Universidad Autónoma Metropolitana
Unidad Azcapotzalco

División de Ciencias Básicas e Ingeniería

Sistema para el Análisis de Reglas de Asociación

Modalidad: Proyecto tecnológico

Felipe Verdín Sandria
2113001355
al2113001355@azc.uam.mx

Josué Figueroa González
Profesor Asociado
Departamento de Sistemas
jfgo@correo.azc.uam.mx

Trimestre 2018 Primavera

7 de Septiembre de 2018

Declaratoria

Yo, Josué Figueroa González, declaro que aprobé el contenido del presente Reporte de Proyecto de Integración y doy mi autorización para su publicación en la Biblioteca Digital, así como en el Repositorio Institucional de UAM Azcapotzalco.

Josué Figueroa González

Yo, Felipe Verdín Sandria, doy mi autorización a la Coordinación de Servicios de Información de la Universidad Autónoma Metropolitana, Unidad Azcapotzalco, para publicar el presente documento en la Biblioteca Digital, así como en el Repositorio Institucional de UAM Azcapotzalco.

Felipe Verdín Sandria

Índice

Índice Tablas

Índice Figuras

Resumen

El proyecto parte de la idea de filtrar las reglas de asociación que fueron previamente generadas por programas como Weka y R, de modo que primeramente se hace una limpieza del archivo fuente y almacenarlas en otro archivo que evita perder el documento de origen, la limpieza permitiendo un análisis más exacto de estas. Después tomando opciones de filtrado que el usuario podrá especificar en la aplicación, se comenzara a tomar las especificaciones y así tomar solo las reglas que cumplen con los requisitos seleccionados en pantalla.

Una vez terminado el análisis se muestran graficas de barra así como de pastel siguiendo un criterio de aquellos que tienen el mismo consecuente y contabilizando los aciertos. Posteriormente el usuario tendrá la opción de guardar un archivo de texto que contiene las reglas de asociación que la aplicación muestra después del filtrado.

Introducción

El constante crecimiento de la información hace que cada vez sea más difícil el análisis de la misma dando cabida a grandes pérdidas en cuanto a datos se refiere, es por eso que una de las herramientas que en los últimos años ha logrado facilitar la comprensión de grandes bancos de información es la minería de datos. Mediante la exploración de grandes cantidades de archivos, la minería de datos obtiene patrones que proporcionan información potencialmente útil para el área que solicita el servicio.

Toda base de datos procesada por la minería de datos atraviesa por ciertas fases que garantizaran un análisis completo de la información, estas son: el filtrado de datos que elimina conocimiento no válidos y desconocido; selección de variables reduce el tamaño de los datos seleccionados conservando aquellos que sean de mayor utilidad; extracción de contenido que obtiene patrones de comportamiento; interpretación y evaluación una vez obtenido un modelo que represente lo buscado, se evalúa para determinar si se soluciona el problema [1].

Las reglas de asociación es una técnica de la minería de datos que determina el comportamiento de la información [2]. Están dadas de la forma $W \Longrightarrow B$ dónde: W es un conjunto de atributos conocidos como antecedentes, estos deben ser verdaderos para ser parte de W y B que representa a un atributo con altas probabilidades de ocurrir si los antecedentes W suceden, a este se le llama consecuente. Ejemplo $\{cebolla, vegetales\} \Rightarrow carne$ indica que si una persona compra cebollas y vegetales, es probable que compre carne.

Sin embargo, la interpretación de las reglas de asociación no es sencillo, ya que se generan muchas de ellas y es complicado elegir las más representativas. Por lo tanto, se propone crear un software que facilite la interpretación de las reglas de asociación previamente obtenidas en programas para análisis de minería de datos relacionadas con factores de deserción escolar generando un resultado que de un mejor entendimiento de las mismas por medio de una interfaz intuitiva que proporcionará una mejor forma de visualizar las reglas contenidas en un archivo de texto.

Antecedentes

Proyectos terminales

Visualización de algoritmos de ordenamiento y búsqueda interna con TikZ [3]

El objetivo de este proyecto es crear una aplicación con interfaz gráfica que permita observar patrones de ordenamiento y búsqueda interna con el fin de facilitar la comprensión de los algoritmos según las propiedades especificadas.

Virtualización de redes de computadoras para el monitoreo y análisis de información entrante no deseada [4]

Este proyecto tiene la finalidad de crear una herramienta que permita analizar el tráfico de entrada en una red para resolver problemas como la fuga de información, reducir los robos y fraudes electrónicos, para así poder reforzar la seguridad de acceso a datos que transitan en el sistema.

Control y visualización de información por medios gráficos digitales [5]

El propósito de este proyecto es dar dos prototipos para el control de información y visualización por medios gráficos digitales a partir de proyectos de arquitectura e ingeniería, busca facilitar el trabajo gracias al uso directo e interactivo de la información digitalizada.

Artículos

Finding Interesting Rules from Large Sets of Discovered Association Rules [6]

Este artículo se enfoca en la creación de reglas de asociación así como sus propiedades, selección de las mismas y una aplicación prototipo que principalmente se enfoca en la selección, búsqueda y representación gráfica de los resultados que genero el sistema de análisis.

La visualización de los datos [7]

Este artículo se encarga de mostrar de manera gráfica los datos en diferentes áreas del conocimiento para así poder observar una representación que permita entender la información de manera más sencilla para el usuario.

Proyectos de otras instituciones

Utilización de visualización de datos en tiempo real para mejorar el monitoreo de tráfico de una red y la eficiencia de los administradores de red [8]

El objetivo de este proyecto es que mediante técnicas de visualización de datos capturados a través de puentes de red inteligentes, se pueda administrar está tomando decisiones conforme a lo que se muestre en los gráficos estadísticos.

Las similitudes y diferencias con el proyecto propuesto se muestran en la Tabla 1.

Referencia	Similitudes	Diferencias
[3]	■ Es una aplicación que permite visualizar los datos para facilitar el entendimiento.	■ Su finalidad son los algoritmos de ordenamiento.
[4]	■ Permite un amplio análisis de la información según parámetros establecidos, basadas en su representación gráfica.	■ Su finalidad es la de mejorar la seguridad. ■ No es una aplicación, solo es el uso de distintas aplicaciones.
[5]	■ Procesar distintos tipos de información para poder mejorar en entendimiento del conocimiento mediante medios visuales.	■ El tipo de información que procesa son imágenes y no reglas de asociación.
[6]	■ Permite visualizar un conjunto de Reglas de asociación de manera textual. ■ Permite visualizar de manera gráfica las Reglas de asociación.	■ Se encarga de generar reglas de asociación a partir de una gran cantidad de datos. ■ No es una implementación.
[7]	■ Los modelos de visualización de datos permiten un mejor entendimiento de los mismos.	■ Parte desde el proceso de recolección de la información. ■ Procesa todo tipo de información para su interpretación.
[8]	■ Permite visualizar de manera gráfica un conjunto de elementos para facilitar la toma de decisiones.	■ Recopila paquetes en el tráfico de la red.

Tabla 1. Comparación cualitativa de los trabajos relacionados con el proyecto propuesto

Justificación

Cuando se generan reglas de asociación y se quieren analizar para elegir las más representativas, se tiene el problema de la gran cantidad que se presentan, lo que hace difícil de asimilarlas para el usuario. La redundancia en las reglas de asociación limita la comprensión de las mismas por esta razón, es necesario crear un sistema que tomando como base un archivo que ya contenga toda una gama de reglas de asociación generadas, permita presentarlas y filtrarlas a un usuario final de dos maneras: mediante un formato gráfico y a través de un archivo de salida que contenga un conjunto más pequeño de reglas que represente las más relevantes de las analizadas por el sistema.

Objetivos

Objetivo General

Diseñar e implementar un sistema para el análisis y presentación visual de reglas de asociación.

Objetivos Específicos

1. Diseñar e implementar un módulo de carga para almacenar en una base de datos las reglas de asociación ya generadas.
2. Diseñar e implementar un módulo de filtrado para establecer parámetros que se desean para el filtrado de las reglas.
3. Diseñar e implementar un módulo de resultados para mostrar en pantalla los resultados de manera textual.
4. Diseñar e implementar un módulo de visualización que muestre los resultados en un formato gráfico.
5. Diseñar e implementar un módulo de exportación que creará un archivo de texto con las reglas ya analizadas y filtradas.

Marco Teórico

Reglas de asociación

Son una herramienta de apoyo en el análisis de un gran conjunto de datos, estos ayudan a obtener relaciones entre toda la información que se está procesando. Estas se componen principalmente de antecedentes y consecuentes que se componen de la siguiente manera:

$$Antecedente^1, Antecedente^2, \dots, Antecedente^n => Consecuente$$

Donde los antecedentes son aquellos ítems que en su conjunto nos dará una consecuencia a la cual se le conoce como consecuente. Las reglas pueden contener un o más antecedente y de la misma manera el consecuente, sin embargo por cuestiones prácticas se utilizaran aquellas reglas que tengan un solo consecuente.

Otro aspecto que ayuda en la extracción de información útil por parte de esta herramienta son los valores que estas contienen, estos son: [9]

- support o soporte da el porcentaje total de apariciones de la regla en todos los datos que fueron analizados, es decir, muestra el promedio de veces que una regla se repite conforme al total de transacciones.
- confidence o confianza muestra el porcentaje de transacciones que al contener uno o más de los antecedentes que la componen darán como resultado el mismo consecuente, en pocas palabras indica el porcentaje de reglas que contienen el mismo consecuente si es que tiene alguno de los ítems de los antecedentes.
- lift o elevación es aquel que muestra la proporción del soporte que se muestra conforme al soporte esperado donde, si su valor es igual a 1 indica que la regla aparece conforme a lo esperado, si es mayor que 1 indica que este aparece más veces de lo esperado y finalmente si es menor a 1 indica que el conjunto aparece menos veces de lo que se estima.

Java Swing

Es una de las paqueterías de Java que se encarga de proporcionar una interfaz gráfica de usuario o GUI a un programa desarrollado en lenguaje Java. Este apareció por primera vez en la versión 1.2 de Java, se compone de una gama amplia de herramientas para diseñar una interfaz amigable para los usuarios. Esta herramienta a diferencia de otras permite personalizar a amplio rango ya que la apariencia de las ventanas puede ser totalmente modificada por el usuario sin agregar código extra, así mismo permite modificar los componentes por defecto que existen en Swing y adaptarlos a lo que necesita el usuario, logrando que sea un sistema muy versátil e intuitivo de manejar. [10]

JFreeChar

Es una librería de complemento para Swing que da al usuario las funciones necesarias para crear gráficos complejos de manera sencilla y completa. Entre lo que permite crear esta paquetería son: gráficos XY, circular, diagrama de Gantt, graficas de barra, single valued, tabla de viento, etc. Para cada una de las opciones que ofrece esta se puede personalizar para que se ajuste a las necesidades del usuario permitiendo el uso de marcadores para identificar de mejor manera cada aspecto que se desea mostrar con la ayuda de esta herramienta. [11]

Visualización de la información

Este término se refiere a la disciplina que supone presentar una vasta cantidad de información de manera clara y a su vez que sea de utilidad. En la visualización de la información se debe de tomar en cuenta que el usuario final por lo general será cualquier persona que pretenda obtener conocimiento de esos datos, de este modo se tiene la obligación de presentar la información de un modo que cualquier persona pueda procesarla y entenderla de manera fácil y sencilla.

