UNIVERSIDAD NACIONAL DE SAN MARTIN

RCSI
https://revistas.unsm.edu.pe/index.php/rcsi
e-ISSN: 2709-992X

Artículo Original
Original Article
Jul-Dic, 2026

Arquitectura cloud-based de gemelos digitales para el diseño óptimo de cadenas de suministro de gran escala

Cloud-based architecture of digital twins applied to the optimal design of large-scale supply chains

Marmolejo-Saucedo, José Antonio[1]*

Rodríguez-Aguilar, Román[2]

Abraham Mendoza[1]

[1]Facultad de Ingeniería, Universidad Panamericana, Álvaro del Portillo 49, Zapopan, Jalisco, 45010, México
[2]Facultad de Ciencias Económicas y Empresariales, Universidad Panamericana, Augusto Rodin 498, Ciudad de México, 03920, México

Recibido: 18 May. 2026 | **Aceptado:** 11 Jul. 2026 | **Publicado:** 20 Jul. 2026

Autor de correspondencia*: jmarmolejo@up.edu.mx

Cómo citar este artículo: Marmolejo-Saucedo, J. A., Rodriguez-Aguilar, R. & Mendoza, A. (2026). Arquitectura cloud-based de gemelos digitales para el diseño óptimo de cadenas de suministro de gran escala. *Revista Científica de Sistemas e Informática, 6*(2), e1682. https://doi.org/10.51252/rcsi.v6i2.1682

RESUMEN

El diseño de redes de suministro de gran escala es un problema MILP NP-duro que requiere representación dinámica y optimización eficiente en contextos industriales. Este estudio propuso, implementó y validó una arquitectura cloud-based de cuatro capas que integra Microsoft Azure Digital Twins (ADT), un modelo MILP multi-escalón, un PSO híbrido y un motor de integración en Python con sincronización bidireccional. La principal contribución algorítmica consistió en emplear la relajación lineal del MILP como función de aptitud del PSO, proporcionando una cota inferior formal del problema original. Se ejecutaron 74 corridas experimentales con un operador de reparación por capacidad, alcanzando una tasa de infactibilidad nula. Las instancias evaluadas, en escalas S, M y XL y con cinco niveles de complejidad, evidenciaron una relación inversa entre la brecha de integralidad y la calidad de la solución PSO. Asimismo, las instancias de baja complejidad combinatoria lograron una consistencia superior al 90% en las decisiones de apertura, preservando el estado decisional del gemelo digital. El marco propuesto integra arquitectura cloud-based, optimización MILP y PSO híbrido para cadenas de suministro con validación experimental.

Palabras clave: Azure Digital Twins; cadena de suministro; diseño de red; gemelo digital; MILP; PSO; Python

ABSTRACT

The design of large-scale supply networks is an NP-hard MILP problem that requires dynamic representation and efficient optimization in industrial contexts. This study proposed, implemented, and validated a four-layer cloud-based architecture that integrates Microsoft Azure Digital Twins (ADT), a multi-echelon MILP model, a hybrid PSO, and a Python-based integration engine with bidirectional synchronization. The main algorithmic contribution consisted of using the linear relaxation of the MILP as the PSO fitness function, providing a formal lower bound for the original problem. A total of 74 experimental runs were conducted using a capacity repair operator, achieving a zero-infeasibility rate. The evaluated instances, across S, M, and XL scales and five complexity levels, showed an inverse relationship between the integrality gap and the quality of the PSO solution. Likewise, instances with low combinatorial complexity achieved over 90% consistency in opening decisions, preserving the decision state of the digital twin. The proposed framework integrates cloud-based architecture, MILP optimization, and hybrid PSO for supply chains with experimental validation.

Keywords: Azure Digital Twins; supply chain; network design; digital twin; PSO, MILP; Python

1. INTRODUCCIÓN

Las cadenas de suministro de gran escala constituyen sistemas logísticos de alta complejidad combinatoria. Dichos sistemas son caracterizados por la interacción de múltiples escalones o niveles de instalaciones como lo son proveedores, plantas de producción, centros de distribución y clientes finales, flujos de materiales e información, restricciones de capacidad heterogéneas y demanda sujeta a variabilidad e incertidumbre (Simchi-Levi et al., 2008). El diseño de redes implica decisiones estratégicas de localización, apertura y cierre de instalaciones, asignación de flujos entre escalones y determinación de niveles de servicio (Pérez-Baquero et al., 2026). Estos modelos matemáticos tienen complejidad NP-hard y la búsqueda de soluciones óptimas supera la capacidad de los métodos exactos convencionales (Geoffrion & Graves, 1974; Melo et al., 2009).

Hoy en día, fenómenos disruptivos como pandemias globales, volatilidad geopolítica, fluctuaciones en la demanda y disrupciones en la cadena de proveedores ha evidenciado las limitantes de los modelos estáticos de diseño de red. Diversos autores sobre gemelos digitales para cadenas de suministro han mencionado el potencial de plataformas basadas en la nube para habilitar la toma de decisiones dinámica. (Marmolejo-Saucedo, 2020) propone el desarrollo de gemelos digitales con un caso de estudio usando programas especializados en diseño de cadenas de suministro, mientras que (Marmolejo-Saucedo, 2022) extiende este enfoque a problemas de optimización a gran escala. El desarrollo de gemelos digitales basados en este tipo de arquitecturas permite la representación y gestión de instalaciones, así como la automatización de procesos críticos, estableciendo un ecosistema digital interconectado capaz de responder de manera ágil a las demandas del entorno (Dominguez Martinez et al., 2026; Wasi et al., 2025).

Un gemelo digital es una representación virtual y dinámica de un sistema físico o lógico, sincronizada con su contraparte real mediante flujos bidireccionales de datos. Esto permite monitorear el estado actual del sistema, simular escenarios futuros y retroalimentar planes operativos para contingencias (Grieves, 2014; Tao et al., 2019). En el contexto de cadenas de suministro, los gemelos digitales ofrecen la posibilidad de modelar la topología de la red, sus parámetros operativos y la interdependencia entre nodos, intercambiando y actualizando datos de manera sistemática (Ivanov & Dolgui, 2021). Microsoft Azure Digital Twins (ADT) es una plataforma como servicio (PaaS) basada en un lenguaje llamado DTDL (Digital Twins Definition Language). Actualmente representa una de las soluciones en la nube más maduras en el mercado y permite la integración de múltiples sistemas y tecnologías de información (Microsoft, 2025c).

Es conocido que el problema de diseño de cadenas de suministro multi-escalón corresponde a un modelo MILP (Farahani et al., 2014; Melo et al., 2009). Este tipo de problemas que consideran instancias de escala industrial, agregan complejidad de cálculo para los optimizadores exactos, ya que pueden requerir elevados tiempos de cómputo (Avella et al., 2023). En este contexto, la optimización por enjambre de partículas (PSO) híbrida reporta reducciones significativas de tiempo de cómputo (Kennedy & Eberhart, 1995; Noel & Jannett, 2004). Sin embargo, los trabajos que usan PSO para resolver problemas MILP de cadenas de suministro no contemplan la integración con plataformas de gemelos digitales basados en la nube (cloud-based).

El análisis de la literatura sugiere una brecha técnica, ya que no existe una arquitectura tecnológica unificada que integre formalmente (i) la representación ontológica y dinámica de una cadena de suministro multi-escalón, (ii) un modelo MILP con acoplamiento directo a dicha representación, y (iii) un PSO híbrido con mecanismos de reparación de factibilidad y evaluación del desempeño

basado en relajación del problema líneal (LP). Por esta razón, las contribuciones principales de este trabajo son: (1) arquitectura de cuatro capas ilustrada en la Figura 1; (2) PSO-híbrido que garantiza acotamiento en la solución óptima, cuyo pseudocódigo se presenta en la Figura 10; y (3) validación experimental sobre múltiples corridas con instancias de diversa complejidad.

2. FUNDAMENTACIÓN TEÓRICA

2.1. Gemelos digitales en la gestión de cadenas de suministro

El concepto de gemelo digital fue formalizado en el ámbito de la manufactura (Grieves, 2014; Grieves & Vickers, 2017) y su adopción en cadenas de suministro se ha acelerado con la convergencia del Internet de las cosas (IoT), computación en la nube y analítica en tiempo real (Lim et al., 2020; Tao et al., 2019). Algunos trabajos en este campo incluyen el diseño de gemelos digitales para cadenas de suministro (Marmolejo-Saucedo, 2020) y su extensión a problemas de optimización de gran escala como el problema de empaque (Marmolejo-Saucedo, 2022), sentando las bases metodológicas sobre las que se construye la integración de una plataforma en la nube con un problema MILP propuesta en el presente estudio.

(Ivanov & Dolgui, 2021) distinguen tres arquetipos de gemelos digitales en la gestión de cadenas de suministro: gemelos de visibilidad, gemelos de análisis y gemelos de optimización. La taxonomía de (Kritzinger et al., 2018) distingue “digital model” (sincronización manual), “digital shadow” (digital unidireccional) y “digital twin” (bidireccional). La arquitectura propuesta implementa el tercer arquetipo, materializado en la propiedad openStatus de los nodos Plant y DC del grafo ADT (ver Figura 7). (Ivanov & Dolgui, 2019; Longo et al., 2023) documentan beneficios de los DT para la resiliencia y la evolución hacia ecosistemas integrados. La escalabilidad de la nube resulta crítica para gestionar los volúmenes de datos de cadenas de suministro de gran escala (Somma et al., 2025; Wasi et al., 2025).

2.2. Microsoft Azure Digital Twins y modelado DTDL

Azure Digital Twins es una plataforma PaaS que permite crear grafos de instancias de gemelos digitales y consultarlos mediante un lenguaje de tipo SQL (Microsoft, 2025c). El modelado se realiza con DTDL v3 (Microsoft, 2023), un esquema basado en JSON-LD que especifica propiedades, relaciones y telemetría. Las clases metamodelo relevantes son: Interface, Property, Relationship y Telemetry. La integración con Azure IoT Hub, Event Grid y Stream Analytics (Milenkovic, 2022; Somma et al., 2025) habilita la sincronización automática ante cambios de estado. La ontología DTDL propuesta para la cadena de suministro (ilustrada en la Figura 3) no se ha reportado en la literatura existente (Microsoft, 2025b), constituyendo una contribución original de este trabajo.

2.3. Optimización de redes de suministro: modelos MILP y enfoques exactos

(Geoffrion & Graves, 1974) propusieron el problema de diseño multi-escalón y ha sido objeto de extensa revisión (Farahani et al., 2014; Melo et al., 2009). Los optimizadores comerciales como Gurobi, CPLEX, CBC, resuelven instancias de tamaño medio mediante branch-and-cut, aunque su desempeño disminuye exponencialmente con el incremento en el número de variables binarias (Avella et al., 2023). En este estudio, la relajación LP proporciona una cota inferior de alta calidad en tiempo de cómputo polinomial, la cual se emplea como función de aptitud del PSO.

