

Modelos de programación Dinamica para el Diseño de un Simulador en Java, con Demanda Variable

Carlos Jairo Carvajal Trujillo

Universidad Cooperativa de Colombia, Cali, Valle, Colombia, carlos.carvajal@campusucc.edu.co

Ivan Andres Ordoñez Castaño

Universidad Cooperativa de Colombia, Cali, Valle, Colombia, ivan.ordóñez@campusucc.edu.co

Dario Quiroga Parra

Universidad Cooperativa de Colombia, Cali, Valle, Colombia, dario.quirogap@campusucc.edu.co

ABSTRACT

In every company should calculate inventories from a demand , in order to provide a client or business processes in time . Inventory carrying costs can represent 25% of the assets. In many cases you have the problem of having inventories above demand incurring overruns or below , leaving him with a bad client. When you go to do the calculations , you must use many variables and constraints and management becomes inextricable whole process and can make mistakes. To help solve this problem a simulator is designed in java in order to calculate inventory costs with all its variables and constraints from a lawsuit. As scientific methodology applied to engineering method was used. Mathematical models were designed, tested and then proceeded to design software using the paradigm of software development and object-oriented unified modeling language . Results : mathematical and computational models of dynamic programming, java program , artifacts and diagrams in UML and user manuals

Keywords: Inventories, demand, mathematical models, dynamic programming, object-oriented software.

RESUMEN

En toda empresa se deben calcular los inventarios a partir de una demanda, con el fin de proveer a un cliente o los procesos de la empresa justo a tiempo. Los costos de llevar inventarios pueden representar el 25% de los activos. En muchos casos se tiene el problema de tener inventarios por encima de la demanda incurriendo en sobrecostos o por debajo, quedándole mal a un cliente. Cuando se van a realizar los cálculos, se deben utilizar muchas variables y restricciones y el manejo vuelve inextricable todo el proceso y se pueden cometer errores. Para ayudar a resolver este problema se diseñó un simulador en java con el objetivo de calcular los costos de inventarios con todas sus variables y restricciones, a partir de una demanda. Dado que la investigación tiene un alto componente matemático, como metodología se usó el método científico aplicado metaheurística a la ingeniería. Se diseñaron los modelos matemáticos, se probaron y posteriormente se procedió a hacer el diseño del software usando el paradigma de desarrollo de software orientado a objetos y el lenguaje de modelamiento unificado. Resultados: los modelos matemáticos y computacionales de programación dinámica, el programa en java, los artefactos y diagramas en UML y los manuales de usuario.

Palabras claves: Inventarios, demanda, modelos matemáticos, programación dinámica, software orientado a objetos.

1. INTRODUCCIÓN

En las empresas de bienes o servicios es necesario mantener inventarios, para el movimiento de materia prima y/o producto terminado, los cuales representan más del 25% de los costos en los activos. Para hacer una apropiada gestión de inventarios, a partir de una demanda determinística, es necesario calcular los costos inherentes al proceso como almacenamiento, transporte, administrativo y de ordenamiento.

Para hacer los cálculos se desarrollan modelos matemáticos de programación dinámica determinística en el contexto de la Investigación de operaciones. El problema en inventarios se presenta cuando, en razón del volumen de datos y la complejidad de los modelos matemáticos, las empresas deciden no hacer los cálculos de inventarios y de pedidos. En este caso, las empresas incurren en problemas de adquirir por encima de la demanda o por debajo, afectando la relación con los clientes y los costos de la empresa.

Dada la dificultad del manejo de inventarios, cuando las empresas deciden comprar un software, este resulta costoso y difícil de usar. El presente trabajo busca ayudar a resolver dicho problema mediante el diseño un producto de software en java, con el objetivo de realizar los cálculos de inventarios con las variables y restricciones pertinentes, a partir de la demanda en n periodos y además que sea muy intuitivo y funcional.

