Apéndice A

Manuales

A.1. Manual de despliegue e instalación

A.1.1. Prerrequisitos

Se requiere contar con:

- Windows 10 con arquitectura de 64 bits. [1]
- Java (versión 8 o superior) JRE. No es necesario el JDK.
- En el caso de utilizar *tags*, se debe utilizar la versión 0.9.289 de `gMaster`. [16]

(TEMPORAL)Se debe tener en cuenta que si se va a utilizar Vensim, (OJO si se puede afirmar que funciona a partir de la 7.XX). También se deberá tener cuidado no manipular con la distribución DSS archivos creados con la distribución PLE, y viceversa.

[1] Para comprobar la arquitectura del sistema: `https://www.computerhope.com/issues/ch001121.htm#:~:text=Press%20and%20hold%20the%20Windows,running%20the%2064%2Dbit%20version.`

A.1.2. Descarga del software

La solución desarrollada consta de tres archivos `.jar` que deberán descargarse de `http://gitlab-locomotion.infor.uva.es/testing-locomotion-infrastructure/semanticmerge/-/tree/Interfaces/Executables`. Estos archivos son:

- `ViewNameAndDelimiterAdder.jar`
- `DelimiterEraser.jar`
- `VensimPlugin4SemanticMerge.jar`

Los dos primeros son aplicaciones auxiliares que se explicarán más adelante. El tercero es la solución principal, desarrollada para facilitar el trabajo con archivos `Vensim` en el control de versiones.

Se debe crear una carpeta para guardar estos tres archivos. En este manual asumiremos que dicha carpeta es `C:\Users\{User}\VensimPlugin4SM`, donde `{User}` debe ser sustituido por el nombre del usuario en Windows. En el caso de guardar los archivos en otra carpeta, se recomienda altamente que la ruta absoluta de dicha carpeta no contenga espacios en blanco y/o caracteres especiales.

A.1.3. Interacciones

Este proyecto no trabaja por sí solo (*standalone*) sino integrado como *plugin* con herramientas externas.

Para poder utilizarlo el usuario necesitará hacerlo a través de alguna de las siguientes aplicaciones: `SemanticMerge` [38] o `PlasticSCM` [42] o `gmaster` [34] previamente a los siguientes pasos de la instalación del proyecto.

La primera de estas tres, `SemanticMerge`, es una herramienta independiente del control de versiones. La funcionalidad de SemanticMerge está integrada tanto en `PlasticSCM` como en `gmaster`, y no es necesario descargarla por separado. Su funcionalidad está integrada en las otras dos herramientas.

Para una guía de instalación de alguna de estas aplicaciones debe dirigirse a las páginas indicadas en las referencias.

A.1.4. Configuración para el uso con SemanticMerge

Si el producto desarrollado se va a utilizar con `SemanticMerge`, se debe tener en cuenta que se trata de un software con licencia. Se debe adquirir o solicitar una clave de acceso, o bien, utilizar la prueba gratuita de 30 días.

Para utilizar herramientas externas utilizando puramente la aplicación de `SemanticMerge`, se debe utilizar la línea de comandos de Windows, a través de las aplicaciones de `CMD`, `Powershell` o cualquier otra aplicación externa que emule una terminal.

El comando a utilizar debe seguir la estructura que se muestra en el siguiente cuadro de texto, indicando la ruta al ejecutable de la herramienta `SemanticMerge`, la ruta a sendos ficheros a comparar, la ruta a la herramienta externa desarrollada en este proyecto, y la ruta a la máquina virtual de `Java`, la cual no es más que el ejecutable de `Java`, y si se ha seguido instalaciones normales, la ruta debería ser idéntica o muy similar. Se recomienda no utilizar espacios en blanco y/o caracteres especiales en las rutas que contienen los archivos necesarios para utilizar la herramienta.

```
C:\Users\{User}\AppData\Local\semanticmerge.\semanticmergetool.exe
--source={absolute path}/example.mdl
--destination={absolute path}/example.mdl
--externalparser="-jar {absolute path}/VensimPlugin4SemanticMerge.jar"
--virtualmachine="C:\Program Files\Java\\jdk-11.0.8\bin\java.exe"
```

A.1.5. Configuración para el uso con PlasticSCM

(TEMPORAL) TODO hablar violeta

A.1.6. Configuración para el uso con `gmaster`

Actualmente `gMaster` es gratuito. En un futuro se planifica una versión gratuita para uso no comercial.

