



**ESCUELA POLITÉCNICA
SUPERIOR DE CÓRDOBA**
Universidad de Córdoba



TRABAJO FIN DE GRADO

Grado en Ingeniería Informática

Aplicación Conversor de Cuestionarios entre XML Test de Moodle y QTI/IMS de CANVAS

Autor

Diego Alejandro Higueta Grisales

Director(es)

Prof. D. Cristóbal Romero Morales

Mayo, 2023



UNIVERSIDAD DE CÓRDOBA



Índice general

1. Introducción	1
1.1. MoodleXML	2
1.2. QTI/IMS	2
1.3. Partes del proyecto	3
2. Definición del problema	5
2.1. Identificación del problema real	5
2.2. Identificación del problema técnico	5
2.2.1. Funcionamiento	6
2.2.2. Entorno	6
2.2.3. Vida esperada	7
2.2.4. Ciclo de mantenimiento	7
2.2.5. Competencia	8
2.2.6. Aspecto externo	8
2.2.7. Estandarización	8
2.2.8. Calidad y fiabilidad	9
2.2.9. Programa de tareas	9
2.2.10. Seguridad	10
3. Objetivos	11
4. Antecedentes	13
4.1. Respondus	13
4.2. Plugins Moodle	13
4.3. QTIVIEWER/CCReader	14
4.4. GetMarked.ia	14
4.5. Código abierto	14
5. Restricciones	17
5.1. Factores Dato	17
5.2. Factores Estratégicos	17
5.2.1. Lenguaje de programación	18
5.2.2. Entorno de desarrollo	18
5.2.3. Desarrollo de la documentación	19
6. Recursos	21
6.1. Humanos	21
6.2. Hardware	21
6.3. Software	21

7. Análisis del sistema	23
7.1. Descripción funcional	23
7.1.1. Diagrama de casos de uso	23
7.1.2. Análisis del caso de uso Ver Detalle de la Solución	26
7.1.3. Análisis del caso de uso Configurar Parámetros de Visualización . .	27
7.2. Descripción de la información	28
7.2.1. Parámetros de un problema	28
7.2.2. Soluciones de un problema	30
7.3. Descripción de la interfaz	32
7.3.1. Representación gráfica de una distribución en planta	32
7.3.2. Representación gráfica del flujo de materiales	32
7.3.3. Representación gráfica de la adyacencia	35
7.3.4. Representación gráfica de la relación de aspecto	37
7.3.5. Representación de textos	40
7.3.6. Representación gráfica del espacio vacío	40
7.4. Especificación de Requisitos	41
7.4.1. Requisitos Funcionales	41
7.4.2. Requisitos de Información	42
7.4.3. Requisitos de Interfaz	42
7.4.4. Requisitos No Funcionales	42
8. Diseño del sistema	43
8.1. Modelo de datos	43
8.2. Modelo arquitectónico	45
8.2.1. Clases	45
8.2.2. Estructuración del código	47
8.2.3. Flujos de datos	50
8.3. Interfaz de usuario	53
9. Pruebas	55
9.1. Pruebas de caja blanca	55
9.2. Pruebas de caja negra	55
9.3. Pruebas de integración	56
9.4. Pruebas de representación	56
9.4.1. Caso de prueba 01	56
9.4.2. Caso de prueba 02	58
9.4.3. Caso de prueba 03	59
9.4.4. Caso de prueba 04	60
9.4.5. Caso de prueba 05	61
9.4.6. Caso de prueba 06	62
9.4.7. Caso de prueba 07	63
9.4.8. Caso de prueba 08	64
9.4.9. Caso de prueba 09	65
9.5. Matriz de cumplimiento	66
10. Conclusiones	67
11. Futuras mejoras	69

12. Planificación del proyecto	71
A. Manual de usuario	75
A.1. Introducción	75
A.2. Instalación	75
A.2.1. Crear entorno Virtual	75
A.2.2. Instalación de paquetes	76
A.2.3. Descargar aplicación	76
A.2.4. Configuración y sincronización de la base de datos	77
A.2.5. Lanzar el proyecto	77
A.2.6. Posibles errores	77
A.3. Entorno virtual preconfigurado	78
A.4. Uso de la aplicación web	78
A.4.1. Pantalla de Inicio	78
A.4.2. Home - Problemas	79
A.4.3. Configuraciones	81
A.4.4. Ejecuciones	83
A.4.5. Soluciones	84
A.5. Uso del lanzador	86
A.6. Desinstalación	86
B. Manual de código	87
B.1. Introducción	87
B.2. Organización del código	88
B.3. Ficheros	91
B.3.1. Ficheros existentes modificados	91
B.3.2. Ficheros nuevos	124

Índice de figuras

7.1. Diagrama de casos de uso	24
7.2. Ejemplo del fichero de parámetros	29
7.3. Ejemplo del fichero de soluciones	31
7.4. Distribución en planta	32
7.5. Comparación distancias Euclídea y Manhattan	33
7.6. Distancia Euclídea por color	33
7.7. Distancia Euclídea por grosor	34
7.8. Distancia Manhattan por color	34
7.9. Distancia Manhattan por grosor	35
7.10. Línea de unión por color	36
7.11. Línea de unión por grosor	36
7.12. Resaltado del lado en común por color	37
7.13. Resaltado del lado en común por grosor	37
7.14. Trapecio para la relación de aspecto deseable	38
7.15. Grado de cumplimiento del aspect ratio deseable con un color	39
7.16. Grado de cumplimiento del aspect ratio deseable con dos colores	39
7.17. Resaltado del espacio vacío con color	40
7.18. Resaltado del espacio vacío con puntos	41
8.1. Resaltado del espacio vacío con puntos	45
8.2. Diagrama de clases	46
8.3. Esquema de estructuración del código	47
8.4. Diagrama de secuencia Visualizar solución	50
8.5. Diagrama de secuencia Cargar parámetros	51
8.6. Diagrama de secuencia Guardar parámetros	52
8.7. Interfaz. Vista previa del detalle de la solución	53
8.8. Interfaz. Modal de configuración de parámetros de visualización	54
9.1. Flujo mediante distancia Euclídea y color de las líneas	57
9.2. Flujo mediante distancia Euclídea y grosor de las líneas	57
9.3. Flujo mediante distancia de Manhattan y color de las líneas	58
9.4. Flujo mediante distancia de Manhattan y grosor de las líneas	58
9.5. Adyacencia mediante resaltado de lado común y color de las líneas	59
9.6. Adyacencia mediante resaltado de lado común y grosor de las líneas	60
9.7. Adyacencia mediante línea de unión y color de las líneas	60
9.8. Adyacencia mediante línea de unión y grosor de las líneas	61
9.9. Relación de aspecto mediante un color	61
9.10. Espacio vacío mediante fondo liso	62
9.11. Espacio vacío mediante estampado de puntos	62

9.12. Opciones de Almacenar/Cargar configuración	63
9.13. Seleccionar instalaciones	64
9.14. Mostrar flujos y adyacencias de las instalaciones seleccionadas	64
9.15. Modificar colores de los distintos elementos	65
12.1. Diagrama de Gantt	72
A.1. Manual de usuario. Inicio	78
A.2. Manual de usuario. Login	79
A.3. Manual de usuario. Problemas	79
A.4. Manual de usuario. Subir problema desde fichero	80
A.5. Manual de usuario. Problema creado	80
A.6. Manual de usuario. Configuraciones	81
A.7. Manual de usuario. Gestor de configuraciones	82
A.8. Manual de usuario. Configuraciones creadas	83
A.9. Manual de usuario. Ejecuciones en espera	83
A.10. Manual de usuario. Soluciones	84
A.11. Manual de usuario. Visualización de la solución	84
A.12. Manual de usuario. Menú de configuración de parámetros de visualización .	85
A.13. Organización del código	86
B.1. Organización del código	88

Índice de cuadros

7.1. Especificación caso de uso Ver Detalle de la Solución	26
7.2. Especificación caso de uso Configurar Parámetros de Visualización	27
8.1. Sumario de datos	44
9.1. Ejemplo matriz de flujos	56
9.2. Ejemplo matriz de adyacencias	59
9.3. Matriz de cumplimiento de requisitos	66
12.1. Distribución temporal	71

Capítulo 1

Introducción

Los grandes cambios que hemos vivido en la sociedad y las personas en los últimos tiempos se han visto reflejados tanto en nuestros hábitos, como formas de consumir formación. En concreto el impulso del e-learning y la formación online, superando los obstáculos de tiempo y espacio, donde se utilizan diferentes plataformas como herramientas de aprendizaje, las cuales han revolucionado la forma en que educadores y estudiantes interactúan y colaboran en entornos educativos virtuales. Entre las plataformas online para el aprendizaje más extendidas a nivel mundial se encuentran Moodle y Canvas LMS.

En este contexto, aunque la tecnología ha evolucionado rápidamente facilitándonos hacer varias cosas de nuestro día a día, existen los desafíos de (*la evaluación para el aprendizaje*) y cómo implementarla en un ambiente online. Las evaluaciones son usadas para determinar el conocimiento de los estudiantes y su dominio en un tema, así como también para identificar áreas de mejora y así poder adaptar la clase o hacer cursos personalizados.

