa de este camino. Asimismo, agradezco a mi abuela por su compañía, afecto y apoyo a lo largo de este proceso.

Juan Sebastián Mantilla Serrano

A título personal, expreso mi más profundo agradecimiento a mi mamá y a mi hermana, por su amor, apoyo incondicional y confianza en cada etapa de este proceso. Gracias por su paciencia, sus palabras de aliento y por acompañarme en todo momento. Sus enseñanzas y valores han sido fundamentales en mi formación personal y profesional.

Hammer Ronaldo Muñoz Hernández

Tabla de Contenido

Pág.

Lista de Tablas

Pág.

Lista de Figuras

Pág.

Lista de Apéndices

Pág.

Resumen

Título: Sistema de simulación para la gestión coordinada de la producción de asociaciones piscícolas pequeñas y medianas en Colombia [*]

Autores: Juan Sebastián Mantilla Serrano, Hammer Ronaldo Muñoz Hernández [**]

Palabras Clave: Simulación, Piscicultura, Dinámica de Sistemas, Redes Neuronales, Horizonte deslizante, Optimización Combinatoria, Programación Lineal Entera-Mixta, Asociatividad.

Descripción:

La piscicultura en Colombia es practicada mayoritariamente por pequeños productores que operan de forma aislada, lo que limita su capacidad para atender demandas de mercado que individualmente resultan inalcanzables, además de generar períodos de inactividad productiva que erosionan la rentabilidad de su infraestructura instalada. Como respuesta a esta problemática, el grupo de investigación SIMON de la Universidad Industrial de Santander ha desarrollado una línea de investigación orientada a tecnificar y democratizar el acceso a herramientas de gestión piscícola para el campesinado colombiano. En este marco, el proyecto inmediatamente precedente formalizó un modelo de Dinámica de Sistemas para la simulación biológica de un estanque individual e integró redes neuronales para el pronóstico del peso y la ración requerida.

El presente trabajo extiende este fundamento hacia la escala asociativa mediante una plataforma de simulación y optimización que coordina los ciclos productivos de múltiples estanques pertenecientes a diferentes productores para garantizar una producción continua frente a una demanda periódica. La plataforma integra tres subsistemas: un motor de simulación que orquesta la dinámica biológica de múltiples estanques mediante un horizonte deslizante con simulación de Monte Carlo; un módulo de ciencia de datos basado en una red neuronal recurrente LSTM que predice el incremento de peso y la ración requerida; y un módulo de optimización matemática que genera el cronograma óptimo de siembras y cosechas minimizando las desviaciones de suministro frente a la demanda bajo restricciones de capacidad y calidad comercial.

El sistema constituye un aporte concreto a la línea de investigación del grupo SIMON orientada al fortalecimiento tecnológico del sector piscícola colombiano, y establece la base sobre la cual escalar hacia una herramienta de apoyo a la toma de decisiones para asociaciones de piscicultores de pequeña y mediana escala.

---

[*] Trabajo de Grado.

[**] Facultad de Ingenierías Físico-Mecánicas. Escuela de Ingeniería de Sistemas e Informática. Director: Hugo Hernando Andrade Sosa, M.Sc.

Abstract

Title: Simulation system for the coordinated management of production in small and medium-sized fish farming associations in Colombia [*]

Authors: Juan Sebastián Mantilla Serrano, Hammer Ronaldo Muñoz Hernández [**]

Keywords: Simulation, Fish Farming, System Dynamics, Artificial Neural Networks, Rolling Horizon, Combinatorial Optimization, Mixed-Integer Linear Programming, Associativity.

Description:

Fish farming in Colombia is predominantly practiced by small-scale producers operating in isolation, which limits their capacity to meet market demands that are individually unattainable and generates productive downtime that erodes the profitability of their installed infrastructure. In response to this challenge, the SIMON research group at the Industrial University of Santander has developed a research line aimed at democratizing access to fish farming management tools for Colombian rural communities. Within this framework, the immediately preceding project formalized a System Dynamics model for the biological simulation of an individual fishpond and integrated neural networks for forecasting fish weight and required feed ration.

The present work extends this foundation to the associative scale through a simulation and optimization platform that coordinates the production cycles of multiple ponds belonging to different producers, with the purpose of ensuring continuous supply against periodic market demand. The platform integrates three subsystems: a simulation engine that orchestrates the biological dynamics of multiple ponds simultaneously through a rolling horizon with Monte Carlo simulation; a data science module based on an LSTM recurrent neural network that predicts daily weight gain and required feed ration; and a mathematical optimization module that generates the optimal seeding and harvesting schedule by minimizing supply deviations against registered demand, subject to physical capacity constraints and commercial quality criteria.

The system represents a concrete contribution to the SIMON research group's line of work oriented toward the technological strengthening of the Colombian fish farming sector, and establishes the foundation upon which to scale toward a decision-support tool for small and medium-scale fish farming associations.

---

[*] Undergraduate Thesis.

[**] Faculty of Physicomechanical Engineering. School of Systems Engineering and Informatics. Research Supervisor: Hugo Hernando Andrade Sosa, M.Sc.

Introducción

La piscicultura de pequeña y mediana escala en Colombia se encuentra en una encrucijada estratégica. A pesar de su probado potencial como impulsor de la economía rural y la seguridad alimentaria, su estructura profundamente fragmentada impone un límite a su crecimiento: operando de forma aislada, la gran mayoría de los productores queda excluida de mercados que demandan suministros constantes y una oferta productiva predecible. Frente a este panorama, la asociatividad emerge como el mecanismo con mayor potencial para transformar esa debilidad estructural en una fortaleza colectiva; sin embargo, su viabilidad práctica depende de que los productores cuenten con herramientas capaces de coordinar sus ciclos productivos con la precisión que los mercados estructurados exigen.

El presente proyecto se ha cimentado sobre una trayectoria de investigación consolidada del grupo SIMON de la Universidad Industrial de Santander, dedicado durante años al diseño de herramientas de modelado y simulación para la piscicultura bajo los paradigmas del Pensamiento Sistémico y la Dinámica de Sistemas. Ese recorrido comenzó con sistemas como PESCO —un ambiente software pedagógico que permitía simular los ciclos básicos de producción y comercialización— y maduró progresivamente a través de versiones sucesivas que ampliaron su alcance desde herramientas móviles básicas, hasta plataformas 3D con entornos de aprendizaje virtual. El hito más reciente en esta evolución fue la integración de redes neuronales con el modelo de Dinámica de Sistemas existente, logrando una gestión operativa automatizada a nivel de un estanque individual. Sin embargo, esta base tecnológica también reveló una nueva frontera a superar: el tránsito desde el control de un agente productivo aislado hacia la coordinación estratégica de una asociación de piscicultores.

Por tanto, la plataforma desarrollada extiende la base tecnológica heredada del grupo SIMON hacia la escala asociativa, integrando el motor de simulación multi-estanque, el módulo predictivo LSTM y un subsistema de optimización matemática bajo una arquitectura de software diseñada para sostener esa complejidad sin sacrificar la mantenibilidad ni la extensibilidad del

sistema. Los detalles del problema que motiva esta integración, su justificación y los objetivos que la orientan se desarrollan en el Capítulo 1. El marco conceptual y tecnológico que la sustenta se expone en el Capítulo 2. El Capítulo 3 expone el enfoque metodológico de la investigación y el diseño de la evaluación del sistema, mientras que el Capítulo 4 describe su diseño y construcción. El Capítulo 5 presenta los resultados obtenidos, y el Capítulo 6 recoge las conclusiones y las líneas de trabajo futuro que se desprenden del proyecto.

1. Presentación

1.1 Planteamiento del Problema

La acuicultura, definida por Merino Archila et al. (2006) como «el cultivo controlado de organismos acuáticos», se ha consolidado como una actividad estratégica para la seguridad alimentaria y la mitigación de la pobreza en diversos países en desarrollo (véase el análisis detallado en la Sección 2.1). Bajo este marco, la piscicultura —especializada en la cría y engorde de peces— ha ganado una notable relevancia en Colombia, favorecida por su vasta extensión territorial y diversidad hidrológica, que ofrecen condiciones ideales para esta práctica (Merino Archila et al., 2006, p. 7). Lo anterior ha permitido posicionarla como una alternativa clave para la producción de proteína, destacando sus beneficios socioeconómicos para los productores de pequeña y mediana escala (Patricia Bonilla, Sara, 2021). Sin embargo, este enorme potencial se ve frenado por una profunda debilidad estructural: la fragmentación del sector. Con más del 96 % de los acuicultores clasificados como de subsistencia o pequeños productores, su aporte conjunto apenas alcanza el 23 % de la producción nacional, una asimetría que evidencia una brecha crítica de productividad y competitividad con los grandes productores (Servicio Estadístico Pesquero Colombiano [SEPEC], 2022).

Dicha fragmentación trasciende lo operativo para convertirse en un desafío estratégico de gran envergadura. Es el caso de los productores con capacidad productiva reducida, que, al operar de forma autónoma, se han enfrentado a una brecha crítica de productividad marcada por una alta incertidumbre económica. Esta condición se manifiesta en tres factores determinantes: (i) la carencia de instrumentos técnicos para proyectar con fiabilidad los volúmenes y cronogramas de cosecha; (ii) la gestión de los estanques como unidades desarticuladas que generan una oferta productiva irregular; y (iii) la imposibilidad de garantizar un suministro constante que cumpla con los estándares de contratos de gran escala (Rueda-Barrios et al., 2019). Como consecuencia, aquellos productores quedan confinados a mercados locales volátiles, lo que compromete drásticamente su

rentabilidad y la estabilidad de sus ingresos (Rueda-Barrios et al., 2019).

1.2 Justificación

Atendiendo a esta problemática, la línea de investigación del Grupo SIMON ya ha sentado precedentes fundamentales. Un proyecto anterior ha constituido un primer acercamiento a la gestión operativa de un estanque individual al automatizar el suministro de insumos (Rojas Prada, 2025). Si bien ese desarrollo representó un avance en el control productivo a microescala, quedó latente la necesidad de una solución para la planificación estratégica colectiva (consúltense a profundidad los antecedentes en la Sección 2.2). Por ello, en el presente se propone un sistema de soporte a la decisión que, mediante la gestión coordinada de múltiples estanques y el uso sinérgico de modelos predictivos y algoritmos de optimización, permita consolidar una línea de producción continua y estructurada.

En definitiva, este sistema no solo ofrece a los piscicultores una herramienta para reducir costos y aumentar la productividad. Su impacto más profundo radica en su capacidad para fomentar una asociatividad funcional y basada en resultados tangibles. Esta se erige no solo como una alternativa de mejora, sino también como un mecanismo de supervivencia y escalamiento competitivo. Al garantizar un suministro constante, se fortalece el poder de negociación del colectivo, facilitando su acceso a mercados de mayor escala y, fundamentalmente, brindándoles una mayor estabilidad económica.

Por tanto, este proyecto se ha centrado en la pregunta fundamental: ¿Cómo coordinar los ciclos de producción de un colectivo de piscicultores que se asocian para atender, de manera conjunta y periódica, una demanda de mercado? En este sentido, la respuesta a esta interrogante es de alta relevancia para la ingeniería y la academia, pues implica la aplicación de técnicas avanzadas de modelado, simulación y aprendizaje automático a una problemática de profundo impacto social y económico para Colombia.

1.3 Objetivos

1.3.1 Objetivo General

Desarrollar un sistema de simulación que dé soporte a la planificación productiva de una asociación de piscicultores orientado a atender una demanda periódica de mercado, asumiendo como base el modelo existente de simulación y gestión predictiva para estanques individuales, desarrollado en el grupo de investigación SIMON.

1.3.2 Objetivos Específicos

- Adaptar la arquitectura del módulo de simulación de producción piscícola con el fin de manejar la información de múltiples estanques de forma autónoma y escalable.
- Ajustar el modelo de red neuronal para que, a partir de los datos de la simulación, prediga las variables clave (p. ej., biomasa, tasa de crecimiento) que determinan la coyuntura ideal de cosecha en cada estanque.
- Implementar un módulo de optimización que, utilizando el pronóstico del modelo neuronal, genere un plan estratégico que determine la coordinación de los ciclos productivos de la asociación con el fin de garantizar una producción continua.
- Validar el sistema de planificación integrada mediante la simulación de escenarios que contemplen diversas variables de producción y demanda para evaluar la viabilidad y eficiencia de los planes estratégicos generados.

2. Marco Referencial

Este capítulo establece los cimientos contextuales, históricos y técnicos que fundamentan la investigación. El recorrido inicia con un marco contextual que analiza el sector piscícola en Colombia y los desafíos de comercialización que justifican la asociatividad. Posteriormente, se presentan los antecedentes, revisando las soluciones tecnológicas globales y la trayectoria del grupo SIMON. En tercer lugar, se desarrollan los fundamentos teóricos, detallando los principios de la Dinámica de Sistemas (DS), el aprendizaje automático y las estrategias de optimización dinámica y estocástica. Finalmente, se describe el marco tecnológico, especificando la arquitectura de software y las herramientas empleadas en el desarrollo del sistema integrado.

2.1 Contexto Socioeconómico y Productivo

2.1.1 La Piscicultura en Colombia: Contexto y Desafíos

La piscicultura en Colombia se caracteriza por su dinamismo y resiliencia, consolidándose como una alternativa clave para la producción de proteína y el crecimiento de la economía rural, sobre todo para los productores de pequeña y mediana escala. Los datos macroeconómicos en el ámbito contemporáneo han evidenciado un crecimiento robusto que incluso supera la media de la economía nacional: según el Ministerio de Agricultura y Desarrollo Rural de Colombia (MinAgricultura, 2025), en la década comprendida entre 2011 y 2020, la producción acuícola experimentó un crecimiento acumulado del 117 %, pasando de 82 622 a 179 351 Mg. Este volumen productivo no solo satisface una parte creciente de la demanda interna, que alcanzó un consumo per cápita de 8,8 kg en 2020, sino que también alimenta un dinámico frente exportador. En 2023, el valor agregado del macrosector de «Agricultura, ganadería, caza, silvicultura y pesca» registró un crecimiento del 1,8 % respecto al año anterior, una cifra significativamente superior al crecimiento del Producto Interno Bruto (PIB), que fue del 0,6 %. Dentro de este macrosector, la división

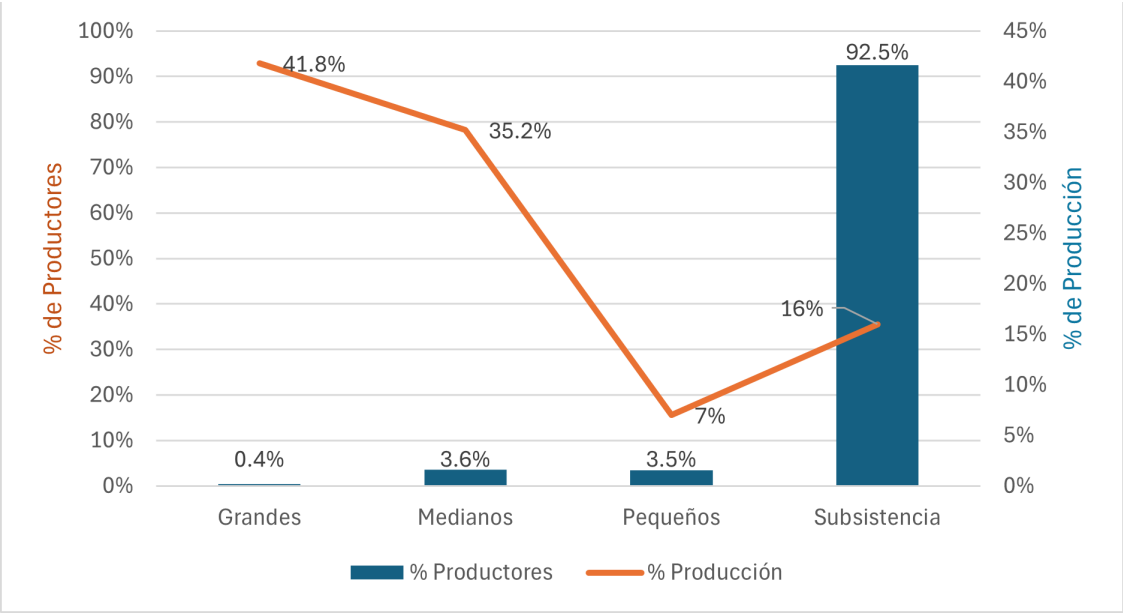
específica de «Pesca y acuicultura» aportó el 3,4 % del valor agregado total del sector primario en el mismo año (Unidad de Planificación Rural Agropecuaria [UPRA], 2023, pp. 3–4). Esta tendencia positiva se confirma con otros informes del MinAgricultura (2025) donde, para el segundo trimestre de 2025, reportaron un crecimiento del 6,1 % para ese mismo subsector, impulsado principalmente por la expansión de la producción de tilapia y camarón orientada a la exportación.

No obstante, frente a esta tendencia positiva en el desempeño macroeconómico, una revisión detallada de la estructura del sector revela una profunda asimetría. Conforme a la Autoridad Nacional de Acuicultura y Pesca (AUNAP, 2023), se definen las siguientes categorías para los acuicultores: el acuicultor de subsistencia corresponde a aquel cuya producción no supera los 10 Mg anuales; en contraste, el productor de gran escala sobrepasa los 240 Mg producidos al año. Esta marcada diferencia en las escalas de producción crea un mercado desigual. Mientras que la gran mayoría de los productores opera a una escala de subsistencia o pequeña, su contribución conjunta al volumen total es desproporcionadamente baja. Esta disparidad se ilustra claramente en la Figura 1 donde se observa que un segmento minoritario de productores de mediana y gran escala concentra la mayor parte del volumen productivo del país, lo que les permite acceder a economías de escala, tecnificar sus procesos y negociar contratos de gran volumen. Por el contrario, la inmensa mayoría de los productores, al operar de forma aislada y con una capacidad productiva limitada, enfrenta una competencia desigual, quedando relegados a mercados locales con precios más volátiles y un poder de negociación casi nulo (Rueda-Barrios et al., 2019).

A ello se suma la estructura de costos inherente a la infraestructura piscícola, en concreto, una parte significativa de los gastos operativos —depreciación de activos, mantenimiento, personal permanente e intereses sobre el capital invertido— permanece constante independientemente del nivel de ocupación de los estanques (Chim et al., s.f.). Un estanque vacío o subutilizado no elimina sus costos fijos, más bien los redistribuye sobre una menor producción, erosionando directamente la rentabilidad del productor. Para el pequeño piscicultor que opera de forma aislada, esta carga resulta especialmente gravosa, pues carece de la escala necesaria para absorberla mediante volumen. La coordinación de los ciclos productivos entre múltiples productores asociados constituye

Figura 1

Disparidad entre productores y volumen de producción en la acuicultura



Nota. Adaptado de AUNAP, 2023

precisamente el mecanismo para reducir estos períodos de inactividad y distribuir la carga de los costos fijos sobre una producción continua y predecible.

De igual manera, la informalidad es otro desafío estructural que perpetúa la desproporción productiva. Si bien existen esfuerzos gubernamentales significativos para contrarrestarla, la magnitud del reto es considerable. Según la AUNAP (2024), durante ese año se avanzó en la formalización de 1600 acuicultores y 19 600 pescadores artesanales. Pese a ello, al contrastar estas cifras con el universo total de pescadores, la brecha sigue siendo abismal. Estimaciones de la Food and Agriculture Organization of the United Nations (FAO, 2023) situaban el número de pescadores artesanales en alrededor de 150 000, indicando que casi el 90 % de la pesca continental era informal. Esto sugiere que, a pesar de los avances, una gran mayoría del sector aún opera al margen de la formalidad. Esta informalidad agrava la falta de coordinación, ya que los actores operan sin datos estandarizados ni visibilidad compartida, un problema que la planificación estratégica busca mitigar.

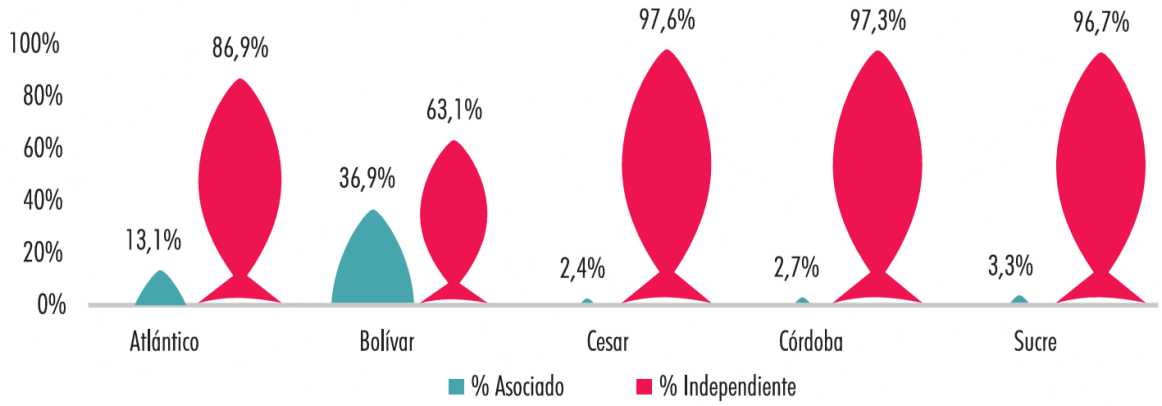
Por su parte, la asociatividad se reconoce ampliamente como el mecanismo más eficaz para superar estos y otros desafíos de los productores de pequeña escala mencionados. Una asociación puede agregar valor productivo para negociar mejores precios, realizar compras conjuntas de insumos y acceder colectivamente a tecnología y asistencia técnica. A su vez, permite la formalización de los productores del sector al desarrollar mecanismos de acción conjunta y cooperación, contribuyendo a que los productores mejoren su participación en el mercado. Un ejemplo de esto es la Federación Colombiana de Acuicultores (FEDEACUA, s.f.), que agremia a productores de diversas regiones del país para fortalecer la cadena productiva y mejorar la competitividad del sector a nivel nacional e internacional. A través de la federación, los piscicultores han logrado estandarizar procesos, acceder a certificaciones de calidad y negociar en bloque, mejorando así sus márgenes de ganancia. También destacan casos de éxito como el de la Asociación de Piscicultores del Huila (*ASOPISHUILA*), departamento donde se concentra la mayor producción de tilapia de Colombia. En este marco, según el Programa Global de Acceso a Mercados (GMAP, s.f.), gracias a la asociatividad, pequeños y medianos productores han podido acceder a tecnología, programas de capacitación en buenas prácticas y canales de comercialización que, de forma individual, serían inalcanzables. Esta integración ha permitido que sus productos lleguen a grandes superficies y al mercado de exportación, incrementando sus ingresos y garantizando una mayor estabilidad económica.

A pesar de estos beneficios evidentes, el nivel de asociatividad entre muchos piscicultores del país sigue siendo bajo. A juicio de AUNAP (s.f.), en promedio, en los departamentos de Atlántico, Bolívar, Cesar, Córdoba y Sucre no rebasan el 12,2 % de todos los acuicultores encuestados, como se muestra detalladamente en la Figura 2; lo cual es consecuente con que el 96,4 % de los acuicultores esté registrado como persona natural. Aunque existen algunas asociaciones, la conformación de organizaciones rurales sólidas y sostenibles enfrenta barreras significativas de índole social y cultural. Entre estas barreras destacan la desconfianza generada por el individualismo de los no cooperadores, quienes no asumen la responsabilidad y pierden de vista el objetivo de la asociación, pero sí participan en los beneficios; y la debilidad en las capacidades de gestión y gobernanza,

debido al desconocimiento de las estrategias de control interno de sus pares asociados (Narváez-Rodriguez, Cristhian Camilo, 2014), a lo que se suma que muchas asociaciones se constituyen con el principal incentivo de acceder a programas de ayuda gubernamental, limitando el desarrollo de una visión empresarial a largo plazo.

Figura 2

Porcentaje de acuicultores asociados e independientes



Nota. Tomado de AUNAP, s.f.

La interrelación entre los factores anteriores revela que la tecnología y la asociatividad no son desafíos independientes, sino que actúan como las dos caras de una misma moneda: mientras que un productor individual carece de la escala necesaria para justificar la inversión en tecnología, una asociación que no disponga de herramientas que faciliten la coordinación y evidencien beneficios tangibles puede sucumbir ante la desconfianza (Narváez-Rodriguez, Cristhian Camilo, 2014).

Bajo esta perspectiva, la herramienta tecnológica desarrollada en esta investigación actúa como el inicio del catalizador para fortalecer el tejido asociativo. Al ofrecer una solución concreta a un problema compartido —la planificación de la producción frente a las exigencias del mercado—, el sistema aporta el valor tangible que los productores necesitan para superar las barreras de la colaboración. Se crea de este modo un círculo virtuoso: la infraestructura digital robustece a la asociación al incrementar eficiencia, mientras que la unión del colectivo viabiliza la adopción de herramientas cada vez más avanzadas (Narváez-Rodriguez, Cristhian Camilo, 2014).

2.1.2 Productividad y Estrategias de Mercadeo para las Asociaciones Piscícolas

La asociatividad emerge como una respuesta estratégica a la fragmentación del sector piscícola, pero su éxito no está garantizado únicamente por la unión de los productores. Para que una asociación sea sostenible y genere valor real, debe enfocarse en dos pilares interconectados: el aumento de la productividad colectiva y la implementación de estrategias de mercadeo inteligentes que capitalicen su nueva escala.

2.1.2.1 Fomento de la productividad a través de la asociatividad. La unión de pequeños productores en una asociación permite superar las limitaciones de la escala individual, lo cual genera eficiencias operativas que se traducen en mayor productividad y rentabilidad. Esto se logra a través de varios mecanismos vinculados entre sí:

- Principalmente es posible acceder a economías de escala mediante la compra conjunta de insumos críticos, como alevines y alimento concentrado, dado que permite a la asociación negociar precios por volumen y así reducir significativamente los costos de producción para cada miembro. De igual manera, la gestión compartida de recursos, como el transporte para la distribución del producto final, puede disminuir los costos logísticos y racionalizar la cadena de suministro (Penrose-Buckley, Chris, 2007).

- Al consolidar la producción, la asociación puede ofrecer volúmenes mayores y más consistentes. Esto le otorga un poder de negociación superior frente a los compradores, algo que le permite acceder a precios más justos y estables, así como superar la volatilidad característica de los mercados locales (Penrose-Buckley, Chris, 2007).

- De forma agrupada, los productores pueden cofinanciar la adopción de tecnologías que serían inaccesibles individualmente, como sistemas de aireación, software de gestión o infraestructura de cadena de frío. La asociación también funciona como un vehículo eficaz para recibir asistencia técnica especializada, obtener capacitación en buenas prácticas productivas

y acceder a líneas de crédito con condiciones más favorables (Narváez-Rodriguez, Cristhian Camilo, 2014).

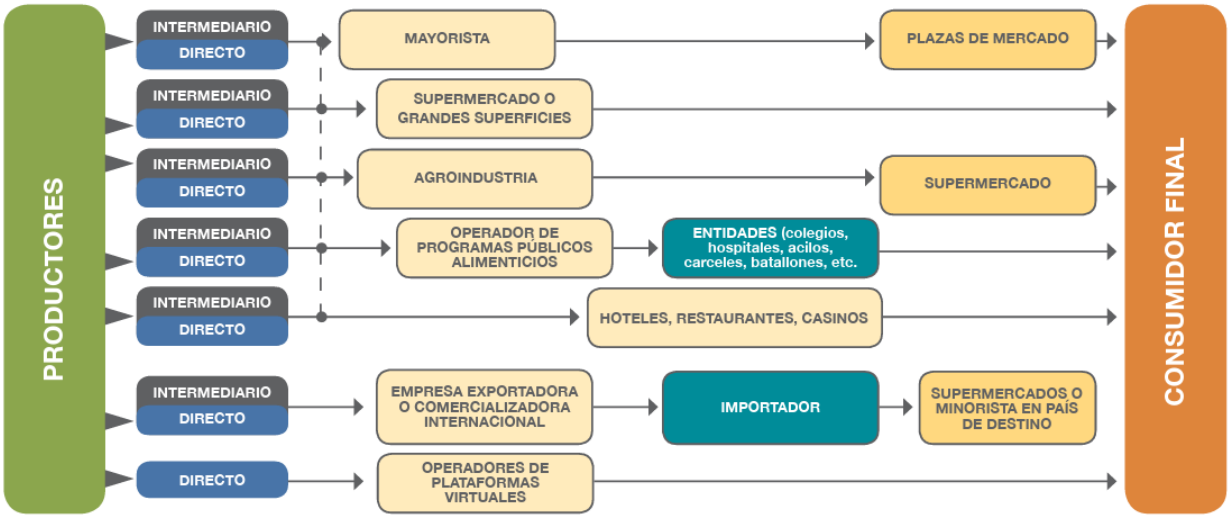
El éxito de estos mecanismos, aun así, no es automático. Depende de factores internos cruciales como la existencia de un liderazgo claro, un fuerte sentido de pertenencia por parte de los miembros y, fundamentalmente, un gobierno cooperativo transparente que genere la confianza necesaria para la acción colectiva.

2.1.2.2 Estrategias de Comercialización para la Producción Conjunta. Una vez que la asociación consolida su oferta productiva, debe definir estratégicamente cómo llevarla al mercado. La comercialización de productos agroalimentarios se estructura en diversos canales de mercado, desde los más directos hasta los más complejos, que involucran una serie de actores clave que conectan a los productores con el consumidor final. De acuerdo con la Dirección de Comercialización de la Agencia de Desarrollo Rural (ADR, s.f.-b), y como se ilustra en la Figura 3, los canales varían en su grado de sofisticación, desde rutas directas hasta la interacción con múltiples intermediarios, abriendo paso a diferentes segmentos. La elección de la estrategia adecuada dependerá de varios factores: la capacidad de producción, el nivel de formalización, la calidad e inocuidad del producto, el acceso a infraestructura logística, las exigencias del mercado, las normativas vigentes y la capacidad financiera de esta.

2.1.2.2.1 Modelo 1. Canales cortos. Este modelo se caracteriza por la ausencia o mínima presencia de intermediarios, permitiendo a los productores conectar directamente con el consumidor final o con los establecimientos comerciales a nivel local. La principal ventaja estratégica de este circuito es que maximiza el margen de rentabilidad del productor al eliminar los costos de intermediación, además de fomentar la confianza directa con el cliente. En Colombia, la Agencia de Desarrollo Rural (ADR, s.f.-a) apoya activamente estos denominados «circuitos cortos» mediante estrategias de comercialización como mercados campesinos, ruedas de negocio locales y programas de compras públicas, consolidando así una fuente de ingresos rápida y justa no solo para las

Figura 3

Principales actores de los canales de mercado



Nota. Tomado de ADR, s.f.-b

asociaciones, sino también para el productor independiente.

2.1.2.2.2 Modelo 2. Encadenamiento productivo. Este modelo implica la participación de uno o más intermediarios y está diseñado para mover grandes volúmenes de producto hacia mercados más amplios y estructurados. Según el Departamento Nacional de Planeación (DNP, 2024), este modelo se puede subdividir en diferentes canales, cada uno con sus particularidades:

- Canal tradicional: históricamente, este ha sido el canal principal para los pequeños productores. Involucra a acopiadores que recogen el producto en las fincas y lo llevan a las centrales de abastos, donde los mayoristas lo distribuyen a los minoristas.
- Canal moderno: aquí se representa una oportunidad para las asociaciones mejor organizadas, dado que incluye la venta a cadenas de supermercados y a la industria alimentaria que utiliza el pescado como materia prima.

- Canal de exportación: es el más exigente, aunque el más rentable; generalmente es una evolución del canal moderno. Requiere que la asociación no solo cumpla con todos los estándares de calidad y logística nacionales, sino también con las certificaciones y normativas del país de destino.

Una estrategia avanzada y resiliente consiste en la articulación de ambos modelos: la asociación puede destinar el grueso de su producción planificada a cumplir con contratos formales (Modelo 2), asegurando un flujo de ingresos estable para cubrir sus costos fijos; asimismo, el excedente productivo, o aquellos lotes que no cumplen con las especificaciones de calibre exigidas por el contrato, pueden comercializarse a través de canales directos (Modelo 1). Este enfoque híbrido permite mitigar los riesgos del mercado, maximizar la rentabilidad y diversificar la base de clientes. Sin embargo, la viabilidad de esta estrategia depende críticamente de una planificación productiva de alta precisión; un reto logístico que, dada su complejidad, hace indispensable la adopción de herramientas computacionales y sistemas de soporte a la decisión.

2.2 Estado del Arte

2.2.1 Soluciones Tecnológicas para la Gestión de la Acuicultura

La transformación digital ofrece respuestas estratégicas a los desafíos de la acuicultura moderna, siendo su implementación fundamental para garantizar la sostenibilidad y la eficiencia operativa. La digitalización ofrece soluciones integrales que abarcan toda la cadena de valor mediante herramientas como el Internet de las Cosas (*IoT*), que facilitan la monitorización en tiempo real mediante sensores; la Inteligencia Artificial (*IA*) y el Aprendizaje Automático (*ML*), que optimizan la toma de decisiones; y la robótica o los drones, que automatizan labores exigentes. A su vez, otras tecnologías, como el *Blockchain*, aseguran la trazabilidad de los productos, mientras que la Realidad Virtual (*RV*) facilita la capacitación técnica (Neil J. Rowan, 2023, p. 366).

La implementación de estas herramientas ha permitido desarrollar proyectos a nivel nacio-

nal, como el sistema de monitoreo IoT implementado por Piamba-Mamian et al. (2021, pp. 3–19) en una piscicultura de trucha en Putumayo, Colombia. Aquel proyecto utilizó sensores de bajo costo para vigilar en tiempo real variables críticas, como el oxígeno y la turbidez, responsables de la alta mortalidad de alevines. Gracias a este sistema, se logró identificar la causa de las pérdidas y aplicar soluciones correctivas eficaces para mejorar la productividad de los pequeños acuicultores del país. A nivel internacional existen diversas plataformas comerciales, cada una con un enfoque particular: AquaManager (2025), enfocado en la gestión del ciclo productivo (engorde y criaderos) con herramientas de IA y IoT que utilizan sensores, cámaras y estimadores de biomasa para la monitorización en tiempo real; y el sistema desarrollado por Ernst et al. (2000, p. 121–179) llamado *AquaFarm*, el cual permite simular procesos físicos, químicos y biológicos de cualquier tipo de instalación acuícola, además de modelar y evaluar estrategias alternativas de diseño y manejo, compilar presupuestos y optimizar la producción de manera iterativa antes de la implementación.

La revisión de las plataformas y herramientas globales revela una tendencia clara: la industria agrotecnológica ha priorizado la sensorización y la eficiencia operativa a nivel de granja. Si bien estos sistemas son eficaces para el monitoreo aislado, carecen de las capacidades algorítmicas para orquestar la producción a gran escala. Este panorama confirma la ausencia de soluciones informáticas diseñadas específicamente para articular redes de productores, dejando un vacío técnico en el mercado para la gestión estratégica colectiva.

2.2.2 El Proyecto Previo del Grupo SIMON: Simulación de un Estanque Individual

El presente proyecto se construye sobre una extensa línea de investigación del grupo SIMON, adscrito a la Escuela de Ingeniería de Sistemas e Informática de la Universidad Industrial de Santander (UIS). Este grupo cuenta con una trayectoria consolidada en la aplicación de modelado y simulación en diversas áreas, incluyendo el sector agrario, con un enfoque en el auxilio a la comunidad campesina.