Según donde tengamos instalado dichas aplicaciones, deberemos dirigirnos al directorio

Si suponemos una instalación típica de `gMaster`, se encontrará una carpeta `gmaster`, en la ruta predeterminada: `C:\Users\{User}\AppData\Local\`, donde `{User}` debe ser sustituido por el nombre del usuario en Windows.

Dentro de esa carpeta se encontrará una denominada `/config`. Se debe crear en dicha carpeta un archivo con nombre `externalparsers.conf`, si no existe anteriormente. Este archivo tiene la función de enlazar formatos no detectados de forma predefinida con sus correspondientes herramientas externas para `SemanticMerge`.

En nuestro caso, debemos asociar la extensión `.mdl` a la ruta absoluta del *plugin* para `Vensim`. Concretamente, el contenido del archivo `externalparsers.conf` será similar al siguiente bloque de texto.

```
.mdl=java -jar C:\Users\{User}\VensimPlugin4SM\VensimPlugin4SemanticMerge.jar
```

Se recomienda no incluir espacios en blanco o caracteres especiales en las carpetas que conforman la ruta donde se encuentra alojado `VensimPlugin4SemanticMerge.jar`.

A.2. Manual de usuario

Una vez descargados todos los archivos y configurada la integración con `SemanticMerge` o `PlasticSCM` o `gmaster` se puede comenzar a trabajar con la herramienta.

Este manual se centrará en describir el uso integrado en `gmaster`. El proceso hace uso de los tres programas listados anteriormente, y aunque se explica en profundidad cada programa en las subsecciones siguientes, a continuación se muestra un diagrama que expone el flujo de trabajo habitual con la herramienta, es decir, fusionar dos ramas. Figura A.1:

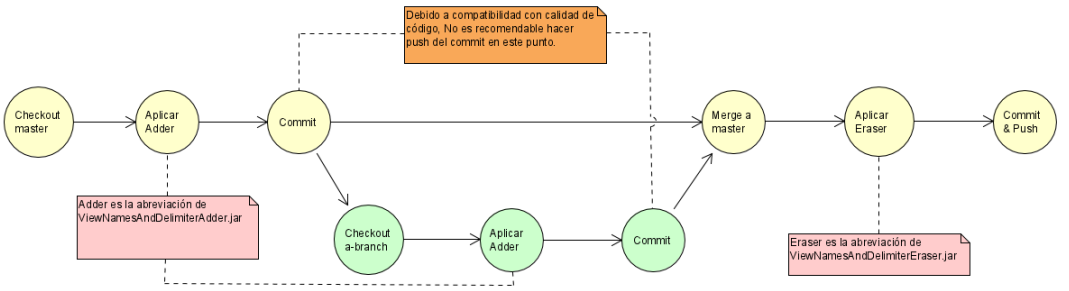

Figura A.1: Diagrama general de la herramienta en la fusión de ramas.

Existe otro flujo de trabajo habitual, donde se quiere comprobar los cambios entre versiones o *commits* en una misma rama. Dicho flujo de trabajo se explica en la siguiente figura, figura A.2. Es importante resaltar que no se debe hacer *push* de los *commits* intermedios, pues esto podría causar problemas con el control de calidad para los archivos `Vensim`.

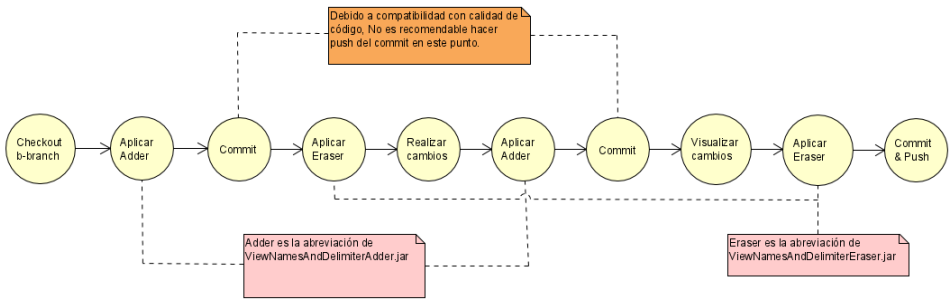

Figura A.2: Diagrama general de la herramienta en la diferenciación de commits.