2.4. Metaheurísticas híbridas para optimización combinatoria en cadenas de suministro

El algoritmo PSO fue introducido inicialmente por (Kennedy & Eberhart, 1995), posteriormente (Shi & Eberhart, 1998) incluyeron el coeficiente de inercia mejorando el desempeño del algoritmo. La variante BPSO (Kennedy & Eberhart, 1997) binariza posiciones mediante función sigmoide y es la técnica estándar en problemas de localización y algunos otros de optimización discreta. (Noel & Jannett, 2004) reportan hibridaciones PSO-LP con mejoras significativas. En el presente trabajo, la hibridación se implementa evaluando la aptitud vía relajación LP del subproblema de flujos con aperturas de instalaciones fijas. Lo anterior reduce el espacio de búsqueda efectivo a solo |I|+|J|| variables. Por su parte, (Guo & Mantravadi, 2025; Tirkolaee et al., 2020) documentan aplicaciones recientes de PSO en cadenas de suministro; sin embargo, no consideran integración con gemelos digitales cloud-based.

3. MATERIALES Y MÉTODOS.

3.1. Formulación matemática del modelo MILP

3.1.1. Conjuntos, parámetros y variables

El modelo matemático incluye una red de tres escalones: I (plantas), J (CEDIS) y K (clientes). Cada planta i tiene un costo fijo de apertura f_i^P, una capacidad p_i y un costo variable de producción v_i; cada CEDIS j tiene costo fijo f_j^D y capacidad s_j; cada cliente k demanda d_k unidades. Los costos de transporte son c_{ij}^{PD} entre planta y CEDIS, y c_{jk}^{Dc} entre CEDIS y cliente. Se incluyen variables: $y_i \in \{0,1\}$ (apertura planta), $z_j \in \{0,1\}$ (apertura CEDIS), $x_{ij} \geq 0$ (flujos de planta a CEDIS), $q_{jk} \geq 0$ (flujos CEDIS a cliente). El grafo ADT (Figura 6) almacena los parámetros como propiedades de las instancias del gemelo digital.

3.1.2. Función objetivo

La función objetivo minimiza el costo total:

$$\min Z = \sum_{i\in I} f_i^{P}\cdot y_i + \sum_{j\in J} f_j^{D}\cdot z_j + \sum_{i\in I}\sum_{j\in J}\left(c_{ij}^{PD}+v_i\right)\cdot x_{ij} + \sum_{j\in J}\sum_{k\in K} c_{jk}^{DC}\cdot q_{jk}$$

3.1.3. Restricciones

Las restricciones que el modelo considera son las típicamente empleadas en los diseños de cadena de suministro (Melo et al., 2009):

(R1) $\sum_{j\in J} q_{jk} = d_k \quad \forall k\in K$

(R2) $\sum_{i\in I} x_{ij} = \sum_{k\in K} q_{jk} \quad \forall j\in J$

(R3) $\sum_{j\in J} x_{ij} \leq p_i\cdot y_i \quad \forall i\in I$

(R4) $\sum_{k\in K} q_{jk} \leq s_j\cdot z_j \quad \forall j\in J$

(R5) $y_i \in \{0,1\}\ \forall i\in I,\ z_j \in \{0,1\}\ \forall j\in J,\ x_{ij} \geq 0,\ q_{jk} \geq 0.$

3.1.4. Relajación lineal y su papel en el PSO

La relajación LP sustituye las condiciones $y_i \in \{0,1\}$ y $z_j \in \{0,1\}$ por los intervalos continuos [0,1]. El valor resultante Z^{LP} acota inferiormente a Z^{MIP}, y esta propiedad es la que justifica usarla como función de aptitud (fitness) del PSO. Si una partícula codifica el vector de aperturas $(\hat{y}, \hat{z})$, su costo real nunca estará por debajo de $Z^{LP}(\hat{y}, \hat{z})$. Dicho de otro modo, la guía de búsqueda no puede ser

sobreoptimista, algo que ninguna función de aptitud heurística garantiza. La implementación resuelve estos subproblemas con Pyomo y CBC/GLPK (Hart et al., 2011).

$$Z^{\mathrm{LP}}(\hat{y}, \hat{z}) \leq Z^{\mathrm{MIP}}$$

3.2. Arquitectura propuesta: integración ADT–MILP–PSO

3.2.1. Visión general de la arquitectura de cuatro capas

La Figura 1 muestra el esquema completo. Se tiene cuatro capas con alcances y limites: Azure Digital Twins (Capa 1), el motor de ejecución en Python (Capa 2), el módulo MILP (Capa 3) y el PSO híbrido (Capa 4). El diseño por capas permite que cada componente sea independiente y reemplazable, por ejemplo, cambiar de optimizador o probar variantes del PSO. La nube aporta la escalabilidad que una cadena de gran escala demanda en volumen de datos y operaciones distribuidas (Marmolejo-Saucedo, 2020; Wasi et al., 2025).

La Figura 2 es el punto de partida de la arquitectura implementada, para ello se debe tener la instancia Azure Digital Twins con estado Activo en Azure Portal. Antes de ejecutar cualquier línea de código Python es necesario tener accesibles los servicios de Azure en la nube. El hostname que aparece en pantalla (adt-scm-suministro.api.eus.digitaltwins.azure.net) es el mismo valor que se registra en el archivo. env del proyecto y que el motor de ejecución Python utiliza para autenticarse mediante el SDK azure-digitaltwins-core.

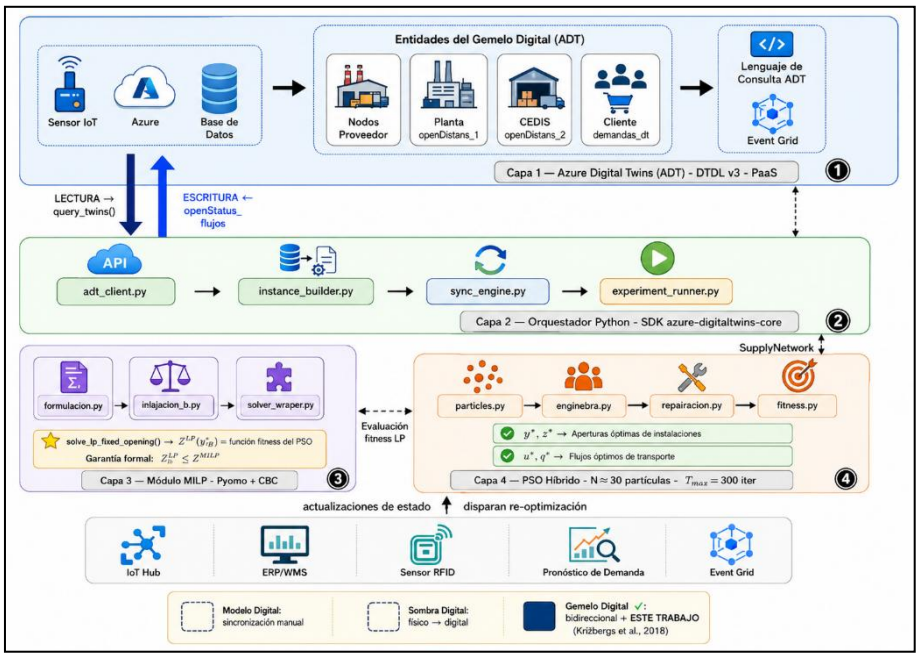

Figura 1. Arquitectura cloud-based de cuatro capas del sistema propuesto: Azure Digital Twins (Capa 1), motor de ejecución en Python (Capa 2), Módulo MILP-Pyomo (Capa 3) y PSO Híbrido (Capa 4) con sincronización bidireccional.

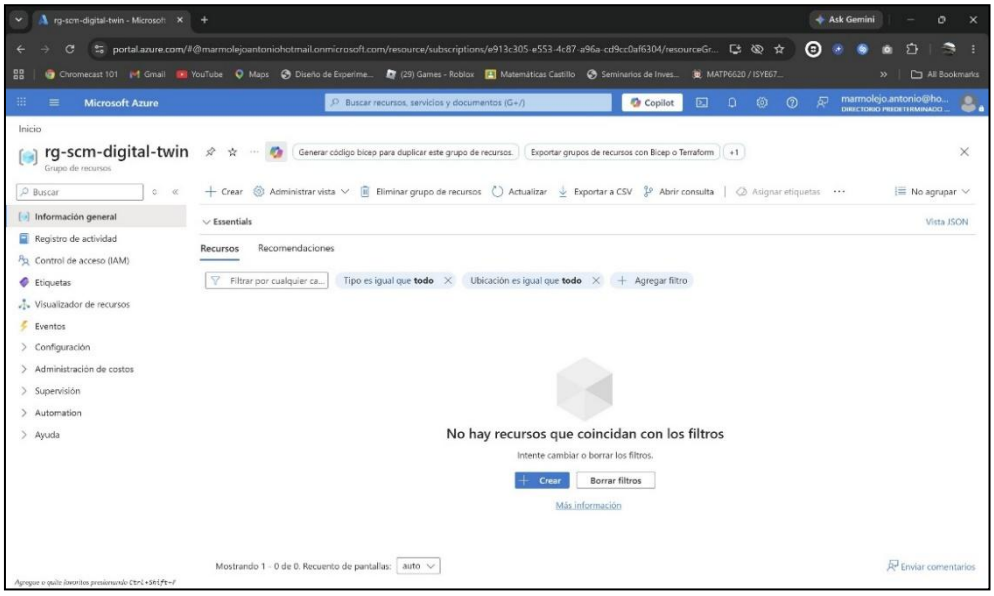

Figura 2. Instancia Azure Digital Twins desplegada en Azure Portal: hostname (adt-scm-suministro.api.eus.digitaltwins.azure.net), región East US y estado Activo, confirmando la disponibilidad del servicio en la nube para la integración con el orquestador Python.

3.2.2. Capa 1: modelado ontológico en Azure Digital Twins

La instancia ADT gestiona un grafo con cuatro tipos de gemelos modelados con DTDL v3 (Microsoft, 2023): Supplier, Plant, DC y Customer. La ontología completa, con propiedades y relaciones DTDL, se ilustra en la Figura 3. La integración con Azure IoT Hub y Event Grid habilita la sincronización automática (Marmolejo-Saucedo, 2022; Milenkovic, 2022; Somma et al., 2025). La propiedad openStatus en los modelos Plant y DC corresponde directamente a las variables de decisión binarias y_i y z_j del MILP. El grafo de instancias específico para una red demo se presenta en la Figura 6.

