RESOLUCIÓN DEL PROBLEMA DEL TAMAÑO DE LOTE MULTINIVEL CAPACITADO APLICANDO OPTIMIZACIÓN POR ENJAMBRE DE PARTÍCULAS CON BUSQUEDA LOCAL

JAIME MANUEL PADRÓN CANO

UNIVERSIDAD TECNOLÓGICA DE BOLÍVAR
FACULTAD DE INGENIERÍAS
PROGRAMA DE INGENIERÍA DE SISTEMAS
CARTAGENA DE INDIAS D.T. Y C.
JUNIO DE 2012

RESOLUCIÓN DEL PROBLEMA DEL TAMAÑO DE LOTE MULTINIVEL CAPACITADO APLICANDO OPTIMIZACIÓN POR ENJAMBRE DE PARTÍCULAS CON BUSQUEDA LOCAL

JAIME MANUEL PADRÓN CANO

Director del trabajo:
WILLIAM CAICEDO TORRES
INGENIERO DE SISTEMAS

UNIVERSIDAD TECNOLÓGICA DE BOLÍVAR
FACULTAD DE INGENIERÍAS
PROGRAMA DE INGENIERÍA DE SISTEMAS
CARTAGENA DE INDIAS D.T. Y C.
JUNIO DE 2012

Cartagena de Indias D. T y C. Junio 27 de 2012

Señores:
COMITÉ CURRICULAR DE EVALUACIÓN DE PROYECTOS
Programa de Ingeniería de Sistema
Facultad de ingeniería
Universidad Tecnológica de Bolívar
La ciudad

Respetados Señores:

Por medio de la presente me permito someter a su consideración la propuesta de Trabajo de grado titulado **"RESOLUCIÓN DEL PROBLEMA DEL TAMAÑO DE LOTE MULTINIVEL CAPACITADO APLICANDO OPTIMIZACIÓN POR ENJAMBRE DE PARTÍCULAS CON BUSQUEDA LOCAL"**, desarrollada por Jaime Manuel Padrón Cano, como requisito para optar al título de Ingeniería de Sistemas, en la que me desempeñaré cumpliendo la función de director.

Atentamente,

_____

WILLIAM CAICEDO TORRES
Dir. Del Trabajo de grado

Cartagena de Indias D. T y C. Junio 27 de 2012

Señores:
Comité curricular del programa de Ingeniería de Sistemas de la Universidad Tecnológica de Bolívar
La ciudad

Cordial saludo:

Por medio de la presente me permito someter a su consideración la propuesta de Trabajo de grado titulado **"RESOLUCIÓN DEL PROBLEMA DEL TAMAÑO DE LOTE MULTINIVEL CAPACITADO APLICANDO OPTIMIZACIÓN POR ENJAMBRE DE PARTÍCULAS CON BUSQUEDA LOCAL"**, desarrollada por Jaime Manuel Padrón Cano, como requisito para optar al título de Ingeniería de Sistema.

Agradeciendo de antemano su atención.

Atentamente,

_____

Jaime Manuel Padrón Cano
Estudiante de Ingeniería de Sistema de la Universidad Tecnológica de Bolívar.
CC No. 1.047.418.708 de Cartagena

Nota de aceptación:

_____

_____

_____

_____

Firma del presidente del jurado

_____

Firma jurado

_____

Firma jurado

DEDICATORIA

A Dios por brindarme la vida tan maravillosa que tengo y darme la oportunidad de estudiar en una gran Universidad, a mi familia por todo el apoyo y el esfuerzo para estudiar estos 6 años dedicado plenamente a mis dos carreras y ayudando a explotar mis capacidades para convertirme en una mejor persona y por último, a una persona que siempre creyó en mí y me motivo a seguir adelante, Melissa B.

Jaime Manuel Padrón Cano

AGRADECIMIENTOS

El autor expresa sus agradecimientos a:

Profesor Jairo Coronado, Ingeniero Industrial y al profesor William Caicedo, Ingeniero de Sistemas, por su gran apoyo para la realización de este trabajo, además por su tiempo y buenos consejos.

Al cuerpo docente del programa de Ingeniería de Sistemas de la Universidad Tecnológica de Bolívar, por convertir mi estadía en la Universidad, una de mis mejores experiencias en la vida.

CONTENIDO

Pág.

LISTA DE FIGURAS

Pág.

LISTA DE TABLAS

Pág.

LISTA DE FORMULAS Y ECUACIONES

Pág.

LISTA DE ANEXOS

Pág.

INTRODUCCIÓN

Un problema que se presentan en los sistemas de manufactura de las empresas, especialmente pequeñas y medianas empresas, es que la programación de la producción está basada bajo modelos de arrastre de la demanda deterministas, es decir, el proceso de la planificación de los tamaños de lotes de insumos o componentes para la fabricación de productos finales que poseen jerarquías multiniveles y restricciones de capacidad como horas hombres, números de maquinas entre otras, son un problema que se pueden convertir en oportunidades de mejora debido a que se pueden equilibran los costos de ordenamiento de un lote y los costos de inventarios por productos que permita obtener una reducción en los costos totales y así mejorar la productividad de las empresas.

Este trabajo presenta una metodología para la solución del problema del tamaño de lote multinivel capacitado, basado en una técnica metaheurísticas llamada optimización por enjambre de partículas o PSO por sus siglas en ingles (*Particle Swarm Optimization*), la cual se ha demostrado que tiene un buen desempeño dentro de las familia de metaheurísticas, así que se propone el desarrollo de este tema agregando el concepto una búsqueda local que permita generar óptimos locales y permita mejorar las soluciones encontradas por las partículas, denominando a está técnica como la utilización de una hiperheurística.

Con base a la descripción general del problema en cuestión se plantea en el primer capítulo realizar un diagnostico mediante la revisión de la literatura para identificar las variables críticas y definir las estrategias que serán utilizadas para el diseño y adaptación del algoritmo a una naturaleza combinatoria, posteriormente en el segundo capítulo se plantea el modelo matemático y el esquema del algoritmo para el problema capacitado y no capacitado con base a la estrategia

definida previamente, además se expone la base teórica para la utilización de la búsqueda local y cómo será la implementación de ésta con el fin encontrar optimas locales que permitan mejorar las soluciones, por último se define el caso de estudio que servirá para probar el algoritmo y los requerimientos básicos que debe implementar la herramienta computacional para lograr este objetivo.

En el tercer capítulo se presenta el desarrollo del alcance propuesto en presente trabajo mediante la descripción de los requerimientos funcionales y diseño de la arquitectura del software para la construcción de una librería en el lenguaje de programación Java que implemente el algoritmo PSO con búsqueda local.

Finalmente en el último capítulo se presentan el análisis y los resultados obtenidos por la herramienta computacional luego de desarrollar el caso de estudio en el cual se evalúan los tiempos de ejecución y el desempeño del algoritmo para resolver el caso de estudio frente a las otras técnicas trabajadas en la literatura.

1. OBJETIVOS

1.1 OBJETIVO GENERAL

Resolver el problema del tamaño de lote en una jerarquía de productos multinivel con restricciones de capacidad aplicando la técnica de la optimización por enjambre de partículas con búsqueda local.

1.2 OBJETIVOS ESPECIFICOS

- ✓ Realizar un diagnóstico al problema del tamaño de lote con jerarquía de productos multinivel con restricciones de capacidad, para identificar las variables más importantes que afectan el contexto del problema y definir las estrategias para el diseño de la adaptación del algoritmo PSO, mediante la revisión de literatura especializada.

- ✓ Establecer el modelo matemático que corresponda al problema, basado en las implementaciones propuestas por los autores en la literatura, con el fin de estructurar el esquema que sirva para desarrollar el algoritmo PSO con búsqueda local.

- ✓ Implementar el algoritmo PSO con búsqueda local, mediante la creación de una librería utilizando el lenguaje de programación Java, teniendo como base la arquitectura del software desarrollada a partir de los requerimientos funcionales del sistema.

- ✓ Analizar e interpretar los resultados encontrados por la librería sobre el caso de estudio propuesto, mediante la comparación del uso de técnicas determinísticas y el algoritmo PSO, determinando así el grado de eficiencia de las soluciones.

2. PROBLEMA DE INVESTIGACIÓN

2.1 DESCRIPCIÓN DEL PROBLEMA

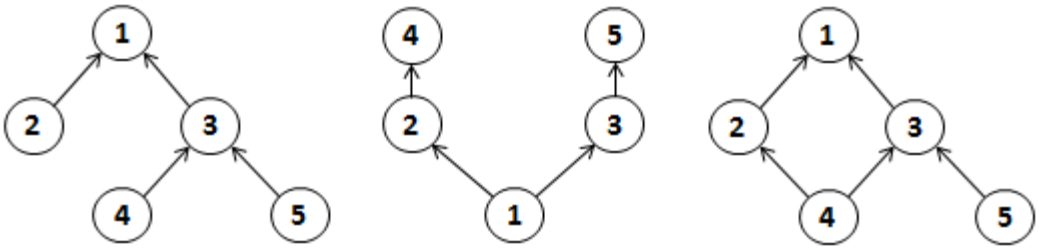
Dentro de las empresas, especialmente manufactureras dedicadas a la producción por lote, existe un problema relacionado al cómo se debe planificar la adquisición de los recursos para producir, teniendo en cuenta variables como costos, tiempo de despacho, inventario de materiales entre otras, esto se define detalladamente dentro de la ciencia detrás de la Planeación de Requerimientos de los Materiales (MRP) que fundamentalmente está definida para establecer el producto adecuado, en la cantidad adecuada y en el momento adecuado.

En consecuencia, surge un problema específico de la búsqueda para obtener el tamaño de lote óptimo para un esquema de producción basado en múltiples productos y en múltiples niveles, este es conocido como el problema del Tamaño de lote multinivel o MLLS por sus siglas en ingles (*Multi-Level lot-sizing*) hace parte de los problemas clásicos de MRP y para resolver este problema, los diferentes autores en la literatura han adoptado diferentes técnicas (Algoritmos Genéticos, Recocido Simulado, Optimización por Colonia de Hormigas); Dellart, et al. (2000) presentaron un algoritmo genético para resolver un problema de estructura general no capacitado; En el 2004, Tang adaptó la técnica del recocido simulado al problema con una estructura en serie no capacitado; Pitakaso et al. (2007) presentaron un sistema de hormigas Max–Min para problemas de estructura general no capacitados; En el 2009, Han, et al. realizaron implementaciones de la técnica de optimización por enjambre de partículas para problemas de tamaño de lote multinivel no capacitados; En el año 2011, Xiao, et al. presentaron la técnica de búsqueda variable de vecinos; en consecuencia gracias a estas técnicas metaheurísticas que están siendo aplicadas con éxito a diversos problemas de esta naturaleza (problemas de difícil solución) se pueden

mejorar los resultados puesto que éstas hacen una búsqueda en profundidad del espacio de soluciones con un esfuerzo computacional aceptable (ÁLVAREZ, 2002) a diferencia de la programación lineal que requiere un mayor esfuerzo computacional y no siempre encuentra los óptimos globales de un problema.

Con base a esto se pueden establecer las variables del problema en cuestión, una de ellas es la clasificación según diferentes categorías acorde a la estructura del producto (Ej.: sistema de un nivel, multinivel, en forma de árbol, de ensamble o generales) (Fig. 1), como también, tanto en el MRP y la Planificación de los requerimientos de manufactura (MRPII), la capacidad es un factor que es tenido en cuenta por la Planificación de los Requerimientos de Capacidad (CRP), se pueden definir varios esquemas (Ej. No capacitado, capacitado con un único recurso y capacitado con múltiples recursos) lo cual aumentaría la complejidad del modelo (GRAVES, 2007), es por eso que el objetivo fundamental de este problema es encontrar el tamaño de lote óptimo a producir en cada nivel tal que minimice el costo fijo total y determine el inventario mínimo a reservar, así que debido a estas consideraciones se puede identificar a este tipo de problema de optimización como un problema de naturaleza combinatoria o un problema NP-Hard (GAREY, 1979).

Figura 1. Los tres tipos principales de estructuras de productos.



Fuente: Han, 2009. P. 1749

2.2 FORMULACIÓN DEL PROBLEMA

Conforme a la complejidad del problema del tamaño de lote multinivel capacitado y a la poca investigación realizada sobre este tema, se busca dar solución mediante la técnica de la optimización por enjambre de partículas, por tal motivo se genera el siguiente problema de investigación:

¿Se podrá calcular el tamaño del lote a problemas con una estructura multinivel capacitado aplicando la técnica de la optimización por enjambre de partículas con búsqueda local?

3. JUSTIFICACIÓN

El problema del tamaño de lote multinivel capacitado corresponde a una situación muy particular dentro de los sistemas de producción de las empresas, por lo general en empresas manufactureras, que juegan un papel importante debido a que puede convertirse en una oportunidad de mejora de procesos, esto con el fin de obtener una reducción de costos significativa para la empresa.

De este modo el análisis y diseño de una buena planificación de los recursos de manufactura conlleva a aumentar la productividad general pero debido a la complejidad de la solución de estos problemas, obtener la solución por medio de la programación lineal no garantiza encontrar soluciones óptimas e incluso podría tardar en el tiempo de búsqueda, en consecuencia la implementación de algoritmos metaheurísticos juegan un papel importante debido a que son técnicas que permite encontrar muy buenas soluciones en un menor tiempo lo cual aumenta el desempeño de la búsqueda.

Por consiguiente, mediante la revisión de la literatura especializada se ha encontrado muy poca investigaciones sobre el tema del Tamaño de Lote Multinivel Capacitado por lo cual se busca aplicar la técnica de la Optimización por Enjambre de Partículas o PSO por sus siglas en ingles (*Particle Swarm Optimization*) la cual es una moderna técnica computacional evolutiva que se basa en el comportamiento de la población (KENNEDY y EBERHART, 1995), que además ha tenido muy buen desempeño dentro de la categoría de los algoritmos metaheurísticos y que no se ha aplicado a este problema en particular, esto con el fin de proveer una solución a este tema.

4. ASPECTOS METODOLOGICOS

4.1 TIPO DE INVESTIGACIÓN

Para la ejecución del trabajo, se utilizará un carácter de investigación descriptivo-correlacional, debido a que la investigación descriptiva “tiene un carácter diagnostico cuando se proponen establecer relaciones causales entre distintos fenómenos” y el carácter correlacional “se ocupa de determinar la variación en unos aspectos en relación con otros. Este estudio es el indicado para organizar las relaciones estadísticas entre las características y la concentración de las causas del fenómeno estudiado. En una situación creada, explica por qué se presenta, en qué grado dos o más de sus variables están relacionadas y en qué circunstancias se produce este estado”. (LANDEAU, 2007, p. 58).

Como primera instancia en la etapa inicial se desarrolla bajo un tipo de investigación descriptiva debido a que se analiza detalladamente todas las características del problema en cuestión, establecer las variables del modelo, revisar la literatura especializada y definir los valores óptimos de los parámetros que afectan al algoritmo de Optimización por Enjambre de Partículas como también encontrar el esquema adecuado que permita adaptar el algoritmo a un problema de naturaleza combinatoria; Como segunda instancia se adoptará un tipo de investigación correlacional debido a que se busca contrastar los parámetros del algoritmo y las variables del problema para resolver el problema y determinar el desempeño mediante un cuadro comparativo entre las soluciones encontradas y los resultados encontrados por los otros autores.

El tipo de investigación aplicada al trabajo es experimental, en este diseño metodológico “el investigador no solo identifica las características que se estudian sino que las controla, las altera o manipula con el fin de observar los resultados al

tiempo que procura evitar que otros factores intervengan en la observación" (GRAJALES, 2000, p. 3) así por medio de este enfoque, revisando información sobre investigaciones en artículos, libros y material de internet relacionado a la técnica de Optimización por Enjambre de Partículas y bajo la asesoría del director de trabajo se busca retroalimentar el modelo conceptual del problema afinando los parámetros del algoritmo que permitan obtener posibles mejores soluciones.

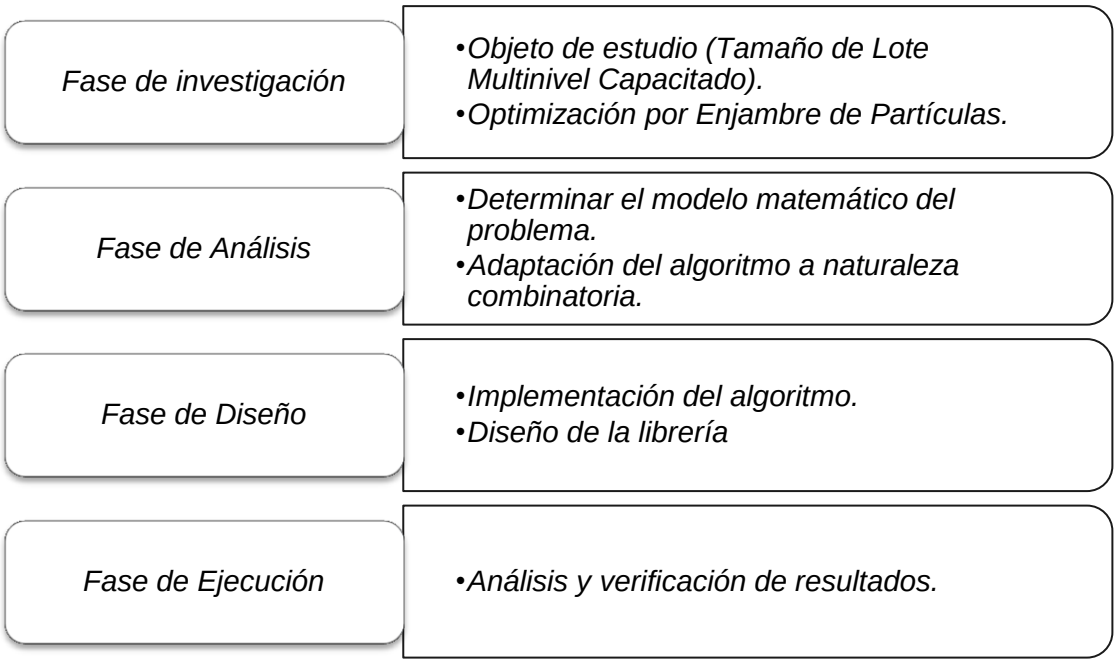
4.2 METODOLOGÍA DE TRABAJO

Para dar solución al problema del tamaño de lote multinivel capacitado aplicando optimización por enjambre de partículas se requieren de procesos secuénciales de investigación, análisis, diseño y ejecución que fundamentarán las bases de este trabajo, por lo cual se deben estructurar el trabajo en fases que determinan el camino a seguir para el logro de los objetivos propuestos.

En consecuencia se exponen las fases que corresponden al esquema metodológico, (Fig. 3) que será la base para el desarrollo de este trabajo, bajo la teoría filosófica de Georg W. Friedrich Hegel que afirma que "*la tesis es una proposición que se da por verdadera hasta el surgimiento de la antítesis, otra proposición que la contradice. El proceso dialéctico entre la tesis y la antítesis concluye con la síntesis*".[1]

1 Definición de tesis. [Citado el 3 de marzo de 2012], Disponible en: <URL: http://definicion.de/tesis/>

Figura 2. Esquema metodológico.



Fuente: Elaboración propia.

4.2.1 Fase de investigación. Consiste en revisar la bibliografía relacionada al problema del tamaño de lote multinivel capacitado con el fin de entender el contexto de este, sumado a la revisión del algoritmo de optimización por enjambre de partículas.

4.2.2 Fase de análisis. Corresponde a determinar el modelo matemático del problema con el fin formular la estructura del algoritmo, analizar las variables y parámetros del modelo, además determinar la relación entre estos.

4.2.3 Fase de diseño. Se diseña una arquitectura de software que será la base fundamental para la construcción de la herramienta computacional que permitirá el desarrollo del caso de estudio a partir de los requerimientos obtenidos en la fase de análisis.

4.2.4 Fase de ejecución. Se debe realizar un análisis de los parámetros iniciales del algoritmo para ejecutar el caso de estudio y permita obtener las soluciones al problema, al final se expondrá la síntesis correspondiente a la eficacia y eficiencia de la herramienta para obtener una solución óptima al problema y sobre el desempeño de éstas frente a las otras técnicas mediante la revisión de las baterías en la literatura.

5. MARCO REFERENCIAL

5.1 MARCO CONCEPTUAL

Teniendo en cuenta el problema en cuestión, se presentan a continuación un conjunto de conceptos generales que se observarán a lo largo del proyecto:

- ✓ **Bill Of Material (BOM).** Corresponde a la lista de los materiales que se especifican para la producción de un producto, por otro lado en el control de la producción, BOM o explosión de materiales se puede representar de manera detallada o gráfica en un árbol y se hace indispensable debido a que permite definir las especificaciones que intervienen en el producto final, mostrando así las sucesivas etapas de fabricación.

- ✓ **Estructura multinivel.** Es una característica de los sistemas de producción en el cual uno o muchos productos se distribuyen en la explosión de materiales por múltiples componentes, por ende cada conjunto de componentes que conforman a un producto representa un nivel. Las estructuras multinivel se puede representar en diferentes jerarquías sistema múltiple de dos o más niveles en forma de árbol, de ensamble o general. (Fig. 1).