La línea de investigación inició con el proyecto denominado Ambiente Software para el

Aprendizaje y Toma de Decisiones (Guerra González et al., 2011), cuyo producto final fue el software *PESCO*, concebido como un «juego serio» y un simulador de práctica para la pedagogía y el aprendizaje de la piscicultura. Esta primera versión constaba de tres componentes: un juego para teléfonos móviles, una aplicación de escritorio llamada Micromundo de Aprendizaje de Sistemas Productivos (*MASIP*) y un sitio web de soporte que alojaba un mercado virtual. El objetivo era eminentemente pedagógico: permitir a los usuarios simular el ciclo básico de adquirir, criar y vender peces para aprender los fundamentos de la producción y la dinámica de oferta y demanda. Una segunda versión bajo el mismo nombre (Castro Castro y Zambrano Urbina, 2012) se enfocó en el mantenimiento y mejora del software, añadiendo funcionalidades como la gestión de sesiones de juego, niveles de dificultad y un mejor seguimiento de datos históricos de precios y transacciones en el sitio web, refinando así la experiencia de aprendizaje. Ambas versiones se centraron en el pilar del aprendizaje, proporcionando una herramienta lúdica para comprender los conceptos básicos de la piscicultura.

El progreso continuó con PESCO v3.0 (Maldonado Vargas, 2018), que representó un salto tecnológico y conceptual al migrar el juego a una plataforma web con gráficos 3D. De igual o mayor relevancia, esta versión introdujo una mayor dificultad de implementación en la fase de intercambio comercial, al ofrecer al jugador tres alternativas de participación en el mercado: producción competitiva individual, participación asociativa para atender una demanda fija y participación asociada dentro de un ambiente competitivo. Este fue el primer antecedente directo en abordar la simulación de estrategias de venta más allá de la simple transacción individual. A la fecha, el proceso avanza hasta PESCO v4.0 (Guio Roa y Vásquez Alcocer, 2024) que consolidó toda la trayectoria en un Ambiente Virtual de Aprendizaje formal, estructurado bajo el concepto de «aula ampliada». Esta última versión integra dos componentes: un componente teórico-práctico en la web —con material de estudio y modelos visibles— y el componente de simulación virtual —el juego PESCO—, que actúa como la herramienta práctica del entorno educativo.

Posteriormente, la investigación se orientó hacia la mejora de la eficiencia en la producción. Un desarrollo clave fue la creación de un ambiente web diseñado por Rojas Prada (2025), el cual

integró el modelo de DS, desarrollado en PESCO, con Redes Neuronales de Memoria a Corto y Largo Plazo (LSTM) para prever el peso de los peces y la ración de alimento requerida. El objetivo de este último sistema era resolver problemas operativos concretos, como la necesidad de pesajes constantes y el desperdicio de alimento, con el fin de lograr una gestión del estanque más eficiente y automatizada. En paralelo, otra línea de investigación desarrollada por Cuadros Sanabria y Rangel Flórez (2021) exploró la instrumentación física de estanques mediante IoT, desarrollando una plataforma para sistemas embebidos, capaz de recolectar datos de sensores (oxígeno, temperatura, entre otros) y controlar actuadores en tiempo real, abordando el control físico del proceso de crianza.

Los trabajos anteriores consolidaron un dominio técnico exhaustivo sobre la modelización del crecimiento del pez y la comercialización de este. No obstante, al proyectar este conocimiento hacia un entorno asociativo, el paradigma de ingeniería debe evolucionar. Mientras que los desarrollos anteriores abordaron la gestión de un estanque como un riguroso problema de control de procesos, la coordinación interconectada de múltiples productores exige transitar hacia modelos de optimización de la cadena de suministro. Es precisamente en esta evolución conceptual e investigativa donde se fundamenta el presente trabajo, representando un salto cualitativo hacia la asociatividad digital del sector.

2.3 Fundamentos Teóricos

2.3.1 Dinámica de Sistemas y Generación de Datos Sintéticos

Según Prieto Mejía (2025), la Dinámica de Sistemas es una metodología desarrollada en la década de 1950 por Jay Forrester en el Instituto Tecnológico de Massachusetts (*MIT*) para entender y gestionar sistemas complejos que cambian a lo largo del tiempo. Su premisa fundamental era que la estructura de un sistema, es decir, el conjunto de interrelaciones y bucles de realimentación entre sus componentes, es el principal determinante de su comportamiento dinámico. A través de

modelos de simulación por computadora, la DS permite visualizar, analizar y experimentar con estas estructuras para identificar los puntos de apalancamiento donde una intervención puede generar mejoras significativas y sostenibles. Es una herramienta transdisciplinaria que se ha aplicado con éxito en campos tan diversos como la ecología, la economía, la gestión empresarial y la salud pública, siendo especialmente potente para modelar sistemas socio-ecológicos como la acuicultura (M. et al., 2023). La DS utiliza un lenguaje visual y conceptual basado en tres elementos fundamentales para representar la estructura de un sistema:

2.3.1.1 ***Stocks*** **(niveles).** Son las acumulaciones dentro de un sistema. Representan las variables que caracterizan el estado del sistema en un momento dado y que no pueden cambiar de forma instantánea, ya que solo se modifican a través de los flujos (Prieto Mejía, 2025). En el contexto de la piscicultura, los *stocks* son elementos intuitivos y medibles como el peso promedio del pez en un estanque (en kg) o la población de peces (en número de individuos). Los niveles le otorgan al sistema su memoria o inercia.

2.3.1.2 ***Flows*** **(flujos).** Son las tasas de cambio que modifican los niveles a lo largo del tiempo. Se representan como «tuberías» con «válvulas» que regulan el caudal que entra o sale de un nivel en una unidad de tiempo. Por ejemplo, el nivel del peso promedio del pez es incrementado por el flujo de la tasa de crecimiento (kg/d), mientras que el flujo de la mortalidad (en número de individuos por día) disminuye la población de peces (Prieto Mejía, 2025).

2.3.1.3 ***Feedback Loops*** **(bucles de realimentación).** Son la esencia de la complejidad dinámica. Un bucle de realimentación es una cadena cerrada de relaciones de causa y efecto, donde una acción inicial se propaga a través del sistema y eventualmente regresa para influir en la acción original (Prieto Mejía, 2025). Según Gudrun Wallentin (s.f.), existen dos tipos de bucles:

2.3.1.3.1 Self-reinforcing Feedback (bucles de refuerzo). Generan crecimiento o declive exponencial, es decir, son procesos que se autoamplifican. En el sistema piscícola, una mayor biomasa permite un mayor consumo de alimento, lo que acelera la tasa de crecimiento, incrementando a su vez la acumulación de biomasa total en el estanque.

2.3.1.3.2 Self-regulating Feedback (bucles de equilibrio). Buscan alcanzar una meta o mantener la estabilidad mediante procesos autocorrectivos. Por ejemplo, un incremento excesivo de la biomasa eleva el consumo de alimento y la producción de residuos orgánicos; esto deteriora la calidad del agua (p. ej., reduciendo los niveles del Oxígeno Disuelto), lo cual aumenta el nivel de estrés de los peces, reduce su tasa de crecimiento y, finalmente, limita el aumento descontrolado de la biomasa.

Bajo este enfoque holístico, el modelo de DS no se concibe meramente como una herramienta de apoyo, sino como la recreación formal del fenómeno real que constituye el objeto de estudio de la presente investigación. En consecuencia, el modelo actúa como el núcleo del sistema propuesto con un doble propósito: primero, sirve como la fuente primaria y autoritaria para la generación de los datos necesarios para el entrenamiento de la red neuronal LSTM; y segundo, actúa como el motor de ejecución en las fases de simulación. Al operar dentro de este mundo virtual validado, la metodología permite describir el comportamiento del sistema mediante experimentos simulados que recrean diversos escenarios de siembra y cosecha de forma controlada y con rigor científico, sin las limitaciones de sesgo o incompletitud de los registros del mundo real. Los detalles exhaustivos sobre las variables, ecuaciones y parámetros de esta estructura dinámica, que ha servido de base para las versiones actuales, se encuentran ampliamente documentados y verificados en el trabajo de Maldonado Vargas (2018).

2.3.2 Aprendizaje Automático y Redes LSTM para Series de Tiempo

Para que un sistema de planificación sea efectivo, debe ser capaz de proyectar la evolución del sistema productivo. Si bien la DS permite realizar estas proyecciones mediante la simulación de experimentos basados en su estructura causal, el llamado módulo inferencial propuesto integra, además, técnicas de aprendizaje automático para robustecer el análisis de datos secuenciales. Dichos datos se estructuran como series de tiempo[1], las cuales permiten inferir tendencias y anticipar el comportamiento dinámico del sistema. Un estanque de piscicultura es una fuente rica de este tipo de datos, donde variables como el peso promedio del pez, la temperatura del agua o las muertes, al ser registradas a intervalos regulares (p. ej., diarios u horarios), constituyen dichas estructuras temporales (Gandh D et al., 2024).

Estos datos históricos no son meros registros; contienen patrones y dependencias temporales que permiten inferir tendencias y anticipar el progreso del sistema. Estas dependencias pueden ser tanto de corto como de largo plazo, lo que plantea el desafío de cómo capturar relaciones que no son inmediatas, sino que se extienden a lo largo de cientos o miles de registros. Es aquí donde entran en juego los modelos de redes neuronales diseñados para datos secuenciales, como las Redes Neuronales Recurrentes (*RNN*). A diferencia de las redes neuronales estándar (*feedforward*) que procesan cada entrada de forma independiente, las RNN están diseñadas para procesar estas series de tiempo. Su arquitectura incluye bucles de realimentación que les permiten pasar información de un paso de la secuencia al siguiente, creando una forma de «memoria» interna (Agmata y Guðmundsson, 2025). Esto significa que la salida de la red en un momento t depende no solo de la entrada en t, sino también de la información procesada en los pasos anteriores $t-1, t-2, \ldots$. Sin embargo, las RNN simples enfrentan una limitación clave: el problema del desvanecimiento del gradiente (*vanishing gradient*). Ello supone que la influencia de las entradas más antiguas disminuye exponencialmente a medida que la secuencia avanza, lo que dificulta enormemente que la red aprenda estas dependencias a largo plazo (Agmata y Guðmundsson, 2025; Merabet y

[1]Entiéndase como secuencias de observaciones de una variable tomadas en puntos sucesivos y equidistantes en el tiempo.

Elsaraiti, 2021). Dado que un ciclo de cultivo piscícola dura meses, esta limitación es crítica.

Para solventar este inconveniente, se desarrollaron arquitecturas más sofisticadas: las Redes Neuronales con Memoria a Corto y Largo Plazo (*LSTM*), que fueron diseñadas, precisamente, para superar el desvanecimiento del gradiente. La innovación central de estas arquitecturas es la «celda de memoria» (*memory cell*) que actúa como un estado interno capaz de transportar información a lo largo de la secuencia. El flujo de información hacia y desde esta celda está regulado por un sistema de tres compuertas (*gates*), que son redes neuronales «pequeñas» que aprenden qué información es relevante y cuál debe ser ignorada (Merabet y Elsaraiti, 2021). A continuación se especifican dichas compuertas.

2.3.2.1 ***Forget Gate*** **(compuerta de olvido).** Analiza la entrada actual (x_t) y la salida del estado oculto anterior (h_{t-1}) para decidir qué información del estado de la celda de memoria anterior (C_{t-1}) debe ser descartada. Su salida es un número entre 0 (olvidar completamente) y 1 (mantener completamente).

2.3.2.2 ***Input Gate*** **(compuerta de entrada).** Determina qué nueva información de la entrada actual debe ser almacenada en el estado de la celda. Se compone de dos partes: una que decide qué valores actualizar y otra que crea un vector de nuevos valores candidatos para añadir al estado.

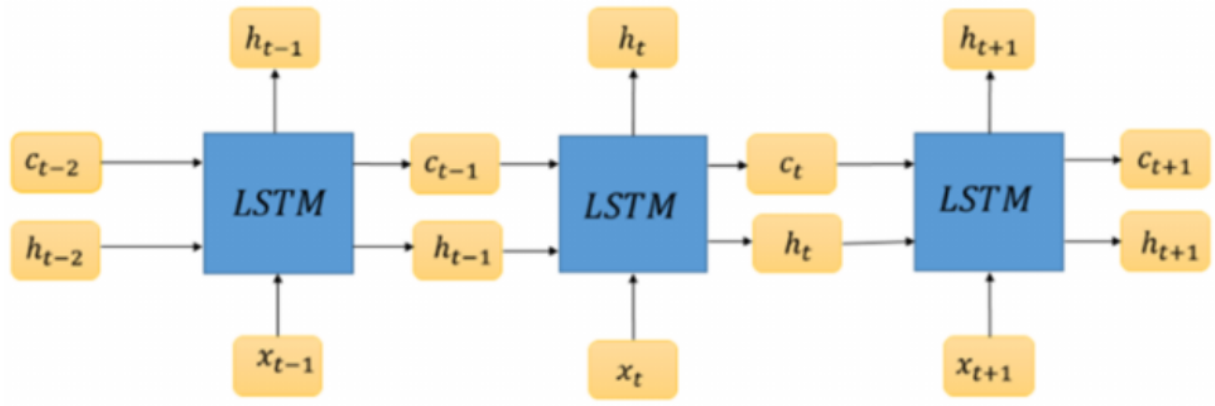
2.3.2.3 ***Output Gate*** **(compuerta de salida).** Decide qué parte del estado de la celda de memoria (C_t) se va a emitir como salida. En otras palabras, filtra el estado de la celda para producir el nuevo estado oculto (h_t), que es el pronóstico para ese paso de tiempo.

En la Figura 4 se representa esquemáticamente el funcionamiento de una red LSTM. Cada n-ésimo bloque incluye una celda de memoria (C_{t+n}) y un estado oculto (h_{t+n}), los cuales se actualizan y transmiten a lo largo de la secuencia. Esta estructura de compuertas permite a la red LSTM mantener un control selectivo sobre su memoria, reteniendo información relevante durante

períodos prolongados y descartando la que no lo es. Esta capacidad ha demostrado ser superior a la de los modelos estadísticos tradicionales en la predicción de series de tiempo extensas, no lineales y no estacionarias, que son comunes en dominios como la meteorología, las finanzas y la agricultura (Merabet y Elsaraiti, 2021).

Figura 4

Arquitectura de una celda de memoria LSTM



Nota. Tomado de Merabet y Elsaraiti, 2021.

La elección de una arquitectura que integre ambos modelos se fundamenta en la complementariedad entre estos enfoques. El modelo de DS provee una base teórica y estructural para la representación del sistema, mientras que la LSTM aporta flexibilidad para capturar patrones no triviales directamente desde los datos. Esta integración se alinea además con implementaciones previas del grupo SIMON, lo que respalda su viabilidad en el contexto del sistema propuesto.

2.3.3 Estrategias de Control y Simulación

2.3.3.1 Control Predictivo Basado en Modelos (MPC). El MPC es un esquema de control en el que se utiliza un modelo para predecir el comportamiento futuro del sistema sobre una ventana de tiempo finita, denominada horizonte. Con base en estas predicciones y el estado actual

medido del sistema, se calculan las entradas de control óptimas respecto a un objetivo de control definido y sujeto a múltiples restricciones. Después de un cierto intervalo de tiempo, el proceso de medición, estimación y cálculo se repite con un horizonte desplazado. Por esta razón, este método también se denomina Control de Horizonte Deslizante (Lucia y Fiedler, 2023).

La esencia del control predictivo se basa en tres elementos clave: un modelo predictivo, la optimización en el rango de una ventana temporal, y la corrección mediante realimentación (Zhang, 2010). Este paradigma, ampliamente consolidado en la industria química y de refinación, ha demostrado su capacidad para gestionar sistemas con múltiples entradas y salidas, manejar retardos y satisfacer restricciones de manera explícita (Max Schwenzer y Abel, 2021).

2.3.3.2 Horizonte Deslizante (*Sliding Horizon*).

El horizonte deslizante es una técnica de planificación dinámica en la que un modelo de optimización dependiente del tiempo se resuelve repetidamente, desplazando el intervalo de planificación hacia adelante en cada iteración (B.V., 2026). A diferencia de una planificación estática —donde todas las decisiones se toman de una sola vez al inicio—, este enfoque delimita y ejecuta únicamente las decisiones del período inmediato, actualiza el estado real del sistema y los pronósticos disponibles, y luego resuelve nuevamente el problema sobre el nuevo horizonte desplazado (Cuisinier et al., 2022). Esto se convierte en una herramienta adecuada para problemas recurrentes y dinámicos donde la información cambia con el tiempo.

2.3.3.3 Simulación de Monte Carlo para el Manejo de la Aleatoriedad.

La Simulación de Monte Carlo es una técnica estadística que utiliza muestreo aleatorio repetido y modelado probabilístico para estimar funciones matemáticas y replicar el comportamiento de sistemas avanzados en los que alguno de sus componentes no es determinista (Harrison, 2010). En lugar de producir un único valor de salida, como haría un modelo determinista, cada ejecución de la simulación genera un resultado distinto a partir de muestras aleatorias de las distribuciones de probabilidad que caracterizan las variables de entrada; la agregación de un gran número de estas réplicas permite

obtener una distribución completa de resultados posibles, junto con estimaciones de su probabilidad de ocurrencia.

En el contexto piscícola, la incertidumbre inherente al proceso biológico es la motivación principal para adoptar este enfoque. Variables como la temperatura del agua y las tasas de mortalidad no son constantes: fluctúan de forma aleatoria bajo condiciones reales. Estudios realizados por MacDonald et al. (2024), sobre producción de tilapia han demostrado que la aplicación de análisis de Monte Carlo —mediante la simulación simultánea de la variabilidad de la tasa de mortalidad, la temperatura media del agua, el costo del alimento y el precio del producto—, permite cuantificar el impacto de estas fuentes de incertidumbre sobre los resultados de producción y orientar la toma de decisiones hacia escenarios con menor riesgo.

2.3.4 Optimización Matemática de la Producción

Los sistemas biológicos se caracterizan por su dinamismo, la incertidumbre inherente a sus procesos y las profundas interdependencias no lineales entre sus componentes. El dinamismo se manifiesta en el cambio constante de las variables de estado del sistema, como el crecimiento de la biomasa, el consumo de alimento y la transformación de los parámetros de calidad del agua (Maldonado Vargas, 2018). La incertidumbre es un factor omnipresente que afecta a múltiples niveles: factores ambientales como la temperatura del agua y los niveles de oxígeno disuelto fluctúan de manera impredecible; variables biológicas como las tasas de crecimiento y mortalidad presentan una variabilidad natural; y factores de mercado, como los precios de los insumos y del producto final, son volátiles (Kunapinun et al., 2025). Finalmente, las interdependencias no lineales complican significativamente el modelado y la gestión. Por ejemplo, la eficiencia con la que un pez convierte el alimento en biomasa, conocida como Factor de Conversión Alimenticia (FC), no es constante, sino que depende de variables como el peso actual del pez y la temperatura del agua (Guerra González et al., 2011).

Para abordar esta intrincación, la Investigación de Operaciones[2] (*IO*) ofrece un conjunto de métodos científicos y matemáticos orientados a mejorar la toma de decisiones (Carravilla y Oliveira, 2013). A diferencia del enfoque primordialmente descriptivo y simulativo de la DS, que ayuda a responder cómo se comportará el sistema ante una política dada, la Programación Matemática (*PM*), una de las principales herramientas de la IO, adopta un enfoque prescriptivo. Su objetivo es determinar cuál es la mejor política que se debe implementar para alcanzar un objetivo determinado. Dentro de este campo, la Programación Lineal y sus extensiones son fundamentales para la planificación de la producción.

2.3.4.1 Programación Lineal. Es una técnica matemática para optimizar un resultado (como el beneficio o el costo) en un modelo cuyos requisitos están representados por relaciones lineales. Según McCarl y Spreen (2020), la formulación canónica de un problema de PL consta de tres componentes principales:

- Función objetivo: Una expresión matemática lineal que se busca maximizar o minimizar. Por ejemplo, minimizar el costo total de una mezcla de alimentos.
- Variables de decisión: Son las cantidades desconocidas y continuas que el modelo debe determinar. Por ejemplo, los kilogramos de cada ingrediente a incluir en una ración.
- Restricciones: Un conjunto de ecuaciones o inecuaciones lineales que representan las limitaciones del sistema, como la disponibilidad de recursos o los requisitos de calidad.

2.3.4.2 Programación Lineal Entera-Mixta (PLEM). Es una extensión de la PL que supera una de sus mayores limitaciones al permitir que algunas o todas las variables de decisión tomen valores enteros o binarios (0 o 1). Esta capacidad de modelar decisiones discretas amplía enormemente el alcance de la optimización matemática, permitiendo abordar problemas de naturaleza lógica y combinatoria (Weintraub y Romero, 2006). El paso de la PL a la PLEM representa más

[2]Disciplina que aplica métodos científicos y matemáticos para mejorar la toma de decisiones en sistemas complejos.

que un simple cambio en el tipo de variables; es un salto cualitativo en la capacidad expresiva del modelo. Mientras que la PL se centra en problemas de asignación de recursos continuos (respondiendo a la pregunta ¿cuánto?), la PLEM permite capturar la esencia de las decisiones de gestión, que a menudo son de naturaleza discreta y lógica (respondiendo a preguntas de ¿cuándo?, ¿cuál?, ¿sí o no?). Esta capacidad es la que eleva la optimización desde un nivel puramente operativo de asignación de recursos a un nivel de planificación táctica y estratégica, permitiendo modelar la interconexión de decisiones a lo largo del tiempo y el espacio, lo cual es fundamental para la gestión integral de una operación acuícola (Weintraub y Romero, 2006).

Para la resolución del modelo matemático resultante se integró una librería o *solver* llamado Gurobi. Según Gurobi Optimization (s.f.), los problemas de tipo PLEM presentan una alta complejidad computacional a medida que aumentan las variables (estanques y periodos de planificación), por lo que el uso de un *solver* especializado resulta indispensable. Gurobi emplea tanto algoritmos avanzados, entre ellos la ramificación y el acotamiento (*Branch-and-Bound*), como planos de corte (*Cutting Planes*), que permiten encontrar la solución óptima en tiempos de ejecución viables para el usuario. La implementación de esta herramienta, facilitada mediante una licencia académica, permite que el sistema genere planes estratégicos coordinados que garantizan un flujo de suministro continuo, materializando así la sinergia entre la simulación de la DS, el pronóstico basado en datos de la LSTM y la capacidad prescriptiva de la PLEM.

2.3.4.2.1 Método Big-M. La Big-M se define como una técnica de la PLEM que utiliza una constante de gran magnitud para expresar implicaciones lógicas y gestionar restricciones de forma dinámica (Piacentini et al., 2018).

2.3.4.3 Optimización Basada en Simulación. La optimización basada en simulación se refiere a la optimización de una función objetivo sujeta a restricciones, donde tanto la función objetivo como las restricciones son evaluables únicamente a través de una simulación estocástica (Satyajith Amaran y Bury, 2016). Esta característica la distingue de la programación matemática

algebraica tradicional, en la que las relaciones del sistema pueden expresarse de forma analítica y cerrada: en la optimización por simulación, el sistema es demasiado exigente para ser descrito con ecuaciones exactas, por lo que la simulación actúa como un evaluador de caja negra que estima el desempeño del sistema bajo distintas decisiones (Satyajith Amaran y Bury, 2016).

Su utilidad responde a una limitación fundamental: no hay ecuaciones analíticas confiables que describan con suficiente precisión el crecimiento de los peces bajo condiciones variables de campo. Ante ello, la simulación es una herramienta primaria para modelar sistemas robustos, especialmente cuando las técnicas analíticas típicas no están disponibles debido a la dimensionalidad de los supuestos del modelo (Taghizadeh, s.f.). Dado que la simulación por sí sola no puede encontrar y sugerir decisiones óptimas, se integra con técnicas de optimización exactas, heurísticas y metaheurísticas. Esta combinación de simulación y optimización ayuda a profundizar en la comprensión de los problemas, facilitando la toma de decisiones más inteligentes y rápidas (Taghizadeh, s.f.).

2.3.4.4 Dominios Dispersos (Sparse Matrices) en Modelos de Gran Escala. El interés en la dispersión surge porque su explotación conduce a ahorros computacionales críticos en problemas de matrices de gran tamaño (Brownlee, 2019). En este sistema, el optimizador PLEM utiliza variables binarias para coordinar múltiples estanques; no obstante, sin un filtrado biológico previo, el espacio de búsqueda superaría las decenas de millones de variables, tornando el problema *NP-hard* e intratable en tiempos prácticos. Mediante el uso de dominios dispersos, el sistema preselecciona únicamente las ventanas de siembra y cosecha que cumplen con los parámetros de biomasa y tasa de crecimiento proyectados por el modelo DS o la red LSTM, según el modo de la simulación (ver Capítulo 4), garantizando así que el flujo de suministro hacia el mercado sea técnica y biológicamente viable.

2.4 Marco Tecnológico

2.4.1 Arquitectura de Software: Diseño Dirigido por el Dominio

La ingeniería de software moderna, cuando se aplica a dominios desafiantes, actúa como el puente traductor entre el mundo físico (los estanques, los peces, el alimento) y el mundo digital (las funciones de pérdida, las matrices dispersas, las restricciones de optimización) (Rouhi y Lano, 2023). Históricamente, muchas aplicaciones informáticas han sido construidas utilizando arquitecturas centradas en los datos (Data-Centric) o guiadas por *frameworks* de infraestructura tecnológica, donde la lógica de las reglas del negocio termina peligrosamente acoplada a sentencias de bases de datos relacionales, a controladores de red o a rutinas de interfaz de usuario (Evans, 2003a). Este acoplamiento genera bases de código frágiles, imposibles de escalar y altamente resistentes a las modificaciones evolutivas.

Así pues, una alternativa desarrollada a este acoplamiento fuerte es el Diseño Dirigido por el Dominio (DDD), este es un enfoque de desarrollo de software propuesto por Evans (2003b), que sitúa el modelo del dominio de negocio en el centro de toda decisión de diseño. Su premisa fundamental es que el software debe expresar y reflejar el conocimiento profundo del problema que resuelve, empleando el mismo lenguaje que usan los expertos del dominio —lo que Evans denomina lenguaje ubicuo— tanto en la comunicación del equipo como en el código fuente. Esto implica organizar el código no alrededor de tecnologías o *frameworks*, sino alrededor de los conceptos del negocio.

DDD se complementa de manera natural con la Arquitectura Hexagonal, también denominada «puertos y adaptadores», propuesta por Cockburn (2005). Esta arquitectura busca crear estructuras débilmente acopladas donde los componentes de la aplicación puedan probarse de manera independiente, sin dependencias de almacenes de datos o interfaces de usuario, facilitando el cambio del *stack* tecnológico con un impacto mínimo o nulo en la lógica de negocio.

Para consolidar esta robustez, el diseño hizo uso extensivo de los patrones tácticos dictados

por Özkan et al. (s.f.):

- Entidades: Objetos de software que poseen una identidad única que trasciende el tiempo y sus atributos.
- Objetos de valor: Representaciones inmutables de medidas, descripciones o características del dominio que carecen de identidad propia, su inmutabilidad facilita cálculos predecibles libres de efectos colaterales en la red neuronal y el optimizador.
- Agregados: Clústeres de Entidades y Objetos de Valor tratados como una única unidad transaccional y de consistencia de datos.
- Servicios del dominio: Contiene la lógica de negocio pura y compleja que no pertenece naturalmente a una sola entidad.

2.4.2 Estándares de Desarrollo: Código Limpio

Tal término fue acuñado y sistematizado por Martin (2008) en su obra homónima, en la que argumenta que el código limpio no es únicamente aquel que funciona correctamente, sino aquel que puede ser comprendido y modificado con facilidad por cualquier miembro del equipo de desarrollo, no solo por quien lo escribió originalmente.

Los principios centrales de esta filosofía que estructuran la implementación del presente sistema son tres. El primero es el uso de nombres expresivos: los identificadores en el código reflejan directamente los conceptos del dominio piscícola, haciendo que la lectura del código sea coherente con el lenguaje del negocio. El segundo es el Principio de Responsabilidad Única (SRP, *Single Responsibility Principle*), según el cual cada clase o módulo debería tener una sola razón para cambiar (Martin, 2008). Este principio garantiza que cada unidad de código sea pequeña, enfocada y comprobable de forma aislada. El tercero es la separación estricta de capas: el código de dominio no conoce los detalles de infraestructura, y viceversa, lo que permite sustituir componentes tecnológicos sin efectos colaterales en la lógica de negocio.

La adherencia a estos estándares convierte cada módulo en una unidad independiente que puede probarse, modificarse y reemplazarse con mínima propagación de errores hacia el resto del sistema.

2.4.3 Lenguajes de Programación y Paradigmas

Con el fin de cumplir con los criterios técnicos de orquestación, persistencia y lógica de negocio, así como el acceso nativo a las bibliotecas de optimización y aprendizaje automático, se seleccionaron dos lenguajes de programación definidos según su rol en la arquitectura.

2.4.3.1 Java. Es un lenguaje de programación de propósito general, orientado a objetos, fuertemente tipado y compilado, cuya plataforma —la Máquina Virtual de Java (JVM)— ofrece portabilidad, robustez y un ecosistema maduro para aplicaciones empresariales de producción (Oracle Corporation, s.f.).

2.4.3.2 Python. Es un lenguaje interpretado, multiparadigma y de tipado dinámico, ampliamente adoptado en la comunidad científica y de inteligencia artificial por la riqueza de su ecosistema de bibliotecas especializadas (Python Software Foundation, s.f.).

2.4.3.3 PostgreSQL. Es un sistema de gestión de bases de datos objeto-relacional de código abierto (ORDBMS, por sus siglas en inglés *Object-Relational Database Management*). Admite gran parte del estándar SQL e incorpora características avanzadas como consultas complejas, claves foráneas, integridad transaccional y control de concurrencia multiversión, lo que lo convierte en una solución robusta y confiable para el almacenamiento y gestión de datos (The PostgreSQL Global Development Group, s.f.).

2.4.4 Entorno de Desarrollo y Herramientas

El desarrollo del sistema se apoyó en un conjunto de herramientas especializadas que soportan distintas etapas del ciclo de vida del software.

2.4.4.1 *Frameworks* de Desarrollo.

2.4.4.1.1 Spring Boot. Este *framework* facilita la creación de aplicaciones basadas en el ecosistema *Spring* listas para producción, que se ejecutan de forma autónoma con una configuración mínima (Spring Boot Team, s.f.).

2.4.4.1.2 FastAPI. Es un *framework* moderno y de alto rendimiento para construir *API*'s con Python, basado en las anotaciones de tipo estándar del lenguaje y conforme a los estándares abiertos OpenAPI y JSON Schema, con generación automática de documentación interactiva (FastAPI, s.f.).

2.4.4.1.3 Streamlit. Es un *framework* de código abierto que permite transformar *scripts* de Python en aplicaciones web interactivas en minutos, sin necesidad de código *HTML*, *CSS* o *JavaScript* (Streamlit, s.f.).

2.4.4.2 Herramientas de Apoyo.

2.4.4.2.1 Postman. Es una plataforma de colaboración para el desarrollo y prueba de APIs que permite construir, ejecutar y documentar solicitudes HTTP de forma interactiva, sin necesidad de escribir código (Postman Inc, s.f.).

2.4.4.2.2 StarUML. Es una herramienta de modelado UML que soporta la creación de diagramas de clases, secuencia, componentes y casos de uso, orientada al diseño de arquitecturas de software antes de su implementación (StarUML, s.f.).

3. Metodología

La construcción de un sistema cibernético y de soporte a la decisión no se reduce a la codificación de algoritmos; requiere partir de una comprensión documentada del contexto socio-productivo en el que ese sistema habrá de operar. Por ello, este capítulo declara las decisiones metodológicas que gobernaron el proyecto, así como las reglas de evaluación del artefacto descrito en el Capítulo 4, que fijan los criterios contra los cuales se contrastaron las evidencias obtenidas en el Capítulo 5.

3.1 Enfoque de Investigación

La investigación se estructura bajo un enfoque cualitativo-técnico, anclado en los principios epistemológicos de la Investigación Basada en el Diseño[1] (*Design Science Research* o *DSR*, por sus siglas en inglés) (Baskerville et al., 2026). Bajo esta lente, la vertiente cualitativa del proyecto se fundamentó en el análisis documental de las barreras culturales hacia la asociatividad y las lógicas empíricas de los acuicultores para la toma de decisiones, caracterizadas a partir de la literatura técnica y socioeconómica del sector consultada en el Capítulo 2, y no en un proceso propio de recolección de datos primarios con productores. Por su parte, la dimensión técnica consistió en la instanciación formal de este conocimiento en una arquitectura computacional robusta. El paradigma cualitativo-técnico reconoce que las especificaciones de software no son dictados estáticos, sino hipótesis de diseño que deben validarse continuamente. Así, el sistema multi-

[1] DSR es un paradigma de investigación orientado a la resolución de problemas mediante la creación y evaluación de artefactos tecnológicos.

estanque es tratado como un «artefacto» científico, cuyo proceso de construcción generó nuevo conocimiento sobre cómo aplicar la inteligencia artificial y la teoría de control a las cadenas de suministro agroalimentarias.

3.2 Proceso de Desarrollo Evolutivo

Si bien en las fases de concepción inicial se planteó un modelo de desarrollo lineal, fundamentado en el paradigma tradicional en cascada, la integración de componentes computacionales heterogéneos[2] introdujo un nivel de incertidumbre algorítmica que exigió una transición imperativa hacia un paradigma de desarrollo evolutivo (Eby, 2016). Este enfoque dinámico facilitó un refinamiento continuo del sistema mediante ciclos iterativos de especificación, construcción y validación concurrente. Dicha progresión aseguró que la arquitectura de software final se adaptara de manera precisa a las necesidades, restricciones y volatilidades del dominio piscícola colombiano. Para ilustrar las divergencias fundamentales que justificaron esta transición arquitectónica y metodológica, la Tabla 1 detalla la comparación técnica entre el modelo tradicional en cascada y el enfoque evolutivo adoptado.

3.3 Estrategia de Validación y Pruebas

En relación con los principios de funcionamiento, las pruebas se organizan por subsistema —pronóstico, optimización y simulación, cuya construcción se formaliza en el Capítulo 4— y culminan con la verificación de la integración de extremo a extremo. Aunque la exposición sigue esta división por subsistema, la evidencia que cada uno produce se orienta a los objetivos específicos del proyecto, a saber, la reingeniería del motor de simulación sustenta la arquitectura escalable del OE1, el subsistema de pronóstico respalda el OE2, el de optimización el OE3, y la validación integrada el OE4.

[2]Componentes tales como la simulación basada en Dinámica de Sistemas (DS), el modelado matemático mediante Programación Lineal Entera-Mixta (PLEM) y el entrenamiento de redes neuronales recurrentes (LSTM)

Tabla 1
Análisis comparativo de las metodologías en cascada y evolutiva

Dimensión del Proyecto	Enfoque Tradicional Secuencial (Cascada)	Enfoque de Desarrollo Evolutivo
Naturaleza de los Requisitos	Completamente definidos, congelados y documentados en la fase inicial del proyecto.	Dinámicos, heurísticos y sujetos a descubrimientos basados en el comportamiento de los datos.
Manejo de la Incertidumbre	Extremadamente baja tolerancia; se asume un comportamiento de software determinista.	Altamente resiliente; la incertidumbre se considera una variable fundamental que guía el diseño.
Flujo de la Ejecución Tecnológica	Fases estrictamente bloqueantes (Diseño → Implementación → Pruebas).	Ciclos iterativos de experimentación, ajuste de hiperparámetros y validación empírica continua.
Punto de Integración de Componentes	Estrategia «Big Bang» al final del ciclo de vida del desarrollo.	Integración continua y temprana; el simulador, el predictor y el optimizador co-evolucionan simultáneamente.
Criterio Principal de Éxito	Cumplimiento estricto del alcance original y los cronogramas predefinidos.	Precisión del modelo de inteligencia artificial y utilidad práctica de los planes de optimización generados.