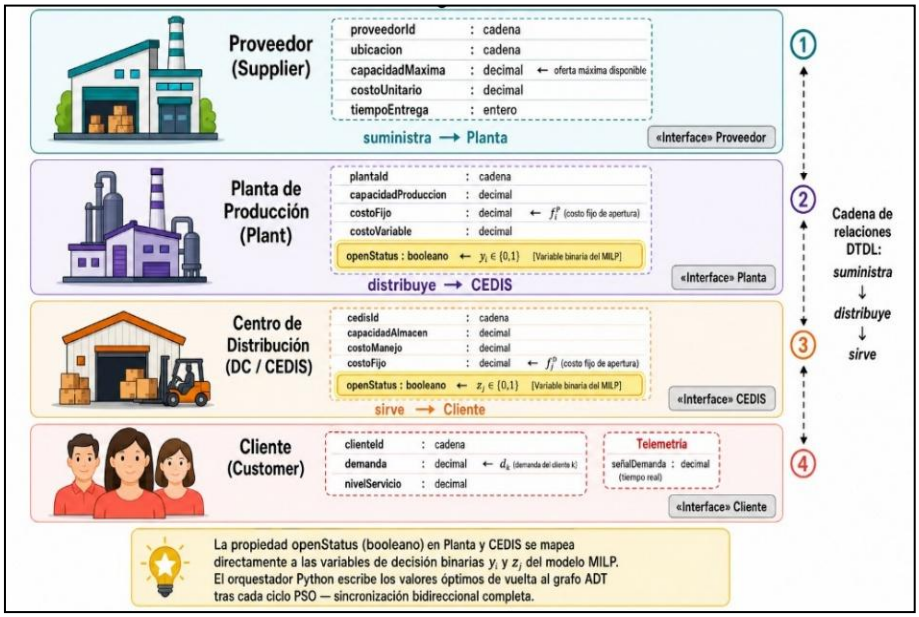

Figura 3. Ontología DTDL v3 para cadena de suministro multi-escalón en Azure Digital Twins

Una vez que la instancia está activa, el siguiente paso consiste en ingresar los cuatro modelos DTDL con create_models.py. La Figura 4 muestra esos modelos ya disponibles en Azure Digital Twins Explorer. La Figura 5 detalla la estructura interna del tipo Customer en formato JSON-LD, donde se

puede ver que las propiedades declaradas, en particular demand (schema: double), son exactamente las que instance_builder.py consulta cuando construye el vector de demanda para el modelo MILP.

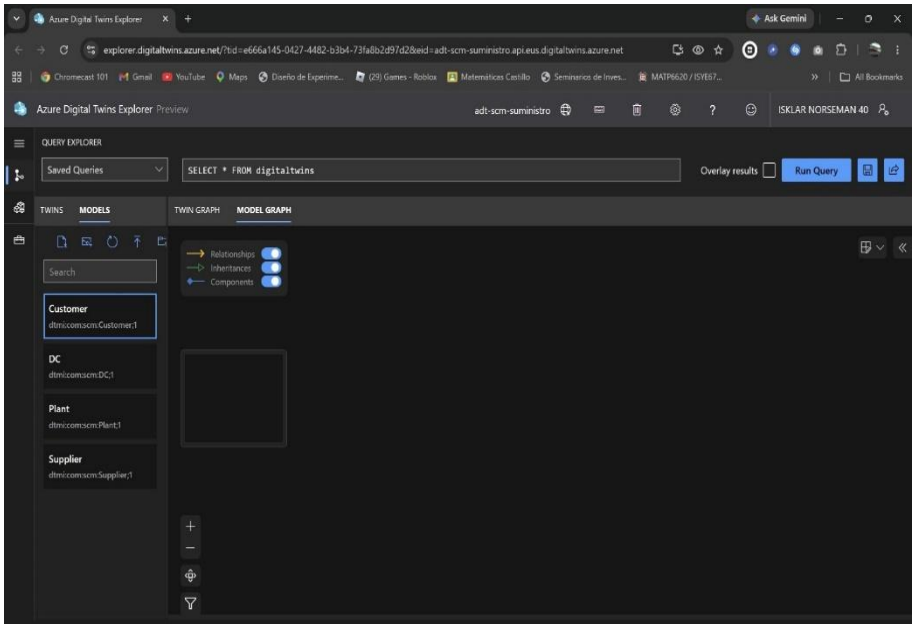

Figura 4. Modelos DTDL de la cadena de suministro cargados en Azure Digital Twins Explorer, con sus identificadores completos (dtmi:com:scm:Customer;1, DC;1, Plant;1, Supplier;1

La definición DTDL del tipo Customer (Figura 5) sigue el estándar JSON-LD de (Microsoft, 2023). Cada propiedad declara su tipo de dato y esquema, lo que permite al motor de ejecución Python leer los valores directamente desde el grafo ADT sin comunicación adicional. El campo de Telemetría incluido en el modelo también deja abierta la puerta a señales de demanda en tiempo real, una extensión natural del sistema para trabajo futuro.

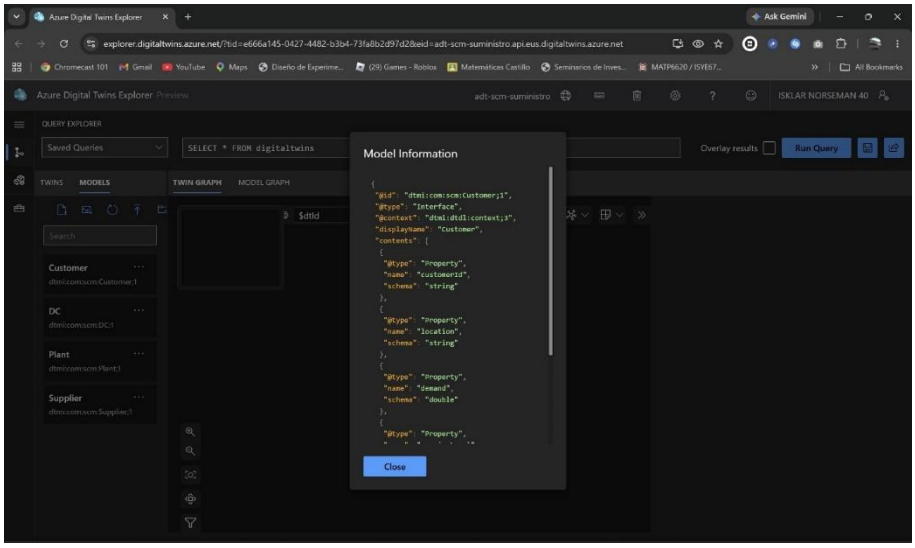

Figura 5. Definición JSON-LD del modelo DTDL Customer en Azure Digital Twins Explorer, con las propiedades customerId, demand, location y serviceLevel. El campo demand (schema: double) es leído por instance_builder.py para construir el vector de demanda del modelo MILP

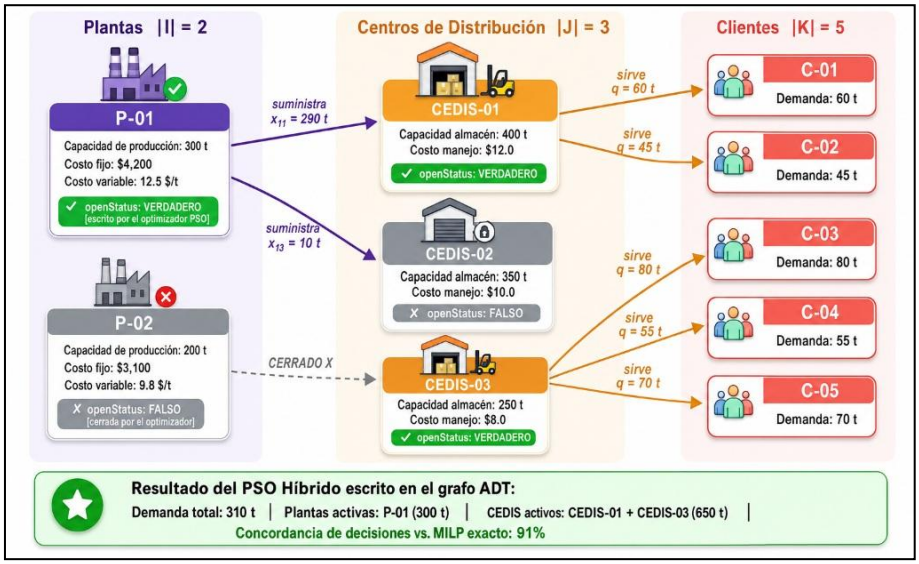

Figura 6. Grafo de instancias en Azure Digital Twins: 2 plantas, 3 CEDIS y 5 clientes con propiedades DTDL, relaciones de flujo (suministra, distribuye, sirve) y los valores de openStatus escritos por el PSO híbrido tras el ciclo de optimización

3.2.3. Capa 2: Motor de ejecución Python (orquestador) e interfaz con ADT

La interfaz se implementa con el SDK azure-digitaltwins-core (Microsoft, 2025a). La Tabla 1 describe los componentes. El protocolo de sincronización bidireccional completo se ilustra en la Figura 7.

Tabla 1. Componentes de integración ADT–Python y protocolo de sincronización.

Componente	Módulo Python	Operación ADT	Descripción
Autenticación	adt_client.py	DefaultAzureCredential	Credenciales mediante Service Principal o Managed Identity. Sin almacenamiento de claves en código.
Lectura de estado	sync_engine.py	query_twins() / get_digital_twin()	Consulta ADT Query Language para obtener capacidades, costos y demandas actuales de todos los nodos del grafo.
Construcción del modelo	instance_builder.py	(local)	Mapeo grafo ADT → objeto SupplyNetwork → ConcreteModel Pyomo con parámetros actualizados dinámicamente.
Ejecución PSO	pso_hibrido.py	(local)	PSO híbrido opera en memoria local. Llama a solve_lp_fixed_openings() como función de fitness en cada evaluación.
Escritura de decisiones	sync_engine.py	update_digital_twin()	Escribe openStatus (bool) en Plant y DC, y flujos óptimos en relaciones DTDL. Implementa la sincronización bidireccional.
Trigger (activador) de reoptimización	adt_client.py	Event Grid subscription	Suscripción a eventos de cambio en ADT. Si demanda o capacidad cambia >5%, dispara automáticamente nuevo ciclo PSO.

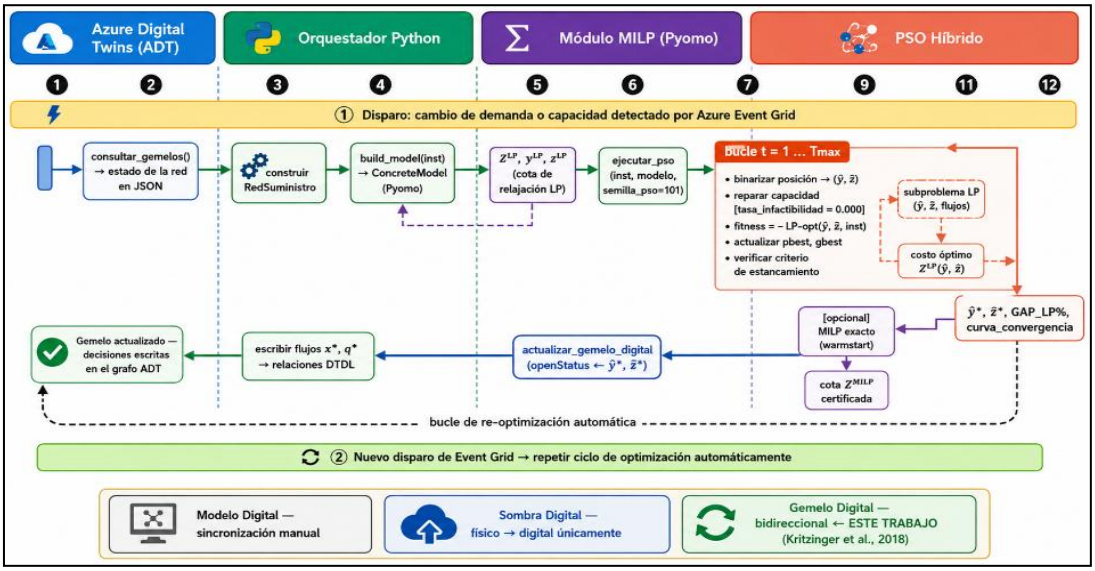

Figura 7. Protocolo de sincronización bidireccional entre Azure Digital Twins y el motor de optimización PSO-MILP: diagrama de secuencia UML con las 12 interacciones del ciclo, desde el inicio por Event Grid hasta la escritura de openStatus en el grafo ADT

La Figura 8 representa todo el sistema. El grafo de instancia en Azure Digital Twins Explorer, con los 10 gemelos conectados mediante flechas que representan las relaciones DTDL. EL panel a la derecha de la imagen DC-02 tiene openStatus = Falso. Ese valor es calculado por el PSO híbrido a partir de la relajación LP del modelo MILP y escrito de vuelta al gemelo mediante update_digital_twin(). En esa propiedad se verifica la sincronización bidireccional que diferencia un Digital Twin de un sistema de monitoreo unidireccional (Kritzinger et al., 2018).