A.2.1. El programa auxiliar ViewNameAndDelimiterAdder

En un principio, deberemos seleccionar los ficheros `Vensim` que queremos utilizar y modificarlos a través del primer programa que modifica los archivos (`ViewNameAndDelimiterAdder.jar`), en este caso, añadir los nombres de las vistas en las ecuaciones y añadir los delimitadores. Estos delimitadores son necesarios para que la herramienta externa `VensimParser` sea capaz de leer los archivos `Vensim` y diferenciar sus partes para que a la hora de resolver un conflicto sea mucho más fácil identificar las causas del mismo. Es importante señalar que una vez aplicado este programa, `Vensim` no será capaz de leer el archivo de forma gráfica. Para ello, se debe utilizar el programa `DelimiterEraser`, el cual se explica más adelante. Las dos interfaces gráficas de los programas `ViewNameAndDelimiterAdder` y `DelimiterEraser` son prácticamente idénticas, por lo que se debe comprobar el nombre de la etiqueta del programa. Tras procesar el archivo, se mantendrá un archivo a modo de *backup* del archivo sin procesar con extensión `.2mdl` y el archivo procesado conservará tanto el nombre como la extensión. Este *backup* como los explicados posteriormente se realiza de forma automática y no requiere de acciones del usuario.

La interfaz inicial, tal y como se aprecia el la figura A.3, consta de únicamente un botón. Dicho botón, `Open file`, como su propio nombre indica, sirve para abrir el archivo. Sabremos que el archivo se ha cargado cuando la interfaz se modifique tal y como se muestra en la figura A.4. En caso de abrir un fichero cuyo formato no es `.mdl` o cualquier otro error, se notificará al usuario a través de esa etiqueta con un llamativo color rojo. En caso de seleccionar un archivo al que ya se la haya aplicado dicho formateo, de nuevo se le indicará al usuario. Una vez cargado el archivo, se debe pulsar el botón de la izquierda, `Process`, el cual procederá a modificar el archivo y conservar el *backup* con la extensión `.2mdl`. Cuando haya finalizado el proceso, se notificará al usuario con un mensaje en color verde. En caso de haber seleccionado el archivo erróneo, se podrá hacer uso del botón de `Cancel` para volver a la interfaz inicial, figura A.3. Si queremos comprobar la ruta completa del fichero que hemos seleccionado,

podremos pulsar en el botón de `Absolute path` para mostrar dicha ruta. Pulsando de nuevo recuperaremos el nombre del archivo sin ruta. Finalmente, y con el objetivo de hacer la experiencia de usuario más clara y concisa, se explicará en la parte inferior de la interfaz que efecto tiene procesar un fichero.

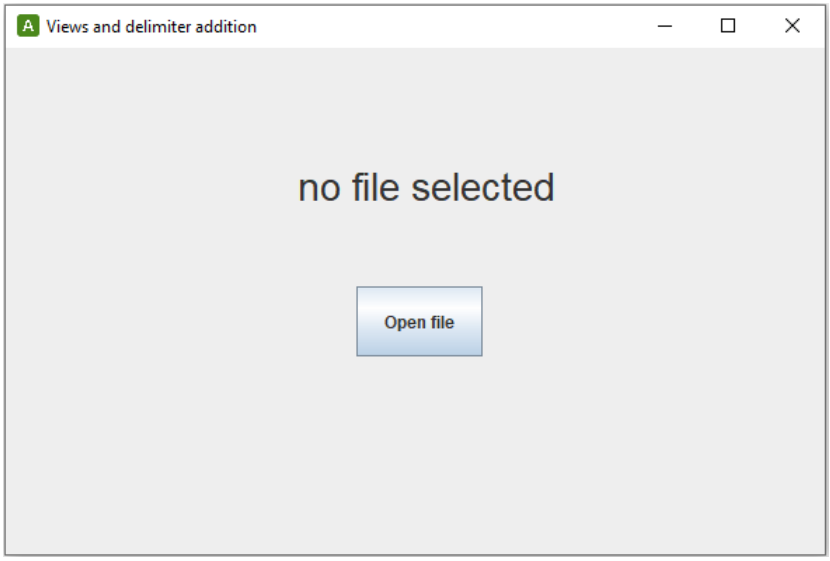

Figura A.3: Pantalla principal de la interfaz de adición de nombres de las vistas y delimitadores.

