Universidad de Cienfuegos “Carlos Rafael Rodríguez”

Facultad de Ingeniería

Carrera Ingeniería Informática

**SISTEMA INFORMÁTICO PARA LA OPTIMIZACIÓN DE PROBLEMAS DE CORTE EN DOS DIMENSIONES.**

**Trabajo de diploma para optar por el título de Ingeniero Informático**

**Autor:**

**Javier Leiva Cáceres**

**Tutor:**

**Juan Felipe Medina Mendieta. Universidad de Cienfuegos.**

**Cienfuegos, Cuba**

**Curso 2017 - 2018**

**Declaración de autoría**

Declaro que soy el único autor de este trabajo y autorizo al Departamento de Informática de la Facultad de Informática en la Universidad de Cienfuegos “Carlos Rafael Rodríguez”, para que hagan el uso que estimen pertinente con el trabajo de diploma.

Para que así conste firmo la presente a los \_\_\_\_ días del mes de \_\_\_ del \_\_\_.

|  |  |
| --- | --- |
| \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  Nombre completo del primer autor | (Si procede)  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  Nombre completo del segundo autor |
|  |  |
| \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  Nombre completo del primer tutor | (Si procede)  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  Nombre completo del segundo tutor |

Los abajo firmantes certificamos que el presente trabajo ha sido revisado según acuerdo de la dirección de nuestro centro y el mismo cumple los requisitos que debe tener un trabajo de esta envergadura referente a la temática señalada.

--------------- ---------------------

Firma Tutor Firma Tutor

---------------- ---------------------

Firma ICT Firma Vicedecano

**Opinión del tutor**

Título: SISTEMA INFORMÁTICO PARA LA OPTIMIZACIÓN DE PROBLEMAS DE CORTE EN DOS DIMENSIONES

Autor: Javier Leiva Cáceres

El corte de materiales en dos dimensiones constituye un problema de gran magnitud y conocida dificultad en su tratamiento. Abordado desde el punto de vista de la programación lineal en enteros presenta tres grandes dificultades: generación de patrones de corte para la obtención del modelo matemático, procesamiento computacional del modelo e interpretación de la solución obtenida. Por ello todo aporte realizado en ese sentido es de mucho valor para el desarrollo de esta teoría lo cual expresa la importancia del problema abordado en el presente trabajo.

El tutor considera que durante la ejecución de este Trabajo de Diploma el diplomante mostró cualidades y llegó a resultados algunos de los cuales se destacan a continuación:

Este trabajo, consecuencia de un proceso de investigación, y continuación de parte del mismo se visualiza en un sistema informático para la optimización de problemas de corte en dos dimensiones, lo cual repercute de manera significativa en la organización, control y ahorro desde el punto de vista económico:

En el mismo se destacan tres resultados fundamentales:

Primero: Creación de un algoritmo para la generación de patrones de corte en dos dimensiones para el caso de corte regulares de forma rectangular.

Segundo: Desarrollo de un software que automatiza el algoritmo de generación de patrones de corte que se integre con el anterior existente y muestre los resultados de forma entendible por el usuario.

Tercero: Documentación de la investigación.

La documentación presentada, está correctamente estructurada en 3 capítulos, incluyendo además introducción, conclusiones, recomendaciones, referencias bibliográficas, bibliografías y anexos. Presenta una redacción coherente, clara, precisa y de rigor técnico.

Se debe destacar que en el transcurso de la investigación el diplomante, Javier, presentó muy alta responsabilidad e independencia tanto en el estudio de la bibliografía, el desarrollo del algoritmo de generación de patrones de corte, del software, así como de la documentación presentada, cumpliendo con la planeación de tiempo prevista en cada período de la investigación. Resaltar que tuvo que enfrentarse a problemas no estudiados en la carrera que requieren dominar técnicas y herramientas del campo de las matemáticas y de la informática, los cuáles solucionó satisfactoriamente fortaleciendo su formación. Destacar su muy alta capacidad de investigación para la asimilación de una teoría de alta complejidad a este nivel, como la que sustenta este Trabajo de Diploma. Demostró contar con conocimientos sólidos que puso en evidencia en la realización del proyecto. Asimiló cada propuesta hecha por el tutor y enriqueció con propias ideas y aportes la investigación.

Los resultados expresados en el Trabajo de Diploma demuestran la importancia en cuanto a organización, control y ahorro de recursos económicos si es aplicada esta investigación. Debido a este impacto económico el tutor considera que el trabajo de Diploma posee valor para ser publicado y/o participar en eventos y Forum.

Por todo lo anteriormente expresado el tutor considera que el estudiante está apto para ejercer como Ingeniero Informático; y propone que se le otorgue al Trabajo de Diploma la calificación de 5 puntos o de trabajo de excelencia.

Y para que así conste, se firma la presente a los \_\_ días del mes de \_\_\_\_\_ del año \_\_\_.

|  |  |
| --- | --- |
| \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  Juan Felipe Medina Mendieta  Máster y profesor asistente. |  |

Fecha: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Agradecimientos**

Agradezco a Dios en primer lugar, por permitirme llegar a este momento tan deseado por mí, por darme fuerzas y poder de discernimiento para lograr saltar y en ocasiones esquivar los obstáculos que se me presentaron a lo largo de mi vida universitaria; también por colocar en mi camino a mi tutor, el que me propuso este tema de investigación que encontré muy interesante y me dediqué a estudiar. Muchas gracias a Juan Felipe Medina por su tiempo dedicado a mí, su puntualidad y responsabilidad en las citas acordadas a pesar de sus responsabilidades como profesor y con su gran familia.

Muchas gracias a todos mis compañeros de aula por soportarme en estos cinco años, pero en especial a José Manuel y Rafa a los que consulté con mucha frecuencia cuando estaba desarrollando el software y siempre se mostraron disponibles.

Muchas gracias a cada uno de los profesores que contribuyeron a mi crecimiento espiritual y ayudaron al fortalecimiento de mis principios y convicciones.

Muchas gracias les quiero dar a mi madre, mi padre, mi abuela Amparo, mi abuelo “*El Gallego*” y mi tío Carlos, especialmente a mi tío, que contribuyó grandemente en mi formación y educación en los valores internacionalmente reconocidos y aceptados, pero lamentablemente tan ausentes en gran parte de nuestra sociedad.

Quiero agradecer de forma muy especial a uno de los más grandes regalos que me ha hecho Dios, mi esposa Diana, por ser tan comprensiva conmigo y por darme alientos y contagiarme con su espíritu optimista cuando en ocasiones le planteaba “*el tema que estoy investigando es muy complejo, no está a mi alcance, creo que voy a buscare otro tema*”.

**Resumen**

La presente investigación se titula “Sistema informático para la optimización de problemas de corte en dos dimensiones” tiene su génesis en la necesidad que existe en muchas industrias de hacer eficientes los procesos de corte de materiales y disminuir de forma considerable los desperdicios. Los procesos como los anteriormente descritos son conocidos comúnmente como problemas de corte de materiales en dos dimensiones.

Estos procesos que se han hecho habituales en muchas industrias pueden generar grandes desperdicios debido a las múltiples combinaciones de corte a realizar basado sólo en el conocimiento acumulado durante años por el operario que los lleva a cabo sin tener en cuanta ramas de la ciencia que pueden mejorar estos procesos como la utilización de métodos matemáticos para la reducción u optimización de los desperdicios.

El resultado más relevante que se puede apreciar en la investigación es la obtención del sistema informático “OPTIVACOMULTI”, el que es capaz de generar un gran número de variantes de corte primarias y de sustitución, y a partir de ellas llegar a la solución óptima que permitan reducir los gastos asociados a los desperdicios generados en el proceso de corte de materiales en dos dimensiones. También muestra la interpretación de los resultados de forma gráfica, de tal manera que sea de fácil comprensión por parte del usuario.

# Summary

The present research is entitled "Computer system for the optimization of cutting problems in two dimensions" has its genesis in the need that exists in many industries to make efficient material cutting processes and significantly reduce waste. Processes such as those described above are commonly known as two-dimensional material cutting problems.

These processes that have become common in many industries can generate great waste due to the multiple combinations of cuts to be made based only on the knowledge accumulated over the years by the operator that carries them out without taking into account branches of science that can be improved. These processes include the use of mathematical methods for the reduction or optimization of waste.

The most relevant result that can be seen in the research is the obtaining of the computer system "OPTIVACOMULTI", which is capable of generating a large number of primary and substitution cutting variants, and from them to reach the optimal solution that allows reduce the expenses associated with the waste generated in the process of cutting materials in two dimensions. It also shows the interpretation of the results graphically, in such a way that it is easily understood by the user.

**Índice**

[Summary I](#_Toc516470528)

[Introducción 2](#_Toc516470529)

[1 – Fundamentos teóricos 8](#_Toc516470530)

[1.1 – Introducción 8](#_Toc516470531)

[1.2 – Descripción del dominio del problema 8](#_Toc516470532)

[1.2.1 Alternativa de solución 10](#_Toc516470533)

[1.3 – Análisis teórico de los métodos de solución. 10](#_Toc516470534)

[1.3.1 – Investigación de operaciones. 10](#_Toc516470535)

[1.3.2 – Programación lineal en enteros. 11](#_Toc516470536)

[1.3.3 – Modelación matemática del problema. 12](#_Toc516470537)

[1.3.4 – Generación de variantes de corte en dos dimensiones. 13](#_Toc516470538)

[1.4 – Otras aplicaciones 20](#_Toc516470539)

[1.5 – Descripción de los sistemas existentes 21](#_Toc516470540)

[1.5.1 Software vinculados con el tema 21](#_Toc516470541)

[1.5.2 Optivacomulti. Solución propuesta. 22](#_Toc516470542)

[1.6 – Tendencias, metodologías y/o tecnologías actuales 22](#_Toc516470543)

[1.6.1 Fundamentos de la metodología utilizada 22](#_Toc516470544)

[1.6.1.1 Lenguaje de Modelamiento Unificado (UML) 22](#_Toc516470545)

[1.6.1.2 Proceso Unificado de Desarrollo (RUP) 23](#_Toc516470546)

[1.6.2 Fundamentación del lenguaje y software utilizado 24](#_Toc516470547)

[1.6.2.2 Lenguaje de programación Java 25](#_Toc516470548)

[1.6.2.3 NetBeans IDE 8.0 como herramienta para el desarrollo del software 26](#_Toc516470549)

[1.7 – Conclusiones 26](#_Toc516470550)

[2 – Análisis y diseño de la propuesta de solución 28](#_Toc516470551)

[2.1 – Introducción 28](#_Toc516470552)

[2.2 – Modelo del dominio 29](#_Toc516470553)

[2.2.1 – Definición de las entidades y los conceptos principales 29](#_Toc516470554)

[2.2.2 – Reglas del negocio a considerar 29](#_Toc516470555)

[2.2.3 – Representación del modelo del dominio 30](#_Toc516470556)

[2.3 – Requisitos 30](#_Toc516470557)

[2.3.1 – Descripción del sistema propuesto 30](#_Toc516470558)

[2.3.2 – Concepción general del sistema 30](#_Toc516470559)

[2.3.3 – Requerimientos funcionales 31](#_Toc516470560)

[2.3.4 – Requerimientos no funcionales 31](#_Toc516470561)

[2.3.5 – Modelo de casos de uso del sistema 32](#_Toc516470562)

[2.3.6 – Actores del sistema 32](#_Toc516470563)

[2.3.7 – Diagramas de casos de uso del sistema 32](#_Toc516470564)

[2.3.8 – Descripción de los casos de uso del sistema 33](#_Toc516470565)

[2.4 – Construcción de la solución propuesta 34](#_Toc516470566)

[2.4.1 – Diagrama de clases del diseño 34](#_Toc516470567)

[2.4.2 – Diseño de archivo 36](#_Toc516470568)

[2.4.3 – Diagrama de archivo 36](#_Toc516470569)

[2.4.4 – Estructura del archivo 36](#_Toc516470570)

[2.4.5 – Diagrama de implementación 38](#_Toc516470571)

[2.4.6 – Principios de diseño 38](#_Toc516470572)

[2.4.7 – Estándares en la interfaz de la aplicación 39](#_Toc516470573)

[2.4.8 – Tratamiento de errores 39](#_Toc516470574)

[2.5 – Conclusiones 39](#_Toc516470575)

[3 – Estudio de factibilidad y validación de la solución 40](#_Toc516470576)

[3.1 – Introducción 40](#_Toc516470577)

[3.2 – Estudio de factibilidad 40](#_Toc516470578)

[3.2.1 – Planificación por casos de usos 40](#_Toc516470579)

[3.2.1.1 – Cálculo de Puntos de Caso de Uso sin ajustar 40](#_Toc516470580)

[3.2.1.2 – Cálculo de Puntos de Caso de Uso ajustados 41](#_Toc516470581)

[3.2.2 – Determinación de los costos 44](#_Toc516470582)

[3.2.3– Beneficios tangibles e intangibles 45](#_Toc516470583)

[3.2.4 – Análisis de costos y beneficios 46](#_Toc516470584)

[3.3 – Validación de la solución 46](#_Toc516470585)

[3.4 – Conclusiones 49](#_Toc516470586)

[Conclusiones 50](#_Toc516470587)

[Recomendaciones 51](#_Toc516470588)

[Referencias bibliográficas 52](#_Toc516470589)

[Anexos 55](#_Toc516470590)

**Índice de tablas**

[Tabla 1. Definición de actores del sistema a automatizar 32](#_Toc516344393)

[Tabla 2. Descripción del caso de uso de sistema < Introducir plan de corte > 33](#_Toc516344394)

[Tabla 3. Descripción del caso de uso de sistema < Generar variante de corte > 34](#_Toc516344395)

[Tabla 4. Descripción del caso de uso de sistema < Mostrar interpretación > 34](#_Toc516344396)

[Tabla 5. Factor de Peso de los Actores sin ajustar 41](#_Toc516344397)

[Tabla 6. Factor de Peso de los Casos de Uso sin ajustar 41](#_Toc516344398)

[Tabla 7. Factor de complejidad técnica 42](#_Toc516344399)

[Tabla 8. Factor de ambiente 43](#_Toc516344400)

[Tabla 9. Tiempo de duración del proyecto 45](#_Toc516344401)

[Tabla 10. Costos totales 45](#_Toc516344402)

**Índice de figuras**

[Figura 1. Tipos de materias primas y piezas 14](#_Toc516482611)

[Figura 2. Pieza 1 insertada en la primera materia prima. 14](#_Toc516482612)

[Figura 3.Primera pieza insertada en la primera área disponible. 15](#_Toc516482613)

[Figura 4. Segunda pieza insertada en posición vertical. 15](#_Toc516482614)

[Figura 5. Primera variante de corte. 16](#_Toc516482615)

[Figura 6. Segunda variante de corte. 16](#_Toc516482616)

[Figura 7. Segunda pieza insertada en el área donde estaba la anterior pieza. 17](#_Toc516482617)

[Figura 8. Tercera variante de corte. 17](#_Toc516482618)

[Figura 9. Cuarta variante de corte. 18](#_Toc516482619)

[Figura 10. Quinta variante de corte. 18](#_Toc516482620)

[Figura 11. Variante de corte 6, 7, 8 y 9. 19](#_Toc516482621)

[Figura 12. Fin del primer ciclo del algoritmo. 19](#_Toc516482622)

[Figura 13. Cortes verticales. 19](#_Toc516482623)

[Figura 14. Estructura de RUP 24](#_Toc516482624)

[Figura 15. Diagramas del modelo de objetos del dominio 30](#_Toc516482625)

[Figura 16. Diagrama del caso de uso del sistema 33](#_Toc516482626)

[Figura 17. Diagramas de clases 35](#_Toc516482627)

[Figura 18. Diagramas de clases persistentes 36](#_Toc516482628)

[Figura 19. Estructura del archivo 36](#_Toc516482629)

[Figura 20. Descripción por campos 37](#_Toc516482630)

[Figura 21. Diagrama de implementación 38](#_Toc516482631)

# Introducción

En diversas industrias existe la necesidad de cortar los materiales en piezas de distintos tamaños para satisfacer la demanda de sus clientes[1], [2]. El problema consiste entonces en hallar una forma eficiente para efectuar dichos cortes, es decir, cubrir todos los requerimientos de cortes mientras se minimiza los recursos y se satisfacen ciertas restricciones[3].

En la literatura son conocidos como problemas de corte de materiales en dos dimensiones a aquellos procesos que a partir de tableros o láminas de grandes dimensiones de madera, metal, papel, plástico o vidrio se le realizan cortes con el objetivo de obtener piezas más pequeñas para ajustarse a las necesidades de los clientes[4].

En muchas industrias se presentan a diario problemas de este tipo, donde se incurren en grandes desperdicios de materia prima, estos gastos pueden ascender a miles de pesos en un año. Un estudio realizado por Carazo y Hurtado en la industria del calzado en Perú lo demuestra. Estos autores plantean que el 24% de empresarios señalan problemas de desperdicio de materiales, un 45% tiene falta de uniformidad de su producto, un 30% reconoce fallas y errores[5].

En estos problemas se cuenta con varias entidades fundamentales:

Materia prima: Tablero o lámina de forma rectangular, del cual se cuenta con un número finito.

Piezas: Rectángulo que surge una vez realizado un corte a la materia prima y su área es menor que la de la materia prima de donde surgió. El número de estas piezas responde a una demanda determinada.

Desperdicios: Recortes de material que quedan después de haber realizado todos los cortes.

Variante de corte: Forma en que se puede hacer cortes en una materia prima. Pueden existir cientos, miles o millones de variantes de corte, dependiendo del tamaño del problema. Mientras mayor sea el número de tipos de piezas que se desean obtener y las materias primas con que se cuenta, mayor será también el número de variantes de corte.

Debido a la complejidad de estos problemas, se asumirá que las materias primas y las piezas tendrán bordes regulares, es decir su forma debe ser rectangular.

En la actualidad es de gran importancia la optimización de los recursos con que se cuenta cuando se desea fabricar un producto determinado. Mayor desperdicio de los recursos y materias primas puede influir en que aumenten los costos de producción, lo que producirá un mayor precio de venta, y como consecuencia cederle espacios a las empresas con que se compite. Cuba no se encuentra aislada de este contexto.

En el territorio nacional -tanto en el sector privado como en el estatal- existen empresas donde se presentan a diario problemas como los descritos anteriormente. Se pudiera mencionar a las UEB Gráfica que existen en algunas provincias del país, donde se cuenta con materias primas de cartulina y papel, las que son cortadas en piezas más pequeñas, las que satisfacen una demanda determinada por tipos. Situaciones muy similares tienen lugar en cristalerías, carpinterías de madera y metal, con la única diferencia que los materiales donde se realizan los cortes no son los mismos.

La Investigación de Operaciones (IO) como rama de las matemáticas, brinda métodos de solución consistentes en el uso de modelos matemáticos y algoritmos con el fin de realizar un proceso de toma de decisiones, teniendo en cuenta la escasez de recursos para determinar cómo se pueden maximizar o minimizar los mismos[6].

Mediante la IO se puede modelar matemáticamente la problemática de cortar una materia prima, obteniendo el mínimo desperdicio (o desperdicio óptimo)[6]. Esta modelación está basada en la Programación Lineal en Enteros (PLE).