En el desarrollo del proyecto se usó el método científico (Gauch 2003) aplicado a la ingeniería (KLEB 2006) con la metaheurística de (GUNTHER 2010), (Fink & Stefan 1999). En la primera fase se planteó el problema con todas sus variables y restricciones, posteriormente se efectuó una búsqueda bibliográfica en donde se investigaron los modelos de inventarios y se diseñaron los modelos matemáticos con todas las operaciones, variables y restricciones; cuando se construyeron los modelos matemáticos se desarrollaron los algoritmos recursivos en programación dinámica y se probaron con modelos típicos de inventarios determinísticos. En la Segunda fase se procedió a la construcción del software en Java, para esto, se usó la metodología para el desarrollo de software orientada a objetos (DSOO) (PRESSMAN 2010), con el proceso racional unificado (RUP) (LARMAN 2002) con sus diferentes fases y artefactos, usando para el modelamiento el lenguaje de modelamiento unificado (UML).

El documento se inicia con un resumen del estudio, la introducción, le siguen en su orden la metodología utilizada, la literatura básica del tema incluyendo el modelo matemático aplicado, a la que prosiguen los cálculos matemático en sus etapas cuatro, tres, dos y uno en su orden; finalmente se presentan las conclusiones, el listado de referencias utilizado y los anexos.

El trabajo es el producto de un proceso de investigación, y presenta como resultado tanto los modelos matemáticos de inventarios determinísticos, de programación dinámica; así como el programa en java, los artefactos propios del DSOO y el manual del usuario. Esta ponencia es un producto del proyecto de investigación: Modelo para calcular el beneficio económico de las actividades productivas informales como base de la planeación, mediante un producto de software; que se está llevando a cabo entre las facultad de Ingeniería y la facultad de Ciencias Administrativas Económicas y Contables; y los grupos de investigación GIHAPED Y OIKOS, de la Universidad Cooperativa de Colombia, sede Cali.

2. METODOLOGÍA

La ejecución del proyecto se dividió en dos fases. En la primera fase se planteó el problema con todas sus variable y restricciones, posteriormente se efectuó una búsqueda bibliográfica en donde se investigaron los modelos de inventarios y se diseñaron los modelos matemáticos con todas las operaciones, variables y restricciones. Así mismo, cuando se construyeron los modelos matemáticos se desarrollaron los algoritmos recursivos aplicando conceptos de programación dinámica. A medida que se desarrollaban los algoritmos, se fueron probando con modelos típicos de inventarios

Una vez concluida la primera fase, se procedió a la construcción del software en Java. Para lo anterior se utilizó la metodología para el desarrollo de software orientada a objetos con el RUP proceso racional unificado) en sus diferentes fases y artefactos usando UML (lenguaje de modelamiento unificado) para el modelado (planos).

Dado que, la investigación tiene un alto componente matemático, la metodología RUP se adaptó a la metaheurística siguiente:

Modelado del negocio: Planteamiento del problema, en los términos de la primera fase con los objetivos del proyecto de investigación, los recursos y los resultados esperados.

Requisitos: Se identificaron las tareas y se definieron los responsables teniendo en cuenta los recursos disponibles, se hizo el análisis de los requerimientos funcionales y no funcionales, a partir de los modelos matemáticos se diseñando los casos de uso, los diagramas de clase y se diseñaron los prototipos de la interfaces.

Diseño: A partir de los requerimientos y los casos de uso se diseñó el diagrama de clases y los diagramas de secuencia, y también se hizo el modelado por componentes.

Implementación: Teniendo en cuenta los diagramas anteriores y los modelos matemáticos construidos en la primera fase, se incorporaron las clases especializadas de los modelos matemáticos, se diseñó la base de datos,

se diseñaron las interfaces y se hizo el modelo de las capas. Posteriormente se hizo la implementación en java y la base de datos en SQL. **Despliegue:** Se realizó todo lo concerniente a las pruebas de caja blanca y caja negra (el diseño de las pruebas se hizo tomando el principio de las máquinas de estado finito), se diseñó el ejecutable y se escribió el manual del usuario.