- ✓ **Inteligencia de enjambre.** Es una rama de la inteligencia artificial que estudia el comportamiento colectivo y las emergentes propiedades del complejo, auto organizado y descentralizado sistema de una estructura social. Dichos sistemas consisten en una simple interacción de agentes organizados en pequeñas sociedades (Enjambres). Aunque cada agente tiene un espacio de acción muy limitado y no hay control central, el comportamiento agregado del enjambre completo exhibe rasgos de

inteligencia, es decir, la capacidad de reaccionar a los cambios ambientales y la capacidad de toma de decisión. Este comportamiento se puede apreciar en cardúmenes de pescados, bandadas de aves, colonia de hormigas y manada de animales. (PARSOPOULOS, 2010)

- ✓ **Librería.** Son denominadas como "*Packages*" o paquetes en el mundo Java, éstas satisfacen la necesidad de reutilizar ciertos componentes que ya han sido escritos, como también proveer una estructura que permita mantener el código organizado.

  Una manera de definir estas librerías es mediante archivos JAR, que son simples archivos comprimidos que incluyen una estructura de directorios con clases, la ventaja de estos es que se pueden distribuir y/o utilizar las clases de manera eficiente a través de un solo archivo.

- ✓ **Optimización.** En términos generales, un problema de optimización busca hallar una solución tal que represente la mejor manera de realizar una actividad concreta, aplicado en el contexto de la investigación de operaciones, se optimiza un problema cuando se maximizan (Utilidades, velocidad, rentabilidad, etc.) o minimizan (Costos, tiempo, riesgo, etc.) algún determinado criterio.

- ✓ **Tamaño de lote.** Corresponde al cálculo que permite determinar los requerimientos netos de los productos en lotes económicamente eficientes para la planta o el proveedor.

5.2 MARCO TEORICO

Para el desarrollo de esta investigación es necesario definir unas bases teóricas que permitan formular argumentos cohesivos y convincentes, y que además permitan confeccionar el diseño metodológico del proyecto, en consecuencia los conceptos más importantes son presentados a continuación:

5.2.1 Heurística y Metaheurísticas. El término heurística proviene del griego *heurisken*, que significa “encontrar” o “descubrir”, éstas se consideran técnicas que buscan soluciones de buena calidad a un costo computacional razonable, aunque sin garantizar la optimalidad de las mismas. En general ni siquiera se conoce el grado de error. El término metaheurísticas introducido por primera vez por Glover en el año de 1996, define a éste etimológicamente del griego meta que significa “más allá” o “nivel superior”, por lo que metaheurística corresponde a “Una heurística de nivel más superior”.

Actualmente una definición muy acertada para el término de metaheurística es expuesta por Ibrahim Osman y Gilbert Laporte (1996) quienes la definen como un proceso iterativo de generación que guía una heurística subordinada, combinando de forma inteligente distintos conceptos para explorar y explotar el espacio de búsqueda, estrategias de aprendizaje se utilizan para estructurar la información con el fin de encontrar de manera eficiente soluciones casi óptimas. Por otro lado Osman y Kelly (1996) proponen otra definición en la cual plantean que la metaheurística son técnicas o métodos aproximados diseñados para resolver problemas de optimización combinatoria, en los que los heurísticos clásicos no son efectivos. Las metaheurísticas proporcionan un marco general para crear nuevos algoritmos híbridos, combinando diferentes conceptos derivados de la inteligencia artificial, la evolución biológica y los mecanismos estadísticos, debido a esto las características más notables de éstas, es que guían el proceso de búsqueda, incluyendo como tal heurísticas subordinadas, otra característica es

que son de uso genérico, es decir, no se especifican para cada tipo de problema por lo que una ventaja frente a otros métodos es su gran flexibilidad para abordar una amplia gama de problemas.

Actualmente existen una gran variedad de Metaheurísticas, tales como el Recocido Simulado, GRASP, la Búsqueda Tabú, Algoritmo Genético, Sistemas Inmune Artificiales, Optimización por Colonia De Hormigas, Optimización por Enjambre De Partículas, Evolución Diferencial, entre otras, que en su mayoría se basan en analogías de la naturaleza para su fundamentación.

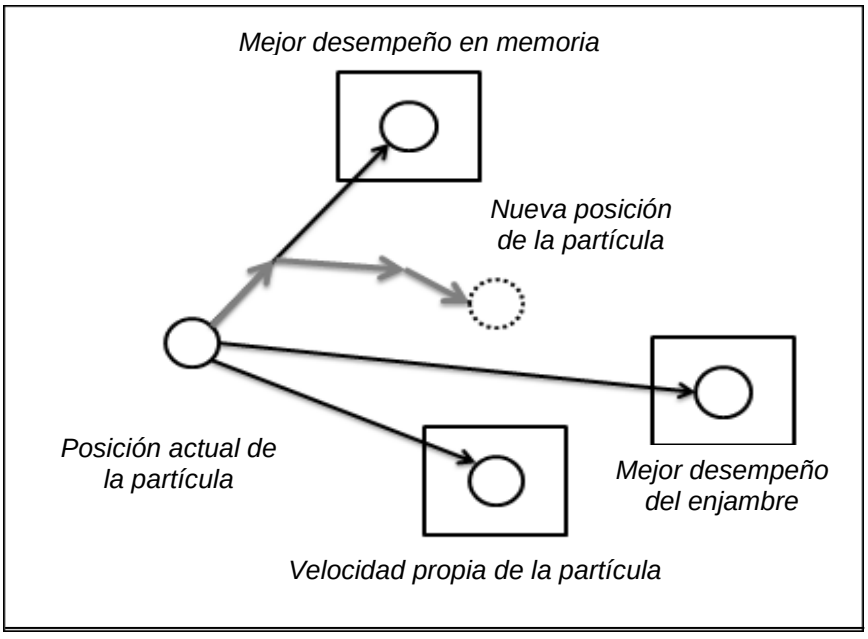
5.2.2 Optimización por Enjambre de Partículas. Según Parsopoulos (2010) define la técnica de optimización por enjambre de partículas como:

> Es un algoritmo de optimización estocástica basado en los modelos de simulación social, que fue propuesto por Kennedy y Eberhart (1995). El algoritmo emplea una población de puntos que se mueven estocásticamente en el espacio de búsqueda, concurrentemente la mejor posición es guardada por cada individuo, ésta característica se conoce como la experiencia y le permite a cada partícula sesgar al resto de la población hacia las regiones más prometedoras detectadas hasta ese momento, es por esto que el esquema de comunicación está determinado por una red social fijo o adaptativa que juega un papel crucial en la convergencia del algoritmo.
>
> El desarrollo de la optimización por enjambre de partículas está basado en el concepto y la reglas que gobiernan socialmente una población en la naturaleza, como cardúmenes de pescados, bandadas de aves y manada de animales, que a diferencia del enfoque de las colonias de hormigas, donde la estigmergia es el principal mecanismo de comunicación a través

del entorno, dichos sistemas de comunicación son más directos y no alteran el entorno.

Con base a lo anterior, se puede afirmar que el movimiento de una partícula está definido bajo tres aspectos básicos, utilizando la experiencia o recorrido a través de la posición con mejor desempeño en memoria de cada partícula, el segundo aspecto es a través del mejor desempeño obtenido por el mejor individuo del enjambre y por último, el recorrido siguiendo su propia velocidad (Fig. 3), a partir de esto emerge un comportamiento interesante llamado inteligencia de enjambre que permite la explotación (mediante la memoria y conociendo la mejor partícula del enjambre) y la exploración (a través de su propia velocidad) del espacio de búsqueda del problema.

Figura 3. Movimiento de una partícula.



Fuente: Particle Swarm Optimization – Maurice Clerc

La optimización por enjambre de partículas es una metaheurísticas aplicada a problemas continuos, por lo cual un enjambre S, se representa como el conjunto de partículas X_i:

$$S = \{X_1, X_2, X_3, \dots, X_n\} \quad \textbf{(1)}$$

Cada una de las partículas corresponde a un vector solución que indica la posición de ésta en el espacio de búsqueda A, por cual cada X_{ij} es una variable que toma valores continuos en la *N-sima* dimensión:

$$X_i = (X_{i1}, X_{i2}, X_{i3}, \dots, X_{in})^T \in A, i = 1,2, \dots, N \quad \textbf{(2)}$$

En esta representación cada índice es asignado arbitrariamente a cada partícula, mientras que N es un parámetro definido en el algoritmo, la función objetiva $f(x)$ debe estar definida en todos los puntos del espacio de búsqueda A, en consecuencia cada partícula posee un único valor en la función $f_i = f(X_i)$, también se asume que cada una de éstas se mueve iterativamente, es decir ésta se actualiza con base a la velocidad obtenida en el paso anterior del algoritmo, por lo que la velocidad propia se denota como:

$$v_i = (v_{i1}, v_{i2}, v_{i3}, \dots, v_{in})^T, \qquad i = 1,2, \dots, N \quad \textbf{(3)}$$

Para calcular la velocidad y la nueva posición de la partícula, Eberhart y Kennedy (1995) proponen la siguiente fórmula:

$$v_{i,j+1} = v_{ij}c_1 + c_{max}R_1\left(P_{ij} - X_{ij}\right) + c_{max}R_2\left(p_g - X_{ij}\right) \quad \textbf{(4)}$$

$$X_{i,j+1} = X_{ij} + v_{i,j+1} \quad \textbf{(5)}$$

Donde R_1 y R_2 representan una numero aleatorio distribuido uniformemente entre [0,1], C_1 y C_{max}, son los factores de peso, conocidos también como factor cognitivo y social respectivamente, además P_{ij} es la mejor posición de una partícula en memoria y P_{gj} es la mejor partícula en el enjambre, para explicar esto se define al conjunto P, que posee la posición en memoria de la mejor ubicación de cada partícula del enjambre:

$$P = \{P_1, P_2, P_3, \dots, P_n\} \quad \textbf{(6)}$$

Cada P_i corresponde al conjunto de posiciones en memoria de cada partícula P_{ij} que define la nueva posición si la posición actual posee un mejor desempeño que la mejor posición en memoria:

$$P_i = (P_{i1}, P_{i2}, P_{i3}, \dots, P_{in})^T \in A, i = 1,2, \dots, N \quad \textbf{(7)}$$

$$P_i = \begin{cases} X_i(t+1), si\ f\big(X_i(t+1)\big) \leq f\big(P_i(t)\big) \\ P_i(t), de\ otro\ modo \end{cases} \quad \textbf{(8)}$$

La partícula con mejor desempeño en el enjambre P_{gj}, es el mejor valor obtenido del vector P, ésta se define como:

$$P_g = f\big(P_i(t)\big) \quad \textbf{(9)}$$

El pseudo-código para el algoritmo de optimización por enjambre de partículas se presenta en la tabla 1.

Tabla 1. Algoritmo de optimización por enjambre de partículas

inicio
$S \leftarrow Inicializar(\,)$
$N \leftarrow tamaño(S)$
para $i = 1$ **hasta** N **hacer**
. **si** $(desempeño(X_i) > P_i\,)$ **entonces**
. . $P_i \leftarrow X_i$
. **fin si**
. **si** $(P_i > P_g\,)$ **entonces**
. . $P_g \leftarrow P_i$
. **fin si**
fin para
mientras que ($Condición\ de\ parada$) **hacer**
. **para** $i = 1$ **hasta** N **hacer**
. . $Escoger\ P_i,\ el\ mejor\ desempeño\ en\ memoria$
. . $v_i \leftarrow v_i c_1 + c_{max} R_1 (P_i - X_i) + c_{max} R_2 (p_g - X_i)$
. . $X_i \leftarrow X_i + v_i$
. **fin para**
. **si** $(desempeño(X_i) > P_i\,)$ **entonces**
. . $P_i \leftarrow X_i$
. **fin si**
. **si** $(P_i > P_g\,)$ **entonces**
. . $P_g \leftarrow P_i$
. **fin si**
fin mientras
imprime $P_g, desempeño(P_g)$
fin

Fuente: Kennedy y Eberhart, 1995

5.2.3 Búsqueda local. Es un método metaheurístico para resolver problemas de computación *HARD,* éste puede ser usado en problemas que pueden ser formulados para encontrar una solución, maximizando así el desempeño entre las soluciones candidatas. Básicamente la búsqueda local trata de moverse a través del espacio solución, aplicando cambios locales hasta encontrar un óptimo global o las iteraciones acaben. Entre las técnicas que utilizan búsqueda local está el *Hill Climbing*, Recocido Simulado, Búsqueda Tabú, Optimización por búsqueda reactiva.

5.2.4 Planeación de los recursos de manufactura. Inicialmente se propuso un sistema que permitiera planear y programar los requerimientos de los materiales en el tiempo para las operaciones de producción que aparecen en el MPS (plan maestro de producción), a esto se le conoció como MRP (Planeación de los requerimientos de los materiales), que contempla al BOM o lista de materiales dentro de uno los aspectos importantes aunque muchos otros no lo eran, debido a eso surgió la planeación de los recursos de manufactura o MRP II, que es un sistema integrado que va más allá del MRP de primera generación.

En términos generales el MRP II coordina las ventas, compras, manufacturas, finanzas e ingeniería al adoptar un plan de producción focal y utilizando una sola base de dato unificada, en detalles más específicos en el proceso de manufactura, éste contempla la planificación de los recursos de capacidad o CRP, necesaria para determinar aspectos más detallados como en el tamaño de lote a producir y/o en la necesidad de planificar el horizonte de tiempo de la producción.

5.2.5 Problema de tamaño de lote multinivel. En la actualidad el problema de tamaño de lote multinivel es considerado uno de los más difíciles problemas de optimización conocidos en el área de las operaciones y gestión de la producción, éste surge en una empresa que utiliza el enfoque del MRP en el cual el sistema se basa en un plan maestro de producción que registra el valor de la demanda

dependiente en cada período de tiempo, esto se hace con respecto a la estructura de la lista de materiales, la evolución del inventario en cada período y bajo el supuesto de que se conocen los tiempos de entrega fijos para cada componente.

Para los modelos de un solo componente y un solo nivel, existen algoritmos determinísticos que permiten calcular los tamaños de lote óptimos, en cambio para los otros modelos, la complejidad es mayor por lo que se utilizan metaheurísticas capaces de obtener muy buenos resultados sin conocer si es un óptimo global.

El problema de tamaño de lote multinivel es un problema de programación entera mixta, el modelo matemático que se adopta lo propone Han (2009) y describe a partir de las siguientes notaciones:

Tabla 2. Notación de las variables del modelo matemático del problema de tamaño de lote multinivel.

$i \rightarrow El\ índice\ de\ cada\ item$
$H_i \rightarrow Costo\ de\ mantener\ el\ item\ i\ en\ inventario$
$K_i \rightarrow Costo\ de\ realizar\ una\ orden\ de\ compra\ del\ item\ i$
$l_i \rightarrow Tiempo\ de\ entrega\ de\ ensamble, producir\ o\ comprar\ el\ item\ i$
$I_{i,j} \rightarrow Nivel\ de\ inventario\ i\ al\ final\ del\ periodo\ t$
$a_{i,j} \rightarrow Matriz\ binaria\ que\ rgistra\ la\ orden\ de\ pedido\ del\ item\ i\ en\ el\ periodo\ j$
$D_{i,j} \rightarrow Demanda\ del\ item\ i\ en\ el\ periodo\ t$
$P_{i,j} \rightarrow Nivel\ de\ reaprovisionamiento\ del\ item\ i\ en\ el\ periodo\ t$
$M \rightarrow Numero\ muy\ grande$
$X_{it} \rightarrow Numero\ de\ unidades\ del\ item\ i\ producidas\ en\ el\ tiempo\ t$
$St_{im} \rightarrow Tiempo\ de\ alistamiento\ requerido\ para\ producir\ el\ item\ i\ en\ la\ maquina\ m$
$CP_{mt} \rightarrow Capacidad\ disponible\ de\ la\ máquina\ m\ en\ el\ periodo\ t$

Fuente: Han, 2009. P. 1750

Donde la función objetiva es:

$$Min \sum_{i=1}^{N} \sum_{t=1}^{T} \left(H_i \times I_{i,t} + K_i \times a_{i,t} \right) \quad \textbf{(10)}$$

Sujeto a (cada restricción debe ser para toda *i, t*):

$$I_{i,t} = I_{i,t+1} + P_{i,t} - D_{i,t} \quad \textbf{(11)}$$

$$D_{i,t} = \begin{cases} D_{i,t} & si\ \delta(i) = \emptyset \\ \sum_{j \in \delta(i)} C_{i,j} \times P_{j,t+l_j} & de\ otro\ modo \end{cases} \quad \textbf{(12)}$$

$$P_{i,t} = a_{i,t} D_{i,t} + \sum_{m=t+1}^{T} \left(a_{i,m} + D_{i,m} \prod_{q=i+1}^{m} (1 - a_{i,q}) \right) \quad \textbf{(13)}$$

$$\sum_{i=1}^{I} l_i * X_{it} + \sum_{i=1}^{I} st_{im} * a_i \leq CP_{mt} \quad \textbf{(14)}$$

$$P_{i,t} - M a_{i,t} \leq 0 \qquad a_{i,t} \in \{0,1\} \quad \textbf{(15)}$$

$$P_{i,t} \geq 0, I_{i,t} \geq 0 \quad \textbf{(16)}$$

La función objetiva (Fórmula 10) trata de minimizar los costos de ordenar un producto y el costo de almacenamiento del producto al final de un periodo; la primera restricción (Fórmula 11) trata de balancear el costo entre la producción y/o

reaprovisionamiento del inventario y la demanda; la segunda restricción (Fórmula 12) provee el método para el cálculo de la demanda; El reaprovisionamiento depende de la decisión de establecer una orden de compra para los siguientes periodos, eso lo determina la tercera restricción (Fórmula 13), la capacidad del sistema está determinada por la cuarta restricción (Fórmula 14) por último la quinta (Fórmula 15) y sexta restricción (Fórmula 16), garantiza que de ordenar está determinado por si se va a producir o no y que el reaprovisionamiento y el inventario no pueden ser menor de 0, respectivamente.

5.2.5.1 Características del problema. Las características más notables del modelo son:

- ✓ **Cantidad de productos.** Es la característica del sistema de MRP que se refiere a cuantos productos ha de fabricar la empresa y se especifican en el esquema de producción, la dos categorías son:
  - ✓ Simple producto
  - ✓ Múltiple producto

- ✓ **Tipo de demanda.** Corresponde a la petición de compras en un horizonte de tiempo del producto(s) a fabricar, se conoce:
  - Demanda constante, que se mantiene igual en todos los periodos del horizonte de tiempo.
  - ✓ Demanda dinámica la cual es variable a lo largo del período *t*.

- ✓ **Diferentes tipos de recursos.** Se refiere a las horas–hombre para realizar un trabajo, insumos para las maquinas, tiempo efectivo de producción o cualquier otro recurso que permita la fabricación del producto.

- ✓ **Diferentes costos de ordenamiento por cada componente.** El costo de realizar un pedido para cada componente en el sistema puede ser constante o variable a lo largo del horizonte de planificación.

- ✓ **Diferentes tiempos de producción o entrega por cada componente.** Se especifica el tiempo de producción o entrega como constante o variable según el componente, incluso se puede suponer que el tiempo de producción o entrega es $\sum l_i = 0$

- ✓ **Cualquier número de productos puede ser fabricado en un cualquier período.** Corresponde a afirmar que el tamaño de lote no se someterá a un sistema justo a tiempo, es decir que la producción del periodo $D_{i,t}$ no será arrastrada por la demanda del periodo $D_{i,t+1}$, por ende el tamaño de lote $P_{i,t}$ puede ser variable para permitir obtener diferentes respuestas que pueden ser óptimas o no.

- ✓ **Una o múltiples maquinas.** La fabricación de un componente se puede realizar en una máquina o en múltiples maquinas de manera concurrente tal que afectaría el tamaño de lote $P_{i,t}$ en el periodo t.

- ✓ **Costo de escasez asociado al no satisfacer las necesidades de la demanda.** Esta característica afecta la función objetiva debido a que se penaliza sino se satisface la demanda en un período t, se puede basar en el supuesto de que no hay costo de escasez.

5.2.5.2 Variantes del problema. Actualmente existen diferentes clasificaciones del problema del tamaño de lote en un sistema de MRP, esto es debido a las diferentes características del modelo que pueden afectar al problema

como tal, que incluso éstas pueden ser tomadas como supuestos al momento de la resolución para reducir la complejidad del problema, en consecuencia los modelos más empleados son:

- ✓ **Modelos sin capacidad restringida.**
  - ❖ Cantidad económica de pedido con un conjunto costos de reaprovisionamiento o *"Economic Order Quantity with Joint Replenishment Cost"* (EOQJR)
  - ❖ Cantidad económica de pedido con un conjunto de reaprovisionamiento en estructuras multinivel o *"Economic Order Quantity in Multi Level Product Structure"* (EOQML)
  - ❖ Problema del tamaño de lote dinámico, multi producto con un conjunto de costos de ordenamiento o *"Multi-item dynamic lot.sizing problem with joint set-up cost"* (LPJS)

- ✓ **Modelos capacitados con demanda constante y/o estacionaria.**
  - ❖ Problema de planificación del tamaño de lote o *"Economic lot-Sizing scheduling problem"* (ELSP)

- ✓ **Modelos capacitados con demanda dinámica.**
  - ❖ Problema de planificación del tamaño de lote discreto o *"Dicrete Lot-Sizing Scheduling Problem*" (DLSP).
  - ❖ Problema de planificación del tamaño de lote capacitado o *"Continous Lot-Sizing Scheduling Problem"* (CLSP).
  - ❖ Problema de planificación del tamaño de lote capacitado o *"General Lot-Sizing Scheduling Problem*" (GLSP).