Una de las prácticas mas usadas para la evaluación online son *los cuestionarios*, esta evaluación se realiza de forma dinámica gracias a los ordenadores que pueden realizar la tarea de selección de preguntas y la posterior evaluación en un instante. Una característica única de los cuestionarios en línea es que, el orden de las preguntas y las opciones de respuestas se pueden presentar de forma aleatoria, para que los estudiantes no reciban el mismo cuestionario. Cualquier cuestionario usado en una clase tradicional puede ser fácilmente adaptado en un cuestionario en línea. Las actividades de calificar los cuestionarios y comparar el desempeño de los estudiantes son muy fáciles de hacer, ya que se llevan a cabo de forma automática por el sistema. Además, los educadores tienen la opción de proporcionar aclaraciones adicionales, instrucciones o explicaciones detalladas en forma de comentarios junto a cada pregunta del cuestionario o cada opción de respuesta. Estos comentarios personalizados pueden ayudar a los estudiantes a comprender mejor el propósito de la pregunta, brindar orientación adicional sobre cómo abordarla y ofrecer

explicaciones claras sobre las opciones de respuesta. [? ?]

Las preguntas de tipo cuestionario pueden tener diferentes estructuras atendiendo a la forma de solicitar la respuesta del usuario y a como e muestra la información o datos de la pregunta. Existen diferentes tipos de cuestionarios, entre los mas utilizados son, verdadero o falso, identificar una o varias respuestas correctas, rellenar espacios en blanco , unión de columnas (emparejamiento), respuestas numéricas entre otros tipos. Los usuarios pueden organizar estas preguntas en bancos de preguntas y utilizarlos para crear pruebas personalizadas. Estos cuestionarios son creados para ser usados en una plataforma o sistema educativo de forma que en muchas ocasiones no pueden ser reutilizados en otras plataformas, ya que no existe un único estándar. Es este el caso vamos hablar de *MoodleXML* de Moodle y *QTI/IMS* usado en Canvas LMS.

1.1. MoodleXML

Moodle XML es un formato de archivo utilizado en el sistema de gestión del aprendizaje Moodle. Moodle es un sistema de gestión del aprendizaje de código abierto que brinda una amplia gama de herramientas para facilitar la enseñanza y el aprendizaje en línea, principalmente en entornos educativos. El formato Moodle XML se utiliza para importar y exportar datos de Moodle, lo que permite a los usuarios compartir contenido, actividades y configuraciones entre diferentes instalaciones de Moodle o con otros sistemas compatibles con el formato XML.

El archivo Moodle XML contiene una estructura jerárquica de datos que representa el contenido de un curso o elementos específicos dentro de un curso, como actividades, cuestionarios, recursos y configuraciones. Puede incluir información sobre el nombre, descripción, valoraciones y otros atributos relacionados con los elementos del curso. Los archivos Moodle XML se pueden crear y editar utilizando herramientas específicas de Moodle, como el editor de actividades de Moodle o mediante herramientas de creación de contenido compatible con Moodle XML. [?]

1.2. QTI/IMS

El formato QTI/IMS es un estándar de intercambio de datos utilizados en sistemas de evaluación educativa en línea. QTI significa Interoperabilidad de Pruebas y Cuestionarios.^{en} inglés, e IMS (IMS Global Learning Consortium) es una organización sin fines de lucro que desarrolla estándares abiertos para la educación y la tecnología. Canvas es compatible con el formato QTI/IMS (IMS QTI), lo que significa que importa y exporta preguntas y pruebas en este formato. Canvas es un sistema de gestión del aprendizaje (LMS) utilizado

por muchas instituciones educativas y organizaciones para la creación, entrega y gestión de cursos en línea.

El formato QTI/IMS utiliza XML (Extensible Markup Language) como lenguaje de marcado para describir la estructura y los atributos de las preguntas y pruebas. Esto permite que el contenido se presente de manera clara y estructurada, lo que facilita su comprensión y reutilización en otras herramientas compatibles con el formato QTI/IMS sin perder su estructura y características. Canvas cuenta con su herramienta de banco de preguntas (Question Bank), que permite a los usuarios crear preguntas y pruebas. [?]

1.3. Partes del proyecto

El proyecto en su totalidad estará formado por 3 manuales: el manual técnico, un manual de usuario y el manual de código.

Este, en concreto, es el manual técnico, documento encargado de explicar detalladamente desde el problema en cuestión, hasta la solución que se le va a dar, pasando por los recursos que se van a utilizar para ello. Su estructura se puede dividir en cinco partes, que a su vez se segmentarán en diversos capítulos:

- **Parte I: Introducción al proyecto.** Aspectos generales del proyecto.
 - *Capítulo 1: Introducción.* Presentación breve del problema que se pretende solucionar y su origen, mostrando las definiciones de aquellos términos que se utilizan frecuentemente a lo largo del manual.
 - *Capítulo 2: Definición del problema.* Definición de los problemas real y técnico de forma clara y concisa.
 - *Capítulo 3: Objetivos.* Relación de los objetivos que se pretenden alcanzar en la aplicación, explicando las funciones que debe cumplir lo que se diseña y apuntando las posibles vías de solución.
 - *Capítulo 4: Antecedentes.* Información relativa a aquellos métodos que tratan de resolver el problema de distribución en planta y los proyectos realizados anteriormente.
 - *Capítulo 5: Restricciones.* Limitaciones en el ámbito del diseño que condicionan la elección de una u otra alternativa, distinguiendo entre los factores dato y los estratégicos.
 - *Capítulo 6: Recursos.* Recursos humanos que intervienen en la elaboración del proyecto, y recursos materiales de hardware y software disponibles que se utilizan para llevarlo a cabo.

- **Parte II: Análisis.** Especificación de lo que debe hacer la aplicación desarrollada.
 - *Capítulo 7: Análisis del sistema.* Análisis del funcionamiento que deberá tener el sistema, la información a manejar, lo que deberá ofrecer la interfaz y los requisitos a satisfacer.
- **Parte III: Diseño.**
 - *Capítulo 8: Diseño del sistema*
- **Parte IV: Pruebas y resultados**
 - *Capítulo 9: Pruebas.* Pruebas a las que ha sido sometido el software durante su desarrollo y pruebas de rendimiento cuando se ha finalizado.
- **Parte V: Conclusiones y Futuras Mejoras**
 - *Capítulo 10: Conclusiones.* Mirada hacia atrás para determinar qué objetivos se han cumplido, cuáles no, y el porqué.
 - *Capítulo 11: Futuras mejoras.* A pesar de que inicialmente todo proyecto trata de ser ambicioso y conseguir un software inmejorable, esto casi nunca se cumple, por lo que este capítulo intentará ser una crítica constructiva para mejorar lo realizado y ampliarlo en el futuro.
 - *Capítulo 12: Distribución temporal.* Plan de trabajo y calendario seguido para el desarrollo del proyecto.

Capítulo 2

Definición del problema

En este apartado se va definir con claridad el problema a resolver. En primer lugar, se identificará el *Problema Real* al que nos enfrentamos desde un punto de vista externo (de un usuario) y de manera superficial. Y, en segundo lugar, se identificará el *Problema Técnico* desde una perspectiva más centrada en su desarrollo mediante la técnica conocida como PDS (*Product Desing Specification*).

2.1. Identificación del problema real

Por un lado Canvas exporta sus cuestionarios en formatos IMS/QTI, pero Moodle no acepta la importación de este formato. Moodle por su lado, recomienda el Formato Moodle XML para importar preguntas, ya que éste permite que se importen la mayor cantidad de datos de la pregunta [?]. Por lo que se necesita una herramienta de transformación entre estos formatos. Así éste Trabajo de Fin de Grado se centrará en el desarrollo de una *aplicación conversora de formatos*, que permita a cualquier profesor exportar e importar los cuestionarios online entre ambas plataformas de aprendizaje y así poder llevar a cabo la reutilización de los bancos de preguntas de otros formatos, así como las preguntas generadas en diferentes formatos (y por lo consiguiente en diferentes plataformas), de forma que se almacene toda la información necesaria para su reutilización.

2.2. Identificación del problema técnico

Siguiendo el esquema de la PDS, a continuación, se expondrán los aspectos técnicos del proyecto con la intención de obtener una descripción formal del problema a resolver.

2.2.1. Funcionamiento

Nos vamos a centrar únicamente en el funcionamiento del módulo de representación. Así pues, se comportará de la siguiente manera:

1. Una vez la aplicación haya generado la conversión, podrá importar directamente el cuestionario en la plataforma correspondiente.
2. Cada cuestionario tendrá su estructura distinta según el tipo, con sus respectivos nombres, marcas y flujo de respuestas.
3. Se guardará las conversiones en una carpeta que estará en la misma localización donde se encuentra el fichero a convertir.
4. Se dará al usuario la opción de elegir el cuestionario a convertir, y cual será el formato final, de QTI/IMS a MoodleXML ó de MoodleXML a QTI/IMS.
5. Debe realizar un tratamiento de errores que evite salidas inesperadas del sistema, evitando la generación de ficheros de salida incoherentes.
6. Y debe ser fácil de usar.