2.1. LA TEORÍA DE INVENTARIOS DETERMINÍSTICOS

La inversión en inventarios puede representar más del 25%, de los activos (VOLLMAN 2000). Inventario es todo activo que se encuentra almacenado y disponible para cuando sea requerido por un cliente. También se puede decir que inventario(CHASE 2004) son las existencias de cualquier artículo o recurso utilizado en una organización. Un sistema de Inventarios es la serie de políticas y controles que monitorean los niveles de inventarios y determinan los niveles que se deben mantener, el momento en que la existencias de deben reponer, y el tamaño que deben tener los pedidos. El solo hecho de mantener o no un inventario generan unos costos que son(WINSTON 2008):

Costos de pedido o alistamiento

Son todos los costos asociados con el re abastecimiento del inventario, estos costos varían con el número de pedido, y son:

- Los costos de requisición.
- Los costos de emitir y seguir la orden de compra.
- Los costos de inspección al recibir y colocar los artículos en inventario y transporte

Costos de mantenimiento

Estos son los costos asociados con mantener una cantidad disponible y varia con el nivel y periodo de tiempo que se mantenga y son:

- Costos de oportunidad de la inversión.
- Costos de almacenamiento (alquiler, calefacción, refrigeración, vigilancia, etc.).
- Deterioro del producto u obsolescencia.
- Impuestos depreciación y seguro.
- Costos de transporte

Costos de quedarse corto (agotado)

Estos son los costos de penalización en que se incurre cuando quedan sin la mercancía. Y los costos por pedido del cliente.

Teniendo en cuenta el análisis de costos surgen las siguientes preguntas:

- ¿Cuánto comprar?
- ¿Cuándo comprar?

Para ayudar a responder las preguntas anteriores se debe aplicar el concepto de demanda determinística que se da, cuando se sabe con seguridad, cuál va a ser el comportamiento de la demanda (etapa) en el tiempo. Como problema de estudio, y para diseñar el software, se escogió el ejercicio de inventarios(Taha 2007): como: se muestra en la Tabla 1 y la Tabla 2, con sus respectivas variables, datos y restricciones. El comportamiento de las Etapas vs Demanda se puede ver, en la Figura 2. Para hallar la solución óptima se diseñó un modelo matemático utilizando grafos y el algoritmo de Dijkstra(Rosen 2012) conocido también como algoritmo del viajante (Winston 2008), con algunas variantes; que se basa en calcular los caminos para cada uno de los nodos, del primero al último y después, del último al primero (en forma recursiva), escogiendo la menor ruta del ultimo al primero.

Las variables con las que se trabaja el modelo son: inventario máximo, compra máxima, costo administrativo, costo unitario y costo de almacenamiento, ver Tabla 3. Para ilustrar se explicaran los cálculos, pasos por paso (Etapa por etapa con cada uno de sus estados) usando un modelo de grafos, ver Figura 1. Las etapas se refieren a cada uno de los meses (rectángulos en la parte superior horizontal – de la 1 a la.4) y cada etapa puede tener varios inventarios finales, que son los estados (óvalos, debajo de cada etapa). El inventario

máximo es el estado que puede tomar cada una de las etapas (rectángulos verticales), el inventario inicial (invi) corresponde a la arista que le llega. La solución se basa en calcular el mejor estado (ovalo) en cada etapa teniendo en cuenta el mejor estado etapa posterior, y trazar la ruta optima entre el mejores estados de cada etapa.

Para calcular el mejor inventario de la **etapa 4**, se deben realizar los cálculos de cada uno de sus inventarios finales (invf – estados posibles), a cada estado se le deben realizar los cálculos en donde los inventarios iniciales (invi) y costo acumulado (cacu), son las aristas que le llegan, de los estados de la etapa 3. Los óptimos están sombreados.

Para calcular el mejor inventario de la **etapa 3**, se deben realizar los Cálculos de cada uno de sus inventarios finales (invf – estados posibles), a cada estado se le deben realizar los Cálculos en donde los inventarios iniciales (invi) y costo acumulado (cacu), son las aristas que le llegan, de los estados de la etapa 2. Los óptimos están sombreados.