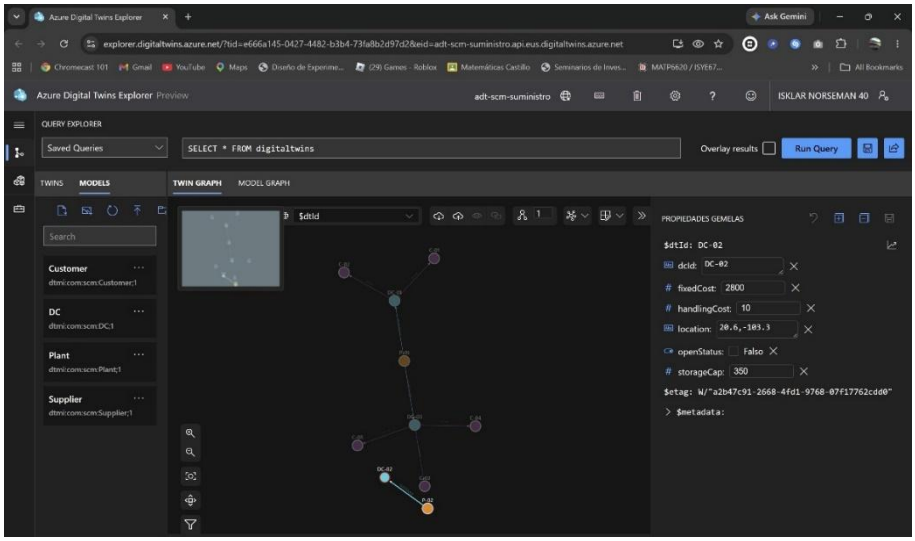

Figura 8. Grafo de instancias en Azure Digital Twins Explorer: 10 gemelos conectados con relaciones DTDL.

La terminal de Anaconda Prompt mostrada en la Figura 9 valida el ciclo del proceso de optimización. El Paso 1 confirma que el motor de ejecución leyó los 10 gemelos desde ADT con una demanda total de 310 unidades. El Paso 2 muestra los resultados del PSO, dando un costo óptimo de $199,572.85 en 40 iteraciones con tasa de infactibilidad de 0.0000. Esto confirma que el operador de reparación funcionó sin fallo en todas las evaluaciones del enjambre. El Paso 3 lista la información enviada al grafo ADT, cerrando el lazo entre el optimizador y el gemelo digital.

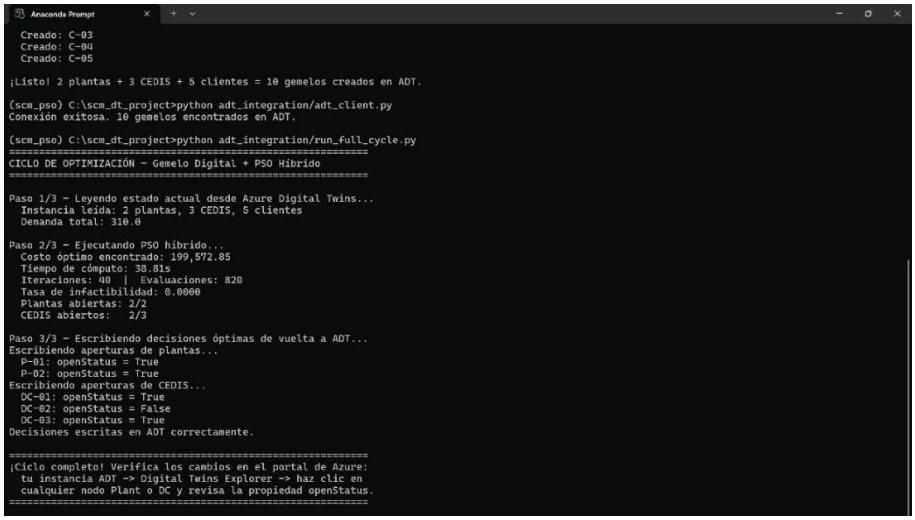

Figura 9. Ejecución del ciclo completo en Anaconda Prompt: Paso 1, lectura desde ADT (demanda total 310 unidades); Paso 2, PSO híbrido (costo $199,572.85, infactibilidad 0.0000, 40 iteraciones); Paso 3, escritura de openStatus al grafo ADT (DC-02: False; resto: True)

El módulo MILP y el PSO no tienen dependencias directas de la API de Azure, lo cual es un aspecto característico del diseño del gemelo digital propuesto. Los módulos reciben y devuelven objetos Python estándar. Esta independencia garantiza la reproducibilidad experimental en entornos sin conectividad a ADT y la portabilidad del sistema para otras plataformas de gemelos digitales, como Eclipse Ditto o FIWARE (Somma et al., 2025). Adicionalmente, la estandarización de formatos de datos y protocolos de comunicación resulta indispensable para garantizar una adecuada interoperabilidad entre plataformas y actores de la cadena de suministro (Wasi et al., 2025).

3.2.4. Capa 4: algoritmo PSO híbrido

Codificación de la partícula

Cada partícula codifica las variables de apertura en un vector de valores reales reales $p_n^t \in \mathbb{R}^{(|I|+|J|)}$ considerando la formulación de (Kennedy & Eberhart, 1997). La binarización se realiza mediante función sigmoide con umbral θ=0.5: $\hat{y}_i = 1$ si $\sigma(p_i) \geq \theta$, donde $\sigma(p) = 1/(1+e^{-p})$. La dimensionalidad del espacio de búsqueda es reducida después de resolver el subproblema LP con instalaciones fijas. Es decir, se pasa de $|I||J|+|J||K|+|I|+|J|$ a solo $|I|+|J|$ variables heurísticas. Los flujos óptimos x_{ij} y q_{jk} se calculan resolviendo dicho subproblema LP. La función de aptitud combina los costos fijos de las aperturas propuestas con el óptimo de los flujos del LP $f(p) = \Sigma_i f_i^P \cdot \hat{y}_i + \Sigma_j f_j^D \cdot \hat{z}_j$ + LP-opt($\hat{y}$, $\hat{z}$). El pseudocódigo completo se muestra en la Figura 10.

$$\sigma(p) = \frac{1}{1+e^{-p}} \quad ; \qquad f(p) = \sum_i^{\square} f_i^P \cdot \hat{y}_i + \sum_j^{\square} f_j^D \cdot \hat{z}_j + \text{LP-opt}(\hat{y}, \hat{z})$$

Operador de reparación de viabilidad

Antes de evaluar cada partícula se verifica que la capacidad instalada cubra la demanda total. Cuando no la cubre, el operador abre instalaciones adicionales en orden descendente de capacidad hasta cerrar el déficit. Es un mecanismo simple y para las 74 corridas del estudio alcanzó la tasa de infactibilidad en 0.000.

Hibridación por inicialización LP

Los parámetros de actualización del PSO siguen el esquema de Clerc-Kennedy: ω=0.729, c_1=c_2=1.494 (Shi & Eberhart, 1998). En la configuración de referencia, se fija la fracción de inicialización LP ρ_LP = 0.0 , ya que esto garantiza la independencia estadística entre las corridas experimentales.

$$v_n^{t+1}=\omega\cdot v_n^t+c_1\cdot r_1\cdot\left(pbest_n-p_n^t\right)+c_2\cdot r_2\cdot(gbest-p_n^t)$$

3.3. Implementación

El sistema está escrito en Python 3.11. Para el MILP usamos Pyomo 6.7+ (Hart et al., 2011) con CBC 2.10+ como solver primario y GLPK 5.0+ para los subproblemas LP; la interfaz con ADT corre sobre azure-digitaltwins-core 1.2+ (Microsoft, 2025a); el motor del PSO se apoya en NumPy 1.26+, el procesamiento de resultados en pandas 2.0+ y las figuras en matplotlib 3.8+. EL gemelo digital está organizado en siete módulos: data_generation.py, milp_model.py, lp_relaxation.py, pso_hibrido.py, experiments.py, plots.py y main.py para que cada módulo pueda probarse por separado. Los experimentos se ejecutan con semillas de instancia y PSO estrictamente independientes (semillas 1–10 para instancias, 101–110 para PSO), con esto se garantiza la independencia estadística de las corridas.