2.2.2. Entorno

En el análisis del entorno de la aplicación que se pretende desarrollar se tendrá en cuenta los siguientes puntos de vista principalmente: entorno de programación, entorno software, entorno de usuario y entorno físico o de trabajo.

- *Entorno de programación:* La aplicación será accesible desde el escritorio, es decir, no hará falta el acceso a Internet ni a navegador. El lenguaje de programación que se utilizará en la implementación es Python haciendo uso de librerías simples y eficientes para analizar y crear datos XML, como lo son `xml.etree.ElementTree` ó `lxml.objectify`.
- *Entorno Software:* Para el correcto funcionamiento de la aplicación serán necesarios los siguientes componentes software:
 - Python 3.7
 - Se recomienda la utilización como sistema operativo cualquier distribución estable del sistema operativo Linux ya que es donde se ha desarrollado el programa, aunque se podrá también ejecutar en Windows y Mac.

- *Entorno de Usuario:* La aplicación que se desarrollará deberá ser lo más intuitiva posible, de modo que pueda ser utilizada por todo tipo de usuarios, aunque no posean conocimientos informáticos.
- *Entorno físico o de trabajo:* Para ejecutarlas, los usuarios deben descargar sus archivos e instalarlos en su máquina. Su instalación no requiere grandes requisitos ya que no ocupa demasiado espacio, pero para su ejecución debe de tenerse en cuenta que debe tener espacio suficiente para guardar los ficheros a convertir y los ficheros finales, convertidos.

2.2.3. Vida esperada

Como se ha comentado, al tratarse de formatos de cuestionarios dependientes de las plataformas que los usan, la vida esperada estará ligada a la aceptación de la estructuras de dichos formatos en las plataformas o a futuras versiones de estos. Por tanto hemos de tener en cuenta que mientras se siga avanzando el tema en cuestión, nos puede llevar a cambiar la metodología que ha sido empleada en el desarrollo de este proyecto, en un plazo de tiempo que puede ser breve o extenso.

2.2.4. Ciclo de mantenimiento

En caso de que el módulo se considere de utilidad durante un período largo de tiempo este deberá ser llevado a cabo por programadores informáticos y se hará atendiendo a las necesidades generadas, pudiendo ser de tres tipos:

- *Perfeccionamiento:* Se basará en mejorar los aspectos que se consideren oportunos o crear nuevas funcionalidades a la aplicación, requeridas por los usuarios del sistema o por los autores de cursos. Estas mejoras pueden consistir en aumentar tipos de cuestionarios a convertir, facilitar el mantenimiento del sistema para posibles cambios futuros, etc.
- *Adaptación:* Conjunto de actividades que se realizarán para adaptar la aplicación al entorno tecnológico, como se comentaba en el apartado anterior, adaptarlo a posibles cambios en el formato de las estructuras que usa para generar las conversiones.
- *Corrección:* Corregir errores que puedan aparecer en su uso diario, no descubiertos hasta el momento, como pueden ser fallos de procesamiento o de implementación.

2.2.5. Competencia

El problema que abordamos es conocido por aquellos usuarios de ambas plataformas que han tenido que rehacer los cuestionarios ya que existen pocas soluciones efectivas ó de pago. Una máxima en el desarrollo de software es no reinventar la rueda, por eso nuestra idea no yace en aportar algo que ya existe; nuestra idea es ofrecer algo que destaque y cumplir con los requisitos y expectativas del usuario de manera efectiva y eficiente. Pensamos que los cuestionarios online son un elemento muy potente para la evaluación de un alumno, eso unido a la posibilidad de crear varios tipos de preguntas y poderlas usar en diferentes plataforma se reflejará en una experiencia de usuario que marcará la diferencia.

En el punto de antecedentes se mencionarán algunas herramientas similares que hayan servido para llevar a cabo este proyecto.

2.2.6. Aspecto externo

En cuanto a la interfaz gráfica de usuario (GUI) se pretende que tenga un diseño atractivo, intuitivo y sencillo de manejar, de forma que el usuario no tenga que leer el manual de usuario ya que los botones seguirán un patrón coherente en toda la aplicación de escritorio, expresando así cada uno su función clara y concisamente, para poder obtener las conversiones sin muchas complicaciones, que es donde radica la importancia de este proyecto.

2.2.7. Estandarización

El visualizador está programado en Python haciendo uso de la librería *tkinter*. Librería bastante extendida y documentada, que no debe plantear, en principio, ningún problema. El código fuente será lo suficientemente claro y legible, estando correctamente documentado para que en un futuro cualquier programador pueda entenderlo y realizar los cambios y/o mejoras que crea convenientes.

Los formatos que vamos a convertir, ambos están estandarizados, por un lado MoodleXML, este formato se basa en XML (eXtensible Markup Language) este lenguaje está completamente estandarizado por el W3C (World Wide Web Consortium), por lo que tampoco deberá suponer ningún problema para la ejecución. Además se aplican otros como: SCORM (Sharable Content Object Reference Model) y IMS Common Cartridge. Por otro lado, QTI es un estándar de IMS (Interoperability Standards for Educational Technology) que se utiliza para describir preguntas y pruebas en formato electrónico. QTI permite la creación y entrega de evaluaciones en línea y su posterior evaluación automática. Se aplican otros como IMS Content Packaging o IMS Metadata.

2.2.8. Calidad y fiabilidad

El objetivo es garantizar un alto nivel de excelencia y confiabilidad llevando a cabo exhaustivas pruebas, con diferentes tipos de preguntas, para identificar posibles puntos o componentes críticos, con el fin de minimizar la probabilidad de errores durante el funcionamiento del software. En caso de que los usuarios introduzcan errores (por ejemplo, con ficheros no validos), se busca gestionarlos adecuadamente.

2.2.9. Programa de tareas

Fase Inicial

A lo largo de esta fase se llevarán a cabo las actividades relacionadas con el estudio para la elaboración del proyecto, que son:

1. Estudio del lenguaje de desarrollo python.
2. Estudio del estándar XML.
3. Estudio del estándar QTI
4. Estudio de las librerías xml.etree.ElementTree y objectify para el manejo de ficheros de lenguajes de marca.
5. Estudio de sistemas educativos y aplicaciones gestoras de cuestionarios.

Fase de Diseño y Desarrollo

En esta fase de ingeniería se procederá al diseño y desarrollo de la solución software, y entre las tareas se encuentran:

1. Estudio de las variables que participan en el sistema:
 - a) Variedad de estructuras XML.
 - b) Variedad de tipos de preguntas.
 - c) Parámetros modificables.
 - d) Verificación y conservación de la información de los cuestionarios.
2. Diseño y desarrollo del sistema.
3. Programación de las funcionalidades.
4. Integración de las funcionalidades en la interfaz
5. Comprobar que el formato resultante es válido

Fase de Prueba y Documentación

En esta fase se llevarán a cabo las actividades correspondientes a las pruebas de la solución desarrollada. Estas pruebas se pueden clasificar en dos categorías.

1. Prueba de Unidad o de Caja blanca: Este tipo de pruebas se realizará durante el periodo que se dedique a la codificación de la aplicación, en la que irán surgiendo diversos errores y se deberán ir corrigiendo mediante la aplicación de pruebas de este tipo.
2. Prueba Funcional o de Caja negra: Este tipo de pruebas se centran en el estudio de la especificación del software, análisis de las funciones que debe realizar, de las entradas y de las salidas.

El resultado de las pruebas nos permitirá la depuración de la solución para obtener una versión final y depurada del prototipo. Finalmente, se desarrollará la documentación correspondiente a la Memoria del Proyecto.

2.2.10. Seguridad

Para el apartado de seguridad se aplican los siguientes criterios

- El programa se podrá ejecutar desde cualquier ordenador con sistema operativos como Windows, Linux y Mac.
- Al ser una aplicación de escritorio, los ficheros a convertir y los convertidos, estarán almacenados localmente en el dispositivo que ha ejecutado el programa, por lo que el nivel de privacidad de los datos será establecido por el propio usuario, quien se encargará de proteger el acceso a ficheros o directorios bajo su juicio.
- Incluye una validación de ficheros de entradas de datos, de extensión xml y verificación de permisos de lectura.
- Actualizaciones y parches, es decir, la seguridad de la aplicación también implica mantenerla actualizada con las últimas correcciones de seguridad y parches. Esto implica seguir buenas prácticas de gestión de versiones y poder permitir futuras actualizaciones periódicas para corregir vulnerabilidades conocidas.
- Se obviarán los medios de seguridad contra copias, ya que se trata de software de libre distribución.