En estos problemas cada variante de corte es una variable y los desperdicios que acarrean dichas variantes, coeficientes de la función objetivo a minimizar. Estos problemas pueden llegar a contar con una gran cantidad de variables puesto que pueden existir múltiples combinaciones de corte a ser aplicadas en una materia prima[6].

Las primeras investigaciones sobre este tema surgieron en la segunda mitad del siglo pasado, la que fue llevada a cabo por Kantorovitch[7].

En Cuba no fue hasta la década del 90 cuando en la Universidad de Cienfuegos, surgieron investigaciones[8] a partir de la utilización de métodos matemáticos, con el fin de disminuir la cantidad de variables en estos problemas, sin que se pierda el sentido matemático de los mismos, para ello se utilizan unas variantes denominadas de sustitución, las cuales se emplean para buscar dependencias lineales entre ellas y las variantes del problema. Aunque este proceso necesita de tiempo, se ha demostrado que, al reducir la cantidad de variables, el tiempo de procesamiento para la obtención de la solución óptima es reducido considerablemente.[8].

Estos trabajos, aprovechando el desarrollo de los poderosos medios de cálculo electrónicos, devinieron en sistemas que calculaban las variantes de corte y reducían de manera significativa las mismas; abarcando solo estos dos aspectos, debido a que necesitaban un alto uso de los recursos de los computadores (memoria y capacidad de procesamiento), los cuales eran limitados (para esos problemas de tanta magnitud). Luego, con la salida de estos sistemas, se debía formular el modelo matemático, procesarlo e interpretar la solución alcanzada[6].

En el año 2008 en dicha universidad se llevó a cabo una investigación[6] que tuvo como finalidad el desarrollo de un software, nombrado OPTIVACORTUNI, capaz de generar todas las variantes de corte para problemas en una dimensión, que se aprovecha del aporte del investigador León Rodríguez[8] para disminuir el número de variables en el problema, obtener la solución óptima e interpretar los resultados de manera que el usuario pueda entender fácilmente.

Cuando se trata con problemas en dos dimensiones crece la complejidad a la hora de generar las variantes de corte, casi siempre lo que se trata es buscar un número grande de estas y a partir de ellas llegar a la solución óptima[9]–[11], debido a que es muy difícil asegurar que se han generado todas las formas posibles de realizar los cortes.

En la actualidad no se dispone de un sistema que obtenga la solución óptima a partir de las variantes de corte en problemas de dos dimensiones. Los software que existen utilizan heurísticas para llegar a la solución, lo que puede incurrir en que el desperdicios no sea el óptimo, debido a que no se puede asegurar que se ha encontrado la mejor solución con respecto a las variantes del problema.

Por ello se ha identificado como **problema a resolver** la necesidad de disponer de un sistema informático para ser utilizado en la optimización del proceso de corte de materiales en dos dimensiones.

Se identifica como **objeto de estudio** el proceso de corte de materiales, mientras que el **campo de acción** se presenta como las tecnologías informáticas en función de la disminución de desperdicios en el proceso de cortes de materiales en dos dimensiones. Se plantea como **idea a defender** que la utilización de un sistema informático tiene influencia positiva en la disminución de los desperdicios, la organización y control del proceso de corte de materiales en dos dimensiones.

Como **objetivo general** se plantea, desarrollar un sistema informático que se complemente con OPTIVACORTUNI para ser utilizado en los proceso de corte de materiales en dos dimensiones.

Del antes mencionado propósito se desprenden los siguientes **objetivos específicos**:

* Analizar las características y conceptos principales del corte en dos dimensiones.
* Diseñar el sistema informático.
* Implementar la determinación de las variantes de corte primarias y de sustitución en dos dimensiones.

Para cumplir los objetivos trazados se desarrollaron las siguientes **tareas**:

* Resumir aspectos de actualidad relacionados con:
* El proceso de corte de materiales en la industria.
* Aplicaciones en otras ramas de la ciencia.
* Herramientas informáticas existentes que puedan servir de soporte para desarrollar el sistema.
* Algoritmizar el proceso de generación de variantes de corte en dos dimensiones.
  + Traducir del lenguaje natural al lenguaje matemático
  + Traducir del lenguaje matemático a pseudocódigo.
* Implementar el algoritmo en un lenguaje de programación.
* Estudio y puesta a prueba del algoritmo implementado.
* Identificar los requerimientos del sistema.
* Construir el módulo de introducción de datos y generación de patrones de corte del sistema informático.
* Integrar los módulos anteriores al nuevo sistema.
* Construir el módulo de interpretación de los resultados.
* Integrar este módulo al nuevo sistema.
* Validar el sistema informático.
* Documentar la investigación.

Los **métodos científicos** utilizados son:

* Del nivel teórico:
* Análisis, síntesis, inducción y deducción para resumir lo investigado y estudiado sobre el proceso de corte de materiales en la industria y las herramientas informáticas existentes que puedan servir de soporte para desarrollar el sistema.
* El método de algoritmización de procesos (tránsito del lenguaje matemático al lenguaje computacional) asociando técnicas novedosas.
* El método de tránsito de lo abstracto a lo concreto para establecer relación entre la interpretación de los resultados y la realidad objetiva existente.
* Del nivel empírico:
* El método de observación para constatar, en determinados momentos de la investigación, el trabajo que se realiza en la industria y/o empresas para optimizar el proceso de corte de materiales.
* La entrevista a operarios, tecnólogos y directivos como método para confrontar la experiencia empírica y los conocimientos con respecto al posible perfeccionamiento del proceso de corte de materiales.

El **aporte** de esta investigación está dado en que por primera vez se desarrolla en la provincia un sistema informático que pueda ser empleado en los proceso de corte de materiales en dos dimensiones.

Este documento se encuentra dividido en tres capítulos, contiene además conclusiones, recomendaciones y referencias bibliográficas.

Contenido por capítulos:

Capítulo 1: Fundamentos teórico

Se describe los fundamentos teóricos del tema a tratar, así como los conceptos relacionados con el dominio del problema. Se detallan los métodos matemáticos que puedan dar respuestas y solución al problema, se explica el algoritmo que ha sido aplicado. Se exponen los lenguajes y metodologías utilizados para el desarrollo de sistemas informáticos, teniendo en cuenta las tendencias actuales, los software que han servido de soporte, y por último las herramientas empleadas para el desarrollo de la aplicación.

Capítulo 2: Análisis y diseño de la propuesta de solución

Se detalla la solución propuesta, donde se emplea Proceso Unificado de Desarrollo de Software (RUP) como la metodología a seguir para identificar las entidades, los conceptos fundamentales en el escenario donde se lleva a cabo el proceso, el diagrama de clases del modelo de objetos del dominio y las reglas del negocio a considerar.

Se identifica los actores, los casos de usos del sistema, las relaciones entre estos se pueden observar en el diagrama de caso de uso del sistema, se describe detalladamente de cada uno de los casos de uso del sistema. Se enumeran los requisitos funcionales y no funcionales.

Se presenta el diagrama de clases del diseño, el diseño de los archivos empleados y el diagrama de implementación.

Capítulo 3: Estudio de factibilidad y validación de la solución

Se realiza un análisis que permita determinar la factibilidad al desarrollar la solución propuesta. Se determinan los beneficios tangibles e intangibles, los cuales deben ser mayores que los costos. Se estiman el esfuerzo humano y el tiempo de desarrollo que se requieren para su elaboración.

Es validada la propuesta de solución.

# 1 – Fundamentos teóricos

## 1.1 – Introducción

En este capítulo se hace referencia al fundamento teórico del tema de investigación. Se describe el dominio del problema, detallando los conceptos principales. Se muestran los primeros estudios relacionados con el tema y las investigaciones que se han llevado a cabo en Cuba. Se habla de la ciencia que brinda solución al problema tratado. Se exponen los métodos de solución brindados por la Investigación de Operaciones. Se explica de forma detallada el algoritmo que permite generar las variantes de corte. Se realiza un análisis de los software existentes que pueden servir de base para el desarrollo del actual sistema. Se estudian las tendencias, tecnologías y metodologías actuales, determinándose cuáles van a ser las utilizadas en el desarrollo del sistema.

## 1.2 – Descripción del dominio del problema

Los procesos de corte de materiales a gran escala tuvieron lugar en la Revolución Industrial, en esta época la producción pasa de pequeños talleres a grandes empresa y fábricas, incorporándose máquinas especializadas destinadas a aumentar la eficiencia productiva. El corte de materiales comienza a formar parte de la cotidianidad de estas fábricas[6]. En la Segunda Guerra Mundial es cuando se piensa en la necesidad urgente de asignar los recursos escasos en las operaciones militares, en problemas tácticos y estratégicos, surgiendo así la disciplina Investigación de Operaciones y dentro de ella los problemas de Programación Lineal, los que están íntimamente relacionados con los conocidos en la literatura como problemas de corte de materiales[12].

En el año 1960 Kantorovitch[7] realiza la primera investigación relacionada con los problemas de corte en dos dimensiones la cual surge como una extensión de estos problemas en una dimensión. Similares problemas fueron tratados por Paull y Walter, Metzger, Eilon [13]–[15] con relativo éxito para problemas pequeños. Gilmore y Gomory[16], [17] resuelve el problema de corte de una dimensión utilizando la técnica de generación de columnas con gran éxito, esto estimula la investigación para el caso de dos dimensiones.

En la Universidad de Cienfuegos en la década del 90 se realizaron algunas investigaciones relacionadas con el tema. De León[8] con su tesis demostró la posibilidad de reducir hasta en un 80% el número de variables en los problemas, lo que disminuye de forma considerable el tiempo en que se encuentra la solución óptima.

Dyckhoff[18] desarrolló un esquema de clasificación. Este esquema permite identificar las propiedades más comunes de los problemas de corte según las siguientes características:

1. Dimensionalidad

* Número de dimensiones.

2. Tipo de asignación

* Todos los tableros y una parte de las piezas demandadas.
* Una parte de los tableros y todas las piezas demandadas.

3. Surtido de tableros almacenados

* Un tablero.
* Tableros idénticos.
* Tableros diferentes.

4. Surtido de piezas demandadas

* Pocas piezas de diferentes tamaños.
* Muchas piezas de muchos tamaños.
* Muchas piezas de relativamente pocas dimensiones.
* Muchas piezas, pero idénticas.

Morabito y Morales[19], [20] hacen la siguiente clasificación de los patrones de corte:

Patrones guillotina y no guillotina:

Un corte es de tipo guillotina si cuando se aplica sobre un rectángulo produce dos nuevos rectángulos, es decir, si el corte va de un extremo a otro del rectángulo original; en otro caso se denomina de tipo no guillotina. Un patrón es de tipo guillotina si se puede obtener por sucesivos cortes de tipo guillotina. Un patrón es no guillotina si es obtenido por sucesivos cortes de guillotina y no guillotina.

En la industria cubana se presentan a diario situaciones como las antes descritas, es muy común que se cuente con tableros de diferentes dimensiones y a partir de ellos obtener muchas piezas de diferentes tamaños, las cuales deben satisfacer el número demandado de cada una. Estos procesos generan desperdicios o sobrantes que no tienen ninguna utilidad práctica.

Si los operarios solo hacen uso de su conocimiento acumulado para realizar los cortes, los desperdicios que se generan en la mayoría de los casos no van a ser los menores. Cuando el número de tipos de piezas que se desean obtener y el número de tipos de materias primas con que se cuenta es muy grande, es muy difícil para el operario determinar la forma óptima de realizar los cortes, por lo que se incurre en desperdicios innecesarios que pueden representar miles de pesos tirados a la basura.

### 1.2.1 Alternativa de solución

A partir del surgimiento y posteriormente la popularización de la Investigación de Operaciones se han empleado sus métodos en la rama empresarial con el fin de alcanzar mayores resultados económicos.

La Programación Lineal en Enteros permite la modelación y obtención de resultados óptimos en los proceso de corte de materiales en dos dimensiones.

Este método permite reducir de forma considerable los desperdicios generados en dicho proceso, y como consecuencia, la reducción de gastos innecesarios.

## 1.3 – Análisis teórico de los métodos de solución.

### 1.3.1 – Investigación de operaciones.

La investigación de operaciones se puede definir como la aplicación de métodos científicos en la mejora de la efectividad en las operaciones, decisiones y gestión[21]–[24], o como la ciencia de aplicar los recursos disponibles para conseguir la satisfacción óptima de un objetivo específico deseado[12].

Esta rama de la matemática surgió como consecuencia del enfrentamiento entre las grandes potencias agrupadas en dos coaliciones militares, los Aliados y las Potencias del Eje. Debido a los esfuerzos bélicos, existía la urgente necesidad de asignar recursos escasos a las distintas maniobras militares y a las actividades que componían cada operación de la manera más eficaz. Los estadounidenses y británicos reunieron a un gran número de investigadores para que aplicaran el método científico a éste y a otros problemas estratégicos y tácticos. Estos son considerados los primeros equipos de Investigación de Operaciones[25].

El éxito en las contiendas bélicas trajo consigo que se despertara el interés y fuese aplicado en un ámbito distinto al militar. Al inicio de la década de los años cincuenta se introdujo el uso de la investigación de operaciones en una serie de organizaciones industriales, de negocios y del gobierno[25].

La revolución de las computadoras trajo consigo que se popularizara esta disciplina. El manejo eficaz de los complejos problemas inherentes a la IO casi siempre requiere un gran número de cálculos. Realizarlos de forma manual puede resultar casi imposible, por lo cual el desarrollo de la computadora electrónica digital, con su capacidad para hacer cálculos aritméticos, miles o tal vez millones de veces más rápido que los seres humanos, fue una gran ayuda para esta disciplina[25].

Los problemas de corte de materiales en dos dimensiones pueden ser modelados mediante la IO, utilizando la programación lineal en enteros.

### 1.3.2 – Programación lineal en enteros.

El desarrollo de la programación lineal ha sido clasificado como uno de los avances científicos más importantes de mediados del siglo xx. Su efecto desde 1950 ha sido extraordinario. En la actualidad es una herramienta de uso que ha ahorrado miles o millones de dólares a muchas compañías o negocios, incluso empresas medianas, en los distintos países industrializados del mundo; su aplicación a otros sectores de la sociedad se ha ampliado con rapidez. Una proporción muy grande de los programas científicos en computadoras está dedicada al uso de la programación lineal[25].

La programación lineal en enteros utiliza un modelo matemático para describir el problema. El adjetivo lineal significa que todas las funciones matemáticas del modelo deben ser funciones lineales. En este caso, la palabra programación no se refiere a términos computacionales; en esencia es sinónimo de planeación. La palabra enteros está relacionada con el dominio de la variable de decisión, en el caso que todas las variables deban tomar valores enteros, se está en presencia de un problema de programación en enteros puro. Si al menos una de las variables puede ser continua se nombra mixto. Por lo tanto, la programación lineal involucra la planeación de actividades para obtener un resultado óptimo; esto es, el resultado que mejor alcance la meta especificada —de acuerdo con el modelo matemático— entre todas las alternativas factibles[6], [25].

Los problemas de corte de materiales en dos dimensiones tratados en la presente investigación se clasifican en puros, puesto que todas las variables -que representan la cantidad de veces a ejecutar un corte- deben ser números enteros.

### 1.3.3 – Modelación matemática del problema.

El problema general de la programación lineal consiste en dada una función lineal de n variables, determinar valores no negativos de las mismas que hagan óptimo (mínimo o máximo) el valor de la función lineal, sujeto a determinadas condiciones que asume la forma de un sistema de ecuaciones y/o inecuaciones lineales[6]. En el caso de la programación lineal en enteros también debe ocurrir que los valores de las variables encontrados sean números enteros.

En los problemas de corte de materiales en dos dimensiones cuando es empleada la programación lineal como método de solución, la función objetivo siempre a minimizar se forma con los desperdicios de cada patrón de corte y las restricciones se componen con las demandas por tipo de pieza y las cantidades que aparecen en aquellos patrones de corte que incluyen dichas piezas[26].

Debido al número de variables que pueden llegar a tener estos problemas, en la presente investigación se incorporan las variantes de corte de sustitución propuestas por el investigador de León[8] que permiten reducir hasta en un 80% el número de variables en dichos problemas. Estas variantes son el resultado de tomar una pieza y colocar - si es posible - las piezas más pequeñas dentro de ella[8].

A continuación se presenta la modelación del problema:[8]

SUJETO A:

DONDE:

n: Cantidad de variantes de corte.( )

m: Cantidad de piezas.

r: Cantidad de tipos de longitudes de materia prima.

Qk: Cantidad de unidades de materia prima de longitud tipo k.

j: Índice de la variante de corte.

i: Índice de la pieza.

k: Índice del tipo de longitud de la materia prima.

Ni: Necesidad de piezas tipo i.

DMP: Desperdicio de materia prima.

djk: Desperdicio de materia prima de longitud tipo k según la variante de corte j.

Pijk: Cantidad de piezas tipo i que se obtienen al cortar una unidad de materia prima de longitud tipo k según la variante de corte j.

xjk: Cantidad de materia prima de longitud tipo k que es necesario cortar según la variante de corte j.

Z: Conjunto de los números enteros.

u: Índice de la variante de sustitución.

t: Cantidad de variantes de sustitución.

xuk: Utilización de la variante de sustitución u en la materia prima de longitud tipo k.

buk: Desperdicio de materia prima de longitud tipo k generado por la variante de sustitución u.

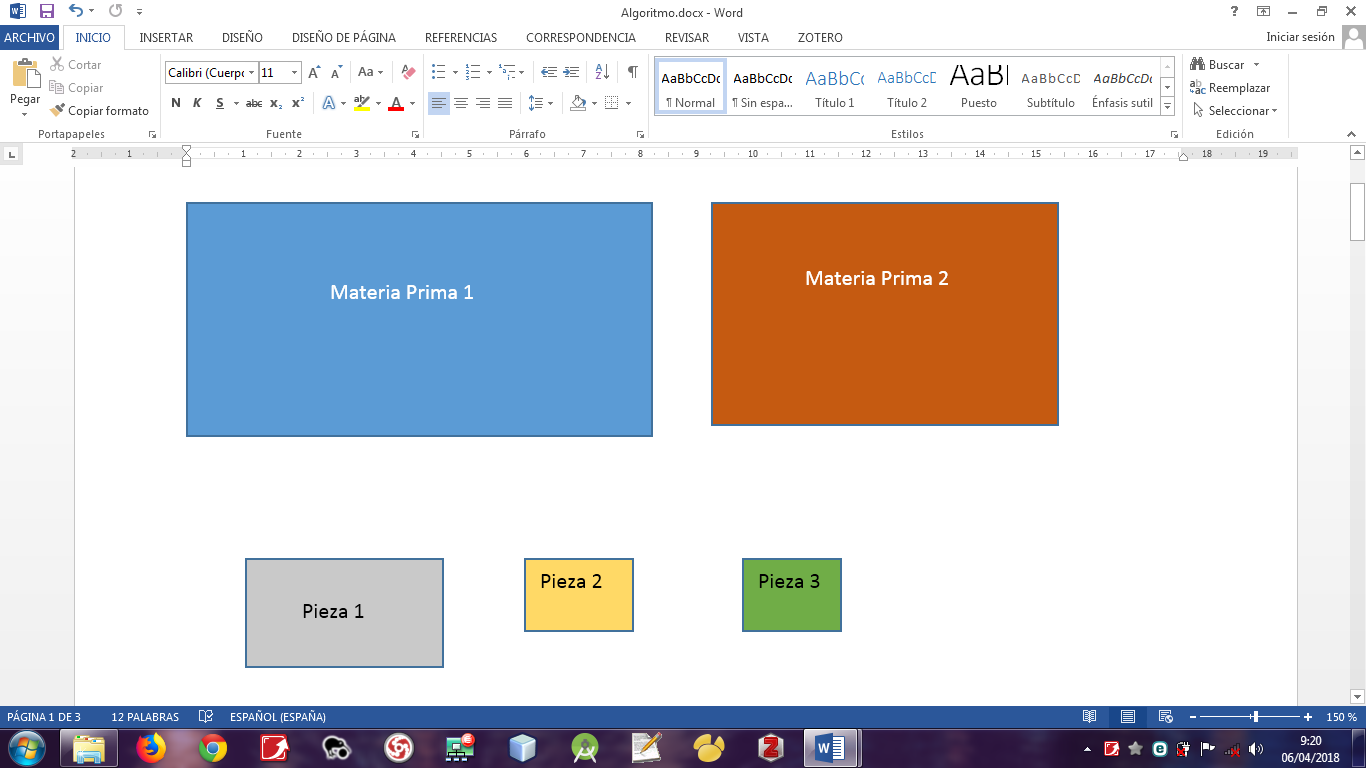
aiuk: Coeficiente de utilización de la pieza i en la variante de sustitución u para la materia prima de longitud tipo k.

nk: Cantidad de variantes de corte para la materia prima de longitud tipo k.