5.2.5.3 Complejidad del problema. Una característica de los problemas de tamaño de lote multinivel es que poseen un amplio espacio de búsqueda es por ello que para encontrar una solución óptima el tiempo de solución aumenta exponencialmente, ya sea por el número de variables o por las restricciones, más aún, se hace mucho más complicado con la introducción de producción multinivel, debido a esto, éste problema es categorizado como NP-Hard, dado que la respuesta no se puede conocer de forma determinística y el cálculo para obtener la solución óptima se lleva a cabo en un tiempo polinomial.

5.2.6 Ingeniería de software. Una de las primeras definiciones de la ingeniería de software fue dada en 1969 por Fritz Bauer, él define la ingeniera de software como "*El establecimiento y uso de principios de ingeniera robustos, orientados a obtener software económico que sea fiable y funcione de manera eficiente sobre maquinas reales.*" Se puede afirmar que la ingeniería de software es el conjunto de métodos, herramientas y procedimientos orientados a la construcción de un software eficiente, que satisfaga las necesidades requeridas.

El objetivo principal de la ingeniería de software es construir una solución de software eficiente que satisfaga las necesidades requeridas por un cliente. Los objetivos específicos son:

- ✓ Proveer los estándares y los modelos que faciliten la comunicación, tanto para clientes como para especialistas en la elaboración de un proyecto de software.
- ✓ Proveer los métodos, herramientas y procedimientos para la "construcción" eficiente del software
- ✓ Proveer parámetros y criterios de evaluación de la calidad del software

Según Roberto Cortez Morales (1998) los parámetros deseables en un software son: Compatibilidad, corrección, eficiencia, flexibilidad, mantenibilidad, portabilidad, confiabilidad, reusabilidad, robustez, salvedad, seguridad, examinabilidad, compresión, uso-amigable, validez y verificación.

5.2.7 Programación orientada a objetos. Según Weitzenfeld (2005), argumenta que existe una estructura de alto nivel llamada objeto, que ofrece dos ventajas sobre la programación tradicional.

La primera es permitir al programador que organice su programa de acuerdo con abstracciones de más alto nivel, siendo éstas más cercanas a la manera de pensar de la gente. En otras palabras, los objetos son las unidades de representación de las aplicaciones, por ejemplo, cuentas de banco, reservaciones de vuelos, etcétera. La segunda es que los datos globales desaparecen, siendo éstos junto con las funciones parte interna de los objetos. Por lo tanto, cualquier cambio en la estructura de alguno de los datos solo debiera afectar las funciones definidas en ese mismo objeto y no en los demás.

Para garantizar un buen análisis y diseño de un software se deben tener en cuenta los siguientes aspectos y conceptos básicos que permitirán mejorar la calidad de este, tales como:

- ✓ **Abstracción.** Este consiste en elevar el nivel de las representaciones necesarias para un sistema de software, de manera que se reduzcan los detalles.

- ✓ **Modularidad**. Este permite dividir unos sistemas en componentes separados. Al contar con abstracciones de más alto nivel, la modularidad de un sistema se logra con base en componentes, también de más alto nivel.

Esto reduce el número final de componentes en un sistema y, a su vez, facilita su operación y mantenimientos.

- ✓ **Extensibilidad.** Esta corresponde a la facilidad en modificar un sistema durante el transcurso de la vida del software.

- ✓ **Reutilización.** Es el reúso de componentes es otro de los mecanismo importantes para administrar la complejidad del software. La reutilización reduce el tiempo de diseño, codificación y costo del sistema al amortizar el esfuerzo sobre varios desarrollos. Mediante la reutilización se aprovechan componentes o bibliotecas ya desarrolladas, logrando una mayor estandarización y simplificación en las aplicaciones.

- ✓ **Encapsulamiento.** Es el mecanismo básico de la programación orientada a objetos para ocultar los detalles internos del objeto de los demás objetos.

- ✓ **Clasificación**. Permite organizar objetos de acuerdo con estructuras comunes. Los objetos con datos y funciones similares se clasifican como si fueran de la misma clase aunque puedan tener datos con valores distintos.

- ✓ **Generalización**. La generalización o especialización se puede definir como que los objetos pueden organizarse como miembros de una misma clase y también se pueden organizar las propias clases de los objetos de acuerdo con sus datos y funciones comunes. Mediante la generalización, las clases con estructura y comportamiento similar se pueden reutilizar en las definición de nuevas clases, siendo éstas más especializadas que las anteriores y se les conoce como subclases, mientras las más generales como superclases. El mecanismo para describir jerarquías se conoce como herencia, que dice que una subclase hereda de una superclase.

- ✓ **Polimorfismo.** Es el concepto más poderoso de la programación orientada a objetos. Mediante el polimorfismo se definen múltiples funciones con nombres e interfaces similares solo que en distintas clases. Las funciones son implementadas de manera diferente en las clases. El polimorfismo es útil para extender la funcionalidad del sistema al definir nuevas clases aun desconocidas al momento de especificarlo.

5.3 ANTECEDENTES

En la presente sección se realizará una revisión a través de las investigaciones hechas sobre el modelo básico del problema de Tamaño de Lote en el sistema de MRP hasta los modelos más complejos y los técnicas o métodos utilizados para resolver estos problemas de manera determinísticos o probabilística, en el cual el objetivo fundamental es establecer bases teóricas para el desarrollo del proyecto.

Inicialmente el concepto de MRP fue propuesto por Joseph Orlicky en la década de los 60‘s trabajando él en el programa de producción de Toyota que posteriormente escribió en su libro, (ORLICKY, 1975) en el cual fundamentó este concepto bajo la máxima de establecer el producto adecuado, en la cantidad adecuada y en el momento adecuado, para ello surgió la necesidad de calcular el tamaño de lote que permitiera reducir los niveles de inventarios de las empresas, las características más notables de este modelo eran la aplicación de un horizonte de planificación, una dependencia temporal que tienen los parámetros del modelo, el número de productos, las etapas de producción y la estructura productiva, ésta última permitía categorizar el problema en nivel simple o multinivel.

Posteriormente en la época de los 80’s el concepto de MRP había comenzado a ser implementado por más de 8000 empresas y convirtiéndose así, en el paradigma de producción de Estados Unidos permitiéndole a las empresas obtener notables resultados en los esquemas de producción, luego en 1983 Oliver

Wigth evoluciona éste concepto de MRP y propone la planeación de los requerimientos de manufactura o MRP II, el cual encierra los conceptos de MRP, CRP (Planeación de los Requerimientos de Capacidad) entre otros, éste último plantea las especificaciones en cuanto a las restricciones de los recursos lo que permite conocer si la capacidad es restringida. Si la capacidad no esta restringida se dice que el problema es con capacidad infinita, de lo contrario se dice que es finita. Con base a esto se encuentran en la literatura diferentes autores que han trabajado con modelos sin capacidad restringida en el cual se supone que los recursos utilizados tienen capacidad infinita; En 1992, Federgruen y Zheng, proponen un modelo para calcular la cantidad económica a pedir con un conjunto de costos de reaprovisionamiento (EOQJR), por otro lado Afentakis y Gavish (1986) propone un modelo de tamaño de lote para productos con estructuras más complejas en el cual la demanda es dinámica.

En los modelos capacitados con demanda constante y/o estacionaria, Pesenti y Ukovich (2003) presentaron la formulación de un problema de planificación de tamaño de lote con la condición de que se presentan restricciones con inconsistencia y el desarrollo de una heurística para su tratamiento, por otra parte los modelos capacitados con demanda dinámica son una extensión del modelo de Wagner y Whitin (1958), considerado para múltiple ítems y restricciones de capacidad en el cual se encuentran diferentes categorías, tal como el modelo discreto de planificación del tamaño de lote (DLSP) el cual consiste en determinar la secuencia y el tamaño de los lotes de producción para múltiples ítems en una sola maquina. (Fleischmann, 1994; Salomón, 1991), la otra categoría corresponde al modelo continuo de tamaño de lote (CSLP) en el cual el lote no es arrastrado por la demanda sino que puede ser menor o incluso 0, (Toniolo, et. Al, 2001), una versión más compleja de este caso fue estudiado por (Almada-Lobo et. Al., 2007). Para encerrar estas categorías, se puede hablar del modelo general de planificación del tamaño de lote (GLSP), en el cual Drexl y Kimms (1997) lo

proponen con un simple nivel y un sistema con múltiple producto trabajaron, igualmente Fleischmann y Meyr (1997) aplican el mismo modelo pero con otras características como múltiple producto, una maquina, secuencia dependiente, costos de ordenamiento, periodos de tiempo cortos y es muy parecido al DLSP.

Luego de revisar los diferentes modelos que los autores han planteado sobre el tema, es importante observar las técnicas y métodos empleados para la solución del problema. (Ver Tabla 3)

Tabla 3. Técnicas y métodos empleados para la resolución del problema de tamaño de lote.

Técnica o método	Problema de planificación de tamaño de lote
Programación matemática	Belvaux y Wolsey, 2000; Clark, 1998; Drexl y Haase, 1995; Drexl y Kimms, 1997; Kang, et al.1999; Wolsey 1997; Kuik et al. 1993.
Recocido simulado	Crauwels et. Al. 1996; Kuik et al. 1993.; Barbarosoglu y Özdamar, 2000; Kim 1996; Tang, 2004; Gaafar et al, 2009; Sarker y Yao, 2003
Algoritmos evolutivos	Hyun et al, 1998; Khouja et al, 1998; Disney et al, 2000; Ip et al, 2000; Kim 1996; Kimms 1999; Dellart et al, 2000; Xie y Dong, 2002.
Búsqueda Tabú	Kuik et al. 1993.; Gopalakrishman et al, 2001; Hindi, 1996; Pereira et al, 2003;
Enjambre de partículas	Han et al, 2009; Wajanawichakon y Pitakaso, 2011; Gaafar y Aly, 2009; Han et al, 2010; Hsu, 2011

Fuente: Elaboración propia

6. DIAGNOSTICO

Según la metodología establecida para el desarrollo del presente trabajo, en la primera fase de investigación se hace necesario consultar en la literatura especializada para obtener un punto de referencia que permita contextualizar el modelo del problema, como también identificar las variables criticas que los autores proponen en sus estudios, posteriormente identificar los autores que han trabajado con el algoritmo de optimización por enjambre de partículas o si ninguno lo ha hecho, determinar que estrategias usar para adaptar el algoritmo PSO a una naturaleza combinatoria, por último, definir el caso de estudio o caso base que permitirá evaluar el desempeño del algoritmo frente a las otras metaheurísticas.

6.1 ANÁLISIS DE LAS VARIABLES

Mediante la revisión de los documentos se ha encontrado que los autores definen las variables críticas y no criticas que afectan al marco del problema, con el fin de formular el modelo matemático, estas variables críticas como tal, corresponden a los factores que afectan en la naturaleza del problema de tamaño de lote como número de componentes, tiempos de entrega, etc. En cambio las variables no críticas se pueden tomar como supuestos liberando así la complejidad del problema; con base a eso, podemos definir que las variables críticas y no críticas de este trabajo son:

Variables críticas:

- ✓ Un solo producto final que se demanda en cada período.
- ✓ Una cantidad determinada de productos y componentes en el sistema.
- ✓ Tipo de demanda constante en un número de períodos.
- ✓ Un único recurso que comparten los componentes que se fabrican.

- ✓ Costos de ordenamiento o producción, costos de almacenamiento y tiempos de ciclo fijos para cada producto o componente del modelo.
- ✓ Inventario inicial de los períodos.
- ✓ Una estructura en ensamble del producto final.
- ✓ Una sola máquina o cadena de proceso del producto.

Variables no críticas.

- ✓ Tiempos de entrega despreciable
- ✓ Múltiples maquinas para fabricar un producto.
- ✓ No hay estructuras generales en el modelo.
- ✓ No hay costos de escasez
- ✓ No hay backlogging

6.2 ESTRATEGIAS PARA EL DISEÑO DEL ALGORITMO PSO

Inicialmente, se presenta al problema de tamaño de lote multinivel capacitado como un problema de complejidad *NP-HARD*, por lo cual la utilización de un técnica metaheurística es indispensable para obtener buenos resultados en un menor tiempo, además se busca evaluar el desempeño del algoritmo de optimización por enjambre de partículas debido al poco trabajo desarrollado en este campo.

En consecuencia, se ha encontrado en la literatura que autores como Han. et al. (2009) y Wajanawichakon con Pitakaso (2011), han propuesto el uso de este algoritmo para solucionar el problema, por lo cual estas dos propuestas serán evaluadas para determinar cual provee una mejor estrategia con base a la facilidad del modelo, en cuanto a la forma de cómo explorar y explotar la superficie

de las soluciones o espacio de búsqueda y a la abstracción de la representación de una solución.

6.2.1 Estrategia 1 – propuesta de Han, et al. En este artículo los autores proponen la implementación del algoritmo de optimización por enjambre de partículas con pesos inerciales flexibles para la solución del problema de tamaño de lote multinivel no capacitado en estructuras de ensamble, con base a esto la adaptación del algoritmo se basó principalmente en la representación abstracta de una partícula o solución y en los símbolos o operadores de la ecuación de velocidad.

La matriz binaria de los ordenamientos o set ups, representa una partícula y se expresa como:

$$A = \begin{bmatrix} a_{1,1} & a_{1,2} & . & . & a_{1,T} \\ a_{2,1} & a_{2,2} & . & . & a_{2,T} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ a_{N,1} & a_{N,1} & . & . & a_{N,T} \end{bmatrix} \quad \textbf{(17)}$$

Donde $a_{i,T} \in \{0,1\}$ denota si se hace o no un ordenamiento del ítem i en el período *t,* adicionalmente para el diseño del algoritmo PSO se redefinen los operadores de datos continuos a operadores matriciales, de tal forma que se puedan evaluar las partículas en la ecuación de velocidad original (4) y (5) a la ecuación (21) y (22).

$$v_{i,j+1} = v_{ij} \otimes c_1 \oplus (c_{max} * R_1) \otimes \left(P_{ij} \boxminus X_{ij}\right) \oplus (c_{max} * R_2) \otimes * \left(p_g \boxminus X_{ij}\right) \quad \textbf{(18)}$$

$$X_{i,j+1} = X_{ij} \boxplus v_{i,j+1} \quad \textbf{(19)}$$

Donde los operadores matriciales son:

- ✓ **Velocidad (V).** La velocidad de un algoritmo PSO se define como una colección de pares numéricos; la longitud de una velocidad V, denotada por $\|V\|$ es igual al número de pares numéricos. Por ejemplo, una velocidad *V* se da como:

  $$V = \big((i_k, j_k)\big), i_k \in \{1 \dots N\}, j_k \in \{1 \dots T\}, k \uparrow_1^{\|V\|}$$

  El número de pares (i_k, j_k) es igual a $\|V\|$. *N* es el número de elementos y *T* es el número de períodos de planificación.

- ✓ **La velocidad más la velocidad ($\oplus$).** La suma de dos velocidades se considera simplemente como una operación combinada de dos velocidades. Con el fin de acortar la longitud de la velocidad resultante, los pares numéricos repetitivos serán eliminados. Un ejemplo es:

  $$V_1 = \big((1,4),(2,3),(5,7)\big) \text{ y } V_2 = \big((1,3),(2,3)\big),$$

  Entonces:

  $$V = V_1 \oplus V_2 = \big((1,3),(1,4),(2,3),(5,7)\big).$$

- ✓ **La velocidad menos la velocidad ($\ominus$).** La sustracción de dos velocidades se considera simplemente como una operación de eliminación y recombinación en dos velocidades. En primer lugar, los pares numéricos

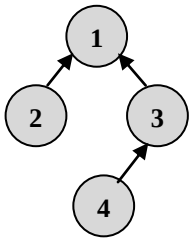
iguales son eliminados de ambas velocidades. Entonces, las dos velocidades se combinan juntas. Un ejemplo es:

$$V_1 = \big((1,4),(2,3),(5,7)\big) \text{ y } V_2 = \big((1,3),(2,3)\big),$$

Entonces,

$$V = V_1 \ominus V_2 = \big((1,3),(1,4),(5,7)\big).$$

- ✓ **Posición más velocidad.** La suma de una posición y la velocidad es una mutación continua en los bits de una posición de acuerdo a las correspondientes pares numérica de la velocidad.



$$\mathrm{A} = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix} \qquad V = \big((1,4),(2,2)\big)$$

$$\mathrm{A} = \begin{pmatrix} 1 & 1 & 1 & \boxed{1} \\ 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix} \Rightarrow \mathrm{A}' = \begin{pmatrix} 1 & 1 & 1 & \boxed{0} \\ 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix} \qquad \mathrm{A} = \begin{pmatrix} 1 & 1 & 1 & 0 \\ 1 & \boxed{1} & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix} \Rightarrow \mathrm{A}' = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & \boxed{0} & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix}$$

Aplicando (1, 4) en A 2) Aplicando (2,2) en A

La matriz *A* se transforma en primer lugar *A'* mediante la mutación de '1' en la posición (1, 4). Entonces, una mutación acumulativo operación en la posición (2, 4) se realiza para satisfacer la restricción de viabilidad mediante el cambio '1' en '1'. Después de eso, *A'* es cambiado más en *A"* sustituyendo el valor '1' en (2, 2) con '0'.

- ✓ **Posición menos posición.** El resultado de una resta entre una posición y la posición de otra es una velocidad. La velocidad resultante se obtiene a través de registrar los diferentes puntos de las dos posiciones.

$$\mathrm{A}' = \begin{pmatrix} 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix} \quad \mathrm{A}'' = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix} \qquad \mathrm{V} = \mathrm{A}' \boxminus \mathrm{A}'' = \begin{pmatrix} 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

$$\mathrm{V} = \mathrm{A}' \boxminus \mathrm{A}'' = \big((1,3),(1,4),(2,2),(2,3)\big)$$

- ✓ **Coeficiente de múltiples velocidades.** Sea *c* un coeficiente real y *V* una velocidad. El resultado de la multiplicación entre c y *V* se obtiene como una velocidad dependiendo basa en los siguientes dos casos de *c*.

  - ❖ Si $|c| < 1$, denotan $\|cV\|$ser el mayor número entero menor que o igual a $c\|V\|$, entonces podemos obtener $c \otimes V = \big((i_k, j_k)\big), k \uparrow_1^{\|cV\|}$ Por ejemplo, vamos a $c = 0.6$ y $V = \big((2,3),(2,5),(3,7)\big)$, entonces $\|V\| = 3$, $c\|V\| = 1.8,\ \|cV\| = 1, c \otimes V = (2,3)$.

  - ❖ Si $|c| \geq 1$, entonces *V* y la longitud de *V* se mantiene sin cambios. Por lo tanto, la aparición de repetitivas pares numéricos se evita y la multiplicación de c y *V* se simplifica.

6.2.2 Estrategia 2 – propuesta por Wajanawichakon y Pitakaso (2011). La estrategia utilizada por los autores en su artículo se basó principalmente en representar una partícula o solución de igual forma que en la estrategia 1, con la diferencia que ésta, agrega una matriz adicional de las velocidades ($V_{i,j}$) a cada partícula, esto con el fin utilizar la ecuación (4) y (5) tal cual como está expresada sobre cada posición del producto i en el periodo j de la matriz $V_{i,j}$.

- ✓ **Generación de la población inicial.**

En consecuencia el algoritmo que ellos proponen consta de generar una población aleatoria de partículas binarias e igualmente generar aleatoriamente con una probabilidad uniforme los valores de la matriz A de set ups, de la siguiente manera:

$$if\ \cup(0,1) > 0{,}5\ entonces\ A_{i,j} = 1$$
$$sino\ A_{i,j} = 0$$

La valor de la velocidad de la partícula *i* en la dimensión *d* se establece como:

$$V_{i,d} = V_{min} + (V_{max} - V_{min}) * rand() \quad \textbf{(20)}$$

Donde:

$$V_{min} = -V_{max} \quad \textbf{(21)}$$

- ✓ **Búsqueda de la población nueva.**

Tenemos que usar dos funciones para la generación de nuevas soluciones, a saber, una función lineal a nivel de pieza para obligar a los valores de velocidad

entre los valores permisibles máximos y mínimos, el intervalo$[v_{min}, v_{max}]$ se utiliza para controlar el valor a niveles mínimos y máximos.

$$h\left(v_{id}^{k}\right) = \begin{cases} v_{max}, & \text{if } v_{id}^{k} > v_{max} \\ v_{id}^{k}, & \text{if } \left|v_{id}^{k}\right| \leq v_{max} \\ v_{min}, & \text{if } \left|v_{id}^{k}\right| < v_{max} \end{cases}$$

Una función sigmoide se utiliza para forzar los valores entre 0 o 1 después de la aplicación de la función lineal a nivel de pieza, la siguiente función sigmoidea se utiliza para escalar las velocidades entre 0 y 1, que luego se utiliza para convertir en valores binarios.