Para calcular el mejor inventario de la etapa 2, se deben realizar los Cálculos de cada uno de sus inventarios finales (invf – estados posibles), a cada estado se le deben realizar los Cálculos en donde los inventarios iniciales (invi) y costo acumulado (cacu), son las aristas que le llegan, de los estados de la etapa 1. Los óptimos están sombreados.

Para calcular el mejor inventario de la **etapa 1**, se deben realizar los cálculos de cada uno de sus inventarios finales (invf – estados posibles), a cada estado se le deben realizar los cálculos en donde los inventarios iniciales (invi) y costo acumulado (cacu), son cero, dado que es la etapa inicial.

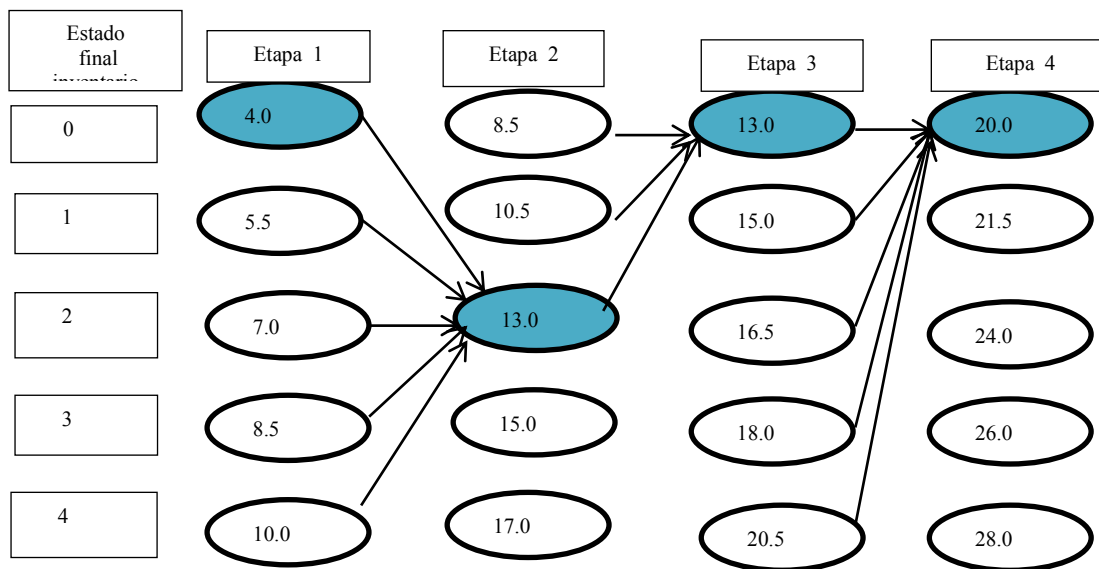


Figura 1- Etapas con sus diferentes estados, costos e inventarios finales

Fuente: los autores tomando como datos de (Taha 2007)

2.2. EL MODELO MATEAMÁTICO

La función matemática recursiva general (Función 1) que resuelve el inventario anterior es la sumatoria de los optimos calculados en las etapas anteriores (Función 2), (Función 3), (Función 4) y (Función 5):

Función 1- Cálculos de los inventarios recursivamente

$$funcion(variables, etapa) = variables + \sum_{0}^{etapa=4} funcion(variables, etapa-1)$$

Como en cada etapa los estados iniciales y finales cambian (inventarios), se tiene que cumplir las restricciones siguientes:

$$\{$$

$$inv_inicial + compra - demanda \leq capacidad_inventario$$

$$inv_inicial + compra - demanda = inv_final$$

$$inv_final \leq capacidad_inventario$$

$$\}$$

Función 2- Cálculos recursivos para la etapa 4

$$funcion(variables,4) = variables + \sum_1^4 funcion(variables,4-1)$$

Ver cálculos etapa 4

Función 3- Cálculos recursivos para la etapa 3

$$funcion(variables,3) = variables + \sum_1^3 funcion(variables,3-1)$$

Ver cálculos etapa 3

Función 4- Cálculos recursivos para la etapa 2

$$funcion(variables,2) = variables + \sum_1^2 funcion(variables,2-1)$$

Ver cálculos etapa 2

Función 5- Cálculos recursivos para la etapa 1

$$funcion(variables,0) = 0$$

Ver cálculos etapa 1

Para efectuar los Cálculos se tiene en cuenta la función recursiva general y se diseña la siguiente: A continuación se representan las funciones recursivas, con sus variables

$$funcion(com,cuni,invi,dem,calm,cadm,cacu,eta, invf) = com*cuni + (com + invi - dem)*calm + cadm + \sum funcion(variables,eta-1, invf)$$