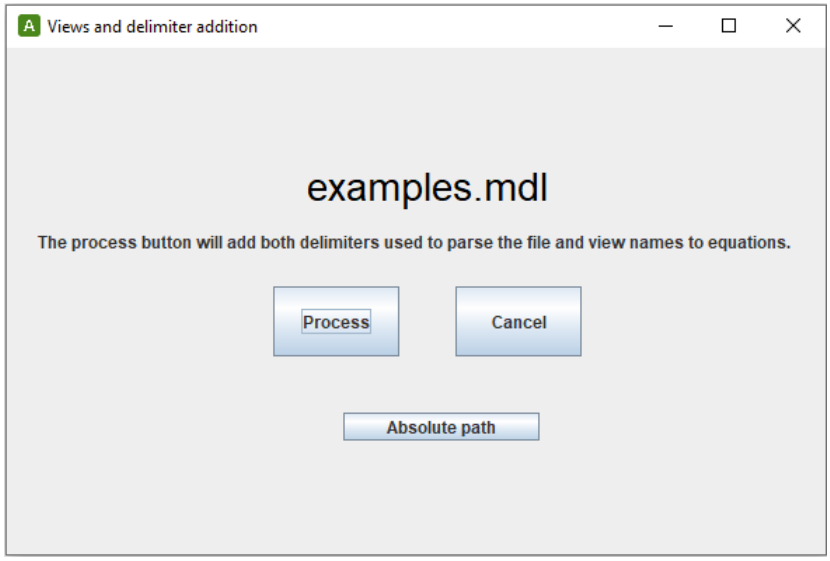

Figura A.4: Pantalla secundaria de la interfaz de adición de nombres de las vistas y delimitadores.

A.2.2. El plugin de Vensim para SemanticMerge dentro de gmaster

Una vez transformados los archivos deseados, se subirán al repositorio. Es importante modificar los archivos, puesto que si no, `gmaster` no será capaz de reconocer los archivos **semánticamente**, sino únicamente como texto plano. Es decir, se conservará la capacidad de lectura convencional de la herramienta, pero no se aplicarán las mejoras que esta herramienta aporta. Tras ello, se procederá bien a crear otra rama o bien a modificar el archivo directamente en la misma rama para posteriormente realizar un nuevo *commit*. Cuando se haya modificado el archivo, se procederá a abrir la herramienta de `gmaster`. Esta detectará que la extensión del archivo es `.mdl` y utilizará la herramienta externa `VensimParser` para realizar el análisis semántico. La herramienta nos mostrará las diferencias entre los dos archivos, proporcionándonos información sobre qué significa cada línea modificada. Cuando hayamos fusionado los archivos correctamente, podremos hacer *commit* & *push*.

En la siguiente imagen, figura A.5, se pueden apreciar las diferentes partes de la interfaz gráfica de `gmaster`. En la primera sección señalada podemos observar la primera nueva característica que trae la herramienta. Se pueden observar el recuento de cambios entre dos versiones de un mismo archivo. En dicha sección se puede observar que los cambios se distinguen en cinco apartados principales: adiciones, modificaciones, eliminaciones, movimientos y renombres. Dentro de cada una de estas secciones, a su vez podemos distinguir la categoría del elemento modificado, como puede ser una ecuación y una vista, así como su nombre. De nuevo incorporado de forma novedosa, en la segunda sección se observan las dos versiones del archivo que se está analizando, mostrando en la parte izquierda la versión más antigua y en la derecha la actual. Existen franjas coloreadas que relacionan los cambios visualmente entre ambas versiones. Al lado de cada cambio aparecerá una inicial, la cual indicará el tipo de modificación de entre las cinco posibles explicadas anteriormente. El primer recuadro en color azul de esta sección, situado a la izquierda, sirve para cambiar entre el modo convencional de control de versiones con texto plano, y el modo semántico propio de `SemanticMerge`. El segundo recuadro situado a la derecha, permite ver los cambios de una forma más visual, utilizando únicamente los grafismos asociados a los cinco tipos posibles de modificaciones. Por último, en la tercera sección se aprecian los archivos modificados en la versión del proyecto o *commit* seleccionado.