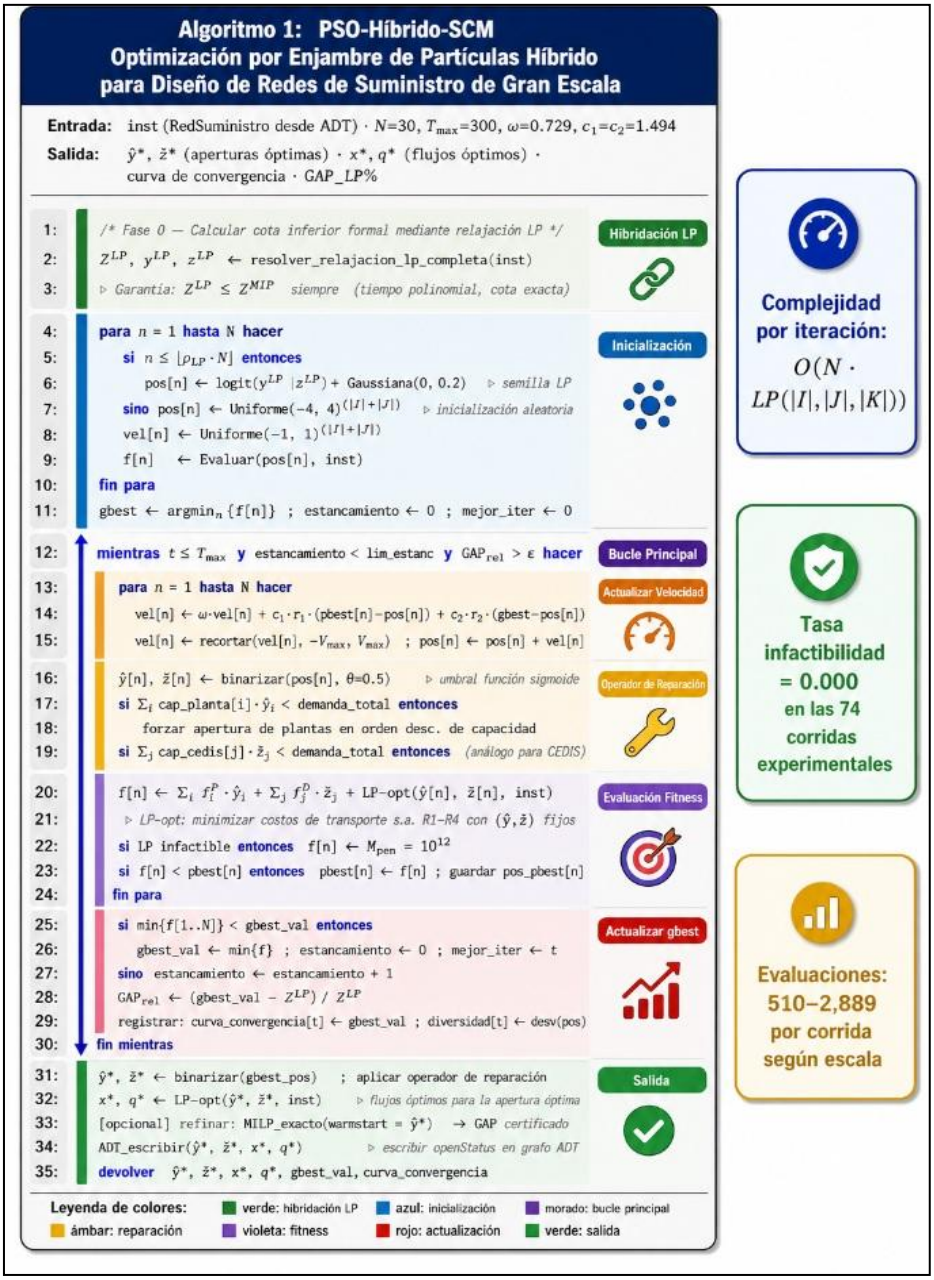

Figura 10. Pseudocódigo del Algoritmo 1 (PSO-Híbrido-SCM) para diseño de redes de suministro de gran escala

La Figura 11 presenta una verificación independiente del estado del grafo ADT, consultado desde Python directamente al API de Azure Digital Twins. Los resultados son consistentes con la solución del PSO, donde P-01, P-02, DC-01 y DC-03 aparecen como True (abiertos) y DC-02 como False (cerrado por el optimizador). Los cinco clientes aparecen como N/A, dado que la propiedad openStatus no aplica al tipo Customer.

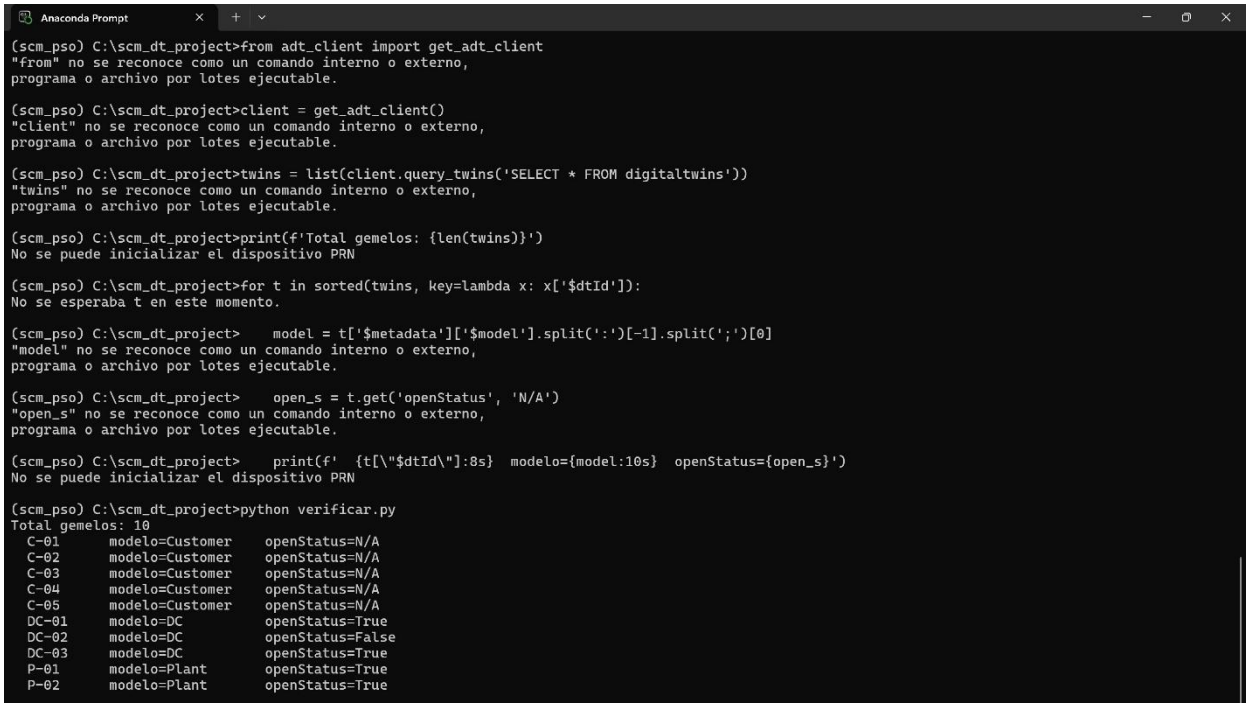

Figura 11. Verificación del estado del grafo ADT tras la optimización mediante consulta directa al API de Azure Digital Twins desde Python

3.4. Diseño experimental

3.4.1. Instancias de prueba

Se generaron instancias sintéticas reproducibles en tres escalas: S (5P, 10DC, 30C), M (10P, 20DC, 80C) y XL (30P, 60DC, 400C). La Tabla 2 resume la estructura dimensional. La escala XL representa un incremento de 3× en variables binarias y ≈14× en variables continuas respecto a M, situando el problema en un régimen de complejidad cualitativamente distinto.

Tabla 2. Estructura de las instancias de prueba por escala.

Escala	|I|	|J|	|K|	Variables binarias	Variables continuas	Total de Variables	Restricciones	Densidad de integralidad
S	5	10	30	15	350	365	55	0.041
M	10	20	80	30	1,800	1,830	130	0.016
XL	30	60	400	90	25,800	25,890	550	0.003

Para cada escala se evaluaron cinco tipos de complejidad: estándar(standard), capacidad ajustada (tight capacity) con capacity_ratio=1.05, costos fijos altos (high fixed costs) usando distribución Pareto, costos asimétricos (asymmetric costs) usando ruido multiplicativo en costos de transporte y caso combinado difícil (combined hard) que considera la capacidad ajustada + costos asimétricos + clientes en clusters. La demanda sigue una distribución log-normal (CV=0.3). El código es reproducible con semillas fijas.

3.4.2. Configuración de los algoritmos

El solver CBC se configuró con ratioGap=0.0001 y time_limit=300 s para escalas S/M y 120 s para escala XL. El PSO híbrido se configuró con: 30 partículas (20 en XL), máximo 300 iteraciones (50 en XL por límite de evaluaciones), ω=0.729, c_1=c_2=1.494, stagnation_limit=60, inicialización aleatoria pura (ρ_LP=0.0). Se realizaron 9 corridas independientes por configuración en escalas S/M (3 instancias × 3 semillas PSO) y 4 corridas en XL (2 instancias × 2 semillas PSO), totalizando 74 corridas.

$$v_n^{t+1}=\omega\cdot v_n^t+c_1\cdot r_1\cdot\left(pbest_n-p_n^t\right)+c_2\cdot r_2\cdot(gbest-p_n^t)$$

3.4.3. Métricas registradas

Las métricas registradas por corrida fueron: GAP_LP (GAP PSO vs. bound LP), GAP_MIP (GAP PSO vs. óptimo MILP), tiempo de cómputo MILP y PSO, número de iteraciones y evaluaciones de función de aptitud, tasa de infactibilidad del enjambre, diversidad final del enjambre, consistencia de decisiones de apertura entre PSO y MILP (plantas y CEDIS), y métricas estructurales de la instancia. Los experimentos se ejecutaron en un equipo ASUS Zenbook Duo UX8406MA con procesador Intel Core Ultra 9 185H (2.50 GHz, arquitectura x86-64) y 32 GB de RAM, bajo Windows 11 Home Single Language (versión 25H2, build 26200.8737), con Python 3.11, CBC 2.10 y GLPK 5.0.

4. RESULTADOS

4.1. Eficacia del operador de reparación

La tasa de infactibilidad del PSO fue de cero en la totalidad de las 74 corridas experimentales, en todas las escalas y en todos los tipos de instancia evaluados. Este resultado confirma que el operador de reparación por capacidad garantiza la factibilidad de todas las evaluaciones del enjambre incluso ante la mayor heterogeneidad de estructuras del problema.

4.2. Calidad de solución y patrón de brecha de integralidad (integrality gap)

La Tabla 3 presenta los resultados comparativos de calidad de solución. La Figura 12 complementa la tabla con una visualización en boxplot del GAP PSO vs. LP bound por escala y tipo de instancia, evidenciando la dispersión relativa entre configuraciones de diferente complejidad. El hallazgo central es la correlación inversa entre la brecha de integralidad del problema MILP y la calidad del PSO, observable en todas las corridas y las tres escalas.

Tabla 3. Resultados comparativos de calidad de solución (media ± std). Celdas verdes: GAP PSO/MILP < 3.5%

Escala	Tipo instancia	MILP GAP LP (%)	GAP PSO/LP (%)	GAP PSO/MILP (%)	Consist P (%)	Consist DC (%)	Infactibilidad
S	Estándar	2.45	3.26 ± 1.48	0.79 ± 0.99	93.3	90.0	0.000
M	Estándar	2.37	3.56 ± 2.02	1.16 ± 1.72	94.4	91.1	0.000
M	Tight capacity	0.70	0.70 ± 0.16	**0.00 ± 0.00**	100.0	100.0	0.000
M	High fixed costs	2.92	5.70 ± 0.99	2.70 ± 1.09	82.2	82.2	0.000
M	Asymmetric costs	2.12	4.01 ± 2.23	1.85 ± 1.84	84.4	88.9	0.000
M	Combined hard	0.40	0.65 ± 0.39	**0.25 ± 0.41**	100.0	96.7	0.000
XL	Estándar	1.53	13.56 ± 3.75	11.85 ± 3.86	70.8	67.9	0.000
XL	Tight capacity	0.28	2.45 ± 1.24	**2.16 ± 1.26**	91.7	92.1	0.000
XL	High fixed costs	1.01	8.66 ± 0.78	7.57 ± 0.81	80.0	70.4	0.000
XL	Asymmetric costs	1.33	14.41 ± 3.77	12.91 ± 3.83	76.7	67.9	0.000

XL	Combined hard	0.25	3.40 ± 0.91	**3.15 ± 0.92**	92.5	90.8	0.000

En instancias con baja brecha de integralidad (tight capacity en M: 0.70%, combined hard en M: 0.40%), el PSO alcanzó el óptimo exacto con GAP medio de 0.00% y 0.25% respectivamente y consistencia de decisiones del 100%. En la escala XL con las mismas configuraciones (tight capacity en XL: 0.28%, combined hard en XL: 0.25%), el GAP subió a 2.16% y 3.15% con consistencia del 91–93%, siendo congruente con el límite de evaluaciones de aptitud más reducido (510 vs. 1,592–2,889 en M). En instancias de mayor complejidad combinatoria (standard y asymmetric en XL), el GAP superó el 11%, lo que indica que 510 evaluaciones de aptitud son insuficientes para explorar adecuadamente un espacio binario de 90 variables.