Se deben cumplir las restricciones siguientes:

$$\{$$

$$invi + com - dem = inf$$

$$invf \leq imax$$

$$com, invi, invf, dem \geq 0$$

$$\}$$

Función para la etapa 4 (eta = 4)

$$funcion(com,cuni,invi,dem,calm,cadm,cacu,4, invf) = com*cuni + (com + invi - dem)*calm + cadm + \sum funcion(variables,4-1, invf)$$

Por cuestiones de espacio en el símbolo sumatoria de las funciones recursivas solo se denotan las variables, etapa (eta) e inventario final (invf). Cuando la función hace el llamado recursivo, el estado se toma como inventario final (, invf). Para visualizar los Cálculos de las etapas con facilidad se utiliza una tabla en donde

se tabulan las etapas con. Los Cálculos de cada estado y sus variables, ver (Tabla 4) El resultado se toma como costo estado (cose). A continuación se muestran los cálculos para cada etapa:

Cálculos para la etapa 4

El estado final inventario ($invf = 0$), tiene las entradas, $invi: 0 \dots 4$ (inventarios acumulados y sus costos) (Tabla 4), se efectúan los cálculos con cada entrada y se escoge el menor resultado, como el óptimo (fila y ovalo sombreada), (Función 6).

Función 6 - Cálculos recursivos de la etapa 4 (eta 4)

$$\text{función}(\text{com}, \text{cuni}, \text{invi}, \text{dem}, \text{calm}, \text{cadm}, \text{cacu}, 4, \text{invf}) = \text{com} * \text{cuni} + (\text{com} + \text{invi} - \text{dem}) * \text{calm} + \text{cadm} + \sum \text{función}(\text{variables}, 4-1, \text{invf})$$

Se deben cumplir las restricciones siguientes:

$$\left\{ \begin{array}{l} \text{invi} + \text{com} - \text{dem} = \text{inf} \\ \text{invf} \leq \text{imax} \\ \text{com}, \text{invi}, \text{invf}, \text{dem} \geq 0 \end{array} \right\}$$

Cálculos para la etapa 3

El estado final inventario ($invf = 0$), tiene las entradas $invi: 0 \dots 2$ (inventarios acumulados y sus costos) Tabla 4. Se efectúan los cálculos con cada entrada y se escoge el menor resultado, como el óptimo (fila y ovalo sombreada), (Función 7).

Función 7 - Cálculos recursivos de la etapa 3 (eta 3)

$$\text{función}(\text{com}, \text{cuni}, \text{invi}, \text{dem}, \text{calm}, \text{cadm}, \text{cacu}, 3, \text{invf}) = \text{com} * \text{cuni} + (\text{com} + \text{invi} - \text{dem}) * \text{calm} + \text{cadm} + \sum \text{función}(\text{variables}, 3-1, \text{invf})$$

Se deben cumplir las restricciones siguientes:

$$\left\{ \begin{array}{l} \text{invi} + \text{com} - \text{dem} = \text{inf} \\ \text{invf} \leq \text{imax} \\ \text{com}, \text{invi}, \text{invf}, \text{dem} \geq 0 \end{array} \right\}$$

El cálculo óptimo, para el estado final inventario 0, de la etapa 3, es 13.0

Cálculos para la etapa 2

El estado final inventario ($invf = 2$), tiene las entradas $invi: 0 \dots 4$ (inventarios acumulados y sus costos) 8Tabla 4), se efectúan los cálculos con cada entrada y se escoge el menor resultado, como el óptimo (fila y ovalo sombreada), (Función 8).