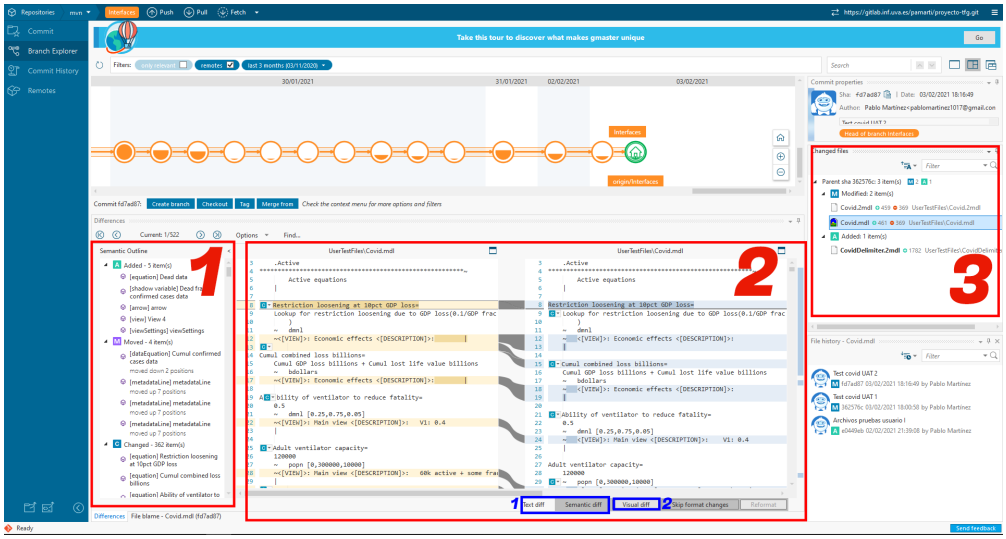

Figura A.5: Interfaz de gmaster y sus correspondientes partes.

A.2.3. El programa auxiliar DelimiterEraser

Una vez tengamos los ficheros finales, deberemos utilizar el segundo programa, `DelimiterEraser` para eliminar los delimitadores de secciones que se utilizan para hacer más fácil el análisis semántico del archivo así como los nombres de las vistas en los comentarios de las ecuaciones. De no hacerlo, no se podrá abrir correctamente el archivo con la interfaz gráfica de `Vensim`. El proceso es el mismo que el del primer programa (`ViewNameAndDelimiterAdder`), y de nuevo, se debe prestar atención a utilizar aquella con el fondo gris predeterminado, puesto que las dos interfaces son prácticamente idénticas. De nuevo, al realizar la transformación de archivos, conservaremos a modo de *backup* el archivo antes de modificarse con formato `NombreDeArchivoDelimiter.2mdl` en el mismo directorio donde se encuentra el archivo a modificar. Dichas interfaces se muestran en la figuras A.6 y A.7.