La Figura 12 presenta esta distribución mediante boxplots separados por escala y tipo de complejidad. Se observa que las instancias combined_hard y tight_capacity exhiben la menor dispersión y los valores de GAP más bajos en ambas escalas, mientras que high_fixed_costs muestra la mayor variabilidad en escala M. La Figura 13 complementa este análisis mostrando el heatmap de la escala M, que separa el efecto de la estructura de la instancia (eje Y) del efecto de la aleatoriedad del algoritmo (eje X). Por ejemplo, para la instancia 2 (Inst-2), el GAP es estable en 1.8% independientemente de la semilla PSO, mientras que las instancias 1 y 3 presentan mayor variabilidad con PSO-101, sugiriendo sensibilidad a la inicialización en esas estructuras específicas.

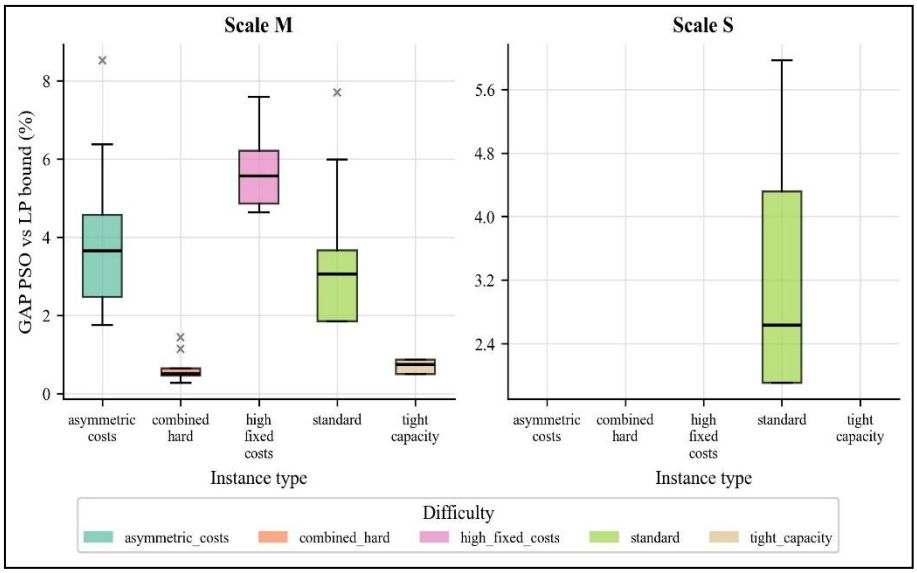

Figura 12. Distribución del GAP PSO vs. cota LP por tipo de instancia en escalas S y M (9 corridas por tipo). Las instancias combined_hard y tight_capacity exhiben los menores GAP, coherente con su baja brecha de integralidad

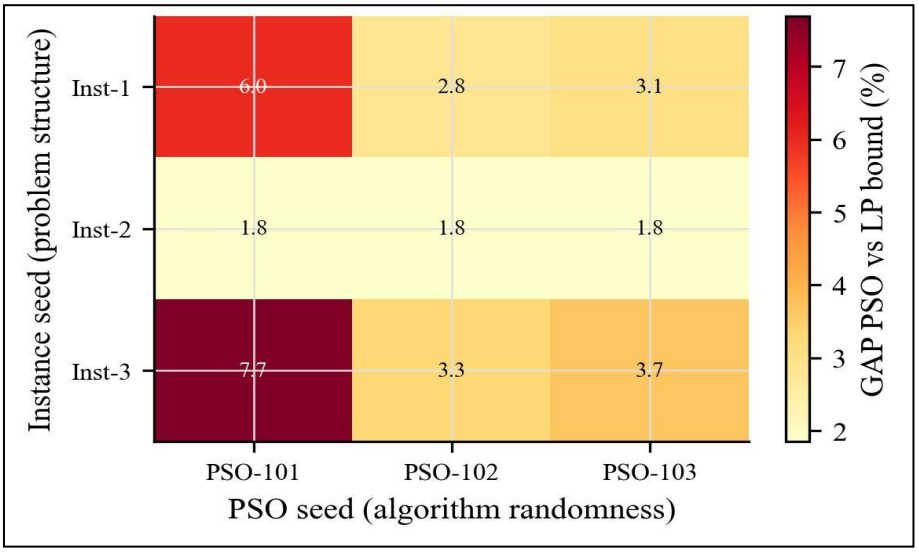

Figura 13. Mapa de calor del GAP PSO vs. cota LP (%) en instancias estándar de escala M. El eje vertical refleja el efecto de la estructura del problema y el horizontal la aleatoriedad del algoritmo (semilla PSO)

4.3. Eficiencia computacional

La Tabla 4 muestra los tiempos de cómputo por escala y tipo de instancia para todos los experimentos. Las Figuras 14 y 15 ilustran visualmente una comparativa de tiempos usando una escala logarítmica para las escalas S/M y XL respectivamente, esto permite identificar las diferencias entre MILP y PSO.

Tabla 4. Tiempos de cómputo: MILP exacto vs. PSO híbrido. Celdas amarillas: ratio PSO/MILP < 30×.

Escala	Tipo instancia	MILP $\bar{t}$ (s)	MILP t_max (s)	PSO $\bar{t}$ (s)	PSO t_max (s)	Ratio PSO/MILP
S	Estándar	0.14	0.21	92.2	138.5	646×
M	Estándar	0.41	0.65	328.0	492.4	811×
M	Combined hard	0.19	0.36	138.6	169.1	734×
XL	Estándar	15.6	25.1	432.8	436.3	**28×**
XL	Tight capacity	2.1	2.4	500.1	556.4	234×
XL	High fixed costs	20.0	24.6	546.4	561.3	**27×**
XL	Asymmetric costs	3.8	4.0	564.0	566.2	148×
XL	Combined hard	2.3	2.6	612.1	642.6	264×

Para las escalas S y M, la Figura 14 muestra que el solver CBC es 646–811× más rápido que el PSO. No obstante, la Figura 15 revela que en escala XL el ratio se reduce a ~60× (MILP: 8.8 s media vs. PSO: 531 s media). Esto describe la zona de transición hacia donde la metaheurística se vuelve computacionalmente competitiva. Los resultados muestran que el enfoque propuesto puede ser escalable, ya que tiene una tendencia de reducción del ratio con la escala, de 811× en M a <30× en las instancias XL más difíciles para el optimizador.

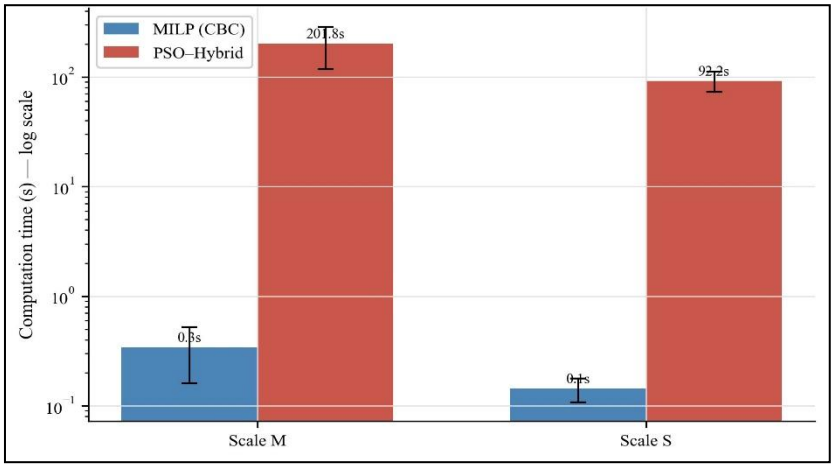

Figura 14. Comparación de tiempos de cómputo MILP (CBC) vs. PSO híbrido en escala logarítmica para las escalas S y M. CBC supera al PSO (ratios 646× y 811× respectivamente)

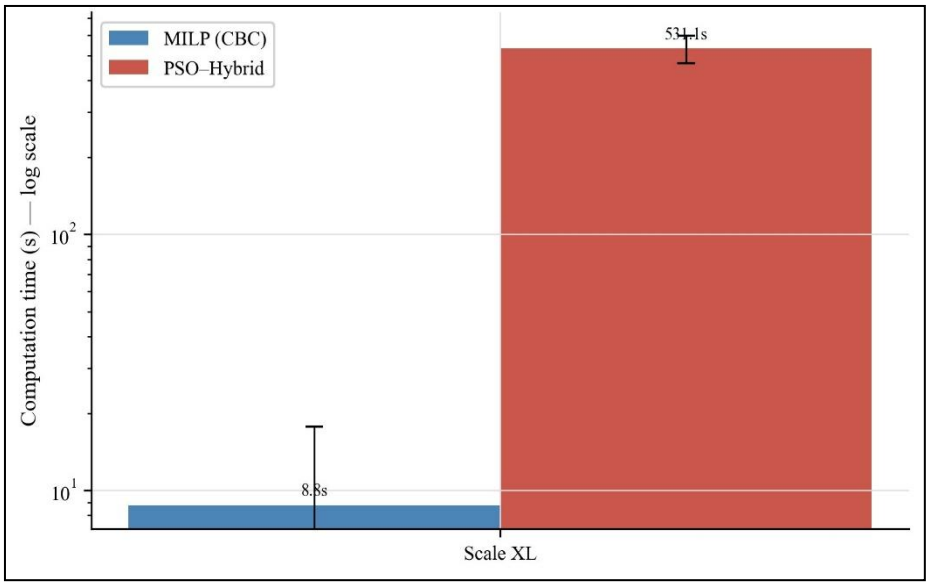

Figura 15. Comparación de tiempos de cómputo MILP vs. PSO híbrido para la escala XL. El ratio PSO/MILP cae a ~60

4.4. Comportamiento del enjambre

La Figura 16 muestra las curvas de convergencia normalizadas (GAP vs. cota LP) para las escalas S y M. En la escala S la convergencia es rápida y el GAP medio estabiliza alrededor del 3.5% en las primeras 15 iteraciones. En la escala M la convergencia es más lenta y el GAP medio se mantiene alrededor del 40–45%, lo que refleja que el promedio incluye instancias de diferentes tipos de complejidad. Un alto gap (high_fixed_costs, asymmetric) elevan el promedio. La Figura 17 muestra la convergencia para escala XL: el GAP medio decrece de ~52% en la iteración 1 a ~43% en la iteración 50, confirmando que el enjambre seguía mejorando al agotar el límite de evaluaciones de aptitud.