Capítulo 3

Objetivos

De acuerdo a la identificación real y técnica del problema, que se ha realizado en el capítulo anterior, a continuación se expondrán todos los objetivos funcionales que se pretenden alcanzar con el desarrollo de este proyecto. El objetivo principal de este proyecto es realizar una aplicación de escritorio que convierta un tipo de cuestionario de un formato XML, el mismo tipo de cuestionario pero en otro formato XML, (MoodleXML a QTI/IMS, ó viceversa) favoreciendo así el uso de cuestionarios en las plataformas que usan estas estructuras. A continuación, vamos a especificar dicho objetivo descomponiéndolo en una serie de subobjetivos que será necesario cumplir para llevar a cabo el proyecto:

- Estudio del estándar MoodleXML y QTI/IMS.
- Estudio de la librería *xml.etree.ElementTree* y *objectify*.
- Estudio de otras herramientas software para el desarrollo del prototipo. Esto incluye para el desarrollo de la interfaz.
- Desarrollar una aplicación sencilla que permita cubrir la necesidad de conversión de cuestionarios en los formatos mencionados.
- La aplicación deberá generar y entender los cuestionarios bien formados de las diferentes estructuras a los que convierta.
- Permitir que el usuario pueda elegir el cuestionario a convertir, cual será el formato final y que el archivo se guarde en el dispositivo.
- Se busca crear una aplicación con una estructura modular que facilite la incorporación de nuevas funcionalidades en el futuro. La aplicación debe ser diseñada de manera escalable, de modo que sea posible añadir módulos adicionales que resulten útiles para el proceso de conversión, y que no hayan sido contemplados en este proyecto.

- Creación de una documentación completa y clara, en la cual se editará el manual de usuario del software desarrollado, así como las memorias y documentación técnica necesaria.

Capítulo 4

Antecedentes

Aunque existen en el mercado multitud de aplicaciones relacionadas con la conversión de formatos, en este apartado, se mencionarán diferentes aplicaciones comerciales relacionadas con el tema a tratar, que han servido de ayuda para concebir una idea más clara del resultado final que se quiere obtener, si nos centramos principalmente en la característica de conversión de los formatos MoodleXML y QTI/IMS, en ambos sentidos. A continuación, se mostrará un breve resumen de su funcionalidad:

4.1. Respondus

Respondus es una compañía que ofrece software y soluciones para la creación, administración y seguridad en exámenes en línea. Es un programa de pago con 30 días gratuitos, pero cuando se probó, solo se obtuvieron resultados insuficientes, ya que convertía un formato QTI a un formato *.doc*, y no especificaba cual era la respuesta correcta. [?]]

4.2. Plugins Moodle

Existen algunos plugins en la comunidad Moodle que realizan la importación de QTI, que desgraciadamente están ahora obsoletos y ya no se mantienen [?]], como son los siguientes:

- *Questionmark QML Importer (Alpha)*: Es una extensión que permite importar cuestionarios y evaluaciones creados en dicho formato, se mantuvo hasta 2018. En nuestras pruebas con varias preguntas de opción múltiple en el estándar QTI 1.2, ninguna de estas preguntas pudo ser importada sin errores con la ayuda de este plugin. Es importante destacar que el término *Alpha* indica que el plugin se encuentra en una fase inicial de desarrollo y es posible que no cuente con todas las características completas o pueda presentar algunos errores.

- *moodle-qformat_imsqti21plugin*: Es una extensión diseñada para el sistema de gestión de aprendizaje Moodle. Este plugin permite la importación y exportación de cuestionarios y evaluaciones en el formato IMS QTI 2.1. Cuando probamos este plugin con las preguntas del estándar QTI 2.1, ni la importación ni la exportación de Moodle XML a QTI 2.1 funcionaron.

4.3. QTIViewer/CCReader

Es una herramienta de software que proporciona una interfaz la cual permite visualizar y leer contenido en formato QTI, lo que facilita la revisión y la interacción con el contenido evaluativo, pero no permite hacer una conversión entre formatos. [?]

4.4. GetMarked.ia

Es una plataforma y servicio en línea que se especializa en la creación, administración y calificación de exámenes y evaluaciones en entornos educativos y empresariales. Utilizando tecnología de inteligencia artificial y análisis de datos, GetMarked.ai automatiza y optimiza el proceso de evaluación, proporcionando una solución eficiente y precisa. Sus características y funcionalidades principales se centran en:

- La creación y diseño de exámenes personalizados.
- La administración de exámenes en línea.
- La corrección y calificación automática de respuestas.
- La generación de informes y análisis detallados.
- La integración con sistemas de gestión del aprendizaje (LMS) y otras herramientas educativas.

GetMarked.ai se enfoca en brindar a educadores, instituciones académicas y organizaciones una forma eficiente de evaluar el aprendizaje, reduciendo la carga administrativa y mejorando la calidad y la eficacia de las evaluaciones. Esta es una permite también la conversión entre formatos, pero es de pago. [?]

4.5. Código abierto

Durante la investigación de herramientas nos encontramos con códigos en el repositorio online GitHub, y otros en páginas web, los cuales sirvieron de guía para realizar nuestro código.

- Generador de preguntas XML de Moodle: Es un módulo para Python. Con este módulo, podemos generar fácilmente un conjunto de preguntas de opción múltiple e importarlas a Moodle. [?]
- text2qti: Convierte archivos de texto sin formato basados en Markdown en cuestionarios en formato QTI (versión 1.2), que pueden ser importados por Canvas y otro software educativo, usando el lenguaje Python. [?]
- moodle2qti [?]
- Procesamiento de datos QTI en Python [?]

Capítulo 5

Restricciones

En este capítulo se expondrán todas las restricciones, o factores limitativos, existentes en el ámbito del diseño que condicionarán el desarrollo de nuestro proyecto. Estos factores limitativos, según su tipo, se pueden clasificar en dos grupos: *Factores Dato* y *Factores Estratégicos*.

5.1. Factores Dato

El problema que se ha planteado apenas presenta barreras que limiten las posibles soluciones, como pudieran ser las relativas al hardware, el software, los plazos, etc. Esto no quiere decir que sean inexistentes:

- Limitaciones en el plazo de entrega. El presente proyecto tiene el plazo máximo de entrega el 10 de Junio de 2023. Para esa fecha, se espera que el proyecto esté totalmente finalizado.
- Limitaciones técnicas. Estas restricciones pueden estar relacionadas con las capacidades técnicas del equipo o las especificaciones, debido a que anteriormente desconocía totalmente el problema presentado.
- Limitaciones económicas. El coste para llevar a cabo el proyecto es mínimo; debido a la utilización de software estándar disponible gratuitamente y hardware propio.

5.2. Factores Estratégicos

Siguiendo los objetivos descritos en el Capítulo 3, a continuación, identificaremos los factores estratégicos que, si bien condicionarán las distintas propuestas alternativas, pueden ser objeto de modificación o elección en uno u otro sentido.

5.2.1. Lenguaje de programación

Se ha optado por usar el lenguaje de programación Python, que a día de hoy, es un lenguaje de programación versátil y potente ya que cuenta con una gran cantidad de bibliotecas y módulos que facilitan la tarea de convertir formatos XML, por ejemplo, se hará uso de las siguientes bibliotecas estándares:

- `xml.etree.ElementTree` ya que proporciona una forma fácil de analizar y manipular documentos XML. [?]
- Biblioteca `lxml` la cual contiene el módulo `objectify`, el cual es altamente eficiente en el procesamiento de XML y XHTML.[?]

Además, Python es multiplataforma, escalable y tiene una comunidad activa [?]. Todo ello irá integrado en una aplicación programada en Python haciendo uso del módulo `tkinter`.

5.2.2. Entorno de desarrollo

El entorno de trabajo será un PC con Ubuntu 20.04.5 LTS, la versión de este sistema operativo cuenta con soporte a largo plazo, lo que la convierte en una opción estable y confiable. [?]

Para creación y modificación del código fuente se ha utilizado la herramienta *Visual Studio Code* para desarrollar la aplicación de escritorio, debido a que ofrece una amplia gama de características, herramientas y extensiones que pueden mejorar la eficiencia y la productividad en el proceso de desarrollo. Además VS Code tiene una excelente integración con Python, lo que proporciona soporte para la depuración de código Python directamente desde el editor, lo que facilita la identificación y solución de problemas en la aplicación de escritorio. [?]

Por último, para la aplicación de escritorio se ha usado *Tk*, una biblioteca de código abierto escrita en C, la cual Python incluye en su librería estándar, en el módulo `tkinter`, que permite interactuar con Tk para desarrollar aplicaciones de escritorio en Python de forma rápida, de modo que cualquier programador con una mínima base de Python puede comenzar rápidamente a crear aplicaciones gráficas profesionales y luego distribuirlas vía herramientas como `cx_Freeze` o `PyInstaller`, que se integran muy bien con Tk. [?]

En su momento, se decidió usar Python debido a que es el mismo lenguaje en el que estábamos programando las funcionalidades del conversor, y facilitaba enlazar la interfaz gráfica.

5.2.3. Desarrollo de la documentación

La documentación será creada utilizando LaTeX a través de la herramienta en línea Overleaf. Inicialmente, se consideró utilizar un editor de texto convencional como Microsoft Word, pero esta opción fue descartada al darnos cuenta de que no se podría alcanzar un nivel de control tan avanzado sobre el texto ni obtener resultados de la misma calidad que con el sistema elegido.