Función 8 Cálculos recursivos de la etapa 2 (eta 2)

$$\text{función}(\text{com}, \text{cuni}, \text{invi}, \text{dem}, \text{calm}, \text{cadm}, \text{cacu}, 2, \text{invf}) = \text{com} * \text{cuni} + (\text{com} + \text{invi} - \text{dem}) * \text{calm} + \text{cadm} + \sum \text{función}(\text{variables}, 2-1, \text{invf})$$

Se deben cumplir las restricciones siguientes:

$$\left\{ \begin{array}{l} \text{invi} + \text{com} - \text{dem} = \text{inf} \\ \text{invf} \leq \text{imax} \\ \text{com}, \text{invi}, \text{invf}, \text{dem} \geq 0 \end{array} \right\}$$

El cálculo óptimo, para el estado final inventario 2, de la etapa 4, es 13.0

Cálculos para la etapa 1

La etapa 1, tiene un inventario inicial de cero, que son los estados iniciales y se efectúan los cálculos matemáticos para deducir los inventarios finales óptimos (Tabla 4), Función 9

Función 9 Cálculos recursivos de la etapa 1 (eta 1)

función(com,cuni,invi,dem,calm,cadm,cacu,1, invf) =
com*cuni +.(com + invf -dem)*calm + cadm + \sum funcion(variables,1-1, invf)
..... retorna valor = 0

Se deben cumplir las restricciones siguientes:

```
{  
  invf=0;  
  invf + com - dem = inf  
  invf <= imax  
  com, invf, dem >= 0  
}
```

El cálculo óptimo, para el estado final inventario 0, de la etapa 1, es 4.0

3. CONCLUSIONES

Los modelos matemáticos para calcular los costos de producción con demandas determinísticas tienen características recursivas dado que se pueden calcular a partir de los cálculos generales y llamar a los iniciales de cada ítem. Cuando se va a desarrollar un software para uso de costos industriales se debe usar el RUP, ya que permite ir construyendo el software a medida que se van desarrollando los modelos matemáticos, con sus componentes y se van probando así mismo, dado que, es una metodología iterativa y escalar.

Los sistemas de manufactura por órdenes de producción tienen características de sistemas discretos ya que ocurren en unos tiempos (etapas) determinados, por lo tanto se le pueden aplicar algoritmos de programación dinámica para su planeación. Los procesos de producción tienen características de fenómenos recursivos ya su simulación y planeación se puede hacer desde la etapa final descomponiendo el costo total en sus variables e ir de la última etapa a la primera y de la primera a la última.

4. AGRADECIMIENTOS

Se agradece el apoyo brindado por las directivas de la Universidad Cooperativa de Colombia, sede Cali: Doctora Graciela Porras y al Doctor Diego Victoria, así como al Ingeniero Julian Sarria decano de la facultad de Ingeniería y a la Ingeniera Josefina de Llano directora de investigación de la facultad de Ingeniería

REFERENCIAS

- CHASE, R., 2004. *Administración de la producción y operaciones para una ventaja competitiva* 10th ed., México: McGraw-Hill.
- Fink, A. & Stefan, V., 1999. Generic metaheuristics application to industrial engineering problems. *Computers & Industrial Engineering*, 37(1-2), pp.281 – 284. Available at: <http://www.sciencedirect.com/science/article/pii/S0360835299000741>.
- Gauch, H., 2003. *Scientific Method in Practice* 1st ed., United Kingdom: Cambridge University Press.
- GUNTHER, Z., 2010. *Metaheuristic Search* 1st ed., New York: Springer.
- KLEB, B., 2006. *Computational Simulations and the Scientific Method*, Aerospace Research Central. Available at: <http://doi.aiaa.org/10.2514/1.12949>.
- LARMAN, C., 2002. *UML y patronos* 2nd ed., España: Prentice Hall.