La finalidad de esta es que podamos representar y presentar información con la finalidad de ampliar el conocimiento, presentar la información de manera visual ayuda a que grandes cúmulos de información se puedan expresar en un conjunto más pequeño que represente lo que realmente es útil de todos los datos analizados. [12]

Desarrollo del proyecto

Para el desarrollo de los módulos de carga, filtrado, resultados, visualización y exportación se utilizó el lenguaje de programación de Java para implementar toda la funcionalidad de estos y para la creación de las interfaces de pantalla se hizo uso de la paquetería Swing de Java que permitió desarrollar una pantalla cómoda, fácil e intuitiva para que cualquier usuario pueda usarla sin problemas.

El proyecto no contempló la generación de reglas de asociación, de modo que se tomó reglas generadas en las aplicaciones Weka y R. Para el desarrollo de la aplicación se utilizaron reglas que contienen uno o más antecedentes y un solo consecuente, tampoco durante el desarrollo se consideró resolver problemas relacionados al uso posterior de las reglas generadas por el sistema. De modo que se usó reglas como las que muestra la *Figura 1*.

```
16,"{CTA=MB,CUR_APR=APROBO} => {CCAL=MB_U}",0.02,0.4,3.17,86
18,"{CTA=MB,TIEM=EXAMEN_T} => {CCAL=MB_U}",0.02,0.4,3.17,86
31,"{CTA=MB,CUR_APR=APROBO,TIEM=EXAMEN_T} => {CCAL=MB_U}",0.02,0.4,3.17,86
2,{CUR_APR=APROBO} => {CCAL=MB_U},0.04,0.26,2.01,155
3,{TIEM=EXAMEN_T} => {CCAL=MB_U},0.04,0.26,2.01,155
15,"{CUR_APR=APROBO,TIEM=EXAMEN_T} => {CCAL=MB_U}",0.04,0.26,2.01,155
4,{CTA=MB} => {CCAL=MB_U},0.06,0.25,2,241
29,"{CTA=MB,CUR_APR=PRIMER,TIEM=UNO_T} => {CCAL=MB_U}",0,0.25,1.97,11
34,"{CTA=MB,CUR_APR=PRIMER,TIEM=SIGUIENTE_T} => {CCAL=MB_U}",0.03,0.23,1.78,1
12,"{CTA=MB,TIEM=UNO_T} => {CCAL=MB_U}",0,0.22,1.76,15
21,"{CTA=MB,TIEM=SIGUIENTE_T} => {CCAL=MB_U}",0.03,0.22,1.73,133
22,"{CTA=MB,CUR_APR=PRIMER} => {CCAL=MB_U}",0.03,0.22,1.73,139
17,"{CTA=B,CUR_APR=APROBO} => {CCAL=MB_U}",0.02,0.18,1.38,69
19,"{CTA=B,TIEM=EXAMEN_T} => {CCAL=MB_U}",0.02,0.18,1.38,69
32,"{CTA=B,CUR_APR=APROBO,TIEM=EXAMEN_T} => {CCAL=MB_U}",0.02,0.18,1.38,69
20,"{CTA=MB,CUR_APR=SEGUNDO} => {CCAL=MB_U}",0,0.13,1.05,13
11,"{CUR_APR=SEGUNDO,TIEM=UNO_T} => {CCAL=MB_U}",0,0.12,0.98,19
14,"{CUR_APR=PRIMER,TIEM=UNO_T} => {CCAL=MB_U}",0.01,0.12,0.97,29
26,"{CTA=B,CUR_APR=PRIMER,TIEM=MAS_DE_DOS_T} => {CCAL=MB_U}",0,0.12,0.96,12
33,"{CTA=MB,CUR_APR=SEGUNDO,TIEM=SIGUIENTE_T} => {CCAL=MB_U}",0,0.12,0.95,7
28,"{CTA=B,CUR_APR=SEGUNDO,TIEM=UNO_T} => {CCAL=MB_U}",0,0.12,0.94,5
1,{TIEM=UNO_T} => {CCAL=MB_U},0.01,0.11,0.9,49
8,"{CUR_APR=MAS_DE_DOS,TIEM=SIGUIENTE_T} => {CCAL=MB_U}",0,0.11,0.89,8
23,"{CUR_APR=PRIMER,TIEM=SIGUIENTE_T} => {CCAL=MB_U}",0.06,0.11,0.89,249
5,{CTA=B} => {CCAL=MB_U},0.04,0.11,0.88,186
7,{CUR_APR=PRIMER} => {CCAL=MB_U},0.08,0.11,0.88,321
27,"{CTA=S,CUR_APR=SEGUNDO,TIEM=UNO_T} => {CCAL=MB_U}",0,0.11,0.88,10
10,"{CUR_APR=PRIMER,TIEM=MAS_DE_DOS_T} => {CCAL=MB_U}",0.01,0.11,0.87,32
30,"{CTA=B,CUR_APR=PRIMER,TIEM=UNO_T} => {CCAL=MB_U}",0,0.11,0.87,10
6,{TIEM=SIGUIENTE_T} => {CCAL=MB_U},0.07,0.11,0.86,280
9,"{CTA=B,TIEM=MAS_DE_DOS_T} => {CCAL=MB_U}",0,0.11,0.85,14
```

Figura 1 Formato de las reglas de asociación a analizar

El módulo de carga se dio por concluido una vez que tomaron las reglas de asociación mostradas en la *Figura 1* y se pasarón a un formato que permitió un análisis más amigable con el sistema creado, de modo que una vez que el módulo se concluyó permitió crear un archivo como el que se muestra en la *Figura 2*.

```
1  CTA=MB, CUR_APR=APROBO, CTA=B | CCAL=S_U | 0.02 | 0.4 | 3.17 | 86
2  CTA=MB, TIEM=EXAMEN_T | CCAL=NA_U | 0.02 | 0.4 | 3.17 | 86
3  CTA=MB, CUR_APR=APROBO, TIEM=EXAMEN_T | CCAL=MB_U | 0.02 | 0.4 | 3.17 | 86
4  CUR_APR=APROBO | CCAL=MB_U | 0.04 | 0.26 | 2.01 | 155
5  TIEM=EXAMEN_T | CCAL=MB_U | 0.04 | 0.26 | 2.01 | 155
6  CUR_APR=APROBO, TIEM=EXAMEN_T | CCAL=MB_U | 0.04 | 0.26 | 2.01 | 155
7  CTA=MB | CCAL=MB_U | 0.06 | 0.25 | 2 | 241
8  CTA=MB, CUR_APR=PRIMER, TIEM=UNO_T | CCAL=B_U | 0 | 0.25 | 1.97 | 11
9  CTA=MB, CUR_APR=PRIMER, TIEM=SIGUIENTE_T | CCAL=MB_U | 0.03 | 0.23 | 1.78 | 123
10 CTA=MB, TIEM=UNO_T | CCAL=MB_U | 0 | 0.22 | 1.76 | 15
11 CTA=MB, TIEM=SIGUIENTE_T | CCAL=MB_U | 0.03 | 0.22 | 1.73 | 133
12 CTA=MB, CUR_APR=PRIMER | CCAL=B_U | 0.03 | 0.22 | 1.73 | 139
13 CTA=B, CUR_APR=APROBO | CCAL=MB_U | 0.02 | 0.18 | 1.38 | 69
14 CTA=B, TIEM=EXAMEN_T | CCAL=S_U | 0.02 | 0.18 | 1.38 | 69
15 CTA=B, CUR_APR=APROBO, TIEM=EXAMEN_T | CCAL=MB_U | 0.02 | 0.18 | 1.38 | 69
16 CTA=MB, CUR_APR=SEGUNDO | CCAL=MB_U | 0 | 0.13 | 1.05 | 13
17 CUR_APR=SEGUNDO, TIEM=UNO_T | CCAL=NA_U | 0 | 0.12 | 0.98 | 19
18 CUR_APR=PRIMER, TIEM=UNO_T | CCAL=MB_U | 0.01 | 0.12 | 0.97 | 29
19 CTA=B, CUR_APR=PRIMER, TIEM=MAS_DE_DOS_T | CCAL=B_U | 0 | 0.12 | 0.96 | 12
20 CTA=MB, CUR_APR=SEGUNDO, TIEM=SIGUIENTE_T | CCAL=MB_U | 0 | 0.12 | 0.95 | 7
21 CTA=B, CUR_APR=SEGUNDO, TIEM=UNO_T | CCAL=MB_U | 0 | 0.12 | 0.94 | 5
22 TIEM=UNO_T | CCAL=MB_U | 0.01 | 0.11 | 0.9 | 49
23 CUR_APR=MAS_DE_DOS, TIEM=SIGUIENTE_T | CCAL=MB_U | 0 | 0.11 | 0.89 | 8
24 CUR_APR=PRIMER, TIEM=SIGUIENTE_T | CCAL=S_U | 0.06 | 0.11 | 0.89 | 249
25 CTA=B | CCAL=MB_U | 0.04 | 0.11 | 0.88 | 186
26 CUR_APR=PRIMER | CCAL=MB_U | 0.08 | 0.11 | 0.88 | 321
27 CTA=S, CUR_APR=SEGUNDO, TIEM=UNO_T | CCAL=MB_U | 0 | 0.11 | 0.88 | 10
28 CUR_APR=PRIMER, TIEM=MAS_DE_DOS_T | CCAL=MB_U | 0.01 | 0.11 | 0.87 | 32
29 CTA=B, CUR_APR=PRIMER, TIEM=UNO_T | CCAL=MB_U | 0 | 0.11 | 0.87 | 10
30 TIEM=SIGUIENTE_T | CCAL=MB_U | 0.07 | 0.11 | 0.86 | 280
31 CTA=B, TIEM=MAS_DE_DOS_T | CCAL=MB_U | 0 | 0.11 | 0.85 | 14
```

Figura 2 reglas de asociación que se generan en el módulo de carga.

La interfaz para este módulo permite examinar en el ordenador cualquier archivo en formato .csv de modo que una vez que se carga crea el archivo que contiene a la *Figura 2*. La interfaz que se diseñó se muestra en la *Figura 3*.