### 1.3.4 – Generación de variantes de corte en dos dimensiones.

La generación de variantes de corte en dos dimensiones es un proceso muy complejo pero necesario para la modelación y futura obtención del valor óptimo en los problemas de Programación Lineal en Enteros. Por ello a continuación se explica el algoritmo diseñado y utilizado para generar dichas variantes a partir de la siguiente problemática:

Se cuenta con dos materias primas y se desean obtener piezas de tres tipos, como se observa en la figura 1.

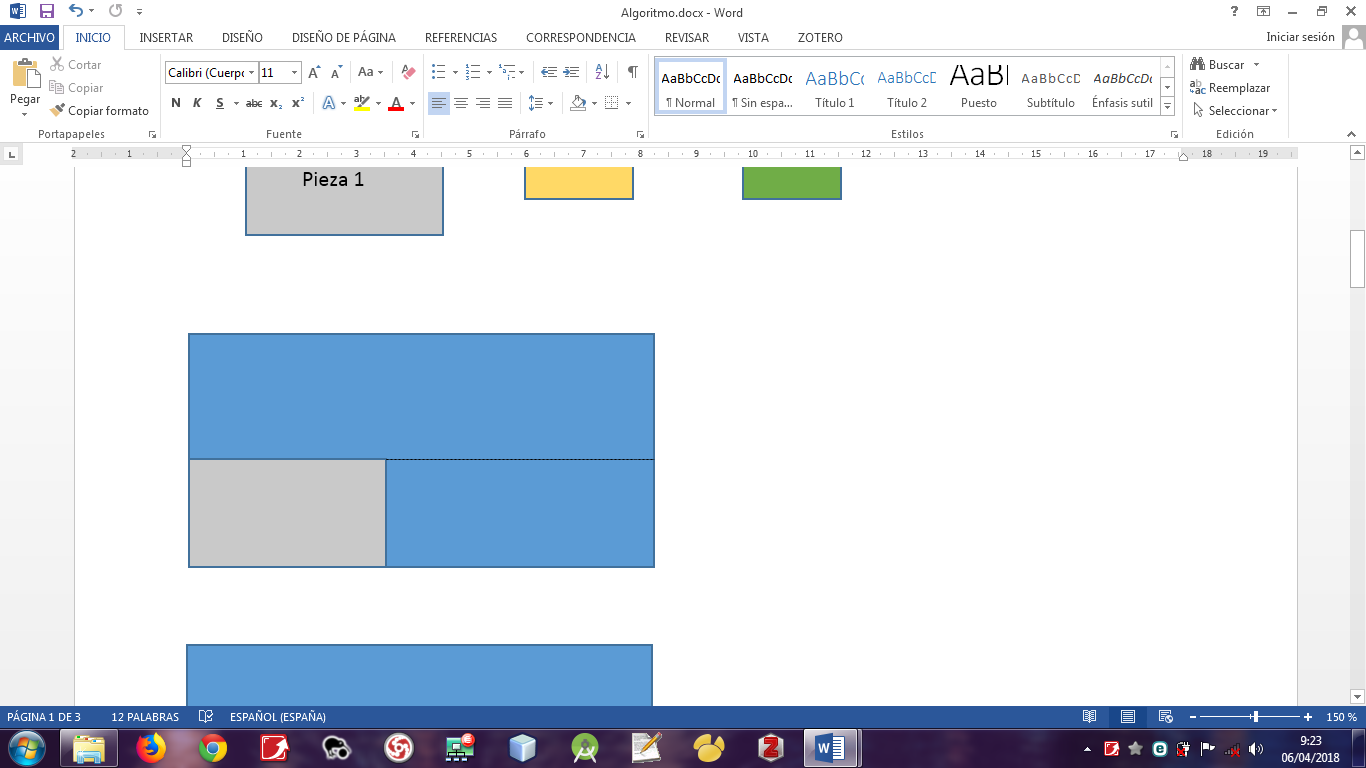


**Figura 1. Tipos de materias primas y piezas**

Primeramente se debe ordenar las materias primas según su área, de mayor a menor, hacer lo mismo con las piezas.

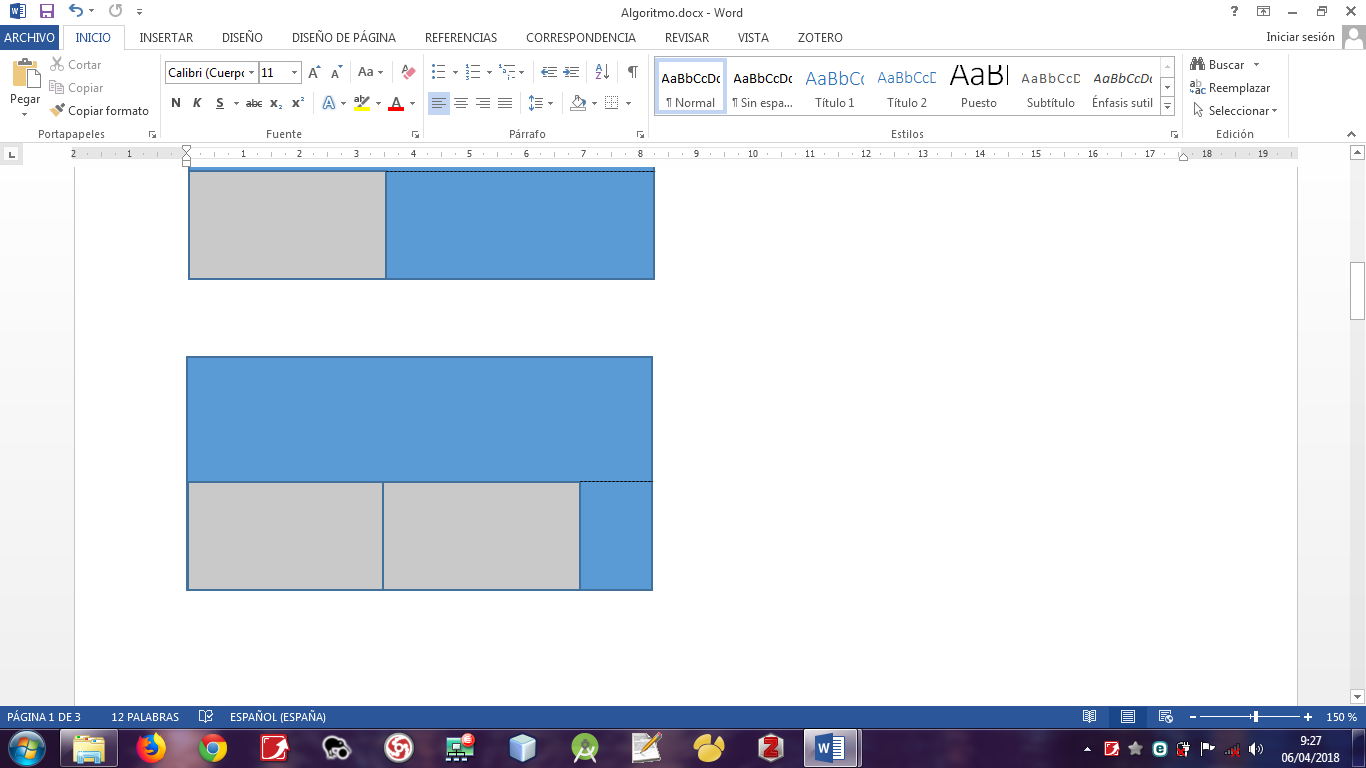
Se toma la primera pieza y se trata de insertar en la primera materia prima (anteriormente ordenadas).

Como se puede apreciar en la figura 2, la pieza 1 puede ser insertada en la materia prima 1, y como resultado ha quedado dos rectángulos o áreas disponibles donde se pueden insertar más piezas, el que se encuentra debajo de la línea negra y el que está por encima de ella.



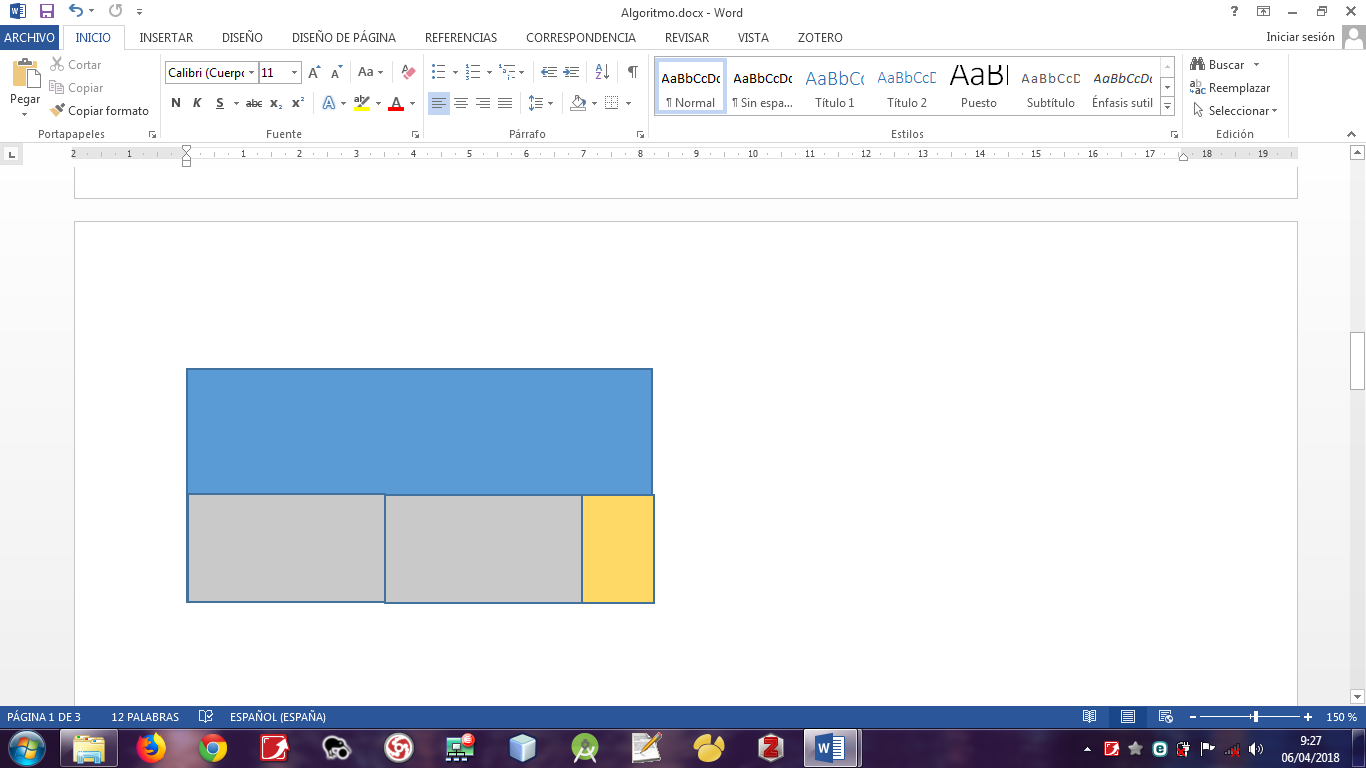
**Figura 2. Pieza 1 insertada en la primera materia prima.**

A continuación se toma nuevamente la primera pieza y se trata de insertar en la primera área disponible, como ocurre en la figura 3.



**Figura 3.Primera pieza insertada en la primera área disponible.**

De no poderse insertar la primera pieza en el área disponible (en este caso la que se encuentra a la derecha de la segunda pieza gris de la figura 3) se gira 90 grados y se verifica si puede ser insertada o no, si tampoco se puede insertar, entonces se toma la segunda pieza y se prueba insertar en posición horizontal, si no se puede insertar se gira 90 grados para probar en posición vertical, como ocurre en la figura 4.



**Figura 4. Segunda pieza insertada en posición vertical.**

La pieza 1 no pudo ser insertada en ninguna posición en el área disponible de la derecha del rectángulo de color gris (véase la figura 3), por lo que se procede a insertar la siguiente pieza (figura 4). Esta no puede ser insertada en posición horizontal pero si en vertical.

Como se puede observar se ha ocupado todo el espacio del área disponible más pequeña, pero esto siempre no ocurre así, en ocasiones queda un área disponible pequeña en la que no puede ser insertada ninguna pieza.

Ahora se procede a realizar el mismo proceso en la segunda área disponible, como se puede ver en la figura 5.

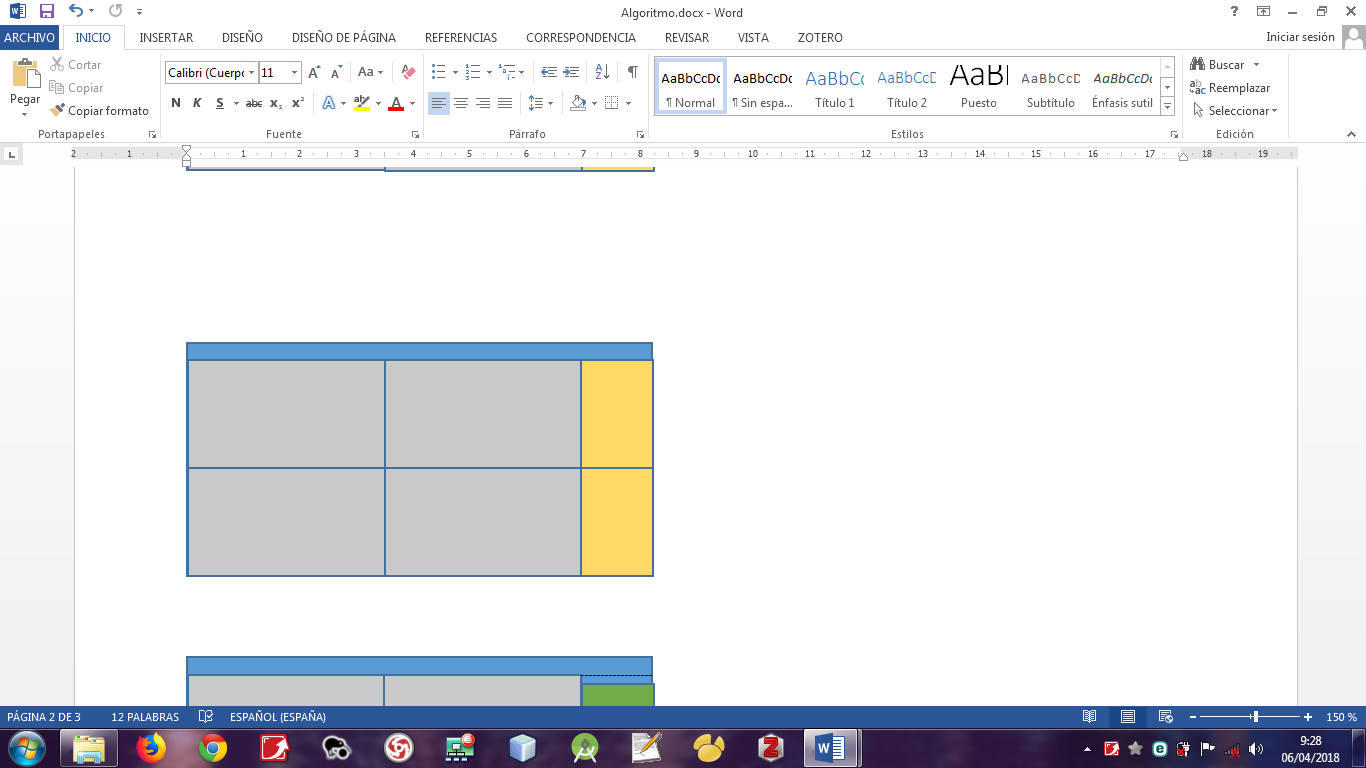


Figura 5. Primera variante de corte.

Ya se ha generado la primera variante de corte, puesto que en el área disponible no se puede insertar ninguna de las tres piezas que se demandan.

Para continuar con las otras variantes de corte se toma la última pieza insertada y se verifica la posición en la que se encuentra, si está en posición horizontal (no es este caso) se trata de poner en posición vertical. Si está en posición vertical, entonces se trata de insertar la siguiente pieza, primero en posición horizontal y si no es posible, entonces en la otra posición, como ocurre en la figura 6.