Esto es:

$$\text{Sigmoid}\left(v_{id}^{k}\right) = \frac{1}{1 + \exp(-v_{id}^{k})} \quad \textbf{(22)}$$

Nuevas partículas se encuentran mediante la actualización de la velocidad y la dimensión respectivamente. Calculamos la velocidad de actualización de v_{id}^{k} con la siguiente ecuación:

$$\Delta(v_{id}^{k-1}) = c_1 r_1\left(pb_{id}^{k-1} - x_{id}^{k-1}\right) + c_2 r_2(gb_{d}^{k-1} - x_{id}^{k-1}) \quad \textbf{(23)}$$

Luego actualizamos la velocidad v_{id} mediante el uso de la función lineal *h*

$$v_{id}^{k} = h\left(v_{id}^{k-1} + \Delta v_{id}^{k-1}\right) \quad \textbf{(24)}$$

Finalmente, se actualiza dimensión d de la partícula *i* tal que

$$x_{id}^{k} = \begin{cases} 1, \text{si } \bigcup(0,1) < \text{Sigmoid } (v_{id}^{k}) \\ 0, \text{de otro modo} \end{cases}$$

6.2.3 Análisis de las estrategias. Luego de haber establecido las 2 estrategias que servirán para la implementación del algoritmo PSO en este trabajo, se presenta en la tabla 4 la matriz de decisión con base a los 3 factores definidos previamente, cada uno con su respectiva ponderación y la valoración de los pesos en un rango del 1 al 10 emitido por un juicio propio.

Tabla 4. Matriz de decisión

Matriz de decisión					
Factor	**Peso**	**Valoración Estrategia 1**		**Valoración Estrategia 2**	
Facilidad del modelo	25%	7	1,75	9	2,25
Exploración y explotación del espacio de búsqueda	40%	9	3,6	8	3,2
Abstracción de la solución	35%	8	2,8	10	3,5
TOTAL			**8,15**		**8,95**

Fuente: Elaboración propia

En consecuencia con los resultados, la estrategia 2 posee mayor valoración (8,95) que la estrategia 1 (8,15) y esto es porque se tiene en cuenta que la propuesta de Wajanawichakon y Pitakaso posee más facilidad para la implementación de las matriz de velocidad que en comparación con la propuesta de Han, et al. con los

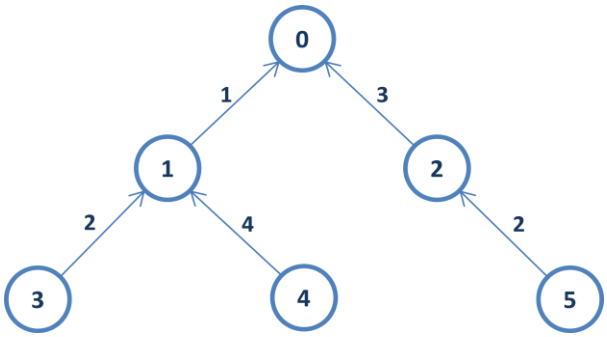
operadores matriciales, además la abstracción de la solución es mucho más completa debido a que contempla una velocidad para cada set up y no una velocidad global como en la estrategia 1 y por último, se puede afirmar que el método de los operadores matriciales permite explorar y explotar el espacio de búsqueda de una mejor manera, sin embargo con la inclusión de una búsqueda local puede mejorar este aspecto y a su vez convertir la estrategia 2 en la estrategia a seguir por este trabajo.

6.3 IDENTIFICACIÓN DEL CASO DE ESTUDIO

Para evaluar el desempeño del algoritmo presentado en este trabajo, se propone el caso de estudio desarrollado por Han (2009) que se tomará como el artículo guía, el cual consta de planificar el tamaño de lote de un producto primario en un lapso de tiempo de 12 períodos, además el producto primario posee una jerarquía multinivel, no posee inventario inicial y el tiempo de entrega es despreciable, en la tabla 5 se presenta la información del caso de estudio.

La explosión de los materiales del producto principal se presentan en la figura 4; el objetivo principal del problema es calcular los tamaños de lote de los productos tal que minimicen el costo de inventario y los costos de set up.

Figura 4. Explosión de materiales del producto principal del caso de estudio.



Fuente: Han, et al. 2009

Tabla 5. Información del caso de estudio

Información del caso de estudio												
Numero del producto	**Relación entre los productos**						**Costo de almacenamiento**			**Costo de set up**		
	Precedencia			**Requerimientos**								
0	**Producto final**			1			1			130		
1	0			1			2			120		
2	0			3			1			25		
3	1			2			2			30		
4	1			4			3			30		
5	2			2			1			40		
T	1	2	3	4	5	6	7	8	9	10	11	12
$D_{i,t}$	32	41	148	36	120	28	32	12	30	10	32	41

Fuente: Han, et al. 2009

Según el artículo guía se fijaron en los parámetros iniciales los valores de 60 partículas, $V_{max} = 0.1 \times N \times T$ y un total de 500 iteraciones, luego de 100 corridas se presentan en la tabla 6 la comparación de los resultados obtenidos por los autores luego de probar la adaptación del algoritmo de optimización por enjambre de partículas con la estrategia de los operadores matriciales, frente a otras técnicas metaheurística como algoritmo genético y Wargner-Within.

Tabla 6. Resultados del caso de estudio.

Resultados del caso de estudio				
Algoritmo	**Mejor**	**Peor**	**Promedio**	**Promedio tiempo de ejecución**
PSO	1895	5565	2143,9	5,3
AG	1895	6446	2526,9	8
WW	2123	2123	2123	<0,1

Fuente: Han, et al. 2009

7. DISEÑO DEL ALGORITMO PSO

Conforme a la metodología propuesta en este trabajo, la siguiente fase corresponde a analizar la estrategia que se adoptó para modificar el algoritmo de optimización por enjambre de partículas a una naturaleza combinatoria, desarrollar el diagrama de flujos del algoritmo, explicar y documentar los detalles de cada proceso y presentar la estrategia para aplicar la búsqueda local, adicionalmente se presentan los requerimientos funcionales que servirán para el diseño y desarrollo de una librería que implemente el algoritmo y provea un código legible y extensible.

7.1 DISEÑO DEL MODELO MATEMÁTICO

Con base al modelo matemático general del problema de tamaño de lote multinivel capacitado, se puede crear el modelo correspondiente a este problema mediante la definición de variables críticas y no críticas expuestas en el diagnostico del trabajo, por lo anterior se puede afirmar que la función objetiva es:

$$Min \sum_{i=1}^{N} \sum_{t=1}^{T} \left(H_i \times IF_{i,t} + K_i \times a_{i,t} \right) \quad \textbf{(25)}$$

Sujeto a (cada restricción debe ser para toda *i, t*):

$$IF_{i,t} = \begin{cases} I_{i,t+1} + P_{i,t} - D_{i,t} & Si\ i = 0 \\ I_{i,t+1} + P_{i,t} - P_{k,t} \cdot Req_k & para\ i > 0 \end{cases} \quad \textbf{(26)}$$

$$Donde\ k\ es\ el\ identificador\ de\ la\ raíz\ del\ producto\ i$$

$$I_{i,t} = IF_{i,t-1} \quad \textbf{(27)}$$

$$P_{i,t} = \begin{cases} \sum_{t=0}^{Prox(ai=1)} D_{i,t} \\ P_{k,t} \cdot Req_k \end{cases} \quad \textbf{(28)}$$

$Donde\ Prox(\ )\ es\ una\ función\ que\ retorna\ índice\ del\ proximo\ setup$

$$\sum_{t=0}^{T} \sum_{i=0}^{N} a_{it} \cdot tc_i \leq Cp_{it} \quad \textbf{(29)}$$

$Donde\ l\ es\ el\ número\ de\ producto\ y\ componentes\ del\ sistema$

$$P_{i,t} \geq 0, IF_{i,t} \geq 0, I_{i,t} \geq 0 \quad \textbf{(30)}$$

Del cual se puede afirmar que la función objetiva es invariante respecto al modelo general, pero las restricciones cambian para calcular el inventario inicial, final y el tamaño de lote, la restricción (26) determina que el inventario final del producto *i* en el período *t* es el inventario inicial más el tamaño de lote a producir menos la demanda en el período *t*, en cambio si es un componente la ecuación varia en cuanto a que la demanda será el tamaño de lote del producto de nivel superior; la restricción (27) corresponde a afirmar que el inventario inicial será igual al inventario final al periodo anterior; la restricción (28) afirma que la planificación del tamaño de lote del producto final en el período *t* se hará mediante la sumatoria de las demandas hasta encontrar un nuevo set up, en cambio para los componentes, se realizará a partir del tamaño de lote del producto de nivel superior por los requerimientos del producto; la restricción (29) afirma que el tiempo de ciclo de cada producto por la planificación del tamaño de lote no puede exceder la

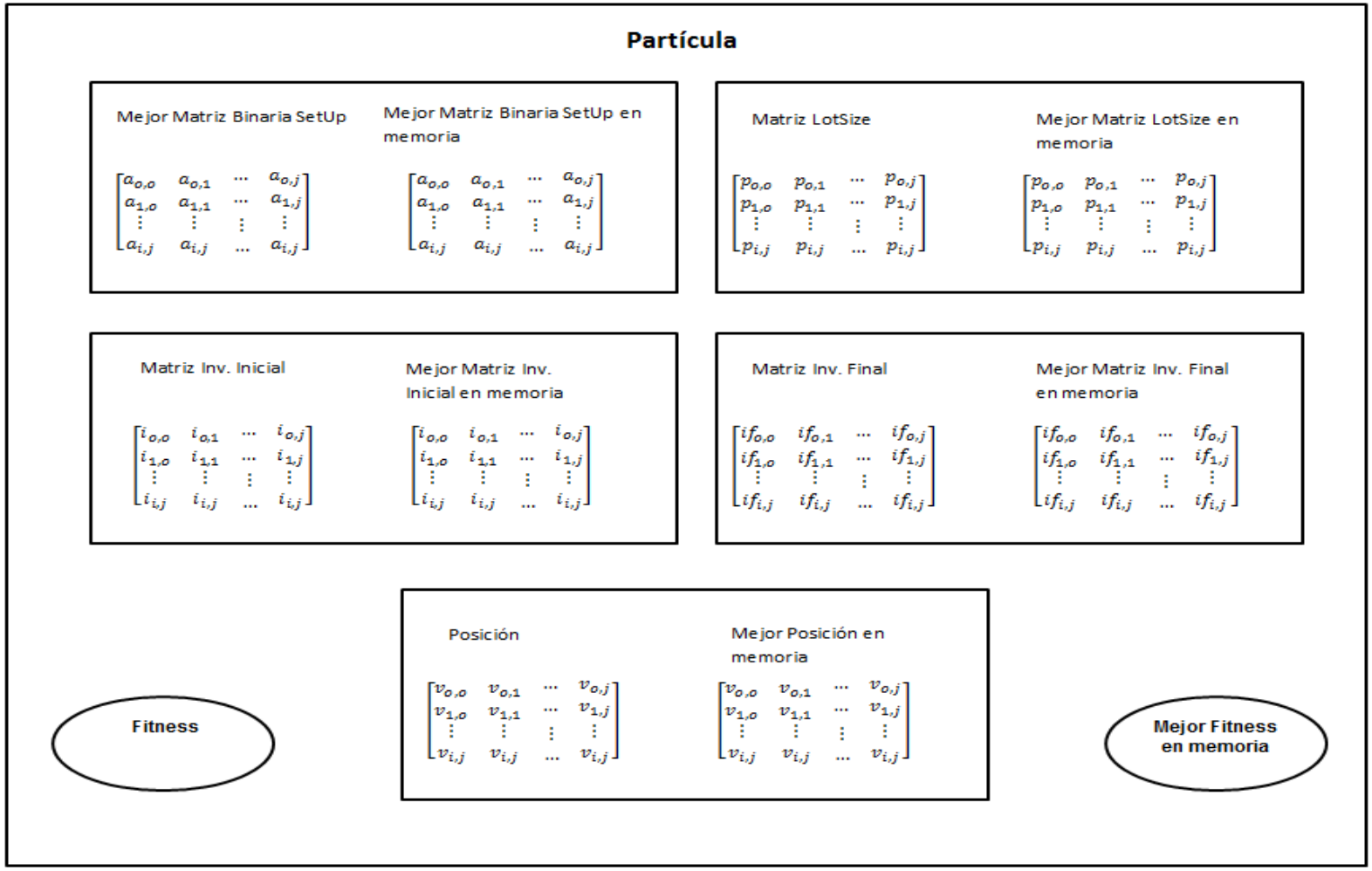
capacidad del sistema en el período *t*; por último se establecen las restricciones de no negatividad para el inventario inicial, final y el tamaño de lote (30).

7.2 DISEÑO DEL ESQUEMA DEL ALGORITMO

El algoritmo de optimización por enjambre de partículas es la metaheurística que va a ser utilizada para determinar la matriz binaria de set up apropiada para cada solución, a partir de ésta se utiliza el modelo matemático para determinar el tamaño de lote, por último se afina la solución mediante una búsqueda local que modifique las velocidades de las partículas con el fin de proveer una mejor solución, a este proceso se le conoce como una hiperheurística.

Inicialmente la representación de una partícula se puede observar en la figura 5, adicionalmente se presenta en la figura 6 el diagrama de flujo del algoritmo detallado por los procesos generales, por último en la figura 7 se detalla el subproceso de la generación de la mejor partícula del enjambre que sirva como base para el diseño de la librería.

Figura 5. Representación de una partícula del enjambre.



Fuente: Elaboración propia

Figura 6. Diagrama de flujo del proceso general del algoritmo.

Fuente: Elaboración propia

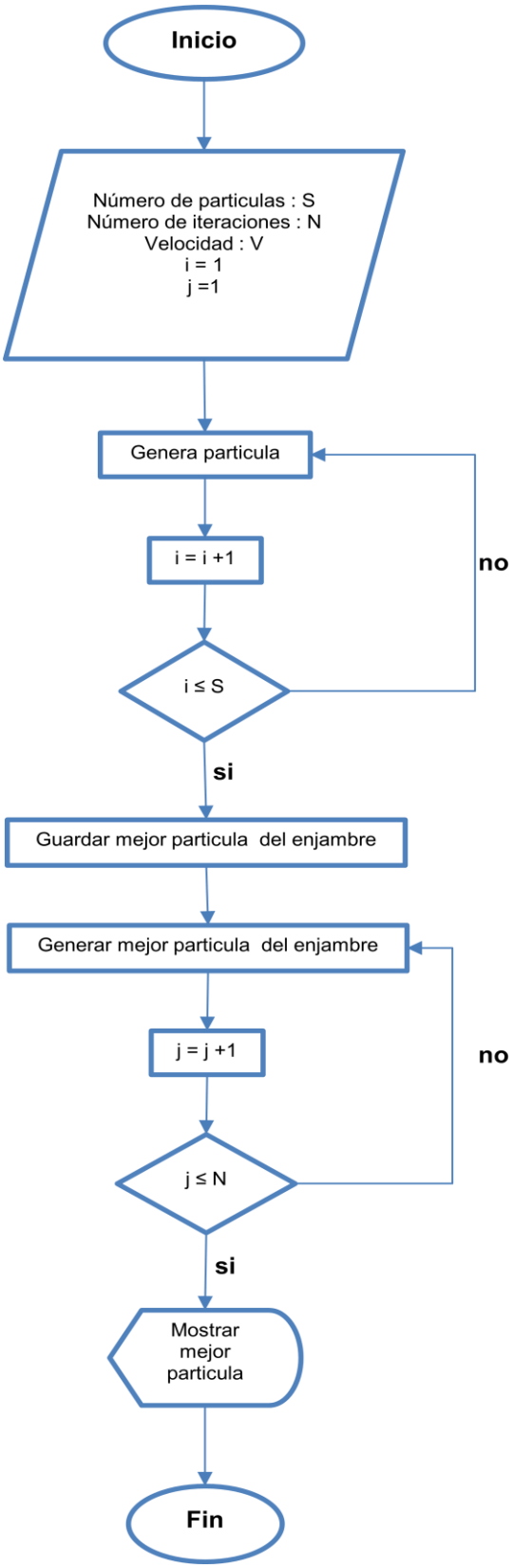
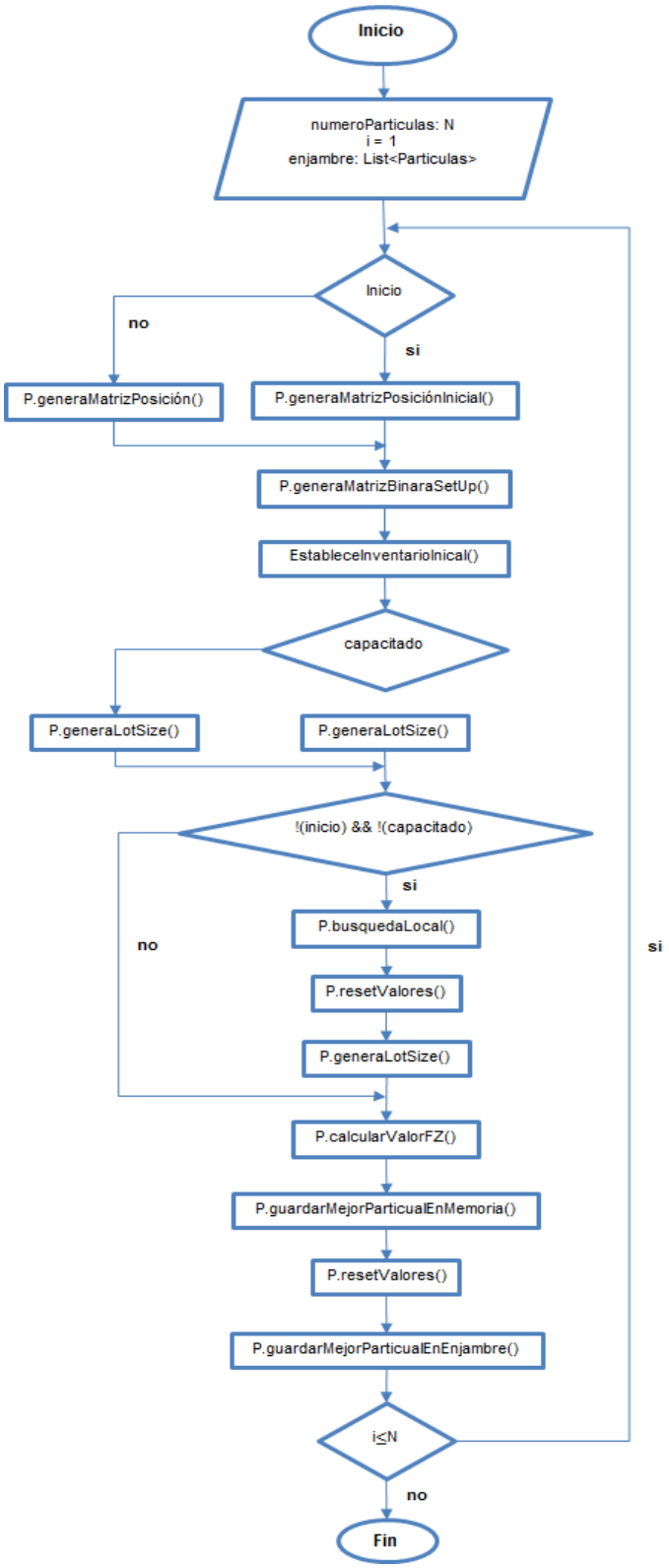


Figura 7. Subproceso de generación de la mejor partículas del enjambre.

Fuente: Elaboración propia



7.3 DISEÑO DE LA BÚSQUEDA LOCAL

El proceso de la búsqueda local corresponde a un subproceso dentro de la generación de la mejor partícula del enjambre; para diseñar la búsqueda local, se tienen en cuenta los parámetros del modelo tanto variables críticas como no criticas y según la técnica a utilizar puede ofrecer mejores o peores resultados.

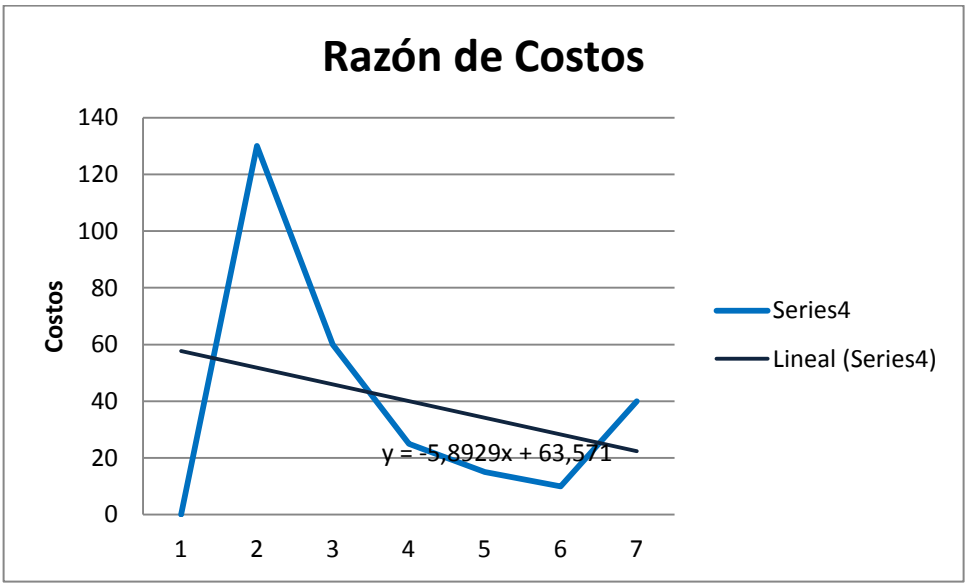
Para el caso de estudio propuesto con anterioridad se utilizará una búsqueda local con base al arrastre de la demanda, es decir la heurística para resolver este problema se basa en que los tiempos de entregas son despreciables y cumple con la característica que la pendiente de la razón de los costos de almacenamiento y costos de set up es decreciente (Negativa) tal como se observa en la figura 8 esto es porque realizar el set up de un producto (componente) de nivel inferior sale más barato que mantener en inventario ese mismo producto, en consecuencia se puede afirmar que al momento de hacer un pedido de un producto se debe arrastrar la demanda de los componentes, ósea presionar a los componentes a realizar un set up para tener inventario para satisfacer la demanda del producto de nivel superior.