Para crear los diagramas necesarios que respalden el análisis y diseño de la aplicación bajo UML (Lenguaje Unificado de Modelado), utilizaremos la herramienta Draw.io en su versión web. Esta herramienta fue elegida debido a que es gratuita y proporciona la mayoría de los tipos de diagramas necesarios. Con ella, podremos representar gráficamente la estructura y las relaciones de la aplicación de manera efectiva. [1]

Capítulo 6

Recursos

Para llevar a cabo el desarrollo del proyecto se utilizarán una serie de recursos humanos, hardware y software los cuáles se detallarán a continuación

6.1. Humanos

El proyecto será realizado por el alumno de Ingeniería Informática Diego Alejandro Higuera Grisales, dirigido y coordinado por: Cristóbal Romero Morales, profesor perteneciente al departamento de Informática y Análisis Numérico de la Universidad de Córdoba.

Dado que este proyecto es un trabajo de fin de carrera, el estudiante encargado del desarrollo asumirá los roles de analista y programador. El director del proyecto tendrá la responsabilidad de orientar al estudiante durante su desarrollo y revisar periódicamente su trabajo para asegurar que avance de manera adecuada y correcta.

6.2. Hardware

El proyecto será desarrollado en los equipos propiedad del alumno.

- **Portátil:** Intel Core i7 8550U, up to 2000 MHz, 8 GB de RAM, Intel(R) UHD Graphics 620, SSD 256 GB.
- **Impresora:** Hp DeskJet 2720.

6.3. Software

En cuanto a los recursos software que se utilizarán para la realización del proyecto:

- Sistemas operativos usados:

Windows 10.

Linux (Ubuntu 20.04.4 LTS).

- Herramientas de desarrollo de la aplicación: Python 3, API de objectify y API de ElementTree.
- Herramienta para la edición de código: editor de código fuente Visual Studio Code (VS Code).
- Herramientas para el desarrollo de la documentación y diagramas:
 - \LaTeX . Sistema de composición de textos usado para realizar la documentación, bajo la herramienta de edición en línea Overleaf.
 - Draw.io. Aplicación web para la realización de diagramas (UML, ER, etc.).

Capítulo 7

Análisis del sistema

Este apartado tiene por objeto realizar un análisis global del sistema exponiendo todos los requisitos que deberá satisfacer, siendo éstos determinados en base al enunciado del problema, tal y como ha sido especificado en el Capítulo 2 de este documento.

7.1. Descripción funcional

A continuación, se expondrán, mediante un diagrama de casos de uso y su especificación, las funcionalidades que deberá satisfacer nuestro sistema para cumplir con los objetivos fijados.

7.1.1. Diagrama de casos de uso

En la documentación técnica de la aplicación web que se ha tomado como base, la de Miguel Ángel Cid García [2], se realiza una descripción funcional completa del sistema, definiendo una variante con varios niveles en el diagrama de casos de uso, cuyo objetivo es obtener un alto nivel de especificación sólo con los casos de uso.

Por nuestra parte, nos ha parecido más interesante reflejar el comportamiento esperado del sistema en un único diagrama, ciñiéndonos así al estándar UML. Esto permitirá tener una visión global de la aplicación, y por ende, situar más fácilmente en el contexto de esta, tanto nuestro módulo como el resto. En la figura 7.1, podemos ver el resultado:

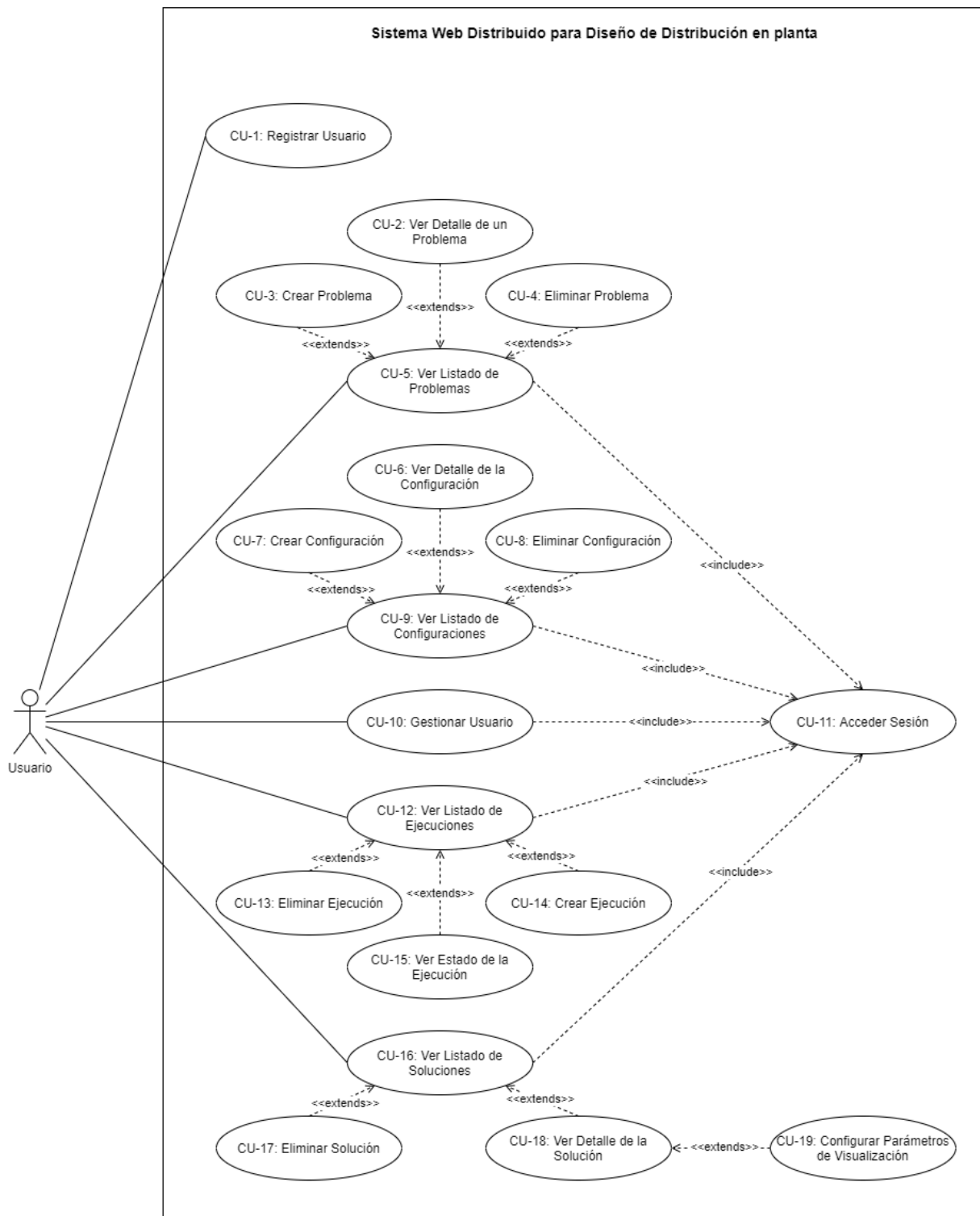


Figura 7.1: Diagrama de casos de uso

En este caso en particular, el análisis se centrará en los casos de uso que intervienen directamente en nuestro sistema, es decir, los que afectan a la visualización de las soluciones y a la modificación dinámica de los parámetros de representación:

- CU-18. Ver Detalle de la Solución.
- CU-19. Configurar Parámetros de Visualización.

7.1.2. Análisis del caso de uso Ver Detalle de la Solución

ID	CU-18
Caso de uso	Ver Detalle de la Solución
Actor	Usuario
Descripción	El usuario visualiza la representación gráfica de la solución, o soluciones, y los porcentajes de cumplimiento de las condiciones establecidas.
Precondición	Estar logado en el sistema.
Escenario de éxito	Flujo principal 1. El usuario pulsa sobre la solución que quiere ver en detalle, en la lista de soluciones generadas. 2. El sistema acepta la petición y devuelve la información de la ejecución terminada, y las posibles soluciones a ese problema, según el algoritmo y la configuración escogida.

Cuadro 7.1: Especificación caso de uso Ver Detalle de la Solución

7.1.3. Análisis del caso de uso Configurar Parámetros de Visualización

ID	CU-19
Caso de uso	Configurar Parámetros de Visualización
Actor	Usuario
Descripción	El usuario puede modificar los distintos parámetros de visualización (flujos, adyacencias, color de los elementos, etc.) y almacenar sus preferencias si lo desea.
Precondición	Estar logado en el sistema.
Escenario de éxito	<p>Flujo principal</p> <ol style="list-style-type: none"> 1. El usuario pulsa sobre el botón de configuración. 2. El sistema muestra los parámetros configurables. 3. El usuario realiza las modificaciones que considere y acepta los cambios. 4. El sistema modifica la representación con los cambios aplicados. <p>Flujo alternativo <i>Almacenar Configuración</i></p> <ol style="list-style-type: none"> 3. El usuario realiza las modificaciones que considere y almacena los datos de la configuración resultante para poder recuperarla cuando desee. 4. El sistema almacena la configuración. <p>Flujo alternativo <i>Recuperar Configuración</i></p> <ol style="list-style-type: none"> 3. El usuario solicita recuperar su configuración de parámetros de visualización almacenada. 4. Si existe, el sistema carga la configuración.