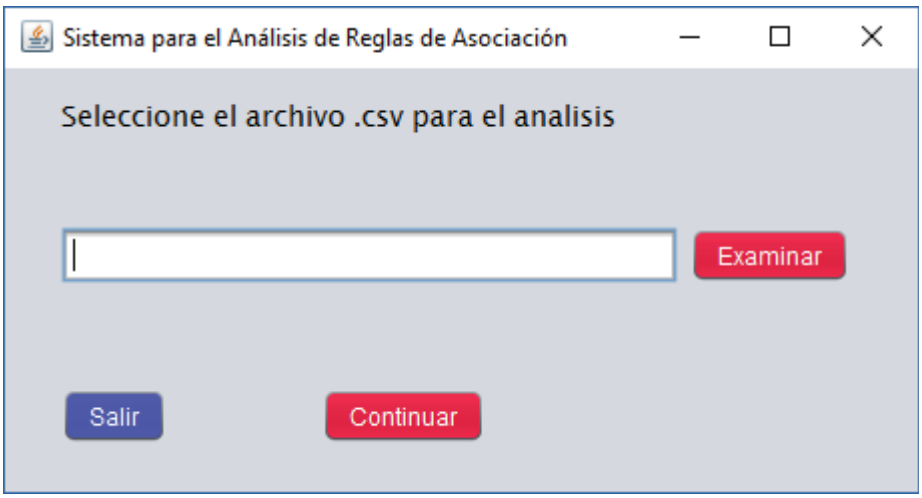

Figura 3 GUI del módulo de carga

El módulo de filtrado se dio por concluido una vez que tomó el archivo con formato más simple como el de la *Figura 2* se pudo filtrar las reglas según las opciones de filtrado que el usuario marca en la interfaz de la *Figura 4*, este se creó pensando en los distintos ítems que contiene el archivo generado por el modulo anterior, es así que se pueden seleccionar hasta tres posibles antecedentes, un consecuente y valor mayores a 0 de support, confidence y lift.

Figura 4 GUI del módulo de filtrado.

Una vez que el usuario seleccionó el filtrado que necesita, el módulo de resultado se realizó pensando en que se pueda visualizar las opciones que se seleccionaron para el filtrado así como en esta misma interfaz que se muestra en la *Figura 5* podemos notar que se diseñó a la par con el módulo de visualización ya que fue más practico mostrarlos al mismo tiempo y se evitó crear más GUI innecesarias para una visualización de la información apropiada, de este modo pudo notar que las gráficas fueron creadas para mostrar el número de coincidencias según el consecuente resultante.

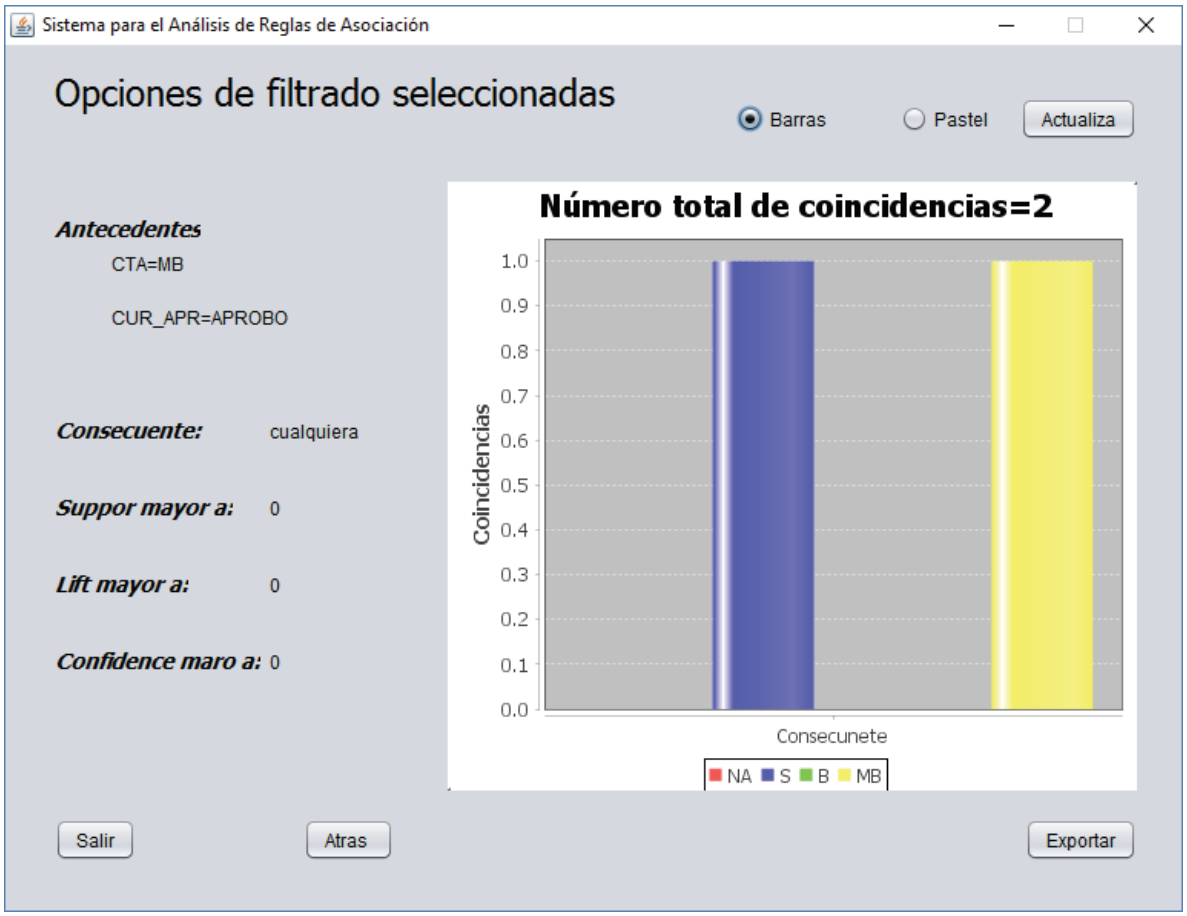

Figura 5 GUI del módulo de resultados y módulo de visualización.

Finalmente el módulo de exportación se dio por finalizado al momento de que exitosamente se pudo generar un archivo .txt que contiene todas las coincidencias que arrojo el módulo de filtrado, así es que al exportar se generó una ventana de navegación que guarda el archivo en el destino especificado por el usuario.

Resultados

El sistema que se desarrolló nos arroja dos resultados que son de utilidad para el usuario según sus necesidades, el primero son las gráficas de coincidencia que se desarrollaron de modo que pueda mostrar en forma de pastel *Figura 6* y de barras *Figura 7* para una mejor visualización de la información, es por eso que el usuario puede entender de una mejor forma los datos que están presentes en el archivo original. Con la ayuda de las gráficas se puede apreciar el número de reglas de asociación que son de utilidad para él según como el usuario la haya filtrado.

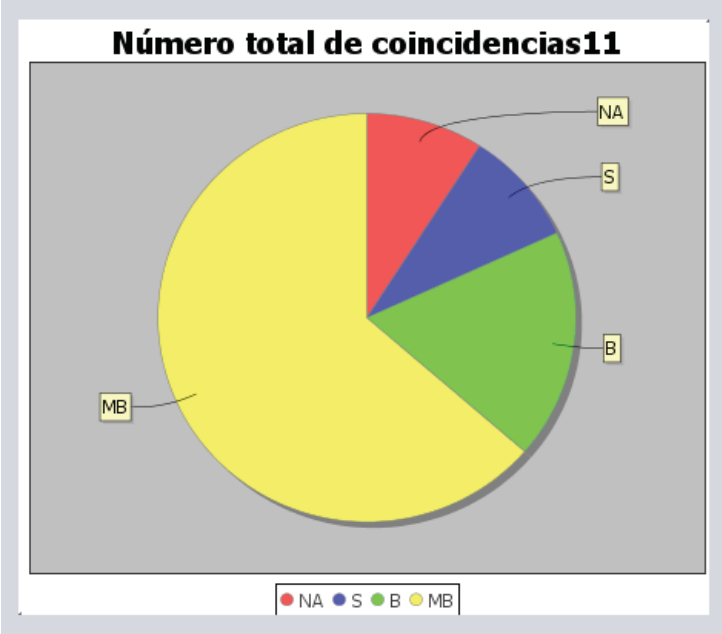

Figura 6 Gráfica de pastel que arroja después del filtrado de las Reglas

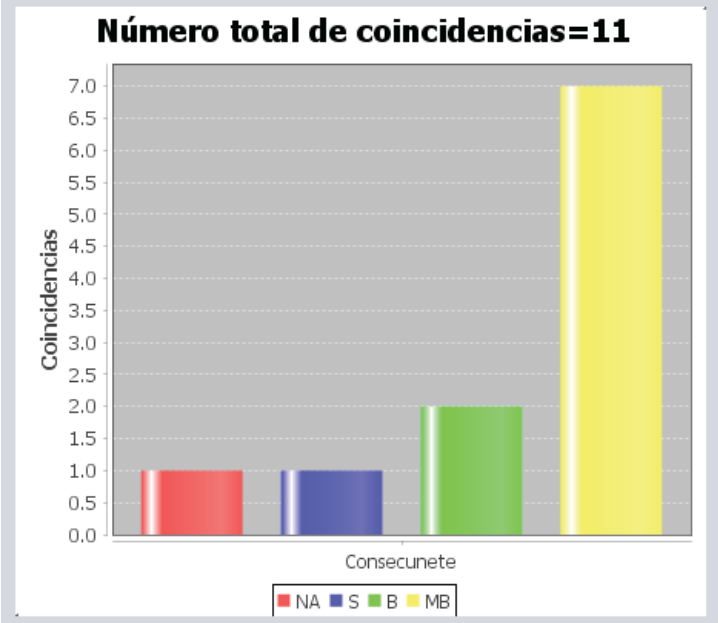

Figura 7 Gráfica de barras que arroja después del filtrado de Reglas

Para complementar la visualización de la información se diseñó el sistema de modo que se puede exportar un archivo de texto. La *Figura 8* que contiene todas las reglas que son el resultado del filtrado del usuario, el formato que se emplea para estos es uno que permite identificar cada uno de los elementos de la regla, por eso es que el uso se le puede dar será variado dependiendo del usuario y sus necesidades

```
1  CTA=MB,  CUR_APR=APROBO,  CTA=B  |  CCAL=S_U  |  0.02  |  0.4  |  3.17  |  86
2  CTA=MB,  TIEM=EXAMEN_T  |  CCAL=NA_U  |  0.02  |  0.4  |  3.17  |  86
3  CTA=MB,  CUR_APR=APROBO,  TIEM=EXAMEN_T  |  CCAL=MB_U  |  0.02  |  0.4  |  3.17  |  86
4  CTA=MB  |  CCAL=MB_U  |  0.06  |  0.25  |  2  |  241
5  CTA=MB,  CUR_APR=PRIMER,  TIEM=UNO_T  |  CCAL=B_U  |  0  |  0.25  |  1.97  |  11
6  CTA=MB,  CUR_APR=PRIMER,  TIEM=SIGUIENTE_T  |  CCAL=MB_U  |  0.03  |  0.23  |  1.78  |  123
7  CTA=MB,  TIEM=UNO_T  |  CCAL=MB_U  |  0  |  0.22  |  1.76  |  15
8  CTA=MB,  TIEM=SIGUIENTE_T  |  CCAL=MB_U  |  0.03  |  0.22  |  1.73  |  133
9  CTA=MB,  CUR_APR=PRIMER  |  CCAL=B_U  |  0.03  |  0.22  |  1.73  |  139
10 CTA=MB,  CUR_APR=SEGUNDO  |  CCAL=MB_U  |  0  |  0.13  |  1.05  |  13
11 CTA=MB,  CUR_APR=SEGUNDO,  TIEM=SIGUIENTE_T  |  CCAL=MB_U  |  0  |  0.12  |  0.95  |  7
12
```

Figura 8 Contenido del archivo .txt que se exporta.