Tabla 7 – Razón de costos de almacenamiento y set up.

Numero del producto	Costo de almacenamiento	Costo de set up	Razón de costo
0	1	130	130
1	2	120	60
2	1	25	25
3	2	30	15
4	3	30	10
5	1	40	40

Fuente: Elaboración propia.

Figura 8. Gráfica de la tendencia de los costos

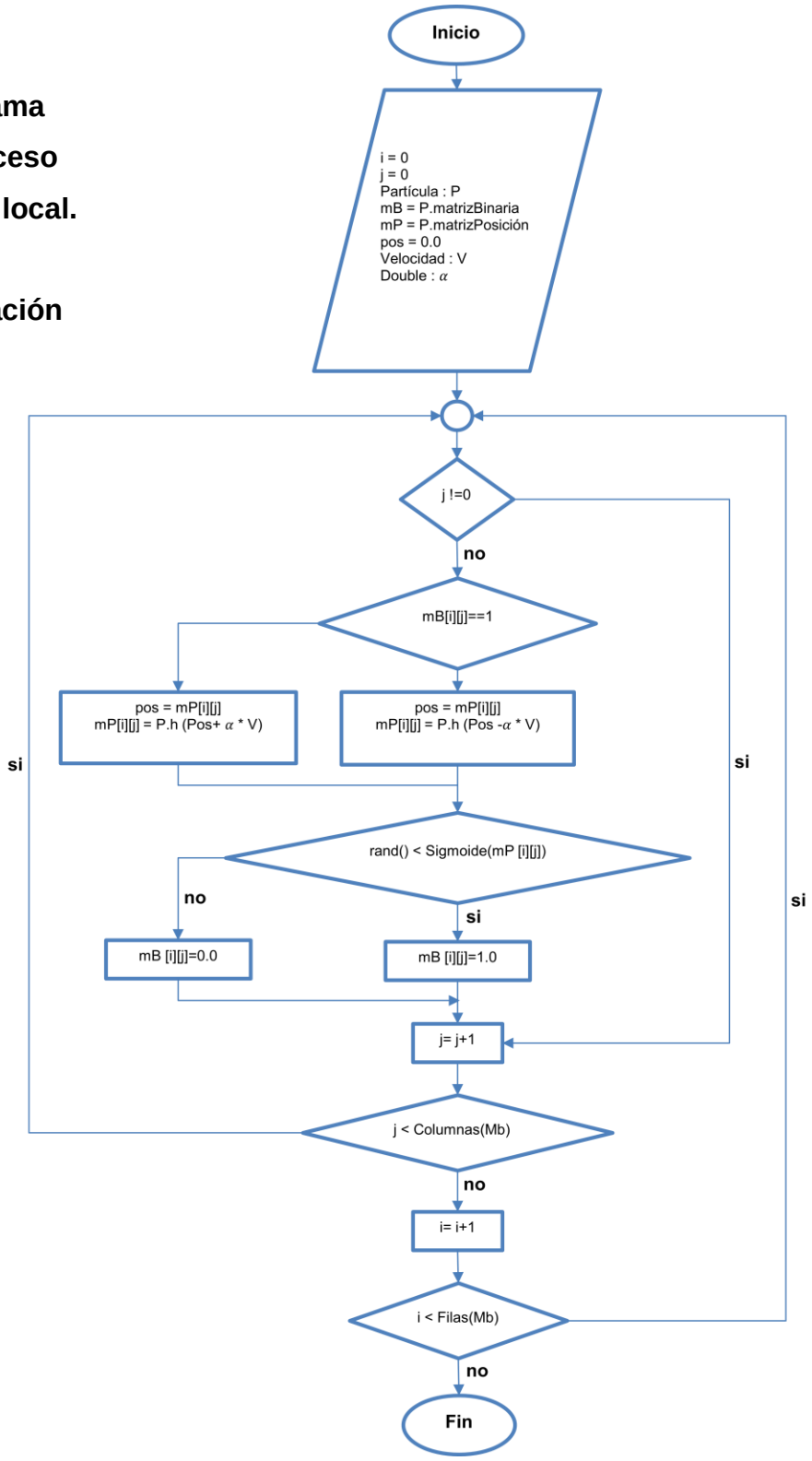


Fuente: Elaboración propia.

Con base a la teoría expuesta sobre el proceso de búsqueda local por arrastre de la demanda a realizar en el sistema, en la figura 9 se presenta el diagrama de flujo del proceso de la búsqueda local.

Figura 9. Diagrama de flujo del proceso de la búsqueda local.

Fuente: Elaboración propia



7.4 DESCRIPCIÓN DE REQUERIMIENTOS DE LA LIBRERÍA

Con base al problema al que se busca dar solución y el planteamiento propuesto con el algoritmo de optimización por enjambre de partículas con búsqueda local, se presentan a continuación los requerimientos funcionales y no funcionales de la librería que permitan cumplir con las características iniciales deseadas de legibilidad y extensibilidad, además sirvan para proveer una solución al caso de estudio propuesto en el capítulo 6.3.

7.4.1 Requerimientos funcionales. Los requerimientos funcionales de un sistema describen lo que el sistema debe hacer. Estos requerimientos dependen del tipo de software que se desarrolle, de los posibles usuarios del software y del enfoque general tomado por la organización al redactar requerimientos. (SOMMERVILLE, 2006).

A continuación se presentan los requerimientos funcionales que la librería debe hacer para el desarrollo del software.

- ✓ RQF–01. Importar el modelo a la librería.
- ✓ RQF–02. Establecer los parámetros iniciales del algoritmo.
- ✓ RQF–03. Establecer los parámetros de la búsqueda local para el arrastre de la demanda.
- ✓ RQF–04. Ejecutar el algoritmo PSO.
- ✓ RQF–05. Exportar la solución encontrada por el algoritmo.

La descripción de los requerimientos funcionales se presentan en el anexo 1.

7.4.2 Requerimientos no funcionales. Los requerimientos no funcionales son aquellos requerimientos que no se refieren directamente a las funciones específicas que proporciona el sistema, sino a las propiedades emergentes de éste como la fiabilidad, el tiempo de respuesta y la capacidad de almacenamiento.

A continuación se describen los requerimientos no funcionales de la librería propuesta.

- ✓ RQNF–01. La librería debe ser extensible para la gestión de importación y exportación de modelos como también proveer una base para crear y adaptar nuevas búsquedas locales.
- ✓ RQNF–02. La librería debe importar el modelo mediante un formato Excel. (Ejemplo: *.xls)
- ✓ RQNF–03. La librería debe ofrecer un API con todas las funcionalidades del sistema para garantizar la legibilidad de ésta.

8. DISEÑO DE LA LIBRERÍA PSOLIB

Con base a la adaptación de la técnica de optimización por enjambre de partículas propuesta para el desarrollo del problema tamaño de lote multinivel capacitado y a los requerimientos expuestos para el desarrollo de la librería, es importante definir una la arquitectura de software a seguir para el diseño y desarrollo de la herramienta computacional, además de utilizar el conjunto de técnicas que éste provee con el fin de garantizar un buen diseño del software

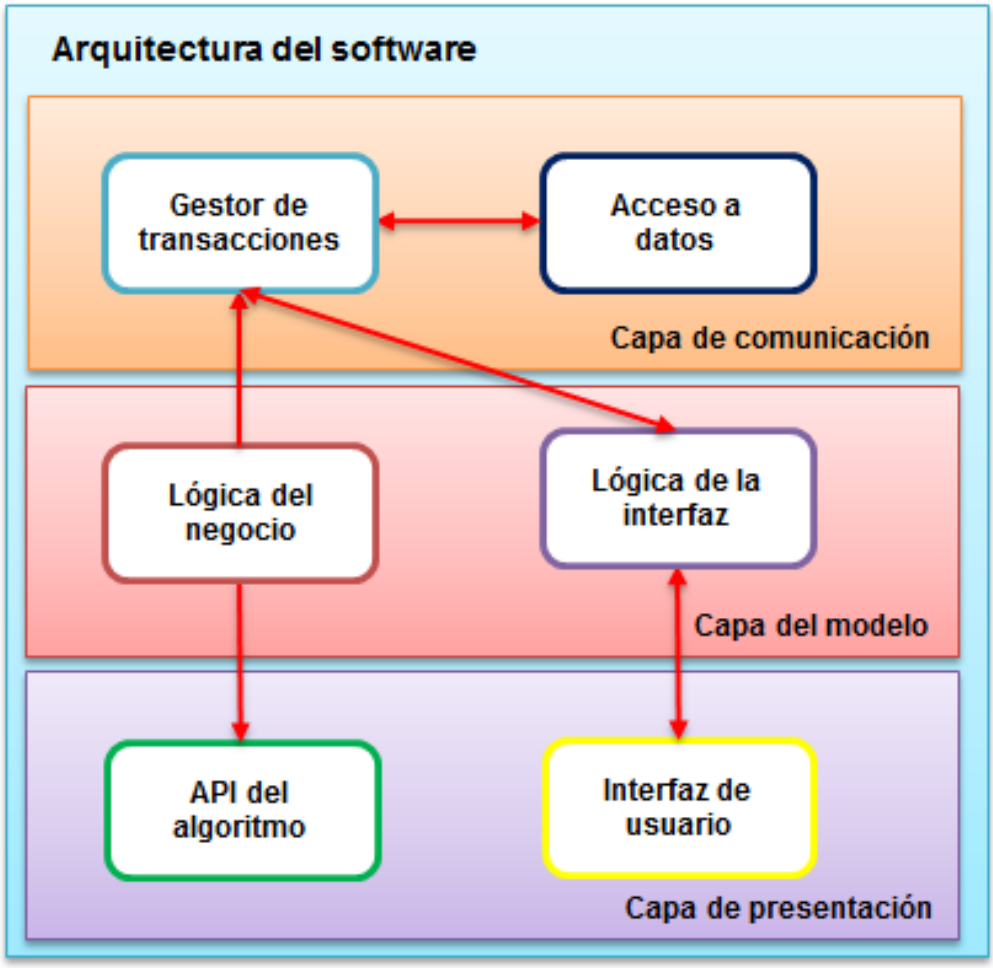
8.1 ARQUITECTURA DE SOFTWARE

Una arquitectura de software define la estructura general de un sistema y varía de acuerdo con el tipo de sistema a desarrollarse. Así, puede estar basada en elementos sencillos o componentes prefabricados de mayor tamaño, y se especifica de acuerdo con los diferentes tipos de sistemas. (Weitzenfeld, 2005).

Además de depender del tipo de sistema a desarrollar, la selección de una arquitectura afecta aspectos como la extensibilidad del sistema. Por lo tanto, la arquitectura debe ser escogida de manera que minimice los efectos de cambios futuros en el sistema.

Para el diseño de ésta librería, la figura 10 propone un modelo en capas, el cual jerarquiza las prioridades y comunicaciones de las capas, además provee un panorama general del software.

Figura 10. Representación de la arquitectura del software.



Fuente: Elaboración propia.

A continuación se presenta la descripción de cada capa conforme a la arquitectura del sistema:

- ✓ **Capa de comunicación.** Ésta depende de la lógica del negocio y sirve como puente para controlar las transacciones entre los datos y la lógica de la interfaz de manera bidireccional, además provee la base para importar y exportar los datos

- ✓ **Capa del modelo.** Esta capa es la encargada del procesamiento del algoritmo de optimización por enjambre de partículas y del control de la lógica de la interfaz presentada por la librería.

- ✓ **Capa de presentación.** En esta capa ofrece al usuario los modos de acceso a la librería mediante la interfaz interna de la librería y el API creado para la solución de diferentes problemas.

8.2 CASOS DE USO

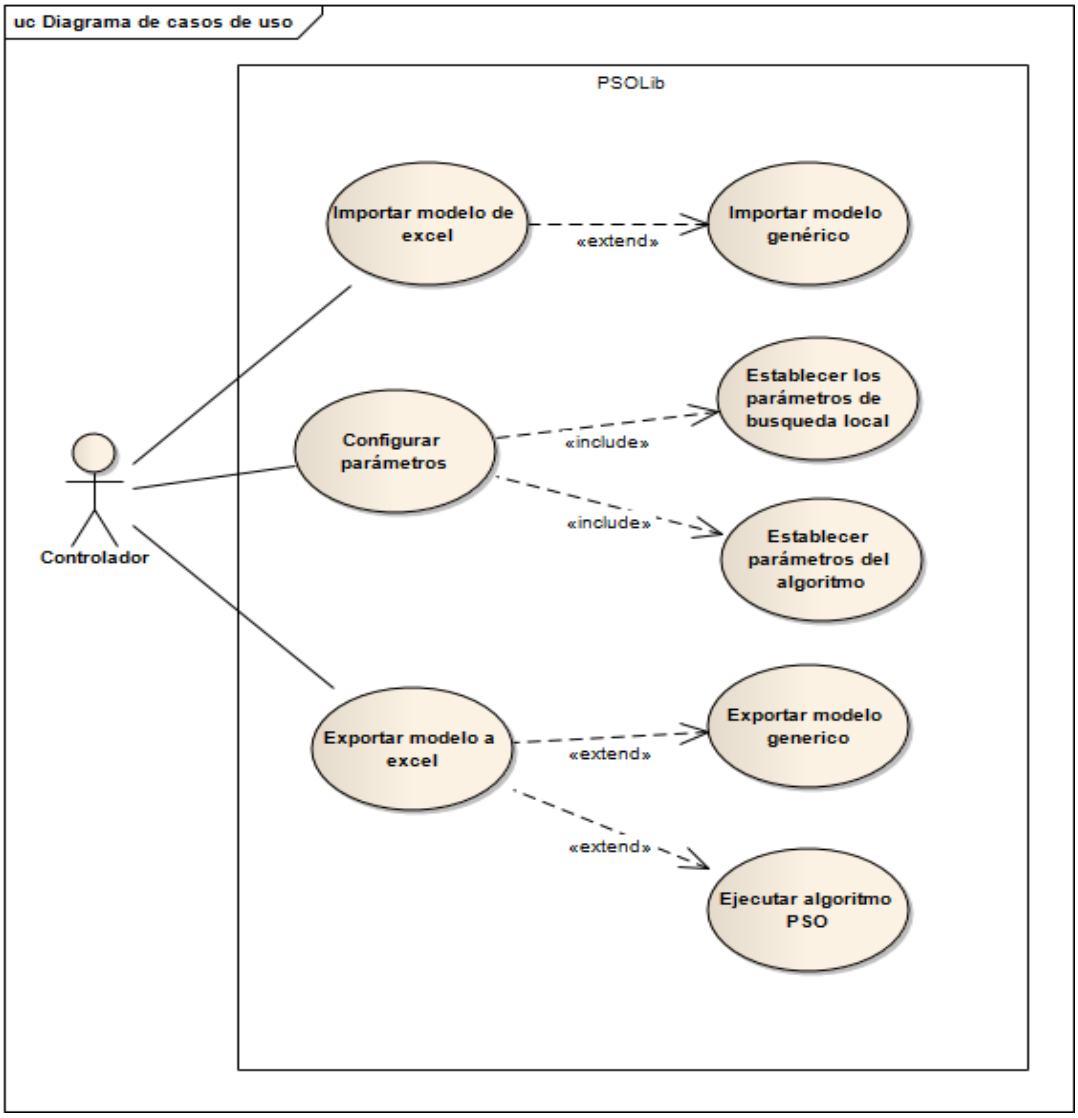
Un caso de uso documenta una interacción entre el software y un actor o más, dicha interacción tiene que ser, en principio, una función autónoma dentro del software; los casos de uso propuestos para este software están en el anexo 2 y poseen las siguientes características:

- ✓ Código
- ✓ Prioridad
- ✓ Nombre
- ✓ Descripción
- ✓ Actores principales
- ✓ Flujo normal
- ✓ Flujo Alternativo
- ✓ Requerimientos relacionados

Entre los actores que participan en un caso de uso, conviene distinguir el actor primario del caso de uso, que es el que lo pone en funcionamiento pidiendo la función correspondiente del software. (CAMPDERRICH, 2003).

Adicionalmente se describe el flujo normal y el flujo alternativo para facilidad al momento de la implementación de la secuencia del software. A continuación se presenta en la figura 11 el diagrama de casos de uso del sistema. En el anexo 3 se presentan los diagramas de secuencia de cada caso de uso.

Figura 11. Diagramas de casos de uso del sistema.



Fuente: Elaboración propia.

8.3 DIAGRAMAS ESTÁTICOS

Los diagramas estáticos o también llamados estructurales se encargan de definir qué elementos (entidades, objetos, áreas, clases, departamentos, componentes etc.) deben de estar definidas dentro del sistema u organización a desarrollar el correspondiente modelado. También se encarga de especificar cómo deben de estar estructurados estos elementos, mientras que los modelos dinámicos se encargan de definir el comportamiento de estos.

8.3.1 Diagrama de clases. Son diagramas que describen la estructura de un sistema donde se muestran sus clases, sus atributos y las relaciones que existen entre estos. Estos diagramas son utilizados en la etapa de análisis y diseño, en primera instancia se plantea el diseño conceptual de la información que se utilizara dentro del sistema y luego se desarrolla la parte de las componentes que se encargaran de las relaciones entre uno y otro.

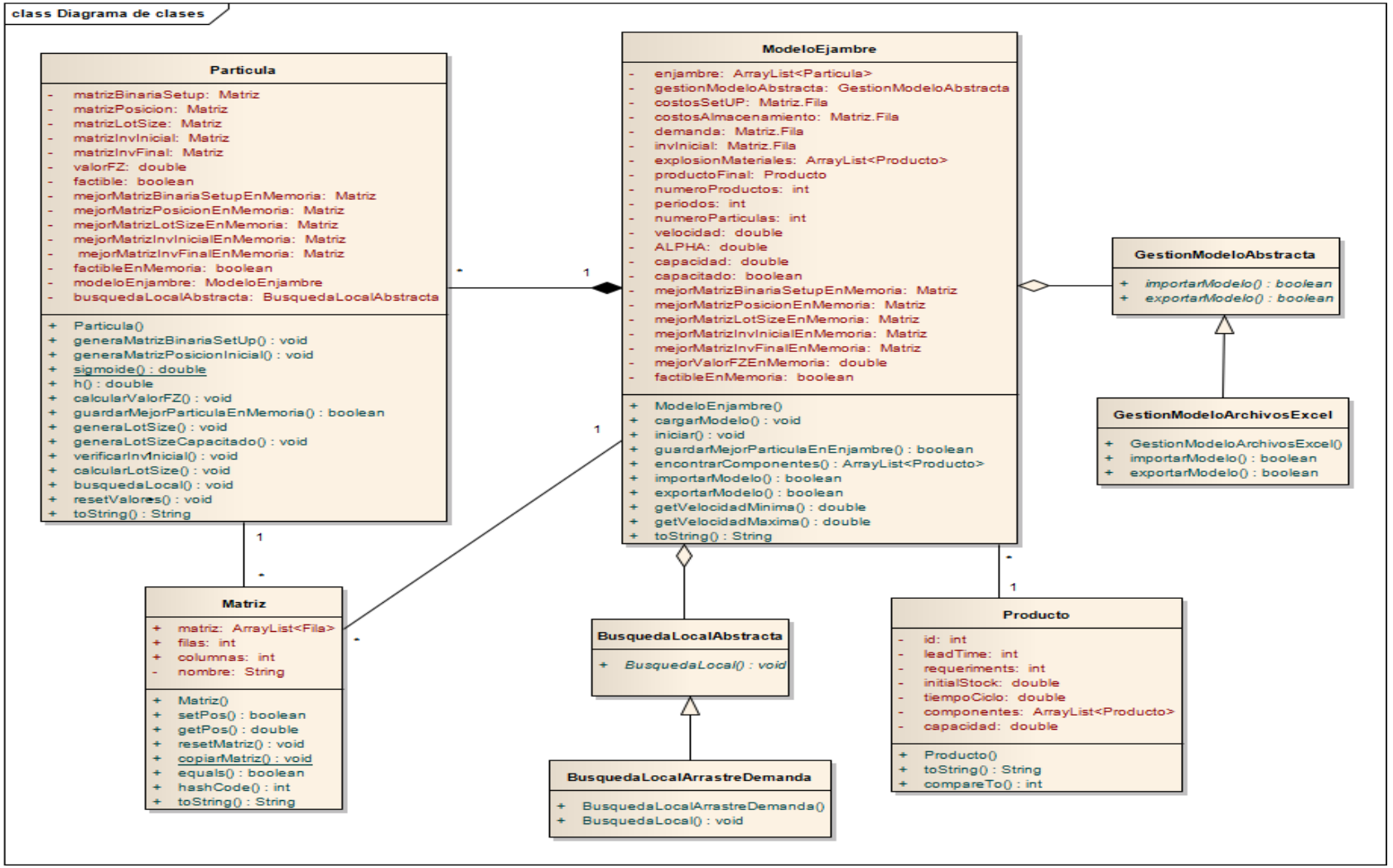
En la figura 12 se muestra el diagrama de clases de la lógica del modelo, teniendo a la clase ModeloEnjambre y Partícula como la base principal del algoritmo PSO; la clase ModeloEnjambre está compuesta de muchas Partículas y un árbol de *N*-ario de la clase Producto, ésta y la clase Matriz están en el paquete utilidades que sirven para la representación del problema, cada partícula guarda en memoria múltiples referencias a la clase Matriz como se presentó en la figura 5 (Representación de una partícula).