Nota. Adaptado de Kavlakoglu, s.f.

3.3.1 Validación del Subsistema de Pronóstico

El subsistema de pronóstico se juzga por su capacidad para reproducir la dinámica biológica del simulador de referencia y, sobre todo, por generalizarla a estanques que nunca vio durante el entrenamiento; pues de esa fidelidad depende que las proyecciones de crecimiento sobre las que decide el optimizador sean confiables. En consecuencia, la validación de este módulo combina una métrica de error acotada a la escala de cada variable, una inspección visual del comportamiento predicho y una batería de tres niveles de exigencia creciente que separa la capacidad de aprender de la capacidad de generalizar.

3.3.1.1 Métrica de evaluación. El desempeño se mide mediante la raíz del error cuadrático medio (*RMSE*), que expresa el error promedio de predicción en las mismas unidades que la variable objetivo. Dado que el RMSE por sí solo depende de la escala de la variable, se complementa con el RMSE normalizado (*NRMSE*), que lo expresa como porcentaje respecto al rango de la variable real, de la siguiente manera:

$$\text{NRMSE}(\,\%) = \frac{\text{RMSE}}{Y_{\text{máx}} - Y_{\text{mín}}} \times 100 \tag{3.1}$$

Sobre esta métrica se fija el criterio cuantitativo de aceptación del subsistema, el cual considera que el modelo reproduce fielmente la dinámica del simulador cuando su NRMSE de validación permanece por debajo de un umbral máximo admisible del 10 % en ambas variables objetivo —el incremento de peso y la ración—, valor que corresponde al límite superior de la categoría «excelente» en la clasificación de desempeño de modelos de simulación biológica propuesta por Despotovic et al. (2015) y ampliamente adoptada en la literatura de modelado de cultivos y crecimiento animal.

3.3.1.2 Validación visual mediante gráficas. Como complemento a las métricas numéricas, se emplearon gráficas para inspeccionar el comportamiento del modelo de forma cualitativa, es decir, se emplearon las trayectorias diarias del incremento del peso y la ración previstas por el LSTM frente a las del simulador; también se comparó el peso acumulado a lo largo del horizonte de predicción. Asimismo, se validan las curvas de pérdida[3] de entrenamiento y validación por *epoch*.

3.3.1.3 Niveles de evaluación. La estrategia de validación se estructuró en tres niveles progresivos para verificar tanto la capacidad de aprendizaje como la de generalización de la red:

a) Conjunto de entrenamiento: Se calcula el RMSE sobre los mismos datos utilizados durante el entrenamiento. Un error alto en este nivel indica un subajuste, esto es, el modelo no logró

[3]Las curvas de pérdida permiten evidenciar si el flujo de aprendizaje convergió de forma estable, identificar el *epoch* en que se obtuvo el modelo óptimo y verificar que no existe sobreajuste a través de la distancia entre ambas curvas a lo largo del entrenamiento.

aprender la dinámica del sistema.

b) Conjunto de validación: Se calcula el RMSE sobre los estanques reservados para validación, que no participaron en ninguna fase del entrenamiento. Este nivel mide la capacidad de generalización ante secuencias no vistas. Una diferencia significativa entre el RMSE de entrenamiento y el de validación sería indicativa de sobreajuste.

c) Conjunto externo independiente: Se genera un dataset adicional con una semilla base distinta a la utilizada durante el entrenamiento, produciendo estanques con condiciones iniciales y trayectorias de temperatura diferentes. El RMSE calculado sobre este conjunto constituye la evidencia más robusta de generalización, pues descarta que los resultados del Nivel 2 sean consecuencia de una partición 80/20 favorable.

3.3.2 Validación del Subsistema de Optimización

A diferencia del subsistema de pronóstico, cuya evaluación se apoya en la comparación contra un valor de referencia conocido, el subsistema de optimización no posee un «valor verdadero» contra el cual contrastar sus salidas: cada plan generado es, por construcción, la solución óptima (o casi óptima) del modelo MILP para el conjunto de parámetros recibido. Por ello, su comprobación no se dirige a medir un error, sino a caracterizar su comportamiento mediante un diseño experimental por bloques. El modelo matemático presentado en el Capítulo 4 identifica tres dimensiones independientes que determinan la complejidad del problema: el número de estanques Γ, el número de pedidos Ψ y el horizonte de planificación T. Cada una de ellas se adoptó como un factor del diseño experimental, donde cada bloque varía deliberadamente una sola dimensión mientras las demás se mantienen fijas, a fin de aislar el efecto individual de cada factor antes de observar su interacción conjunta. La Tabla 2 resume el propósito y la configuración de dichos bloques.

Aunque los seis bloques comparten la misma infraestructura experimental, la evidencia que

Tabla 2
Diseño experimental por bloques para la validación del sistema

Bloque	Γ	Ψ	T	**Propósito**
0 – Control	1	1	fijo	Verificar, en el caso más simple posible, que el ciclo completo de siembra-cosecha se ejecuta correctamente, y obtener una línea base de tiempo de respuesta.
A – Sensibilidad a Γ	variable	fijo	fijo	Caracterizar cómo escala el tiempo de respuesta al aumentar el número de estanques de la asociación.
B – Sensibilidad a Ψ	fijo	variable	fijo	Caracterizar cómo escala el tiempo de respuesta al aumentar el número de pedidos simultáneos.
C – Sensibilidad a T	fijo	fijo	variable	Identificar el horizonte mínimo de planificación a partir del cual el sistema logra producir planes viables, dado el ciclo biológico de la especie.
D – Escenarios combinados	variable	variable	fijo (amplio)	Evaluar la viabilidad y eficiencia de los planes generados al escalar conjuntamente Γ y Ψ, simulando asociaciones de tamaño creciente.
E – Sensibilidad a la función objetivo	fijo	fijo	fijo	Calibrar la penalización por déficit ϵ de la función objetivo, justificando el valor adoptado en lugar de asumirlo.

Nota. $\Gamma \equiv$ cantidad de estanques, $\Psi \equiv$ cantidad de pedidos, $T \equiv$ valor del horizonte

producen no pertenece a un solo objetivo, sino que se reparte según la propiedad que cada uno aísla: el Bloque A, al caracterizar cómo escala el tiempo de respuesta con el número de estanques, constituye la evidencia cuantitativa de la escalabilidad exigida por el OE1; los Bloques C y E respaldan la validez del plan estratégico del OE3; y los Bloques 0, B y D —junto con el análisis de viabilidad económica— sustentan la evaluación de eficiencia y viabilidad del OE4.

Aun así, para que las comparaciones entre bloques y entre configuraciones dentro de un mismo bloque fueran válidas, se fijó un conjunto común de parámetros que permanece constante a lo largo de todo el experimento, salvo en el bloque donde dicho parámetro es precisamente la

variable de estudio: la fecha inicial de la simulación, el modo de simulación (DS, para aislar el efecto del optimizador del de la red neuronal), la penalización por déficit ($\epsilon = 10$), la semilla aleatoria (42) y el número de réplicas Monte Carlo por proyección (20). Mantener estos valores fijos garantiza que cualquier variación observada en el tiempo de respuesta o en la calidad del plan sea atribuible exclusivamente a la dimensión de estudio, y no a diferencias incidentales en la configuración del experimento.

3.3.3 Validación del Subsistema de Simulación

El subsistema de simulación es la fuente de verdad biológica de todo el sistema, ya que alimenta el entrenamiento de la red neuronal y sustenta las proyecciones sobre las que decide el optimizador. Dado que es el mismo subsistema quien genera su propia referencia contra la cual contrastarse, su validación tampoco se orienta a medir un error, sino a comprobar las dos propiedades de las que depende el resto del sistema, por un lado, que el motor sea reproducible, y por el otro, que el percentil de referencia sobre el que se construyen las proyecciones sea estable.

La reproducibilidad se comprueba verificando el determinismo del motor, lo que significa que, al realizar dos ejecuciones con idéntica semilla, se debe producir exactamente la misma trayectoria. La comprobación es directa: se generan dos conjuntos de datos con la misma semilla y se contrastan celda a celda, exigiendo una diferencia máxima absoluta nula. De forma complementaria, se utiliza un control positivo con una semilla distinta, cuya trayectoria sí debe diferir, a fin de descartar que la identidad provenga de una ausencia de aleatoriedad y no de un sembrado genuinamente reproducible.

La estabilidad se examina mediante la convergencia del muestreo Monte Carlo. Como las réplicas intervienen únicamente en la proyección biológica que alimenta al optimizador y no en la física realizada del estanque, su efecto se observa sobre las variables biológicas proyectadas por individuo (el peso y la supervivencia). Por ende, al mantener fijos la semilla, el estanque, la demanda y repetir la proyección variando el número de réplicas, se puede registrar cómo dicha

estimación se estabiliza a medida que las réplicas aumentan.

Como extensión cualitativa, se documenta mediante un cliente REST la correcta ejecución de la máquina de estados de cada estanque a lo largo de su ciclo de vida —de disponible a sembrado, en engorde y finalmente cosechado—, evidenciando que las transiciones ocurren en los instantes previstos por el plan.

3.3.4 Integración y Sistema Completo

Además de la validación individual de cada subsistema, la correctitud del sistema integrado exige verificar que los datos fluyen sin pérdidas ni transformaciones indebidas a través de la cadena completa: la base de datos PostgreSQL como repositorio central, el motor Java como orquestador de la lógica de negocio, el módulo Python como solucionador MILP y el panel Streamlit como capa de visualización. Una falla en cualquier frontera de esta cadena, por ejemplo, una conversión de unidades incorrecta entre Java y Python, o una consulta SQL que omita parte de los registros, puede producir resultados aparentemente válidos en cada subsistema en aislamiento pero incorrectos en el sistema completo.

La estrategia de validación de integración se apoyó en el Bloque 0 como prueba de extremo a extremo, que comprende la ejecución del ciclo completo de un único estanque y un único pedido sobre el sistema real, cualquier ruptura en la cadena de comunicación se manifiesta como un error observable (excepción, tiempo de espera agotado o resultado vacío) antes de que el experimento pueda completarse. El hecho de que el Bloque 0 produjera un cronograma válido, ejecutara la cosecha en el período previsto y reportara los indicadores económicos correctos en el panel de Streamlit constituyó la evidencia primaria de que la integración entre los cuatro componentes funciona.

En apoyo de lo anterior, se verificaron tres invariantes de frontera entre componentes:

a) Java y Python: el motor Java serializa el estado del sandbox (capacidades, lotes activos y pedidos pendientes) como un objeto JSON que es recibido e interpretado sin pérdida por

el módulo de optimización Python, el cual devuelve el plan resultante en el mismo formato estructurado. La integridad de esta frontera se comprobó contrastando manualmente los valores de biomasa y fechas enviados en la solicitud con los que el optimizador reportó en su salida durante las primeras ejecuciones del Bloque 0.

b) Java y PostgreSQL: el motor persiste el estado de cada iteración diaria en la base de datos relacional y lo recupera en la iteración siguiente sin que se acumulen registros duplicados ni se pierdan actualizaciones. Esta propiedad se verificó consultando directamente las tablas `estados_lote` y `escenarios` al finalizar el Bloque 0 y comprobando que el número de filas coincidía con el número de iteraciones ejecutadas.

c) PostgreSQL y Streamlit: el panel de visualización consulta los datos del escenario activo a través de la API REST del motor Java y los presenta sin transformaciones adicionales. Se verificó que los valores numéricos de biomasa entregada, utilidad y cumplimiento mostrados en el panel coincidieran con los almacenados en la base de datos para el mismo `escenario_id`.

En este mismo diseño por bloques, las configuraciones se registraron en dos familias de métricas: una de eficiencia y otra de viabilidad del plan. La primera es el tiempo de respuesta, entendido como el intervalo transcurrido entre el envío de la solicitud de iteración de los escenarios y la respuesta del servidor. Para los Bloques A, B y C se cronometró únicamente la primera invocación del horizonte deslizante ($t_0 = 0$), instante en el que todos los estanques se encuentran disponibles y, por tanto, el espacio de combinaciones Ω alcanza su tamaño máximo para esa configuración; las invocaciones posteriores, al heredar estanques ya ocupados, son estrictamente menos costosas de proyectar. La segunda es la viabilidad del plan, evaluada en el Bloque D ejecutando el ciclo completo del horizonte deslizante hasta el cumplimiento o vencimiento de la totalidad de los pedidos, y registrando por escenario el porcentaje de cumplimiento (biomasa entregada sobre biomasa requerida), el exceso de producción y el déficit acumulado. A estas dos familias se añade, de forma paralela, un conjunto de indicadores económicos consolidados sobre el plan ejecutado (utilidad neta, retorno sobre la inversión y utilización de los estanques), incorporado

a solicitud del director del proyecto por su condición de factor determinante de la viabilidad práctica de un plan en el contexto colombiano.

4. Desarrollo del Sistema

Este capítulo describe la instanciación del diseño metodológico presentado en el Capítulo 3, esto es, la construcción del artefacto que, bajo el paradigma DSR, constituye el vehículo de generación de conocimiento del proyecto. La exposición sigue el orden en que se tomaron las decisiones de ingeniería: primero se presenta el diseño arquitectónico que gobierna la estructura del sistema; después, el análisis funcional que delimita sus responsabilidades; a continuación, el desarrollo de los tres subsistemas; y, finalmente, la integración y los contratos de comunicación entre componentes.

4.1 Diseño Arquitectónico del Sistema

Para gobernar la complejidad técnica y conceptual descrita anteriormente, y asegurar que el artefacto resultante no se degrade en lo que Laribee (2009) denomina el anti-patrón de «gran bola de lodo» (*Big Ball of Mud*), el diseño estructural, la orquestación de componentes y la disposición de los directorios de código se fundamentaron en los preceptos de la Arquitectura Basada en el Dominio (*Domain-Driven Design*, o DDD). En consecuencia, el sistema se concibe como una terna de subsistemas autónomos: un motor de simulación sobre el ecosistema Java, un módulo de redes neuronales en Python y una infraestructura de optimización matemática mediante la librería Gurobi.

En aras de gestionar interdependencias y evitar el acoplamiento tóxico, el modelado estratégico de DDD opera mediante la definición de Contextos Delimitados[1] (*Bounded Contexts*) y sus

[1] Un Contexto Delimitado establece un límite semántico y tecnológico claro donde un modelo de dominio específico tiene validez absoluta (Laribee, 2009).

relaciones de integración, documentadas en el Mapa de Contexto (*Context Maps*). De esa manera, se formalizaron tres delimitaciones:

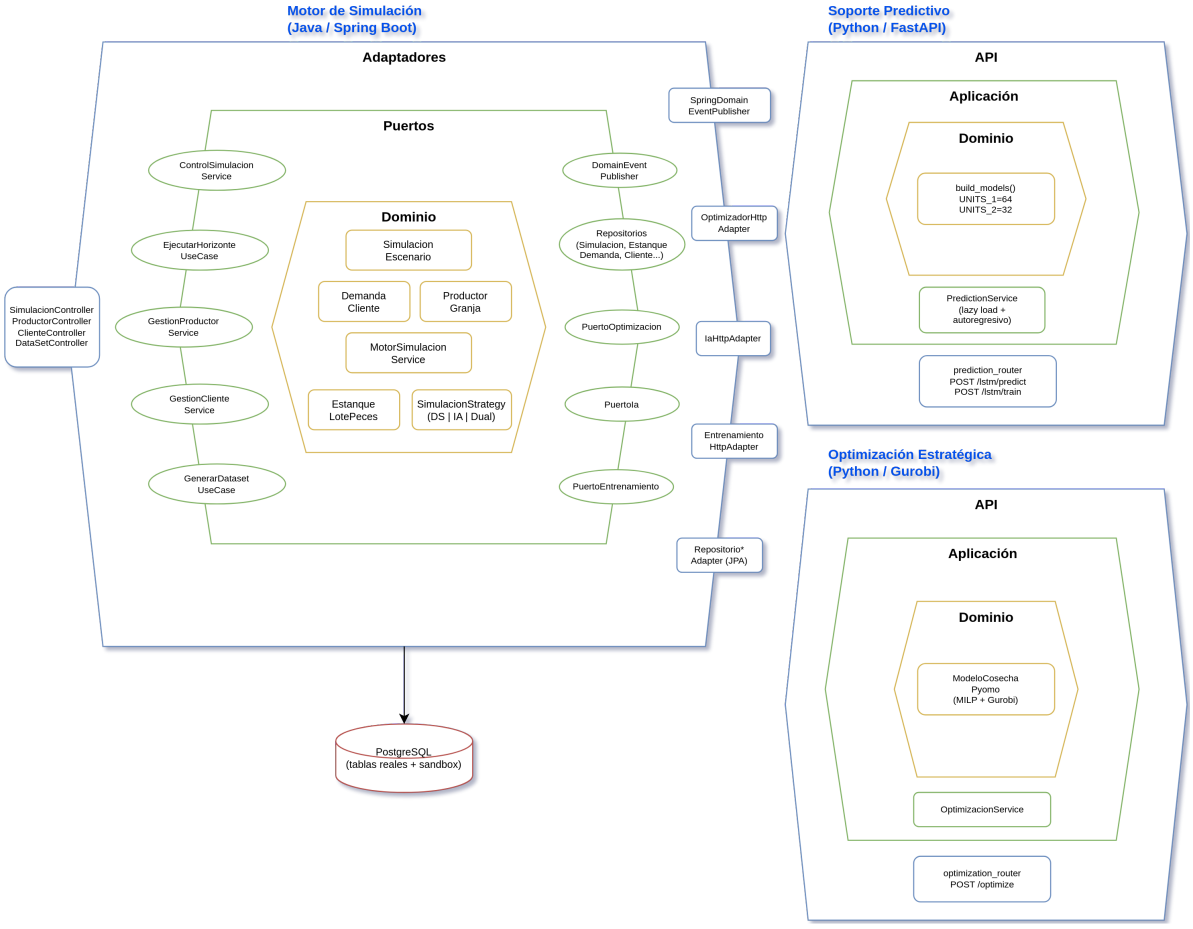
- Contexto Operativo de Simulación (Java): Responsable de la ejecución de las reglas de la DS y de la gestión de los agregados del dominio: productores, estanques, demandas y escenarios de simulación. Su lenguaje ubicuo describe entidades concretas del negocio: `Estanque`, `LotePeces`, `Simulacion`, `Demanda`.
- Contexto de Soporte Predictivo (Python): Dominio estadístico aislado, diseñado para el entrenamiento y la inferencia de la red LSTM. Su lenguaje ubicuo no describe estanques, sino secuencias temporales, tensores, compuertas de olvido, escaladores y métricas de error.
- Contexto Estratégico de Optimización (Python/Gurobi): Dominio prescriptivo donde la información biológica proyectada se transforma en variables de decisión binarias y continuas, bajo las restricciones operativas de la PLEM.

Con el objetivo de asegurar que estos contextos convivan sin comprometer la autonomía, se adoptó el patrón de Arquitectura Hexagonal, cuya estructura se ilustra en la Figura 5. El núcleo del hexágono lo ocupa el modelo de dominio puro: los agregados junto al servicio de dominio `MotorSimulacionService` y la familia de estrategias `SimulacionStrategy`. Ninguno de estos elementos tiene dependencia de Spring Boot, JPA, TensorFlow ni de ningún otro detalle de infraestructura. La capa intermedia la conforman los puertos: las interfaces de entrada definen el contrato que la capa de aplicación expone hacia el exterior, las interfaces de salida definen lo que el dominio necesita del exterior sin conocer su implementación. Finalmente, la capa exterior la ocupan los adaptadores: a la izquierda, los controladores REST actúan como adaptadores primarios que traducen las peticiones HTTP hacia los puertos de entrada; a la derecha, las interfaces actúan como adaptadores secundarios que implementan los puertos de salida conectando el dominio con FastAPI y PostgreSQL, respectivamente.

En una arquitectura monolítica tradicional, tales capas habrían quedado profundamente entrelazadas con las clases de simulación o las consultas a la base de datos, convirtiendo al

Figura 5

Arquitectura hexagonal del sistema integrado



algoritmo en un componente cautivo del simulador virtual (Özkan et al., s.f.). Por supuesto, al desacoplar las responsabilidades, se asegura no solo la mantenibilidad del sistema multi-estanque, sino también la reutilización del dominio de optimización en otros escenarios productivos del sector piscícola.

4.2 Análisis Funcional

Previo al desarrollo del proyecto, se realizó un análisis funcional del sistema antes de su materialización, con el propósito de delimitar su alcance operativo, identificar los actores

involucrados y establecer los casos de uso que orientaron las decisiones del diseño de la capa de aplicación. En atención al carácter investigativo del proyecto, este análisis no se derivó de un proceso formal de elicitación de requisitos con usuarios finales; por el contrario, surgió de la especificación técnica del dominio piscícola y de los objetivos específicos definidos. En consecuencia, los casos de uso documentados reflejan las capacidades que el sistema efectivamente implementa, y no una visión de producto orientada al mercado.

De ahí que la Figura 6 presente el Diagrama de Casos de Uso del sistema y las relaciones entre los requisitos identificados. Por tanto, se reconocen tres actores principales cuyos roles y responsabilidades son independientes entre sí:

- Operador de la asociación: Representa al agente responsable de gestionar el contexto de negocio de la asociación piscícola. Sus interacciones con el sistema comprenden el registro de la infraestructura productiva —productores, granjas y estanques— y la configuración de la demanda comercial mediante el registro de clientes, la creación de pedidos puntuales y la firma de contratos recurrentes. En el estado actual del sistema, este actor no dispone de una interfaz gráfica dedicada; sus operaciones se ejecutan directamente sobre la API REST expuesta por el motor de simulación.

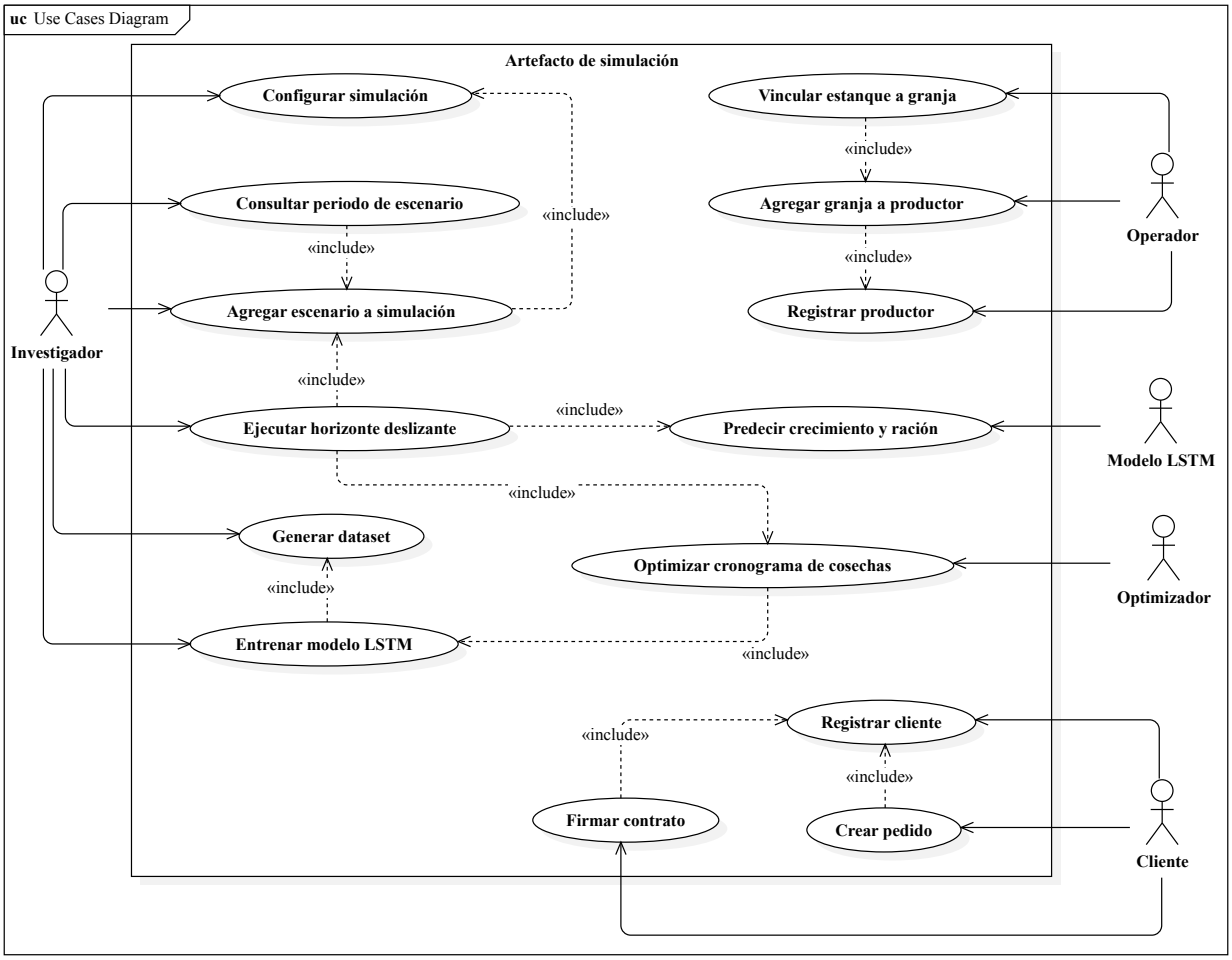
- Cliente: Representa al agente externo generador de la demanda que el sistema debería satisfacer. Si bien en la implementación actual el Cliente no interactúa directamente con la plataforma —sus pedidos y contratos son registrados por el Operador en su nombre—, se modela como actor independiente porque sus casos de uso asociados están implementados como operaciones diferenciadas en la capa de aplicación y la arquitectura prevé su activación futura mediante un módulo de autenticación y una interfaz propia, sin que ello requiera modificaciones al dominio existente.

- Investigador: Es el actor central del proyecto. Controla el ciclo de vida completo de la simulación: configura los parámetros experimentales, compone los escenarios a partir del estado registrado por el Operador, ejecuta los horizontes deslizantes y consulta los resultados

para su análisis. Adicionalmente, gestiona el ciclo de entrenamiento del modelo LSTM mediante la generación de datasets sintéticos.

Figura 6

Diagrama de casos de uso del sistema



Asimismo, la Tabla 3 establece la correspondencia entre los casos de uso referidos y los objetivos específicos del proyecto, para evidenciar el alcance funcional implementado, el cual responde directamente a las metas de investigación formuladas. La tabla precedente pone en manifiesto una limitación que conviene señalar con transparencia: el cuarto objetivo específico, orientado a la validación mediante escenarios, no tiene un caso de uso exclusivo en el sistema porque la validación no es una funcionalidad del artefacto, sino una actividad de investigación que consume sus resultados. La consulta de períodos de escenario es la operación que da soporte

técnico a esa validación, pero el análisis interpretativo de los resultados ocurre fuera del sistema, específicamente en el Capítulo 5 de este documento.

Tabla 3
Trazabilidad entre casos de uso y objetivos específicos

Caso de uso	Objetivo
Registrar productor, agregar granja, vincular estanque	OE1
Configurar simulación, agregar escenario, ejecutar horizonte deslizante	OE1
Generar dataset, entrenar modelo LSTM	OE2
Predecir crecimiento y ración	OE2
Optimizar cronograma de cosechas	OE3
Registrar cliente, crear pedido, firmar contrato	OE3
Consultar período de escenario	OE4

4.3 Desarrollo e Implementación de los Subsistemas

La naturaleza iterativa y flexible del enfoque evolutivo adoptado permitió que los subsistemas no se concibieran como entidades aisladas, sino como componentes interdependientes de una arquitectura integral desarrollados de manera concurrente. Esta simultaneidad en la construcción fue fundamental para propiciar la preservación de la integridad semántica y la consistencia del lenguaje ubicuo a través de los diferentes contextos delimitados.

No obstante, la ejecución técnica obedeció a una jerarquía de prioridades estratégicas que permitió la instanciación progresiva y estable del artefacto. En los apartados siguientes, se detalla el curso procedimental seguido para la construcción e integración de cada subsistema, estructurado bajo el orden de precedencia técnica establecido durante la investigación.

4.3.1 Formulación del Modelo de Optimización

La primera etapa metodológica consistió en formular el modelo de optimización encargado de determinar el cronograma estratégico de siembra y cosecha de la asociación. Para ello, fue

necesario responder una cuestión central: ¿qué planificación maximiza el beneficio colectivo sin incumplir la demanda agregada?

La resolución de esta interrogante exigió, en primera instancia, la abstracción y formalización matemática de las unidades críticas del sistema productivo, siendo estas las siguientes entidades vectoriales base:

- Estanque (i): Representado como una unidad discreta de producción condicionada por estados lógicos binarios (activación de siembra o cosecha) y delimitada por parámetros continuos de capacidad de biomasa total (K_i).
- Cronograma (t): Estructurado como un horizonte de planificación discreto en intervalos diarios ($t \in \{0, \ldots, T\}$), donde T delimita la ventana temporal de evaluación estratégica.
- Demanda (j): Representada como un vector dinámico multidimensional de requisitos externos. Esta entidad indexa tres ejes críticos: el rango de peso promedio del pez individual aceptado por el mercado ($\omega_j^{[\text{mín,máx}]}$), el volumen de biomasa total solicitado (ψ_j) y el periodo de tiempo estricto de entrega (T_j).

Ante este panorama, la evaluación de alternativas de desarrollo evidenció una restricción crítica: la complejidad computacional del dominio del problema. A saber, la asignación óptima de eventos de siembra y cosecha comparten características con diversos problemas de planificación y programación de la producción, conocidos por presentar una elevada complejidad computacional, especialmente a medida que aumenta el horizonte temporal y el número de estanques considerados. Intentar resolver este modelo mediante la lógica imperativa tradicional en el ecosistema Java (a través de estructuras de control anidadas o heurísticas codificadas a mano) resultaría en una complejidad ciclomática inmanejable para el software y en una degradación severa de la calidad de las soluciones, quedando propensas a óptimos locales.

Por el contrario, la delegación de esta carga computacional a la librería Gurobi hace posible encapsular la complejidad algorítmica directamente en un motor de optimización comercial de

alto rendimiento. Esto certifica la obtención de soluciones con cotas de optimalidad global demostrables matemáticamente, liberando al simulador en Java de la lógica de resolución combinatoria y permitiéndole enfocarse estrictamente en la ejecución transaccional y la gestión de estados del sistema.

La construcción del modelo matemático final no se concibió como una fase estática, sino como una implementación incremental fundamentada en la relajación progresiva de supuestos. Este enfoque de refinamiento continuo tolera la transición de una abstracción teórica elemental hacia un modelo matemático robusto, capaz de asimilar las restricciones reales de la asociatividad piscícola. Para documentar esta evolución y evitar puntos ciegos en la formulación, el modelo se estructuró en diferentes fases secuenciales, descritas en el Apéndice A.

Aun así, con todo ese progreso, el modelo final presenta un conjunto de supuestos respaldados por la práctica del sector piscícola colombiano:

- Cultivo monosexo: El sistema asume poblaciones exclusivamente masculinas en todos los estanques. Esta condición es la práctica estándar en el cultivo comercial de piscicultura en Colombia, donde la reversión sexual es el método más empleado para evitar la reproducción espontánea[2] (Merino Archila et al., 2006).

- Densidad como parámetro fijo por estanque: La capacidad máxima de biomasa recomendada de cada estanque se calcula como el producto de su volumen y su densidad de siembra; ambos parámetros dependen del tipo de estanque (Merino Archila et al., 2006), por lo que se asume que el productor reconoce sus propias limitaciones.

- Rango de pesos promedio aceptable por la demanda: Desde el punto de vista comercial, el mercado paga por kilogramo de peso vivo dentro de rangos de talla predefinidos. Esta realidad impone que, por ejemplo, una cosecha de peces con peso promedio inferior a 150 g es rechazada por el mercado o pagada a precios no rentables (Merino Archila et al., 2006).

[2]La reproducción ocurre a partir del segundo o tercer mes de cultivo, genera sobrepoblación de alevines que compiten por recursos con los individuos en engorde.

De acuerdo con aquellos supuestos, se desglosa matemáticamente la formulación del modelo, detallando sus índices, parámetros, variables de decisión y restricciones:

4.3.1.1 Conjuntos e índices. Los conjuntos definen el alcance espacial y temporal del optimizador, permitiendo la gestión individualizada de cada recurso dentro de la asociación piscícola:

- $i \in \gamma = \{1,\ 2,\ \ldots,\ \Gamma\}$: Conjunto de estanques de la asociación.
- $t, s \in \tau = \{0,\ 1,\ \ldots,\ T\}$: Conjunto de periodos discretos en el horizonte temporal.
- $j \in \psi = \{1,\ 2,\ \ldots,\ \Psi\}$: Conjunto de pedidos determinados por el mercado.
- $\Omega = \{(i,\ s,\ t)\}$: Conjunto de todas las tuplas válidas (dominio disperso).

  Nota metodológica: Las tuplas se generan únicamente para los instantes de cosecha con demanda vigente, es decir, $t \in \{T_j \mid j \in \psi\}$. Si el estanque i está vacío, se generan tuplas $\forall_s \in \tau \wedge t > s$; si ya está ocupado, se genera un único periodo de siembra representativo ($s = -1$, denotando el estado heredado) y se evalúan las posibles cosechas $t > -1$. Adicionalmente, el motor de simulación poda las rutas cuya biomasa proyectada $B_{i,s,t}$ excede la capacidad de carga tolerada del estanque, $K_i \cdot (1 + \theta)$, antes de construir el modelo. La única excepción son los lotes heredados ($s = -1$) que carecen de alguna ruta dentro de dicha capacidad: para no dejar el estanque sin salida factible, se conserva forzosamente su cosecha válida más temprana.
- $\Omega_i \subseteq \Omega$: Subconjunto de tuplas válidas exclusivamente para el estanque i.

4.3.1.2 Parámetros del sistema. Los parámetros representan los valores de entrada fijos obtenidos del subsistema de simulación y las condiciones contractuales del mercado (véase la Tabla 4).

Tabla 4
Parámetros definidos en el modelo matemático

Símbolo	Descripción	Dominio	Unidad
$\omega_{i,s,t}$	Peso promedio esperado del pez en el estanque i, sembrado en s y cosechado en t.	$\mathbb{R}^+$	g
μ_i	Volumen máximo recomendado del estanque.	$\mathbb{R}^+$	m^3
ρ_i	Densidad máxima recomendada del estanque.	$\mathbb{R}^+$	kg/m^3
$B_{i,s,t}$	Biomasa proyectada de la ruta (i, s, t), estimada por el simulador como el producto de la población inicial, la probabilidad de supervivencia y el peso promedio esperado.	$\mathbb{R}^+$	kg
θ	Tolerancia admitida sobre la capacidad de carga del estanque.	$\mathbb{R}^+$	-
ψ_j	Biomasa total requerida por el pedido j.	$\mathbb{R}^+$	kg
T_j	Instante de tiempo exacto en el que el pedido j exige su entrega.	$T \in \tau$	-
$\omega_j^{[\text{mín,máx}]}$	Intervalo de peso promedio del pez aceptable para el pedido j.	$\mathbb{R}^+$	g
J_i	Conjunto de pedidos asignados originalmente al estanque i.	$J \subseteq \psi$	-
$\delta_{i,j,s,t}$	Parámetro indicador que valida si el pez del estanque i cumple con el peso y tiempo del pedido j.	$\{0, 1\}$	-
M	Cota superior de cualquier asignación individual, usada para la activación de restricciones lógicas. Se calcula como el máximo entre la mayor capacidad de carga tolerada y la mayor biomasa proyectada.	$\mathbb{R}^+$	-

4.3.1.3 Variables de decisión. El modelo debe determinar las acciones óptimas de cosecha para satisfacer la demanda colectiva. Así, se definen las siguientes variables de decisión:

- $\chi_{i,s,t} \in \{0, 1\} : \begin{cases} 1, & \text{Si el estanque } i \text{ se siembra en } s \text{ y se cosecha en } t. \\ 0, & \text{En otro caso.} \end{cases}$
- $v_{i,j} \in \mathbb{R}^+$: Biomasa total cosechada en el estanque i asignado al pedido j.
- $H_j^- \in \mathbb{R}^+$: Variable de desviación negativa (déficit). Representa la biomasa faltante para cumplir el pedido j.
- $H_j^+ \in \mathbb{R}^+$: Variable de desviación positiva (exceso). Es la biomasa producida por encima de lo exigido por el pedido j; al recogerse el lote completo, este sobrante no se descarta sino que el productor lo comercializa por canal directo (venta directa).