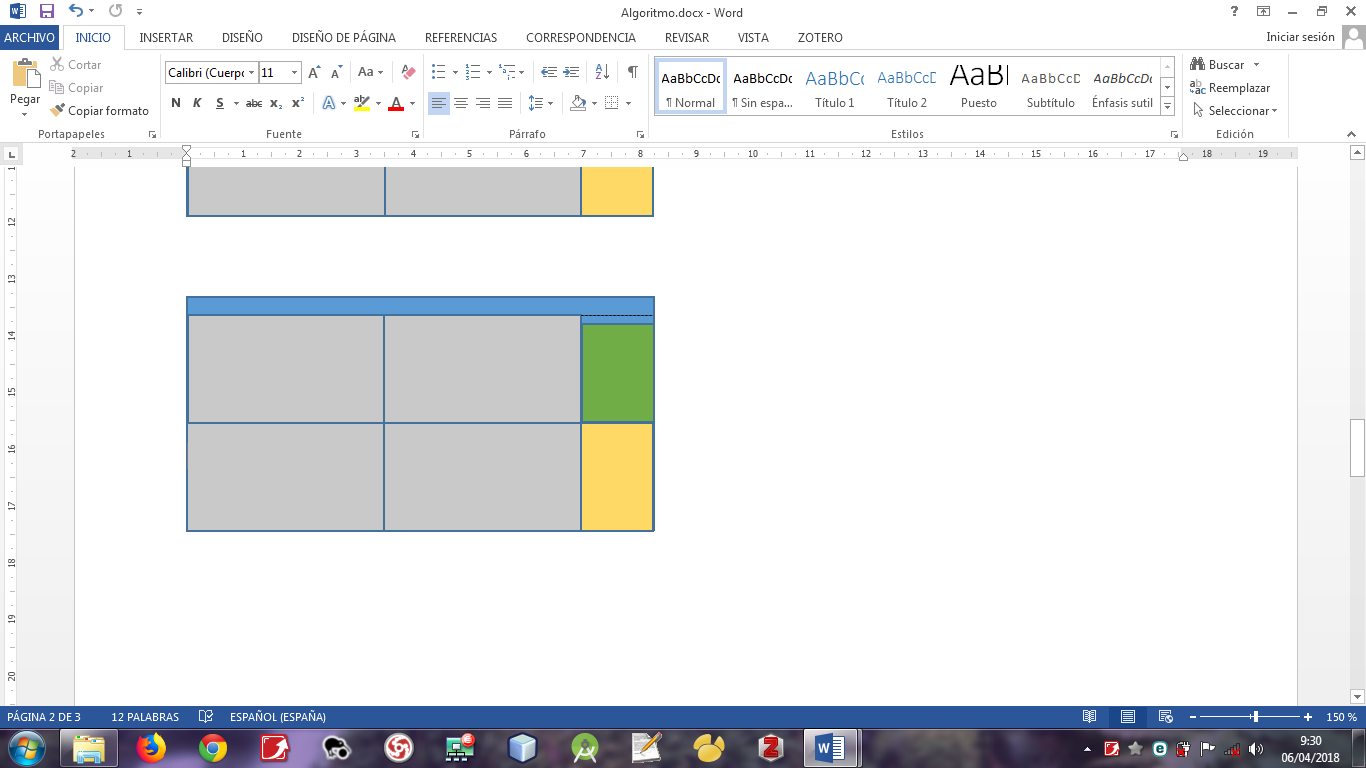
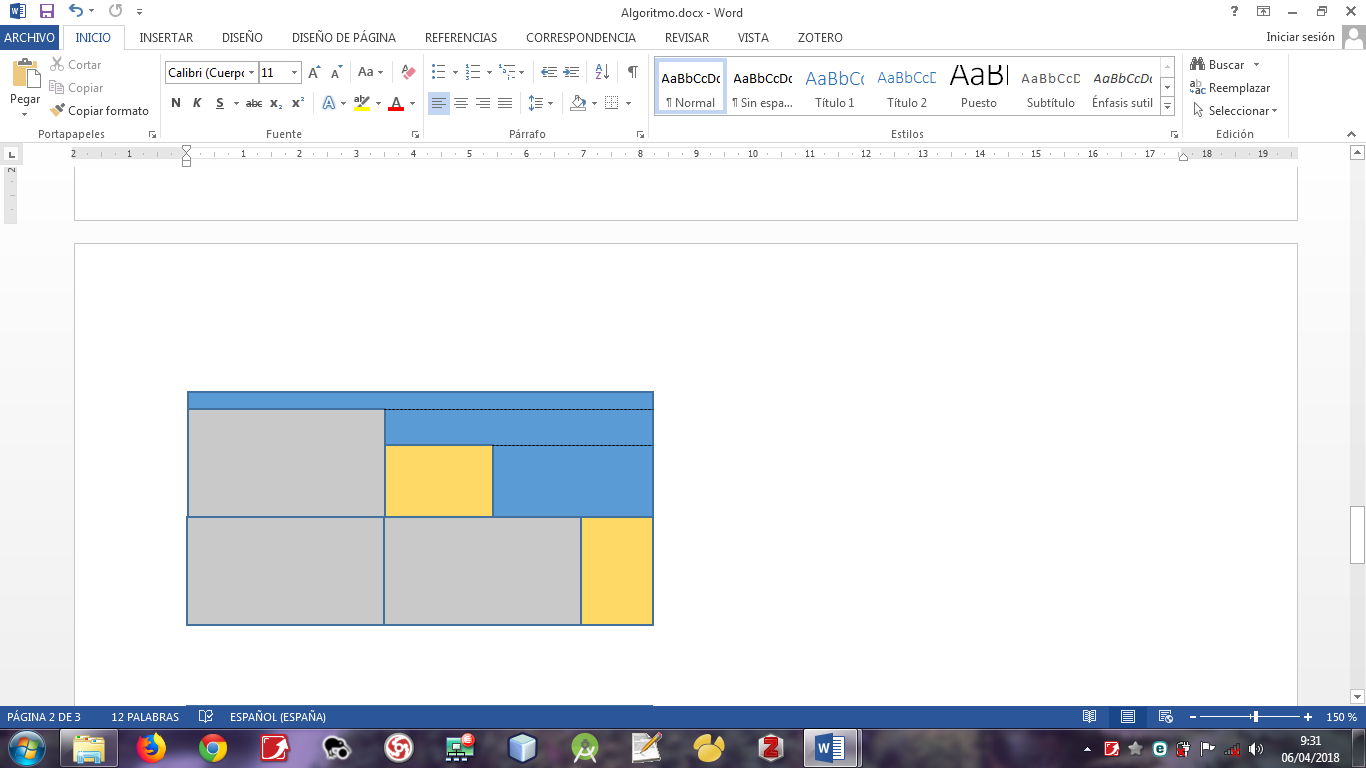


Figura 6. Segunda variante de corte.

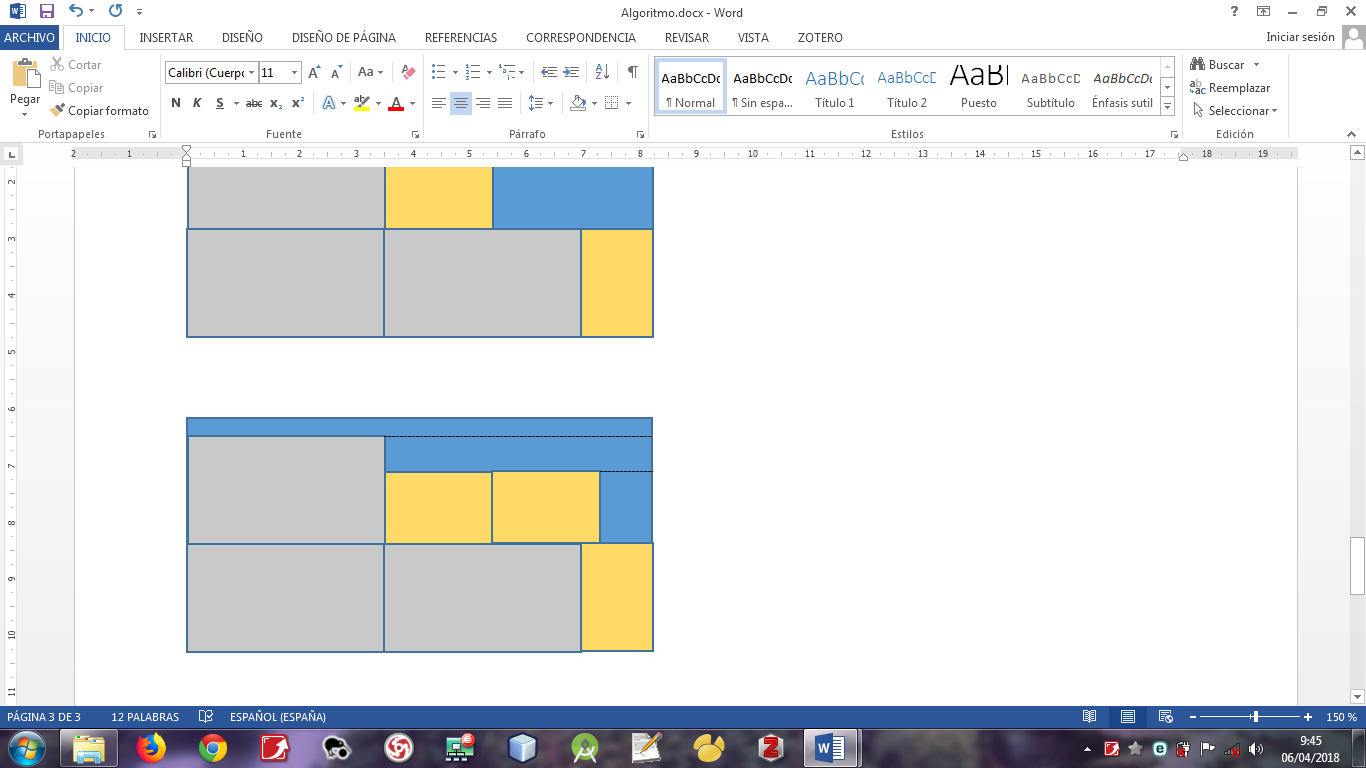
Como se puede apreciar en la anterior figura, ya no se puede insertar ninguna pieza en las áreas disponibles, por lo que se considera que se ha generado la segunda variante de corte.

Para generar la siguiente variante se toma otra vez la última pieza insertada (de color verde en la figura 6), como es la más pequeña no puede ser insertada otra pieza en su lugar, por lo tanto se elimina y se pasa a la penúltima pieza insertada(gris), que en este caso es una pieza de tipo 1. Se trata de insertar la siguiente pieza en el área donde se encontraba la anterior pieza. Nótese en la figura 7 que en vez de dos áreas disponibles ahora existen tres, pues el ancho de la pieza insertada es menor al de la pieza que se encuentra a su lado.



**Figura 7. Segunda pieza insertada en el área donde estaba la anterior pieza.**

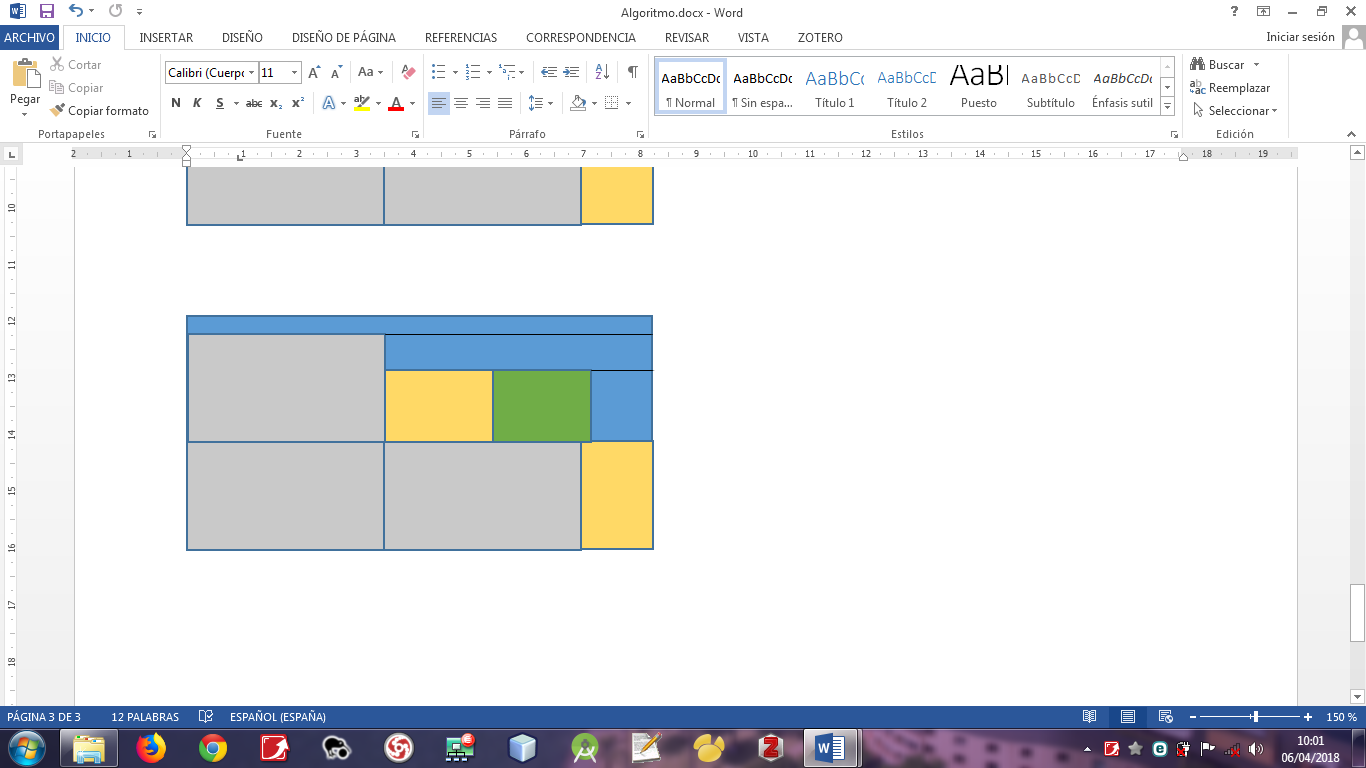
Ahora se procede a tratar de insertar la pieza de tipo 1 en el área disponible que se encuentra a la derecha de la pieza de tipo 2 (amarilla), pero esta acción no se puede realizar, la pieza no pude ser insertada en posición horizontal ni vertical, por lo que entonces se debe probar con la siguiente pieza, como ocurre en a figura 8. Nótese que siempre se trata de insertar la pieza más grande en posición horizontal, de no poderse insertar se gira 90 grados, si tampoco procede, se intenta el mismo procedimiento con las siguientes piezas.



**Figura 8. Tercera variante de corte.**

De nuevo se trata de insertar la primera pieza en el área que se encuentra a la derecha de la pieza amarrilla, pero no puede ser insertada en ninguna de las dos posiciones, lo mismo ocurre con las restantes piezas. Se procede a hacer lo mismo en las restantes áreas disponibles, pero sucede lo mismo. Por tanto, se ha generado la tercera variante de corte.

Ahora se trata de girar 90 grados la última pieza insertada, pero no se puede hacer, pues el área disponible donde ella había sido insertada no lo permite. Entonces se procede a tratar de insertar la siguiente pieza, obsérvese en la siguiente figura. Se ha generado la cuarta variante de corte.



**Figura 9. Cuarta variante de corte.**

Para generar la quinta variante de corte se prueba girando la última pieza (en la figura 9 es la pieza verde), pero esta no cabe en posición vertical en el área donde se había insertado. Como es la pieza más pequeña entonces se elimina y se pasa a la anterior pieza insertada (en la figura 9 es la pieza amarilla en posición horizontal) y se ubica en posición vertical, véase la figura 10. Entonces a su lado se inserta la pieza de mayor área.

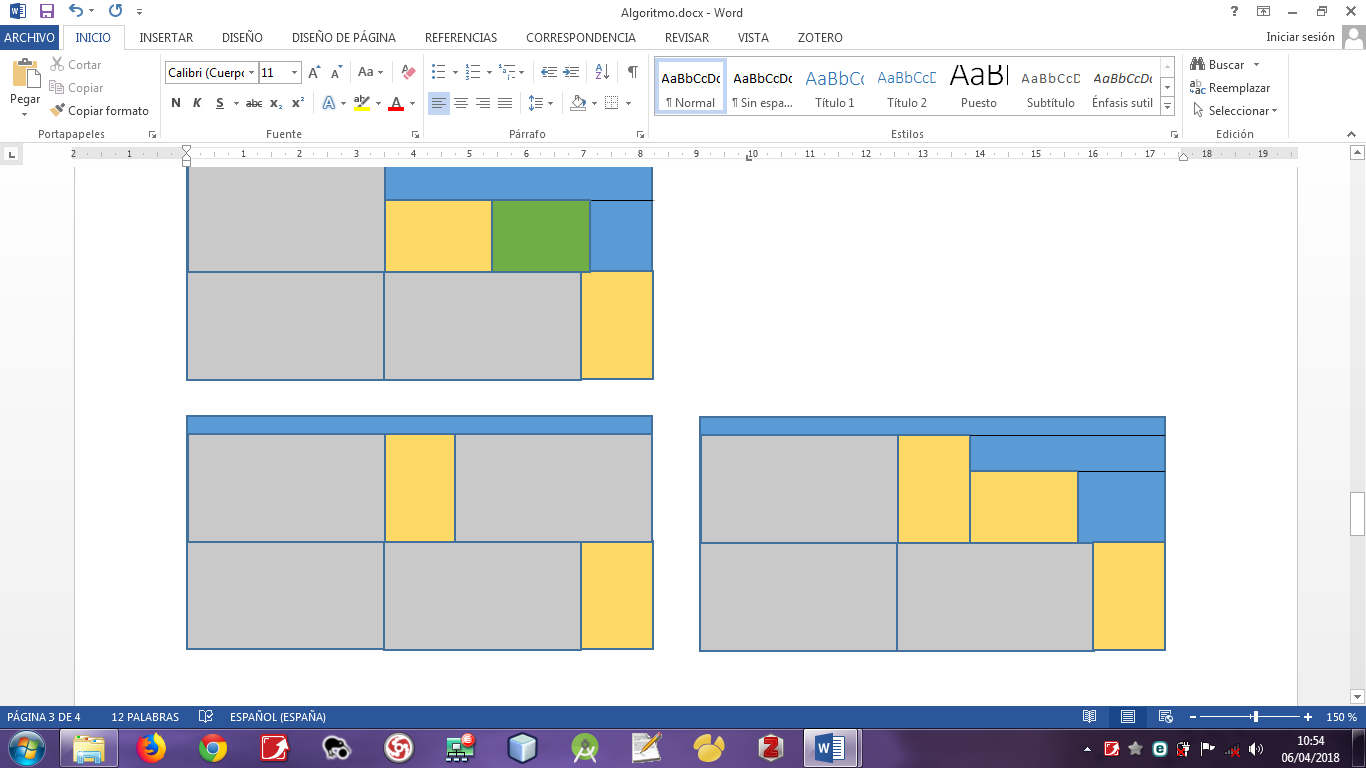
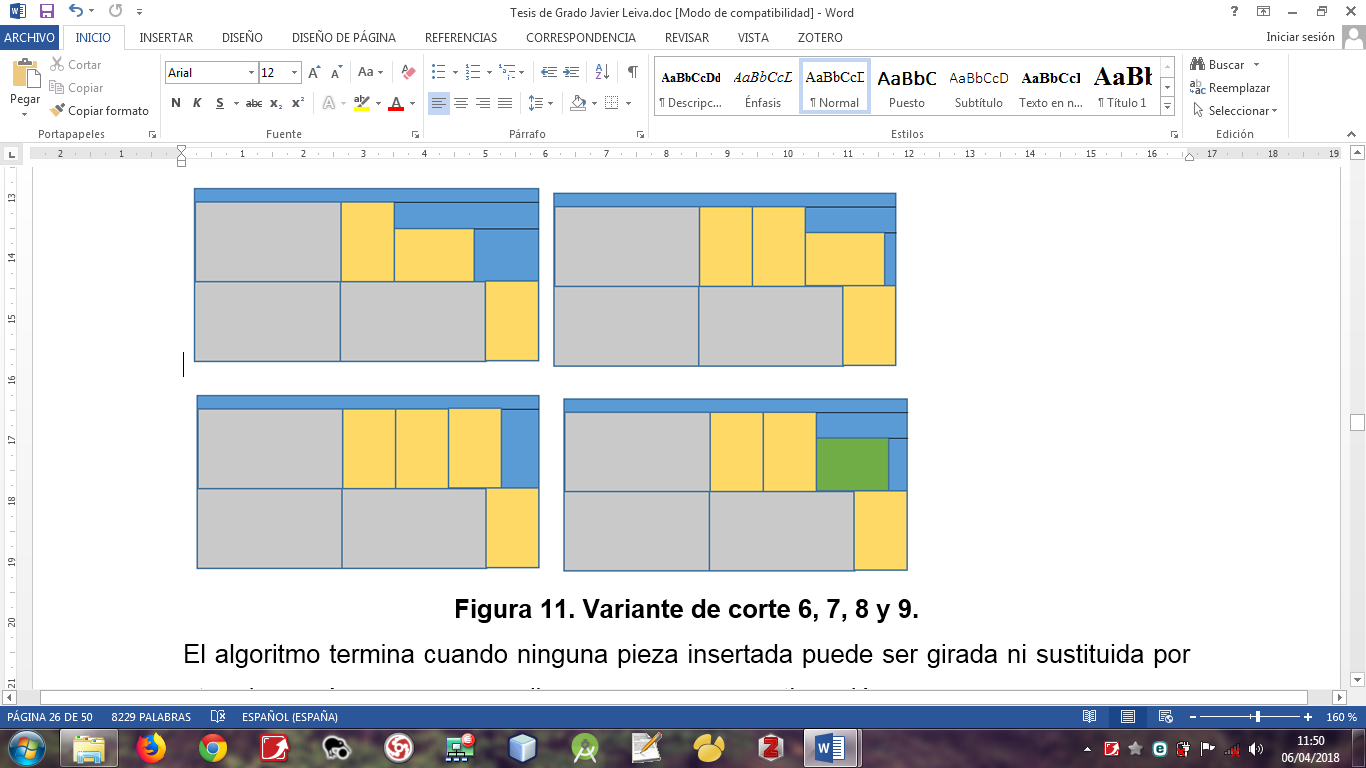