Conclusiones

El proyecto que se desarrolló durante el trimestre se completó de una manera satisfactoria gracias a que se pudo implementar un Sistema para el Análisis de Reglas de Asociación funcional y con lo especificado anteriormente en la propuesta, debido a que se presentó el módulo de carga que toma un archivo que contiene reglas que no han sido procesadas. Así es que se les dio un formato amigable con el desarrollo del resto de módulos y se almacenaron para su análisis dentro del sistema.

Posteriormente se presentó la función principal del sistema que es el módulo de filtrando, este supuso la mayor carga del proyecto ya que de este dependía la funcionalidad total del sistema propuesto, inicialmente se tenía un diseño y un algoritmo que permitía solo reglas de asociación con dos antecedentes pero, se tomó la decisión final de poder procesar n antecedentes y así se tuvo una fluidez más apropiada para el resto de módulos.

Para la creación del módulo de resultados y el de visualización se tenía planeado en un inicio que cada una tuviera su propia GUI, sin embargo resulto ser más práctico y útil mostrarlas en la misma, ya que esto permitió facilitar la visualización de la información así como un aspecto más amigable con los usuarios sin comprometer al conocimiento que el sistema aporta.

Finalmente para el ultimo módulo se realizó de modo que el usuario pudiese seleccionar la ubicación para guardar el archivo con las reglas ya filtradas, esto se hizo para evitar que se pierdan análisis previos hechos por el usuario y evitar y una fuga de conocimiento de los datos que se están obteniendo con la ayuda del sistema desarrollado y así tener todo lo necesario para el posterior uso de los resultados que arroja el sistema.

Para concluir se puede afirmar que todos los objetivo propuestos fueron cumplidos ya que se pudo desarrollar un sistema totalmente funcional según los especificado así como algunas mejoras que facilitaron el procesamiento de la información por parte de este. Es así que el sistema se desempeña con una fluidez que permitirá analizar archivos que contengan una gran cantidad de reglas de asociación.

Bibliografía

[1] (2011) KDD: Proceso de Extracción de Conocimiento. [En línea]. Disponible: http://www.webmining.cl/2011/01/proceso-de-extraccion-de-conocimiento/

[2] Reglas de Asociación-IBM. [En línea]. Disponible: https://www.ibm.com/support/knowledgecenter/es/SS3RA7_16.0.0/com.ibm.spss.modeler.help/clementine/nodes_associationrules.htm

[3] J. Bocanegra y Y. Tovar, "Visualización de algoritmos de ordenamiento y búsqueda interna con TikZ", Proyecto Tecnológico, División de Ciencias Básicas e Ingeniería, Universidad Autónoma Metropolitana Azcapotzalco, México, 2011.

[4] A. Martínez y C. Servín, "Virtualización de redes de computadoras para el monitoreo y análisis de información entrante no deseada", Proyecto Tecnológico, División de Ciencias Básicas e Ingeniería, Universidad Autónoma Metropolitana Azcapotzalco, México, 2007.

[5] M. Baruch, "Control y visualización de información por medios gráficos digitales", Tesis, División de Ciencias y Artes para el Diseño, Universidad Autónoma Metropolitana Azcapotzalco, México, 2015.

[6] M. Klemettinen, H. Mannila, P. Ronkainen, H. Toivonen, y A. I. Verkamo, "Finding Interesting Rules from Large Sets of Discovered Association Rules", *CIKM '94 Proceedings of the third international conference on Information and knowledge management*, pp. 401–407, dic. 1994.

[7] J. L. Valero, "La visualización de datos", *Ámbitos. Revista Internacional de Comunicación*, vol. 25, 2014.

[8] S. Pizarro, "Utilización de visualización de datos en tiempo real para mejorar el monitoreo de tráfico de una red y la eficiencia de los administradores de red", Tesis, Magister en Ingeniería Informática, Universidad Andrés Bello, Chile.

[9] Características de una regla de asociación [En línea]. Disponible: https://www.ibm.com/support/knowledgecenter/es/SSEPGG_8.2.0/com.ibm.im.visual.doc/idmu0mst25.html

[10] (2018) Swing (biblioteca gráfica) [En línea]. Disponible: https://es.wikipedia.org/wiki/Swing_(biblioteca_gr%C3%A1fica)

[11] JFreeChart [En línea]. Disponible: http://www.juntadeandalucia.es/servicios/madeja/contenido/recurso/240

[12] Visualización de información: ¿arte o ciencia? [En línea]. Disponible: https://ignasialcalde.es/visualizacion-de-informacion-arte-o-ciencia/

Anexos

Reglas.java

Contiene un objeto que contiene todas las propiedades de una regla se asociacion.

```
package reglas.clases;

import java.util.Iterator;
import java.util.LinkedList;

public class Reglas {
    private LinkedList<String> antecedentes;
    private String consecuente;
    private String support;
    private String lift;
    private String confidence;
    private String count;

    public LinkedList<String> getAntecedentes() {
        return antecedentes;
    }

    public void setAntecedentes(LinkedList<String> antecedentes) {
        this.antecedentes = antecedentes;
    }

    public String getConsecuente() {
        return consecuente;
    }

    public void setConsecuente(String consecuente) {
        this.consecuente = consecuente;
    }

    public String getSupport() {
        return support;
    }

    public void setSupport(String support) {
        this.support = support;
    }

    public String getLift() {
        return lift;
    }

    public void setLift(String lift) {
        this.lift = lift;
    }

    public String getConfidence() {
        return confidence;
    }

    public void setConfidence(String confidence) {
```

```
        this.confidence = confidence;
    }

    public String getCount() {
        return count;
    }

    public void setCount(String count) {
        this.count = count;
    }


    public String toString(){
        String result="";
        int i=0;
        Iterator it = antecedentes.iterator();
            while (it.hasNext()) {
                result = result + it.next() + ", ";
                i++;
    }
            result=result.substring(0,result.length()-2);

        return result + " | " + consecuente + " | " + support + " | " +
lift + " | " + confidence + " | " + count;
    }
}
```

LimpiarFuente.java

Toma el archivo .csv que contiene las reglas de asociación sin procesar y las optimiza a un formato más simple y fácil de manipular.

```
package reglas.limpieza;

import java.io.BufferedReader;
import java.io.File;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.FileReader;
import java.io.PrintStream;
import java.util.LinkedList;
import reglas.clases.Reglas;

public class LimpiarFuente {
    private Reglas regla;

    public LimpiarFuente(){
        regla = new Reglas();
    }

    public void procesaArchivo(File sucio) throws FileNotFoundException{
      File archivo = null;
      FileReader fr = null;
      BufferedReader br = null;
```

```
      FileOutputStream os = new FileOutputStream("test/ReglasClean.txt");
      PrintStream ps = new PrintStream(os);


      try {
         archivo = sucio;
         fr = new FileReader (archivo);
         br = new BufferedReader(fr);
         String linea;

         while((linea=br.readLine())!=null){
            linea=eliminaNumero(linea);
            linea=obtenerAntecedentes(linea);
            linea=obtenerConsecuente(linea);
            linea=obtenerSupport(linea);
            linea=obtenerLift(linea);
            linea=obtenerConfidence(linea);
            obtenerCount(linea);
            ps.println(regla.toString());
            }
      }
      catch(Exception e){
      }finally{
         try{
            if( null != fr ){
               fr.close();
            }
         }catch (Exception e2){
         }
      }


   }

     public static String eliminaNumero(String cadena){
       int pos, pos2;
       String aux;
       pos=cadena.indexOf(',');
       if(cadena.indexOf('"')==-1)
       pos2=2;
       else
       pos2=3;
       aux=cadena.substring(pos+pos2);
       return aux;
   }

   public String obtenerAntecedentes(String cadena){
       LinkedList<String> lista = new LinkedList<String>();
       int pos, pos2;
       lista.clear();
       String aux, aux2;
       pos=cadena.indexOf('}');
       aux=cadena.substring(0,pos);
       pos2=aux.indexOf(',');

       if(pos2==-1){
           lista.add(aux);
```

```
        }
        else{

            while(pos2!=-1){

                aux2=aux.substring(0,pos2);
                aux=aux.substring(pos2+1,aux.length());
                lista.add(aux2);
                pos2=aux.indexOf(',');
            }
            lista.add(aux);

        }
        regla.setAntecedentes(lista);
        aux2=cadena.substring(pos+6);
        return aux2;
    }

    public String obtenerConsecuente(String cadena){
        int pos, pos2;
        String aux;
        pos=cadena.indexOf('}');
        if(cadena.indexOf("")==-1)
        pos2=2;
        else
        pos2=3;
        regla.setConsecuente(cadena.substring(0,pos));
        //consecuente=cadena.substring(0,pos);
        aux=cadena.substring(pos+pos2);
        //System.out.println("Posicion = "+ pos);
        return aux;
    }

    public String obtenerSupport(String cadena){
        int pos;
        String aux;
        pos=cadena.indexOf(',');
        regla.setSupport(cadena.substring(0,pos));
        //support=cadena.substring(0,pos);
        aux=cadena.substring(pos+1);
        return aux;
    }

    public String obtenerLift(String cadena){
        int pos;
        String aux;
        pos=cadena.indexOf(',');
        regla.setLift(cadena.substring(0,pos));
        //lift=cadena.substring(0,pos);
        aux=cadena.substring(pos+1);
        return aux;
    }

    public String obtenerConfidence(String cadena){
        int pos;
        String aux;
        pos=cadena.indexOf(',');
```

```
        regla.setConfidence(cadena.substring(0,pos));
        aux=cadena.substring(pos+1);
        return aux;
    }

    public void obtenerCount(String cadena){
        regla.setCount(cadena);
          }
}
```