4.3.1.4 Función objetivo (Z). El objetivo del modelo es minimizar las desviaciones de producción de la asociación piscícola, penalizando el déficit con mayor severidad que el exceso —en especial el de los pedidos temporalmente cercanos— y otorgándole prioridad a los estanques ya sembrados sobre los que no. La expresión matemática se denota de la siguiente manera:

$$\text{Minimizar } Z = \sum_{j \in \psi} (H_j^+ + (\epsilon \cdot U_j \cdot H_j^-)) + \sum_{(i,s,t) \in \Omega} (\alpha_s \cdot \chi_{i,s,t}) \tag{4.1}$$

Donde,

- $\epsilon \in \mathbb{R}^+$: Penalización base por déficit de biomasa.
- $U_j = 1 + \frac{T_{\text{máx}} - T_j}{T_{\text{máx}}}$: Factor de urgencia, le concede prioridad a las entregas más inmediatas sobre las posteriores, siendo $T_{\text{máx}} = \text{máx}_{j \in \psi}\, T_j$ el instante de entrega más lejano entre los pedidos vigentes.
- $\alpha_s = \begin{cases} \alpha_1 \in \mathbb{R}^+, & \text{Si } s \geq 0. \\ \alpha_2 \in \mathbb{R}^-, & \text{Si } s = -1. \end{cases}$: Prioridad de recursos, permite asignarles pedidos a los estanques ya sembrados sobre los vacíos.

4.3.1.5 Restricciones del sistema. Para respaldar la viabilidad técnica y biológica de los planes generados, el modelo se sujeta a las siguientes restricciones:

a) Cumplimiento de la demanda: Asegura que la sumatoria de biomasa de todos los estanques asignados a por lo menos un pedido, ajustada por las desviaciones, sea exactamente igual a la demanda exigida.

$$\sum_{i \in \gamma} (\nu_{i,j}) - H_j^+ + H_j^- = \psi_j,\ \forall_j \in \psi \tag{4.2}$$

b) Restricción lógica temporal: Avala que cada estanque tenga a lo sumo un solo evento de siembra y cosecha válido dentro del horizonte de planificación actual.

$$\sum_{(s,t)\in\Omega_i} \chi_{i,s,t} \leq 1, \ \forall_i \in \gamma \tag{4.3}$$

La factibilidad de este planteamiento depende de la implementación de un horizonte deslizante en el motor de simulación (nótese en la Subsección 4.3.2), mediante el cual el sistema se realimenta y actualiza de forma iterativa sus estados, de ahí que no requiera la determinación explícita del optimizador para culminar el ciclo de vida de los estanques.

c) Compromiso físico de la cosecha: Cada estanque aporta su biomasa a los pedidos que vencen en el instante t si y solo si ha sido programado para su cosecha en el mismo instante. Al cosechar, la asignación total del estanque es *igual* a la biomasa proyectada $B_{i,s,t}$ de la ruta activada: se recoge la totalidad del lote, de modo que la fracción que supera la demanda del pedido se contabiliza como exceso (H_j^+) —biomasa que el productor comercializa por canal directo— en lugar de descartarse.

$$\sum_{j\in\ \psi\ |T_j=t} \nu_{i,j} = \sum_{s\ |(s,t)\in\ \Omega_i} (B_{i,s,t} \cdot \chi_{i,s,t}), \begin{cases} \forall_i \in \gamma \\ \forall_t \in \tau \end{cases} \tag{4.4}$$

En caso de que la biomasa de una ruta sea desconocida, el valor comprometido degrada a la capacidad de carga tolerada del estanque, $K_i \cdot (1+\theta)$, donde $K_i = \mu_i \cdot \rho_i$. La coherencia entre ambas cotas está garantizada por la poda previa de rutas descrita en la construcción del conjunto Ω: toda ruta admitida en el modelo satisface $B_{i,s,t} \leq K_i \cdot (1+\theta)$, salvo las cosechas forzadas de lotes heredados.

d) Filtro de calidad comercial: Mediante el uso del método Big-M (M), se asegura que la biomasa entregada a un pedido sea nula si el estanque no cumple con los criterios de calidad precalificados. Matemáticamente, si el estanque i no es seleccionado para cosecha en el

instante T_j o no satisface el rango de peso exigido, el lado derecho de la inecuación se anula, forzando a $\nu_{i,j}$ ser cero. En caso contrario, M libera la variable para que el modelo determine la cantidad óptima de biomasa a extraer según la biomasa proyectada de la ruta ($B_{i,s,t}$) y el requerimiento del pedido (ψ_j).

$$\nu_{i,j} \leq M \cdot \sum_{s|(s,T_j)\in\Omega_i} \delta_{i,j,s,T_j} \cdot \chi_{i,s,T_j}, \begin{cases} \forall_i \in \gamma \\ \forall_j \in \psi \end{cases} \tag{4.5}$$

Siendo,

$$\delta_{i,j,s,t} \in \{0,1\}, \begin{cases} 1, & \text{Si s } \neq -1 \ \wedge \ \omega_{i,s,t} \in [\omega_j^{\text{mín}}, \omega_j^{\text{máx}}] \wedge t = T_j. \\ 1, & \text{Si } s = -1 \ \wedge (j \in J_i \ \vee \ \omega_{i,s,t} \in [\omega_j^{\text{mín}}, \omega_j^{\text{máx}}]) \wedge t = T_j. \\ 0, & \text{Alternativamente.} \end{cases} \tag{4.6}$$

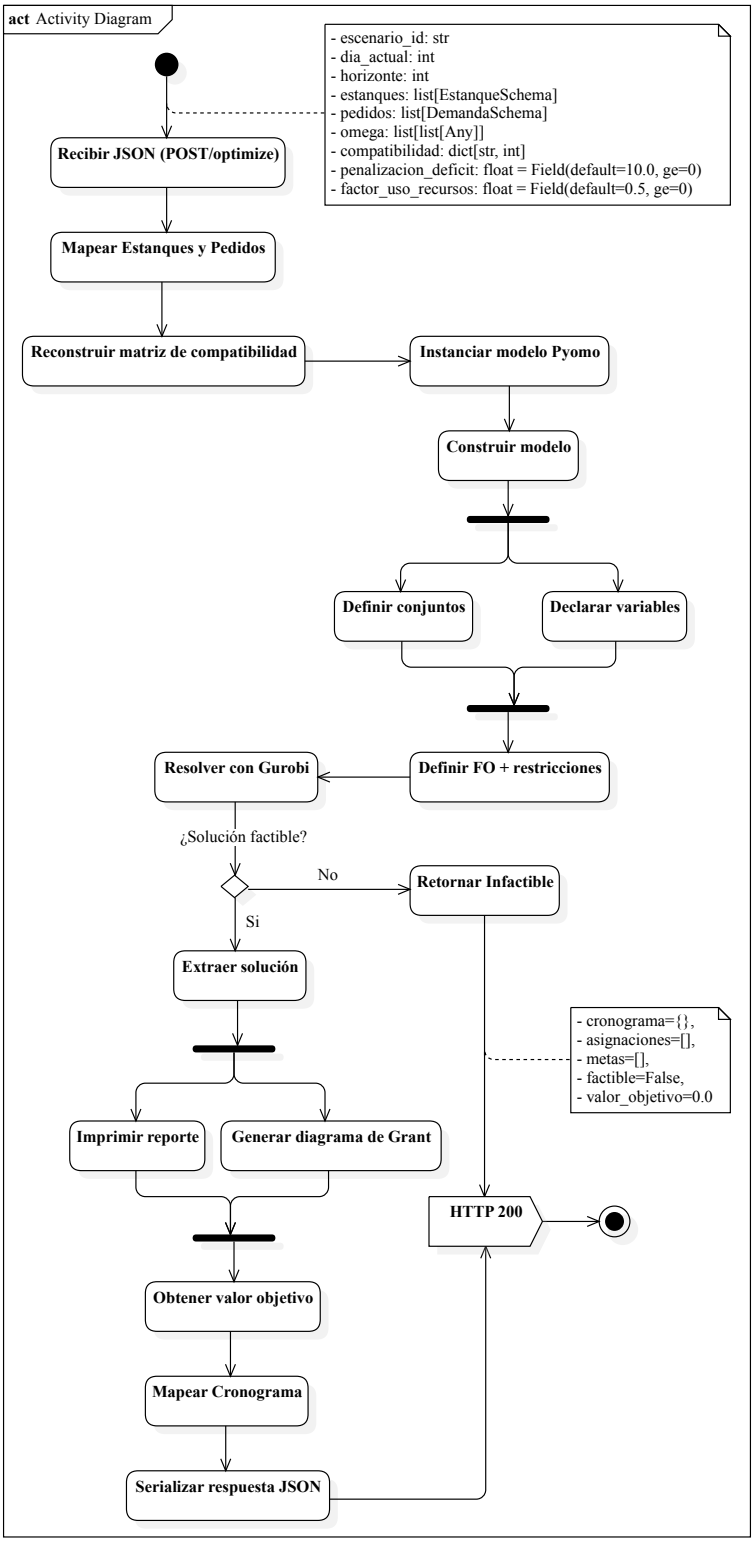
La implementación del anterior modelo supera la mera formulación algebraica para convertirse en un componente ejecutable (detállese en la Sección 4.4). Por ende, el proceso de interacción de este subsistema se detalla en el Diagrama de Actividades (ver Figura 7), el cual describe el ciclo de vida de una solución de planificación, que va desde la ingesta de parámetros biológicos y contractuales, hasta la extracción de una solución factible.

4.3.2 Reingeniería del Módulo de Simulación Multi-Estanque

Definido el problema de optimización, el siguiente paso consistió en desarrollar el motor de simulación encargado de representar la evolución biológica de los estanques y suministrar la información requerida por el optimizador. En consonancia con esta decisión, la revisión detallada de la precedente arquitectura —utilizada para el control y simulación de un único estanque en el proyecto de Rojas Prada (2025)— evidenció que no ofrecía la escalabilidad ni adaptabilidad necesarias para

Figura 7

Diagrama de actividades para el subsistema de optimización



operar como un módulo centralizado. No obstante, se rescataron los algoritmos fundamentales que regían el comportamiento biológico aislado, factorizándolos para permitir su ejecución autónoma y concurrente. Esta transformación arquitectónica hizo posible que el subsistema dejara de ser un simulador descriptivo de un solo agente para convertirse en un motor de ejecución capaz de orquestar la dinámica de una asociación completa, donde cada estanque opera como una instancia independiente dentro de un ecosistema mayor.

Con base en lo anterior, el motor se puso en marcha al dividir el dominio lógico por los *Bounded Contexts* o contextos delimitados, dentro de los cuales se definieron las Raíces de Agregación (*Aggregate Root*'s) que actúan como fronteras transaccionales estrictas. Para visualizar en conjunto las relaciones entre cada una de estas delimitaciones lógicas, en el Diagrama de Clases del Dominio (Figura 8) se representan de forma precisa sus entidades; a continuación se describe cada una de sus agrupaciones:

- Módulo comercial: En este conjunto se materializan los requisitos correspondientes al proceso de comercialización de las ofertas producidas, desde la creación de la identidad del cliente, hasta la generación de la demanda. Adicionalmente, para este servicio la demanda tiene dos formas de representación posible, brindando la extensión futura de diferentes funcionalidades o restricciones para ambos (entiéndase mejor en la Sección 6.1):
    - Pedido: Son instancias independientes y puntuales en el tiempo de una demanda, se definen por su fecha de expedición, la biomasa total, el intervalo de pesos promedio aceptable de cada producto, el cliente que genera la demanda y su estado actual.
    - Contrato: Se definen simplemente como un conjunto determinista de pedidos con fechas de expedición secuenciales de intervalos constantes.
- Módulo de infraestructura: En este se concentra la lógica asociada al ciclo operativo de los elementos espaciales, tal que por el paradigma de la programación orientada a objetos, se exige su abstracción para profundizar en una jerarquía: un Productor es la entidad legal que puede poseer múltiples granjas, y cada Granja es el contenedor físico de los estanques. Esta

estructura permite agregar nuevos productores a la asociación sin modificar la lógica del motor.

- Módulo operativo: Su responsabilidad es la de encapsular todos los parámetros biológicos para la trazabilidad del crecimiento y el comportamiento de cada uno de los estanques. Asimismo, para abstraer fácilmente esos datos, se define el Estanque como el contenedor de un Lote de Peces; así, un estanque está vacío cuando no tiene dicho lote.

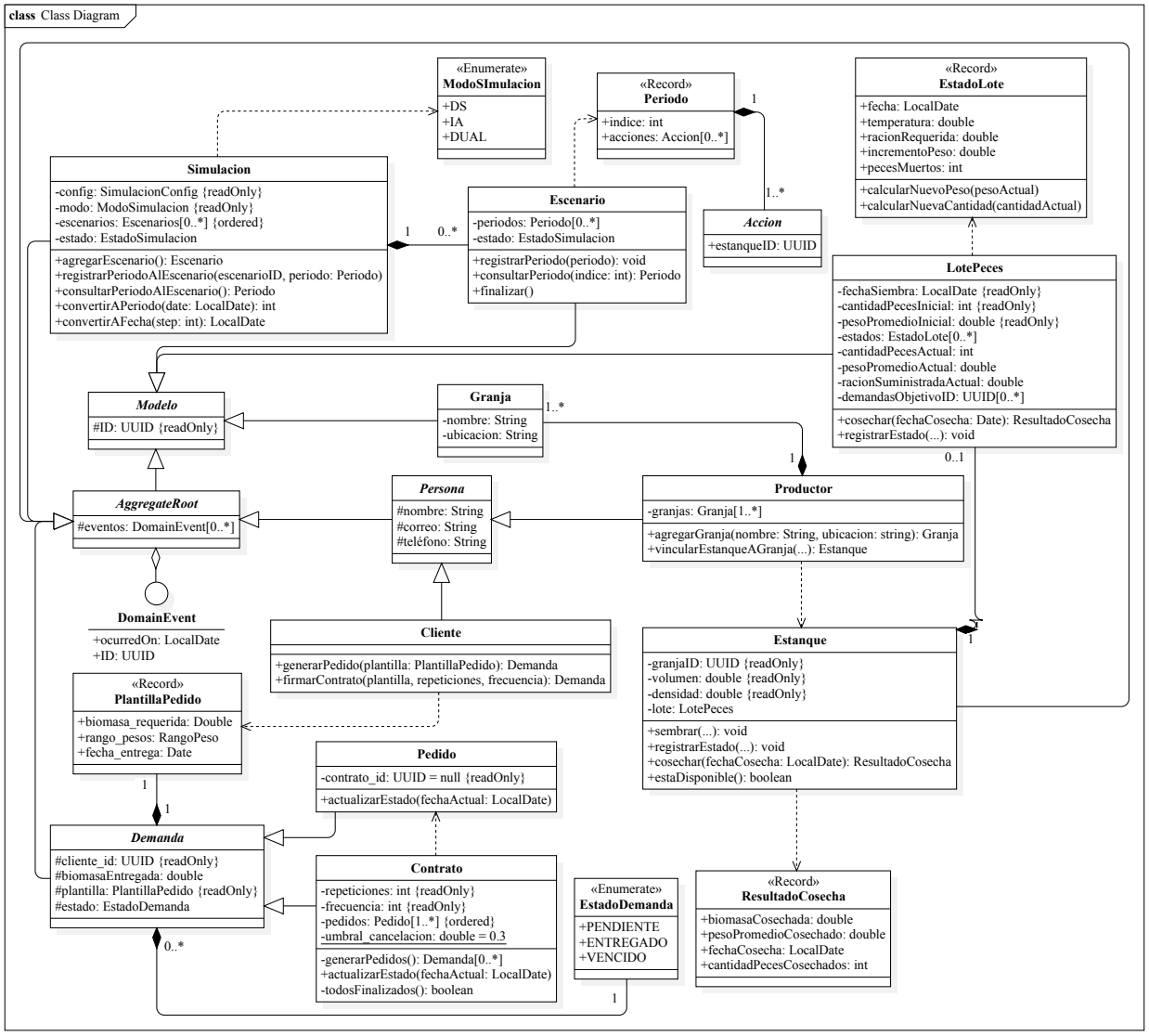
- Módulo del tiempo: Aquí se encapsula el reloj del sistema, es decir, su función es sincronizar el tiempo virtual de todos los estanques y coordinar la persistencia de datos para la creación de diferentes escenarios.

La interacción dinámica entre estos agregados del dominio y las capas de infraestructura durante la ejecución de un ciclo productivo se documenta mediante dos diagramas de secuencia complementarios, cada uno enfocado en un nivel de abstracción distinto: el Diagrama de Secuencia General (Figura 9) describe el flujo de control que atraviesa las capas arquitectónicas del subsistema durante la ejecución del caso de uso Ejecutar Horizonte Deslizante; a su vez, con la intención de mantener cohesión en el nivel de abstracción de las capas arquitectónicas, el paso `avanzarHorizonte` se presenta como un mensaje sin respuesta explícita, delegando su descomposición interna al Diagrama de Secuencia Específico (Figura 11).

Paralelamente, para proteger la consistencia en la generación de escenarios masivos y la posterior ingesta de datos por parte de Gurobi, el componente de persistencia fue sometido a una reestructuración profunda. Si bien en el trabajo previo del grupo se utilizó satisfactoriamente MongoDB por su flexibilidad en el manejo de documentos NoSQL, los requisitos analíticos de este proyecto exigieron un rigor superior en la integridad referencial. La decisión de migrar hacia un motor de base de datos SQL (relacional) se fundamenta en la capacidad de estas estructuras de ejecutar consultas analíticas complejas y agregaciones masivas sobre el estado histórico de la asociación. El motor relacional ofrece un rendimiento superior para aplanar la complejidad del dominio y suministrar los vectores de parámetros limpios que requiere el solucionador MILP. El

Figura 8

Diagrama de clases del dominio

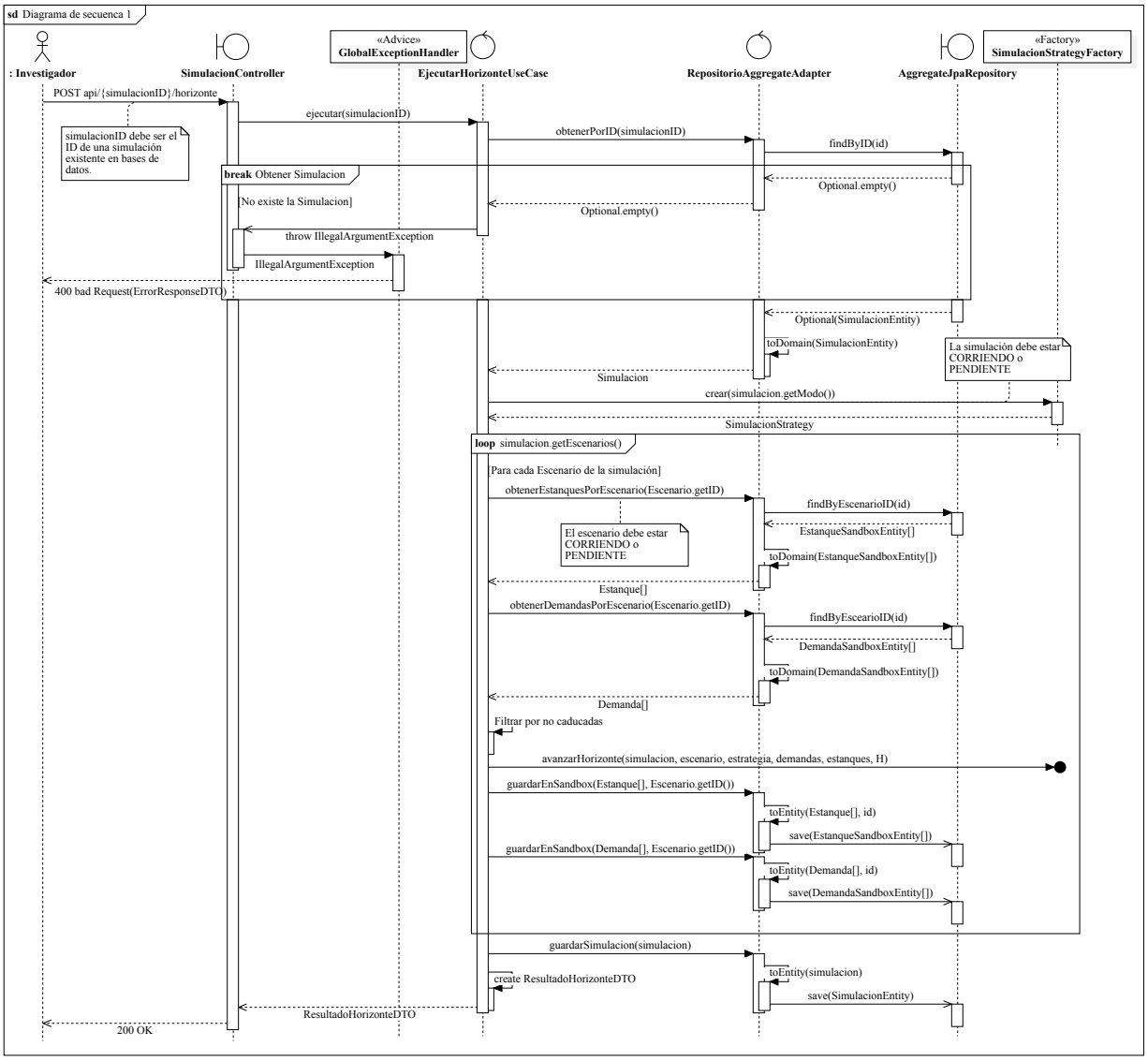


modelo físico resultante, diseñado para mitigar el desajuste objeto-relacional (*Object-Relational Impedance Mismatch*), se detalla en el Diagrama Entidad-Relación (Figura 10), el cual descompone las jerarquías de los agregados en un esquema normalizado capaz de operar como fuente de verdad.

Complementariamente, a lo largo del código se identificaron patrones de diseño tanto estructurales como de comportamiento, junto con técnicas arquitectónicas específicas que responden a los requisitos de alta variabilidad del dominio piscícola. Seguidamente, se desarrolla una descripción de cada uno de estos recursos, abordando tanto su implementación como las consideraciones

Figura 9

Diagrama de secuencia general

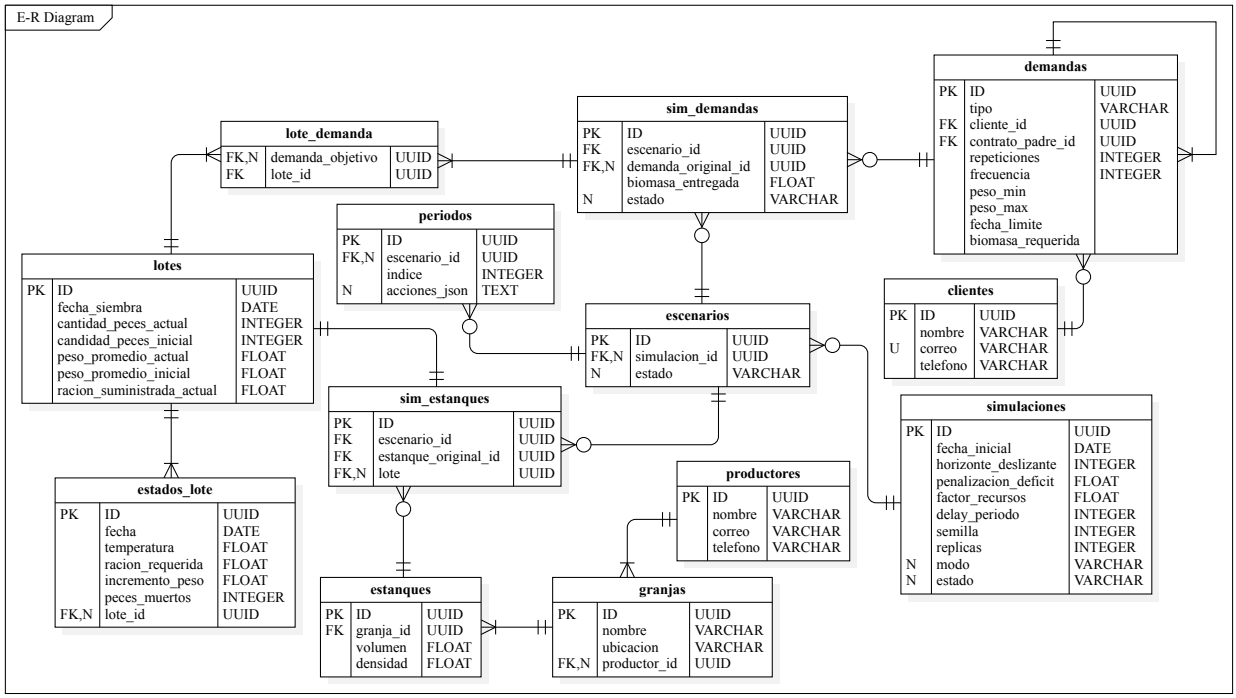


de ingeniería que fundamentan su inclusión en la solución propuesta.

4.3.2.1 Aislamiento de escenarios (*Sandbox Pattern*). Durante una simulación, el sistema necesita ejecutar operaciones sobre los estanques y las demandas (tales como proyecciones de siembra y cosecha) sin afectar el estado real o físico de la producción actual. A través de este mecanismo, el motor instancia copias de trabajo efímeras de los *Aggregate Root*'s que evolucionan

Figura 10

Diagrama entidad-relación del dominio

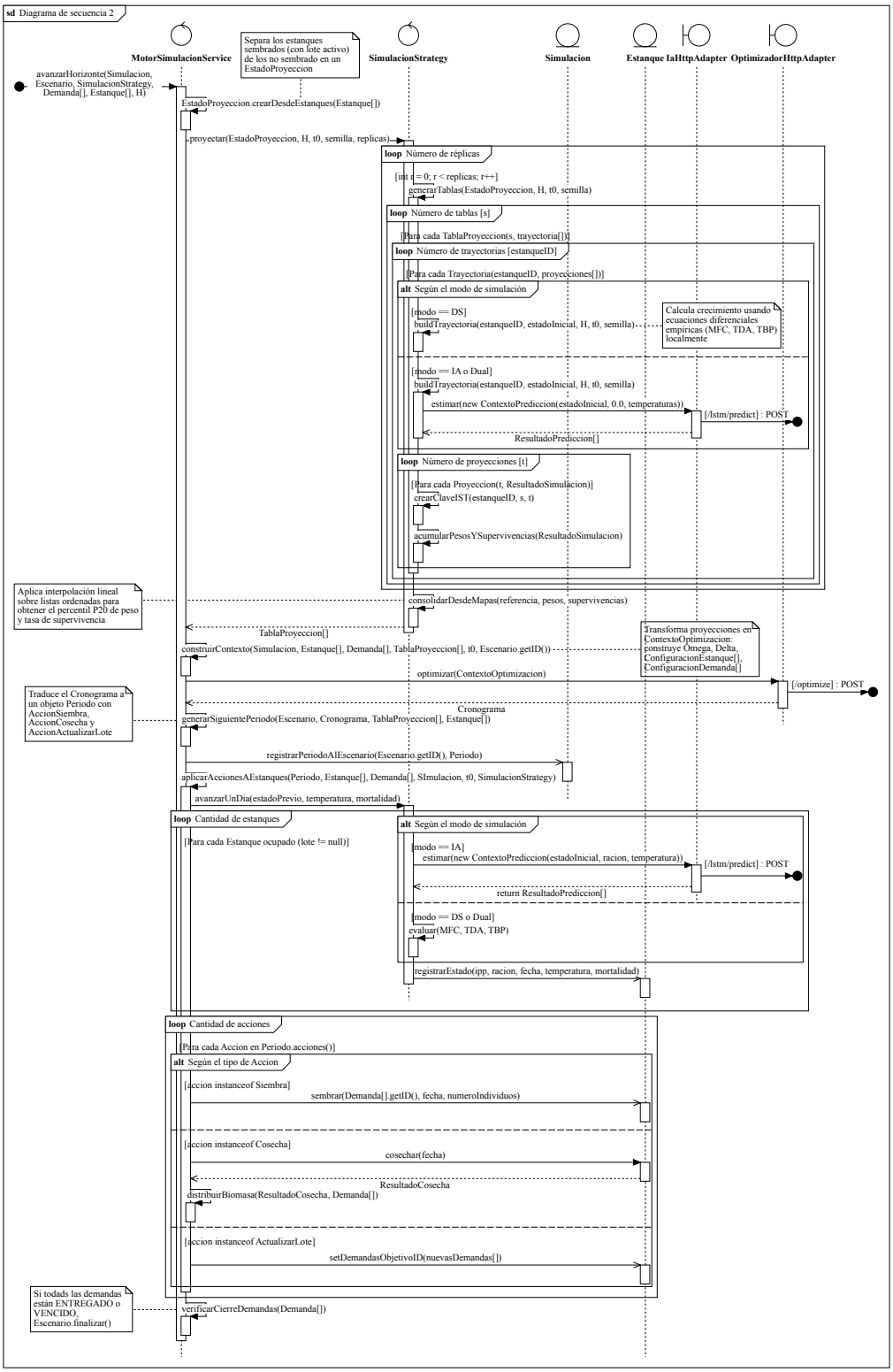


de forma aislada para cada escenario simulado. De ese modo, sin el *Sandbox*, el escenario A y el escenario B competirían por el estado del estanque E1, y la simulación de uno contaminaría al otro. El sandbox convierte cada escenario en un universo cerrado que puede evolucionar de forma independiente.

4.3.2.2 Strategy Pattern. El motor de simulación soporta múltiples mecanismos de proyección biológica y matemática, cuya lógica interna difiere drásticamente en su origen computacional y su naturaleza algorítmica. Sin un mecanismo de abstracción adecuado, el flujo principal de simulación estaría plagado de condicionales sobre el modo activo, lo que haría que cualquier adición o modificación de un algoritmo de crecimiento requiriese intervenir en el núcleo del mismo. El *Strategy Pattern* resuelve esto al encapsular la familia completa de algoritmos de proyección bajo la interfaz común `SimulacionStrategy`, permitiendo modificar dicho comportamiento predictivo sin alterar la estructura de orquestación del sistema.

Figura 11

Diagrama de secuencia específico



La interfaz define el contrato que toda estrategia debe cumplir: proyectar un horizonte temporal de tablas de trayectorias (`proyectar`), avanzar el estado biológico de un estanque en un único paso diario (`avanzarUnDia`), y generar variables estocásticas reproducibles de temperatura y mortalidad (`temperatura`, `mortalidad`, `mezclarSemilla`). El resto del sistema —el `MotorSimulacionService`, el proceso de Monte Carlo— opera exclusivamente contra esta interfaz, sin conocimiento de la estrategia concreta en ejecución.

Se definen tres estrategias concretas, seleccionadas en tiempo de ejecución mediante `SimulacionStrategyFactory` en función del modo configurado por el usuario:

- Modo solo DS: Calcula el crecimiento biológico localmente mediante un sistema de ecuaciones diferenciales empíricas parametrizadas con tres tablas de interpolación: la fracción de mantenimiento del alimento consumido (*MFC*), la tasa diaria de alimentación (*TDA*) y la tasa de biomasa por temperatura (*TBP*). A partir de estos coeficientes, la estrategia calcula en cascada la ración requerida, la ingesta de mantenimiento, la ingesta destinada a producción y, finalmente, el incremento neto de peso del pez, sin depender de ningún servicio externo. Este modo es el más reproducible y el de menor latencia, ya que toda la computación ocurre dentro de la JVM.

- Modo solo IA: Delega las proyecciones al subsistema de predicción de Python (detállese en la Subsección 4.3.3). Para construir una trayectoria de H días, la estrategia genera previamente la secuencia completa de temperaturas estocásticas para ese horizonte y las envía en una única solicitud al modelo LSTM, el cual devuelve la lista de pares (Δpeso, ración) correspondientes. Este enfoque minimiza la latencia de red al evitar H solicitudes individuales, colapsando la proyección en una sola llamada por trayectoria.

- Modo híbrido: Combina ambos subsistemas con una separación explícita de responsabilidades: utiliza el servicio de la LSTM para construir las tablas de proyección del horizonte (es decir, para planificar), mientras que delega en el modelo de DS el avance diario real durante la ejecución del escenario (es decir, para simular). Esta asimetría es deliberada: el modelo

neuronal captura las correlaciones no lineales entre variables ambientales y biológicas de manera más expresiva para la planificación a mediano plazo, mientras que las ecuaciones diferenciales ofrecen un mayor control y auditabilidad para el registro histórico paso a paso.

4.3.2.3 *Template Method Pattern* y simulación Monte Carlo.

El modelado de la incertidumbre biológica en la piscicultura exige un enfoque estocástico que trascienda la proyección determinista de un único escenario futuro. Aunque el sistema biológico está influenciado por una amplia variedad de factores estocásticos, el modelo propuesto incorpora únicamente:

- Tasa de mortalidad: Se expresa como una variable aleatoria uniforme en el intervalo [0, 0,002], lo que produce una mortalidad acumulada por ciclo productivo consistente con el rango del 20 %, documentado para la especie bajo condiciones de manejo técnico adecuado (Merino Archila et al., 2006). La supervivencia se calcula de forma compuesta día a día mediante:

$$S_t = S_{t-1} \cdot (1 - m_t),\ m_t \equiv \text{tasa de mortalidad del día } t. \tag{4.7}$$

- Temperatura del agua: Se plantea con una distribución gaussiana centrada en 24 °C, acotada entre 16 y 32 °C, rango que corresponde a las condiciones térmicas generales documentadas en Colombia (Merino Archila et al., 2006).

Sobre estos factores se implementó una simulación de Monte Carlo que ejecuta N réplicas probabilísticas independientes sobre el mismo horizonte temporal, generando una distribución empírica de posibles trayectorias futuras de temperatura y supervivencia. Cada réplica opera con una semilla derivada determinísticamente de la semilla global de la simulación mediante una función de mezcla de hash multiplicativo, lo que garantiza que los resultados sean reproducibles ante la misma configuración inicial.

La estructura algorítmica de este proceso se organiza mediante el patrón *Template Method*. La clase abstracta `BaseSimulacionStrategy` define el esqueleto invariable del algoritmo en su método `proyectar`: itera sobre las N réplicas, invoca en cada una el método `generarTablas` para

construir la estructura completa de trayectorias, acumula los valores de peso y supervivencia en mapas indexados por la clave compuesta (estanqueID, s, t), y al finalizar todas las réplicas consolida la distribución resultante al percentil de decisión P_{20}. El paso variable del algoritmo —la construcción de la trayectoria biológica diaria para un estanque específico, método `buildTrayectoria`— es declarado abstracto en la clase base y delegado a la subclase activa del patrón *Strategy*. De esta forma, la mecánica del muestreo estadístico y la consolidación de percentiles quedan completamente aisladas de la lógica predictiva de crecimiento; las subclases solo implementan cómo calcular una trayectoria.