Existen dos representaciones abstractas para la gestión del modelo y la búsqueda local, diseñadas bajo el patrón de comportamiento “Estrategia”, este patrón de diseño permite mantener un conjunto de algoritmos de entre los cuales el objeto cliente puede elegir aquel que le conviene e intercambiarlo dinámicamente según sus necesidades, además le permite al software obtener la característica de

extensible en diferentes contextos, éstas dos estrategias permiten importar y exportar el modelo a un archivo Excel (*.xls) y crear la búsqueda local conforme a la heurística propuesta para resolver el problema.

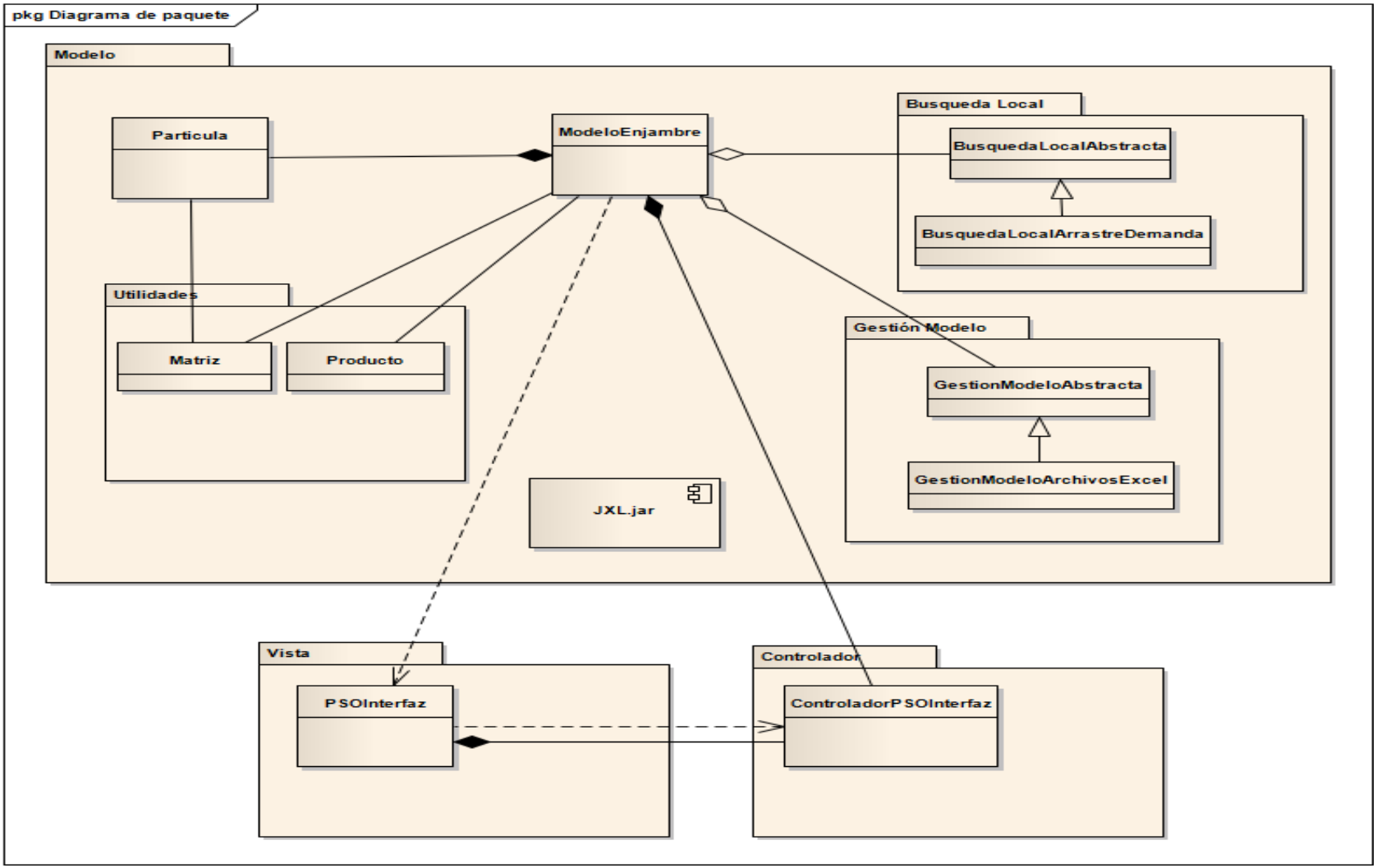
8.3.2 Diagrama de paquetes. Entre los diagramas estáticos podemos encontrar el diagrama de clases y el diagrama de paquetes, estos dos permiten especificar y visualizar las relaciones de dependencias que existe entre las clases y los paquetes que forman parte de la solución, respectivamente. En la figura 12 se presentan el diagrama de clases de la lógica del modelo y en la figura 13 se muestra el diagrama de paquetes correspondiente al panorama global del sistema y la relación con la arquitectura del software propuesta.

Figura 12. Diagrama de clases del sistema.



Fuente: Elaboración propia.

Figura 13. Diagrama de paquetes del sistema.



Fuente: Elaboración propia.

9. ANÁLISIS Y DISCUSIÓN DE LOS RESULTADOS

Conforme a la metodológica planteada, en la última fase se procede a ejecutar la librería como herramienta aplicativa para dar solución al caso de estudio, comparar el desempeño de esta frente a los resultados obtenidos (Presentados en el capítulo 6.3) y evaluar las soluciones para el modelo capacitado, así que para llevar a cabo esto se realiza un análisis de los parámetros iniciales, el cálculo del tamaño de la muestra para obtener un valor con 95% de confianza, un cuadro comparativo entre las soluciones encontradas por la librería y las presentadas en la literatura, presentar las condiciones en las que se probó la librería y por último el análisis del modelo capacitado.

9.1 ANÁLISIS DE LOS PARÁMETROS DEL ALGORITMO PSO

Una de las características de la técnica de optimización por enjambre de partículas corresponde a evaluar el desempeño de éste en diferentes contextos debido a la posibilidad obtener diferentes resultados según los parámetros de entrada, así que para el desarrollo de ésta adaptación del algoritmo al problema de la planificación del tamaño de lote se evalúan los siguientes parámetros conforme a los propuestos por el artículo guía.

- ✓ Numero de iteraciones
- ✓ Numero de partículas
- ✓ Velocidad Inicial

Adicionalmente se evalúa el factor de alfa (α) de porcentaje de aceleración de velocidad para la búsqueda local y se toma como referencia para los factores cognitivos y social el valor propuesto por los autores de dos (2) para cada factor.

El número de iteraciones corresponde al número de veces que se realizara una generación del enjambre para obtener la mejor partícula, las características a tener en cuenta son que a un número bajo de iteraciones, las partículas no tienden a converger por ende la respuesta no es confiable en cambio, a un numero grande de partículas se pueden incurrir en tiempos de ejecución innecesarios, con base a esto se probarán para 100, 300 y 500 (Propuesto por el autor) número de iteraciones.

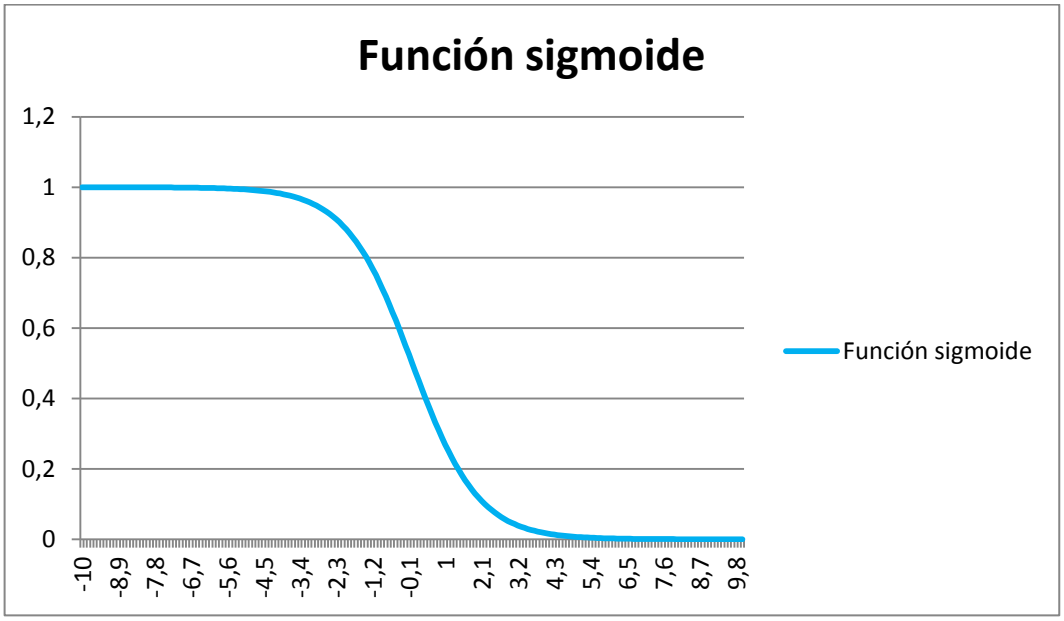
El número de partículas es una variable que determina el nivel de exploración del algoritmo, es decir, a mayor número de partículas se puede explorar el espacio de búsqueda con más profundidad aunque esto no garantice encontrar óptimos globales, en efecto el programa será evaluado con un numero de 20, 40 y 60 (Propuesto por el autor) partículas.

La velocidad inicial es un factor muy importante debido que éste determina si evaluar un set up como verdadero o falso; a medida que la velocidad sea mayor en el sistema aumenta el nivel de explotación del algoritmo dado que la generación de set up se ve más sesgada cuando la velocidad alcanza los límites máximos o mínimos y la velocidad es mayor de cuatro (4) o menor de menos cuatro (-4), tal como se puede apreciar en la figura 14 en la gráfica de la función sigmoide, por otro lado ésta característica disminuye el nivel de exploración gracias a que como éste está más sesgado no le da libertad a las partículas de incurrir en nuevas soluciones. Con base a lo expuesto anteriormente se evaluarán velocidades de 4, 6 (Propuesta por el autor como $0.1 \times N \times T$) y 10.

Por último el factor de aceleración alfa (α), es empleado en la búsqueda local de arrastre por demanda para “jalar” a los componentes del producto final a incurrir en un set up, a medida que el factor alfa éste entre 0.1 y 0.5, motiva en una menor

medida a que se realicen los set up a como si el valor fuera 0.6 y 1.0. El rango de alfa está entre 0.1 y 1.0. La evaluación de la herramienta se realizará con base a un alfa de 0.2, 0.5 y 0.9.

Figura 14. Gráfica de la función sigmoide.



Fuente: Elaboración propia.

En la tabla 8 se presentan los resultados obtenidos luego de haber ejecutado la aplicación tres veces con los parámetros establecidos, cabe resaltar la importancia del factor alfa el cual no siempre ofrece buenas soluciones cuando se toma el valor de 0.2, además de mencionar que el peor promedio (2903,3) se dio cuando se evaluó el algoritmo con 500 iteraciones, 60 partículas, una velocidad de 10.0 y alfa de 0.2, es lógico pensar que sesgar los set up con una alta velocidad y no permitir a la búsqueda local arrastrar la demanda de un componente no haya permitido obtener buenas soluciones, en cambio el mejor promedio (1900,3) se dio

cuando se evaluó el algoritmo con 500 iteraciones, 60 partículas, una velocidad de 4.0 y alfa de 0.9, a diferencia del peor caso, se intentó elevar la exploración con una velocidad más baja y explotar la búsqueda local con un alfa más alto.

Tabla 8. Análisis de los parámetros iniciales del algoritmo.

Análisis de parámetros del algoritmo											
			Número de iteraciones								
			100			300			500		
		Alfa	20	40	60	20	40	60	20	40	60
Velocidad	4	0.2	2448,3	2303,6	2482,6	2399,3	2196,3	2746	2318,6	2195,6	2265,6
		0.5	2249	2188	2325,3	2189,6	2386	2204,6	2368	2167,6	2202,6
		0.9	2166	2163	1955,6	2144	2081,3	1978,3	2062,3	2043,3	1900,3
	6	0.2	2218	2307,6	2193,3	2354,3	2198,3	2074,6	2488	2152,6	2124,6
		0.5	2119	2089,3	2077,6	2182,3	1993	2016,3	2060,3	2028,6	2057,6
		0.9	2257	2152,6	2123,6	2022	1974,3	2077	2038,3	2036,3	1950,3
	10	0.2	2561,6	3033	2698,6	2404	2414,6	2614	2424,6	2432,3	2903,3
		0.5	2186	2271	2194,6	2340	2051,3	2199,6	2277,6	2296,6	2066,6
		0.9	2283,3	2201	2209,6	2410	2117,6	2232,3	2292,6	2339	2176,6

Fuente: Elaboración propia.

En consecuencia, para la evaluación real del algoritmo los parámetros establecidos serán los mismos que se encontraron en el mejor promedio:

- ✓ 500 iteraciones
- ✓ 60 partículas
- ✓ Velocidad de 4.0
- ✓ Alfa de 0.9

9.2 CÁLCULO DE LA MUESTRA ÓPTIMA

Para realizar un análisis comparativo entre los propuestos por los autores sobre el caso de estudio y los encontrados por esta herramienta, es necesario realizar un estudio de la muestra para garantizar una confiabilidad en los resultados, esto es por el hecho que las respuestas halladas son probabilísticas y hacen que los resultados sean variables.

Para el estudio de la muestra se tiene la características que el tamaño de la población de soluciones es extremadamente amplio, por no decir que es infinito, esto es debido a la categoría del problema como *NP-HARD* o $N \to \infty$, por lo tanto se debe tomar una pre muestra con el fin de determinar la varianza de está e introducir el valor en la formula (31) del tamaño de la muestra para obtener el valor deseado.

$$n = \frac{Z_{\alpha/2}^2 \cdot \sigma^2}{e^2} \qquad \textbf{(31)}$$

Para efecto de condiciones estables se tomará un nivel de confianza del 95% y un error máximo de ± 25, en la tabla 9 se presenta los 10 valores de la premuestra y el valor de la varianza de los datos.

Tabla 9. Datos de la premuestra.

X_i	1	2	3	4	5	6	7	8	9	10	Varianza (σ)
Fitness de la solución (X_i)	1895	1895	1911	1927	2002	1979	2112	1895	1902	2045	5760,67778

Fuente: Elaboración propia.

Luego de haber tomado la premuestra se calcula el valor de la muestra utilizando los valores obtenidos en la ecuación (31), dando como resultado un tamaño de muestra de 35.

9.3 ANÁLISIS COMPARATIVO

Conforme al número de muestras obtenidas para determinar la media y la desviación de las soluciones con un nivel de confianza del 95%, se presenta a continuación la tabla 10 con las soluciones obtenidas luego de ejecutar el algoritmo 35 veces.

Teniendo en cuenta los resultados propuestos por el autor presentado en la tabla 6 (Presentado en el capítulo 6.3), se puede observar que las soluciones de la adaptación del algoritmo de optimización por enjambre de partículas propuesta en este trabajo $(\mu = 2008{,}485)$ son mejor en un 6,31% a las propuestas por el autor $(\mu = 2143{,}94)$, aunque el tiempo de ejecución es ligeramente mayor por 2 segundos, además en ambos se lograron llegar al óptimo global aunque la peor solución encontrada por el algoritmo fue 2199 a diferencia del propuesto por el autor de 5565.

Tabla 10. Tabla de las ejecuciones del problema.

Tabla de resultados					
1	2065	7,278	**19**	1911	7,388
2	2082	7,301	**20**	1902	7,282
3	**2199**	**7,35**	**21**	2086	7,285
4	2070	7,28	**22**	2152	7,335
5	1902	7,26	**23**	1994	7,281
6	1909	7,286	**24**	1979	7,289
7	1972	7,355	**25**	2124	7,298
8	2045	7,283	**26**	1909	7,29
9	1911	7,389	**27**	2129	7,284
10	1964	7,276	**28**	1979	7,298
11	2063	7,299	**29**	1941	7,303
12	2120	7,3	**30**	**1895**	**7,322**
13	2132	7,324	**31**	**1895**	**7,303**
14	2082	7,313	**32**	2082	7,296
15	2056	7,277	**33**	1932	7,356
16	1936	7,302	**34**	1959	7,286
17	2046	7,341	**35**	1979	7,349
18	**1895**	**7,317**	**Promedio**	**2008,48571**	**7,30787571**

Fuente: Elaboración propia.

9.4 ANÁLISIS DEL MODELO CAPACITADO

El caso de estudio propuesto no posee un modelo capacitado, sin embargo para probar la funcionalidad de la librería se propone capacitar el modelo mediante la inclusión de tiempos de ciclos en los productos que son ensamblados y probar con una capacidad al 20%, 40% y 60% de la capacidad total del sistema.

Para el modelo matemático del cálculo del tamaño de lote multinivel capacitado se agrega la restricción de capacidad como se presenta en la ecuación (29) que indica que el tamaño de lote a fabricar o comprar de un producto i en el período t multiplicado por el tiempo de ciclo del producto, no puede exceder la capacidad máxima de ese producto en el período t, conforme a esto se redefinen los parámetros del caso de estudio presentados en la tabla 5 del capítulo 6.3, presentados en la tabla 11.

Tabla 11. Información del caso de estudio capacitado.

Información del caso de estudio capacitado													
Numero del producto	**Relación entre los productos**						**Costo de almacenamiento**			**Costo de set up**			**Tiempo de ciclo**
	Precedencia			**Requerimientos**									
0	**Producto final**			1			1			130			1.0
1	0			1			2			120			1.0
2	0			3			1			25			1.0
3	1			2			2			30			0.0
4	1			4			3			30			0.0
5	2			2			1			40			0.0
T	1	2	3	4	5	6	7	8	9	10	11	12	
$D_{i,t}$	32	41	148	36	120	28	32	12	30	10	32	41	**562**

Fuente: Elaboración propia.

De lo anterior se afirma que se desprecia el tiempo de ciclos del producto 3, 4 y 5 debido a que son componentes, en cambio el producto 0, 1 y 2 poseen un tiempo de ciclo de 1.0 (Unidades de capacidad), la capacidad se determina mediante la multiplicación del porcentaje de capacidad por la demanda total, suponiendo entonces que no más del porcentaje de capacidad es capaz de fabricarse en un período. En la tabla 12 se presentan los valores de la capacidad y los resultados obtenidos. En el anexo 12, 13 y 14 se presenta los tamaños de lotes correspondientes a las soluciones obtenidas por la herramienta.

Tabla 12. Tabla con los porcentajes y los valores de capacidad del sistema.

	Porcentaje de capacidad (%)		
Prueba	20	40	60
Capacidad	112	224	337
Fitness	2775	2152	1939

Fuente: Elaboración propia.

.

9.5 DETALLES DE LA EJECUCIÓN

Debido a los distintos desempeños que se pueden presentar al momento de ejecutar el algoritmo mediante el uso de la librería, se especifica que la ejecución de las pruebas realizadas en este análisis fue hecha en un computador con las siguientes características técnicas:

- ✓ Procesador → AMD Phenom™ II N640 Dual-Core Processor 2.90 GHz
- ✓ RAM → 4,00 GB
- ✓ Sistema operativo → Windows 7 Home Basic
- ✓ Marca → Hewlett-Packard
- ✓ Modelo → HP Pavilion dv5 Notebook

10. CONCLUSIONES

Con base al desarrollo realizado sobre la metodología propuesta para solucionar el problema del tamaño de lote multinivel capacitado se realizó un diagnostico del contexto actual en la que los autores de la literatura especializada han trabajado con este problema y se logró determinar una estrategia eficiente y práctica para adaptar el algoritmo de optimización por enjambre de partículas a una naturaleza combinatoria.

Una vez encontrada la estrategia se logró establecer el modelo matemático correspondiente al problema y estructurar el esquema para obtener la solución con base al algoritmo propuesto y a la heurística de la búsqueda local, ésta técnica conocida como una hiperheurística se fundamentó principalmente en el caso de estudio y las características de este.

Luego de crear el esquema del algoritmo, se definieron los requerimientos funcionales y no funcionales que sirvieron como base para el desarrollo y diseño de una arquitectura del software que mediante la creación de diagramas de comportamientos y estáticos permitieran llevar a cabo el diseño óptimo de una librería que soportara métodos y funciones para solucionar el problema en el lenguaje de programación Java.

Posteriormente se implementó la librería PSOLib, que es producto de este trabajo y permitió solucionar el problema en cuestión tal como se había formulado en la hipótesis, además mediante ésta se analizó y evaluó el caso de estudio por lo cual nos lleva a decir que el algoritmo propuesto fue capaz de encontrar el óptimo del problema, reducir en un 6,31% la media propuesta por los autores y disminuir la generación de malas soluciones en el problema, adicionalmente para el mismo

problema capacitado se evaluó la herramienta para un porcentaje de capacidad del 20%, 40 y 60%, proveyendo soluciones factibles y de muy buen desempeño con base al modelo no capacitado.