OperacionesReglas.java

Toma el archivo de texto con las reglas ya limpias y las comienza a procesar una a una preparándolas para un posterior filtrado.

```
package reglas.limpieza;

import java.io.BufferedReader;
import java.io.File;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.FileReader;
import java.io.PrintStream;
import java.util.LinkedList;
import reglas.clases.Reglas;

public class LimpiarFuente {
    private Reglas regla;

    public LimpiarFuente(){
        regla = new Reglas();
    }

    public void procesaArchivo(File sucio) throws FileNotFoundException{
      File archivo = null;
      FileReader fr = null;
      BufferedReader br = null;
      FileOutputStream os = new FileOutputStream("test/ReglasClean.txt");
      PrintStream ps = new PrintStream(os);


      try {
         archivo = sucio;
         fr = new FileReader (archivo);
         br = new BufferedReader(fr);
         String linea;

         while((linea=br.readLine())!=null){
            linea=eliminaNumero(linea);
            linea=obtenerAntecedentes(linea);
            linea=obtenerConsecuente(linea);
            linea=obtenerSupport(linea);
            linea=obtenerLift(linea);
            linea=obtenerConfidence(linea);
            obtenerCount(linea);
```

```
            ps.println(regla.toString());
            }
      }
      catch(Exception e){
      }finally{
         try{
            if( null != fr ){
               fr.close();
            }
         }catch (Exception e2){
         }
      }


   }

     public static String eliminaNumero(String cadena){
       int pos, pos2;
       String aux;
       pos=cadena.indexOf(',');
       if(cadena.indexOf('"')==-1)
       pos2=2;
       else
       pos2=3;
       aux=cadena.substring(pos+pos2);
       return aux;
   }

   public String obtenerAntecedentes(String cadena){
       LinkedList<String> lista = new LinkedList<String>();
       int pos, pos2;
       lista.clear();
       String aux, aux2;
       pos=cadena.indexOf('}');
       aux=cadena.substring(0,pos);
       pos2=aux.indexOf(',');

       if(pos2==-1){
           lista.add(aux);
       }
       else{

           while(pos2!=-1){

               aux2=aux.substring(0,pos2);
               aux=aux.substring(pos2+1,aux.length());
               lista.add(aux2);
               pos2=aux.indexOf(',');
           }
           lista.add(aux);

       }
       regla.setAntecedentes(lista);
       aux2=cadena.substring(pos+6);
       return aux2;
   }
```

```
    public String obtenerConsecuente(String cadena){
        int pos, pos2;
        String aux;
        pos=cadena.indexOf('}');
        if(cadena.indexOf('"')==-1)
        pos2=2;
        else
        pos2=3;
        regla.setConsecuente(cadena.substring(0,pos));
        //consecuente=cadena.substring(0,pos);
        aux=cadena.substring(pos+pos2);
        //System.out.println("Posicion = "+ pos);
        return aux;
    }

    public String obtenerSupport(String cadena){
        int pos;
        String aux;
        pos=cadena.indexOf(',');
        regla.setSupport(cadena.substring(0,pos));
        //support=cadena.substring(0,pos);
        aux=cadena.substring(pos+1);
        return aux;
    }

    public String obtenerLift(String cadena){
        int pos;
        String aux;
        pos=cadena.indexOf(',');
        regla.setLift(cadena.substring(0,pos));
        //lift=cadena.substring(0,pos);
        aux=cadena.substring(pos+1);
        return aux;
    }

    public String obtenerConfidence(String cadena){
        int pos;
        String aux;
        pos=cadena.indexOf(',');
        regla.setConfidence(cadena.substring(0,pos));
        aux=cadena.substring(pos+1);
        return aux;
    }

    public void obtenerCount(String cadena){
        regla.setCount(cadena);
          }
}
```

OperacionesItem.java

Permite que a partir de las reglas que se ingresan al sistema pueda llenar los combo boxes que tiene las opciones de filtrado tanto en antecedentes y consecuentes.

```
package reglas.procesos;

import java.util.LinkedList;
import javax.swing.JComboBox;
import javax.swing.JOptionPane;

public class OperacionesItem {
    private LinkedList<String> listaCta;
    private LinkedList<String> listaCur;
    private LinkedList<String> listaTiem;
    private LinkedList<String> listaCon;


    public OperacionesItem(){
        listaCta = new LinkedList<String>();
        listaCon = new LinkedList<String>();
        listaCur = new LinkedList<String>();
        listaTiem = new LinkedList<String>();
    }


    public String obtenerConcecunete(String cadena) {
        String aux;
        cadena=cadena.trim();

aux=cadena.substring(cadena.indexOf("|")+1,cadena.length()).trim();
        aux=aux.substring(0,aux.indexOf("|")).trim();
        return aux;
    }

    public LinkedList<String> obtenerAntecedentes(String cadena){
        LinkedList<String> lista = new LinkedList<String>();
        int pos, pos2;
        cadena = cadena.trim();
        lista.clear();
        String aux, aux2;
        pos=cadena.indexOf('|');
        aux=cadena.substring(0,pos);
        pos2=aux.indexOf(',');

        if(pos2==-1){
            lista.add(aux);
        }
        else{

            while(pos2!=-1){

                aux2=aux.substring(0,pos2);
                aux=aux.substring(pos2+1,aux.length());
                lista.add(aux2);
                pos2=aux.indexOf(',');
            }
            lista.add(aux);
```

```
        }
        return lista;
    }

    public void agregarOpcionesFiltrado(String aux){
        String aux2;
        int count=0;
        if(aux.contains("CTA")){
            if(listaCta.isEmpty()){
                aux=aux.substring(aux.indexOf("=")+1,
aux.length()).trim();
                listaCta.add(aux);
            }
            else{
                for(int j=0;j<listaCta.size();j++){
                    aux=aux.substring(aux.indexOf("=")+1,
aux.length()).trim();
                    aux2=listaCta.get(j).trim();
                    if(aux.compareTo(aux2)==0){
                        count++;
                        break;
                    }
                }
                if(count==0){
                listaCta.add(aux);
                }
            }
        }
        else if(aux.contains("CUR")){
            if(listaCur.isEmpty()){
                aux=aux.substring(aux.indexOf("=")+1,
aux.length()).trim();
                listaCur.add(aux);
            }
            else{
                for(int j=0;j<listaCur.size();j++){
                    aux=aux.substring(aux.indexOf("=")+1,
aux.length()).trim();
                    aux2=listaCur.get(j).trim();
                    if(aux.trim().compareTo(aux2)==0){
                        count++;
                        break;
                    }
                }
                if(count==0) {
                        listaCur.add(aux);
                }
            }
        }
        else if(aux.contains("TIEM")){
            if(listaTiem.isEmpty()){
                aux=aux.substring(aux.indexOf("=")+1,
aux.length()).trim();
                listaTiem.add(aux);
            }
            else{
                for(int j=0;j<listaTiem.size();j++){
```

```
                    aux=aux.substring(aux.indexOf("=")+1,
aux.length()).trim();
                    aux2=listaTiem.get(j).trim();
                    if(aux.trim().compareTo(aux2)==0){
                        count++;
                        break;
                    }
                }
                if(count==0){
                        listaTiem.add(aux);
                }
            }
        }
        else {
            JOptionPane.showMessageDialog(null, "El formato de las reglas
no es compatible con el sistema");
        }
    }

    public void agregaCon(String aux){
        String aux2;
        int count=0;
        if(listaCon.isEmpty()){
                aux=aux.substring(aux.indexOf("=")+1, aux.length());
                listaCon.add(aux);
            }
            else{
                for(int j=0;j<listaCon.size();j++){
                    aux=aux.substring(aux.indexOf("=")+1, aux.length());
                    aux2=listaCon.get(j);
                    if(aux.compareTo(aux2)==0){
                        count++;
                        break;
                    }
                }
                if(count==0){
                    listaCon.add(aux);
                }
            }
    }

    public JComboBox<String> llenaCta(JComboBox<String> aux){
        for(int i=0; i<listaCta.size();i++){
            aux.addItem(listaCta.get(i));
        }

        return aux;
    }

    public JComboBox<String> llenaCur(JComboBox<String> aux){
        for(int i=0; i<listaCur.size();i++){
            aux.addItem(listaCur.get(i));
        }
        return aux;
    }

    public JComboBox<String> llenaTiem(JComboBox<String> aux){
```

```
        for(int i=0; i<listaTiem.size();i++){
            aux.addItem(listaTiem.get(i));
        }

        return aux;
    }

    public JComboBox<String> llenaCon(JComboBox<String> aux){
        for(int i=0; i<listaCon.size();i++){
            aux.addItem(listaCon.get(i));
        }

        return aux;
    }


}
```

ComparaAtributos.java

Aqui se realizan las comparaciones entre las reglas de asociacion ingresadas en un inicio y las opciones de filtrado seleccionadas;

```
package reglas.procesos;

import java.util.LinkedList;
import reglas.clases.Reglas;

public class ComparaAtributos{

    private static LinkedList<String> lista1;
    private static LinkedList<String> lista2;
    private static String aux1,aux2;
    private static float auf1, auf2;

    public ComparaAtributos(){
        lista1 = new LinkedList<String>();
        lista2 = new LinkedList<String>();
    }

    public boolean comparaReglas(Reglas raux, Reglas regla){
        int count=0, hit=0;
        lista1 = raux.getAntecedentes();
        lista2 = regla.getAntecedentes();
        count=lista1.size();
        for(int i=0;i<lista2.size();i++){
            aux2=lista2.get(i).trim();

            if(count==hit){
                break;
            }
            for(int j=0;j<lista1.size();j++){
                aux1=lista1.get(j).trim();
                if(aux1.compareTo(aux2)==0 ||
aux1.compareTo("cualquiera")==0){
```

```
                    hit++;
                    break;
                }
            }
        }
        if(count==hit){
            aux1=raux.getConsecuente().trim();
            aux2=regla.getConsecuente().trim();
            if(aux1.compareTo(aux2)==0 ||
aux1.compareTo("cualquiera")==0){
                auf1=Float.parseFloat(raux.getSupport().trim());
                auf2=Float.parseFloat(regla.getSupport().trim());
                if(auf1<=auf2 || auf1==-1){
                    auf1=Float.parseFloat(raux.getLift().trim());
                    auf2=Float.parseFloat(regla.getLift().trim());
                    if(auf1<auf2 || auf1==-1){

auf1=Float.parseFloat(raux.getConfidence().trim());

auf2=Float.parseFloat(regla.getConfidence().trim());
                        if(auf1<auf2 || auf1==-1){
                            return true;
                        }else return false;
                    }else return false;
                }else return false;
            }else return false;
        }else {
            return false;
        }
    }

    public LinkedList<Integer> contadorConcecuentes(LinkedList<Reglas>
listaReglas){
        int na=0, s=0, b=0, mb=0;
        String aux;
        Reglas reglaAux;
        LinkedList<Integer> listaContador = new LinkedList<Integer>();
        for(int i=0;i<listaReglas.size();i++){
            reglaAux=listaReglas.get(i);
            aux=reglaAux.getConsecuente().trim();
            if(aux.contains("=NA")){
                na++;
            }else if(aux.contains("=S")){
                s++;
            }else if(aux.contains("=B")){
                b++;
            }else if(aux.contains("=MB")){
                mb++;
            }
        }
        listaContador.add(na);
        listaContador.add(s);
        listaContador.add(b);
        listaContador.add(mb);

        return listaContador;
    }
```

```
}
```

GeneraGraficas.java

A partir de las coincidencias que arroja al filtrar, se generan contadores y los elementos que conformaran a las gráficas.

```
package reglas.procesos;

import java.util.LinkedList;
import org.jfree.data.category.DefaultCategoryDataset;
import org.jfree.data.general.DefaultPieDataset;
import org.jfree.data.general.PieDataset;
import org.jfree.data.xy.DefaultXYDataset;
import org.jfree.data.xy.XYDataset;


public class GeneraGrafica {


    public DefaultCategoryDataset  datosBarra(LinkedList<Integer>
listaCon){

        String elementoNa = "NA";
        String elementoS = "S";
        String elementoB = "B";
        String elementoMB = "MB";

        String categoriaCal = "Consecunete";

        int na, s, b, mb;

        na=listaCon.get(0);
        s=listaCon.get(1);
        b=listaCon.get(2);
        mb=listaCon.get(3);

        DefaultCategoryDataset ds = new DefaultCategoryDataset( );

      ds.addValue( na , elementoNa , categoriaCal );
      ds.addValue( s , elementoS , categoriaCal );
      ds.addValue( b, elementoB , categoriaCal );
      ds.addValue( mb , elementoMB , categoriaCal );

      return ds;
    }

    public PieDataset datosPastel(LinkedList<Integer> listaCon){

      DefaultPieDataset ds = new DefaultPieDataset();

      int na, s, b, mb;

      na=listaCon.get(0);
      s=listaCon.get(1);
```

```
      b=listaCon.get(2);
      mb=listaCon.get(3);

      ds.setValue( "NA" , new Double( na ) );
      ds.setValue( "S" , new Double( s ) );
      ds.setValue( "B" , new Double( b ) );
      ds.setValue( "MB" , new Double( mb ) );

      return ds;
   }


}
```