Una consideración de ingeniería relevante en esta implementación es la gestión eficiente de la memoria. Dado que el número de réplicas puede ser elevado y cada una genera tablas de trayectorias proporcionales al producto réplicas $\times H^2 \times$ estanques, mantener todas las réplicas en memoria simultáneamente resulta prohibitivo. Para mitigar este problema, la implementación acumula únicamente los valores escalares de peso y supervivencia en listas dentro de dos mapas globales (uno por variable), descartando cada réplica completa tras su procesamiento y dejándola elegible para el recolector de basura. El pico de memoria se reduce así de $O(R \times H^2 \times E)$ a $O(H^2 \times E)$, donde R es el número de réplicas y E el número de estanques.

La consolidación final aplica interpolación lineal sobre las listas ordenadas de valores para obtener el percentil P_{20} tanto del peso estimado como de la probabilidad de supervivencia. La elección de este cuantil como cota de decisión responde a un criterio de gestión de riesgo: si el optimizador MILP planificara sus cosechas sobre el escenario esperado (mediana o media), el plan sería matemáticamente óptimo bajo el supuesto de que la producción real coincide con la proyección central, lo que en la práctica ocurre solo en aproximadamente la mitad de los ciclos productivos. Al utilizar el P_{20}, el sistema planifica sobre un valor que la producción real iguala o supera en aproximadamente cuatro de cada cinco realizaciones posibles, lo que se traduce en compromisos de entrega conservadores pero con una alta probabilidad de cumplimiento efectivo, reduciendo el riesgo de déficit frente a los clientes de la asociación.

4.3.2.4 ***State Machine Pattern.*** Este patrón gobierna de forma estricta las transiciones de estado lógicas por las que transcurren las entidades `Simulación` y `Escenario`, cuyos estados válidos se restringen a: PENDIENTE, CORRIENDO, PAUSADA, CANCELADA y FINALIZADA. Las reglas de transición están codificadas directamente en los métodos del agregado, rechazando de forma atómica cualquier mutación inválida (p. ej., transitar una simulación de CANCELADA a CORRIENDO). Asimismo, se implementó una transición automática: cuando la totalidad de los `Escenarios` asociados a una `Simulación` alcanzan el estado FINALIZADA, la entidad padre muta a ese mismo estado de forma síncrona.

Por otra parte, la máquina de estados también gobierna el ciclo productivo de cada lote con cuatro estados definidos en función del peso promedio de los peces: `CRIANZA` (peso < 20 g), `PRE_ENGORDE` (20–150 g), `ENGORDE` (≥ 150 g) y `COSECHADO`. Las transiciones son unidireccionales y determinadas exclusivamente por el peso calculado en cada iteración diaria; ninguna transición inversa ni salto de estado es posible por diseño de la implementación.

Así pues, centralizar esta lógica previene la dispersión de condicionales booleanos complejos a lo largo de la capa de aplicación, blindando el sistema contra estados huérfanos o inconsistentes.

4.3.2.5 ***Mapper Pattern.*** Con la finalidad de acatar el aislamiento de la capa de dominio exigido por DDD y evitar el acoplamiento con la infraestructura, se implementaron dos familias de mappers especializados utilizando la librería *MapStruct*. Estas clases resuelven la conversión estructural de objetos en tiempo de compilación:

- Mappers de Respuesta: Transforman los *Aggregate Root*′s del dominio en Objetos de Transferencia de Datos (*DTO*′s) sanitizados para su exposición segura a través de la API REST.
- Mappers de Persistencia: Traducen las entidades ricas del dominio a entidades declarativas de JPA para su almacenamiento en la base de datos relacional, resolviendo el desajuste de impedancia entre el modelo de objetos y el esquema tabular.

En síntesis, la orquestación de estos patrones de diseño confiere al motor de simulación

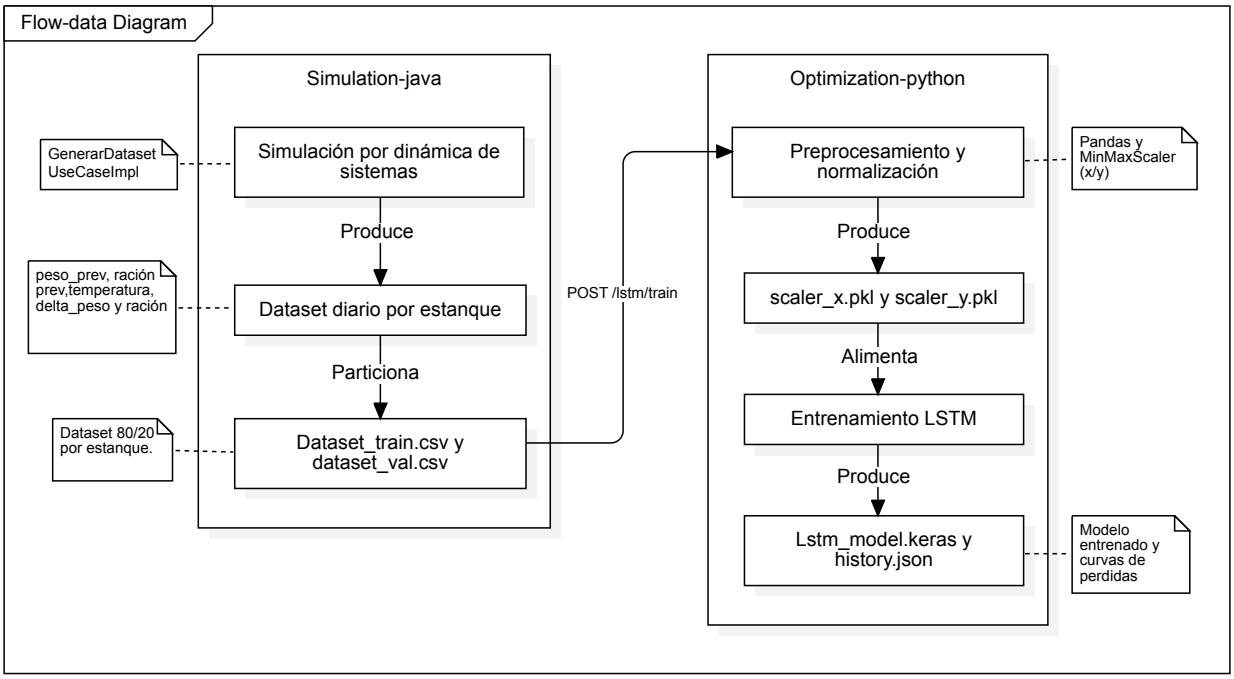
la capacidad de operar como un generador de datos sintéticos estocásticos de alta fidelidad. Al aislar el comportamiento biológico fluctuante de las restricciones comerciales y estructurales, el sistema contribuye a que los escenarios evaluados representen fielmente la incertidumbre operativa, proveyendo al módulo de optimización matemática un conjunto de parámetros de entrada validados, viables y consistentes con la realidad física de la asociación piscícola.

4.3.3 Calibración y Entrenamiento del Módulo Predictivo

Con el simulador operando como generador de datos sintéticos, el desarrollo continuó con el entrenamiento del módulo predictivo encargado de aproximar la dinámica biológica del sistema. Este subsistema precisa aprender, a partir de las observaciones generadas por el simulador, una representación funcional capaz de anticipar las variables biológicas clave (incremento de peso y ración requerida) que determinan la coyuntura ideal de cosecha de cada estanque. Dado que estas variables evolucionan secuencialmente día a día y dependen del historial reciente del pez, se adoptó una arquitectura de red neuronal recurrente de tipo LSTM, naturalmente adecuada para capturar dependencias temporales de esta naturaleza. Este subsistema retoma como punto de partida el modelo predictivo desarrollado previamente por el grupo SIMON para la gestión de estanques individuales, cuya validez ya había sido demostrada en ese contexto. No obstante, su adaptación a la escala asociativa —donde el pronóstico de cada estanque alimenta las decisiones de un optimizador— exige un cuidado particular con la estabilidad del error, pues el esquema de inferencia autorregresivo realimenta cada predicción como entrada del día siguiente y puede acumular desviaciones a lo largo del horizonte. Por esta razón, la reformulación descrita a continuación no persigue únicamente unificar y simplificar la representación de los datos, sino robustecer el comportamiento predictivo del modelo para contener dicha acumulación antes de que se propague hacia las etapas de optimización. Para llevar a cabo el aprendizaje, se calibró desde la generación y estructuración del *dataset* sintético hasta la configuración final del entrenamiento de la red, proceso que se ilustra de principio a fin en la Figura 12.

Figura 12

Flujo de datos del módulo predictivo, desde la generación sintética hasta el entrenamiento



4.3.3.1 Estructura del dataset. En contraste con los dos *datasets* reportados en el proyecto previo, en la solución propuesta se unifican a una sola estructura que alimenta simultáneamente la predicción del incremento de peso y de la ración, eliminando así la necesidad de mantener *datasets* agrupados por las variables objetivo. Más aún, en la nueva estructura se incorpora la columna *racion_prev* como una variable de entrada para soportar el esquema de inferencia autorregresiva, y se retira la variable *peso_max* dado que se define como un límite biológico dependiente de la especie del pez considerada. La estructura unificada resultante del *dataset* se define en la Tabla 5.

Los datos de entrenamiento se generan sintéticamente a partir de la homologación del simulador de DS desarrollado anteriormente. Asimismo, para garantizar reproducibilidad, el muestreo de las variables estocásticas se inicializa a partir de una semilla base fija. A partir de esta configuración, cada simulación generada avanza día a día calculando el incremento de peso y la ración requerida en función del peso del pez y la temperatura, hasta alcanzar un horizonte máximo de 300 días o hasta que el crecimiento diario sea insignificante; este último criterio se define cuando la

Tabla 5
Estructura del dataset de entrenamiento

Variable	Descripción	Unidad / tipo
estanque_id	Identificador único del estanque	`UUID`
t	Día dentro del ciclo del estanque	$\mathbb{Z}^+$
peso_prev	Peso del pez en el día anterior	g
racion_prev	Ración suministrada el día anterior	g/d
temperatura	Temperatura del agua	°C
delta_peso	Incremento de peso del día	g
racion	Ración requerida del día	g/d

tasa de crecimiento relativo es inferior al 0,1 % durante un periodo consecutivo de 5 días.

Para exponer el modelo a trayectorias representativas de todo el ciclo productivo, el peso inicial de cada estanque se muestreó mediante una distribución log-uniforme en el intervalo biológicamente válido para la especie. En el caso de la tilapia (*Oreochromis spp.*), especie de referencia adoptada en este trabajo, el peso mínimo de siembra corresponde a 10 g al inicio de la fase de alevinaje, mientras que el peso máximo biológico documentado es de 3000 g, valor que representa el límite superior del potencial de crecimiento individual de la especie bajo condiciones de cultivo controlado (Merino Archila et al., 2006). Al generar trayectorias que abarcan este intervalo completo, el modelo LSTM puede aproximar el comportamiento del crecimiento en todas sus etapas, incluida la desaceleración observada cuando el pez se aproxima a su potencial biológico máximo. Así pues, el muestreo log-uniforme en este intervalo proporciona una representación más equilibrada de las distintas escalas de peso, evitando que el conjunto de entrenamiento quede sesgado hacia los pesos grandes, que dominarían bajo una distribución lineal uniforme. Bajo esta formulación, el uso del sistema con otras especies requeriría la recalibración de dichas constantes y el reentrenamiento del modelo LSTM con un dataset generado bajo los parámetros biológicos de la especie objetivo, lo cual constituye una extensión natural del trabajo descrita en la Sección 6.1.

Una vez generado el *dataset*, se aplicaron dos etapas de preprocesamiento antes de alimentar al modelo:

a) División del entrenamiento / validación: El *dataset* se dividió en un 80 % para entrenamiento

y un 20 % para validación, realizando la separación a nivel de estanque completo para evitar que el modelo acceda a información futura de una misma secuencia durante la evaluación.

b) Normalización: Las variables de entrada y salida presentan rangos de magnitud muy distintos, por lo que para evitar que las variables de mayor magnitud dominen el cálculo del gradiente penalizando implícitamente a las de menor escala, se aplicó una normalización llamada *min-máx* que escala todos los valores al rango $[0, 1]$, ajustando los parámetros del escalador exclusivamente sobre el conjunto de entrenamiento. Los escaladores resultantes se persisten en los archivos con extensión `.pkl` para facilitar la misma transformación durante la inferencia en producción.

4.3.3.2 Arquitectura del modelo. La decisión de unificar los modelos se fundamenta en que ambas salidas comparten las mismas variables de entrada (*peso_prev*, *racion_prev*, *temperatura*). Por tanto, la arquitectura está compuesta por cuatro capas, las cuales se pueden observar en la Figura 13.

- Capa de enmascaramiento (*Masking*): Permite procesar secuencias de longitud variable ignorando los pasos de tiempo marcados con *mask_value* $= -1{,}0$, de modo que los valores de relleno no afectan el aprendizaje.

- Primera capa recurrente — LSTM(64): Extrae patrones temporales de bajo nivel, es decir, correlaciones inmediatas entre el peso del pez, la ración previa y la temperatura. Sus 64 neuronas ocultas le confieren capacidad para capturar las no linealidades de las tablas biológicas *tda* (para el peso) y *tbp* (para la temperatura).

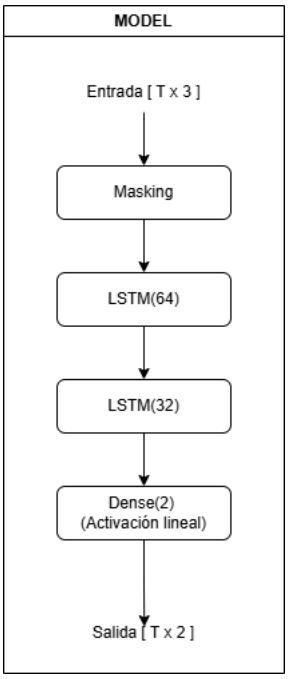
- Segunda capa recurrente — LSTM(32): Construye representaciones de nivel superior sobre los patrones ya extraídos, aprendiendo relaciones más abstractas como la desaceleración progresiva del crecimiento conforme el pez se aproxima a su peso máximo.

- Capa de salida — TimeDistributed Dense(2): Aplica una capa densa de dos neuronas a cada paso de tiempo de forma independiente, generando una salida de datos de la forma

(*delta_peso*, *racion*) por día. Se utiliza la activación lineal dado que ambas variables objetivo son continuas y no están acotadas superiormente.

Figura 13

Arquitectura de red neuronal LSTM



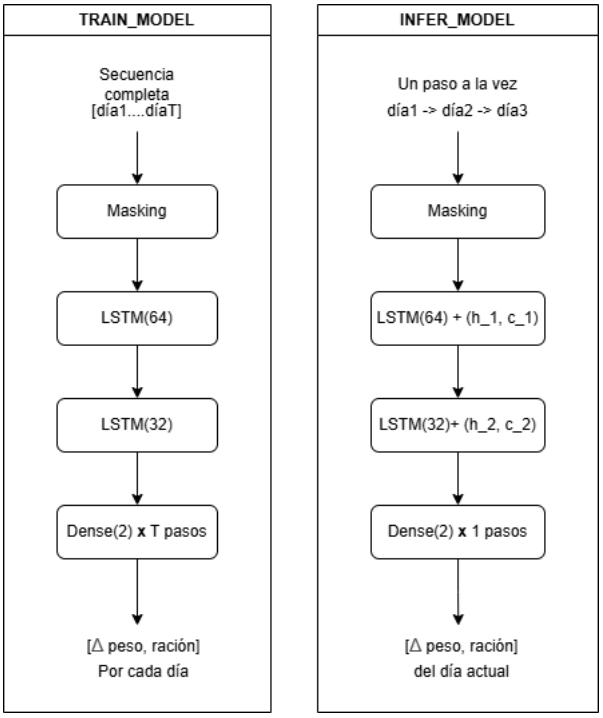
La arquitectura se implementa mediante dos modelos que comparten exactamente los mismos pesos internos, pero con interfaces distintas según el contexto de uso. Por un lado, el `train_model` recibe la secuencia completa del estanque de una sola vez y se utiliza durante el entrenamiento, permitiendo que el gradiente fluya a través de toda la historia temporal de manera eficiente. Por otro lado, el `infer_model` recibe un único paso de tiempo junto con los estados ocultos de la red y los devuelve actualizados tras cada predicción, habilitando la inferencia autorregresiva en producción donde las salidas de un día se convierten en las entradas del día siguiente. Dado que ambos comparten los mismos pesos, entrenar el primero actualiza automáticamente al segundo.

4.3.3.3 Configuración del entrenamiento. El entrenamiento se configuró con los hiperparámetros descritos en la Tabla 6. Asimismo, en la Figura 14 se visualizan ambos modelos

manteniendo la misma estructura mencionada anteriormente.

Figura 14

Arquitectura de red neuronal LSTM con dos modelos paralelos que comparten los mismos pesos



4.4 Integración de Componentes y Comunicación (API)

Los tres subsistemas que conforman el artefacto operan como procesos independientes que se comunican a través de interfaces contractuales bien definidas. Esta independencia de despliegue es una consecuencia directa de la arquitectura hexagonal descrita anteriormente: ningún subsistema conoce los detalles de implementación de los demás; únicamente conoce el contrato de la interfaz que expone cada uno. La presente sección describe cómo se materializa esa integración a nivel de red, qué contratos de datos se intercambian en cada dirección, y cuál es la topología de despliegue del sistema completo.

Tabla 6
Hiperparámetros del entrenamiento

Parámetro	Valor	Justificación
Optimizador	Adam	Adapta el learning rate por parámetro; estándar para LSTM.
Learning rate	0,001	Valor por defecto; se mantuvo por estabilidad observada.
Función de pérdida	MSE	Penaliza proporcionalmente más los errores grandes.
Tamaño de batch	1 secuencia	Cada estanque tiene longitud distinta.
Épocas máximas	100	Límite superior; rara vez se alcanza con early stopping.
Paciencia early stopping	15 *epoch*′s	Permite recuperación de mínimos locales transitorios.
Semilla	42	Fija la inicialización aleatoria.

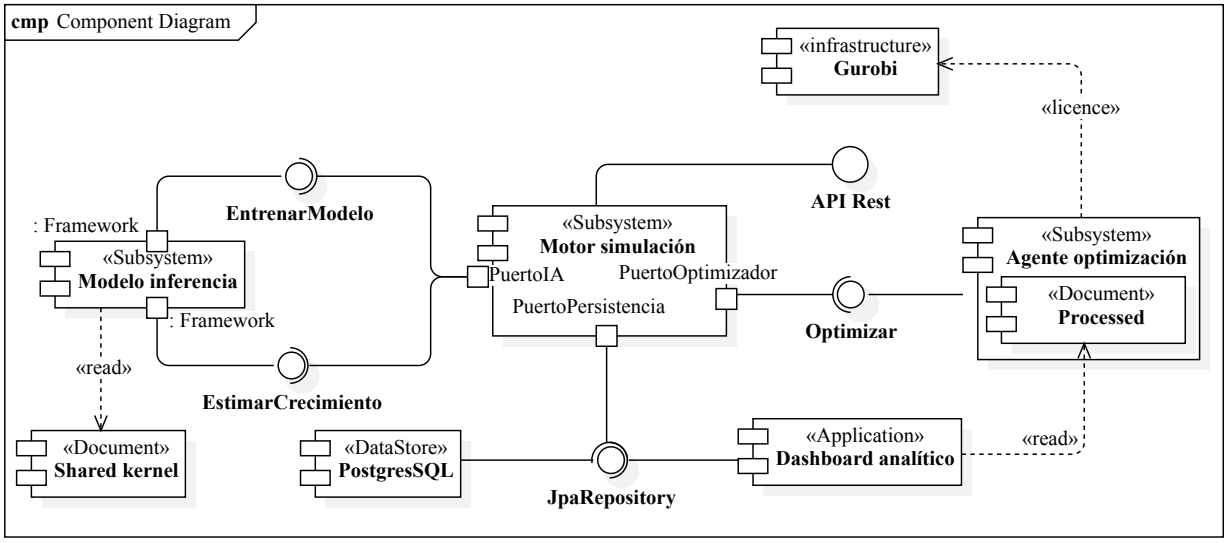
4.4.1 Topología General del Sistema

La arquitectura de integración sigue un patrón de orquestación centralizada: el motor de simulación actúa como el cliente activo que inicia todas las llamadas entre subsistemas, mientras que FastAPI actúa exclusivamente como servidor de cómputo. PostgreSQL es el almacén singular de estado persistente y es accedido exclusivamente por Spring Boot mediante el ORM JPA/Hibernate. El subsistema Python nunca accede directamente a la base de datos; recibe los parámetros de entrada por HTTP, realiza el cómputo solicitado y devuelve el resultado. Streamlit, por su parte, actúa como capa de visualización: en coherencia con la restricción anterior, no accede a la base de datos, sino que consulta los resultados de las simulaciones ejecutadas a través de la misma API REST del motor Java y los presenta en un panel interactivo. Este panel consolida, sobre el plan ya ejecutado, los indicadores de cumplimiento de la demanda y los indicadores económicos —utilidad neta, retorno sobre la inversión, biomasa producida y utilización de los estanques—, tanto a nivel agregado de la asociación como en el detalle por pedido y por estanque, y constituye la superficie sobre la que se apoya el análisis de viabilidad del Capítulo 5. La Figura 15 presenta el Diagrama de Componentes del sistema que expone esta topología y los flujos de comunicación entre dichos procesos.

Todos los canales de comunicación entre Spring Boot y FastAPI son peticiones HTTP

Figura 15

Diagrama de componentes del artefacto



síncronas sobre la misma red interna del contenedor Docker, lo que garantiza latencia mínima entre procesos y elimina la necesidad de un broker de mensajes (para más detalles, véase el Apéndice B).

4.4.2 Protocolo de Comunicación con el Estimador

El subsistema de inteligencia artificial expone dos operaciones bajo el prefijo `/lstm`:

- *POST `/lstm/predict`:* Canal de inferencia en tiempo real. Spring Boot lo invoca durante cada avance del horizonte deslizante, una vez por estanque y por réplica de Monte Carlo. El cuerpo de la solicitud (Figura 16) transporta el estado actual del pez y la secuencia de H temperaturas proyectadas para el horizonte. La respuesta contiene una lista de H pares (Δpeso, ración), uno por día proyectado.

- *POST `/lstm/train`:* Canal de reentrenamiento asíncrono. Spring Boot lo invoca desde `EntrenamientoHttpAdapter` después de escribir los archivos `.csv` en el directorio compartido `/shared-kernel/`. La respuesta es inmediata con código HTTP 202 (*Accepted*); el entrenamiento se ejecuta en un hilo de fondo en Python, protegido por un mutex

(`threading.Lock`) para rechazar solicitudes concurrentes de entrenamiento con código 409 (*Conflict*). Una vez completado el proceso, el modelo actualizado queda disponible en disco para inferencias posteriores.

Figura 16

Estructura JSON para el intercambio de parámetros biológicos (simulación - LSTM)

```
{
   "peso_inicial": double,
   "racion_inicial": double,
   "temperaturas": [
      double
   ]
}
```

Nota. El campo `temperaturas` contiene exactamente H valores en grados Celsius, uno por día del horizonte proyectado. El campo `racion_inicial` corresponde a la ración suministrada el día anterior; se inicializa en cero si no se dispone de historial previo.

En el lado de Java, ambas operaciones están encapsuladas en `IaHttpAdapter`, que implementa el puerto de dominio `PuertoIa`. El adaptador configura un `PoolingHttpClientConnectionManager` con un máximo de 100 conexiones abiertas y 20 por ruta, reutilizando conexiones TCP entre solicitudes en lugar de establecer una nueva por cada llamada.

4.4.3 Contrato de Interfaz con el Optimizador

Este subsistema expone una única operación POST `/optimize`. Spring Boot la invoca desde `OptimizadorHttpAdapter` una vez por ciclo del horizonte deslizante, después de que el motor de Monte Carlo ha consolidado las tablas de proyección. El cuerpo de la solicitud representa la serialización del `ContextoOptimizacion` del dominio Java hacia el esquema que Pyomo[3] necesita para construir el modelo MILP. Esta traducción requiere tres transformaciones no triviales:

[3]Pyomo (*Python Optimization Modeling Objects*) es un lenguaje de modelado algebraico de código abierto que formula el problema de optimización en términos matemáticos y lo delega al solver externo, en este caso Gurobi.

a) Los conjuntos de objetos de valor `ClaveIST` y `ClaveIJST` se serializan como listas de listas y como diccionarios con claves de cadena, respectivamente, dado que JSON no admite tuplas ni objetos compuestos como claves de mapa. En el lado Python, `OptimizacionService` reconstruye las tuplas originales a partir de esas cadenas antes de construir los conjuntos de Pyomo.

b) La capacidad máxima recomendada de cada estanque se expresa como *capacidad_kg*, calculada en Java como el producto $\rho \times \mu$ del estanque real.

c) Los parámetros de control del optimizador —penalización por déficit y factor de uso de recursos— se transmiten directamente desde `SimulacionConfig`, lo que permite al usuario calibrar el comportamiento del solucionador sin recompilar el sistema.

La respuesta que Python devuelve (Figura 17) incluye el cronograma de siembras y cosechas por estanque, las asignaciones de biomasa a pedidos y las métricas de cumplimiento de cada demanda.

Si el solucionador Gurobi agota su límite de tiempo de 600 segundos sin encontrar una solución óptima, pero dispone de una solución incumbente, Python la devuelve marcada como factible en lugar de abortar la solicitud. Esta decisión de diseño evita que una instancia MILP de gran escala bloquee indefinidamente el hilo de Spring Boot, el cual tiene configurado un timeout de respuesta de 15 minutos en `OptimizadorHttpAdapter` precisamente para absorber los peores casos de combinatoria.

4.4.4 Directorio Compartido (`shared-kernel`)

El flujo de entrenamiento del modelo LSTM introduce un mecanismo de integración adicional distinto al canal HTTP. Cuando el motor de simulación completa la generación de un dataset sintético, `EntrenamientoHttpAdapter` escribe dos archivos `.csv` en esa ruta (`dataset_train.csv` y `dataset_val.csv`). Inmediatamente después notifica a FastAPI mediante POST

Figura 17

Contrato JSON resultante del proceso de optimización (Gurobi - simulación)

```
{
"cronograma": {
    str: {
        "siembra": int,
        "cosecha": int
    }
},
"asignaciones": [
    {
        "estanque_id": str,
        "pedido_id": str,
        "kg": double
    }
],
"metas": [
    {
        "pedido": str,
        "demanda": double,
        "entregado": double,
        "exceso": double,
        "deficit": double
    }
],
"factible": boolean,
"valor_objetivo": float
}
```

Nota. El campo `factible` indica si Gurobi encontró al menos una solución incumbente dentro del límite de tiempo configurado.

`/lstm/train` y Python lee los archivos directamente desde esa misma ruta para iniciar el proceso de entrenamiento.

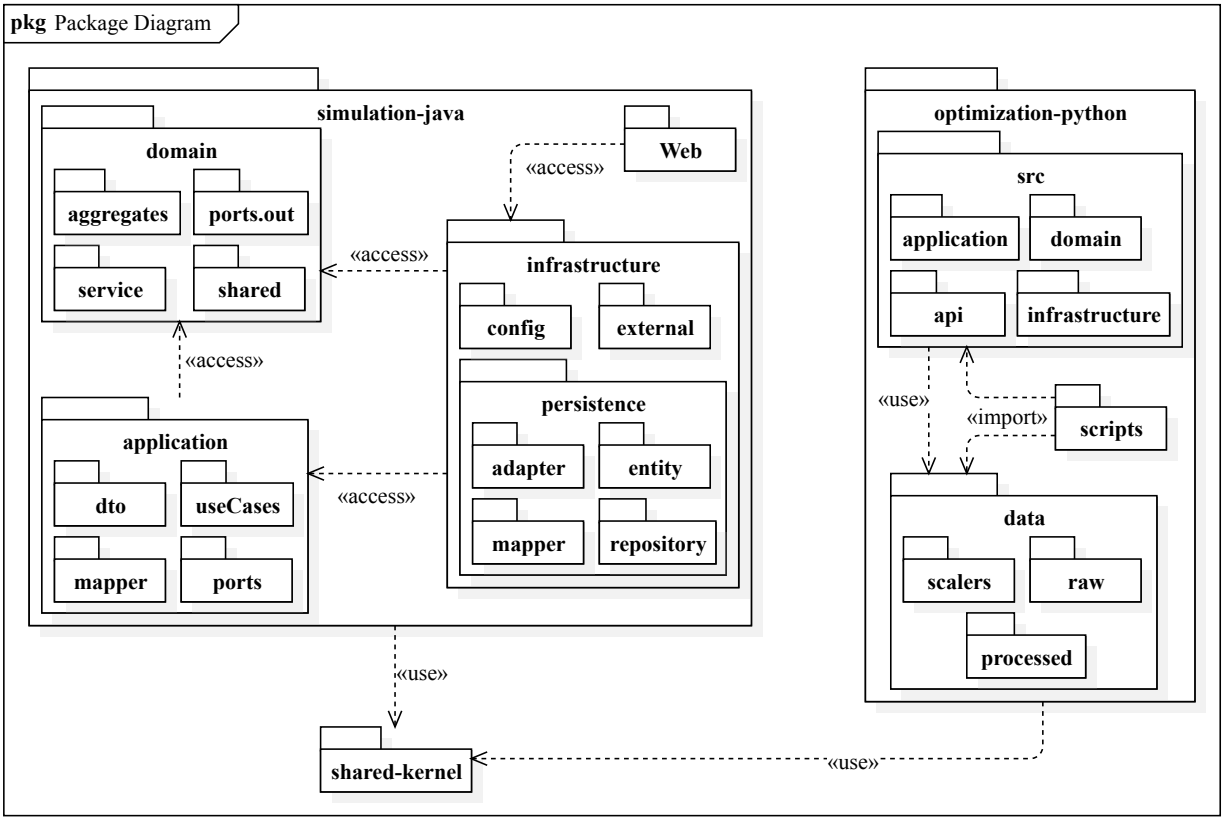
Este esquema de integración por archivo compartido es preferible a transmitir el dataset completo en el cuerpo JSON en la solicitud, dado que los datasets pueden contener decenas de miles de filas y su serialización JSON introduciría una sobrecarga de memoria y latencia considerable en ambos procesos.

De manera complementaria, la organización interna de cada subsistema refleja su arquitec-

tura de capas y justifica la ubicación de los patrones de diseño descritos anteriormente. La Figura 18 presenta el Diagrama de Paquetes de cada entorno de desarrollo y su relación con la carpeta común `shared-kernel`.

Figura 18

Diagrama de paquetes del artefacto



5. Resultados

Este capítulo presenta la evidencia de cumplimiento de los cuatro objetivos específicos del proyecto, en el mismo orden en que fueron formulados en el Capítulo 1. Cada sección contrasta los resultados obtenidos con los criterios de evaluación declarados en la Sección 3.3, de manera que la verificación de cada objetivo se realiza contra métricas y condiciones de éxito definidas antes de la

ejecución de los experimentos. Las decisiones de diseño e implementación que hicieron posibles estos resultados no se repiten aquí; se encuentran documentadas en el Capítulo 4.

5.1 Solidez del Sistema

Antes de contrastar los objetivos específicos, se verifican las dos propiedades transversales de las que depende la validez de todo el diseño experimental: que el motor de simulación sea reproducible, condición que legitima fijar la semilla en los bloques, y que el percentil utilizado en Monte Carlo sobre el que se construyen las proyecciones sea estable con el número de réplicas adoptado. Ambas no constituyen evidencia de un objetivo en particular; más bien, son el fundamento que hace confiables las mediciones de los cuatro objetivos.

5.1.1 Determinismo del Motor

Se comprobó el determinismo generando dos veces el conjunto de datos con la misma semilla (42) y contrastando sus 300 registros celda a celda: la diferencia máxima absoluta resultó nula en todas las columnas, es decir, ambas ejecuciones produjeron trayectorias idénticas. Como control positivo, una tercera ejecución con una semilla distinta (10) sí divergió, con una diferencia máxima de 33,29 g en el peso proyectado, lo que confirma que la identidad anterior no proviene de una ausencia de aleatoriedad, sino de un sembrado genuinamente reproducible. La Tabla 7 resume el contraste.

Tabla 7
Verificación del determinismo del motor de simulación

Contraste	**Semillas**	**Máx. $\lvert\Delta\rvert$**	**¿Trayectoria idéntica?**
Repetición (reproducibilidad)	42 vs. 42	0	Sí
Control positivo	42 vs. 10	33,29 g	No

Esta identidad exacta es la evidencia empírica que respalda la simplificación metodológica de fijar la semilla en los bloques de tiempo, pues, como el plan resultante no varía entre repeticiones, cronometrar cada configuración de los Bloques A, B y C mide limpiamente el costo computacional y no el ruido del azar. En cambio, la calidad del plan sí depende de la semilla, porque esta gobierna la trayectoria biológica proyectada y, con ella, los parámetros que recibe el optimizador; por ello el Bloque D se repite sobre varias semillas independientes.

5.1.2 Convergencia del Muestreo Monte Carlo

Manteniendo fijos la semilla (42), el estanque y la demanda, se ejecutó una simulación por cada número de réplicas $R \in \{1, 2, 5, 10, 20, 50, 100\}$ y se registró el percentil P_{20} del peso y la tasa de supervivencia proyectados de una ruta de referencia fija (siembra en el día 0, cosecha al vencimiento del pedido 257) en la primera petición del horizonte ($t_0 = 0$). La Tabla 8 muestra cómo dicha estimación se estabiliza al aumentar R.

Tabla 8
Convergencia del peso proyectado frente al número de réplicas Monte Carlo

R	**Peso proyectado [g]**	**Supervivencia proyectada [%]**	**Tiempo de cómputo [ms]**	**Variación relativa vs. $R = 1000$ [%]**
1	584,113	0,760	406	2,401
2	586,191	0,763	496	2,765
5	585,709	0,761	928	2,681
10	583,081	0,761	1658	2,220
20	574,252	0,767	3506	0,672
50	568,070	0,767	9225	0,411
100	570,853	0,767	17 754	0,076
200	571,934	0,767	46 492	0,266
500	571,988	0,768	127 273	0,275
1000	570,417	0,767	242 081	0

Tomando $R = 1000$ como valor de referencia por ser la mayor cantidad de réplicas evaluada, la Figura 19 muestra que, para $R \leq 10$, el peso proyectado todavía oscila de forma apreciable

(variación de hasta 2,77 %), mientras que a partir de $R = 20$ la variación cae por debajo de 0,7 % y se mantiene en ese orden para todo R mayor. La supervivencia proyectada (panel central) exhibe el mismo patrón, aunque con un rango absoluto mucho más estrecho, propio de una variable acotada entre 0 y 1, pues fluctúa entre 76,0 % y 76,3 % para $R \leq 10$ y se asienta en 76,7 % desde $R = 20$ en adelante. El panel inferior, en cambio, muestra que el tiempo de cómputo no comparte ese aplanamiento, dado que crece de forma aproximadamente proporcional a R en todo el rango evaluado, de 406 ms en $R = 1$ a poco más de cuatro minutos en $R = 1000$. La combinación de ambos comportamientos es la que justifica adoptar $R = 20$ como valor transversal del diseño experimental, debido a que por debajo de ese punto las proyecciones todavía no son confiables, y por encima el costo computacional sigue aumentando sin que la precisión mejore en una proporción comparable.