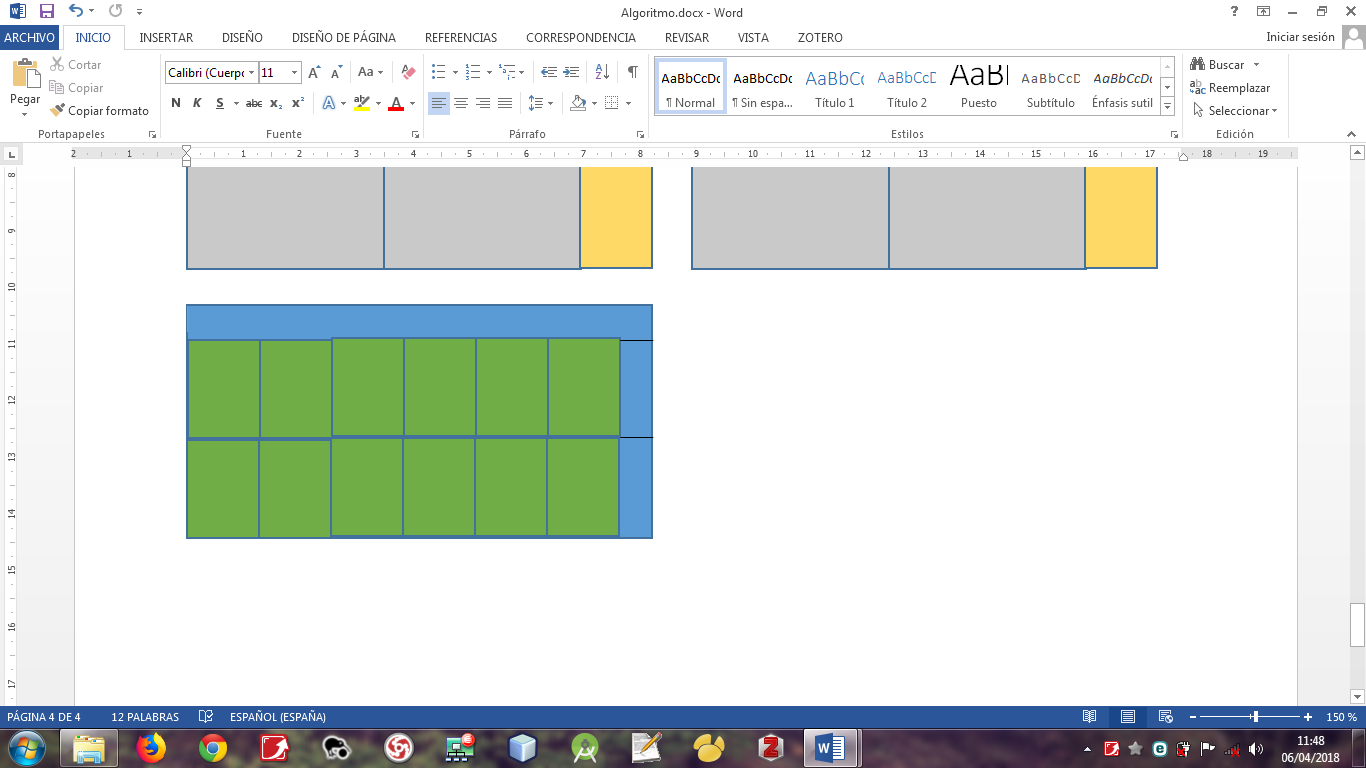
Figura 10. Quinta variante de corte.

Como se pude apreciar el procedimiento consiste en una vez generada la variante de corte, para llegar a la siguiente se toma la última pieza insertada, si está en posición horizontal se trata de poner en posición vertical, a continuación se trata de insertar en las áreas disponibles la pieza de mayor área, de no poderse realizar se gira y se prueba con cada una de las piezas, si la pieza estaba en posición vertical se prueba insertar la pieza que le sigue según el área, la que debe ser probada en las dos posiciones (véanse las variantes de corte de la figura 11). Si ninguna pieza se puede colocar en las áreas disponibles se dice que se ha generado una nueva variante de corte y se procede a realiza el mismo procedimiento.



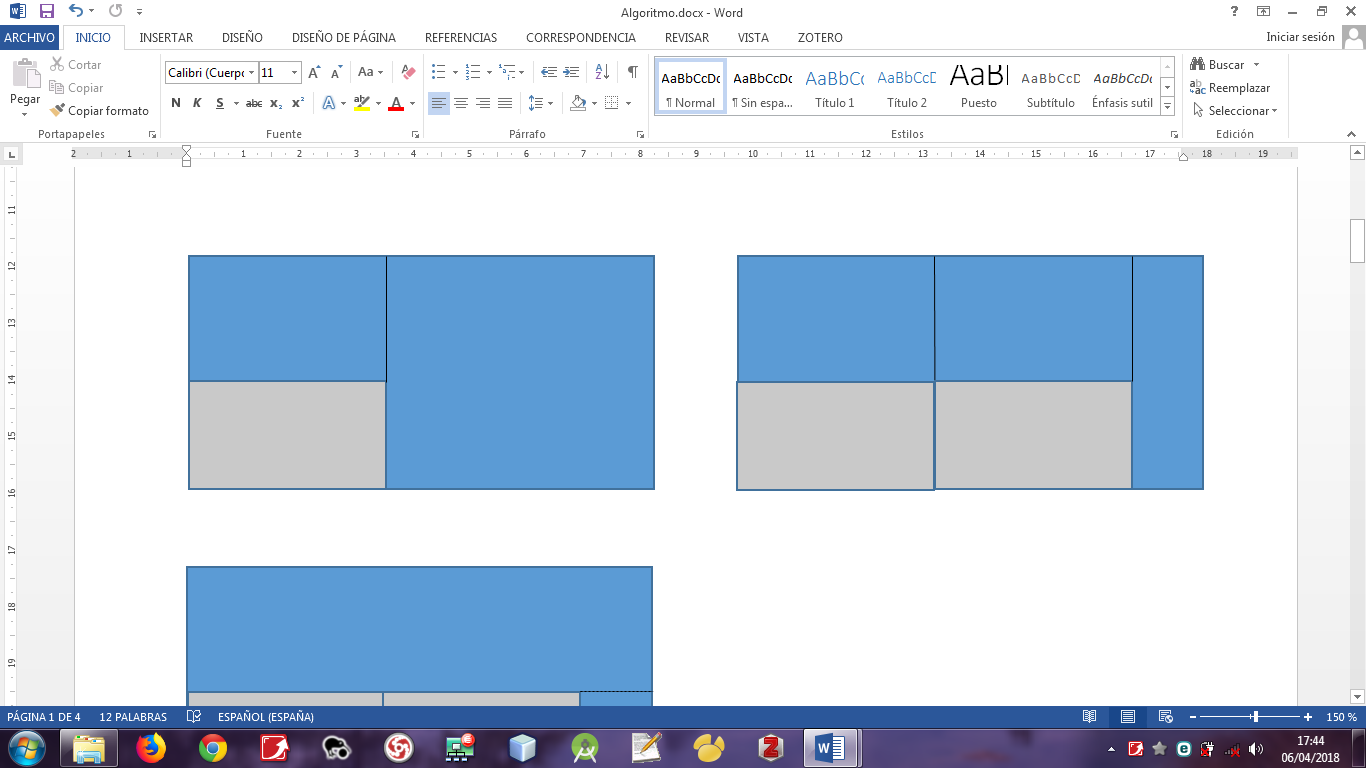
**Figura 11. Variante de corte 6, 7, 8 y 9.**

El primer ciclo del algoritmo termina cuando ninguna pieza insertada puede ser girada ni sustituida por otra pieza más pequeña que ella, como ocurre a continuación.



**Figura 12. Fin del primer ciclo del algoritmo.**

A continuación se procede a realizar el mismo algoritmo desde el principio pero ahora realizando los cortes de forma vertical, como se muestra a continuación.



**Figura 13. Cortes verticales.**

Realizar los cortes verticales hará que se generen algunas variantes de corte que no fueron encontradas con anterioridad. Sin dudas existen muchas variantes de corte repetidas que deben ser eliminadas.

El mismo procedimiento anteriormente descrito debe ser realizado con la segunda materia prima, y de esta forma se obtienen las variantes de corte para cada materia prima con que se cuenta.

## 1.4 – Otras aplicaciones

Los problemas de corte en dos dimensiones se pueden aplicar en otras situaciones como las que se describen a continuación.

En muchas ocasiones unidades de almacenamiento grandes, tales como pallets o contenedores son utilizadas para transportar objetos más pequeños. Estos problemas son conocidos como problemas de empaquetado[4].

Los problemas de corte de materiales en dos dimensiones y los de empaquetado, a primera vista dispares, están conceptualmente muy relacionados debido a la dualidad entre el material y el espacio ocupado por éste[18]. En ambos casos, existen dos tipos de objetos, grandes y pequeños, y el espacio definido por el objeto grande ha de ser ocupado por los objetos pequeños siguiendo ciertas normas. En ambos tipos de problemas, el ajuste entre objetos grandes y pequeños ha de ser lo más eficiente posible, de acuerdo con la función objetivo establecida, que en muchos casos se reducirá a minimizar el espacio no utilizado[4].

Otra aplicación del problema un tanto diferente se encuentra en el área de la ciencia de la computación; por ejemplo en el cálculo de la memoria virtual en las computadoras (Virtual Storage Computer-VSC).

La capacidad de memoria principal es un recurso limitado y extremadamente valioso en un computador digital, por tanto su utilización eficiente es indispensable. Los programas de computador debido a su tamaño, muchas veces no pueden ser almacenados totalmente en la memoria principal. De ahí que para mejorar la utilización de memoria, los programas deben ser estructurados de forma que diferentes segmentos puedan ser asignados a la misma área de memoria en instantes de tiempo definidos[27].

El problema consiste en organizar los segmentos de un programa en las páginas. Estas páginas tienen tamaño fijo y los segmentos tienen tamaños menores que el tamaño de una página; un segmento de un programa puede ser conformado por datos, funciones, procedimientos, entre otros. Dicho de otra forma, el problema consiste en determinar el menor número posible de páginas y también la organización de los segmentos en las páginas; esto es utilizar al máximo la capacidad de memoria sin desperdiciar espacio ocioso de memoria disponible[27].

## 1.5 – Descripción de los sistemas existentes

En la actualidad existen varios software relacionados con el tema a investigar, pero que tienen el inconveniente que los algoritmos empleados no garantizan la obtención de la solución óptima, pues utilizan algoritmos heurísticos que reducen de forma considerable el tiempo con que se encuentra la solución, pero no garantiza que sea la mejor. A continuación se presentan algunos de estos software.

**1.5.1 Software vinculados con el tema**

* CutLogic: es un optimizador 1D y 2D para cortar materiales rectangulares en industrias como la de madera, muebles, metal, cristal, etc. Permite cortes de guillotina y no guillotina, y aplica técnicas basadas en algoritmos heurísticos para proporcionar una optimización de alta calidad y confianza para varios escenarios de corte.
* CutMaster 2D: es un paquete de software profesional. Permite reducir el desperdicio y el coste ya que realiza cortes de alto rendimiento reutilizando los recortes. Utiliza algoritmos avanzados diseñados especialmente para optimizar la disposición de las piezas en el material, que puede ser metal, madera, cristal y similares. Trata de maximizar la productividad al ahorrar tiempo necesario para crear y analizar patrones.
* CorteCerto: es un paquete que permite realizar cortes en una y dos dimensiones, emplea heurísticas para obtener la solución, el tipo de corte es guillotinado. Incluye análisis económicos en los resultados que presenta.
* Cutting Optimization: es un programa para corte 1D y 2D. Se puede usar para el corte de hojas rectangulares de madera, cristal, metal y otros materiales industriales. También se usa para cortar piezas lineales como barras de madera, tuberías, barras de metal, etc.
* VarCort: Paquete realizado a principios de la década del noventa en la Universidad de Cienfuegos con el objetivo de generar las variantes de corte primarias en problemas de corte unidimensional. Cuenta con un ambiente MS-DOS. VarCort genera todas las variantes de corte primarias.
* VCORTE: Paquete realizado a mediados de la década del noventa en la Universidad de Cienfuegos con el objetivo de generar las variantes de corte primarias, las variantes de sustitución y eliminar aquellas dependientes, en problemas de corte unidimensional. Cuenta con un ambiente MS-DOS.
* Optivacortuni: Este software especializado en encontrar solución a problemas de corte unidimensional, integra la generación de variantes de corte primarias y de sustitución, la búsqueda de dependencias entre las variantes de corte, la eliminación de aquellas variantes de corte dependientes, la construcción del modelo matemático, el procesamiento computacional del modelo matemático y la interpretación económica de la solución óptima del modelo.

**1.5.2 Optivacomulti. Solución propuesta.**

El sistema propuesto lleva por nombre Optivacomulti, que significa **opti**mización de **va**riantes de **cort**e **multi**dimensionales.

El presente software genera un gran número de variantes de corte primarias y de sustitución, luego se combina buscando dependencias, lo que permite reducir considerablemente el número de variables en el problema, posibilitando la viabilidad de utilizar un algoritmo exacto que encuentra la solución óptima, lo que trae como consecuencia mayores beneficios económicos. Una vez calculada la solución es interpretada, haciendo corresponder cada variable con la variante de corte asociada y el valor con la cantidad de veces a realizar dicho patrón de corte. Esta solución se muestra de forma gráfica, lo que facilita la comprensión por parte del usuario.

## 1.6 – Tendencias, metodologías y/o tecnologías actuales

**1.6.1 Fundamentos de la metodología utilizada**

**1.6.1.1 Lenguaje de Modelamiento Unificado (UML)**

“UML” son las siglas de Unified Modeling Lenguage, notación (esquemática en su mayor parte) con que se construyen sistemas por medio de conceptos orientados a objetos. Se define como un lenguaje que permite especificar, visualizar y construir los artefactos de los sistemas de software[28]. Es un sistema notacional (que, entre otras cosas, incluye el significado de sus notaciones) destinado a los sistemas de modelado que utilizan conceptos orientados a objetos[29].

El Lenguaje de Modelado Unificado es un lenguaje estándar para escribir planos de software. UML puede utilizarse para visualizar, especificar, construir y documentar los artefactos de un sistema[29].

UML sirve para el modelado completo de sistemas complejos, tanto en el diseño de los sistemas como para la arquitectura hardware donde se ejecuten. Otro objetivo de este modelado visual es que sea independiente del lenguaje de implementación, de tal forma que los diseños realizados usando UML se puedan implementar en cualquier lenguaje que soporte las posibilidades de UML (principalmente lenguajes orientados a objetos)[30].

Se ha escogido este lenguaje de modelado para el desarrollo del sistema que se propone por la integración de este lenguaje con la metodología RUP.

**1.6.1.2 Proceso Unificado de Desarrollo (RUP)**

El Proceso Unificado de Desarrollo (Rational Unified Process en inglés, habitualmente resumido como RUP) es un proceso de desarrollo de software desarrollado por la empresa Rational Software, actualmente propiedad de IBM[31], [32]. Esta metodología ha sido seleccionada pues junto con UML, constituye la metodología estándar más utilizada para el análisis, diseño, implementación y documentación de sistemas orientados a objetos.

El Proceso Unificado de Desarrollo fue creado, y concebido debido a la experiencia alcanzada, por el mismo grupo de expertos que crearon UML: Ivar Jacobson, Grady

Booch y James Rumbaugh en el año 1998[33]–[35].

El objetivo de esta metodología es guiar el proceso de software hacia un producto de alta calidad y que cumpla con los requerimientos de los usuarios dentro de una planificación y presupuesto establecidos. Esta metodología utiliza UML como lenguaje de modelado[6].

Los autores de RUP destacan que el proceso de software propuesto por RUP tiene tres características esenciales: está dirigido por los Casos de Uso, está centrado en la arquitectura, y es iterativo e incremental.

El proceso puede ser descrito en dos dimensiones o ejes:

Eje horizontal: Representa el tiempo y es considerado el eje de los aspectos dinámicos del proceso. Indica las características del ciclo de vida del proceso expresado en términos de fases, iteraciones e hitos. Se puede observar en la Figura 14 que RUP consta de cuatro fases: Inicio, Elaboración, Construcción y Transición. Cada fase se subdivide a la vez en iteraciones.

Eje vertical: Representa los aspectos estáticos del proceso. Describe el proceso en términos de componentes de proceso, disciplinas, flujos de trabajo, actividades, artefactos y roles.



Figura 14. Estructura de RUP

**1.6.2 Fundamentación del lenguaje y software utilizado**

**1.6.2.1 Paradigma de la programación orientada a objetos**

La programación orientada a objetos (OOP) es un modo de desarrollar software describiendo problemas mediante el uso de elementos u objetos desde el espacio del problema y no mediante un conjunto de pasos secuenciales que se ejecutarán en el ordenador[36]–[38].

La programación orientada a objetos es más que una simple colección de lenguajes de programación nuevos. La programación orientada a objetos es una nueva forma de pensar acerca de lo que significa computar, acerca de cómo podemos estructurar la información dentro de un computador[39].

Un buen diseño conlleva a componentes reutilizables, extensibles y sostenibles. Estos componentes son lo bastante flexibles como para controlar los cambios que se puedan producir en el entorno a lo largo del tiempo, dado que el trabajo principal de estos objetos es precisamente intercambiar mensajes entre sí[36]. Para ello emplea técnicas como las que se expresan a continuación: herencia, modularidad, polimorfismo, encapsulamiento que permiten expresar un programa como un conjunto de objetos, que colaboran entre ellos para realizar tareas. Esto permite hacer los programas y módulos más fáciles de escribir, mantener y reutilizar[6].