Finalmente cabe resaltar que librería PSOLib fue hecha bajo los conceptos de legibilidad y extensibilidad, los cuales permiten a esta librería seguir creciendo a medida que se estudia en este campo, así que como trabajo futuro se busca seguir indagando sobre el problema de tamaño de lote, incluir nuevas variables y adaptar esta herramienta para la solución de problemas más complejos.

11. BIBLIOGRAFÍA

AFENTAKIS, P.; GAVISH, B. Optimal Lot-Sizing Algorithms for Complex Product Structures, Operations Research, (1986), P. 237-249.

ALMADA-LOBO, B.; KLABJAN, D.; CARRAVILLA, M.A.; OLIVEIRA, J. Multiple machine continuous setup lotsizing with sequence-dependent setups. 2007. International Journal of Production Research, 45(20). 2007. P. 4873 – 4894

ALVAREZ, Concepción Maroto. SORIA, Javier Alcaraz y GARCIA, Ruben Ruiz. Investigación Operativa: Modelos y técnicas de optimización. Editorial de la UPV. 2002. 424 páginas. 84-9705-239-0. P. 400

BARBAROSOGLU, G.; ÖZDAMAR, L. Analysis of solution space-dependent performance of simulated annealing: the case of the multi-level capacited lot sizing problem, Computers and Operations Research 27. 2000. P. 895-903.

BELVAUX, G.; WOLSEY, L. A, Lot-sizing problems: Modelling issues and a specialized branchand – cut system *BC-PROD*, Management Science 46. 2000. P. 724-738.

CAMPDERRICH, Benet. Ingeniería del software. Primera edición Editorial UOC, 2003, p 85, isbn 8484297934

CLARK, A. A Local Search Approach to Lot Sequencing and Sizing. Proceeding of the 2000. 2000

CORTÉS MORALES, R., Introducción al análisis de sistemas y la ingeniería de Software Roberto Cortés Morales, Primera edición. Costa Rica.: EUNED, 1998.P. 7-10, ISBN 978-9977-64-961-0.

CRAUWELS, H.A.J.; POTTS, C.N.; VAN WASSENHOVE, L.N. Local search heuristics for single-machine scheduling with bacthing to minimize the number of late jobs, European Journal of Operational Research 90. 1996. P. 200-213.

DELLART, N.; JEUNET, J.; JONARD, N., A genetic algorithm to solve the general multi-level lot-sizing problem with time-varying costs, International Journal of Production Research 68 (2000). P.241-257.

DISNEY, S.M.; NAIM, M.N.; TOWILL, D.R. Genetic algorithm optimisation of a class of inventory control systems, International Journal of Production Economics 68. 2000. P. 259-278.

DREXL, A.; HAASE. K. Proportional lotsizing and scheduling. International Journal of Production Economics, Vol. 40. 1995. P. 73-87.

DREXL, A.; KIMMS, A. Lot sizing and scheduling – survey and extensions. 1997. European Journal of Operational Research 99. P. 221–235.

FEDERGRUEN, A.; ZHENG, Y. S. (1992). The joint replenishment problem with general joint cost structures. Operations Research, 40, 384–403.

FLEISCHMANN, B. The Discrete Lot-sizing and Scheduling Problem. European Journal of Operational Research, 44. 1990. P. 337-348.

FLEISCHMANN, B.; MEYR, H. The General Lotsizing and Scheduling Problem. *OR Spektrum 19.* 1997. P. 11-21.

GAAFAR, L.K.; NASSEF, A.O.; ALY, A.I. Fixed-quantity dynamic lot sizing using simulated annealing. International Journal of Advanced Manufacturing Technology Vol. 41. 2009. P. 122

GAAFAR, L.K.; ALYA, A.S. Applying particle swarm optimisation to dynamic lot sizing with batch ordering. International Journal of Production Research Vol 47. 2009. P. 3345-3361.

GAREY, M. y JOHNSON, D. Computers and Intractability: A Guide to the Theory of NP-Completeness. Primera Edición. San francisco: W. H. Freeman, 1979. 340 páginas. 978-0716710455

GLOVER, F. Future paths for integer programming and links to artificial intelligence. Computers and Operations Research. 1986. Vol. 13, No. 5. p. 533-549.

GOPALAKRISHNAN, M.; DING, K.; BOURJOLLY, J.M.; MOHAN, S. A Tabu-Search Heuristic for the Capacitated Lot-Sizing Problem with Set-up Carryover. Management Science Vol 47. 2001. P. 851-863.

GRAVES, S. C., RINNOOY KAN, A.H.G., ZIPKIN, P.H. Handbooks in Operations Research and Management Science. Volumen 4 Logistics of Production and Inventory. 757 páginas. 978-0444874726

GRAJALES, Tevni. Tipos de investigación. (2000) 4 páginas.

HAN, Y.; TANG, J. F.; KAKU, I.; & MU, L. F. Solving incapacitated multilevel lotsizing problem using a particle swarm optimization with flexible inertial weight. Computers and Mathematics with Applications. 57 (2009). P.1748–1755.

HAN, Y.; KAKU, I.; GENGUI, Z.; JIANHU, C.; HUAZHEN, L. An Application of Particle Swarm Optimization to Uncapacitated Multilevel Lot-Sizing Problems with Series Structure. Proceedings of the IEEE International Conference on Computational Intelligence and Security, Vol 2. 2010. P. 541 – 544.

HINDI, K.S. Solving the CLSP by a Tabu Search Heuristic. The Journal of the Operational Research Society Vol. 47, No. 1. 1996. P. 151-161.

HSU, S. A modified particle swarm optimization to dynamic lot and customer order problems. APIEMS 2011 Conference Proceedings. 2011.

HYUN, C.J.; KIM, Y.; KIM, Y.K. A genetic algorithm for multiple objective sequencing problems in mixed model assembly lines. Computers and Operations Research, vol. 25, No. 7/8. 1998. P. 675-690.

IP, W.H.; LI, Y.; MAN, K.F.; TANG, K.S. Multi-product planning and scheduling using genetic algorithm approach, Computers & Industrial Engineering 38. 2000. P. 283-296.

KANG, S.; MALIK, K; THOMAS, L.J. Lotsizing and Scheduling on Parallel Machines with Sequence-Dependent Setup Costs, Management Science, Vol. 45. 1995

KENNEDY J, Eberhart R. Particle Swarm Optimization. In: Proceedings of the IEEE international conference on neural networks, vol. IV, Perth, Australia. Piscataway, NJ: IEEE Service Centre, 1995. p. 1942–8.

KIM, J.; KIM, Y. Simulated Annealing and Genetic Algorithms for Scheduling Products with Multi-level Product Structure, Computers Operations Research, Vol. 23, No. 9. 1996 P. 857-868.

KIMMS, A. A genetic algorithm for multi-level, multi-machine lot sizing and scheduling. Computers & Operations Research 26. 1999. P. 829-848.

KHOUJA, M.; MICHALEWICZ, Z.; WILMOT, M. The use of genetic algorithms to solve the economic lot size scheduling problem, European Journal of Operational Research 110. 1998. P. 509-524.

KUIK, R.; SALOMON. M.; VAN WASSENHOVE, L.N.; MAES, J. Linear programming, simulated annealing and tabu search heuristics for lotsizing in bottleneck assembly systems, IIE Transactions, vol. 25, N. 1. 1993. P. 62-72.

LANDEAU, Rebeca. Elaboración de Trabajos de Investigación. Editorial Alfa. 2007. 187 páginas. 980-354-214-1. P. 58

PARSOPOULOS, Konstantinos E.; VRAHATIS, Michael N. Particle Swarm Optimization and intelligence, Advances and applications. Premier Reference Source. 2010. 310 páginas. 978-1615206667. P. 16

PEREIRA, A.; CARVALHO, F.; CONSTANTINO, M.; PEDROSO, J. Random start search and tabu search for a discrete lot-sizing and scheduling problema. Computer Decision-Making. 2003. P. 575-600.

PESENTI, R.; UKOVICH, W. Economic Lot Scheduling on Multiple Production Lines with Resource Constraints, International Journal of Production Economics, 81-82. 2003. P. 469–481

PITAKASO, R.; ALMEDER, C.; DOERNER, K.F.; HARTL, R.F. A MAX_MIN ant system for unconstrained multi-level lot-sizing problems. Computers and Operations Research 34 (2007). P.2533-2552.

ORLICKY, J. Material requirements planning: the new way of life in production and inventory management. McGraw Hill Companies. 1975. 292 páginas.

OSMAN, I.H. y LAPORTE,G. Metaheuristics: A bibliography. En: Annals of Operations Research. 1996. No. 63. p. 513-623.

OSMAN, I.H. y KELLY J.P. Meta-Heuristics: theory and applications. Boston: Kluwer Academic. 1996. p. 467-487.

SALOMON, M. Deterministic Lotsizing Models for Production Planning. Springer Verlag. 1991.

SARKER, R.; YAO, X. Simulated annealing and joint manufacturing Batch-sizing. Yugoslav Journal of Operations Research 13 Number 2. 2003. P. 245-259

SOMMERVILLE, Ian. Ingeniería del software. Séptima Edición. México D.F.: Pearson Addisen Wesley, 2006. p 110. ISBN 84-7829-074-5.

TANG, O. Simulated annealing in lot sizing problems, International Journal of Production Economics 88 (2004). P.173-181

TONIOLO, A.; CLARK, A. A survey of lot-sizing and scheduling models. Operational Research Society. 2001. P. 938–947,

WAGNER, H.; WHITIN, T.M. Dynamic version of the Economic Lot-Size Model. Management Science, 5. 1958. P. 89-96.

WAJANAWICHAKON, K.; PITAKASO, R. Solving large unconstrainted multi level lot-sizing problem by a binary particle swarm optimization. International Journal of Management Science and engineering Management. 2011. P.143-141

WEITZENFELD, A. Ingeniería de software orientada a objetos con UML, Java e Internet, Primera edición. México D.F.: Thomson, 2005, P. 22-28, ISBN 970-686-1 90-4.

WOLSEY, L.A. MIP Modelling of Changeovers in Production Planning and Scheduling Problems. European Journal of Operational Research 99. 1997. P. 154-161.

XIAO, Yiyong; KAKU, Ikou; ZHAO, Qiuhong; ZHANG, Renqian. A variable neighborhood search based approach for uncapacitated multilevel. Computer and industrial engineering, 60 (2011). P.218-227

XIE, J.; DONG, J.F. Heuristic genetic algorithm for general capacitated lot-sizing problems. Computer and Mathematics with Applications 44. 2002. P. 263-276.

Anexo 1. RQF–01

ID	RQF–01
Descripción	Importar el modelo a la librería.
Descripción detallada	El sistema debe proveer la opción para ingresar el modelo a partir de un archivo Excel, el cual tiene un diseño previo (ver Anexo 5, Diseño del modelo) para cargar al sistema.
Entrada	Archivo de Excel (*.xls)
Salida	Modelo del problema
Excepciones	El fichero no corresponda a un archivo Excel o el modelo no sea cargado correctamente, se notificará al usuario
Prioridad	Alta

Fuente: Elaboración propia.

Anexo 2. RQF–02

ID	RQF–02
Descripción	Establecer los parámetros iniciales del algoritmo
Descripción detallada	El usuario deberá seleccionar los parámetros para la ejecución del algoritmo, tal como el número de iteraciones, el número de partículas y la velocidad del modelo. y valor alfa de la aceleración en la búsqueda local.
Entrada	Parámetros del modelo
Salida	Modelo actualizado
Excepciones	El numero de iteraciones, numero de partículas y la velocidad no puede pasar del rango previsto. Si esto ocurre se lo notifica al usuario.
Prioridad	Alta

Fuente: Elaboración propia.

Anexo 3. RQF–03

ID	RQF–03
Descripción	Establecer los parámetros de la búsqueda local para el arrastre de la demanda.
Descripción detallada	El usuario deberá seleccionar el valor alfa de la aceleración en la búsqueda local.
Entrada	Valor de alfa.
Salida	Modelo actualizado
Excepciones	El valor de alfa no puede pasar del rango previsto. Si esto ocurre se lo notifica al usuario.
Prioridad	Media

Fuente: Elaboración propia.

Anexo 4. RQF–04

ID	RQF–04
Descripción	Ejecutar el algoritmo PSO.
Descripción detallada	El sistema debe proveer las funciones y métodos para ejecutar el problema luego que el modelo este cargado al sistema.
Entrada	Modelo del problema
Salida	Modelo actualizado con la respuesta
Excepciones	
Prioridad	Alta

Fuente: Elaboración propia.

Anexo 5. RQF–05

ID	RQF–05
Descripción	Exportar la solución encontrada por el algoritmo.
Descripción detallada	El sistema deberá exportar a un fichero la información del modelo actualizado con la solución provista por el algoritmo.
Entrada	Modelo del problema actualizado
Salida	Archivo Excel (*.xls)
Excepciones	Se notifica al usuario si el archivo está abierto o siendo ejecutado por otro programa.
Prioridad	Alta

Fuente: Elaboración propia.

Anexo 6. CDU–01

<table>
<tr><td>Código</td><td>CDU–01</td></tr>
<tr><td>Prioridad</td><td>Alta</td></tr>
<tr><td>Nombre</td><td>Importar modelo de Excel</td></tr>
<tr><td>Descripción</td><td>El sistema debe proveer la opción para ingresar el modelo a partir de un archivo Excel para cargar al sistema.</td></tr>
<tr><td>Actores</td><td>Controlador</td></tr>
<tr><td>Flujo normal</td><td>
<table>
<tr><th>Acción del controlador</th><th>Acción del sistema</th></tr>
<tr><td>1. Seleccionar “Importar Modelo”</td><td>2. Abrir un seleccionador de archivos</td></tr>
<tr><td>3. Escoger el fichero en el cual se encuentra el modelo</td><td>4. Cargar el modelo al sistema</td></tr>
<tr><td></td><td>5. Notificar al usuario que el modelo fue cargado exitosamente</td></tr>
</table>
</td></tr>
<tr><td>Flujo Alternativo</td><td>4.1 Notificar al usuario que el modelo NO fue cargado correctamente.</td></tr>
<tr><td>Requerimientos relacionados</td><td>RQF–01</td></tr>
</table>

Fuente: Elaboración propia.

Anexo 7. CDU–02

<table>
<tr><td>Código</td><td>CDU–02</td></tr>
<tr><td>Prioridad</td><td>Alta</td></tr>
<tr><td>Nombre</td><td>Configurar parámetros</td></tr>
<tr><td>Descripción</td><td>El sistema debe proveer la opción al usuario para ingresar los parámetros del algoritmo.</td></tr>
<tr><td>Actores</td><td>Controlador</td></tr>
<tr><td>Flujo normal</td><td>
<table>
<tr><th>Acción del controlador</th><th>Acción del sistema</th></tr>
<tr><td>1. Ingresar el número de iteraciones</td><td></td></tr>
<tr><td>2. Ingresar el número de partículas</td><td></td></tr>
<tr><td>3. Ingresar la velocidad del sistema</td><td></td></tr>
<tr><td>4. Ingresar el valor de alfa</td><td></td></tr>
<tr><td>5. Establecer si es capacitado</td><td></td></tr>
<tr><td>6. Establecer parámetros</td><td>7. Cargar los parámetros al modelo</td></tr>
</table>
</td></tr>
<tr><td>Flujo Alternativo</td><td>7.1 Notifica si el número de iteraciones, el número de partículas, la velocidad o el valor de alfa no se pasa del rango previsto.</td></tr>
<tr><td>Requerimientos relacionados</td><td>RQF–02 , RQF–03</td></tr>
</table>

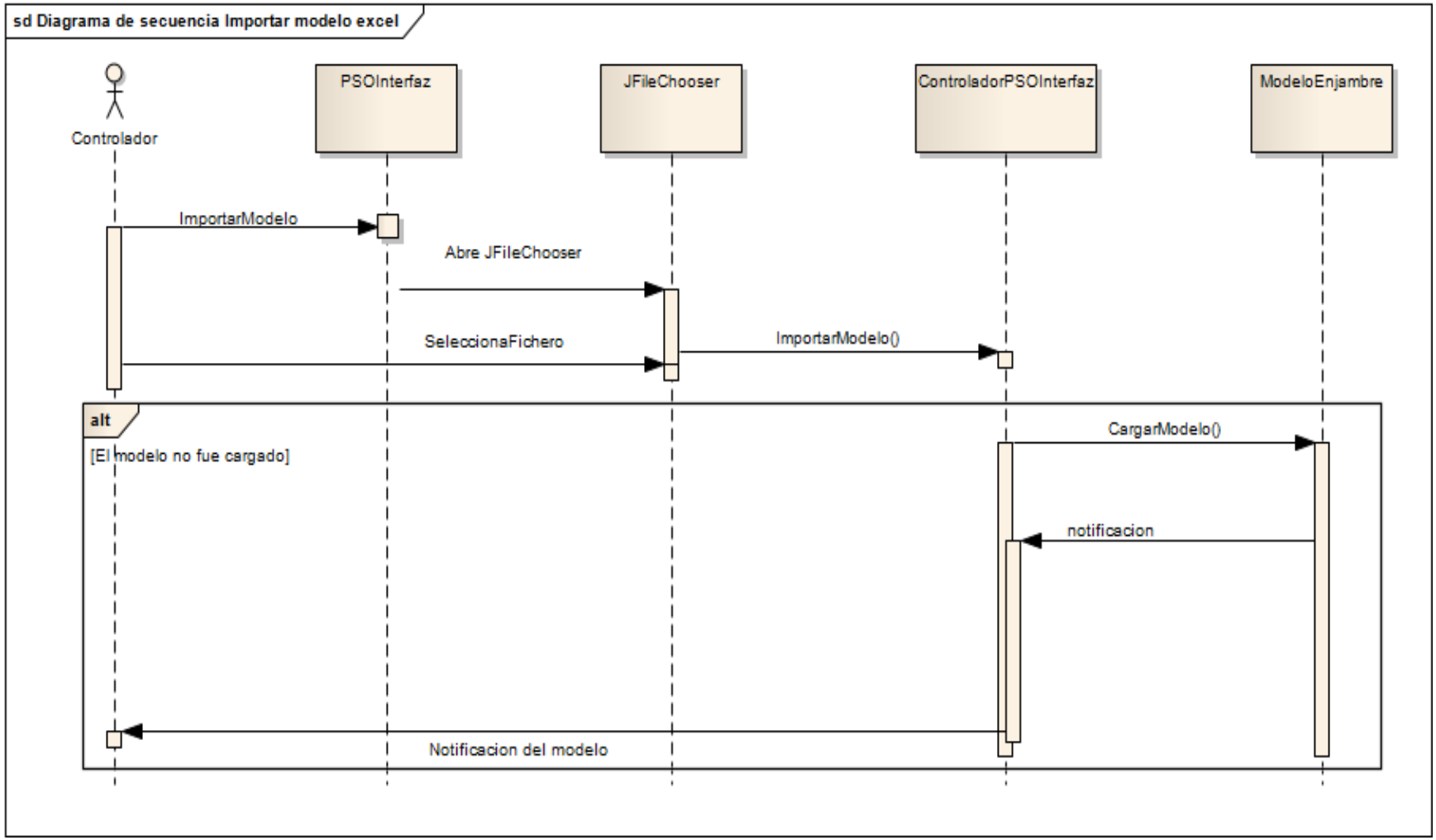
Fuente: Elaboración propia.