FrExaminar.java

GUI donde el usuario buscara en el ordenador el archivo .csv a analizar;

```
package reglas.frames;

import java.io.File;
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.IOException;
import java.util.logging.Level;
import java.util.logging.Logger;
import reglas.limpieza.LimpiarFuente;
import javax.swing.JFileChooser;
import javax.swing.JOptionPane;

public class FrExaminar extends javax.swing.JFrame {
    JFileChooser seleccionar = new JFileChooser();
    File archivo;
    FileInputStream entrada;
    LimpiarFuente limpieza = new LimpiarFuente();


    public FrExaminar() {
        initComponents();
        setLocationRelativeTo(null);
        this.setTitle("Sistema para el Análisis de Reglas de
Asociación");
    }

    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">
    private void initComponents() {

        btnContinuar = new javax.swing.JButton();
        tfRuta = new javax.swing.JTextField();
        btnExaminar = new javax.swing.JButton();
        lbInstruccion = new javax.swing.JLabel();
        btnSalir = new javax.swing.JButton();


setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
```

```
        setBackground(new java.awt.Color(0, 0, 0));

        btnContinuar.setBackground(new java.awt.Color(205, 3, 46));
        btnContinuar.setForeground(new java.awt.Color(255, 255, 255));
        btnContinuar.setText("Continuar");
        btnContinuar.setCursor(new
java.awt.Cursor(java.awt.Cursor.HAND_CURSOR));
        btnContinuar.addActionListener(new
java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnContinuarActionPerformed(evt);
            }
        });

        btnExaminar.setBackground(new java.awt.Color(205, 3, 46));
        btnExaminar.setForeground(new java.awt.Color(255, 255, 255));
        btnExaminar.setText("Examinar");
        btnExaminar.setCursor(new
java.awt.Cursor(java.awt.Cursor.HAND_CURSOR));
        btnExaminar.addActionListener(new java.awt.event.ActionListener()
{
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnExaminarActionPerformed(evt);
            }
        });

        lbInstruccion.setFont(new java.awt.Font("Lucida Sans Unicode", 0,
14)); // NOI18N
        lbInstruccion.setText("Seleccione el archivo .csv para el
analisis ");

        btnSalir.setBackground(new java.awt.Color(51, 51, 255));
        btnSalir.setForeground(new java.awt.Color(255, 255, 255));
        btnSalir.setText("Salir");
        btnSalir.setCursor(new
java.awt.Cursor(java.awt.Cursor.HAND_CURSOR));
        btnSalir.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnSalirActionPerformed(evt);
            }
        });

        javax.swing.GroupLayout layout = new
javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addGap(28, 28, 28)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
                    .addComponent(lbInstruccion,
javax.swing.GroupLayout.PREFERRED_SIZE, 409,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addGroup(layout.createSequentialGroup()
```

```
                        .addComponent(tfRuta,
javax.swing.GroupLayout.PREFERRED_SIZE, 308,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                        .addComponent(btnExaminar))
                    .addGroup(layout.createSequentialGroup()
                        .addComponent(btnSalir)
                        .addGap(77, 77, 77)
                        .addComponent(btnContinuar)))
                .addContainerGap(19, Short.MAX_VALUE))
        );
        layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addContainerGap()
                .addComponent(lbInstruccion,
javax.swing.GroupLayout.PREFERRED_SIZE, 36,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addGap(37, 37, 37)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BA
SELINE)
                    .addComponent(tfRuta,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(btnExaminar))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 52,
Short.MAX_VALUE)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BA
SELINE)
                    .addComponent(btnContinuar)
                    .addComponent(btnSalir))
                .addGap(24, 24, 24))
        );

        pack();
    }// </editor-fold>

    private void btnContinuarActionPerformed(java.awt.event.ActionEvent
evt) {
        FrObtenDatos llenado = new FrObtenDatos();

        if(tfRuta.getText().isEmpty()){
            JOptionPane.showMessageDialog(null, "Aun no has cargado un
Archivo");
        }else{
            llenado.setVisible(true);
            this.setVisible(false);
            llenado.abrirArchivo();
        }
        try {
            limpieza.procesaArchivo(archivo);
```

```
        } catch (FileNotFoundException ex) {

Logger.getLogger(FrExaminar.class.getName()).log(Level.SEVERE, null, ex);
        }
    }

    private void btnSalirActionPerformed(java.awt.event.ActionEvent evt)
{
        this.dispose();
    }

    private void btnExaminarActionPerformed(java.awt.event.ActionEvent
evt) {
        if(seleccionar.showDialog(null,
"Abrir")==JFileChooser.APPROVE_OPTION){
            archivo=seleccionar.getSelectedFile();
            if(archivo.canRead()){
                if(archivo.getName().endsWith("csv")){
                    try {
                        tfRuta.setText(archivo.getCanonicalPath());
                    } catch (IOException ex) {

Logger.getLogger(FrExaminar.class.getName()).log(Level.SEVERE, null, ex);
                    }
                }else{
                    JOptionPane.showMessageDialog(null, "El archivo no es
compatible, seleccione un archivo .csv");

                }

            }
        }
    }


    public static void main(String args[]) throws FileNotFoundException {
        try {
            for (javax.swing.UIManager.LookAndFeelInfo info :
javax.swing.UIManager.getInstalledLookAndFeels()) {
                if ("Nimbus".equals(info.getName())) {

javax.swing.UIManager.setLookAndFeel(info.getClassName());
                    break;
                }
            }
        } catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(FrExaminar.class.getName()).log(java.u
til.logging.Level.SEVERE, null, ex);
        } catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(FrExaminar.class.getName()).log(java.u
til.logging.Level.SEVERE, null, ex);
        } catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(FrExaminar.class.getName()).log(java.u
til.logging.Level.SEVERE, null, ex);
```

```
        } catch (javax.swing.UnsupportedLookAndFeelException ex) {

java.util.logging.Logger.getLogger(FrExaminar.class.getName()).log(java.u
til.logging.Level.SEVERE, null, ex);
        }
        //</editor-fold>

        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new FrExaminar().setVisible(true);
            }
        });

    }

    // Variables declaration - do not modify
    private javax.swing.JButton btnContinuar;
    private javax.swing.JButton btnExaminar;
    private javax.swing.JButton btnSalir;
    private javax.swing.JLabel lbInstruccion;
    private javax.swing.JTextField tfRuta;
    // End of variables declaration
}
```