**1.6.2.2 Lenguaje de programación Java**

Java es un lenguaje de programación orientado a objeto[40]–[43], desarrollado por un equipo de Sun Microsystems dirigido por James Gosling en el año 1995. Según su desarrollador Java es simple, orientado a objeto, distribuido, robusto, seguro, de arquitectura neutral, portable, multiplataforma y dinámico. Se ha convertido en un lenguaje muy popular, muchas aplicaciones web son desarrolladas con este lenguaje, aplicaciones de escritorio y de teléfonos celulares con sistema operativo Android, incluso este sistema operativo ha sido desarrollado usando Java[44].

Pero quizá lo que más guste a los programadores son un par de aspectos que le hacen muy cómodo y agradable de usar para programar, por ello se ha escogido como el lenguaje que permita desarrollar el software propuesto:[45]

* La sencillez y elegancia de cómo se escriben los programas en Java. A ello se une que es un lenguaje orientado a objetos que evita muchas preocupaciones a los programadores. En el proceso de compilación se realizan multitud de comprobaciones que permiten eliminar muchos posibles errores posteriores.
* Las bibliotecas ya definidas que proporciona el lenguaje y que el programador puede utilizar sin tener que hacerlas de nuevo.

Java incorpora nuevas tareas como un recolector de elementos y elimina aspectos de C++ confusos y muy poco utilizados como la sobrecarga de operadores. Otro aspecto de la simplicidad de Java es que nada es realmente nuevo. Si observa el conjunto de funciones de Java junto a la historia de la informática, descubrirás que todo procede de algún otro lugar[36].

* Las clases proceden de C++ y Smalltalk, aunque en Java están limitadas a una sola herencia de implementación.
* Las interfaces proceden de Objective-C y proporcionan a Java herencia de interfaces múltiple.
* Los paquetes creados en Modula agregaron a Java el espacio de nombre jerárquico y las unidades de desarrollo lógicas.
* La concurrencia estaba en Mesa y en Java trajo la compatibilidad con multitarea.
* El control de excepciones procedía de Modula-3 y agregó a Java métodos que declaraban lo que producían.
* Los vínculos dinámicos y la recogida automática de elementos no utilizados procedía de Lisp, lo que le concedió a Java la capacidad para cargar clases adicionales a medida que se necesitan y liberar memoria cuando no es necesaria.

Java incluye una biblioteca muy extensa (árbol de clases) que provee funcionalidad para casi todo lo que el programador pueda necesitar. Esto abarca desde manejo de cadenas de caracteres (strings) hasta Sockets (redes, comunicaciones) e interfaz gráfica[46].

**1.6.2.3 NetBeans IDE 8.0 como herramienta para el desarrollo del software**

NetBeans es un entorno de desarrollo integrado libre, hecho principalmente para el lenguaje de programación Java. Existe además un número importante de módulos para extenderlo. NetBeans IDE es un producto libre y gratuito sin restricciones de uso.

NetBeans es un proyecto de código abierto de gran éxito con una gran base de usuarios, una comunidad en constante crecimiento, y con cerca de 100 socios en todo el mundo. Sun MicroSystems fundó el proyecto de código abierto NetBeans en junio de 2000 y continúa siendo el patrocinador principal de los proyectos (Actualmente Sun Microsystems es administrado por Oracle Corporation).

## 1.7 – Conclusiones

El empleo de un software para optimizar el proceso de corte de materiales en dos dimensiones que integre la generación de las variantes de corte primaria y de sustitución, busque dependencias entre ellas reduciendo la cantidad de variables, conforme el modelo matemático de programación lineal en enteros, encuentre la solución óptima del modelo e interprete los resultados de forma comprensible para el usuario es una alternativa muy buena cuando se quieren optimizar los desperdicios en industrias donde se realicen dichos procesos.

El algoritmo diseñado e implementado para generar variantes de corte en dos dimensiones es capaz de encontrar un gran número de ellas, lo que hace posible la obtención de una muy buena solución óptima.

El empleo de NetBeans IDE junto al lenguaje de programación Java -por su sencillez, las riquezas de sus bibliotecas, su seguridad, entre otras ventajas-, han sido escogidos para el desarrollo de la presente investigación. La robustez de este lenguaje ha hecho que alcance una popularidad tremenda, y es mucha la ayuda y documentación que se puede encontrar sobre él.

# 2 – Análisis y diseño de la propuesta de solución

## 2.1 – Introducción

Cuando se trabaja en la construcción de un producto o sistema, es importante ejecutar una serie de pasos predecibles —el mapa de carreteras que lo ayuda a obtener a tiempo un resultado de alta calidad—. El mapa que se sigue se llama “proceso del software”[47].

El modelado es esencial en la construcción de software para comunicar la estructura de un sistema complejo, especificar el comportamiento deseado del sistema, comprender mejor lo que estamos construyendo, descubrir oportunidades de simplificación y reutilización. Un modelo proporciona "los planos" de un sistema y puede ser más o menos detallado, en función de los elementos que sean relevantes en cada momento[48]. Es por ello que este capítulo se dedica a la modelación del sistema utilizando la metodología brindada por el Proceso Unificado de Desarrollo de Software (RUP), la cual se encuentra basada en el Lenguaje Unificado de Modelación (UML) y se presenta como el resultado de más de 40 años de experiencia en el desarrollo de sistemas[6].

Con el objetivo de que se comprenda el proceso y el entorno donde se origina la investigación se presentará lo siguiente:

* Las entidades y los conceptos principales presentes en el entorno en el que funcionará el sistema propuesto.
* El diagrama de clases del modelo de objetos del dominio.
* Las reglas del negocio a considerar.
* Descripción detallada del sistema propuesto.
* La captura de los requisitos funcionales y no funcionales.
* Los actores y los casos de uso asociados a ellos.
* La descripción detallada de cada caso de uso.

Además se expone el diseño del sistema a partir de la solución propuesta que satisface las necesidades del cliente, esta se describe mediante:

* El diagrama de clases del diseño.
* El diseño de los archivos generados por el sistema.
* El diagrama de implementación.
* Los principios de diseño empleados.
* El tratamiento de errores.

## 2.2 – Modelo del dominio

Su utilidad radica en ser una forma de “inspiración” para el diseño de los objetos software. Es entrada para muchos de los artefactos que se construyen en un proceso software. Un modelo de dominio muestra las clases conceptuales significativas en un dominio del problema. Se centra en las abstracciones relevantes, vocabulario del dominio e información del dominio. Es el artefacto clave del análisis orientado a objetos[49].

### 2.2.1 – Definición de las entidades y los conceptos principales

El **plan de corte** es la solicitud que presenta la demanda de **piezas** y la disponibilidad de **materias primas**. El **operario** encargado de realizar los cortes a las materias primas para obtener las piezas demandadas. El **desperdicio** resultado de la realización de los cortes.

### 2.2.2 – Reglas del negocio a considerar

Aquí son presentadas las principales reglas que deben cumplirse para que el negocio funcione de forma fluida.

* El plan de corte es fundamental para la ejecución de los cortes.
* La disponibilidad de materias primas debe satisfacer la demanda de piezas.
* El área de las piezas debe ser inferior a la de la materia prima.

### 2.2.3 – Representación del modelo del dominio

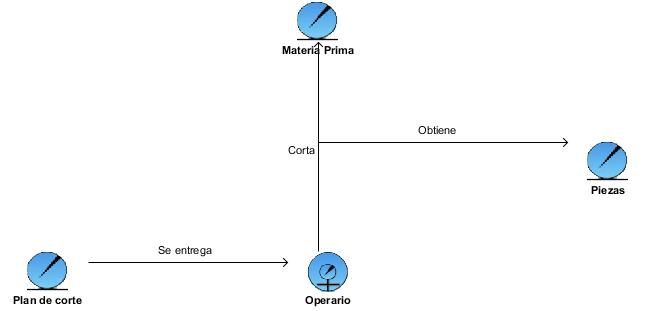


Figura 15. Diagramas del modelo de objetos del dominio

## 2.3 – Requisitos

### 2.3.1 – Descripción del sistema propuesto

### 2.3.2 – Concepción general del sistema

El sistema informático propuesto hará uso de varios módulos que forman parte del software OPTIVACORTUNI, entre los que se encuentran el de optimización y el de búsqueda de dependencias.

El módulo de optimización contiene la programación del algoritmo Branch and Bound que permite obtener las solución óptima en los problemas de Programación Lineal en Enteros.

El módulo de búsqueda de dependencias permitirá reducir el número de variables en el modelo sin que cambie el sentido del mismo.

Como propio el sistema dispondrá de una interface que permita que el usuario pueda ingresar el plan de corte, para ello debe agregar las materias primas con que cuenta y las piezas que desea obtener. Contará con el módulo de generación de variantes de corte primarias y de sustitución en dos dimensiones, las cuales son empleadas en el módulo de búsqueda de dependencias.

También contará con el módulo de la interpretación de la solución, que hará posible que el usuario pueda observar de forma gráfica como deben ser ejecutados los cortes que permiten obtener las piezas demandadas por el mismo.

### 2.3.3 – Requerimientos funcionales

A continuación se presentan los requisitos funcionales del sistema.

1. Insertar Pieza
2. Eliminar Pieza
3. Insertar Materia Prima
4. Eliminar Materia Prima
5. Generar variantes de corte
6. Mostrar interpretación de resultados.

### 2.3.4 – Requerimientos no funcionales

Los requerimientos o requisitos no funcionales también conocidos como “requisitos adicionales” describen propiedades o cualidades que el producto debe tener tales como: restricciones del entorno o de la implementación, rendimiento, dependencias de la plataforma, facilidad de mantenimiento, extensibilidad y fiabilidad. Se refieren a las características que hacen al producto atractivo, usable, rápido o confiable y no a las funciones específicas que entrega el sistema[6].

A continuación se presentan los requisitos no funcionales del sistema.

* Requerimientos de apariencia o interfaz externa.
* La interface hará uso de los estándares de ventanas empleados en el sistema operativo Windows y que está ampliamente divulgado en el mundo por su fácil comprensión y empleo por parte de los usuarios.
* Requerimientos de uso.
* El sistema podrá ser empleado por cualquier usuario que tenga conocimiento básico de los componentes que se emplean en las ventanas tradicionales de Windows.
* Requerimientos de rendimiento.
* El algoritmo de generación de variantes de corte en dos dimensiones hace uso de gran cantidad de recurso, por lo que el tiempo de respuesta es imposible de predecir con exactitud, y estará en correspondencia con los datos insertados.
* Requerimientos de soporte.
* El sistema estará diseñado para que pueda ser perfeccionado y modificado en el futuro.
* Requerimiento de software.
* Se debe tener instalado la versión 1.8 o superior de la Máquina Virtual de Java
* Versión de Windows XP o superior.
* Requerimiento de hardware.
* Para la instalación del sistema se necesitará una computadora con las siguientes características como mínimo.
* Procesador de velocidad 2.60 GHz
* 4 GB de RAM
* 1 GB de HDD libre
* Requerimientos de Seguridad
* Se deberá garantizar los tres pilares fundamentales de la seguridad informática que son: disponibilidad, confidencialidad e integridad de los datos.

### 2.3.5 – Modelo de casos de uso del sistema

El modelo de casos de uso del sistema sirve de intérprete entre los desarrolladores y los clientes. Permite que los desarrolladores de software y los clientes lleguen a un acuerdo sobre los requisitos. El modelo de casos de uso sirve como acuerdo entre clientes y desarrolladores, y proporciona la entrada fundamental para el análisis, el diseño y las pruebas[50].

### 2.3.6 – Actores del sistema

|  |  |
| --- | --- |
| **Nombre del actor** | **Descripción** |
| **Operario** | **Es el usuario que utiliza el sistema y tiene acceso a todas sus funcionalidades.** |

Tabla 1. Definición de actores del sistema a automatizar

### 2.3.7 – Diagramas de casos de uso del sistema

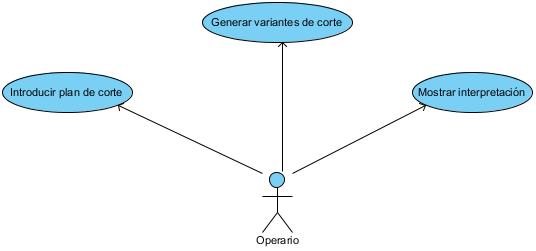


Figura 16. Diagrama del caso de uso del sistema

### 2.3.8 – Descripción de los casos de uso del sistema

|  |  |
| --- | --- |
| **Caso de uso** | Introducir plan de corte |
| **Actores** | Operario |
| **Propósito** | Introducir el plan de corte |
| **Resumen**  El caso de uso se inicia cuando el Operario desea ingresar el plan de corte. Después el sistema solicita introducir las longitudes de las materias primas y las piezas, así como la disponibilidad de las primeras y la demanda de las segundas. El caso de uso concluye una vez insertado el plan de corte por parte del operario y comprobado que los datos son válidos. | |
| **Referencias** | R1, R2, R3, R4 |
| **Precondiciones** | El sistema debe haberse iniciado. |
| **Post-condiciones** | El sistema contiene el plan de corte |
| **Prototipo** | Anexo 1 |

Tabla 2. Descripción del caso de uso de sistema < Introducir plan de corte >

|  |  |
| --- | --- |
| **Caso de uso** | Generar variante de corte |
| **Actores** | Operario |
| **Propósito** | Generar las variantes de corte primarias y de sustitución. |
| **Resumen**  El caso de uso se inicia cuando el operario desea generar las variantes de corte primarias y de sustitución. Esta acción se lleva a cabo una sola vez, solo si existen cambios en las longitudes de las materias primas o las piezas, o en la disponibilidad de las primeras o demanda de las segundas debe realizarse nuevamente esta acción. El caso de uso termina una vez generadas todas las variantes de corte primarias y de sustitución. | |
| **Referencias** | R5 |
| **Precondiciones** | El sistema debe haberse iniciado y haberse introducido el plan de corte. |
| **Post-condiciones** | El sistema contiene las variantes de corte primarias y de sustitución. |
| **Prototipo** | Anexo 2 |

Tabla 3. Descripción del caso de uso de sistema < Generar variante de corte >

|  |  |
| --- | --- |
| **Caso de uso** | Mostrar interpretación |
| **Actores** | Operario |
| **Propósito** | Mostrar de forma gráfica las formas en que deben ser realizados los cortes. |
| **Resumen**  El caso de uso se inicia cuando el operario desea visualizar de forma gráfica la forma en que se deben ejecutar los cortes para satisfacer su demanda. Luego, se pasa a buscar la solución óptima. El caso de uso finaliza con la visualización de la solución óptima del problema. | |
| **Referencias** | R6 |
| **Precondiciones** | El sistema debe haberse iniciado y haberse introducido el plan de corte. |
| **Post-condiciones** | - |
| **Prototipo** | Anexo 3 |

Tabla 4. Descripción del caso de uso de sistema < Mostrar interpretación >

## 2.4 – Construcción de la solución propuesta

### 2.4.1 – Diagrama de clases del diseño

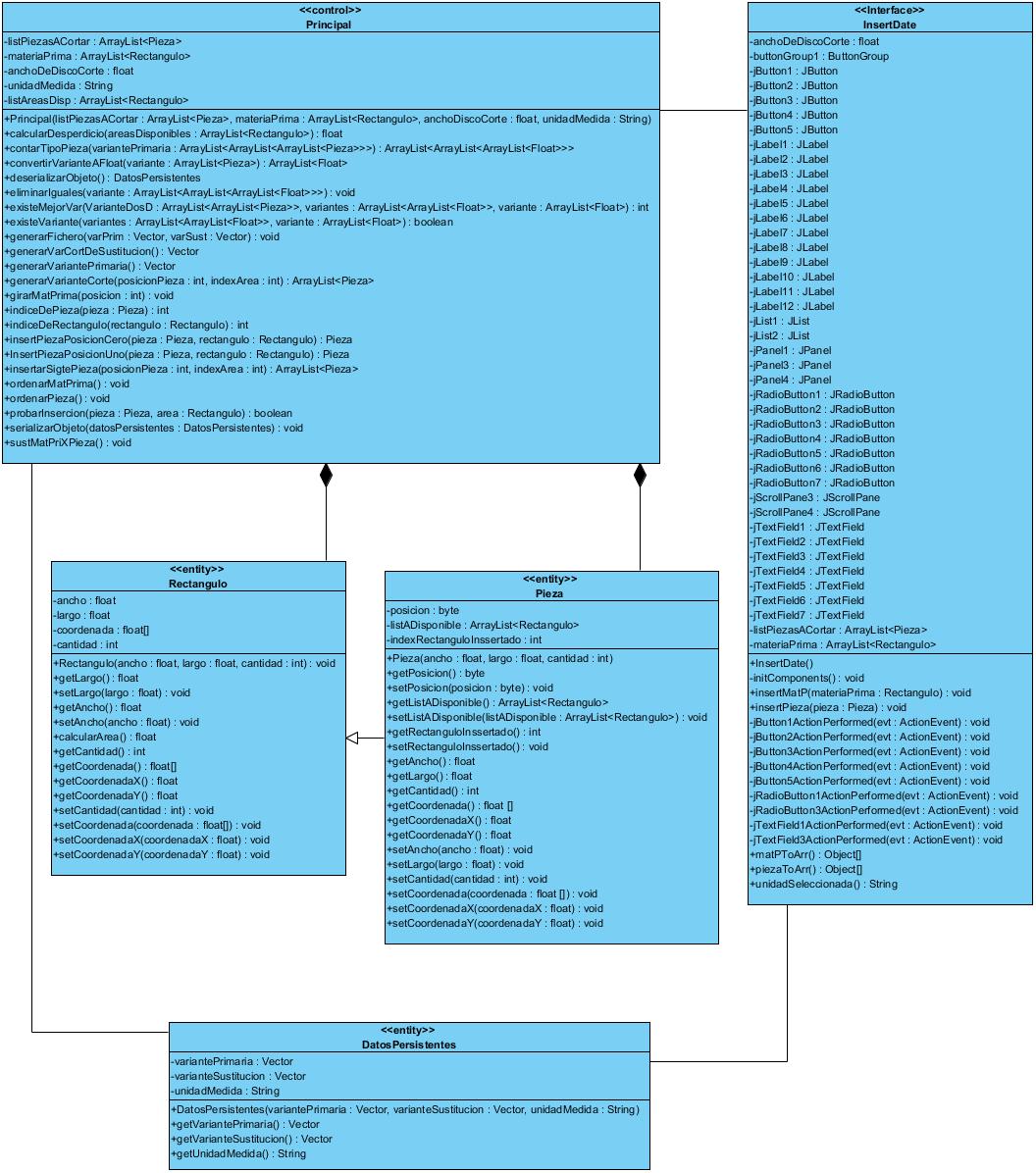
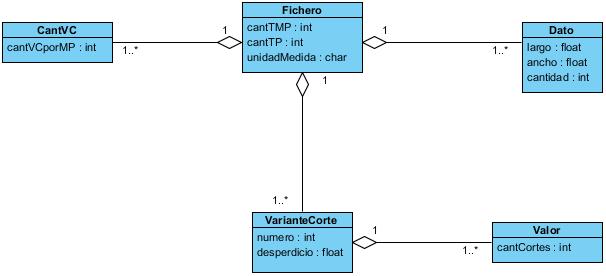


Figura 17. Diagramas de clases

### 2.4.2 – Diseño de archivo

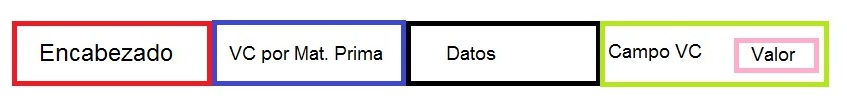
### 2.4.3 – Diagrama de archivo

Representación gráfica del diagrama de clases persistentes.