Anexo 8. CDU–03

<table>
<tr><td>Código</td><td>CDU–03</td></tr>
<tr><td>Prioridad</td><td>Alta</td></tr>
<tr><td>Nombre</td><td>Exportar modelo a Excel</td></tr>
<tr><td>Descripción</td><td>El sistema debe ejecutar el algoritmo luego que esté cargado y exportar la información al archivo en el cual se encontraba el modelo.</td></tr>
<tr><td>Actores</td><td>Controlador</td></tr>
<tr><td>Flujo normal</td><td>
<table>
<tr><th>Acción del controlador</th><th>Acción del sistema</th></tr>
<tr><td>1. Selecciona “Exportar Modelo”</td><td>2. El sistema ejecuta el algoritmo</td></tr>
<tr><td></td><td>3. El sistema exporta la información cargada al fichero de salida</td></tr>
<tr><td></td><td>4. El sistema notifica al usuario que el proceso terminó</td></tr>
</table>
</td></tr>
<tr><td>Flujo Alternativo</td><td>4.1 El sistema notifica una excepción por si el archivo destino está abierto</td></tr>
<tr><td>Requerimientos relacionados</td><td>RQF–04 , RQF–05</td></tr>
</table>

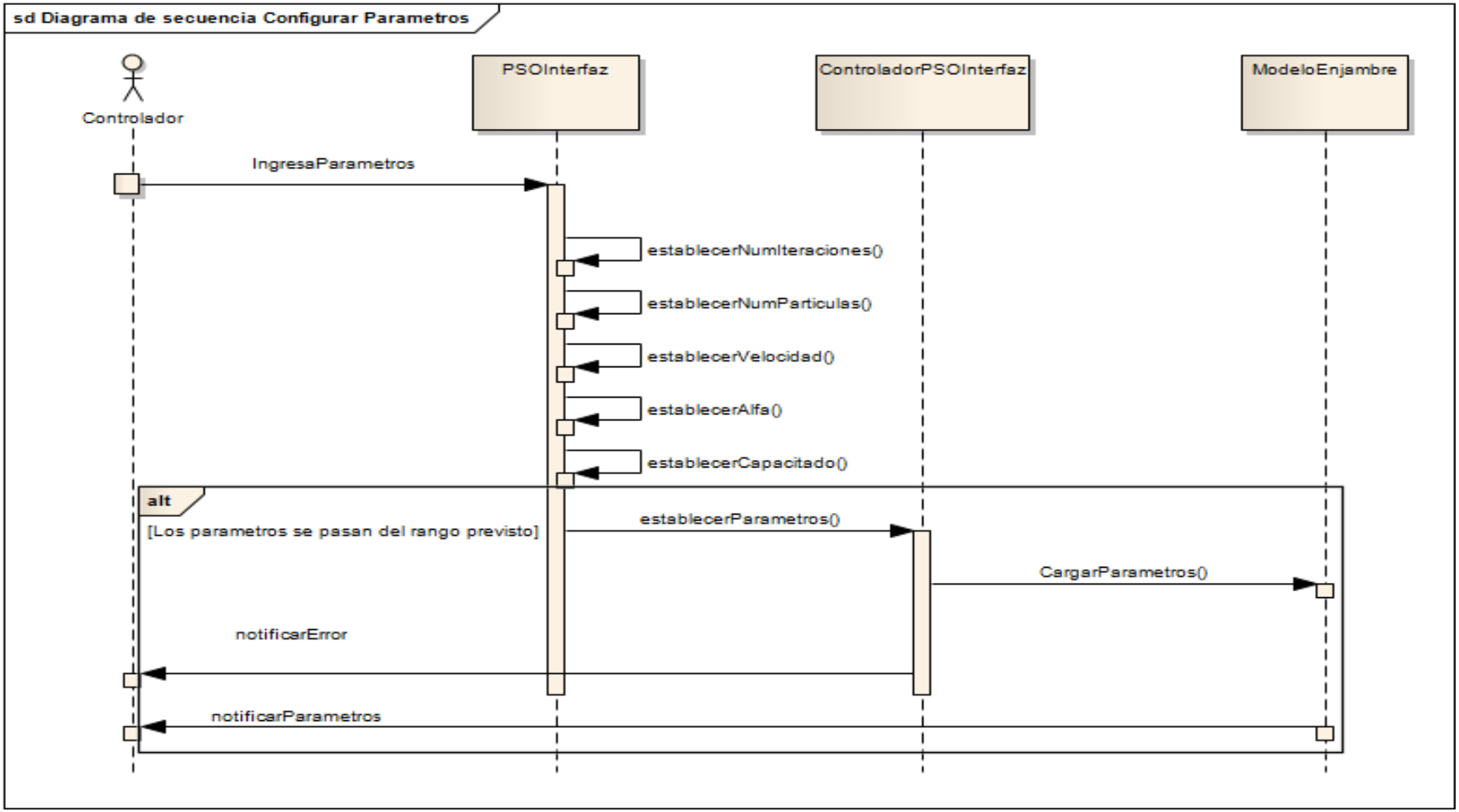
Fuente: Elaboración propia.

Anexo 9. Diagramas de secuencia del caso de uso CDU–01



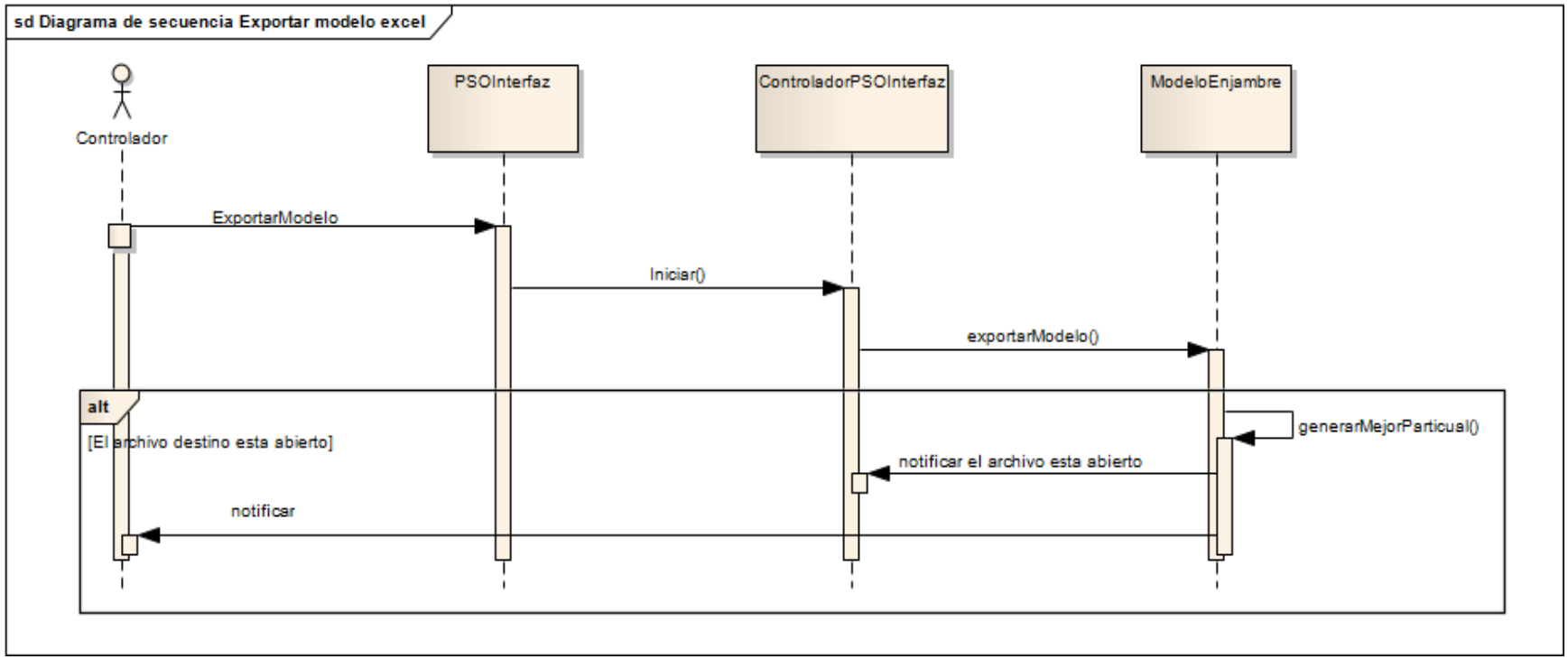
Fuente: Elaboración propia.

Anexo 10. Diagramas de secuencia del caso de uso CDU–02



Fuente: Elaboración propia.

Anexo 11. Diagramas de secuencia del caso de uso CDU–03



Fuente: Elaboración propia.

Anexo 12. Solución para el modelo capacitado al 20%

Matriz binaria de set ups											
1	0	1	1	1	0	0	1	0	1	0	0
1	0	1	1	1	0	0	1	0	1	0	0
1	0	1	1	1	0	0	1	0	1	0	0
1	0	1	1	1	0	0	1	0	1	0	0
1	0	1	1	1	0	0	1	0	1	0	0
1	0	1	1	1	0	0	1	0	1	0	0
Matriz del tamaño de lote											
112	0	112	112	112	0	0	42	0	72	0	0
112	0	112	112	112	0	0	42	0	72	0	0
336	0	336	336	336	0	0	126	0	216	0	0
224	0	224	224	224	0	0	84	0	144	0	0
448	0	448	448	448	0	0	168	0	288	0	0
672	0	672	672	672	0	0	252	0	432	0	0
FITNESS:	**2775**										

Fuente: Elaboración propia.

Anexo 13. Solución para el modelo capacitado al 40%

Matriz binaria de set ups											
1	0	1	1	0	0	1	0	0	0	0	0
1	0	1	1	0	0	1	0	0	0	0	0
1	0	1	1	0	0	1	0	0	0	0	0
1	0	1	1	0	0	1	0	0	0	0	0
1	0	1	1	0	0	1	0	0	0	0	0
1	0	1	1	0	0	1	0	0	0	0	0
Matriz del tamaño de lote											
73	0	148	184	0	0	157	0	0	0	0	0
73	0	148	184	0	0	157	0	0	0	0	0
219	0	444	552	0	0	471	0	0	0	0	0
146	0	296	368	0	0	314	0	0	0	0	0
292	0	592	736	0	0	628	0	0	0	0	0
438	0	888	1104	0	0	942	0	0	0	0	0
FITNESS:	**2152**										

Fuente: Elaboración propia.

Anexo 14. Solución para el modelo capacitado al 60%

Matriz binaria de set ups											
1	0	1	0	1	0	0	0	0	1	0	0
1	0	1	0	1	0	0	0	0	1	0	0
1	0	1	0	1	0	0	0	0	1	0	0
1	0	1	0	1	0	0	0	0	1	0	0
1	0	1	0	1	0	0	0	0	1	0	0
1	0	1	0	1	0	0	0	0	1	0	0
Matriz del tamaño de lote											
73	0	184	0	222	0	0	0	0	83	0	0
73	0	184	0	222	0	0	0	0	83	0	0
219	0	552	0	666	0	0	0	0	249	0	0
146	0	368	0	444	0	0	0	0	166	0	0
292	0	736	0	888	0	0	0	0	332	0	0
438	0	1104	0	1332	0	0	0	0	498	0	0
FITNESS:	**1939**										

Fuente: Elaboración propia.

Anexo 15. Formato de la plantilla del modelo Excel

<table>
<tr><td>ITEMS</td><td></td><td></td><td></td><td></td></tr>
<tr><td>Numero de productos del sistema</td><td></td><td></td><td></td><td></td></tr>
<tr><td>BOM</td><td></td><td></td><td></td><td></td></tr>
<tr><td>Producto(i) *</td><td>Requerimientos del producto(i)</td><td>Componente(j) del producto(i)</td><td>Requerimientos del componente(i, j)</td><td>FIN</td></tr>
<tr><td>INVINICIAL</td><td></td><td></td><td></td><td></td></tr>
<tr><td>Inventario inicial del producto(i)</td><td></td><td></td><td></td><td></td></tr>
<tr><td>LEADTIME</td><td></td><td></td><td></td><td></td></tr>
<tr><td>Lead Time del producto(i) **</td><td></td><td></td><td colspan="2" rowspan="2">Palabras reservadas</td></tr>
<tr><td>TIEMPOCICLO</td><td></td><td></td></tr>
<tr><td>Tiempos de ciclo del producto(i)</td><td></td><td></td><td colspan="2" rowspan="13">Datos a ingresar del modelo

*El producto inicial tiene ID = 0

** El modelo propuesto tiene al lead time como una variable no crítica

*** La capacidad depende si se selecciona un modelo capacitado</td></tr>
<tr><td>CAPACIDAD</td><td></td><td></td></tr>
<tr><td>Capacidad del producto(i) ***</td><td></td><td></td></tr>
<tr><td>PERIODOS</td><td></td><td></td></tr>
<tr><td>Número de periodos del sistema</td><td></td><td></td></tr>
<tr><td>ALMACENAMIENTO</td><td></td><td></td></tr>
<tr><td>Costo de almacenamiento del producto(i)</td><td></td><td></td></tr>
<tr><td>SETUP</td><td></td><td></td></tr>
<tr><td>Costo de set up del producto(i)</td><td></td><td></td></tr>
<tr><td>DEMANDA</td><td></td><td></td></tr>
<tr><td>Demanda del producto principal en el número de periodos</td><td></td><td></td><td colspan="2" rowspan="2">Opcional, dependiente al numero de componentes de un producto</td></tr>
<tr><td>FIN</td><td></td><td></td></tr>
</table>

Fuente: Elaboración propia.

Anexo 16. Manuel de usuario de PSOLib.

MANUAL DE USUARIO

PSOLIB

HECHO POR:

JAIME MANUEL PADRÓN CANO

CONTENIDO

VENTANA PRINCIPAL.

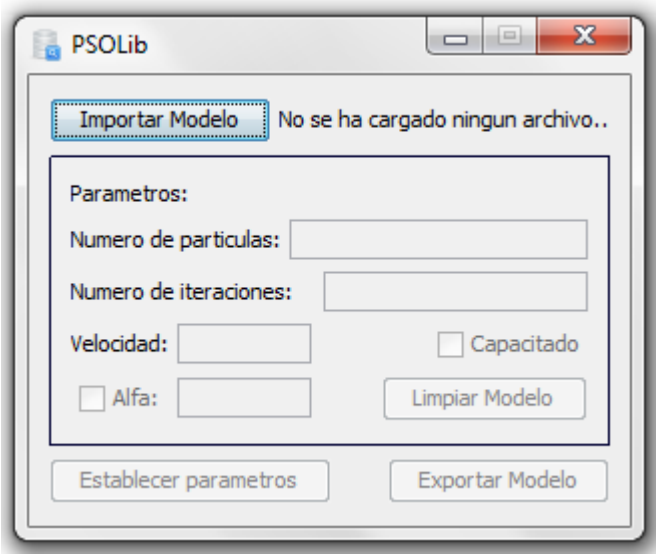


Figura 1

La ventana principal de PSOLib se presenta en la figura 1, cuenta con un botón para importar un modelo a partir de un archivo Excel (*.xls) definido previamente con el formato presentado en el anexo 15.

IMPORTAR MODELO

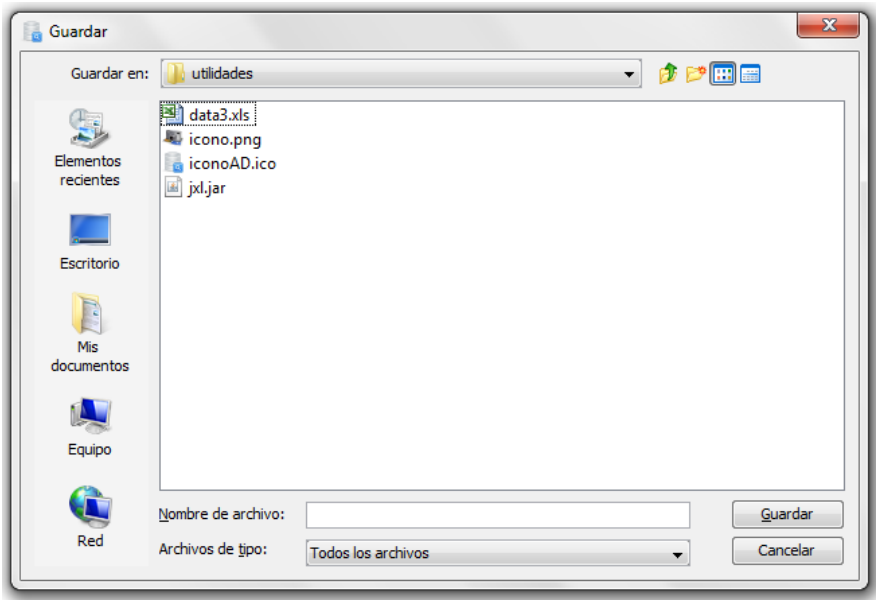


Figura 2.

Se selecciona el archivo con la información del modelo a partir del selector de archivos para cargar al modelo.

MODIFICAR PARÁMETROS

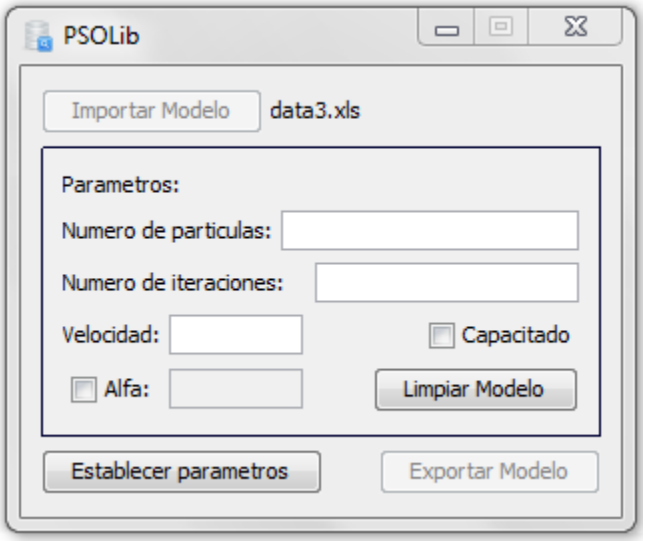


Figura 3.

Luego de importar el modelo se habilita la pestaña de los parámetros, se llena la información correspondiente al modelo (Número de partículas, número de iteraciones, velocidad) y se establece si el modelo es capacitado o no.

El valor de alfa es 0.5 por defecto a menos que no se llena el campo.

EXCEPCIONES

Las excepciones que se presentan en la interfaz de la librería para la solución de modelos corresponden a la gestión de ficheros:

- Error en la extensión del fichero
- Error al cargar el archivo
- Error al escribir en el archivo

Excepciones en los parámetros:

- Numero de partículas está en el rango de 0 a 500.
- Numero de iteraciones está en el rango de 0 a 1000.
- La velocidad está en el rango de 0,5 y 10.
- El valor de alfa está en el rango de 0,1 y 1.

EXPORTAR MODELO

Luego de establecer los parámetros del algoritmo se establecen los parámetros (Botón establecer parámetros) si no hay ninguna excepción el botón exportar modelo se habilitará y luego que se presione, éste preguntará si desea exportar, si la decisión es SI (Ver figura 4), ejecutará el algoritmo establecido según los parámetros y exportará el modelo al archivo seleccionado inicialmente.

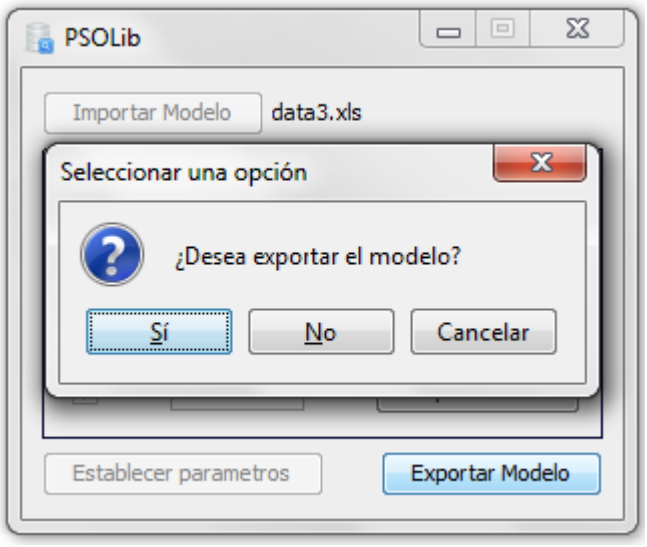
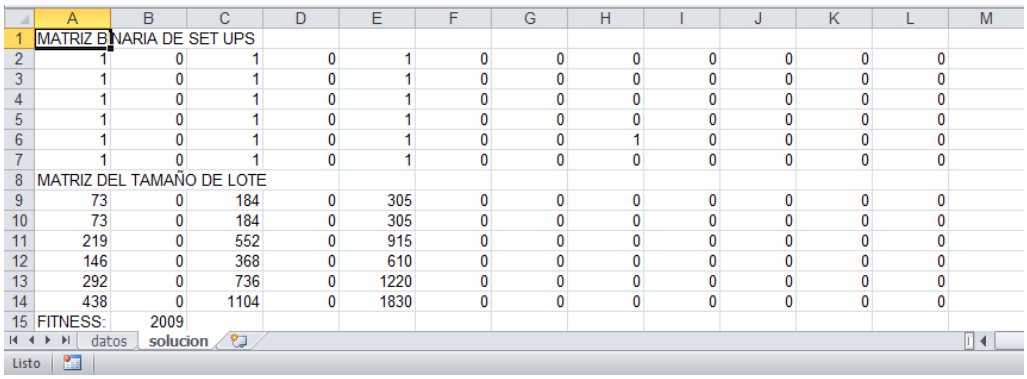


Figura 4.

Al final se reiniciarán los campos de la librería cuando el porcentaje haya llegado a 100%.

OBSERVAR SOLUCIONES

Luego de haber cargado la respuesta al archivo, se puede abrir el fichero y encontrar la solución en la hoja 2 correspondiente al tamaño de lote, la matriz binaria de los set ups y el valor del costo del modelo. (Fitness) Ver figura 5.



	A	B	C	D	E	F	G	H	I	J	K	L	M
1	MATRIZ BINARIA DE SET UPS												
2	1	0	1	0	1	0	0	0	0	0	0	0	
3	1	0	1	0	1	0	0	0	0	0	0	0	
4	1	0	1	0	1	0	0	0	0	0	0	0	
5	1	0	1	0	1	0	0	0	0	0	0	0	
6	1	0	1	0	1	0	0	1	0	0	0	0	
7	1	0	1	0	1	0	0	0	0	0	0	0	
8	MATRIZ DEL TAMAÑO DE LOTE												
9	73	0	184	0	305	0	0	0	0	0	0	0	
10	73	0	184	0	305	0	0	0	0	0	0	0	
11	219	0	552	0	915	0	0	0	0	0	0	0	
12	146	0	368	0	610	0	0	0	0	0	0	0	
13	292	0	736	0	1220	0	0	0	0	0	0	0	
14	438	0	1104	0	1830	0	0	0	0	0	0	0	
15	FITNESS:	2009											

datos | solucion

Listo

Figura 5.