Cuadro 7.2: Especificación caso de uso Configurar Parámetros de Visualización

7.2. Descripción de la información

Los datos manejados en el sistema se dividen en dos grandes bloques; los parámetros del problema y las soluciones generadas.

7.2.1. Parámetros de un problema

Se trata de los datos que utiliza la aplicación, y sobre los que se aplica un algoritmo para obtener las posibles soluciones. Los ficheros de parámetros de un problema presentan la siguiente estructura:

- Descripción del problema.
- Información adicional.
- Listado de plantas. Lista ordenada de los pisos que forman parte del problema con información de la forma de cada uno. Para cada piso se indica:
 - Número de planta.
 - Polígono de la planta.
 - Exterior de la planta. Indica mediante una cadena el tipo de exterior que linda con la planta en sus distintas aristas. Este campo es opcional.
- Listado de instalaciones. Datos relevantes sobre las instalaciones de cada planta:
 - Nombre.
 - Tipo de instalación.
 - Área mínima.
 - Ancho mínimo.
 - Ar_min, Ar_opt_start, Ar_opt_end y Ar_max. Datos definitorios de la relación de aspecto.
- Listado de tablas. En este apartado se muestran todas las relaciones que existen entre las instalaciones de cada planta. Se podrán definir distintos tipos de tablas (en forma de matriz) para representar preferencias relativas a los flujos o las adyacencias.

```
(
  'descripcion_del_problema',
  'informacion_adicional',

  (#Lista de plantas
    (0, [(0, 0), (18.5, 0), (18.5, 9), (0, 9)], []),
  ),

  (#Lista de instalaciones
    ('A-Stables', 'region', 0.27, 0, 1, 1, 4, 4),
    ('B-Slaughter', 'region', 0.18, 0, 1, 1, 4, 4),
    ('C-Entrails', 'region', 0.27, 0, 1, 1, 4, 4),
    ('D-Leacher', 'region', 0.18, 0, 1, 1, 4, 4),
    ('E-Aeration chamber', 'region', 0.18, 0, 1, 1, 4, 4),
    ('F-Refrigeration chamber', 'region', 0.18, 0, 1, 1, 4, 4),
    ('G-Entrails chamber', 'region', 0.09, 0, 1, 1, 4, 4),
    ('H-Boiler room', 'region', 0.09, 0, 1, 1, 4, 4),
    ('I-Compressor room', 'region', 0.09, 0, 1, 1, 4, 4),
    ('J-Shiping', 'region', 0.24, 0, 1, 1, 4, 4),
    ('K-Offices', 'region', 0.60, 0, 1, 1, 4, 4),
    ('L-Product shiping', 'region', 0.42, 0, 1, 1, 4, 4),
  ),

  {#Lista de tablas
    'Flow' : [
      [0, 100, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
      [0, 0, 15, 10, 60, 0, 0, 0, 0, 0, 0, 0],
      [0, 0, 0, 0, 0, 0, 15, 0, 0, 0, 0, 0],
      [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 10],
      [0, 0, 0, 0, 0, 60, 0, 0, 0, 0, 0, 0],
      [0, 0, 0, 0, 0, 0, 0, 0, 0, 60, 0, 0],
      [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 15],
      [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
      [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
      [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
      [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
    ],

    'Adjacency' : [
      [0, 1, 0, 0, -1, -1, 0, 0, -1, 0, 0, 0],
      [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
      [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
      [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 10],
      [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
      [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
      [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
      [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
      [-1, 0, -1, -1, 0, 0, -1, 0, -1, 0, 0, -1],
      [-1, 0, -1, 1, 0, 0, -1, 0, -1, 0, 0, -1],
      [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
    ],
  },
)
```

Figura 7.2: Ejemplo del fichero de parámetros

7.2.2. Soluciones de un problema

Por otro lado, tenemos los ficheros con la soluciones que se hayan generado a petición del usuario. Su estructura es:

- Nombre del problema.
- Parámetros del lanzamiento.
- Cabecera de evaluación. Descripción de qué representa cada uno de los valores devueltos como evaluación. Es necesario y no debe repetirse en cada solución, ya que es lo mismo para todas las soluciones. No obstante, sería necesario revisar cada algoritmo para saber que significan.
- Soluciones generadas. Lista ordenada de las soluciones que contiene la siguiente información:
 - Generación. Número de generación de la solución (si es imposible precisar mostrará el valor -1).
 - Evaluación. Valores de aptitud otorgados por el algoritmo.
 - Genotipo. Tipo de individuo generado tras aplicar el algoritmo. Se representa como una cadena de caracteres y puede tener diferentes formatos en función del algoritmo que haya sido aplicado.
 - Plantas.
 - Instalaciones. Lista ordenada con la información de las instalaciones:
 - Nombre.
 - Nivel. Planta en la que se encuentra.
 - Categoría. Hace referencia al tipo de instalación (pasillo, habitación, ascensor, etc).
 - Polígono. Coordenadas de la instalación para su representación gráfica.
- Fecha.
- Servidor.
- Procesador.
- Tiempo.


```
(
  'nombre_del_problema',|
  'parametros_de_lanzamiento',

  ['EvaluacionUsuario', 'Flujo', 'Distancia', 'Adyacencia', 'Penalizacion'],

  [
    (
      # Numero de generación de la solución
      -1,
      # Valores de amplitud otorgados a la solución
      [1, 2, 3, 4, 5],
      # Genotipo
      'genotipo',
      # Planta o posibles formas si no se definió
      None,
      # Lista de instalaciones
      [ # (nombre, nivel_de_la_planta, tipo_de_instalacion, coordeandas)
        ('A-Stables', 0, None, ((7, 9), (15, 9), (15, 0), (7, 0))),
        ('B-Slaughter', 0, None, ((15, 6), (18, 6), (18, 0), (15, 0))),
        ('C-Entrails', 0, None, ((0, 8), (2, 8), (2, 1.5), (0, 1.5))),
        ('D-Leacher', 0, None, ((2, 5), (4, 5), (4, 2), (2, 2))),
        ('E-Aeration chamber', 0, None, ((4, 9), (7, 9), (7, 6), (4, 6))),
        ('F-Refrigeration chamber', 0, None, ((4, 6), (7, 6), (7, 1.5), (4, 1.5))),
        ('G-Entrails chamber', 0, None, ((0, 1.5), (2, 1.5), (2, 0), (0, 0))),
        ('H-Boiler room', 0, None, ((0, 9), (2, 9), (2, 8), (0, 8))),
        ('I-Compressor room', 0, None, ((4, 1.5), (7, 1.5), (7, 0), (4, 0))),
        ('J-Shiping', 0, None, ((15, 9), (18, 9), (18, 6), (15, 6))),
        ('K-Offices', 0, None, ((2, 9), (4, 9), (4, 5), (2, 5))),
        ('L-Product shiping', 0, None, ((2, 2), (4, 2), (4, 0), (2, 0))),
      ]
    ),
  ],

  'fecha_y_hora',
  'nombre_del_servidor',
  'procesador_de_la_CPU',
  'tiempo_de_ejecucion',
)
```

Figura 7.3: Ejemplo del fichero de soluciones

7.3. Descripción de la interfaz

A continuación, se explican los aspectos de las soluciones que se han de representar y las diferentes configuraciones para cada uno de ellos.

7.3.1. Representación gráfica de una distribución en planta

Forma en la que se distribuyen las distintas instalaciones de una planta, atendiendo al resultado de aplicar un algoritmo.

Tanto las plantas como las instalaciones quedan definidas por una lista de puntos expresados como pares de coordenadas (x,y) .

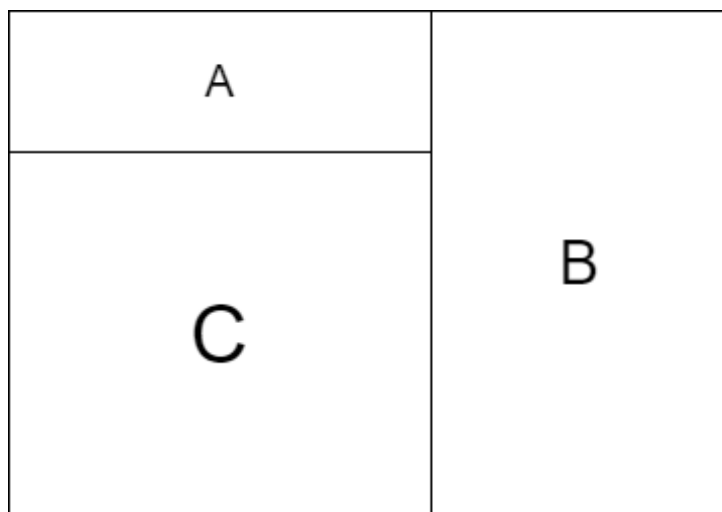


Figura 7.4: Distribución en planta

Además de mostrar la distribución de las distintas instalaciones, como veremos en los siguientes apartados, la intención es que el visualizador refleje todos los aspectos relevantes (flujo de materiales, distancias y adyacencias entre instalaciones, espacios vacíos, etc.) de forma clara y diferenciada.