****

**Figura 18. Diagramas de clases persistentes**

### 2.4.4 – Estructura del archivo

****

**Figura 19. Estructura del archivo**

A continuación se describe cada campo del archivo:

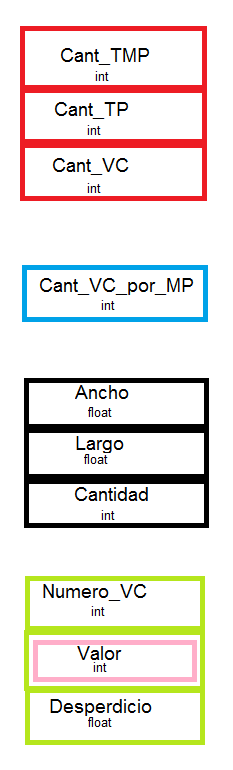


Figura 20. Descripción por campos

Descripción de los campos del encabezado:

Cant\_TPM: cantidad de tipos de materias primas con que se dispone.

Cant\_TP: cantidad de tipos de piezas con que se dispone.

Cant\_VC: cantidad de variantes de corte.

Descripción del campo VC por Mat. Prima.

Cant\_VC\_por\_MP: cantidad de variantes de corte por tipo de materia prima.

Descripcion de los campos de Datos.

Ancho: ancho de la materia prima o pieza.

Largo: largo de la materia prima o pieza

Cantidad: cantidad de materias primas o piezas.

Descripción de los campos Campo VC.

Número\_VC: Número que representa una variante del total.

Valor: Valores de la variante de corte que se interpreta como la cantidad de cada pieza a obtener.

Desperdicio: Sobrante de materia prima al aplicar la variante de corte.

### 2.4.5 – Diagrama de implementación

Representación gráfica del diagrama de implementación.

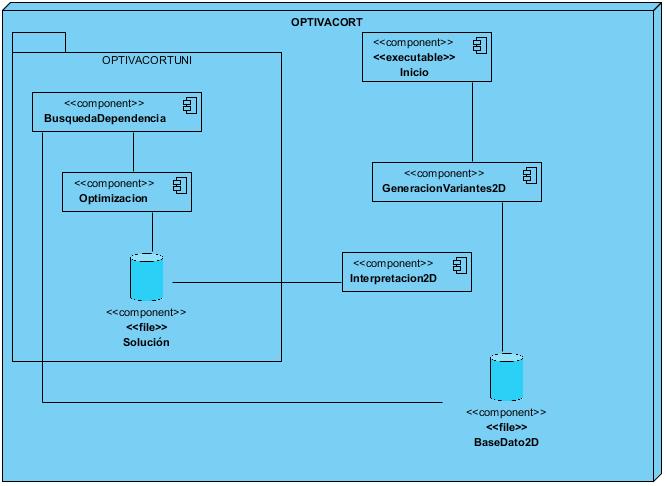


Figura 21. Diagrama de implementación

### 2.4.6 – Principios de diseño

La fase de diseño es la que le sigue a la de análisis y la que se antepone a la de implementación[6]. El diseño del software comienza una vez que se han analizado y modelado los requerimientos, es la última acción de la ingeniería de software dentro de la actividad de modelado y prepara la etapa de construcción (generación y prueba de código)[47]. En esta fase se obtiene una solución lógica sustentada en el paradigma de la programación orientada a objetos[6]. Se agrupa el conjunto de principios, conceptos y prácticas que llevan al desarrollo de un sistema o producto de alta calidad. Los principios de diseño establecen una filosofía general que guía el trabajo de diseño que debe ejecutarse[47]. Se modela el sistema y se encuentra su forma (incluida la arquitectura) para que soporte todos los requisitos incluyendo los requisitos no funcionales y otras restricciones que se le suponen[50].

### 2.4.7 – Estándares en la interfaz de la aplicación

Para el diseño de la interface se han estudiado otros software relacionados con el tema, los que han servido de base para el diseño del mismo.

Se ha tenido en cuenta los estándares establecido y ampliamente difundidos por el sistema operativo Windows, la interface propuesta cuenta con ventanas y botones que permitan la fácil interacción entre el usuario y el sistema.

Se eligieron colores tenues que resultan agradables a la vista de los que lo empleen.

### 2.4.8 – Tratamiento de errores

El tratamiento de errores es esencial en cualquier sistema, para obtener resultados correctos es necesario que los datos introducidos sean válidos. Para ello se debe validar que la introducción de un plan de corte cuente con todos los datos necesarios y correctos que permitan la generación de las variantes de corte. Cuando exista algún dato erróneo debe mostrar un mensaje alertando dicha falla.

## 2.5 – Conclusiones

Una vez concluido el capítulo se tiene una idea clara y detallada del entorno donde se desarrolla la investigación.

Con la metodología RUP como referente se ha podido documentar todo lo concerniente al desarrollo del sistema propuesto, se han utilizado varios diagramas, entre los que se encuentran: modelo del dominio, que explica de forma clara el funcionamiento del negocio, y diagrama de casos de uso del sistema, que muestra los actores con los casos de uso asociados a ellos.

En la fase de diseño se han definido los diagramas de clases del diseño, el diagrama de clases persistentes y el diagrama de implementación, los que sirven con referentes para la posterior fase de construcción.

# 3 – Estudio de factibilidad y validación de la solución

## 3.1 – Introducción

En este capítulo se lleva a cabo el estudio de factibilidad, para ello se ha escogido el método que emplea los Casos de Uso para realizar los análisis que determine el tiempo estimado de desarrollo del sistema, el esfuerzo humano así como los costos y beneficios tangibles e intangibles. Con este método se puede determinar si es factible o no que se dedique tiempo para construir el sistema.

## 3.2 – Estudio de factibilidad

### 3.2.1 – Planificación por casos de usos

La estimación mediante el análisis de Puntos de Casos de Uso es un método propuesto originalmente por Gustav Karner, y posteriormente refinado por muchos otros autores. Se trata de un método de estimación del tiempo de desarrollo de un proyecto mediante la asignación de "pesos" a un cierto número de factores que lo afectan, para finalmente, contabilizar el tiempo total estimado para el proyecto a partir de esos factores.

### 3.2.1.1 – Cálculo de Puntos de Caso de Uso sin ajustar

El primer paso para la estimación consiste en el cálculo de los Puntos de Casos de Uso sin ajustar. Este valor, se calcula a partir de la siguiente ecuación:

UUCP = UAW + UUCW

Donde:

UUCP: Puntos de Casos de Uso sin ajustar

UAW: Factor de Peso de los Actores sin ajustar:

UUCW: Factor de Peso de los Casos de Uso sin ajustar

Para ello se deben determinar los dos sumandos que se encuentran en la ecuación, proceso que se lleva a cabo a continuación.

**3.2.1.1.1 Factor de Peso de los Actores sin ajustar**

Este valor se calcula mediante un análisis de la cantidad de Actores presentes en el sistema y la complejidad de cada uno de ellos. La complejidad de los Actores se establece teniendo en cuenta en primer lugar si se trata de una persona o de otro sistema, y en segundo lugar, la forma en la que el actor interactúa con el sistema.

|  |  |  |
| --- | --- | --- |
| **Actor** | **Tipo** | **Factor de Peso** |
| Operario | Complejo | 3 |

Tabla 5. Factor de Peso de los Actores sin ajustar

UAW=Ʃ (Factor de Peso)i

UAW=3

**3.2.1.1.2 Factor de Peso de los Casos de Uso sin ajustar**

Este valor se calcula mediante un análisis de la cantidad de Casos de Uso presentes en el sistema y la complejidad de cada uno de ellos. La complejidad de los Casos de Uso se establece teniendo en cuenta la cantidad de transacciones efectuadas en el mismo, donde una transacción se entiende como una secuencia de actividades atómica, es decir, se efectúa la secuencia de actividades completa, o no se efectúa ninguna de las actividades de la secuencia.

|  |  |  |
| --- | --- | --- |
| **Casos de Usos** | **Tipo** | **Factor de Peso** |
| Introducir plan de corte | Simple | 5 |
| Generar variante de corte | Simple | 5 |
| Mostrar interpretación | Simple | 5 |

Tabla 6. Factor de Peso de los Casos de Uso sin ajustar

UUCW= Ʃ(Factor de Peso)i

UUCW=5+5+5

UUCW=15

Cuando ya se tienen las dos variables necesarias para calcular los puntos de caso de uso sin ajustar se procede a realizar su cálculo como se muestra a continuación.

UUCP = UAW + UUCW

UUCP = 3+15

UUCP = 18

### 3.2.1.2 – Cálculo de Puntos de Caso de Uso ajustados

Una vez que se tienen los Puntos de Casos de Uso sin ajustar, se debe ajustar éste valor mediante la siguiente ecuación:

UCP = UUCP \* TCF \* EF

Donde:

UCP: Puntos de Casos de Uso ajustados

UUCP: Puntos de Casos de Uso sin ajustar

TCF: Factor de complejidad técnica

EF: Factor de ambiente

**3.2.1.2.1 Factor de complejidad técnica**

Este coeficiente se calcula mediante la cuantificación de un conjunto de factores que determinan la complejidad técnica del sistema. Cada uno de los factores se cuantifica con un valor de 0 a 5, donde 0 significa un aporte irrelevante y 5 un aporte muy importante. En la siguiente tabla se muestra el significado y el peso de cada uno de éstos factores y el valor que se le ha asignado.

|  |  |  |  |
| --- | --- | --- | --- |
| **Factor** | **Descripción** | **Peso** | **Valor** |
| T1 | Sistema distribuido | 2 | 0 |
| T2 | Objetos de performance o tiempo de respuesta | 1 | 3 |
| T3 | Eficiencia del usuario final | 1 | 4 |
| T4 | Procesamiento interno complejo | 1 | 5 |
| T5 | El código debe ser reutilizable | 1 | 5 |
| T6 | Facilidad de instalación | 0.5 | 5 |
| T7 | Facilidad de uso | 0.5 | 5 |
| T8 | Portabilidad | 2 | 3 |
| T9 | Facilidad de cambio | 1 | 5 |
| T10 | Concurrencia | 1 | 0 |
| T11 | Incluye objetivos especiales de seguridad | 1 | 3 |
| T12 | Provee acceso directo a terceras partes | 1 | 0 |
| T13 | Se requiere facilidades especiales de entrenamiento a usuarios | 1 | 0 |

Tabla 7. Factor de complejidad técnica

El Factor de complejidad técnica se calcula mediante la siguiente ecuación:

TCF=0.6+0.01\*Ʃ(Pesoi\*Valori)

TCF=0.6+0.01\*(38)

TCF=0.98

**3.2.1.2.2 Factor de ambiente**

Las habilidades y el entrenamiento del grupo involucrado en el desarrollo tienen un gran impacto en las estimaciones de tiempo. Estos factores son los que se contemplan en el cálculo del Factor de ambiente. El cálculo del mismo es similar al cálculo del Factor de complejidad técnica, es decir, se trata de un conjunto de factores que se cuantifican con valores de 0 a 5.

En la siguiente tabla se muestra el significado, el peso y el valor asignado de cada uno de éstos factores.

|  |  |  |  |
| --- | --- | --- | --- |
| **Factor** | **Descripción** | **Peso** | **Valor** |
| E1 | Familiaridad con el modelo de proyecto utilizado | 1.5 | 3 |
| E2 | Experiencia en la aplicación | 0.5 | 3 |
| E3 | Experiencia en orientación a objetos | 1 | 4 |
| E4 | Capacidad del analista líder | 0.5 | 3 |
| E5 | Motivación | 1 | 5 |
| E6 | Estabilidad de los requerimientos | 2 | 5 |
| E7 | Personal part-time | -1 | 3 |
| E8 | Dificultad del lenguaje de programación | -1 | 1 |

Tabla 8. Factor de ambiente

El Factor de ambiente se calcula mediante la siguiente ecuación:

EF=1.4-0.03\*Ʃ(Pesoi\*Valori)

EF=1.4-0.03\*(22.5)

EF=0.725

Ya se puede determinar el valor de los puntos de caso de uso ajustados.

UCP = UUCP \* TCF \* EF

UCP = 18\*0.98\*0.725

UCP = 12.8

### 3.2.2 – Determinación de los costos

Karner originalmente sugirió que cada Punto de Casos de Uso requiere 20 horas-hombre. Posteriormente, surgieron otros refinamientos que proponen una granularidad algo más fina, según el siguiente criterio:

* Se contabilizan cuántos factores de los que afectan al Factor de ambiente están por debajo del valor medio (3), para los factores E1 a E6. En este caso es 0
* Se contabilizan cuántos factores de los que afectan al Factor de ambiente están por encima del valor medio (3), para los factores E7 y E8. En nuestro caso es 0

**Factor de conversión**

- Como el total es 0, se utiliza el factor de conversión 20 horas-hombre/Punto de Casos de Uso, es decir, un Punto de Caso de Uso toma 20 horas-hombre.

FC=20 horas-hombre/Punto de Casos de Uso

El esfuerzo en horas-hombre viene dado por:

E = UCP \* CF

E= 12.8 \* 20

E=256

Donde:

UCP: Puntos de Casos de Uso ajustados

CF: factor de conversión

Se debe tener en cuenta que éste método proporciona una estimación del esfuerzo en horas-hombre contemplando sólo el desarrollo de la funcionalidad especificada en los casos de uso.

Finalmente, para una estimación más completa de la duración total del proyecto, hay que agregar a la estimación del esfuerzo obtenida por los Puntos de Casos de Uso, las estimaciones de esfuerzo de las demás actividades relacionadas con el desarrollo de software.

Para ello se puede tener en cuenta el siguiente criterio, que estadísticamente se considera aceptable. El criterio plantea la distribución del esfuerzo entre las diferentes actividades de un proyecto, según la siguiente aproximación:

|  |  |  |  |
| --- | --- | --- | --- |
| **Actividad** | **Porcentaje** | **Horas/Hombre** | **Días/Hombre** |
| Análisis | 10.00% | 64 | 8 |
| Diseño | 20.00% | 128 | 16 |
| Programación | 40.00% | 256 | 32 |
| Pruebas | 15.00% | 96 | 12 |
| Sobrecargas | 15.00% | 96 | 12 |
| Total | 100% | 640 | 80 |

Tabla 9. Tiempo de duración del proyecto

El tiempo total aproximado necesario para el desarrollo del software es de tres meses y diez días si suponemos que se trabajan 24 días por mes.

Obviamente, estos valores no son absolutos sino que pueden variar de acuerdo a las características de la organización y del proyecto.

**Costo**

Costo = tiempo \* Salario Promedio

Costo = 3.33\*450

Costo = 1499

Los costos en los que se incurriría de desarrollarse el sistema serían:

|  |  |
| --- | --- |
| **Cálculo de:** | **Valor** |
| Esfuerzo | 256 horas/hombre |
| Tiempo de desarrollo | 3.3 meses |
| Cantidad de hombres | 1 |
| Costo | 1499 |
| Salario medio | 450 |

Tabla 10. Costos totales

### 3.2.3– Beneficios tangibles e intangibles

El software que se propone en la presente investigación es capaz de generar beneficios tangibles e intangibles de ser utilizado en cualquier industria o fábrica donde existan problemas como los que se describen con anterioridad.

Los beneficios intangibles son aquellos que resultan difíciles o prácticamente imposibles de cuantificar en unidades monetarias, y entre ellos podemos encontrar la satisfacción que sienten los operarios al contar con una herramienta que facilita la realización de los cortes a las materias primas; la eliminación de un fenómeno lamentablemente tan común en la sociedad cubana “*el robo*”, debido a que el sistema que se propone establece la forma en que se deben realizar los corte y los desperdicios generados por los mismos, por lo que no hay espacio para que alguien se apropie de ellos.

Los beneficios tangibles son los más significativos, pues de ser utilizado el software propuesto, se reducen de forma considerable los gastos innecesarios debido a la optimización de los desperdicios.

### 3.2.4 – Análisis de costos y beneficios

Desarrollar cualquier producto informático tiene asociado un costo determinado, el cual debe ser comparado con los beneficios para determinar si es factible o no la construcción e implementación del mismo.

En la presente investigación en particular los costos son extremadamente inferiores a los beneficios generados, pues los aproximadamente 1500 pesos que se necesitan para desarrollar dicho software representan una fracción muy pequeña en relación con los beneficios que puede ocasionar su uso. De ser empleado el producto de software que se propone se pueden ahorrar miles, incluso millones de pesos en un año debido a la optimización de los desperdicios generados en el proceso de corte.

## 3.3 – Validación de la solución

A continuación, se presenta la validación de la solución propuesta a través de la comparación de los desperdicios obtenidos al aplicar diferentes planes de corte utilizando el programa Optivacomulti y el programa profesional Corte Certo 2D. Se utiliza un enfoque inferencial paramétrico utilizando para ello una muestra de tamaño 22, se verifica la condición de normalidad y se realiza la prueba T de comparación entre dos muestras relacionadas. El experimento fue diseñado en dos etapas: en la primera etapa se seleccionaron de forma aleatoria los planes de corte que debían ser procesados y en la segunda etapa se introdujo los planes de corte en cada uno de los software, estudiando fundamentalmente el desperdicio final una vez obtenida la solución. Los resultados se muestran a continuación:

|  |  |  |
| --- | --- | --- |
| Plan de corte | Desperdicio, dado en m2, al utilizar el software Optivacomulti | Desperdicio, dado en m2, al utilizar el software Corte Certo 2D |
| 1 | 653,50 | 653,69 |
| 2 | 797,50 | 797,70 |
| 3 | 1244,50 | 2015,01 |
| 4 | 1187,00 | 1957,84 |
| 5 | 1269,00 | 2039,28 |
| 6 | 1556,00 | 2326,76 |
| 7 | 2078,50 | 2848,69 |
| 8 | 1447,00 | 2216,96 |
| 9 | 1569,50 | 1570,12 |
| 10 | 1954,00 | 2723,79 |
| 11 | 663,00 | 1652,81 |

Tabla 11. Desperdicios de planes de corte

Las variables, para el procesamiento estadístico, se definen de la siguiente forma:

x: Desperdicio obtenido, dado en m2, al procesar un plan de corte utilizando el software Optivacomulti.

y: Desperdicio obtenido, dado en m2, al procesar un plan de corte utilizando el software Corte Certo 2D.

Se comprueba si las muestras provienen de poblaciones con distribución normal aplicando la prueba Kolmogorov-Smirnov. Las hipótesis se contrastan de las siguientes formas:

H0: La población de la cual proviene x (o y) sigue una distribución normal.

H1: La población de la cual proviene x (o y) no sigue una distribución normal.