PRESSMAN, R., 2010. *Ingeniería de software* 7th ed., Mexico: Mc Graw Hill.
Rosen, K., 2012. *Discrete Mathematics and Its Applications* 7th ed., United States: Mc Graw Hill.
Taha, A., 2007. *Operations Research* 8th ed., United State: Prentice Hall.
VOLLMAN, T., 2000. *Administración integral de la producción y de los inventarios* 2nd ed., Mexico: Limusa.
WINSTON, W., 2008. *Operations Research: Applications & Algorithms* 4th ed., United States: Thomson.

ANEXOS

Tabla 1- Variables y restricciones de los inventarios

| | | |
|----------------------|------|-----|
| Inventario máximo | imax | 4 |
| Compra máxima | cmax | 5 |
| Costo administrativo | cadm | 3 |
| Costo unitario | cuni | 1 |
| Costo almacenamiento | calm | 0.5 |
| Etapas | eta | 4 |
| Demanda | dem | |

Fuente: Los autores

Tabla 2- Demanda por etapas

| Eta- eta | Demanda- dem |
|-------------|-----------------|
| 1 | 1 |
| 2 | 3 |
| 3 | 2 |
| 4 | 4 |

Fuente: tomado de (Taha 2007)

Tabla 3 - Variables para los estados de las etapas

| | | |
|----------------------|------|---|
| Inventario máximo | imax | Es la capacidad que tiene la bodega |
| Compra máxima | cmax | Son los recursos disponibles para comprar |
| Costo administrativo | cadm | Es lo que vale producir una orden de compra |
| Costo unitario | cuni | El valor de cada unidad |
| Costo almacenamiento | calm | Es lo que vale tener una unidad en bodega |
| Etapas | eta | Son los mese en que se demandara y tendrá el producto en bodega |
| Demanda | dem | Cantidad que se va a vender |
| Inventario inicial | invi | Es la existencia de productos al iniciar la etapa |
| Inventario final | invf | Es la existencia de productos al final la etapa |
| compra | com | Es la cantidad que se adquiere en cada estado de la etapa |
| Costo acumulado | cacu | Costo acumulado estado óptimo de la etapa anterior |
| Costo estado | cose | Es el Cálculos del costo de cada estado en una etapa |

Fuente: Los autores

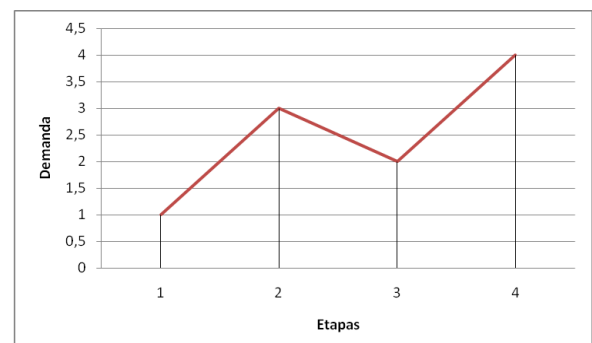


Figura 2- Demanda por etapas

Fuente: tomado de (Taha 2007)