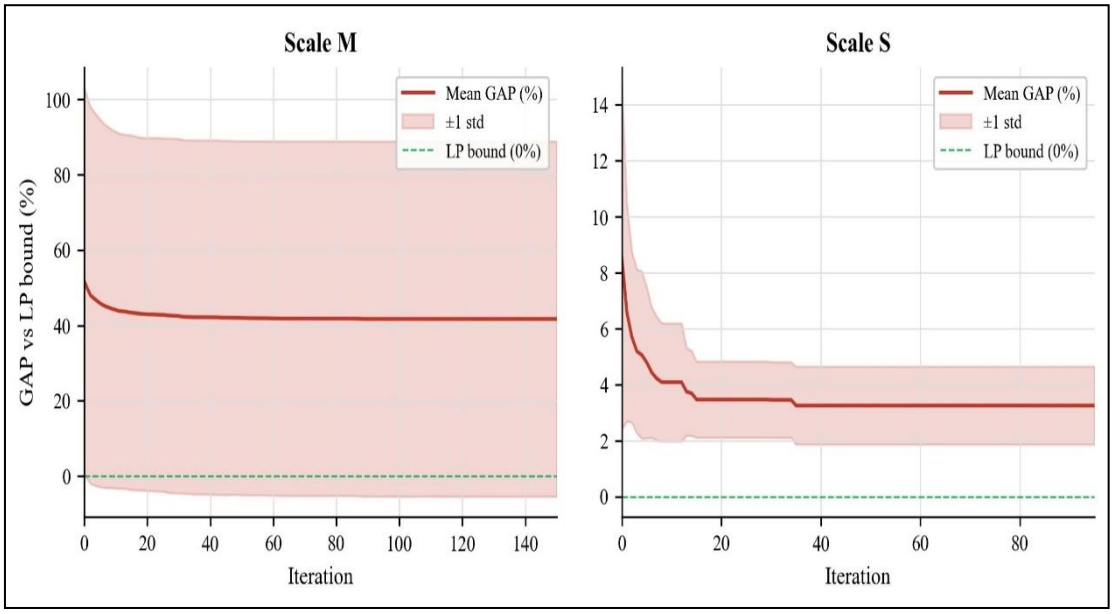

Figura 16. Curvas de convergencia del PSO híbrido para escalas S y M (media ± 1 std). La línea punteada indica la cota LP (GAP = 0%). La convergencia en escala S es notablemente más rápida que en M

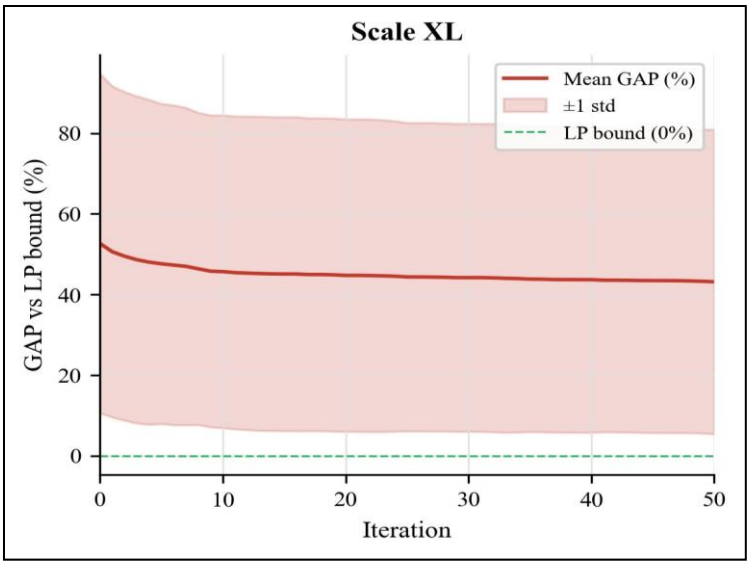

Figura 17. Curvas de convergencia del PSO híbrido para escala XL (media ± 1 std, 20 corridas). El GAP medio decrece de ~52% a ~43% en 50 iteraciones sin alcanzar el criterio de estancamiento

4.5. Análisis de escalabilidad: comparación integrada S/M/XL

La Figura 18 muestra el boxplot de GAP para la escala XL por tipo de instancia. Esta información complementa la Tabla 3 con la distribución completa. Se observa que combined_hard y tight_capacity exhiben los GAP más bajos (mediana ~3.5–3.6%). Esto es coherente con su baja brecha de integralidad, mientras que asymmetric_costs y standard muestran las mayores medianas (~13–15%).

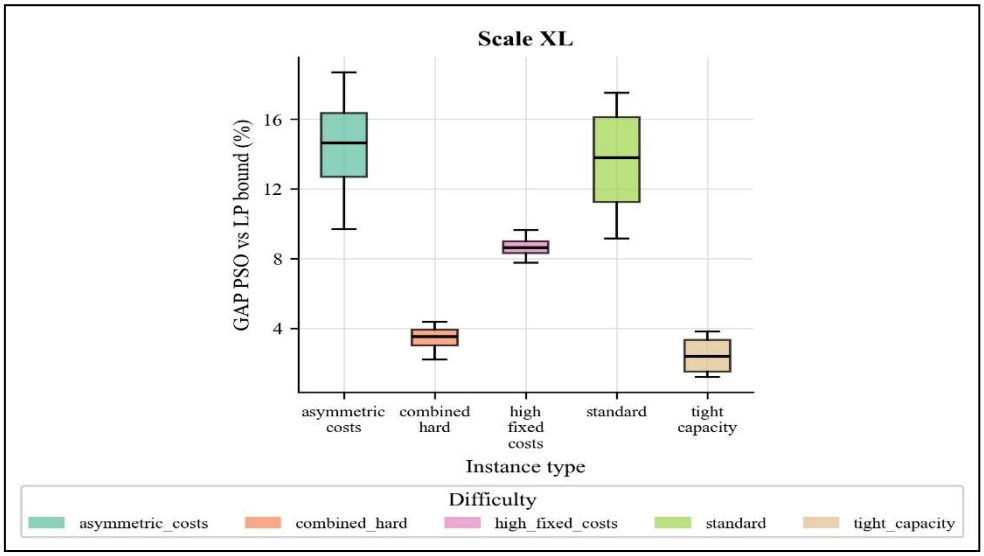

Figura 18. Distribución del GAP PSO vs. cota LP por tipo de instancia en escala XL (20 corridas). Los tipos combined_hard y tight_capacity exhiben las menores medianas (~3.5%); asymmetric_costs y standard las mayores (~13-15%)

5. DISCUSIÓN

5.1. La hibridación LP como mecanismo adaptativo

El hallazgo más relevante obtenido en la experimentación fue la correlación sistemática entre brecha de integralidad y calidad PSO, esta se observa en las 74 corridas y los tres órdenes de magnitud de tamaño de problema, ver la Figura 12 para escalas S/M y la Figura 18 para XL. Esta correlación es consecuencia directa del mecanismo de hibridación, es decir, cuando el gap es bajo, la señal LP discrimina con alta resolución entre configuraciones binarias de diferente calidad y el enjambre converge rápidamente (Figuras 16 y 17). Cuando el gap es alto, la señal LP es menos efectiva. Esto justifica la hibridación propuesta de enfoques basados en funciones de aptitud heurísticas (Noel & Jannett, 2004; Tirkolaee et al., 2020). La implicación práctica para el gemelo digital es que el sistema puede calcular la brecha de integralidad antes de invocar el PSO y usarla como predictor de la calidad esperada, guiando la asignación del límite de evaluaciones.

5.2. Calidad topológica: el gemelo digital refleja la red óptima

La consistencia de decisiones de apertura supera el 90% incluso en escala XL para instancias de baja brecha de integralidad, ver Tabla 3. Esto significa que la propiedad openStatus escrita en el grafo ADT (Figura 3) es topológicamente correcta en más del 90% de los nodos de la red, garantizando que el estado de decisiones del gemelo es fiable para la operación subsecuente. Este resultado valida los beneficios documentados de los DT para la resiliencia de cadenas de suministro (Ivanov & Dolgui, 2021; Longo et al., 2023; Marmolejo-Saucedo, 2020). Sin embargo, en instancias de mayor complejidad (XL-standard, XL-asymmetric), la concordancia cae al 68–77%, subrayando la importancia del mecanismo de refinamiento MILP exacto como extensión futura de este trabajo.

La Figura 19 muestra la arquitectura del grafo (gemelo). Las propiedades del gemelo P-01 en Azure Digital Twins Explorer, con openStatus = Verdadero junto a los parámetros que el motor de ejecución leyó al inicio del ciclo (fixedCost: $4,200, productionCap: 300 t, variableCost: $12.5/t). El valor de openStatus fue calculado por el PSO, enviado al API de Azure como una operación JSON Patch y mostrado en el grafo. La información fluye del optimizador al gemelo. Lo anterior ubica

esta implementación en la categoría Digital Twin (bidireccional) de la taxonomía de (Kritzinger et al., 2018), a diferencia de un Digital Shadow donde el flujo es unidireccional.

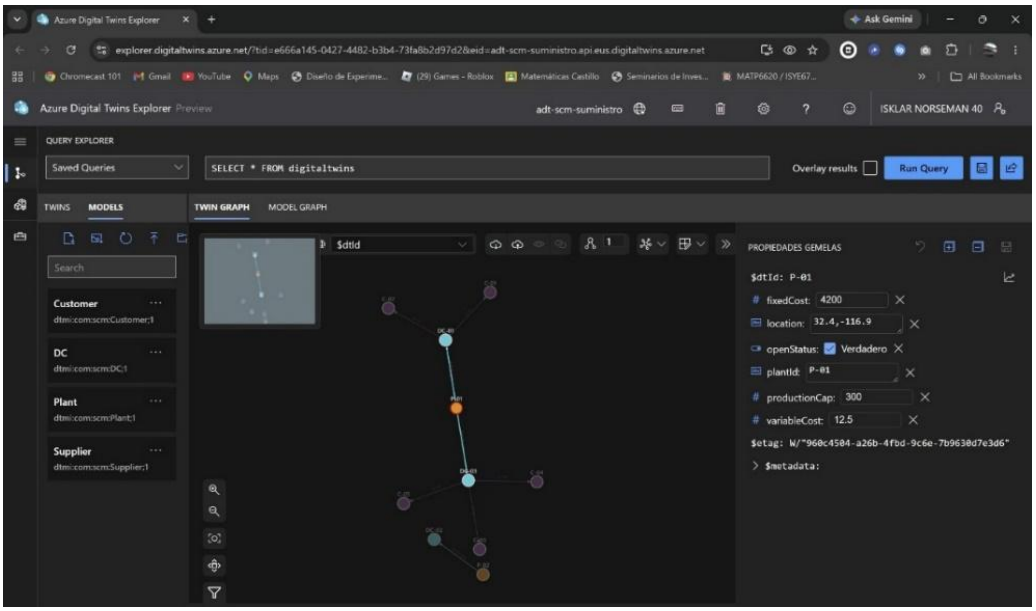

Figura 19. Propiedades del gemelo P-01 en Azure Digital Twins Explorer: openStatus = Verdadero escrito por el PSO híbrido mediante update_digital_twin(), junto con fixedCost ($4,200), productionCap (300 t) y variableCost ($12.5/t) leídos desde el grafo ADT al inicio del ciclo

5.3. Escalabilidad y zona de transición competitiva

Las Figuras 14 y 15 muestran que el ratio PSO/MILP calculado sobre los tiempos medios agregados por escala, decrece de 811× en M-estándar a ~60× en XL. Esta reducción describe la zona de transición hacia donde la metaheurística se vuelve computacionalmente competitiva. Sin embargo, la literatura reporta que instancias con más de 200 variables binarias pueden requerir incluso horas de cómputo sin garantía de optimalidad (Avella et al., 2023; Melo et al., 2009). La experiencia previa en gemelos digitales de gran escala (Marmolejo-Saucedo, 2022) sugiere que instancias de escala XXL en entornos industriales reales requerirían necesariamente la combinación PSO+ADT para mantener tiempos de respuesta operacionalmente válidos.