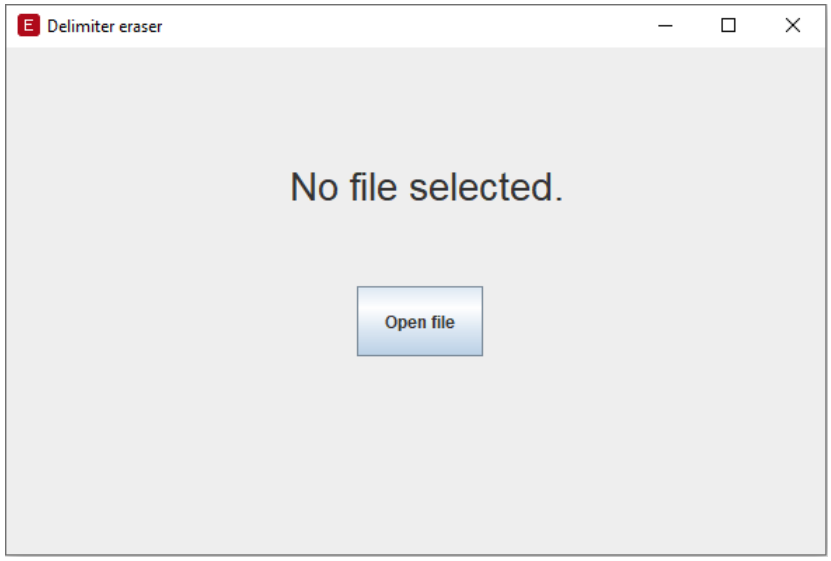

Figura A.6: Pantalla principal de la interfaz de eliminación de delimitadores.

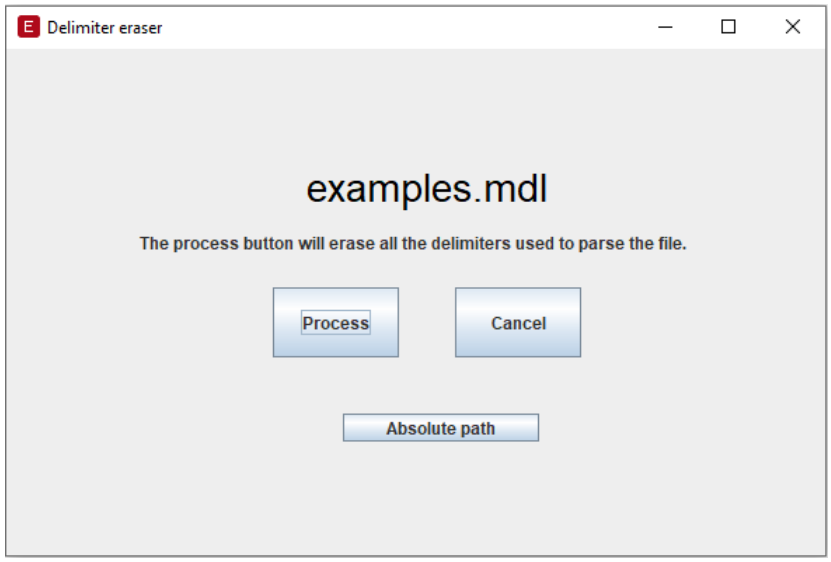

Figura A.7: Pantalla secundaria de la interfaz de eliminación de delimitadores.

A.2.4. Enlaces de interés.

En caso de querer conocer más acerca del funcionamiento específico de `SemanticMerge` y/o `gmaster` o cómo ambas herramientas implementan herramientas o *parsers* externos, se recomienda consultar [24], [34], [36] y [38] en la bibliografía del proyecto.

Apéndice B

Resumen de enlaces adicionales

Los enlaces útiles de interés en este Trabajo Fin de Grado son:

- Repositorio del proyecto: `https://gitlab.inf.uva.es/pamarti/proyecto-tfg`.

Estructura del repositorio (carpeta de la memoria)

Bibliografía

[1] andrew1234. Java swing jfilechooser. https://www.geeksforgeeks.org/java-swing-jfilechooser/, 2018. Accesed: 2021-19-01.

[2] Apache. Maven. https://maven.apache.org/, 2020. Accesed: 2020-9-23.

[3] Baeldung. Java with antlr. https://www.baeldung.com/java-antlr, 2020. Accesed: 2020-9-15.

[4] Daniel Bazaco. Final vensim grammar in antlr 4, developed for this project. https://gitlab.inf.uva.es/danbaza/tfg-sonarvensim/-/blob/master/src/main/antlr/Model.g4, 2019. Accesed: 2020-9-15.

[5] Rocket Chat. Rocket chat. https://rocket.chat/, 2020. Accesed: 2020-9-23.

[6] CoronA. Antlr 4.5 - mismatched input 'x' expecting 'x'. https://stackoverflow.com/questions/29777778/antlr-4-5-mismatched-input-x-expecting-x, 2015. Accesed: 2020-9-18.