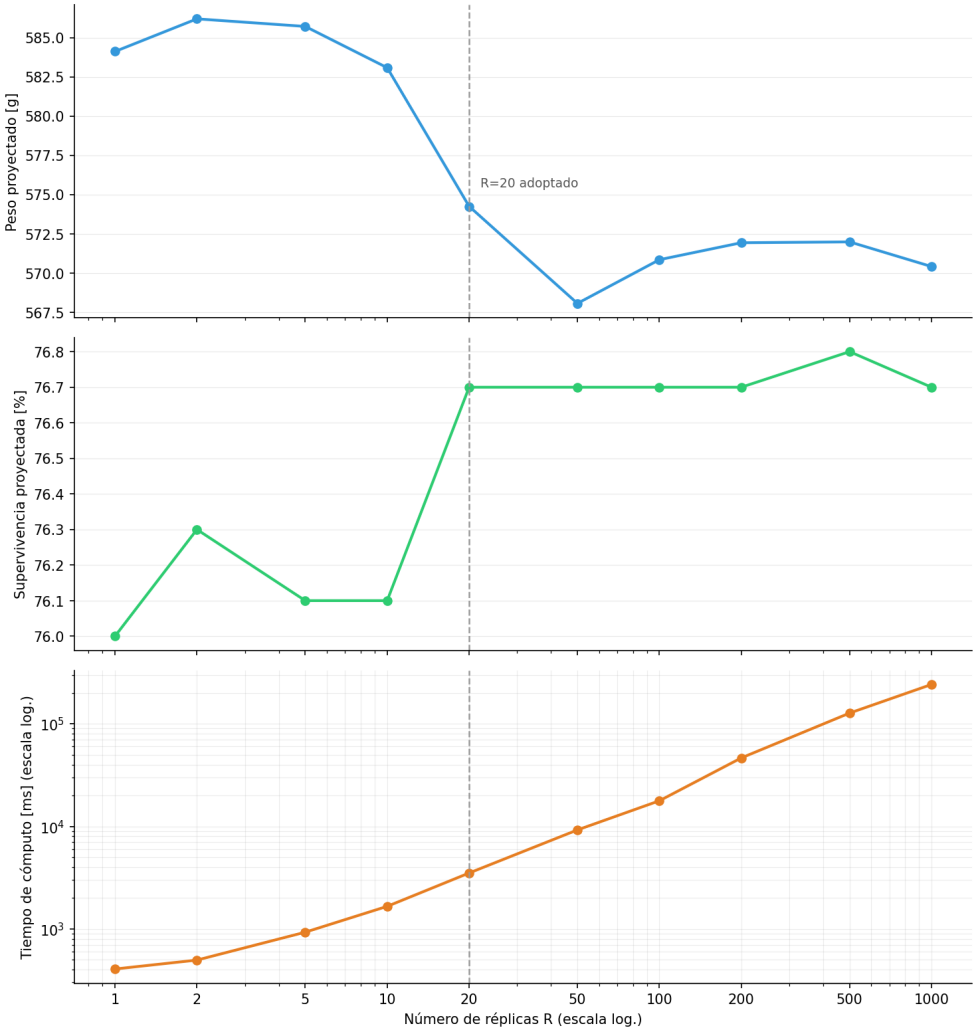
En conjunto, ambas propiedades sostienen la validez del diseño experimental que se contrasta en el resto de este capítulo. El determinismo confirmado avala fijar la semilla al cronometrar los Bloques A, B y C, mientras que la convergencia respalda adoptar $R = 20$ como número de réplicas transversal.

5.2 OE1 – Arquitectura multi-estanque escalable

El primer objetivo específico exige que la arquitectura maneje múltiples estanques de forma autónoma y escalable. Dado que la reingeniería que hizo posible esas dos propiedades pertenece al *cómo* y se documenta en el Capítulo 4, el objeto de análisis en esta sección es su manifestación observable, es decir, si declarar y coordinar un número creciente de estanques exige del operador un esfuerzo proporcional al tamaño de la asociación, o si el sistema absorbe esa complejidad internamente.

Figura 19

Convergencia del peso, la supervivencia proyectados y del tiempo de cómputo frente al número de réplicas Monte Carlo

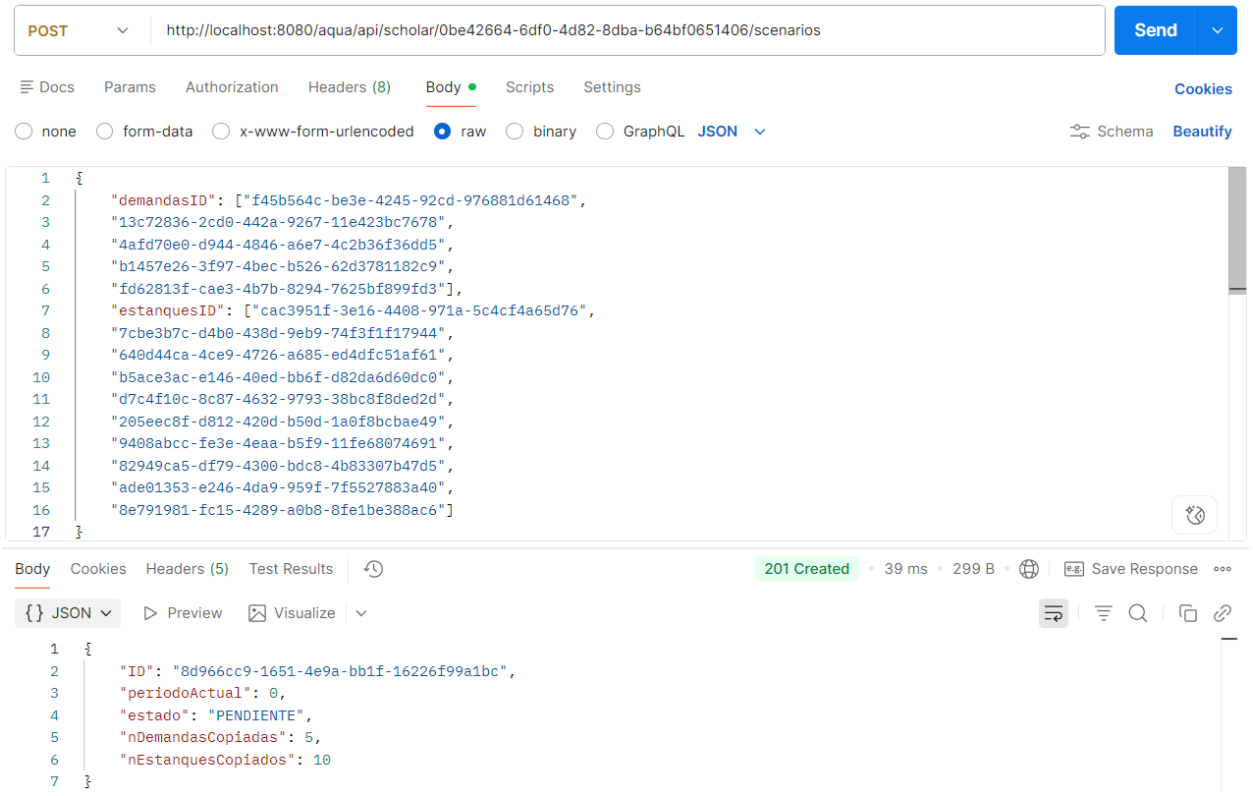


5.2.1 Autonomía y Configuración Escalable

La configuración de la asociación se realiza en una única llamada, tal que el escenario de simulación se crea declarando de una sola vez la lista completa de estanques disponibles, sin necesidad de registrar cada uno mediante una petición independiente. La Figura 20 muestra el cuerpo del *request* utilizado durante las pruebas para crear un escenario con $\Gamma = 10$ estanques, evidenciando que la arquitectura es escalable también en su interfaz de configuración.

Figura 20

Creación de un escenario con múltiples estanques en una única petición ($\Gamma = 10$)

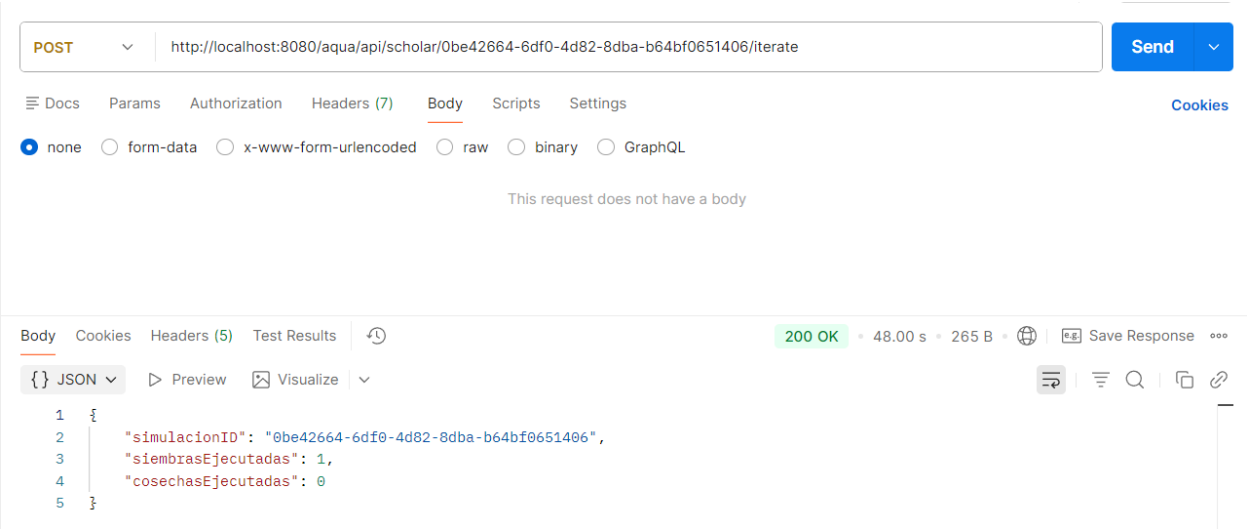


Una vez configurada la asociación, el mecanismo de entrada para la operación diaria es un único *endpoint* REST (`POST /aqua/api/scholar/{id}/iterate`), cuyo cuerpo no contiene ningún parámetro específico de estanque. La Figura 21 muestra la petición realizada durante las pruebas, confirmando que la interfaz expuesta al usuario no requiere conocimiento alguno de la distribución interna de los ciclos productivos.

Internamente, esa única llamada desencadena la proyección Monte Carlo de cada estanque disponible, la resolución del modelo MILP y la escritura del plan resultante en la base de datos, evidenciando que el sistema es autónomo. La Figura 27 (presentada en detalle en la Sección 5.4) muestra el producto directo de esa llamada: 10 estanques con fechas de siembra y cosecha diferentes, ninguna de las cuales fue especificada por el operador. Cada fila del diagrama es una decisión independiente calculada por el sistema a partir del estado biológico actual de ese estanque, del

Figura 21

Petición HTTP de iteración desde cliente REST ($\Gamma = 10$, $\Psi = 5$)



inventario de pedidos pendientes y de los parámetros de la función objetivo.

5.2.2 Complejidad del Tiempo de Respuesta (Bloque A)

La complejidad estructural se cuantifica midiendo cómo responde el tiempo de cómputo cuando crece el número de estanques de la asociación. Manteniendo fijos el número de pedidos ($\Psi = 5$) y el horizonte de planificación, se cronometró la primera petición del horizonte deslizante ($t_0 = 0$, peor caso para esta dimensión) para asociaciones de tamaño creciente. La Tabla 9 resume los tiempos promedio obtenidos sobre tres réplicas por configuración, y la Figura 22 ilustra gráficamente esta misma tendencia.

Los datos evidencian una relación prácticamente lineal entre Γ y el tiempo de respuesta dentro del rango evaluado, pues cada estanque adicional añade un costo aproximadamente constante (entre 4,2 y 5,0 s por estanque), sin que aparezca una degradación superlineal que comprometa la operación de asociaciones de tamaño moderado. Esa linealidad confirma que la arquitectura escala sin penalización estructural, absorbiendo internamente el costo de coordinar más estanques, dentro

Tabla 9

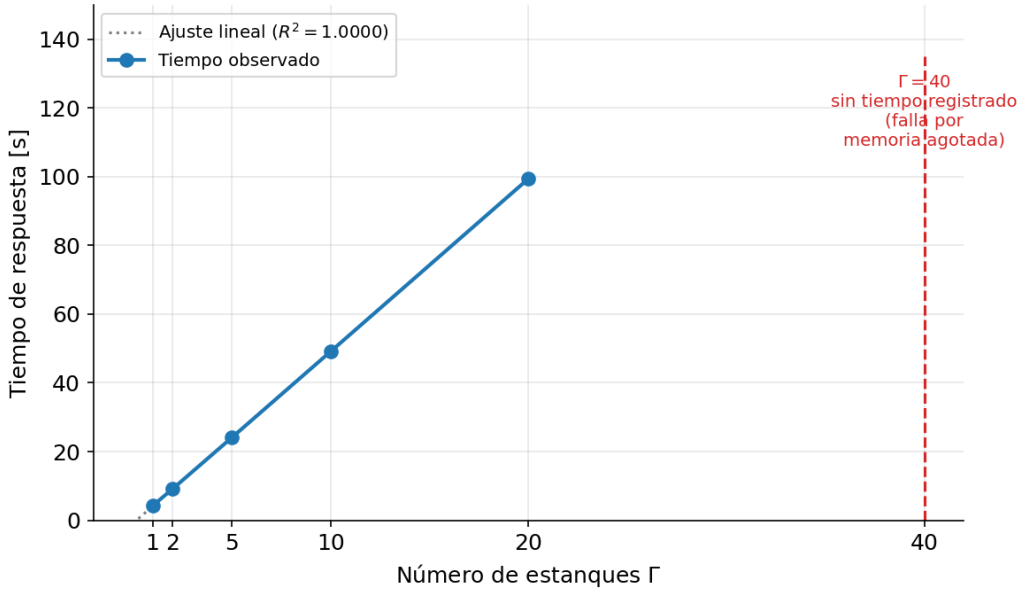
Tiempo de respuesta promedio en función del número de estanques (Bloque A)

Γ	Intento 1 [s]	Intento 2 [s]	Intento 3 [s]	Promedio [s]
1	4,07	4,21	4,39	4,22
2	9,04	9,08	9,11	9,08
5	23,96	24,00	24,19	24,05
10	50,43	48,98	48,00	49,14
20	96,66	98,94	102,78	99,46
40	Error	Error	Error	-

del rango en que la memoria disponible sostiene la proyección Monte Carlo. No obstante, al extender la prueba a Γ = 40, los tres escenarios fallaron, un comportamiento consistente con el agotamiento de la memoria durante la proyección Monte Carlo, y no con un límite de tiempo de cómputo del optimizador. Por tanto, la cantidad máxima de estanques simulables bajo estos recursos se ubica, en consecuencia, entre Γ = 20 y Γ = 40.

Figura 22

Tiempo de respuesta en función del número de estanques (Bloque A)



5.3 OE2 – Modelo predictivo calibrado

El segundo objetivo específico atañe al ajuste del modelo de red neuronal recurrente LSTM encargado de predecir, a partir de los datos generados por el simulador de dinámica de sistemas, las variables biológicas que determinan la coyuntura ideal de cosecha de cada estanque, estas son, el incremento diario de peso y la ración requerida. La presentación de resultados sigue el orden natural del proceso de calibración: primero se determina el volumen de datos sintéticos, después se caracteriza el *dataset* final adoptado y el comportamiento observado durante el entrenamiento, y finalmente se contrasta la capacidad predictiva del modelo frente al simulador de referencia, tanto de forma cuantitativa como visual.

5.3.1 Tamaño del Dataset

Con el propósito de determinar el volumen de datos necesario para obtener un modelo con desempeño aceptable, se entrenó el modelo con diferentes cantidades de estanques simulados, manteniendo la misma semilla base en todos los casos para garantizar comparabilidad. Los tamaños evaluados fueron 20, 50, 100, 200 y 400 estanques, correspondientes a divisiones del 80 % para entrenamiento y el 20 % para validación.

La Tabla 10 presenta los resultados obtenidos. Se demuestra que el modelo requiere un mínimo de datos para generalizar la dinámica biológica, ya que las pruebas con 20, 50 y 100 estanques arrojaron márgenes de error considerables. Esta tendencia cambia al alcanzar los 200 estanques, donde las métricas de error mejoran notablemente.

5.3.2 Dataset Generado

Partiendo de la cantidad aceptable a generar, el dataset comprende 400 estanques simulados, cada uno con una trayectoria de crecimiento diaria que abarca diferentes condiciones iniciales

Tabla 10
Análisis de sensibilidad del error respecto al tamaño de muestras N

N	**Train**	**Val**	**RMSE Δpeso** (g)	**RMSE Ración** (g/d)	**NRMSE Δpeso**	**NRMSE Ración**
20	16	4	1,4259	4,0988	22,16 %	10,56 %
50	40	10	1,2649	3,0363	19,66 %	13,29 %
100	80	20	1,2297	3,3394	19,11 %	11,89 %
200	160	40	0,303	0,9265	4,71 %	2,46 %
400	320	80	0,226	0,8117	3,51 %	2,50 %

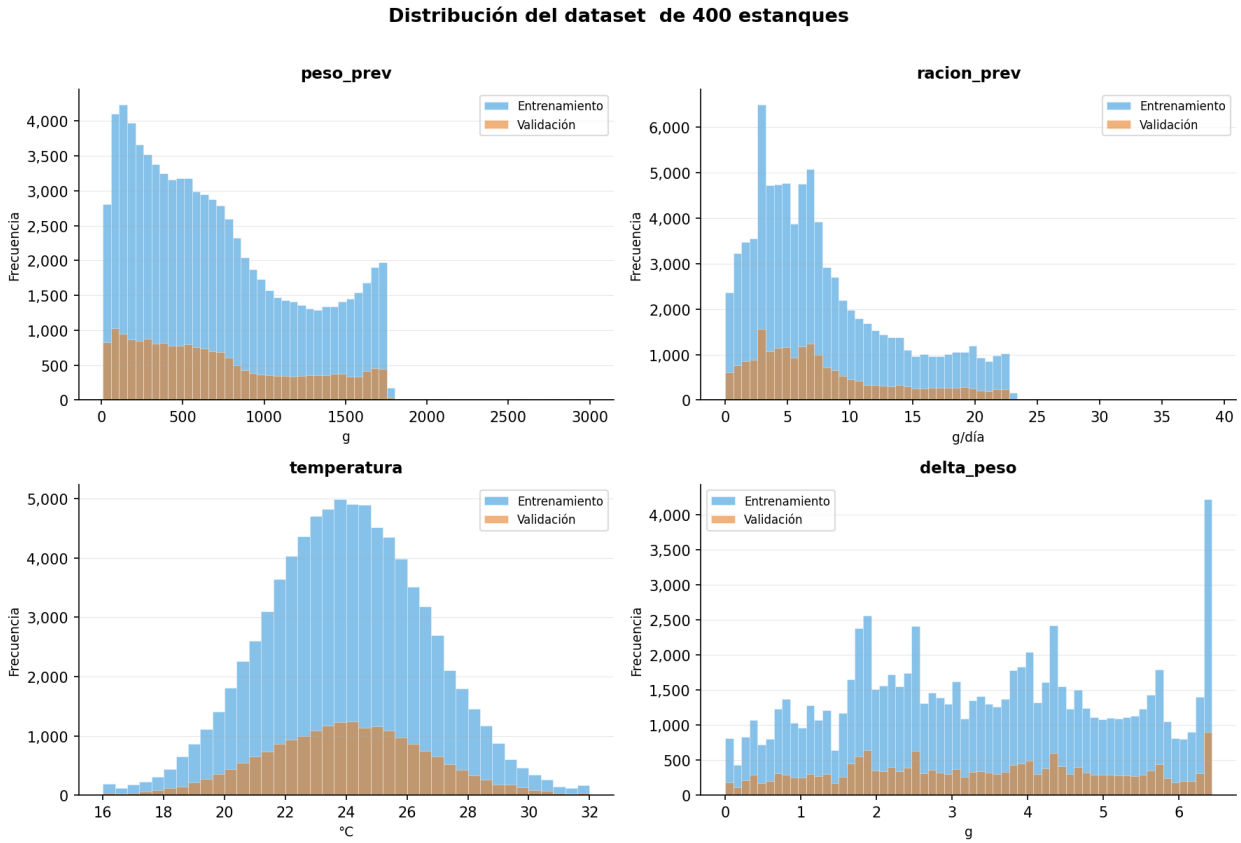
de peso y temperatura. La partición asigna 320 estanques al conjunto de entrenamiento y 80 al conjunto de validación, quedando compuesto por 83 863 filas para entrenamiento y 20 216 filas para validación, sumando un total de 104 079 registros. En la Figura 23 se presenta la superposición de las distribuciones para los conjuntos de entrenamiento y validación, confirmando que la partición aleatoria por estanque garantiza muestras comparables. Por un lado, se observa un sesgo hacia valores bajos en la variable *peso_prev*, lo cual es un resultado directo del muestreo log-uniforme utilizado para la inicialización; esta estrategia asegura una representación proporcional de las diversas etapas del ciclo productivo. Por otro lado, las variables *racion* y *delta_peso* reflejan el comportamiento derivado del modelo de dinámica de sistemas, mientras que la temperatura exhibe una distribución normal centrada en 24 °C.

5.3.3 Proceso de Entrenamiento

El modelo fue entrenado durante un máximo de 100 *epoch*′s. En este experimento el entrenamiento completó las 100 épocas, alcanzando la mejor configuración de pesos en la época 92 con una pérdida de validación de *val_loss* = 0,010 79. La Figura 24 presenta la evolución de la pérdida de entrenamiento y de validación a lo largo de los *epoch*′s. Ambas curvas descienden de forma sostenida, lo que indica que el modelo está aprendiendo patrones generalizables y no memorizando el conjunto de entrenamiento.

Figura 23

Distribución de los datos de entrenamiento y validación por variable



5.3.4 Trayectorias Diarias

Una vez entrenado el modelo LSTM, se contrastan los pronósticos con los valores producidos por el simulador de DS sobre tres estanques del conjunto de validación. La Figura 25 muestra que el modelo LSTM sigue la tendencia del simulador en los tres estanques, capturando los cambios asociados a variaciones de temperatura.

A su vez, la Figura 26 compara la trayectoria del peso acumulado real, obtenida integrando los incrementos diarios producidos por el simulador, con la reconstruida por el modelo LSTM a partir de sus predicciones de *delta_peso* en modo autorregresivo.

En los tres estanques evaluados el peso acumulado predicho por el LSTM se mantiene

Figura 24

Curva de pérdida de entrenamiento y validación por epoch

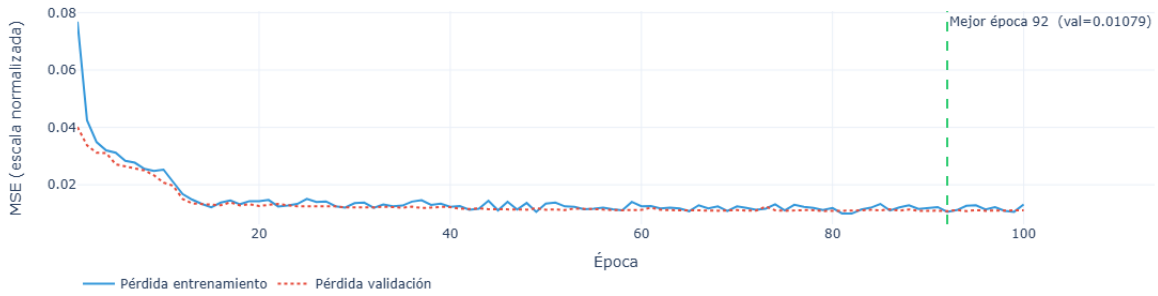
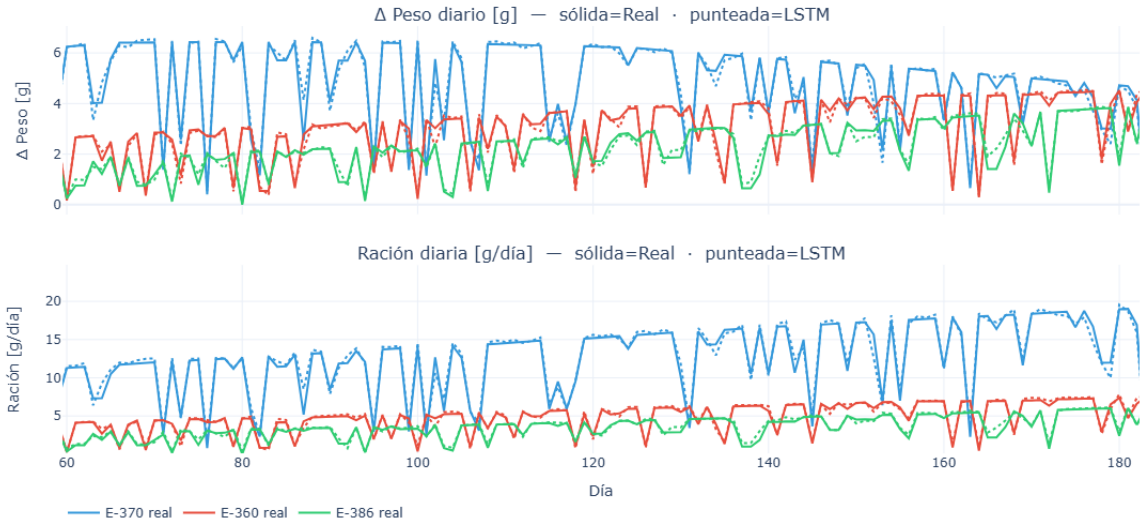


Figura 25

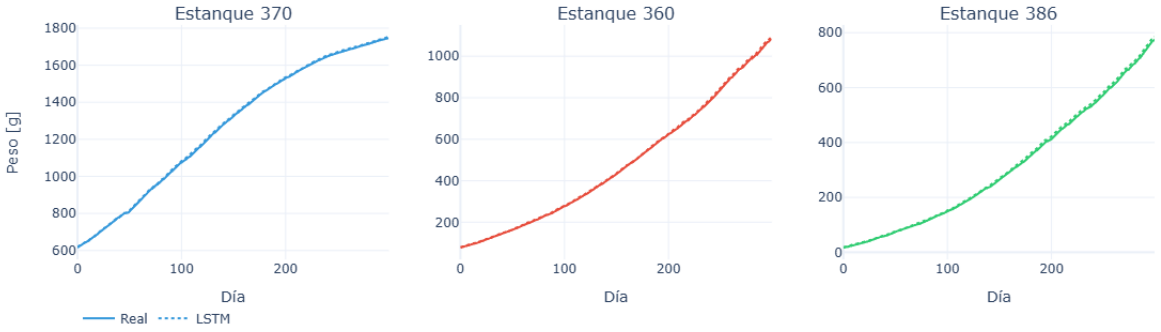
DS vs. LSTM en la estimación de Δpeso y ración suministrada para los estanques 370, 360 y 386



cercano a la referencia a lo largo de todo el horizonte, sin que se observe una divergencia. Por tanto, se evidencia un resultado final cercano al real.

Figura 26

DS vs. LSTM en el crecimiento del pez para los estanques 370, 360 y 386



5.3.5 Evaluación del Rendimiento Predictivo

Para probar la capacidad de generalización del modelo se evaluaron tres niveles. El Nivel 1 evalúa el modelo sobre el mismo conjunto de entrenamiento y el Nivel 2 lo evalúa sobre el conjunto de validación reservado durante el entrenamiento. La Tabla 11 muestra los resultados de ambos niveles.

Tabla 11

Evaluación entre el conjunto de entrenamiento y de validación

Conjunto	**RMSE Δpeso** (g)	**RMSE Ración** (g/d)	**NRMSE Δpeso**	**NRMSE Ración**
Entrenamiento	0,2579	0,7841	4,01 %	2,01 %
Validación	0,226	0,8117	3,51 %	2,50 %

Como resultado, el modelo obtiene errores comparables en ambos conjuntos, sin que se observe una degradación significativa al pasar de entrenamiento a validación. El Nivel 3, en cambio,

evalúa el modelo sobre conjuntos generados con semillas distintas, lo que permite simular datos de operación completamente independientes. La Tabla 12 resume los resultados para cuatro semillas distintas.

Tabla 12
Evaluación entre conjuntos externos generados con semillas diferentes

Conjunto	RMSE Δpeso (g)	RMSE Ración (g/d)	NRMSE Δpeso	NRMSE Ración
Semilla 1	0,2654	0,7667	4,13 %	2,35 %
Semilla 60	0,2642	0,7675	4,11 %	2,35 %
Semilla 95	0,2657	0,7689	4,13 %	2,26 %
Semilla 201	0,2621	0,780	4,07 %	2,46 %

Los resultados son estables: el RMSE de Δpeso presenta pequeñas variaciones entre semillas y el NRMSE se mantiene por debajo del 4,2 % en todos los casos. Esta consistencia a través de cuatro configuraciones de datos independientes confirma que el rendimiento observado en validación no es producto de una partición favorable, sino una propiedad robusta del modelo.

5.4 OE3 – Plan estratégico de coordinación

El tercer objetivo busca implementar un módulo de optimización para generar planes estratégicos concretos que coordinen los ciclos productivos de una asociación multi-estanque y así garantizar una producción continua. Por tanto, el objeto de análisis es el plan mismo: su estructura temporal, el espacio de decisión explorado para construirlo y su capacidad efectiva de cumplir la demanda.

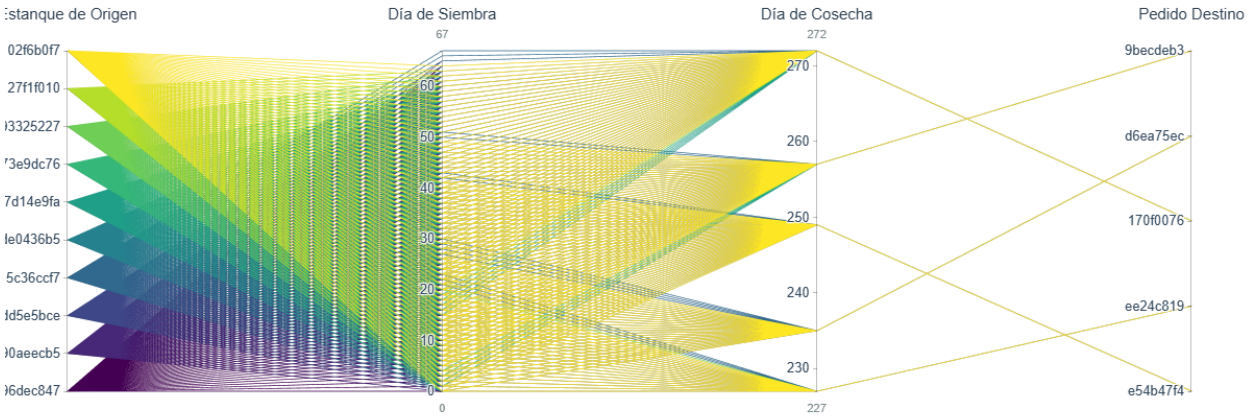
5.4.1 Cronograma de Operaciones

Antes de siquiera crear el cronograma, Gurobi obtiene un conjunto de combinaciones (i, s, t, j) que representan el estanque de origen, día de siembra, día de cosecha y pedido destino, respectivamente. En la Figura 27 se visualizan, mediante coordenadas paralelas, las relaciones del

conjunto; cada línea representa una ruta factible que el optimizador consideró antes de seleccionar el subconjunto de asignaciones que minimiza la función objetivo.

Figura 27

Espacio de búsqueda explorado por el optimizador ($t_0 = 0$)



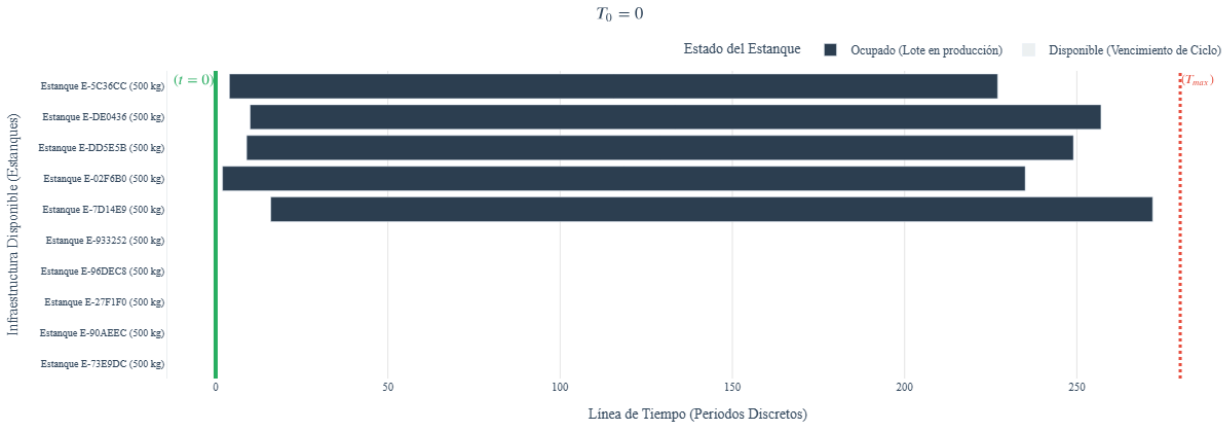
La densidad de líneas en el gráfico refleja la complejidad combinatoria del problema: para $\Gamma = 10$ estanques y un horizonte de 280 días, el espacio Ω incluye del orden de miles de tuplas válidas (i, s, t). El color de cada línea codifica el estanque de origen, evidenciando que el optimizador distribuyó las asignaciones entre todos los estanques disponibles y no concentró la producción en un subconjunto reducido.

Como resultado, Gurobi genera un cronograma, presentado como un Diagrama de Gantt (Figura 28), según las combinaciones factibles presentadas anteriormente, eligiendo la que mejores resultados tenga. Cada barra horizontal representa un ciclo productivo completo, desde la siembra hasta la cosecha, asignado a un estanque por el modelo MILP.

El plan resultante activa 5 de los estanques disponibles, con siembras escalonadas entre los días 0 y 50 y cosechas distribuidas entre los días 200 y 280.

Figura 28

Cronograma de operaciones generado por el optimizador ($t_0 = 0$, $\Gamma = 10$, $\Psi = 5$)

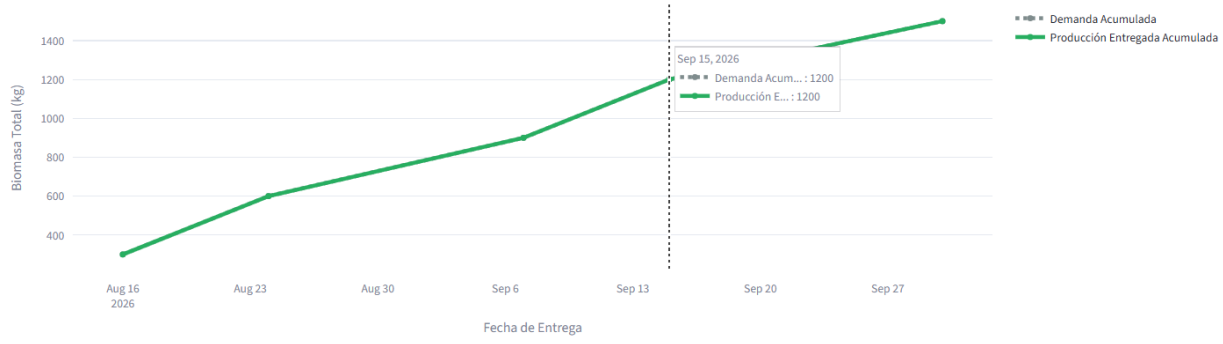


5.4.2 Cumplimiento de la Demanda

La Figura 29 contrasta la demanda acumulada (línea gris discontinua) con la producción entregada acumulada (línea verde sólida) a lo largo del horizonte de simulación completo, una vez ejecutadas todas las iteraciones del horizonte deslizante.

Figura 29

Producción entregada acumulada vs. demanda acumulada ($\Gamma = 10$, $\Psi = 5$, demanda total 1500 kg)



La curva de producción está sobrepuesta a la línea de la demanda desde la primera entrega

y mantiene esa posición durante todo el horizonte. Esto confirma que la asociación logró una producción continua, es decir, no hubo períodos de desabastecimiento ni acumulación tardía de entregas.