FrObtenerDatos.java

GUI donde se seleccionaran las opciones de filtrado.

```
package reglas.frames;

import java.io.BufferedReader;
import java.io.File;
import java.io.FileReader;
import java.util.LinkedList;
import javax.swing.JOptionPane;
import reglas.clases.Reglas;
import reglas.procesos.OperacionesItem;
import reglas.procesos.OperacionesReglas;

public class FrObtenDatos extends javax.swing.JFrame {

   private static Reglas regla;
   private static Reglas raux;
   private OperacionesItem item;
   private OperacionesReglas operaciones;

    public FrObtenDatos() {
        initComponents();
        setLocationRelativeTo(null);
        this.setTitle("Sistema para el Análisis de Reglas de
Asociación");
        regla = new Reglas();
        operaciones= new OperacionesReglas();

    }
```

```
    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">
    private void initComponents() {

        jPanel1 = new javax.swing.JPanel();
        btnRegresar = new javax.swing.JButton();
        jButton2 = new javax.swing.JButton();
        btnSalir = new javax.swing.JButton();
        lbTitulo = new javax.swing.JLabel();
        lbCta = new javax.swing.JLabel();
        lbCur = new javax.swing.JLabel();
        lbAnt3 = new javax.swing.JLabel();
        cbCta = new javax.swing.JComboBox<String>();
        cbCur = new javax.swing.JComboBox<String>();
        cbTiem = new javax.swing.JComboBox<String>();
        lnConsecuente = new javax.swing.JLabel();
        cbConsecuente = new javax.swing.JComboBox<String>();
        lbSupport = new javax.swing.JLabel();
        lbLift = new javax.swing.JLabel();
        lbConfidence = new javax.swing.JLabel();
        txtSupport = new javax.swing.JTextField();
        txtLift = new javax.swing.JTextField();
        txtConfidence = new javax.swing.JTextField();
        jLabel1 = new javax.swing.JLabel();
        jLabel2 = new javax.swing.JLabel();

        javax.swing.GroupLayout jPanel1Layout = new
javax.swing.GroupLayout(jPanel1);
        jPanel1.setLayout(jPanel1Layout);
        jPanel1Layout.setHorizontalGroup(

jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADI
NG)
            .addGap(0, 100, Short.MAX_VALUE)
        );
        jPanel1Layout.setVerticalGroup(

jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADI
NG)
            .addGap(0, 100, Short.MAX_VALUE)
        );


setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);

        btnRegresar.setBackground(new java.awt.Color(51, 51, 255));
        btnRegresar.setForeground(new java.awt.Color(255, 255, 255));
        btnRegresar.setText("Regresar");
        btnRegresar.addActionListener(new java.awt.event.ActionListener()
{
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnRegresarActionPerformed(evt);
            }
        });
```

```
        jButton2.setBackground(new java.awt.Color(205, 3, 46));
        jButton2.setForeground(new java.awt.Color(255, 255, 255));
        jButton2.setText("Aceptar");
        jButton2.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                jButton2ActionPerformed(evt);
            }
        });

        btnSalir.setBackground(new java.awt.Color(51, 51, 255));
        btnSalir.setForeground(new java.awt.Color(255, 255, 255));
        btnSalir.setText("Salir");
        btnSalir.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnSalirActionPerformed(evt);
            }
        });

        lbTitulo.setFont(new java.awt.Font("Lucida Sans Unicode", 0,
16)); // NOI18N
        lbTitulo.setText("Selección de Opciones de Filtrado");

        lbCta.setText("CTA");

        lbCur.setText("CUR_APR");

        lbAnt3.setText("TIEM");

        cbCta.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                cbCtaActionPerformed(evt);
            }
        });

        cbCur.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                cbCurActionPerformed(evt);
            }
        });

        lnConsecuente.setText("Selecciona Consecuente");

        lbSupport.setText("Support mayor a:");

        lbLift.setText("Lift mayor a:");

        lbConfidence.setText("Confidence mayor a:");

        txtSupport.setText("0");

        txtLift.setText("0");

        txtConfidence.setText("0");

        jLabel1.setText("Selecciona Antecedentes");

        jLabel2.setText("CCAL");
```

```
        javax.swing.GroupLayout layout = new
javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
layout.createSequentialGroup()
                .addGap(58, 58, 58)
                .addComponent(btnSalir)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                .addComponent(jButton2)
                .addGap(58, 58, 58))
            .addGroup(layout.createSequentialGroup()
                .addGap(21, 21, 21)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
                    .addGroup(layout.createSequentialGroup()
                        .addComponent(lbTitulo)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 164,
Short.MAX_VALUE)
                        .addComponent(btnRegresar)
                        .addGap(58, 58, 58))
                    .addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TR
AILING)

.addGroup(javax.swing.GroupLayout.Alignment.LEADING,
layout.createSequentialGroup()
                                .addGap(14, 14, 14)
                                .addComponent(lbCur)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                                .addComponent(cbCur,
javax.swing.GroupLayout.PREFERRED_SIZE, 155,
javax.swing.GroupLayout.PREFERRED_SIZE))
                            .addGroup(layout.createSequentialGroup()
                                .addGap(35, 35, 35)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TR
AILING)

.addGroup(layout.createSequentialGroup()
                                        .addComponent(lbAnt3)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                                        .addComponent(cbTiem,
javax.swing.GroupLayout.PREFERRED_SIZE, 155,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addGroup(layout.createSequentialGroup()
```

```
                                        .addComponent(lbCta)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                                        .addComponent(cbCta,
javax.swing.GroupLayout.PREFERRED_SIZE, 155,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addGroup(layout.createSequentialGroup()
                                        .addComponent(jLabel2)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                                        .addComponent(cbConsecuente,
javax.swing.GroupLayout.PREFERRED_SIZE, 155,
javax.swing.GroupLayout.PREFERRED_SIZE)))))
                        .addGap(45, 45, 45)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TR
AILING)
                            .addComponent(lbSupport)
                            .addComponent(lbLift)
                            .addComponent(lbConfidence))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
                            .addComponent(txtConfidence,
javax.swing.GroupLayout.PREFERRED_SIZE, 59,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING, false)
                                .addComponent(txtSupport,
javax.swing.GroupLayout.DEFAULT_SIZE, 59, Short.MAX_VALUE)
                                .addComponent(txtLift)))

.addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
                    .addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
                            .addComponent(lnConsecuente)
                            .addComponent(jLabel1))
                        .addGap(0, 0, Short.MAX_VALUE))))
        );
        layout.setVerticalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addContainerGap()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BA
SELINE)
                    .addComponent(lbTitulo)
                    .addComponent(btnRegresar))
                .addGap(29, 29, 29)
                .addComponent(jLabel1)
```

```
                .addGap(32, 32, 32)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BA
SELINE)
                    .addComponent(lbCta)
                    .addComponent(cbCta,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(lbSupport)
                    .addComponent(txtSupport,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addGap(39, 39, 39)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BA
SELINE)
                    .addComponent(lbCur)
                    .addComponent(cbCur,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(lbLift)
                    .addComponent(txtLift,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addGap(39, 39, 39)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BA
SELINE)
                    .addComponent(lbAnt3)
                    .addComponent(cbTiem,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(lbConfidence)
                    .addComponent(txtConfidence,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addGap(62, 62, 62)
                .addComponent(lnConsecuente)
                .addGap(29, 29, 29)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BA
SELINE)
                    .addComponent(jLabel2)
                    .addComponent(cbConsecuente,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 51,
Short.MAX_VALUE)
```

```
.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BA
SELINE)
                    .addComponent(jButton2)
                    .addComponent(btnSalir))
                .addGap(38, 38, 38))
        );

        pack();
    }// </editor-fold>

    public void abrirArchivo(){

      File archivo = null;
      FileReader fr = null;
      BufferedReader br = null;
      regla = new Reglas();
      item = new OperacionesItem();

      try {
         archivo = new File ("test/ReglasClean.txt");
         fr = new FileReader (archivo);
         br = new BufferedReader(fr);
         LinkedList<String> lista = new LinkedList<String>();
         String linea,aux;
         cbCta.addItem("cualquiera");
         cbCur.addItem("cualquiera");
         cbTiem.addItem("cualquiera");
         cbConsecuente.addItem("cualquiera");
         while((linea=br.readLine())!=null){
            lista = item.obtenerAntecedentes(linea);
            aux = item.obtenerConcecunete(linea);
            item.agregaCon(aux);

            for(int i=0;i<lista.size();i++){
                aux=lista.get(i);
                item.agregarOpcionesFiltrado(aux);
            }
            }
         cbCta=item.llenaCta(cbCta);
         cbCur=item.llenaCur(cbCur);
         cbTiem=item.llenaTiem(cbTiem);
         cbConsecuente=item.llenaCon(cbConsecuente);

      }
      catch(Exception e){
      }finally{
         try{
            if( null != fr ){
               fr.close();
            }
         }catch (Exception e2){
         }
      }

    }
```

```
    private void cbCurActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
    }

    private void cbCtaActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
    }

    private void btnRegresarActionPerformed(java.awt.event.ActionEvent
evt) {
        FrExaminar regresa = new FrExaminar();
        regresa.setVisible(true);
        this.setVisible(false);
    }

    private void btnSalirActionPerformed(java.awt.event.ActionEvent evt)
{
        this.dispose();

    }

    private void jButton2ActionPerformed(java.awt.event.ActionEvent evt)
{
        FrResultado resultado = new FrResultado();
        LinkedList<String> lista = new LinkedList<String>();
        raux=new Reglas();
        String aux;
        OperacionesReglas operaciones = new OperacionesReglas();

        aux=(String)cbCta.getSelectedItem();
        aux=aux.trim();
        if(aux.compareTo("cualquiera")!=0){
            lista.add("CTA="+aux);
        }

        aux=(String)cbCur.getSelectedItem();
        aux=aux.trim();
        if(aux.compareTo("cualquiera")!=0){
            lista.add("CUR_APR="+aux);
        }

        aux=(String)cbTiem.getSelectedItem();
        aux=aux.trim();
        if(aux.compareTo("cualquiera")!=0){
            lista.add("TIEM="+aux);
        }
        if(lista.isEmpty()){
            JOptionPane.showMessageDialog(null, "Selecciona al menos una
opcion del filtrado");
        }
        else{
            raux.setAntecedentes(lista);
        }
        aux=(String)cbConsecuente.getSelectedItem();
        aux=aux.trim();
```

```
        if(aux.compareTo("cualquiera")==0){
            raux.setConsecuente(aux);
        }else
            raux.setConsecuente("CCAL="+aux);

        //raux.setConsecuente((String) cbConsecuente.getSelectedItem());
        raux.setSupport(txtSupport.getText());
        raux.setLift(txtLift.getText());
        raux.setConfidence(txtConfidence.getText());

        resultado.setLabel(raux);
        this.dispose();
        resultado.setVisible(true);
        this.setVisible(false);
        //operaciones.asignaFiltro(raux);
        //System.out.println(raux.toString() + " al presionar
continuar");

    }


    public static void main(String args[]) {

        try {
            for (javax.swing.UIManager.LookAndFeelInfo info :
javax.swing.UIManager.getInstalledLookAndFeels()) {
                if ("Nimbus".equals(info.getName())) {

javax.swing.UIManager.setLookAndFeel(info.getClassName());
                    break;
                }
            }
        } catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(FrObtenDatos.class.getName()).log(java
.util.logging.Level.SEVERE, null, ex);
        } catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(FrObtenDatos.class.getName()).log(java
.util.logging.Level.SEVERE, null, ex);
        } catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(FrObtenDatos.class.getName()).log(java
.util.logging.Level.SEVERE, null, ex);
        } catch (javax.swing.UnsupportedLookAndFeelException ex) {

java.util.logging.Logger.getLogger(FrObtenDatos.class.getName()).log(java
.util.logging.Level.SEVERE, null, ex);
        }

        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new FrObtenDatos().setVisible(true);
            }
        });
    }
```

```
    // Variables declaration - do not modify
    private javax.swing.JButton btnRegresar;
    private javax.swing.JButton btnSalir;
    private javax.swing.JComboBox<String> cbConsecuente;
    private javax.swing.JComboBox<String> cbCta;
    private javax.swing.JComboBox<String> cbCur;
    private javax.swing.JComboBox<String> cbTiem;
    private javax.swing.JButton jButton2;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JLabel jLabel2;
    private javax.swing.JPanel jPanel1;
    private javax.swing.JLabel lbAnt3;
    private javax.swing.JLabel lbConfidence;
    private javax.swing.JLabel lbCta;
    private javax.swing.JLabel lbCur;
    private javax.swing.JLabel lbLift;
    private javax.swing.JLabel lbSupport;
    private javax.swing.JLabel lbTitulo;
    private javax.swing.JLabel lnConsecuente;
    private javax.swing.JTextField txtConfidence;
    private javax.swing.JTextField txtLift;
    private javax.swing.JTextField txtSupport;
    // End of variables declaration
}
```