[7] Departamento de Informática. Archivos cloud vensim. https://cloud.infor.uva.es/index.php/s/3r85R4oavbe3ivk, 2020. Accesed: 2020-9-30.

[8] Universidad de Valladolid. Guía docente de la signatura. trabado de fin de grado (mención ingeniería de software). https://www.inf.uva.es/wp-content/uploads/2016/06/G46976.pdf, 2020. Accesed: 2020-10-2.

[9] Universidad de Valladolid. Preguntas frecuentes. https://www.uva.es/export/sites/uva/2.docencia/2.02.mastersoficiales/2.02.13.preguntasfrecuentes/index.html, 2020. Accesed: 2020-10-2.

[10] eclass. ¿cuáles son los eventos de scrum? https://blog.eclass.com/cuales-son-los-eventos-de-scrum-conocelos-aqui-0, 2020. Accesed: 2021-22-01.

[11] Norberto Figuerola. Riesgos: Plan de mitigación vs plan de contingencia vs fallback plan. https://articulospm.files.wordpress.com/2015/06/riesgos-plan-mitigacion-vs-plan-contingencia-vs-fallback-plan.pdf, 2015. Accesed: 2020-9-26.

[12] Tobi G. Java special characters in regex. https://stackoverflow.com/questions/14134558/list-of-all-special-characters-that-need-to-be-escaped-in-a-regex, 2019. Accesed: 2020-11-11.

[13] Research Gate. Locomotion h2020. https://www.locomotion-h2020.eu/, 2020. Accesed: 2020-9-16.

[14] Research Gate. Medeas. https://www.medeas.eu/#home, 2020. Accesed: 2020-9-16.

[15] GitLab.org. Gitlab. https://gitlab.com/gitlab-org, 2020. Accesed: 2020-9-23.

[16] gMasterRealeaseNotes. Release notes. https://gmaster.io/releasenotes/0.9.471.0/-6, 2021. Accesed: 2021-11-02.

[17] Object Management Group. Unified modeling language. https://www.uml.org/, 2021. Accesed: 2021-11-02.

[18] James P. Houghton. test-models. https://github.com/SDXorg/test-models, 2020. Accesed: 2020-9-30.

[19] Joel Francia Huambachano. ¿qué es scrum? https://www.scrum.org/resources/blog/que-es-scrum, 2017. Accesed: 2021-22-01.

[20] Change Vision Inc. Astah professional. https://astah.net/products/astah-professional/, 2021. Accesed: 2021-11-02.

[21] La información. ¿cuál es el coste real de tener un trabajador para una empresa? https://www.lainformacion.com/practicopedia/cual-es-el-coste-real-de-un-trabajador-para-una-empresa/6491464/#:~:text=Cotizaciones%20a%20la%20Seguridad%20Social,26.300%20euros%20a%20la%20empresa., 2019. Accesed: 2020-9-26.

[22] Jitsi.org. Jitsi meet. https://meet.jit.si/, 2020. Accesed: 2020-9-23.

[23] Bart Kiers. If/else statements in antlr using listeners. https://stackoverflow.com/questions/15610183/if-else-statements-in-antlr-using-listeners, 2013. Accesed: 2020-9-15.

[24] Sergio L. Using external parsers with gmaster. http://blog.gmaster.io/2018/03/using-external-parsers-with-gmaster.html, 2018. Accesed: 2020-14-11.

[25] Pablo Santos Luaces. charposition. https://github.com/PlasticSCM/charposition, 2016. Accesed: 2020-31-10.

[26] Pablo Martínez López. Gramática en antlr4 para vensim. https://gitlab.inf.uva.es/pamarti/proyecto-tfg/-/blob/master/src/main/antlr4/es/uva/inf/grammar/Grammar.g4, 2020. Accesed: 2020-9-29.

[27] Pablo Martínez López. Simplejavaparser-semanticmerge. https://github.com/HylianPablo/SimpleJavaParser-SemanticMerge, 2020. Accesed: 2020-9-29.

[28] McGraw-Hill. Business dynamics. http://www.mhhe.com/business/opsci/sterman/models.mhtml, 2001. Accesed: 2020-9-30.

[29] Microsoft. Visual studio code. https://code.visualstudio.com/, 2020. Accesed: 2020-9-23.