5.4.3 Umbral de Viabilidad del Plan (Bloque C)

La capacidad de generar un plan viable no es incondicional, pues depende de que el horizonte de planificación sea lo bastante amplio para que el ciclo biológico alcance el peso comercial exigido por la demanda. Para delimitar ese punto, manteniendo fijos $\Gamma = 6$ y $\Psi = 5$, se varió el horizonte de planificación y se identificó a partir de qué valor el sistema lograba producir planes con al menos una siembra viable. La Tabla 13 presenta los tiempos de respuesta promedio junto con la indicación de si se registró alguna siembra exitosa.

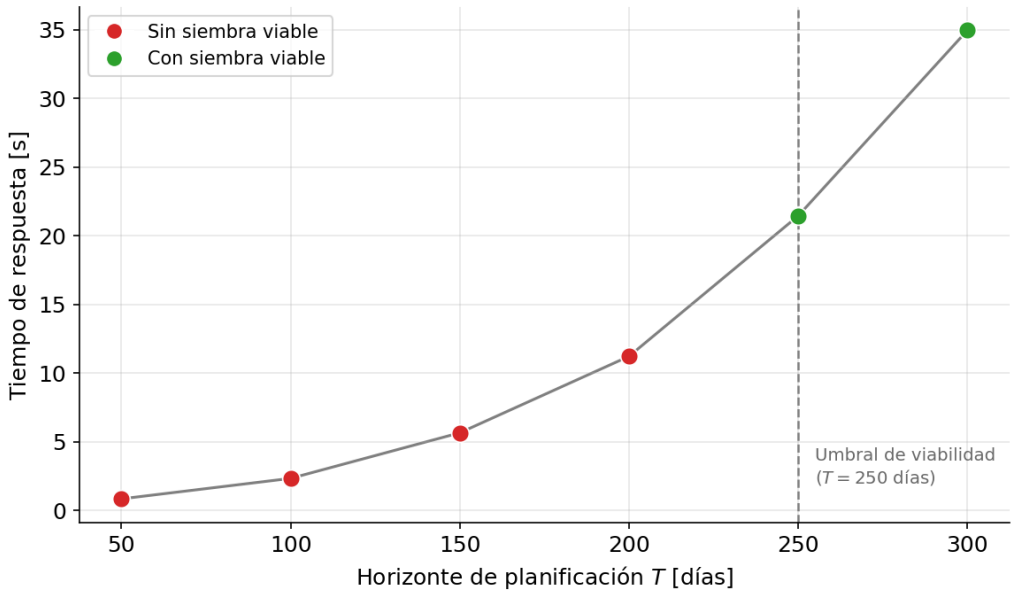
Tabla 13
Tiempo de respuesta promedio y viabilidad en función del horizonte de planificación (Bloque C)

T (Días)	Intento 1 (s)	Intento 2 (s)	Intento 3 (s)	Promedio (s)	¿Hubo siembra?
50	0,994	0,728	0,775	0,83	No
100	2,40	2,24	2,31	2,32	No
150	5,52	5,60	5,81	5,64	No
200	10,60	10,79	12,25	11,21	No
250	21,45	21,38	21,48	21,44	Sí
300	36,60	37,55	33,71	34,95	Sí

Los resultados, visualizados en la Figura 30, delimitan un umbral biológico, para esta configuración (estanques y pedidos con peso objetivo entre 400 y 550 g), donde ningún plan resultó viable para $T \leq 200$ días, mientras que a partir de $T = 250$ días el sistema comenzó a generar siembras exitosas de forma consistente. Este resultado es independiente de la capacidad de cómputo disponible, de modo que, por debajo del umbral, ninguna combinación de siembra-cosecha permite que el pez alcance el peso comercial exigido dentro del horizonte evaluado, sin importar cuántos estanques o pedidos estén disponibles para coordinar.

Figura 30

Tiempo de respuesta y umbral de viabilidad en función del horizonte de planificación (Bloque C)



5.4.4 Calibración de la Función Objetivo (Bloque E)

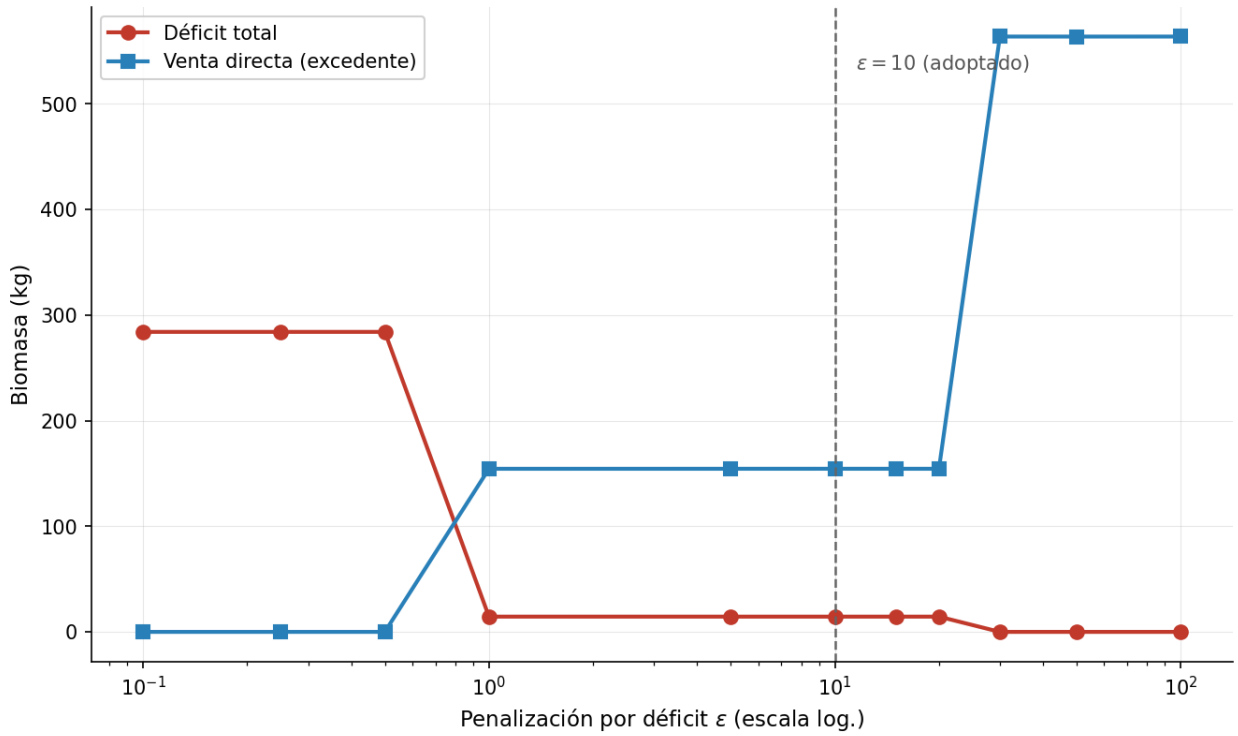
La validez del plan no solo exige un horizonte biológicamente suficiente; exige también que los pesos de la función objetivo estén adecuadamente fijados, pues son ellos los que determinan cómo el modelo arbitra los compromisos al construir el cronograma.

De esta manera, la primera variable es la penalización por déficit (ϵ), la cual se barre, de forma independiente, sobre un escenario de $\Gamma = 12$ estanques de 500 kg de capacidad y $\Psi = 6$ pedidos de tamaño único —entre 520 y 950 kg, para una demanda total de 4110 kg— agrupados en tres fechas de entrega. La exigencia del escenario no proviene de una capacidad insuficiente, ya que la hay de sobra, sino de su carácter discreto, en particular la demanda de cada fecha no es un múltiplo exacto de la biomasa que rinde un estanque, de modo que cubrir la última fracción de un pedido obliga a cosechar un estanque completo cuyo sobrante se convierte en excedente. El modelo debe entonces elegir, en el margen, entre dejar un déficit o incurrir en ese exceso, y es precisamente ϵ quien gobierna ese arbitraje. La Figura 31 y la Tabla 14 resumen cómo se desplaza

el par déficit-excedente a lo largo del rango evaluado.

Figura 31

Evolución del déficit y del excedente (venta directa) en función de la penalización ϵ (Bloque E)



El comportamiento reproduce el compromiso característico de la programación por metas: a medida que aumenta ϵ, el optimizador prioriza cada vez más evitar el incumplimiento, de modo que el déficit se reduce, de 708 kg con $\epsilon = 0{,}1$ hasta anularse a partir de $\epsilon \approx 8$, a costa de un excedente que crece en sentido inverso, de cero a 562 kg. La transición no es continua, sino escalonada, porque cada estanque marginal se abre por completo, es decir, el salto entre $\epsilon = 5$ y $\epsilon = 8$ elimina los últimos 44 kg de déficit al precio de 379 kg adicionales de venta directa, lo que evidencia los retornos decrecientes de seguir penalizando el incumplimiento. El valor adoptado ($\epsilon = 10$) cae con margen dentro del régimen de cobertura total, este garantiza el abastecimiento íntegro, déficit nulo desde $\epsilon \approx 8$, sin que un ϵ mayor aporte mejora alguna ni genere excedente adicional injustificado, lo que fundamenta empíricamente la elección que hasta aquí se había mantenido como supuesto.

El segundo peso de la función objetivo es la prioridad de recurso α_s, pero no se somete a un

Tabla 14
Déficit, venta directa y cobertura para valores representativos de ϵ (Bloque E)

ϵ	**Déficit (kg)**	**Venta directa (kg)**	**Cobertura (%)**
0,10	708,1	0,0	82,77
0,25	338,7	54,1	91,76
0,50	44,3	183,3	98,92
1	44,3	183,4	98,92
5	44,2	183,4	98,92
8	0,0	562,4	100,00
10	0,0	562,4	100,00
100	0,0	562,4	100,00

barrido porque no constituye un parámetro calibrable, sino una política de diseño. La preferencia por reutilizar los estanques ya sembrados antes de iniciar ciclos nuevos está determinada por el dominio del término de uso de recursos; en términos prácticos, las rutas de un lote heredado reciben un premio negativo y las siembras nuevas una penalización positiva, de modo que el modelo antepone el inventario existente para cualquier valor positivo de α_s.

5.4.5 Uso Exclusivo del Modo DS en la Validación

Antes de cerrar este objetivo, es necesario precisar que los bloques de validación del subsistema de optimización presentados en esta sección y en la Sección 5.5 se ejecutaron íntegramente bajo el modo de simulación DS, pese a que la redacción del tercer objetivo específico plantea que el plan estratégico se construye «utilizando el pronóstico del modelo neuronal».

Esta decisión no compromete la validez de las conclusiones alcanzadas, porque el OE2 ya demostró que el modelo LSTM reproduce la dinámica del simulador DS con un NRMSE inferior al 4,2 % incluso ante estanques generados con semillas jamás vistas durante el entrenamiento (Tablas 11 y 12), y que la trayectoria de peso acumulado que reconstruye en modo autorregresivo permanece cercana a la referencia a lo largo de todo el horizonte, sin divergencia apreciable (Figura 26). Dado que el optimizador consume únicamente el percentil P_{20} de peso y supervivencia proyectado por estanque, y no la trayectoria diaria completa, es razonable esperar que alimentarlo

con proyecciones del modo IA en lugar de DS produzca cronogramas estructuralmente equivalentes a los aquí presentados, dentro del margen de error ya caracterizado.

Más allá de esta equivalencia funcional, la elección del modo DS evita también la sobrecarga computacional que introduce, por diseño, la arquitectura del modo IA: mientras que el modo DS resuelve cada proyección de crecimiento localmente dentro de la JVM, el modo IA delega esa misma proyección al servicio de predicción de Python, incurriendo en una solicitud HTTP y en el costo de inferencia del modelo LSTM por cada trayectoria evaluada. Dado que la iteración de planificación del horizonte deslizante proyecta una trayectoria independiente por cada día candidato de siembra y por cada réplica de Monte Carlo, este costo se multiplica junto con el tamaño del espacio de candidatos evaluado, una sobrecarga que se hizo evidente durante la operación del sistema y que motiva, también desde una perspectiva práctica, aislar el costo propio del optimizador del costo de la red neuronal durante la validación.

5.5 OE4 – Validación integral del sistema

El último objetivo evalúa la viabilidad y la eficiencia de los planes estratégicos generados por el sistema al simular escenarios que contemplan diversas variables de producción y demanda.

5.5.1 Bloque 0 – Caso de Control

En el caso más simple posible, un único estanque y un único pedido, el ciclo completo de siembra y cosecha se ejecutó, confirmando la funcionalidad del horizonte deslizante de extremo a extremo. El seguimiento del tiempo de respuesta a lo largo de las iteraciones diarias muestra un patrón consistente con el diseño arquitectónico del motor de simulación, mientras el estanque permanece disponible (sin lote sembrado), cada llamada del horizonte proyecta una trayectoria de crecimiento independiente para cada día candidato de siembra dentro del horizonte, mientras que una vez sembrado, solo se proyecta una única trayectoria heredada. La Tabla 15 ilustra esta

transición.

Tabla 15
Tiempo de respuesta antes y después del día de siembra (Bloque 0)

Iteración	Tiempo (s)	Estado del estanque
0	4,17	Disponible
1	4,12	Crianza (Siembra)
2	0,237	Crianza
18	0,230	Crianza
19	0,254	Pre-Engorde
50	0,235	Pre-Engorde
100	0,217	Pre-Engorde
111	0,229	Pre-Engorde
112	0,185	Engorde
200	0,249	Engorde
212	0,217	Cosecha

En la iteración 0, con el estanque disponible, el tiempo de respuesta fue de 4,17 s; en la iteración 1, en la que el optimizador decide sembrar, el tiempo aún refleja el costo de evaluar los candidatos de siembra (4,12 s). A partir de la iteración 2, ya con el lote sembrado, el tiempo cae a un rango de entre 0,185 y 0,254 s durante toda la fase de producción hasta que la cosecha ocurre en la iteración 212 (0,217 s). Este comportamiento confirma que el costo computacional del horizonte deslizante no es constante en el tiempo, sino que depende del estado de ocupación de los estanques en el instante de cada simulación.

5.5.2 Bloque B – Sensibilidad al Número de Pedidos

Bajo el mismo procedimiento de cronometraje de la primera petición del horizonte deslizante ($t_0 = 0$), pero manteniendo fijo el número de estanques ($\Gamma = 5$) y variando el número de pedidos simultáneos, se obtuvieron los resultados de la Tabla 16.

A diferencia de Γ, el número de pedidos tiene una incidencia marginal sobre el tiempo de respuesta, tal como contrasta la Figura 32 al superponer ambos bloques, a saber, al multiplicar Ψ por 20 (de 1 a 20), el tiempo promedio solo se incrementa en un 12,4 % (de 23,96 a 26,93 s), frente

Tabla 16
Tiempo de respuesta promedio en función del número de pedidos (Bloque B)

Ψ	**Intento 1 (s)**	**Intento 2 (s)**	**Intento 3 (s)**	**Promedio (s)**
1	23,74	23,78	24,36	23,96
2	24,45	23,33	23,11	23,63
5	23,29	23,78	24,54	23,87
10	23,84	26,02	24,73	24,86
20	27,47	26,32	27,01	26,93
40	31,71	30,05	30,97	30,91

al incremento de aproximadamente 2257 % observado al multiplicar Γ por el mismo factor en el Bloque A. Incluso extendiendo la comparación hasta $\Psi = 40$, más allá del punto en que Γ ya había agotado la memoria disponible, el incremento acumulado en el Bloque B es de apenas un 29,0 % (de 23,96 a 30,91 s). Esta asimetría se explica por el origen del costo computacional dominante: la proyección Monte Carlo del motor de simulación depende de Γ y de T, pero es independiente de Ψ; el número de pedidos solo incide indirectamente, a través del tamaño de las variables de asignación y las restricciones de calidad que recibe el optimizador MILP, un efecto comparativamente menor frente al costo base de la proyección biológica. En consecuencia, se concluye que el número de estanques, y no el número de pedidos, es el factor que determina el límite práctico de tiempo de respuesta del sistema.

5.5.3 Bloque D – Escalas de Asociación

En este punto, el interés se centra en contrastar el comportamiento del sistema integrado, motor de simulación, optimizador y el *dashboard* de seguimiento, a través de tres escalas de asociación crecientes, desde una pequeña hasta una grande. Para cada escenario se ejecutó el ciclo completo del horizonte deslizante, registrando la demanda total, la producción y el cumplimiento resultante. La Tabla 17 resume los resultados obtenidos.

Cabe aclarar que la columna *Cumplimiento* de la Tabla 17 expresa el porcentaje de la demanda del contrato efectivamente entregada y, por construcción, está acotada al 100 %, es decir,

Figura 32

Asimetría entre la sensibilidad del tiempo de respuesta a Γ (Bloque A) y a Ψ (Bloque B)

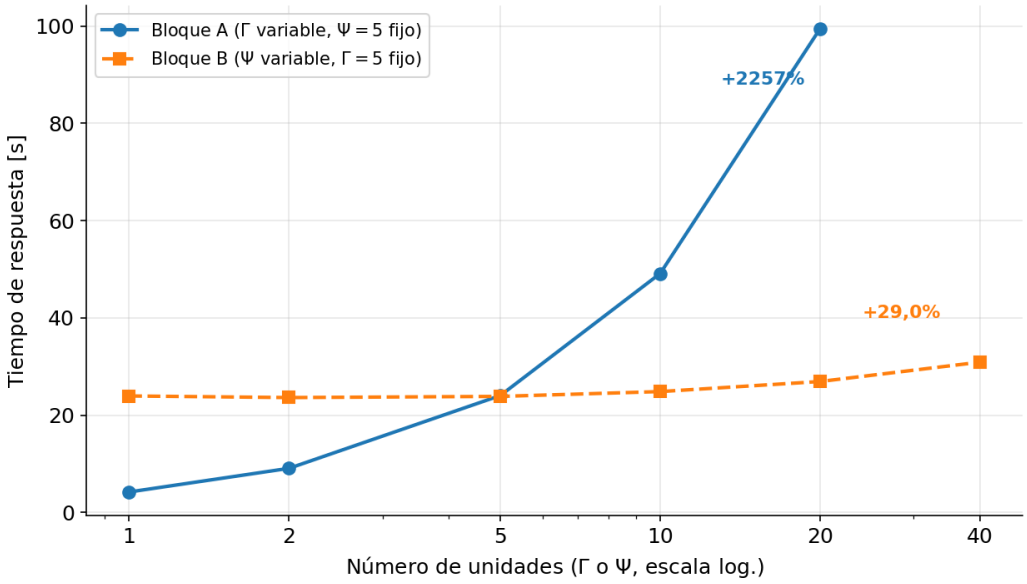


Tabla 17

Demanda, producción y cumplimiento por escala de asociación (Bloque D)

Escenario	Semilla	Γ	Ψ	Demanda (kg)	Producción (kg)	Venta directa (kg)	Cumplimiento
Pequeña	42	5	3	900	1404	504	100 %
Pequeña	10	5	3	900	1449	549	100 %
Pequeña	70	5	3	900	1408	508	100 %
Mediana	42	10	5	1500	2483	983	100 %
Mediana	10	10	5	1500	2434	934	100 %
Mediana	70	10	5	1500	2390	890	100 %
Grande	42	20	10	3000	4475	1475	90 %
Grande	10	20	10	3000	4817	1817	100 %
Grande	70	20	10	3000	4381	1381	90 %

cada pedido recibe a lo sumo la biomasa que solicitó. La biomasa producida por encima de esa cantidad no infla el cumplimiento, sino que se contabiliza aparte como *venta directa*, el excedente que el productor comercializa por canal directo, que es la columna homónima de la tabla. Bajo esa lectura, el primer resultado es que el déficit se mantuvo nulo en Pequeña y Mediana a lo largo de las tres semillas evaluadas en cada escenario, lo que evidencia que el sistema integrado sostiene el abastecimiento completo en esas dos escalas con independencia de la trayectoria biológica muestreada. En el escenario Grande, sin embargo, el resultado no es uniforme entre semillas: solo la semilla 10 alcanzó el 100 % de cumplimiento, mientras que las semillas 42 y 70 llegaron ambas al 90 %, es decir, el déficit apareció en dos de las tres simulaciones. Esta sensibilidad a la semilla, ausente en las escalas menores, indica que el margen que protege la continuidad del abastecimiento se estrecha a medida que crece la escala de la asociación, al punto de que el déficit resultó ser el caso más frecuente, no la excepción, en la escala Grande. Esto es consistente con que cada estanque admite un único ciclo de siembra-cosecha por horizonte, de modo que, ante un número creciente de pedidos compitiendo por un mismo conjunto de estanques, alguna trayectoria biológica particular puede dejar sin cubrir una fecha puntual, incluso cuando la producción agregada del escenario supera ampliamente la demanda total.

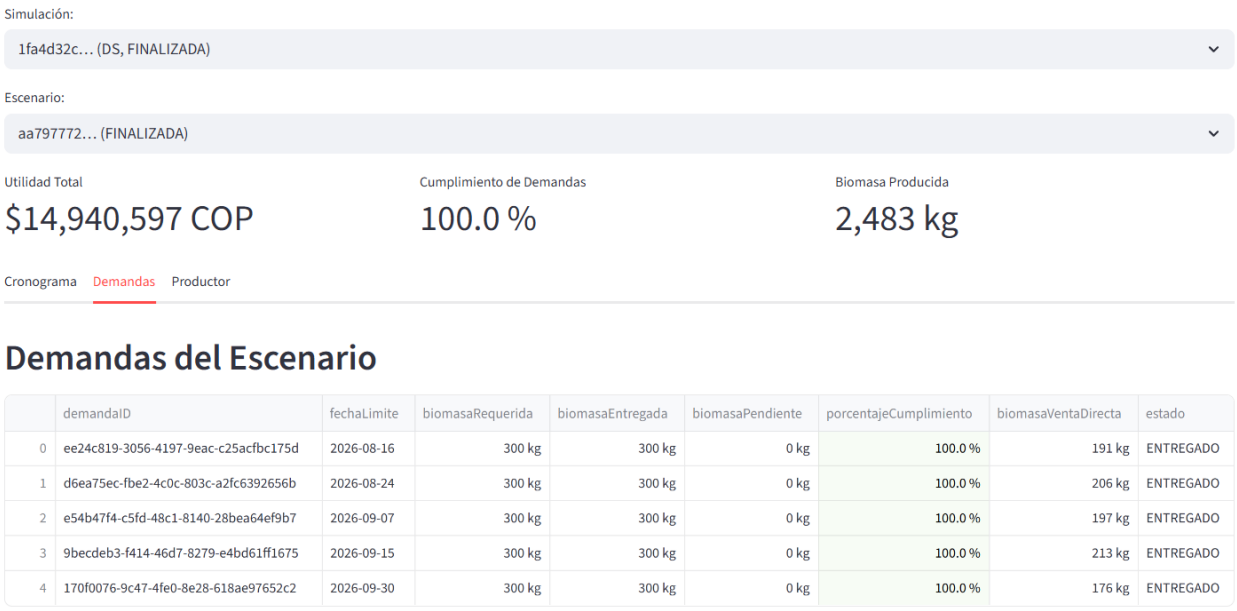
El segundo resultado es que la venta directa en kilogramos sí crece con el tamaño de la asociación, aunque no en la misma proporción que la demanda: el escenario Pequeña se ubica entre 504 y 549 kg según la semilla, Mediana entre 890 y 983 kg, y Grande entre 1381 y 1817 kg. En términos absolutos los tres rangos quedan claramente separados y crecientes, pero conviene notar que en el escenario Grande multiplica por 4 la capacidad instalada de Pequeña ($\Gamma = 20$ frente a $\Gamma = 5$) y solo entre 2,5 y 3,6 veces su venta directa, es decir, el excedente crece más lento que la escala del sistema. Esto se debe a la capacidad de cada estanque ($K_i = 500\,\text{kg}$), que es mayor al tamaño individual de los pedidos, de modo que una cosecha genera un excedente frente al pedido asignado, y los tres escenarios mantienen aproximadamente la misma proporción $\Gamma : \Psi$ (cercana a $2 : 1$), por lo que ninguno dispone de mayor margen de maniobra para mitigar este efecto.

5.5.4 Viabilidad Económica del Plan Ejecutado

El cumplimiento de la demanda es condición necesaria, pero no suficiente para validar la viabilidad de un plan estratégico, con lo cual un plan que entregue toda la biomasa requerida, pero a un costo superior al ingreso generado carece de valor práctico para la asociación. Por esta razón, el sistema integra un *dashboard* con estadísticas económicas que, una vez concluido el horizonte de simulación, consolida sobre el plan ejecutado los indicadores financieros y de cumplimiento fundamentales, que incluyen, la utilidad total neta, el porcentaje de cumplimiento de demandas y la biomasa total producida, junto con el detalle de entrega por pedido. La Figura 33 presenta estos indicadores para el escenario Mediana del Bloque D ($\Gamma = 10$, $\Psi = 5$, demanda total de 1500 kg).

Figura 33

Estadísticas económicas del plan ejecutado ($\Gamma = 10$, $\Psi = 5$, escenario Mediana)



	demandaID	fechaLimite	biomasaRequerida	biomasaEntregada	biomasaPendiente	porcentajeCumplimiento	biomasaVentaDirecta	estado
0	ee24c819-3056-4197-9eac-c25acfbc175d	2026-08-16	300 kg	300 kg	0 kg	100.0 %	191 kg	ENTREGADO
1	d6ea75ec-fbe2-4c0c-803c-a2fc6392656b	2026-08-24	300 kg	300 kg	0 kg	100.0 %	206 kg	ENTREGADO
2	e54b47f4-c5fd-48c1-8140-28bea64ef9b7	2026-09-07	300 kg	300 kg	0 kg	100.0 %	197 kg	ENTREGADO
3	9becdeb3-f414-46d7-8279-e4bd61ff1675	2026-09-15	300 kg	300 kg	0 kg	100.0 %	213 kg	ENTREGADO
4	170f0076-9c47-4fe0-8e28-618ae97652c2	2026-09-30	300 kg	300 kg	0 kg	100.0 %	176 kg	ENTREGADO

Los tres indicadores globales del panel confirman la viabilidad del plan. La utilidad total neta alcanzó \$14 940 597 COP, lo que corresponde a la suma de la utilidad de cada uno de los estanques involucrados. El cumplimiento de demandas fue del 100 %, de modo que los cinco pedidos registran estado `ENTREGADO` con biomasa pendiente de 0 kg y la biomasa total producida

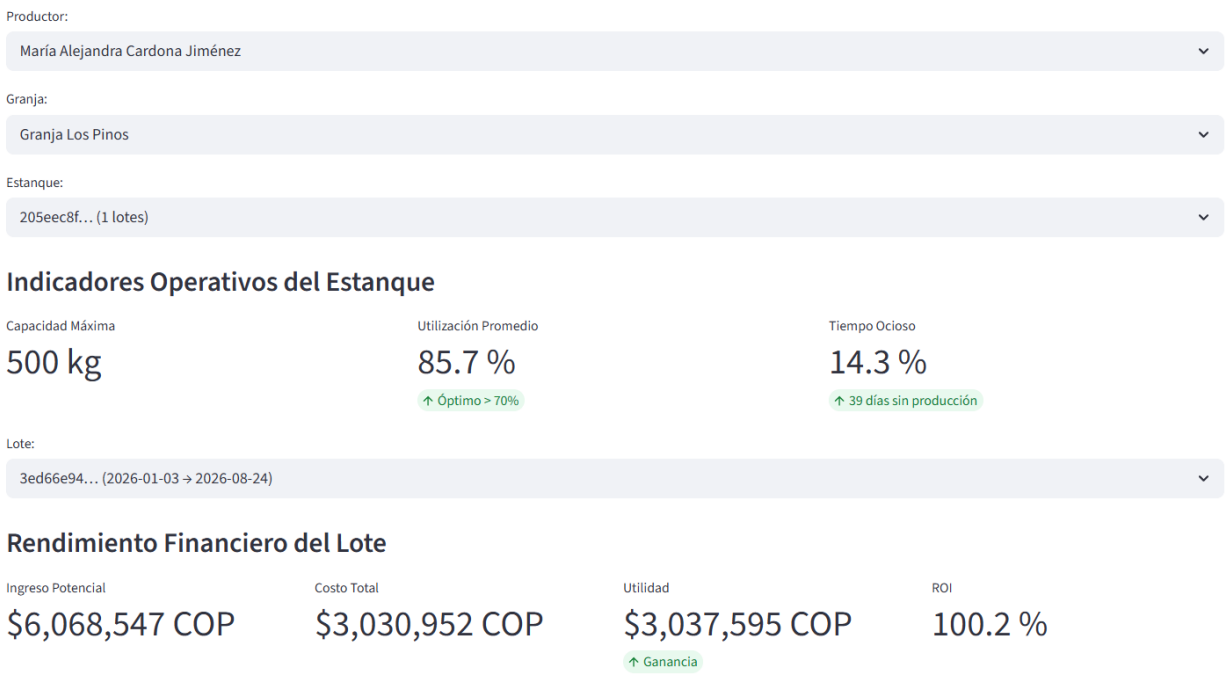
fue de 2483 kg frente a 1500 kg demandados; los 983 kg restantes constituyen el exceso total que se comercializa como venta directa.

El detalle por pedido muestra una venta directa relativamente uniforme de los cinco pedidos, entre 176 y 213 kg adicionales sobre cada demanda de 300 kg, correspondiendo el menor excedente al pedido más tardío (30 de septiembre) y el mayor al del 15 de septiembre, no al más próximo ni al más lejano. Esto sugiere que, a esta escala, el reparto del excedente entre pedidos responde más a la disponibilidad puntual de estanques listos para cosechar en cada fecha que a un efecto sistemático de la lejanía del plazo.

A nivel de estanque individual, la Figura 34 presenta los indicadores operativos y financieros para un lote representativo del escenario.

Figura 34

Indicadores operativos y rendimiento financiero por estanque (lote representativo, escenario Mediana)



La utilización promedio del estanque fue de 85,7 %, superando holgadamente el umbral óptimo de 70 % definido por las buenas prácticas piscícolas, a pesar de un período ocioso de 39

días (14,3 %) sin producción, calculado sobre el horizonte efectivamente ejecutado hasta la fecha del último pedido. El retorno sobre la inversión (ROI) del lote alcanzó el 100,2 %, derivado de un ingreso potencial de \$6 068 547 COP frente a un costo total de \$3 030 952 COP, con una utilidad neta por lote de \$3 037 595 COP.

6. Conclusiones

La piscicultura colombiana, practicada mayoritariamente por productores de pequeña y mediana escala que operan de forma aislada, enfrenta una brecha estructural de productividad que les impide, por sí solos, garantizar el suministro constante que exigen los contratos de gran volumen. Frente a este planteamiento, el presente trabajo abordó la pregunta de cómo coordinar los ciclos de producción de un colectivo de piscicultores asociados, mediante la construcción y validación de un artefacto computacional bajo el paradigma de la Investigación Basada en el Diseño. Dicho artefacto extendió hacia la escala asociativa el modelo de simulación y gestión predictiva para estanques individuales desarrollado previamente por el grupo SIMON, articulando tres subsistemas —simulación, pronóstico y optimización— cuyo cumplimiento se contrasta a continuación, objetivo por objetivo, contra la evidencia presentada en el Capítulo 5.

Se adaptó la arquitectura del módulo de simulación bajo los preceptos del Diseño Basado en el Dominio y una arquitectura hexagonal, lo que permitió declarar y coordinar un número creciente de estanques mediante una única llamada, sin que el operador deba conocer la distribución interna de los ciclos productivos. La Sección 5.2.2 evidenció que esta capacidad es también escalable en su comportamiento computacional, pues el tiempo de respuesta crece de forma prácticamente lineal con el número de estanques, sin degradación superlineal, hasta el límite impuesto por la memoria disponible para la proyección Monte Carlo (entre $\Gamma = 20$ y $\Gamma = 40$). Con ello, el primer objetivo específico se cumplió en su doble exigencia de autonomía y escalabilidad.

Se ajustó el modelo de red neuronal recurrente LSTM para predecir el incremento de peso

y la ración requerida a partir de los datos generados por el simulador de referencia. El análisis de sensibilidad al tamaño del *dataset* (Tabla 10) determinó que un volumen de 400 estanques simulados era necesario para que el error de predicción normalizado descendiera. Más aún, la evaluación en tres niveles crecientes de exigencia (Tablas 11 y 12) confirmó que dicho desempeño se sostiene ante estanques generados con semillas aleatorias nunca observadas durante el entrenamiento, con lo cual el modelo no solo aprendió la dinámica biológica del simulador, sino que la generalizó, condición indispensable para que el optimizador decida sobre proyecciones confiables.

Se implementó un módulo de optimización basado en un modelo de Programación Lineal Entera Mixta resuelto mediante Gurobi, capaz de generar, a partir del pronóstico del modelo neuronal, un cronograma de siembras y cosechas que coordina los ciclos productivos de la asociación. La producción entregada acumulada se mantuvo superpuesta a la demanda a lo largo de todo el horizonte evaluado, sin períodos de desabastecimiento. Adicionalmente, el diseño experimental delimitó las condiciones de validez del plan generado: un horizonte de planificación de al menos 250 días para que el ciclo biológico alcance el peso comercial exigido (Bloque C), y una penalización por déficit $\epsilon = 10$ que busca garantizar el abastecimiento íntegro sin generar excedente injustificado (Bloque E). De esta manera, el tercer objetivo específico se cumplió al producir un plan viable y, además, uno cuyos límites de aplicabilidad quedaron explícitamente caracterizados.

Se validó el sistema de planificación integrada mediante un diseño experimental por bloques que contempló, desde el caso de control más simple hasta asociaciones de escala creciente, diversas combinaciones de estanques, pedidos y horizontes de planificación (Sección 5.5). El Bloque D mostró que, en la mayoría de los casos evaluados (78 % de las nueve corridas, combinando las tres escalas y las tres semillas), el sistema priorizó cumplir la demanda generando excedente antes que incumplirla, alcanzando un cumplimiento del 100 % en las escalas pequeña y mediana de forma consistente. Únicamente en la escala grande, y solo en dos de las tres semillas evaluadas, apareció déficit, lo cual es atribuible a la naturaleza estocástica del crecimiento del pez combinada con la restricción de que cada estanque admite un único ciclo de producción por horizonte, de modo que una trayectoria biológica particular puede dejar sin cubrir una fecha puntual pese a que

la producción agregada del escenario supere ampliamente la demanda total. Con ello, el cuarto objetivo específico se cumplió al demostrarse que los planes estratégicos generados son factibles desde la perspectiva operativa y, a la vez, rentables desde la perspectiva económica del productor.

En conjunto, la evidencia presentada permite afirmar que el objetivo general del proyecto se cumplió: el sistema desarrollado da soporte efectivo a la planificación productiva de una asociación de piscicultores, coordinando de forma autónoma, escalable y económicamente viable los ciclos de múltiples estanques para atender una demanda periódica de mercado. Más allá de la validación técnica, este resultado adquiere su verdadero significado en el propósito que motivó el proyecto desde su planteamiento, el cual es ofrecer a productores de pequeña y mediana escala, que de forma aislada no podrían cumplir con contratos de gran volumen, un mecanismo concreto para hacerlo de manera conjunta. Al garantizar un suministro constante y predecible, el sistema fortalece el poder de negociación colectivo de la asociación y sienta una base tecnológica tangible sobre la cual el sector piscícola colombiano puede consolidar una asociatividad funcional: más que una alternativa de mejora, un mecanismo de supervivencia y escalamiento competitivo frente a los grandes productores. Las líneas de trabajo futuro presentadas a continuación delimitan, con la misma honestidad con la que se contrastaron estos resultados, hasta dónde llega el alcance actual de este primer artefacto y qué tendría que extenderse para consolidarlo como una herramienta real de apoyo a la decisión.