Tabla 4- Cálculos de todas las etapas con sus estados

| Estadio inventario inicial - invi() | | | | | | | | | | | | | | | | |
|-------------------------------------|-----|------|----------|---|----------|---|----------|---|----------|---|----------|---|-------------|------|-----|-------------|
| Etapa | dem | invf | invi(0) | | invi(1) | | invi(2) | | invi(3) | | invi(4) | | cose optimo | invi | com | Costo local |
| | | | cose com | | cose com | | cose com | | cose com | | cose com | | | | | |
| 1.0 | 1.0 | 0.0 | 4.0 | 1 | | | | | | | | | 4.0 | 0.0 | 1.0 | 4.0 |
| 1.0 | 1.0 | 1.0 | 5.5 | 2 | | | | | | | | | 5.5 | 0.0 | 2.0 | 5.5 |
| 1.0 | 1.0 | 2.0 | 7.0 | 3 | | | | | | | | | 7.0 | 0.0 | 3.0 | 7.0 |
| 1.0 | 1.0 | 3.0 | 8.5 | 4 | | | | | | | | | 8.5 | 0.0 | 4.0 | 8.5 |
| 1.0 | 1.0 | 4.0 | 10.0 | 5 | | | | | | | | | 10.0 | 0.0 | 5.0 | 10.0 |
| Etapa | dem | invf | invi(0) | | invi(1) | | invi(2) | | invi(3) | | invi(4) | | cose optimo | invi | com | Costo local |
| | | | cose com | | cose com | | cose com | | cose com | | cose com | | | | | |
| 2.0 | 3.0 | 0.0 | 10.0 | 3 | 10.5 | 2 | 11.0 | 1 | 8.5 | 0 | | | 8.5 | 3.0 | 0.0 | 0.0 |
| 2.0 | 3.0 | 1.0 | 11.5 | 4 | 12.0 | 3 | 12.5 | 2 | 13.0 | 1 | 10.5 | 0 | 10.5 | 4.0 | 0.0 | 0.5 |
| 2.0 | 3.0 | 2.0 | 13.0 | 5 | 13.5 | 4 | 14.0 | 3 | 14.5 | 2 | 15.0 | 1 | 13.0 | 0.0 | 5.0 | 9.0 |
| 2.0 | 3.0 | 3.0 | | | 15.0 | 5 | 15.5 | 4 | 16.0 | 3 | 16.5 | 2 | 15.0 | 1.0 | 5.0 | 9.5 |
| 2.0 | 3.0 | 4.0 | | | | | 17.0 | 5 | 17.5 | 4 | 18.0 | 3 | 17.0 | 2.0 | 5.0 | 10.0 |
| Etapa | dem | invf | invi(0) | | invi(1) | | invi(2) | | invi(3) | | invi(4) | | cose optimo | invi | com | Costo local |
| | | | cose com | | cose com | | cose com | | cose com | | cose com | | | | | |
| 3.0 | 2.0 | 0.0 | 13.5 | 2 | 14.5 | 1 | 13.0 | 0 | | | | | 13.0 | 2.0 | 0.0 | 0.0 |
| 3.0 | 2.0 | 1.0 | 15.0 | 3 | 16.0 | 2 | 17.5 | 1 | 15.5 | 0 | | | 15.0 | 0.0 | 3.0 | 6.5 |
| 3.0 | 2.0 | 2.0 | 16.5 | 4 | 17.5 | 3 | 19.0 | 2 | 20.0 | 1 | 18.0 | | 16.5 | 0.0 | 4.0 | 8.0 |
| 3.0 | 2.0 | 3.0 | 18.0 | 5 | 19.0 | 4 | 20.5 | 3 | 21.5 | 2 | 22.5 | | 18.0 | 0.0 | 5.0 | 9.5 |
| 3.0 | 2.0 | 4.0 | | | 20.5 | 5 | 22.0 | 4 | 23.0 | 3 | 24.0 | | 20.5 | 1.0 | 5.0 | 10.0 |
| Etapa | dem | invf | invi(0) | | invi(1) | | invi(2) | | invi(3) | | invi(4) | | cose optimo | invi | com | Costo local |
| | | | cose com | | cose com | | cose com | | cose com | | cose com | | | | | |
| 4.0 | 4.0 | 0.0 | 20.0 | 4 | 21.0 | 3 | 21.5 | 2 | 22.0 | 1 | 20.5 | 0 | 20.0 | 0.0 | 4.0 | 7.0 |
| 4.0 | 4.0 | 1.0 | 21.5 | 5 | 22.5 | 4 | 23.0 | 3 | 23.5 | 2 | 25.0 | 1 | 21.5 | 0.0 | 5.0 | 8.5 |
| 4.0 | 4.0 | 2.0 | | | 24.0 | 5 | 24.5 | 4 | 25.0 | 3 | 26.5 | 2 | 24.0 | 1.0 | 5.0 | 9.0 |
| 4.0 | 4.0 | 3.0 | | | | | 26.0 | 5 | 26.5 | 4 | 28.0 | 3 | 26.0 | 2.0 | 5.0 | 9.5 |
| 4.0 | 4.0 | 4.0 | | | | | | | 28.0 | 5 | 29.5 | 4 | 28.0 | 3.0 | 5.0 | 10.0 |

Fuente: Los autores