Se utiliza un nivel de significación de 0,05 y se desea probar el incumplimiento de la región crítica para aceptar H0 y concluir que ambas muestras provienen de poblaciones que siguen distribución normal.

Los resultados de la prueba se muestran en la tabla 12. En ambas muestras la significación asintótica es mayor que 0,05 por lo que, con un 95% de confianza, no hay razones para dudar que siguen distribución normal.

|  |  |  |  |
| --- | --- | --- | --- |
| **Prueba de Kolmogorov-Smirnov para una muestra** | | | |
|  | | Desperdicio al aplicar un plan de corte utilizando Optivacomulti | Desperdicio al aplicar un plan de corte utilizando Corte Certo 2D |
| N | | 11 | 11 |
| Parámetros normalesa,b | Media | 1310,8636 | 1891,1500 |
| Desviación típica | 478,04843 | 695,31419 |
| Diferencias más extremas | Absoluta | ,131 | ,175 |
| Positiva | ,131 | ,124 |
| Negativa | -,125 | -,175 |
| Z de Kolmogorov-Smirnov | | ,435 | ,579 |
| Sig. asintót. (bilateral) | | ,991 | ,891 |
| a. La distribución de contraste es la Normal. | | | |
| b. Se han calculado a partir de los datos. | | | |

**Tabla 12. Prueba de bondad de ajuste (Tomado de SPSS)**

Basados en los resultados anteriores se puede realizar una prueba de comparación de medias relacionadas tomando como unidad muestral el plan de corte (prueba T para muestras pareadas). El planteamiento de las hipótesis sería el siguiente:

H0: El promedio de x no es menor que el promedio de y.

H1: El promedio de x es menor que el promedio de y.

Se utiliza un nivel de significación de 0,05 y se desea probar el cumplimiento de la región crítica para aceptar H1 y concluir que como promedio los desperdicios obtenidos utilizando el software Optivacomulti son menores que los obtenidos utilizando Corte Certo 2D.

Los resultados de la prueba se muestran en la tabla 13. La significación asintótica es menor que 0,05 por lo que, con un 95% de confianza, no hay razones para dudar que los desperdicios obtenidos utilizando el software Optivacomulti son menores que los obtenidos utilizando Corte Certo 2D.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Prueba de muestras relacionadas** | | | | | | | | | |
|  | | Diferencias relacionadas | | | | | t | gl | Sig. (bil) |
| Media | Desv. típ. | Error típ. de la media | 95% Inter. de conf. para la diferencia | |
| Inferior | Superior |
| Par1  x - y |  | -580,3 | 378,1 | 114 | -834, 3 | -326,3 | -5,1 | 10 | ,000 |

**Tabla 13. Prueba T para muestras pareadas (Tomado de SPSS)**

## 3.4 – Conclusiones

El sistema que se propone trae como consecuencia que se obtengan una serie de beneficios tangibles e intangibles que hacen muy factible la construcción del mismo. Entre ellos se pueden encontrar la contribución a la disminución del desperdicio y el control y organización del proceso de corte de materiales en dos dimensiones.

Con el estudio de factibilidad se ha determinado que se requiere de aproximadamente tres meses y medios y del $1500,00 para la construcción del software por un hombre.

Se ha demostrado estadísticamente mediante una prueba de comparación de medias que los desperdicios generados por el sistema propuesto son menores que los que se generan cuando se emplea un programa profesional reconocido y establecido en el mercado como Corte Certo 2D.

.

# Conclusiones

1. El empleo del sistema informático OPTIVACOMULTI en entornos donde se realicen cortes a los materiales en dos dimensiones puede contribuir positivamente en la disminución de desperdicios, la organización y el control de dicho proceso.
2. El algoritmo diseñado e implementado permite la obtención de un gran número de variantes de corte, lo que lo convierte en un algoritmo robusto para estos tipos de problemas.
3. Se ha probado que los desperdicios generados cuando se emplea el software propuesto son inferiores, como promedio, que los generados por un software profesional.
4. El informe del presente trabajo de diploma se puede convertir en documento de consulta en industrias donde se desarrollen procesos de corte de materiales en dos dimensiones.

# Recomendaciones

1. Utilizar el sistema informático OPTIVACOMULTI (que integra la generación de variantes de corte primarias y de sustitución en dos dimensiones, la búsqueda de dependencias y eliminación de dependientes, la modelación matemática del problema, el procesamiento computacional del modelo matemático y la interpretación de la solución óptima de forma gráfica) en la optimización del proceso de corte en la industrias o entidades cubana que en sus quehacer productivo desarrollen este proceso.
2. Emplear el software propuesto en otras aplicaciones de los problemas de corte en dos dimensiones, como pudiera ser el empaquetado de materiales.
3. Seguir investigando para lograr extender el sistema a problemas en tres dimensiones.

# Referencias bibliográficas

[1] M. Rothe, M. Reyer, y R. Mathar, «Process Optimization for Cutting Steel-Plates.», en *ICORES*, 2017, pp. 27–37.

[2] D. A. Wuttke y H. S. Heese, «Two-dimensional cutting stock problem with sequence dependent setup times», *Eur. J. Oper. Res.*, vol. 265, n.o 1, pp. 303–315, 2018.

[3] J. E. Cevallos, D. S. Mauricio, y G. D. Rodríguez, «Un Algoritmo GRASP de corte guillotina 2D con Agrupamiento y Rotación».

[4] Parreño Torres, Francisco, «Algoritmos heurísticos y exactos para problemas de corte no guillotina en dos dimensiones», Universidad de Valencia, Valencia, 2004.

[5] S. M. Carazo y E. A. Hurtado, «La industria del Cuero y Calzado en el Perú: Innovando para Competir, serie Cadenas Productivas», *MITINCI Lima-Perú*, 1998.

[6] Medina, Juan F., «OPTIVACORTUNI. Sistema informático para resolver problemas de optimización en procesos de corte de materiales”», Carlos Rafael Rodríguez, Cienfuegos, 2008.

[7] L. V. Kantorovich, «Mathematical methods of organizing and planning production», *Manag. Sci.*, vol. 6, n.o 4, pp. 366–422, 1960.

[8] León Rodríguez, Narciso R. de, «Métodos Matemáticos en la Dirección de los Procesos de Corte de Materiales», Carlos Rafael Rodríguez, Cienfuegos, 1996.

[9] K. V. Anand y A. R. Babu, «Heuristic and genetic approach for nesting of two-dimensional rectangular shaped parts with common cutting edge concept for laser cutting and profile blanking processes», *Comput. Ind. Eng.*, vol. 80, pp. 111–124, 2015.

[10] S. Octarina, M. Radiana, y P. B. Bangun, «Implementation of pattern generation algorithm in forming Gilmore and Gomory model for two dimensional cutting stock problem», en *IOP Conference Series: Materials Science and Engineering*, 2018, vol. 300, p. 012021.

[11] S. S. B. Intiyot y C. Jeenanunta, «A column generation technique with multiple sub-problems for 2-dimensional cutting stock problem».

[12] A. Ramos, P. Sánchez, J. M. Ferrer, J. Barquín, y P. Linares, «Modelos matemáticos de optimización», *Publ. Téc.*, vol. 1, 2010.

[13] A. E. Paull y J. R. Walter, «The trim problem: an application of linear programming to the manufacture of newsprint paper», *Econometrica*, vol. 23, n.o 3, p. 336, 1955.

[14] R. W. Metzger, *Elementary mathematical programming*. Wiley, 1958.

[15] S. Eilon, «Optimizing the shearing of steel bars», *J. Mech. Eng. Sci.*, vol. 2, n.o 2, pp. 129–142, 1960.

[16] P. C. Gilmore y R. E. Gomory, «A linear programming approach to the cutting-stock problem», *Oper. Res.*, vol. 9, n.o 6, pp. 849–859, 1961.

[17] P. C. Gilmore y R. E. Gomory, «A linear programming approach to the cutting stock problem—Part II», *Oper. Res.*, vol. 11, n.o 6, pp. 863–888, 1963.

[18] Dyckhoff H., «A typology of cutting and packing problems», *European Journal of Operational Research*, 1990.

[19] Morabito, R. y Morales, A., «A simple and effective recursive procedure for the manufacturer’s pallet loading problem», *Journal of the Operational Research Society*, vol. 49, 1998.

[20] Morabito, R. y Morales, A., «A simple and effective recursive procedure for the manufacturer’s pallet loading problem», *J. Oper. Res. Soc.*, vol. 50, 1998.

[21] M. G. Vieten, «Numerical Solution of Stochastic Control Problems Using the Finite Element Method», PhD Thesis, The University of Wisconsin-Milwaukee, 2018.

[22] M. Bansal y Y. Zhang, «Two-stage stochastic (and distributionally robust) p-order conic mixed integer programs: Tight second stage formulations», 2018.

[23] E. M. E. Mhamdi, R. Guerraoui, L. N. Hoang, y A. Maurer, «Removing Algorithmic Discrimination (With Minimal Individual Error)», *ArXiv Prepr. ArXiv180602510*, 2018.

[24] N. Araya, F. A. Lucay, L. A. Cisternas, y E. D. Galvez, «Design of Desalinated Water Distribution Networks: Complex Topography, Energy Production, and Parallel Pipelines», *Ind. Eng. Chem. Res.*, 2018.

[25] Hillier, Frederick S. y Lieberman, Gerald J., *Introducción a la Investigación de Operaciones*, 9.a ed. México, 2010.

[26] Pilar, Felipe, *Programación Matemática*. Ciudad de La Habana: Empresa Nacional de Producción del Ministerio de Educación Superior, 1983.

[27] D. Avila y R. Sumactika, «Un estudio algorítmico del problema de corte y empaquetado 2D», 2007.

[28] J. Gray y B. Rumpe, *UML customization versus domain-specific languages*. Springer, 2018.

[29] Lerman, Craig, *UML y patrones. Introducción al análisis y diseño orientado a objeto*, vol. 1, 2 vols. La Habana: Félix Varela, 2004.

[30] Morejón Vega, Y, «Sistema Informático para la Gestión de la Información de los Modelos Estadísticos de la Universidad de Cienfuegos.», Universidad de Cienfuegos, Cienfuegos.

[31] M. H. bin Abdul Hamid y N. bin Katmon, «ChefBook’s Mobile Application».

[32] R. F. Razali y N. Idris, «KS Bridal Management System».

[33] S. M. Pauzi y N. Idris, «Inno Biologics Sdn Bhd Production and Management System».

[34] N. K. A. B. Jesmen y N. B. B. Ahmad, «Mobile Homestay Booking System (jHo)».

[35] U. S. M. Jalil y M. A. Alias, «EasyCook Mobile Application».

[36] Zukowski, John, *Programación Java 2*, vol. 1. La Habana: Félix Varela, 2006.

[37] C. Pozrikidis, *Programming and Graphics Introduction to C++*. Springer, 2017.

[38] L. Joseph, «Fundamentals of Python for Robotics Programming», en *Robot Operating System for Absolute Beginners*, Springer, 2018, pp. 95–126.

[39] Budd, Timothy A, *Introducción a la Programación Orientada a Objetos*. La Habana: Editorial de Libros Para la Educación, 2005.

[40] K. Sharan, «Java APIs, Extensions and Libraries».

[41] L. Expressions y K. Sharan, «Java Language».

[42] J. Farrell, «Java Programming, Loose-leaf Version», 2017.

[43] P. Jackson, «Long Term Assessment of Object Strength in a Web Service as Managed by the Garbage Collection in Java Based Services», 2018.

[44] Liang, Y. D, *Introduction to Java programing*, 10.a ed. Pearson Education, 2015.

[45] Sánchez Allende, J, Fernández-Toribio, G. H, Fernández Manjón, B, Moreno Díaz, P, Reinoso Peinado, A. J, y Sánchez-Cortés, R. S, *Programación en Java 2*. España: McGraw-Hill, 2005.

[46] P. Sznajdleder, «Java a fondo», *B. Aires Alfaomega Grupo Ed. Argent.*, 2013.

[47] R. S. Pressman y J. M. Troya, «Ingeniería del software», 1988.

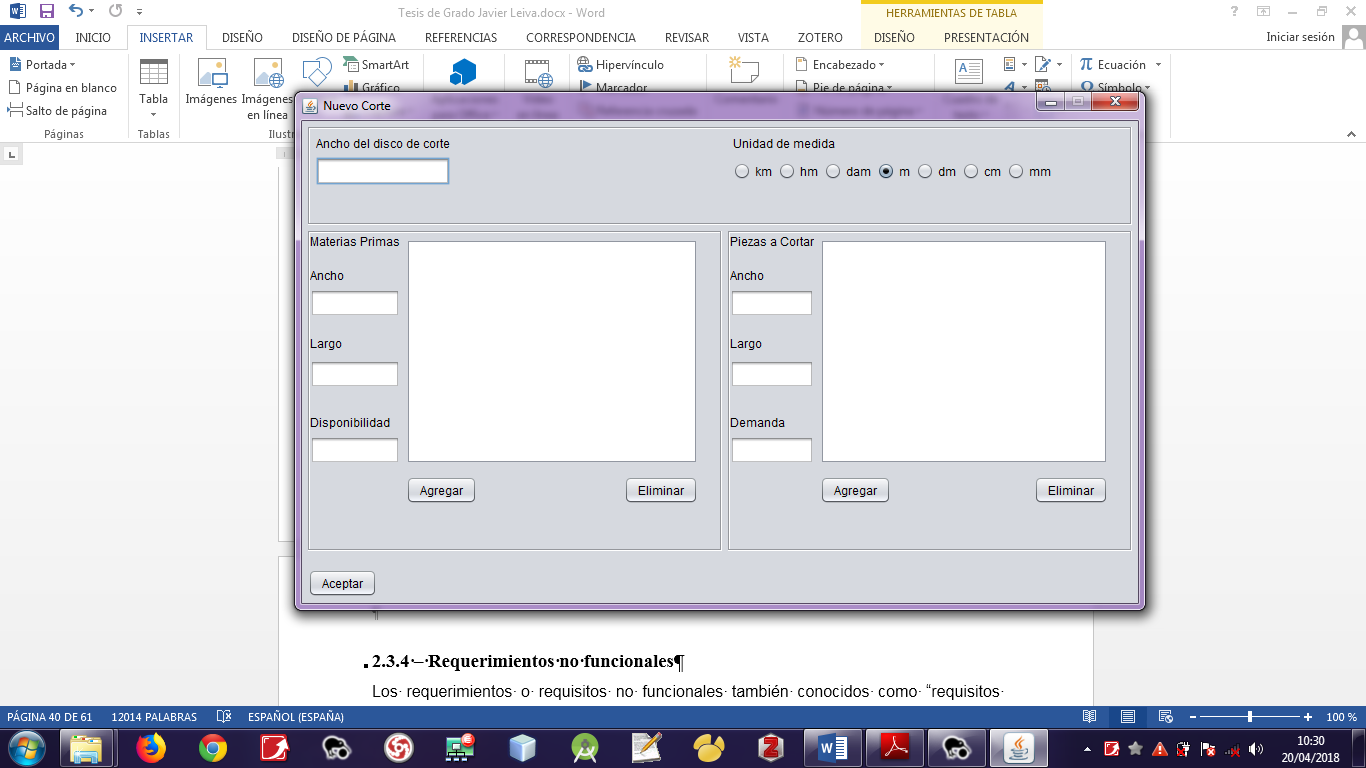
[48] G. Booch, J. Rumbaugh, I. Jacobson, J. S. Martínez, y J. J. G. Molina, *El lenguaje unificado de modelado*, vol. 1. Addison Wesley Madrid, 1999.

[49] A. García-Holgado y F. J. García-Peñalvo, «Modelo de dominio», 2018.

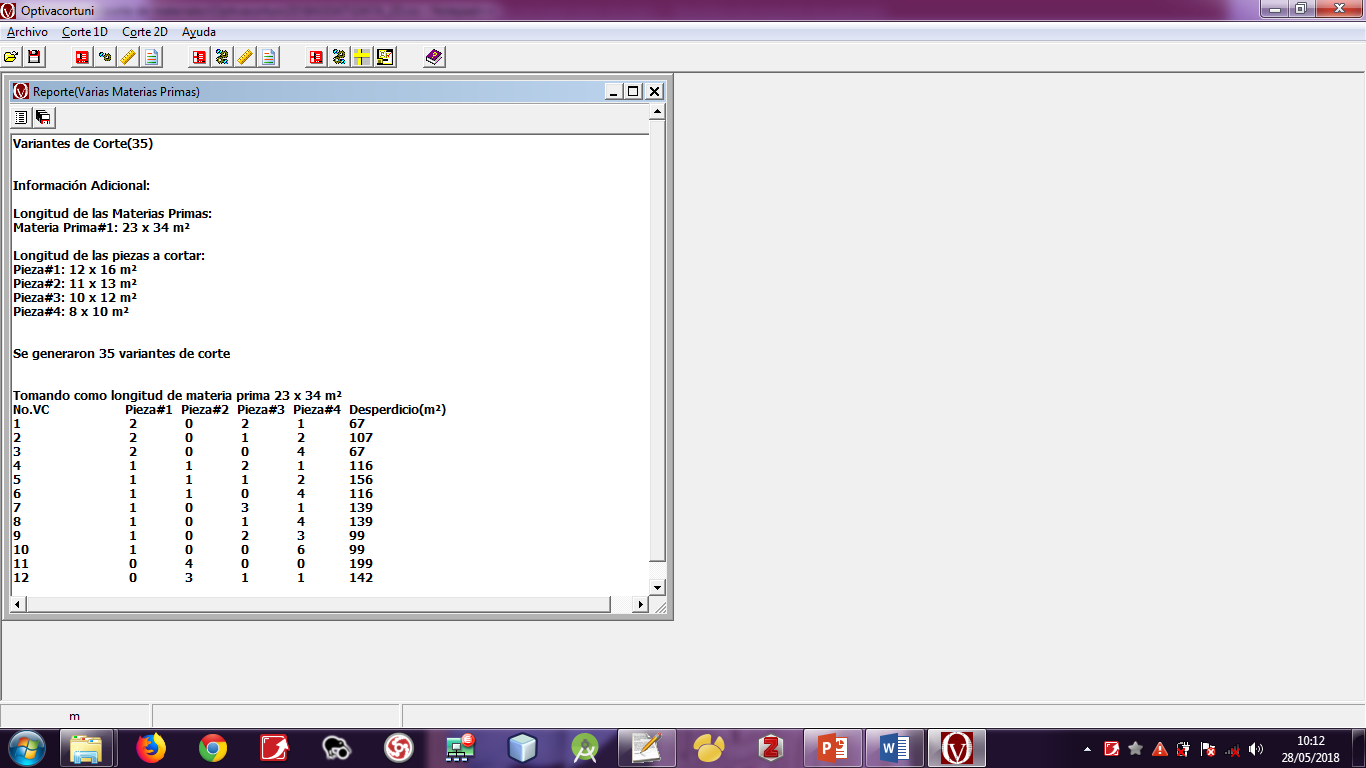
[50] Jacobson, Ivar, *El Proceso Unificado de Desarrollo de Software*, vol. 1. La Habana: Felix Varela, 2004.

**Anexos**

**Anexo1. Introducir plan de corte**



**Anexo 2 – Generar variantes de corte**



**Anexo 3 – Mostrar interpretación gráfica.**