6.1 Trabajo Futuro

El sistema desarrollado es un primer artefacto funcional, no un punto de llegada, de modo que varias decisiones de alcance tomadas en el Capítulo 4 abren líneas de extensión concretas. Las siguientes van de lo más incremental —ampliar variables y modelos de dominio ya existentes— a lo más disruptivo —sustituir la fuente de datos del modelo predictivo o el paradigma de optimización mismo—, y cierran con una línea transversal que reorienta el artefacto hacia el usuario final.

6.1.1 Parametrización de las Condiciones Iniciales de Siembra

El sistema desarrollado opera hoy sobre dos supuestos respecto al arranque de cada ciclo productivo. La primera es que la siembra se realiza con alevines del peso mínimo biológico de 10 g documentado para la tilapia. Lo segundo es que la cantidad sembrada corresponde a la capacidad de carga recomendada del estanque, derivada de su volumen y densidad de siembra según las buenas prácticas del sector (Merino Archila et al., 2006). Ambos supuestos son deliberados y consistentes con el carácter investigativo del artefacto, pues fijan un punto de partida normativo —la condición ideal de siembra bajo las recomendaciones técnicas— que permite caracterizar el comportamiento del sistema sin introducir la variabilidad de las restricciones particulares de cada productor. Bajo estas condiciones estándar, el sistema es plenamente funcional y los planes generados son válidos. No obstante, un productor real puede enfrentar circunstancias que se apartan de este punto de partida, pues es posible que no disponga de la inversión inicial para sembrar la totalidad de alevines que el sistema recomienda, o que en el mercado de alevines no consiga individuos del peso exacto de 10 g, sino de una talla distinta. Una extensión natural del sistema consiste en exponer el peso inicial y la cantidad de siembra como parámetros configurables por el productor al iniciar cada ciclo, en lugar de asumirlos como valores fijos. La arquitectura del dominio ya contempla el mecanismo necesario para ello, pues el optimizador modela los estanques ya ocupados como lotes heredados con un estado inicial arbitrario de peso y ración; formalizar la siembra parametrizada como un caso general de esa misma capacidad no exigiría rediseñar el modelo, sino generalizar la interfaz de entrada que hoy fija dichas condiciones.

6.1.2 Incorporación de Variables Geoespaciales

Una limitación del sistema desarrollado es que la variable de ubicación geográfica de las granjas no incide actualmente en los parámetros de simulación más allá de la agrupación estructural de estanques por granja. Sin embargo, la literatura técnica (Merino Archila et al., 2006) documenta

que la distancia a centros de distribución, la calidad del suelo y la disponibilidad hídrica son variables que afectan directamente los costos operativos y la viabilidad del proyecto productivo. Una extensión futura del sistema podría incorporar estas variables geoespaciales como parámetros de entrada para el optimizador, permitiendo generar planes de producción diferenciados según la ubicación relativa de cada granja dentro de la asociación.

6.1.3 Extensión del Modelo de Demanda y la Logística de Entrega

El módulo comercial define hoy al `Contrato` de forma simplificada, como una serie de pedidos con fechas fijas y periódicas, mientras que cada `Pedido` se resuelve, al finalizar, en uno de dos únicos desenlaces terminales —`ENTREGADO` o `VENCIDO`—, sin estados intermedios de cumplimiento parcial. Esta simplificación es adecuada para validar el sistema, pero deja fuera matices comerciales reales, tales como contratos con tolerancias de fecha o de tamaño en vez de valores fijos, penalizaciones graduales por incumplimiento parcial en lugar del corte binario actual, o una prioridad explícita por cliente que el optimizador pueda ponderar frente a la urgencia de la fecha límite. Enriquecer ambas representaciones con estas funcionalidades permitiría que el modelo de asignación arbitre no solo por fecha, sino también por el valor comercial relativo de cada compromiso. En la misma línea, el optimizador supone hoy que cada estanque se cosecha exactamente en la fecha en que vence el pedido que abastece, pues las rutas de producción solo se generan para los instantes de cosecha que coinciden con una demanda vigente, sin contemplar ningún almacenamiento intermedio entre la cosecha y la entrega. Este supuesto omite la logística real de comercialización, en la que la cadena de frío y la gestión de inventarios permiten desacoplar el momento de la cosecha del momento de la entrega, cosechando anticipadamente y conservando el producto, o consolidando varias cosechas para atender un mismo pedido. Incorporar un modelo de inventario con sus costos y límites de conservación asociados le otorgaría al optimizador un grado de libertad adicional para programar las cosechas según la disponibilidad biológica de cada estanque y no únicamente según el calendario de entregas.

6.1.4 Generalización a Otras Especies

El simulador de dinámica de sistemas y el modelo LSTM están calibrados exclusivamente para la tilapia (*Oreochromis spp.*), cuyos límites biológicos de siembra y peso máximo se tomaron directamente de la literatura técnica (Merino Archila et al., 2006). Extender el sistema a otras especies acuícolas relevantes para el sector colombiano exigiría recalibrar estas constantes biológicas y regenerar el *dataset* sintético de entrenamiento bajo los parámetros propios de la especie objetivo, para luego reentrenar el modelo LSTM sobre esos datos. Dado que la arquitectura del motor de simulación ya trata dichas constantes como parámetros del dominio y no como valores codificados en la lógica de negocio, esta extensión no exige rediseñar el sistema, sino recalibrarlo.

6.1.5 Ampliación de Variables del Simulador y del Optimizador

Tanto el simulador de dinámica de sistemas como el optimizador deciden hoy con un conjunto deliberadamente acotado de variables: temperatura y ración para la proyección biológica, biomasa y fechas para la asignación. Esta acotación fue necesaria para mantener el problema tratable dentro del alcance de este trabajo, pero deja fuera variables con incidencia real sobre una producción piscícola, como la calidad del agua (oxígeno disuelto, pH, amonio), episodios de mortalidad no determinista asociados a enfermedades, o la variabilidad del precio de mercado del producto final. Incorporar estas variables, tanto en la proyección biológica como en la función objetivo del optimizador, permitiría que el plan estratégico reaccione a condiciones de producción más cercanas a las de una asociación real.

6.1.6 Algoritmos Genéticos como Alternativa Metaheurística

El modelo MILP resuelto con Gurobi garantiza una solución óptima o cercana a la óptima, pero esa garantía tiene un costo, ya que el Bloque A (Sección 5.2.2) mostró que su tiempo de

respuesta crece con el número de estanques, con un límite práctico identificado en $\Gamma = 40$ por agotamiento de memoria; para asociaciones que requieran superar ese límite, una alternativa consiste en explorar algoritmos genéticos u otras metaheurísticas poblacionales, las cuales renuncian a la garantía de optimalidad exacta a cambio de tiempos de cómputo más predecibles y menos sensibles al crecimiento combinatorio del espacio de soluciones.

6.1.7 Aprendizaje a partir de Datos Reales de Producción

El modelo LSTM se entrenó con datos sintéticos generados por el simulador de dinámica de sistemas, y no con mediciones tomadas directamente de estanques en operación. Esta decisión fue metodológica y respondió a la conveniencia investigativa del proyecto, pues el simulador ofrece un entorno controlado, reproducible y libre de la variabilidad de un despliegue de campo, además de suplir la ausencia de un conjunto de datos reales instrumentado para la asociación objetivo. Su contrapartida es que la fidelidad del modelo predictivo queda acotada por la fidelidad del propio simulador, es decir, el LSTM aprende la dinámica biológica tal como el simulador la representa, no tal como ocurre en el estanque físico. Una línea de trabajo futuro consiste en alimentar el aprendizaje con datos reales capturados en los estanques mediante sensores de bajo costo y muestreos biométricos periódicos —en la línea de las plataformas de instrumentación por Internet de las Cosas ya revisadas en el Capítulo 2—, sustituyendo progresivamente el *dataset* sintético por observaciones de campo. Más aún, este esquema habilitaría un aprendizaje continuo en el que el modelo se reentrena y refina con los datos que la asociación acumula a lo largo del tiempo, cerrando la brecha entre la dinámica simulada y la observada, y ajustándose a las condiciones particulares de cada granja y cada temporada.

6.1.8 Planificación Estratégica mediante Aprendizaje por Refuerzo

El enfoque actual resuelve el plan estratégico recalculando el modelo MILP completo en cada invocación del horizonte deslizante, sin amortizar ningún aprendizaje entre escenarios o entre iteraciones sucesivas de una misma asociación. Trondsen y Hansen (2021) exploraron precisamente esta alternativa para un problema de programación de producción de salmón estructuralmente análogo al de este trabajo, modelándolo como un Proceso de Decisión de Markov y resolviéndolo mediante programación dinámica aproximada y aprendizaje por refuerzo, en contraste con un modelo de programación entera mixta usado como referencia. Sus resultados son honestos sobre el estado del arte de esta alternativa: la mejor política aprendida alcanzó apenas cerca del 80 % de la utilidad del modelo exacto en instancias pequeñas, un desempeño que los propios autores consideran insuficiente para producción real, aunque sostienen que el enfoque tiene un potencial considerable aún por desarrollar. Esto sugiere que una futura política de aprendizaje por refuerzo para este sistema no debería reemplazar de entrada al optimizador MILP, sino evaluarse contra él como línea base, del mismo modo en que aquí se usó al MILP determinista para calibrar el comportamiento del sistema completo.

6.1.9 Del Artefacto de Investigación al Producto para el Usuario Final

Todas las líneas anteriores extienden las capacidades técnicas del sistema, pero la más transversal de las extensiones es de naturaleza distinta y atañe a su destinatario. El análisis funcional ya precisó que los casos de uso del sistema no partieron de la experiencia de un productor real, sino de la especificación técnica del dominio y de los objetivos de investigación. En consecuencia, el resultado es un artefacto de validación conceptual, orientado netamente a demostrar la viabilidad de coordinar los ciclos productivos de una asociación, y no un producto de software concebido desde la experiencia del productor. El paso siguiente para consolidarlo como una herramienta real de apoyo a la decisión consiste en aproximarse a las asociaciones y productores del sector mediante

entrevistas e historias de usuario, que permitan comprender sus flujos de trabajo, restricciones y expectativas reales, y traducir ese conocimiento en un software orientado al usuario final, con interfaces, mecanismos de autenticación y experiencias de uso diseñados a partir de sus necesidades y no únicamente de las capacidades técnicas ya implementadas.

Referencias Bibliográficas

Agencia de Desarrollo Rural. (s.f.-a). *Comercialización*. ADR. https://www.adr.gov.co/atencion-y-servicios-a-la-ciudadania/comercializacion/

Agmata, A., & Guðmundsson, S. (2025). Convolutional-LSTM approach for temporal catch hotspots (CATCH): an AI-driven model for spatiotemporal forecasting of fisheries catch probability densities [En línea]. *Biological Methods and Protocol*. https://pmc.ncbi.nlm.nih.gov/articles/PMC12203189/

AquaManager. (2025). *AquaManager* [[Foro/Blog de discusión]]. https://www.aqua-manager.com

Autoridad Nacional de Acuicultura y Pesca. (2023, 6 de diciembre). Por medio de la cual se establece la clasificación de los acuicultores comerciales en Colombia [En línea]. https://aunap.gov.co/download/por-medio-de-la-cual-se-establece-la-clasificacion-de-los-acuicultores-comerciales-en-colombia-de-acuerdo-con-la-actividad-el-sistema-y-el-volumen-de-produccion-y-se-deroga-la-resolucion-no/

Autoridad Nacional de Acuicultura y Pesca. (2024, 29 de noviembre). *El sector de la pesca y la acuicultura crece un* 18,2 % *en el último año*. AUNAPColombia. https://aunap.gov.co/el-sector-de-la-pesca-y-la-acuicultura-crece-un-18-2-en-el-ultimo-ano/

Autoridad Nacional de Acuicultura y Pesca. (s.f.). *Informe final de la caracterización del estado actual de la acuicultura en los departamentos de Atlántico, Bolívar, Cesar y Córdoba* [En línea]. Autoridad Nacional de Acuicultura y Pesca. https://www.aunap.gov.co/documentos/biblioteca/Informe-final-de-caracterizacion-Atlantico-Bolivar-Cesar-Cordoba-AUNAP.pdf

Baskerville, R., Pries-Heje, J., & Venable, J. R. (2026). MEDS: Methodology for Evaluation in Design Science. *European Journal of Information Systems*, *0*(0), 1-18. https://doi.org/10.1080/0960085X.2026.2627280

Brownlee, J. (2019). *A Gentle Introduction to Sparse Matrices for Machine Learning* [En línea]. https://machinelearningmastery.com/sparse-matrices-for-machine-learning/

B.V., A. (2026). *Implementing a Model with a Rolling Horizon* [En línea]. https://documentation.aimms.com/language-reference/advanced-language-components/time-based-modeling/implementing-a-model-with-a-rolling-horizon.html

Carravilla, M. A., & Oliveira, J. F. (2013). Operations Research in Agriculture: Better Decisions for a Scarce and Uncertain World [En línea]. *Agris on-line Papers in Economics and Informatics*. https://www.researchgate.net/publication/260465710_Operations_Research_in_Agriculture_Better_Decisions_for_a_Scarce_and_Uncertain_World

Castro Castro, J. J., & Zambrano Urbina, M. R. (2012). *AMBIENTE SOFTWARE INTEGRADO POR UN JUEGO PARA TELÉFONOS MÓVILES, UN SITIO WEB Y UNA APLICACIÓN PARA COMPUTADOR PERSONAL, PARA EL APRENDIZAJE Y TOMA DE DECISIONES VERSIÓN 2.0* (Tesis de pregrado) [En línea]. Universidad Industrial de Santander. https://noesis.uis.edu.co/handle/20.500.14071/27070

Chim, L., Home, M., & Shaw, S. (s.f.). *Aquaculture economics: identification and management of production cost* (Special Publication No. 12) [En línea]. Bredene, Belgica. https://archimer.ifremer.fr/doc/00000/2478/2108.pdf

Cockburn, A. (2005). *The Hexagonal (Ports and Adapters) Architecture* [En línea]. https://fabiofumarola.github.io/nosql/readingMaterial/Evans03.pdf

Cuadros Sanabria, A. J., & Rangel Flórez, D. K. (2021). *Aplicación móvil orientada al mejoramiento de la valoración del producto agrícola y la toma de decisiones en la comercialización* (Tesis de pregrado) [En línea]. Universidad Industrial de Santander. https://noesis.uis.edu.co/handle/20.500.14071/41574

Cuisinier, É., Lemaire, P., Penz, B., Ruby, A., & Bourasseau, C. (2022). New rolling horizon optimization approaches to balance short-term and long-term decisions: An application to energy planning [En línea]. *Energy*. https://doi.org/https://doi.org/10.1016/j.energy.2021.122773

Departamento Nacional de Planeación. (2024). *Misión para la Transformación del campo. Propuesta para desarrollar un modelo eficiente de comercialización y distribución de productos* [En

línea]. DNP. Bogotá D.C. https://colaboracion.dnp.gov.co/CDT/Agriculturapecuariofore stal%20y%20pesca/Propuesta%20para%20desarrollar%20un%20modelo%20eficiente% 20de%20Comercializaci%C3%B3n%20y%20Distribuci%C3%B3n%20de%20Productos. pdf

Despotovic, M., Nedic, V., Despotovic, D., & Cvetanovic, S. (2015). Review and statistical analysis of different global solar radiation sunshine models. *Renewable and Sustainable Energy Reviews*, *52*, 1869-1880. https://doi.org/10.1016/j.rser.2015.08.035

Dirección de Comercialización de la Agencia de Desarrollo Rural. (s.f.-b). *Guía Técnica: Requisitos para acceso a mercados agroalimentarios* [En línea]. Agencia de Desarollo Rural y MinAgricultura. https://www.adr.gov.co/wp-content/uploads/2021/07/Requisitos-para-Acceso-a-Mercados-Agroalimentarios.pdf ISBN 978-56571-1-3

Eby, K. (2016). When to Choose Waterfall Project Management Over Agile [En línea]. *smartsheet*. https://es.smartsheet.com/when-choose-waterfall-project-management-over-agile

Ernst, D. H., Bolte, J. P., & Nath, S. S. (2000). Modeling Aquaculture Production Systems: A Review [En línea]. *Aquacultural Engineering*. https://www.sciencedirect.com/science/article/abs/pii/S0144860900000455. ID: 271324

Evans, E. (2003a). *Diseño basado en dominios: abordar la complejidad en el corazón del software* [En línea]. Addison Wesley. https://es.scribd.com/document/982362547/Espanol-Eric-Evans-2003-Domain-Driven-Design-Tackling-Complexity-in-the-Heart-of-Software. ISBN 0-321-12521-5

Evans, E. (2003b). Domain-Driven Design: Tackling Complexity in the Heart of Software [En línea]. https://fabiofumarola.github.io/nosql/readingMaterial/Evans03.pdf

FastAPI. (s.f.). *FastAPI* [En línea]. https://fastapi.tiangolo.com/

Federación Colombiana de Acuicultores. (s.f.). *Revista FEDEACUA*. FEDEACUA. Consultado el 9 de octubre de 2025, desde https://fedeacua.org/page/revista

Food and Agriculture Organization of the United Nations. (2023, 10 de noviembre). *Fisheries and Aquaculture Country Profiles. Colombia*. FAO. https://www.fao.org/fishery/en/facp/col

Gandh D, R., V P, H., Haq K P, R. A., & Bhide, A. (2024). Attention-driven LSTM and GRU deep learning techniques for precise water quality prediction in smart aquaculture [En línea]. *Aquaculture International*. https://www.researchgate.net/publication/382109876_Attention-driven_LSTM_and_GRU_deep_learning_techniques_for_precise_water_quality_prediction_in_smart_aquaculture

Gudrun Wallentin. (s.f.). *UNIGIS module: Spatial Simulation*. UNIGIS. https://unigis-salzburg.github.io/Opt_Spatial-Simulation/system-dynamics.html

Guerra González, L. E., Rios Porras, C. A., & Maestre Gongora, G. P. (2011). *Ambiente software integrado por un juego para teléfonos móviles, un sitio web y una aplicación para computador personal, para el aprendizaje y toma de decisiones* (Tesis de pregrado) [En línea]. Universidad Industrial de Santander. https://noesis.uis.edu.co/handle/20.500.14071/25040

Guio Roa, A., & Vásquez Alcocer, B. (2024). *AMBIENTE VIRTUAL DE APRENDIZAJE PARA LA PRODUCCIÓN Y MERCADEO PISCÍCOLA PESCO VERSIÓN 4.0* (Tesis de pregrado) [En línea]. Universidad Industrial de Santander. https://noesis.uis.edu.co/handle/20.500.14071/15946

Gurobi Optimization. (s.f.). *Gurobi Optimizer Reference Manual* [En línea]. https://docs.gurobi.com/projects/optimizer/en/current/index.html

Harrison, R. L. (2010). Introduction To Monte Carlo Simulation [En línea]]. https://pmc.ncbi.nlm.nih.gov/articles/PMC2924739/

Kavlakoglu, E. (s.f.). *Agile vs. Waterfall*. IBM. https://www.ibm.com/think/topics/agile-vs-waterfall

Kunapinun, A., Fairman, W., S. Wills, P., & Ouyang, B. (2025). Integrating Bi-LSTM and physics constraint models for improved biomass and water quality prediction in the aquaculture systems [En línea]. *Machine Learning from Challenging Data 2025*. https://www.researchgate.net/publication/392213307_Integrating_Bi-LSTM_and_physics_constraint_models_for_improved_biomass_and_water_quality_prediction_in_the_aquaculture_systems

Laribee, D. (2009). Best Practice - An Introduction To Domain-Driven Design. *MSDN Magazine*, *24*(02). https://learn.microsoft.com/en-us/archive/msdn-magazine/2009/february/best-practice-an-introduction-to-domain-driven-design

Lucia, S., & Fiedler, F. (2023). *Basics of model predictive control* [En línea]. https://www.do-mpc.com/en/latest/theory_mpc.html

M., M., J., A. K., S., S., M., J., P., V., & J., S. D. (2023). Dynamic modelling of coastal aquaculture systems: A Review [En línea]. *Aquatic Ecosystem Health and Management*, *26*(3). https://doi.org/10.14321/aehm.026.03.40

MacDonald, A., Serpetti, N., & Franco, S. C. (2024). Optimising seafood nutritional value and environmental sustainability in aquaculture through a novel integrated modelling tool applicable to IMTA and monoculture [En línea]. *Aquaculture*. https://doi.org/https://doi.org/10.1016/j.aquaculture.2024.741046

Maldonado Vargas, A. (2018). Ambiente software para el aprendizaje y toma de decisiones sobre la producción y mercadeo piscícola versión 3.0. *Andrade Sosa, Hugo Hernando*.

Martin, R. C. (2008). Clean Code: A Handbook of Agile Software Craftsmanship [En línea]. https://dl.acm.org/doi/10.5555/1388398

Max Schwenzer, T. B., Muzaffer Ay, & Abel, D. (2021). Review on model predictive control: an engineering perspective [En línea]. *The International Journal of Advanced Manufacturing Technology*. https://doi.org/https://doi.org/10.1007/s00170-021-07682-3

McCarl, B. A., & Spreen, T. H. (2020). *APPLIED MATHEMATICAL PROGRAMMING USING ALGEBRAIC SYSTEMS* [En línea]. https://agecoresearch.tamu.edu/mccarl/wp-content/uploads/sites/4/2023/12/thebook.pdf

Merabet, A., & Elsaraiti, M. (2021). A Comparative Analysis of the ARIMA and LSTM Predictive Models and Their Effectiveness for Predicting Wind Speed [En línea]. *Energies*, 1-16. https://www.researchgate.net/publication/355429916_A_Comparative_Analysis_of_the_ARIMA_and_LSTM_Predictive_Models_and_Their_Effectiveness_for_Predicting_Wind_Speed

Merino Archila, M. C., Salazar Ariza, G., & Gómez León, D. (2006). *Guía Práctica de Piscicultura en Colombia* [En línea]. Instituto Nacional de Desarrollo Rural (INCODER). Bogotá D.C. https://www.aunap.gov.co/documentos/OGCI/Guia-Practica-de-Piscicultura-en-Colombia.pdf ISBN 958-3-8172-1

Ministerio de Agricultura y Desarrollo Rural de Colombia. (2025, 21 de agosto). *PIB agropecuario crece* 3,8 % *en el segundo trimestre del año y continúa consolidando al campo como motor de la economía del país*. MinAgricultura. https://www.minagricultura.gov.co/noticias/Paginas/PIB-agropecuario-crece-3-8-en-el-segundo-trimestre-del-ano-y-continua-consolidando-campo-como-motor-de-la-economia-pais.aspx

Narváez-Rodriguez, Cristhian Camilo. (2014). Asociaciones y Cooperativas Rurales: factores internos y externos que influyen en su estabilidad y eficiencia. Una reflexión sobre el caso de Viotá, Cundinamarca [En línea]. *Cooperativismo y Desarrollo*, *22*(104), 63-81. https://revistas.ucc.edu.co/index.php/co/article/view/971

Neil J. Rowan. (2023). The role of digital technologies in supporting and improving fishery and aquaculture across the supply chain – Quo Vadis? [En línea]. *Aquaculture and Fisheries*. https://www.sciencedirect.com/science/article/pii/S2468550X22001010. ID: 315459

Oracle Corporation. (s.f.). *JDK 21 Documentation* [En línea]. https://docs.oracle.com/en/java/javase/21/

Özkan, O., Babur, Ö., & van den Brand, M. (s.f.). Domain-Driven Design in Software Development: A Systematic Literature Review on Implementation, Challenges, and Effectiveness [En línea]. *Journal of Systems and Software*. https://arxiv.org/html/2310.01905v4

Patricia Bonilla, Sara. (2021). *Análisis de Oportunidades de Mercado* [En línea]. Programa Global de Acceso a Mercados (GMAP), Organización de las Naciones Unidas para el Desarrollo Industrial, Norad, Ministerio de Comercio, Industria y Turismo y Colombia Productiva. https://www.colombiaproductiva.com/PTP/media/documentos/Sectoriales/PuntoAqua/Capacitaciones/Analisis-de-Mercados-Acuicultura-GMAP-Colombia-VNoviembre-4.pdf

Penrose-Buckley, Chris. (2007). *Organizaciones de Productores: Guía para el desarrollo de empresas rurales colectivas* [En línea]. Intermón Oxfam. https://base.socioeco.org/docs/organizacion_de_productores.pdf ISBN 978-84-8452-329-1

Piacentini, C., Castro, M. P., Cire, A. A., & Beck, J. C. (2018). Compiling Optimal Numeric Planning to Mixed Integer Linear Programming [En línea]. https://doi.org/https://doi.org/10.1609/icaps.v28i1.13919

Piamba-Mamian, T. M., Zambrano, L. E., Montaño-Rúales, L. A., & Rojas Gonzales, F. A. (2021). Implementación de un sistema de monitoreo IoT aplicado a una piscicultura de trucha [En línea]. *Informador técnico*. https://dialnet.unirioja.es/servlet/articulo?codigo=7868820

Postman Inc. (s.f.). *Postman documentation overview* [En línea]. https://learning.postman.com/docs/introduction/overview

Prieto Mejía, S. (2025). *Dinámica de Sistemas* [En línea]. Ediciones Unimagdalena. https://www-digitaliapublishing-com.bibliotecavirtual.uis.edu.co/a/175927. ISBN 9789587468373

Programa Global de Acceso a Mercados. (s.f.). *La asociatividad en las Cadenas de Valor de la Acuicultura de Camarón en Tumaco y Tilapia en el Huila* [En línea]. Colombia Productiva, MinComercio, Norad y Organización de las Naciones Unidas para el Desarrollo Industrial. https://www.colombiaproductiva.com/PTP/media/documentos/Sectoriales/PuntoAqua/Capacitaciones/La-asociatividad-en-las-cadenas-de-valor-de-la-acuicultura-de-camaron-y-tilapia-en-Colombia.pdf

Python Software Foundation. (s.f.). *Python 3.14.4 documentation* [En línea]. https://docs.python.org/3/

Rojas Prada, W. S. (2025). *Ambiente web para la simulación de la producción piscícola en estanques gestionados con aprendizaje automático* (Tesis de pregrado) [En línea]. Universidad Industrial de Santander. https://noesis.uis.edu.co/handle/20.500.14071/45412

Rouhi, A., & Lano, K. (2023). Towards a pattern-based model transformation framework. *Software: Practice and Experience*, *53*, 1815-1849. https://doi.org/10.1002/spe.3215

Rueda-Barrios, G. E., Bohórquez-Farfán, L., Reyes-Figueroa, J. C., & Gómez-Díaz, D. (2019). Diagnóstico de las unidades productivas en el sector piscícola de Santander (Colombia). *Espacios*, *40*(28). ISBN 0798-1015.

Satyajith Amaran, B. S., Nikolaos V. Sahinidis, & Bury, S. J. (2016). Simulation optimization: a review of algorithms and applications [En línea]. *Annals of Operations Research*. https://doi.org/https://doi.org/10.1007/s10479-015-2019-x

Servicio Estadístico Pesquero Colombiano. (2022). Reporte Estadístico del Sector Pesquero y de la Acuicultura de Colombia.

Spring Boot Team. (s.f.). *Documentation Overview* [En línea]. https://docs.spring.io/spring-boot/documentation.html

StarUML. (s.f.). *StarUML Documentation* [En línea]. https://docs.staruml.io/

Streamlit. (s.f.). *Streamlit documentation* [En línea]. https://docs.streamlit.io/

Taghizadeh, E. (s.f.). *Simulation-Based Optimization: Stimulate To Test Potential Scenarios And Optimize For Best Performance* [En línea]. https://www.informs.org/Publications/OR-MS-Tomorrow/Simulation-Based-Optimization-Stimulate-To-Test-Potential-Scenarios-And-Optimize-For-Best-Performance

The PostgreSQL Global Development Group. (s.f.). *What Is PostgreSQL?* [En línea]. https://www.postgresql.org/docs/current/intro-whatis.html

Trondsen, T. H., & Hansen, V. (2021). *An application of approximate dynamic programming/-reinforcement learning to salmon production scheduling* (Tesis de maestría) [nva type: DegreeMaster]. Norges teknisk-naturvitenskapelige universitet. https://api.nva.unit.no/publication/0198ef758d8c-3adbbaa4-dd9c-4b5b-9b7c-e3954f099436

Unidad de Planificación Rural Agropecuaria. (2023). *Producto Interno Bruto (PIB) IV trimestre y año 2023* [[Diapositivas]]. https://www.agronet.gov.co/Lists/Boletin/Attachments/21552/PIB%20IV%20trimestre%202023.pdf

Weintraub, A., & Romero, C. (2006). Operations Research Models and the Management of Agricultural and Forestry Resources: A Review and Comparison [En línea]. *INFORMS Journal on Applied Analytics*, *36*, 446-455. https://doi.org/10.1287/inte.1060.0222

Zhang, P. (2010). CHAPTER 2 - Industrial control engineering [En línea], 41-70. https://doi.org/https://doi.org/10.1016/B978-1-4377-7807-6.10002-6

Apéndices

Apéndice A

Supuestos del modelo de optimización

Tabla A1
Comparativa de evolución y relajación de supuestos del modelo de optimización.

Fase del modelo	Supuestos Iniciales	Abstracción / Modificación	Justificación Técnica
Modelo lineal binario de optimización multi-estanque con única demanda constante y rutinaria	▪ Crecimiento constante de los peces ▪ Producción discreta por estanque ▪ Capacidad fija por estanque ▪ La demanda es única, constante y secuencial ▪ Un estanque por productor	▪ Ninguna	Permitir la verificación inicial de la lógica de flujo del cronograma sin inducir infactibilidad por restricciones físicas.

Fase del modelo (Cont.)	Supuestos Iniciales	Abstracción / Modificación	Justificación Técnica
Modelo lineal binario de optimización multi-estanque con única demanda constante, pero con frecuencia conocida	▪ Crecimiento constante de los peces ▪ Producción discreta por estanque ▪ Capacidad fija por estanque ▪ La demanda es única ▪ Un estanque por productor	▪ Formalización de parámetros de contrato: frecuencia, inicio y pedidos totales.	Alinear el modelo matemático con las obligaciones comerciales y la volatilidad de pedidos en mercados estructurados.
Modelo lineal binario de optimización multi-estanque con demandas constantes, pero crecimiento constante	▪ Crecimiento constante predefinido de los peces ▪ Producción discreta por estanque	▪ Implementación de *Pool* Global de productores. ▪ Promedio de perfiles de temperatura para un crecimiento de peces constante.	Reducir la carga computacional ante la naturaleza *NP-Hard* del problema, permitiendo la escalabilidad a nivel de asociación colectiva sin perder representatividad biológica.

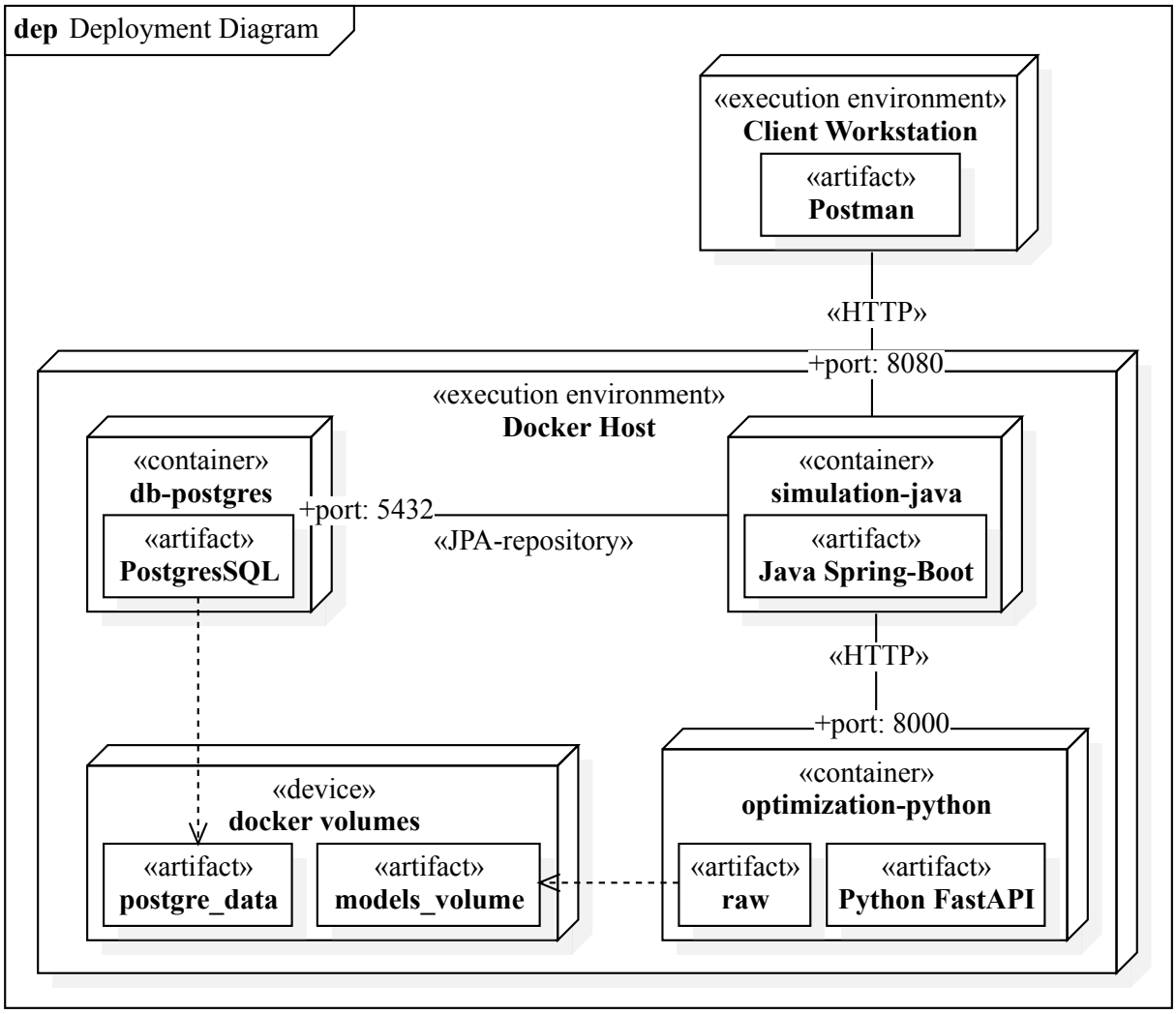
Fase del modelo (Cont.)	Supuestos Iniciales	Abstracción / Modificación	Justificación Técnica
Modelo lineal entero-mixto de optimización multi-estanque con demandas determinadas	▪ Crecimiento constante de los peces ▪ Producción discreta por estanque ▪ Capacidad fija por estanque ▪ Un estanque por productor	▪ Sincronización $t = 0$ con la fecha actual. ▪ Restricción de estanques vacíos en estado inicial.	Establecer una línea base operativa coherente para la toma de decisiones inmediata en escenarios de planeación desde cero.
Modelo lineal entero-mixto de optimización multi-estanque con demanda relajada	▪ Producción discreta por estanque	▪ Se relaja la fijeza de la demanda introduciendo penalizaciones por insatisfacción. ▪ Se integran variables de decisión multi-periodo.	Soportar la incertidumbre operativa y coordinar las entregas parciales de múltiples productores sin que el solucionador falle por infactibilidad ante picos de demanda.

Apéndice B

Diagrama de despliegue

Figura B1

Despliegue del artefacto con Docker



Nota. El diagrama muestra cómo se construye la imagen docker del software desarrollado, asignando volúmenes de datos para la persistencia y los puertos de red entre los diferentes subsistemas.