[30] migraciones y seguridad social Ministerio de trabajo. Boletín oficial del estado. iii. otras disposiciones. https://www.boe.es/boe/dias/2019/06/03/pdfs/BOE-A-2019-8222.pdf, 2019. Accesed: 2020-9-26.

[31] MiryamGSM. External parser sample. https://github.com/PlasticSCM/external-parser-sample/tree/master-SCM20098, 2017. Accesed: 2020-8-28.

[32] picodotdev. Cómo ejecutar un proceso del sistema con java. https://picodotdev.github.io/blog-bitix/2016/03/como-ejecutar-un-proceso-del-sistema-con-java/, 2016. Accesed: 2020-16-12.

[33] PlasticSCM. Semanticmerge pricing. https://users.semanticmerge.com/Checkout, 2020. Accesed: 2020-9-26.

[34] Plastic SCM. gmaster - git gui. https://gmaster.io/, 2021. Accesed: 2021-21-01.

[35] Shodor. Vensim models. http://shodor.org/talks/ncsi/vensim/, 2005. Accesed: 2020-9-19.

[36] Códice Software. External parsers. https://semanticmerge.com/documentation/external-parsers/external-parsers-guide, 2016. Accesed: 2020-9-19.

[37] Códice Software. Advanced version control - pocket guide. https://www.plasticscm.com/documentation/advanced-version-control-guide#two-way-merge, 2020. Accesed: 2020-9-17.

[38] Códice Software. Semanticmerge 2.0. https://semanticmerge.com/, 2020. Accesed: 2020-7-11.

[39] Códice Software. Semanticmerge features. http://www.semanticmerge.com/features, 2020. Accesed: 2020-9-17.

[40] Códice Software. Semanticmerge intro guide. https://semanticmerge.com/documentation/intro-guide/semanticmerge-intro-guide, 2020. Accesed: 2020-9-17.

[41] Códice Software. Xdiff and xmerge. https://www.plasticscm.com/features/xmerge, 2020. Accesed: 2020-9-17.

[42] Códice Software. Plasticscm - the distributed version control for big projects. https://www.plasticscm.com/, 2021. Accesed: 2021-2-10.

[43] Saumitra Srivastav. Antlr4 - visitor vs listener pattern. https://saumitra.me/blog/antlr4-visitor-vs-listener-pattern/, 2017. Accesed: 2020-9-18.

[44] Johannes Link Matthias Merdes Marc Philipp Juliette de Rancourt Christian Stein Stefan Bechtold, Sam Brannen. Junit 5 user guide. https://junit.org/junit5/docs/current/user-guide/, 2020. Accesed: 2020-23-12.

[45] Ventana Systems. Defined and shadow variables. https://www.vensim.com/documentation/22890.htm, 2020. Accesed: 2020-10-6.

[46] Ventana Systems. Sketch information. https://www.vensim.com/documentation/ref_sketch_format.htm, 2020. Accesed: 2020-9-30.

[47] Ventana Systems. Sketch objects. https://www.vensim.com/documentation/index.html?23275.htm, 2020. Accesed: 2020-10-3.

[48] Ventana Systems. Variable types. https://www.vensim.com/documentation/ref_variable_types.htm, 2020. Accesed: 2020-9-16.

[49] Ventana Systems. Vensim. https://vensim.com/, 2020. Accesed: 2020-23-12.

[50] Linus Torvalds. Git. https://git-scm.com/, 2020. Accesed: 2020-9-23.

[51] TutorialsPoint. Compiler design - symbol table. https://www.tutorialspoint.com/compiler_design/compiler_design_symbol_table.htm#:~:text=Symbol%20table%20is%20an%20important,synthesis%20parts%20of%20a%20compiler., 2020. Accesed: 2020-10-7.

[52] Wangdq. How to display antlr tree gui. https://stackoverflow.com/questions/23809005/how-to-display-antlr-tree-gui, 2014. Accesed: 2020-9-24.

[53] Wikipedia. Swing (biblioteca gráfica). https://es.wikipedia.org/wiki/Swing_(biblioteca_gr%C3%A1fica), 2021. Accesed: 2021-20-01.