FrResultado.java

GUI que presenta las opciones que fueron seleccionadas a la hora del filtrado, además de mostrar las gráficas resultantes.

```
package reglas.frames;

import java.awt.Dimension;
import java.io.BufferedWriter;
import java.io.File;
import java.io.FileWriter;
import java.util.LinkedList;
import javax.swing.JFileChooser;
import javax.swing.JOptionPane;
import org.jfree.chart.ChartFactory;
import org.jfree.chart.ChartPanel;
import org.jfree.chart.JFreeChart;
import org.jfree.data.category.DefaultCategoryDataset;
import org.jfree.data.general.PieDataset;
import reglas.clases.Reglas;
import reglas.procesos.GeneraGrafica;

import reglas.procesos.OperacionesReglas;

public class FrResultado extends javax.swing.JFrame {
    private LinkedList<Reglas> listaReglas;
    private LinkedList<Integer> listaCon;
```

```
    public FrResultado() {
        initComponents();
        setLocationRelativeTo(null);
        this.setTitle("Sistema para el Análisis de Reglas de
Asociación");
        this.setResizable(false);
        asignarRadioButtons();
        listaReglas = new LinkedList<Reglas>();
        listaCon= new LinkedList<Integer>();


    }

    public void obtenContador(LinkedList<Integer> listaCon){
        this.listaCon=listaCon;
    }

    public void asignarRadioButtons(){
        bgGraficas.add(rbBarras);
        bgGraficas.add(rbPastel);
    }

     private void crearBarra(){
        jpGraficas.removeAll();
        GeneraGrafica datos = new GeneraGrafica();
        DefaultCategoryDataset dataset = datos.datosBarra(this.listaCon);
        int nCoincidencias;

        nCoincidencias=
this.listaCon.get(0)+this.listaCon.get(1)+this.listaCon.get(2)+this.lista
Con.get(3);


        JFreeChart barChart = ChartFactory.createBarChart(
         "Número total de coincidencias=" +
nCoincidencias,"","Coincidencias",
         dataset);

        ChartPanel cp = new ChartPanel(barChart);
        cp.setVisible(true);
       Dimension dim = jpGraficas.getSize();

                cp.setBounds(2, 2, dim.width, dim.height);
                jpGraficas.add(cp);
                repaint();
    }

     private void crearPastel(){
        jpGraficas.removeAll();
        GeneraGrafica datos = new GeneraGrafica();
        PieDataset dataset = datos.datosPastel(this.listaCon);
        int nCoincidencias;

        nCoincidencias=
this.listaCon.get(0)+this.listaCon.get(1)+this.listaCon.get(2)+this.lista
Con.get(3);
```

```
        JFreeChart pieChart = ChartFactory.createPieChart(
         "Número total de coincidencias" + nCoincidencias,
         dataset);

        ChartPanel cp = new ChartPanel(pieChart);
        cp.setVisible(true);
               Dimension dim = jpGraficas.getSize();

                cp.setBounds(2, 2, dim.width, dim.height);
                jpGraficas.add(cp);
                repaint();
    }

    public void setLabel(Reglas raux){
        OperacionesReglas operaciones = new OperacionesReglas();
        LinkedList<String> lista = new LinkedList<String>();
        LinkedList<LinkedList> listaDeListas = new
LinkedList<LinkedList>();
        lista = raux.getAntecedentes();
        listaDeListas=operaciones.procesa(raux);
        this.listaCon=listaDeListas.get(1);
        this.listaReglas=listaDeListas.get(0);
        rbBarras.setSelected(true);
        crearBarra();
        int tam = lista.size();

        if(tam==1){
           lbA1.setText(lista.get(0));
        }else if(tam==2){
           lbA1.setText(lista.get(0));
           lbA2.setText(lista.get(1));
        }else{
            lbA1.setText(lista.get(0));
            lbA2.setText(lista.get(1));
            lbA3.setText(lista.get(2));
        }
        lbConse.setText(raux.getConsecuente());
        lbSup.setText(raux.getSupport());
        lbLift.setText(raux.getLift());
        lbConf.setText(raux.getConfidence());


    }



    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">
    private void initComponents() {

        bgGraficas = new javax.swing.ButtonGroup();
        btnSalir = new javax.swing.JButton();
        btnExportar = new javax.swing.JButton();
        btnAtras = new javax.swing.JButton();
        antecedenteslb = new javax.swing.JLabel();
        lbA1 = new javax.swing.JLabel();
```

```
        lbA2 = new javax.swing.JLabel();
        jLabel4 = new javax.swing.JLabel();
        jLabel5 = new javax.swing.JLabel();
        jLabel6 = new javax.swing.JLabel();
        jLabel7 = new javax.swing.JLabel();
        lbConse = new javax.swing.JLabel();
        lbSup = new javax.swing.JLabel();
        lbLift = new javax.swing.JLabel();
        lbConf = new javax.swing.JLabel();
        lbA3 = new javax.swing.JLabel();
        jLabel1 = new javax.swing.JLabel();
        rbBarras = new javax.swing.JRadioButton();
        rbPastel = new javax.swing.JRadioButton();
        jpGraficas = new javax.swing.JPanel();
        btActualiza = new javax.swing.JButton();


setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);

        btnSalir.setText("Salir");
        btnSalir.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnSalirActionPerformed(evt);
            }
        });

        btnExportar.setText("Exportar");
        btnExportar.addActionListener(new java.awt.event.ActionListener()
{
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnExportarActionPerformed(evt);
            }
        });

        btnAtras.setText("Atras ");
        btnAtras.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnAtrasActionPerformed(evt);
            }
        });

        antecedenteslb.setFont(new java.awt.Font("Tahoma", 3, 14)); //
NOI18N
        antecedenteslb.setText("Antecedentes");

        jLabel4.setFont(new java.awt.Font("Tahoma", 3, 14)); // NOI18N
        jLabel4.setText("Consecuente: ");

        jLabel5.setFont(new java.awt.Font("Tahoma", 3, 14)); // NOI18N
        jLabel5.setText("Suppor mayor a:");

        jLabel6.setFont(new java.awt.Font("Tahoma", 3, 14)); // NOI18N
        jLabel6.setText("Lift mayor a:");

        jLabel7.setFont(new java.awt.Font("Tahoma", 3, 14)); // NOI18N
        jLabel7.setText("Confidence maro a:");
```

```
        jLabel1.setFont(new java.awt.Font("Tahoma", 0, 24)); // NOI18N
        jLabel1.setText("Opciones de filtrado seleccionadas");

        rbBarras.setText("Barras");

        rbPastel.setText("Pastel");
        rbPastel.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                rbPastelActionPerformed(evt);
            }
        });


jpGraficas.setBorder(javax.swing.BorderFactory.createBevelBorder(javax.sw
ing.border.BevelBorder.RAISED));

        javax.swing.GroupLayout jpGraficasLayout = new
javax.swing.GroupLayout(jpGraficas);
        jpGraficas.setLayout(jpGraficasLayout);
        jpGraficasLayout.setHorizontalGroup(

jpGraficasLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
            .addGap(0, 445, Short.MAX_VALUE)
        );
        jpGraficasLayout.setVerticalGroup(

jpGraficasLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
            .addGap(0, 392, Short.MAX_VALUE)
        );

        btActualiza.setText("Actualiza");
        btActualiza.addActionListener(new java.awt.event.ActionListener()
{
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btActualizaActionPerformed(evt);
            }
        });

        javax.swing.GroupLayout layout = new
javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addGap(32, 32, 32)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
                    .addGroup(layout.createSequentialGroup()
                        .addComponent(jLabel1)
                        .addGap(79, 79, 79)
                        .addComponent(rbBarras)
                        .addGap(48, 48, 48)
                        .addComponent(rbPastel)
```

```
.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                        .addComponent(btActualiza))
                    .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
layout.createSequentialGroup()
                        .addComponent(btnSalir)
                        .addGap(109, 109, 109)
                        .addComponent(btnAtras)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                        .addComponent(btnExportar))
                    .addGroup(layout.createSequentialGroup()
                        .addComponent(antecedenteslb)
                        .addGap(0, 0, Short.MAX_VALUE))
                    .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
layout.createSequentialGroup()
                        .addGap(0, 0, Short.MAX_VALUE)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
                            .addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
                                    .addComponent(jLabel4)
                                    .addComponent(jLabel5)
                                    .addComponent(jLabel6)
                                    .addComponent(jLabel7))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
                                    .addComponent(lbConf)
                                    .addComponent(lbLift)
                                    .addComponent(lbSup)
                                    .addComponent(lbConse)))
                            .addGroup(layout.createSequentialGroup()
                                .addGap(37, 37, 37)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
                                    .addComponent(lbA3)
                                    .addComponent(lbA2)
                                    .addComponent(lbA1))))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 106,
Short.MAX_VALUE)
                        .addComponent(jpGraficas,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)))
                .addGap(29, 29, 29))
        );
        layout.setVerticalGroup(
```

```
layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LE
ADING)
                    .addGroup(layout.createSequentialGroup()
                        .addGap(16, 16, 16)
                        .addComponent(jLabel1)
                        .addGap(66, 66, 66)
                        .addComponent(antecedenteslb)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                        .addComponent(lbA1)
                        .addGap(19, 19, 19)
                        .addComponent(lbA2)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 30,
Short.MAX_VALUE)
                        .addComponent(lbA3)
                        .addGap(38, 38, 38)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BA
SELINE)
                            .addComponent(jLabel4)
                            .addComponent(lbConse))
                        .addGap(33, 33, 33)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BA
SELINE)
                            .addComponent(jLabel5)
                            .addComponent(lbSup))
                        .addGap(33, 33, 33)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BA
SELINE)
                            .addComponent(jLabel6)
                            .addComponent(lbLift))
                        .addGap(33, 33, 33)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BA
SELINE)
                            .addComponent(jLabel7)
                            .addComponent(lbConf))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 105,
Short.MAX_VALUE))
                    .addGroup(layout.createSequentialGroup()
                        .addGap(34, 34, 34)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BA
SELINE)
                            .addComponent(rbBarras)
                            .addComponent(rbPastel)
                            .addComponent(btActualiza))
```

```
.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                        .addComponent(jpGraficas,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addGap(18, 18, 18)))

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BA
SELINE)
                    .addComponent(btnSalir)
                    .addComponent(btnExportar)
                    .addComponent(btnAtras))
                .addGap(32, 32, 32))
        );

        pack();
    }// </editor-fold>

    private void btnAtrasActionPerformed(java.awt.event.ActionEvent evt)
{

        FrObtenDatos llenado = new FrObtenDatos();
        llenado.setVisible(true);
        this.setVisible(false);
        llenado.abrirArchivo();
    }

    private void btnSalirActionPerformed(java.awt.event.ActionEvent evt)
{

        this.dispose();
    }

    private void rbPastelActionPerformed(java.awt.event.ActionEvent evt)
{

    }

    private void btActualizaActionPerformed(java.awt.event.ActionEvent
evt) {
       if(rbBarras.isSelected()){
           crearBarra();
       }else if (rbPastel.isSelected()){
           crearPastel();
       }
    }

    private void btnExportarActionPerformed(java.awt.event.ActionEvent
evt) {
        JFileChooser guardarComo = new JFileChooser();
        guardarComo.setApproveButtonText("Guardar");
        guardarComo.showSaveDialog(null);
        Reglas raux = null;
        File archivo = new File(guardarComo.getSelectedFile()+".txt");
        try {
```

```
            BufferedWriter salida = new BufferedWriter(new
FileWriter(archivo));
            for(int i=0;i<listaReglas.size();i++){
                raux=listaReglas.get(i);
                salida.write(raux.toString());
                salida.newLine();
            }
            salida.close();
        } catch (Exception e) {
        }
        JOptionPane.showMessageDialog(null, "El archivo se guardo con
éxito");
    }

    public static void main(String args[]) {

        try {
            for (javax.swing.UIManager.LookAndFeelInfo info :
javax.swing.UIManager.getInstalledLookAndFeels()) {
                if ("Nimbus".equals(info.getName())) {

javax.swing.UIManager.setLookAndFeel(info.getClassName());
                    break;
                }
            }
        } catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(FrResultado.class.getName()).log(java.
util.logging.Level.SEVERE, null, ex);
        } catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(FrResultado.class.getName()).log(java.
util.logging.Level.SEVERE, null, ex);
        } catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(FrResultado.class.getName()).log(java.
util.logging.Level.SEVERE, null, ex);
        } catch (javax.swing.UnsupportedLookAndFeelException ex) {

java.util.logging.Logger.getLogger(FrResultado.class.getName()).log(java.
util.logging.Level.SEVERE, null, ex);
        }
        //</editor-fold>

        /* Create and display the form */
        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new FrResultado().setVisible(true);
            }

        });
    }

    // Variables declaration - do not modify
    private javax.swing.JLabel antecedenteslb;
    private javax.swing.ButtonGroup bgGraficas;
    private javax.swing.JButton btActualiza;
```

```
    private javax.swing.JButton btnAtras;
    private javax.swing.JButton btnExportar;
    private javax.swing.JButton btnSalir;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JLabel jLabel4;
    private javax.swing.JLabel jLabel5;
    private javax.swing.JLabel jLabel6;
    private javax.swing.JLabel jLabel7;
    private javax.swing.JPanel jpGraficas;
    private javax.swing.JLabel lbA1;
    private javax.swing.JLabel lbA2;
    private javax.swing.JLabel lbA3;
    private javax.swing.JLabel lbConf;
    private javax.swing.JLabel lbConse;
    private javax.swing.JLabel lbLift;
    private javax.swing.JLabel lbSup;
    private javax.swing.JRadioButton rbBarras;
    private javax.swing.JRadioButton rbPastel;
    // End of variables declaration
}
```