5.4. Comportamiento del enjambre y presupuesto óptimo

La Figura 17 muestra que en la escala XL el enjambre seguía mejorando al agotar el límite de 50 iteraciones (best_iter medio = 44.8 para XL-standard), a diferencia de XL-combined_hard donde la convergencia se produce tempranamente (best_iter = 12.0). La diversidad inicial del enjambre XL (>20 en la iteración 0) es considerablemente mayor que en M (~12) y S (~8), reflejando el mayor espacio de búsqueda. Este comportamiento sugiere que el límite recomendado para XL sería de al menos 150 iteraciones con 30 partículas (equivalente al configurado en escala M), produciendo GAPs comparables a los observados en M para los mismos tipos de instancia.

5.5. Limitantes

En todas las escalas evaluadas CBC certifica la optimalidad en 0.14–20.0 s, condición bajo la cual el PSO no presenta ventaja en tiempo de cómputo. La configuración XL con 510 evaluaciones alcanza el límite de iteraciones sin convergencia por estancamiento en instancias de mayor gap. No se consideró el efecto de la latencia que se puede generar a escala industrial. La validación de la integración ADT solo se realizó sobre instancias de demostración de escala reducida. Todas las

instancias son sintéticas; la validación con datos reales de cadenas industriales se plantea como trabajo futuro.

FINANCIAMIENTO

Los autores no recibieron ningún patrocinio para llevar a cabo este estudio-artículo.

CONFLICTO DE INTERESES

No existe ningún tipo de conflicto de interés relacionado con la materia del trabajo.

CONTRIBUCIÓN DE AUTORES

Conceptualización: Marmolejo-Saucedo, J. A.; Rodríguez-Aguilar, R.; Mendoza, A. Curación de datos: Rodríguez-Aguilar, R.; Mendoza, A. Análisis formal: Marmolejo-Saucedo, J. A.; Rodríguez-Aguilar, R. Investigación: Marmolejo-Saucedo, J. A.; Rodríguez-Aguilar, R.; Mendoza, A. Metodología: Marmolejo-Saucedo, J. A.; Rodríguez-Aguilar, R.; Mendoza, A. Administración del proyecto: Marmolejo-Saucedo, J. A. Software: Rodríguez-Aguilar, R.; Mendoza, A. Validación: Marmolejo-Saucedo, J. A.; Mendoza, A. Visualización: Rodríguez-Aguilar, R.; Mendoza, A. Redacción – borrador original: Marmolejo-Saucedo, J. A.; Rodríguez-Aguilar, R.; Mendoza, A. Redacción – revisión y edición: Marmolejo-Saucedo, J. A.; Rodríguez-Aguilar, R.; Mendoza, A.

REFERENCIAS

Avella, P., Calamita, A., & Palagi, L. (2023). A computational study of off-the-shelf MINLP solvers on a benchmark set of congested capacitated facility location problems. *ArXiv*. https://doi.org/10.48550/arXiv.2303.04216

Dominguez Martinez, V., Parody De La Cruz, J. D., Guerra Manjarrez, J. F., & Rodriguez Velandia, D. A. (2026). *Digital twins in healthcare (2018–2024): A scientometric analysis with emphasis on cardiology and perspectives for Colombia*. En M. Zuluaga (Ed.), *Scientometric Perspectives on Biomedical Research: Applications Across Oncology, Infectious Diseases, Bioinformatics, and Medical Technologies* (Vol. 1, pp. 33-50). Contemporary Biomedical Research. Pro-Metrics. https://doi.org/10.47909/978-9916-9331-7-6.5

Farahani, R. Z., Rezapour, S., Drezner, T., & Fallah, S. (2014). Competitive supply chain network design: An overview of classifications, models, solution techniques and applications. *Omega*, *45*, 92–118. https://doi.org/10.1016/j.omega.2013.08.006

Geoffrion, A. M., & Graves, G. W. (1974). Multicommodity Distribution System Design by Benders Decomposition. *Management Science*, *20*(5), 822–844. https://doi.org/10.1287/mnsc.20.5.822

Grieves, M. (2014). *Digital Twin: Manufacturing Excellence through Virtual Factory Replication*.

Grieves, M., & Vickers, J. (2017). Digital Twin: Mitigating Unpredictable, Undesirable Emergent Behavior in Complex Systems. In F.-J. Kahlen, S. Flumerfelt, & A. Alves (Eds.), *Transdisciplinary Perspectives on Complex Systems: New Findings and Approaches* (pp. 85–113). Springer International Publishing. https://doi.org/10.1007/978-3-319-38756-7_4

Guo, D., & Mantravadi, S. (2025). The role of digital twins in lean supply chain management:

review and research directions. *International Journal of Production Research*, *63*(5), 1851–1872. https://doi.org/10.1080/00207543.2024.2372655

Hart, W. E., Watson, J.-P., & Woodruff, D. L. (2011). Pyomo: Modeling and solving mathematical programs in Python. *Mathematical Programming Computation*, *3*(3), 219–260. https://doi.org/10.1007/s12532-011-0026-8

Ivanov, D., & Dolgui, A. (2019). The impact of digital technology and Industry 4.0 on the ripple effect and supply chain risk analytics. *International Journal of Production Research*, *57*(3), 829–846. https://doi.org/10.1080/00207543.2018.1488086

Ivanov, D., & Dolgui, A. (2021). A digital supply chain twin for managing the disruption risks and resilience in the era of Industry 4.0. *Production Planning & Control*, *32*(9), 775–788. https://doi.org/10.1080/09537287.2020.1768450

Kennedy, J., & Eberhart, R. (1995). Particle swarm optimization. In *Proceedings of ICNN'95 - International Conference on Neural Networks* (Vol. 4, pp. 1942–1948). IEEE. https://doi.org/10.1109/ICNN.1995.488968

Kennedy, J., & Eberhart, R. C. (1997). A discrete binary version of the particle swarm algorithm. In *1997 IEEE International Conference on Systems, Man, and Cybernetics: Computational Cybernetics and Simulation* (Vol. 5, pp. 4104–4108). IEEE. https://doi.org/10.1109/ICSMC.1997.637339

Kritzinger, W., Karner, M., Traar, G., Henjes, J., & Sihn, W. (2018). Digital Twin in manufacturing: A categorical literature review and classification. *IFAC-PapersOnLine*, *51*(11), 1016–1022. https://doi.org/10.1016/j.ifacol.2018.08.474

Lim, K. Y. H., Zheng, P., & Chen, C.-H. (2020). A state-of-the-art survey of Digital Twin: techniques, engineering product lifecycle management and business innovation perspectives. *Journal of Intelligent Manufacturing*, *31*(6), 1313–1337. https://doi.org/10.1007/s10845-019-01512-w

Longo, F., Mirabelli, G., Padovano, A., & Solina, V. (2023). The Digital Supply Chain Twin paradigm for enhancing resilience and sustainability against COVID-like crises. *Procedia Computer Science*, *217*, 1940–1947. https://doi.org/10.1016/j.procs.2022.12.394

Marmolejo-Saucedo, J. A. (2020). Design and Development of Digital Twins: A Case Study in Supply Chains. *Mobile Networks and Applications*, *25*(6), 2141–2160. https://doi.org/10.1007/s11036-020-01557-9

Marmolejo-Saucedo, J. A. (2022). Digital Twin Framework for Large-Scale Optimization Problems in Supply Chains: A Case of Packing Problem. *Mobile Networks and Applications*, *27*(5), 2198–2214. https://doi.org/10.1007/s11036-021-01856-9

Melo, M. T., Nickel, S., & Saldanha-da-Gama, F. (2009). Facility location and supply chain management - A review. *European Journal of Operational Research*, *196*(2), 401–412. https://doi.org/10.1016/j.ejor.2008.05.007

Microsoft. (2023). Digital Twins Definition Language (DTDL), Version 3. In *Azure Open Digital Twins*. https://azure.github.io/opendigitaltwins-dtdl/DTDL/v3/DTDL.v3.html

Microsoft. (2025a). Azure Digital Twins Core client library for Python (azure-digitaltwins-core).

In *Python Package Index (PyPI)* (1.3.0). Python Software Foundation. https://pypi.org/project/azure-digitaltwins-core/

Microsoft. (2025b). What is an ontology in Azure Digital Twins? In *Microsoft Learn*. https://learn.microsoft.com/en-us/azure/digital-twins/concepts-ontologies

Microsoft. (2025c). What is Azure Digital Twins? In *Microsoft Learn*. https://learn.microsoft.com/en-us/azure/digital-twins/overview

Milenkovic, M. (2022). Internet of Things: System Reference Architecture. *ArXiv*. https://doi.org/10.48550/arXiv.2204.01872

Noel, M. M., & Jannett, T. C. (2004). Simulation of a new hybrid particle swarm optimization algorithm. In *Proceedings of the Thirty-Sixth Southeastern Symposium on System Theory* (pp. 150–153). IEEE. https://doi.org/10.1109/SSST.2004.1295638

Pérez-Baquero, R.-S., Murgas Gutierrez, J. P., Arzuaga Gomez, A. C., & Suarez Ravelo, C. D. (2026). Decision technologies in control engineering: A scientometric review of LQR control for inverted pendulum systems (2006–2025). *DecisionTech Review*, *6*, 1–14. https://doi.org/10.47909/dtr.36

Shi, Y., & Eberhart, R. C. (1998). A modified particle swarm optimizer. In *1998 IEEE International Conference on Evolutionary Computation Proceedings; IEEE World Congress on Computational Intelligence* (pp. 69–73). IEEE. https://doi.org/10.1109/ICEC.1998.699146

Simchi-Levi, D., Kaminsky, P., & Simchi-Levi, E. (2008). *Designing and Managing the Supply Chain: Concepts, Strategies, and Case Studies* (3rd ed.). McGraw-Hill/Irwin.

Somma, A., Amalfitano, D., De Benedictis, A., & Pelliccione, P. (2025). TwinArch: A digital twin reference architecture. *Journal of Systems and Software*, 112613. https://doi.org/10.1016/j.jss.2025.112613

Tao, F., Zhang, H., Liu, A., & Nee, A. Y. C. (2019). Digital Twin in Industry: State-of-the-Art. *IEEE Transactions on Industrial Informatics*, *15*(4), 2405–2415. https://doi.org/10.1109/TII.2018.2873186

Tirkolaee, E. B., Goli, A., Faridnia, A., Soltani, M., & Weber, G.-W. (2020). Multi-objective optimization for the reliable pollution-routing problem with cross-dock selection using Pareto-based algorithms. *Journal of Cleaner Production*, *276*, 122927. https://doi.org/10.1016/j.jclepro.2020.122927

Wasi, A. T., Anik, M. A., Rahman, A., Hoque, M. I., Islam, M. D. S., & Ahsan, M. M. (2025). A Theoretical Framework for Graph-based Digital Twins for Supply Chain Management and Optimization. *ArXiv*. https://doi.org/10.48550/arXiv.2504.03692