7.3.2. Representación gráfica del flujo de materiales

Cantidad de materiales que circulan entre las instalaciones de una misma planta. Vendrán definidos a través de una matriz, que relaciona todas las instalaciones, en la definición del problema. El flujo de materiales se podrá representar atendiendo a distintos parámetros:

- Tipo de representación.
 - Distancia Euclídea. Distancia que hay entre dos puntos unidos por una línea recta, la cual se deduce a partir del teorema de Pitágoras. [3]

- Distancia de Manhattan. Suma de las diferencias absolutas de las coordenadas de ambos puntos. [4]

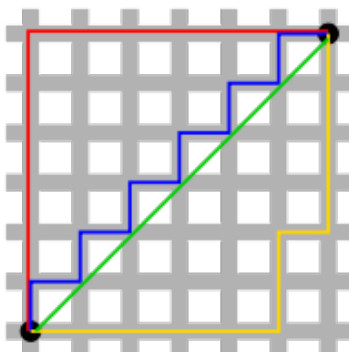


Figura 7.5: Comparación distancias Euclídea y Manhattan

Las líneas roja, azul y amarilla representan la distancia de Manhattan y tienen la misma longitud. Por su parte, la línea verde representa la distancia Euclídea.

- Modo de representación.
 - Color de las líneas. Las líneas tendrán un determinado color en función del grado del flujo.
 - Grosor de las líneas. El grosor de las líneas variará en función de la dimensión del flujo.

A continuación, las distintas combinaciones de representación:

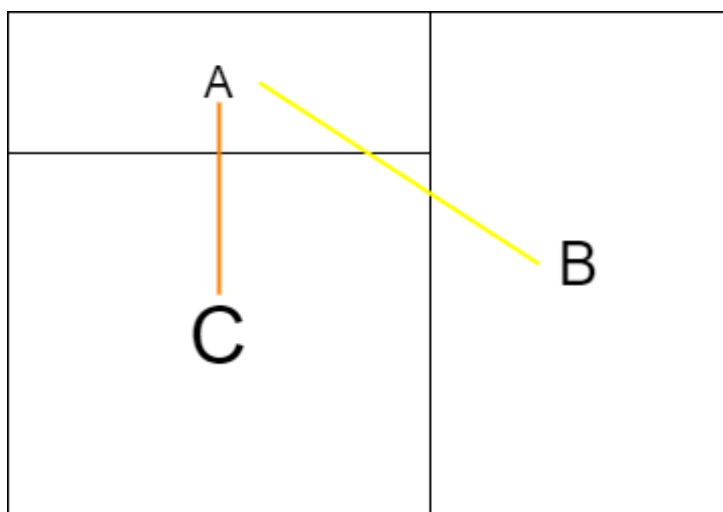


Figura 7.6: Distancia Euclídea por color

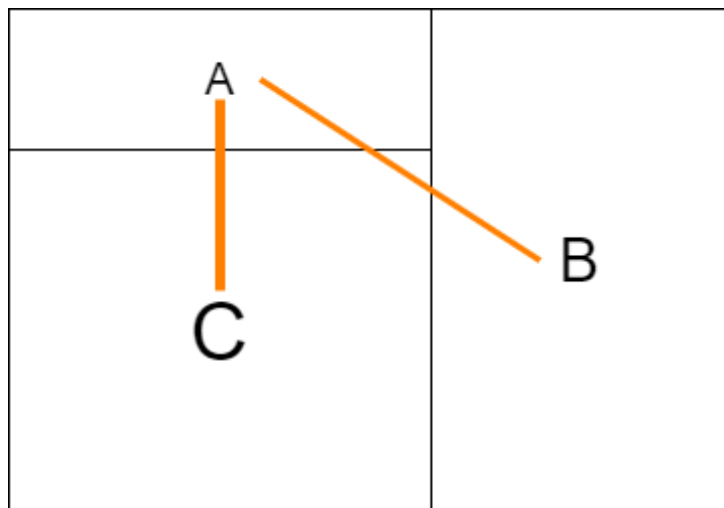


Figura 7.7: Distancia Euclídea por grosor

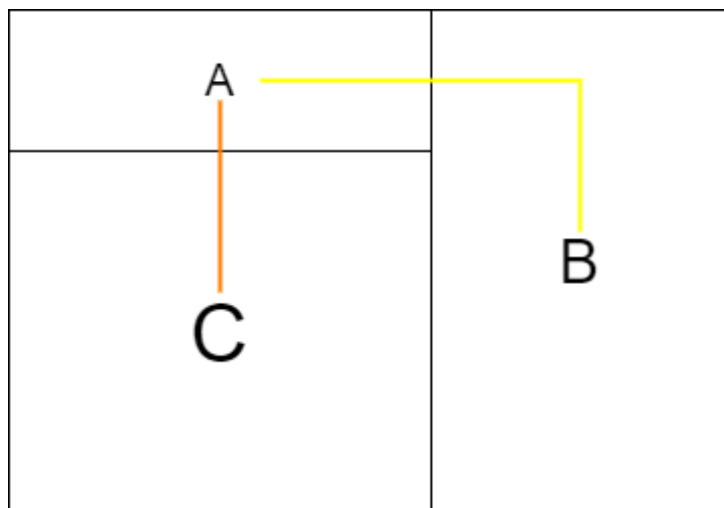


Figura 7.8: Distancia Manhattan por color

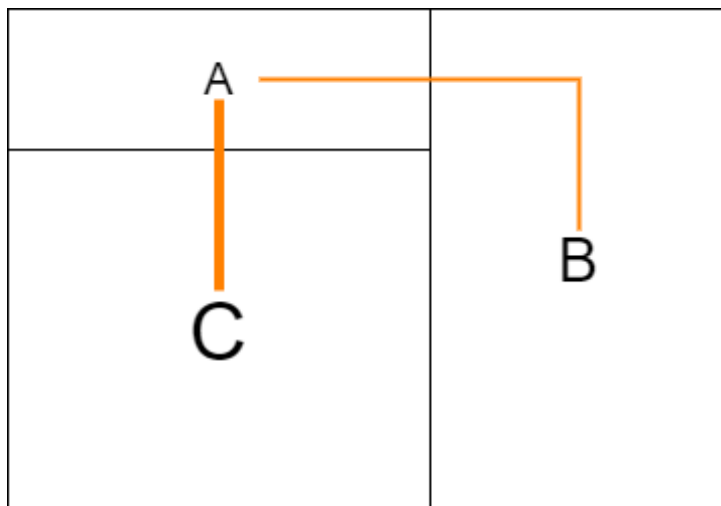


Figura 7.9: Distancia Manhattan por grosor

7.3.3. Representación gráfica de la adyacencia

Reflejará si la solución cumple, o no, con las preferencias de adyacencias que hayan sido establecidas entre las instalaciones.

Del mismo modo que en el apartado anterior, habrá una matriz específica en la definición del problema en el que vendrán definidas, y los parámetros de representación serán modificables:

- Tipo de representación.
 - Línea de unión. Línea recta que unirá los centros de las instalaciones.
 - Resaltado del lado en común. En este caso, si existe una restricción de adyacencia que afecta a dos instalaciones que en la solución no aparecen juntas, no se representará, puesto que al no tener ninguna arista o parte de ella común es imposible realizarlo de esta manera.
- Modo de representación.
 - Color de las líneas. Las líneas tendrán un determinado color en función del cumplimiento de adyacencia.
 - Grosor de las líneas. El grosor de las líneas variará en función del cumplimiento de adyacencia.

A continuación, las distintas combinaciones de representación:

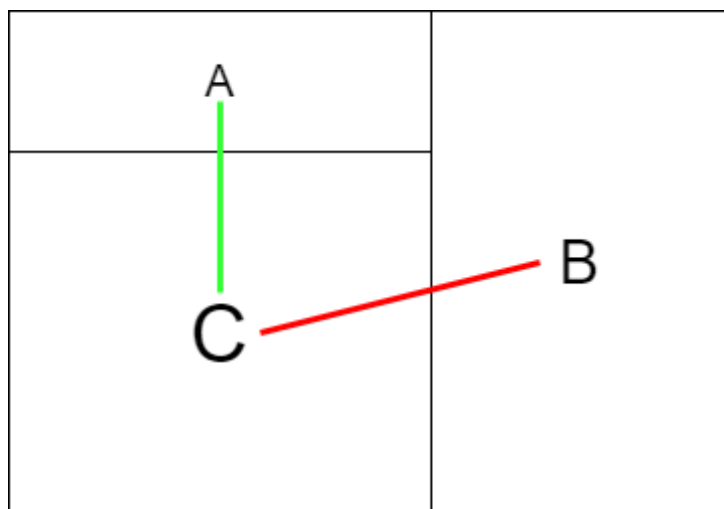


Figura 7.10: Línea de unión por color

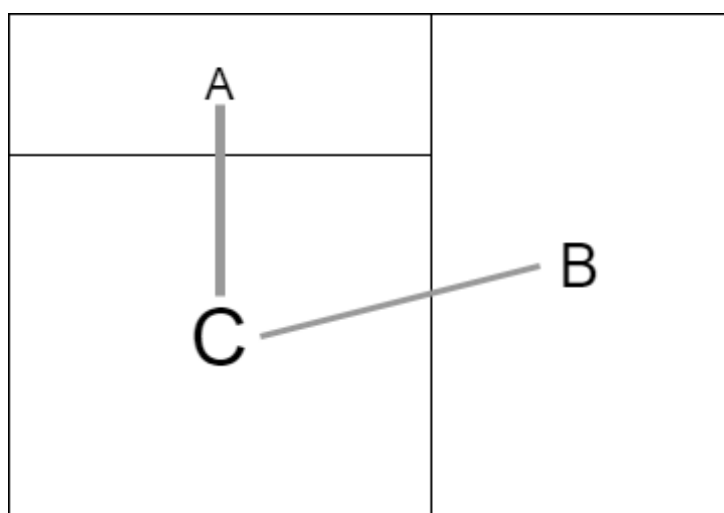


Figura 7.11: Línea de unión por grosor

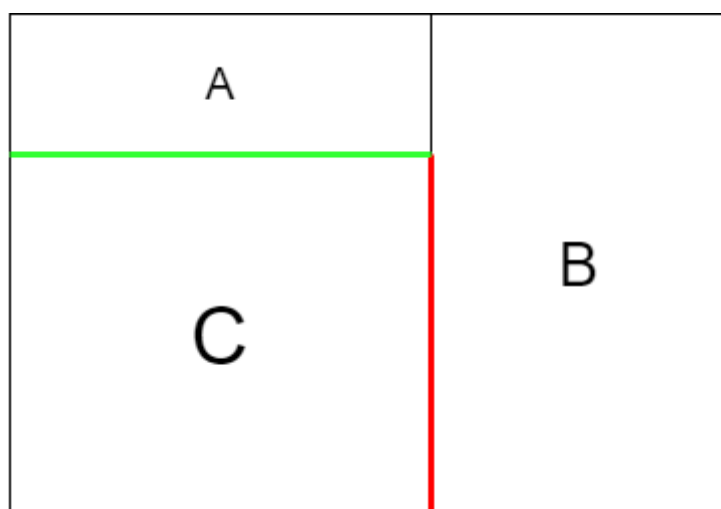


Figura 7.12: Resaltado del lado en común por color

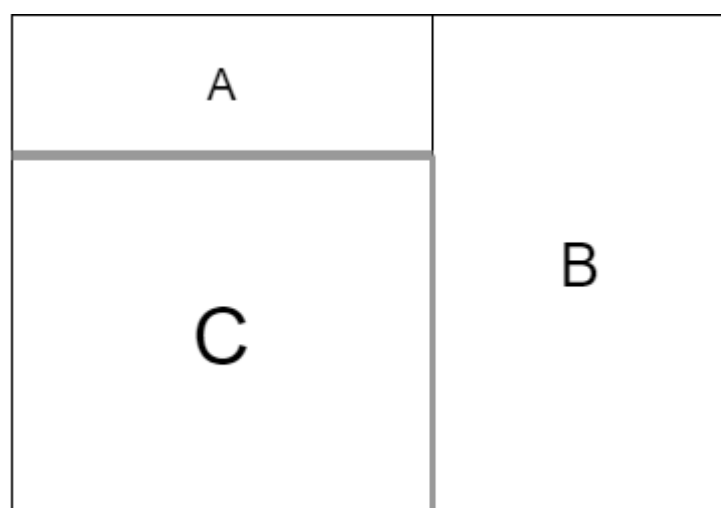


Figura 7.13: Resaltado del lado en común por grosor

7.3.4. Representación gráfica de la relación de aspecto

Indica en que grado cumple una instalación con la relación de aspecto esperada. Al contrario que en el caso de los flujos o las adyacencias, la información acerca de la relación de aspecto o *aspect ratio* no viene definida en una matriz, sino que se evaluará atendiendo a unos valores mínimos, óptimos y máximos definidos independientemente para cada instalación.

Francisco Raso Lucena nos explica detalladamente el criterio seguido para la evaluación de este aspecto a través de una gráfica en la que se puede distinguir un polígono con forma de trapecio. [5]

El eje X representa la relación de aspecto obtenida y tendrá valores mayores o iguales a uno, ya que resulta de dividir el lado mayor entre el lado menor (teniendo que calcular previamente el rectángulo circunscrito en caso de que no se trate de un polígono rectangular). Por su parte, el eje Y representa el grado de deseo, que oscilará entre cero (mínimo) y uno (máximo).

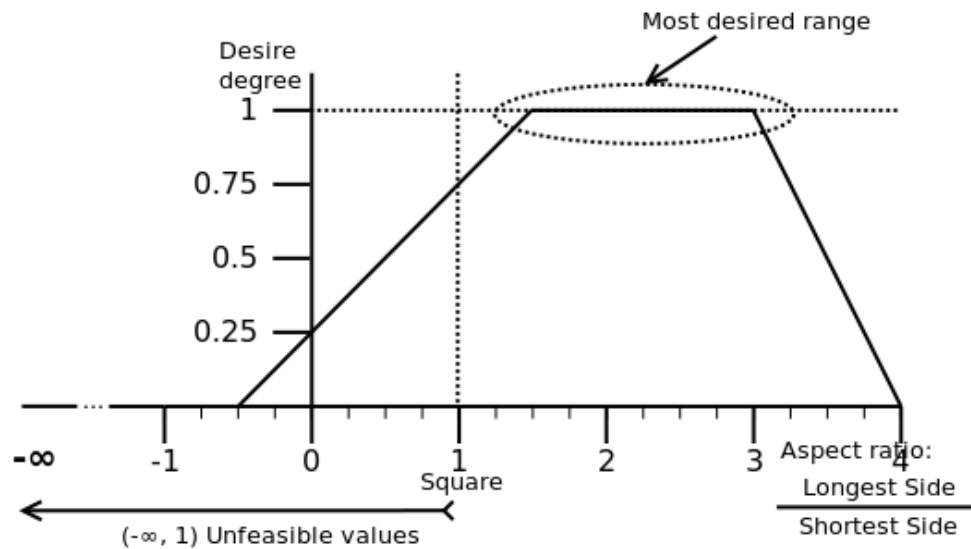


Figura 7.14: Trapecio para la relación de aspecto deseable

Los puntos que forman el trapecio se corresponden con los indicados en la definición del problema para la instalación: valor mínimo $(-0,5)$, valores óptimos o deseados $(1,5)$ y (3) , y valor máximo (4) . Estos valores indican la posición en el eje X, teniendo el máximo y mínimo un grado de deseo igual a cero, mientras que los valores óptimos tendrían un grado de deseo igual a 1.

En cuanto a la representación gráfica, el usuario podrá elegir entre las dos opciones disponibles:

- Con un color. Se escogerá el color deseado y se mostrará en tonos más claros si existe un buen grado de cumplimiento, mientras que se aplicarán tonos más oscuros cuando el grado de cumplimiento sea menor.
- Con dos colores. Las instalaciones tendrán un determinado color en función del cumplimiento de la relación de aspecto.

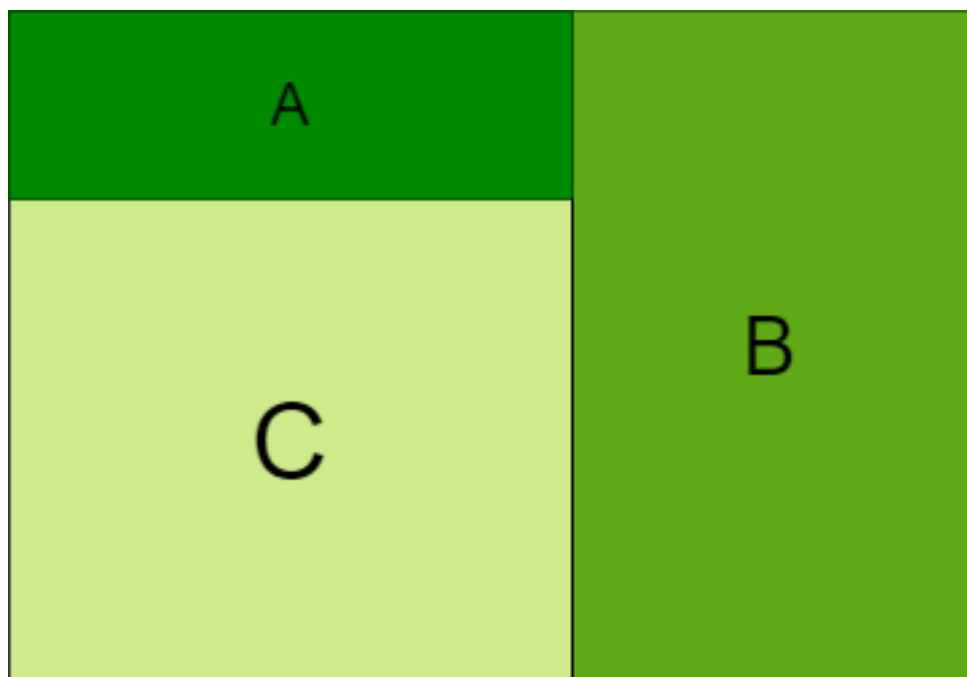


Figura 7.15: Grado de cumplimiento del aspect ratio deseable con un color

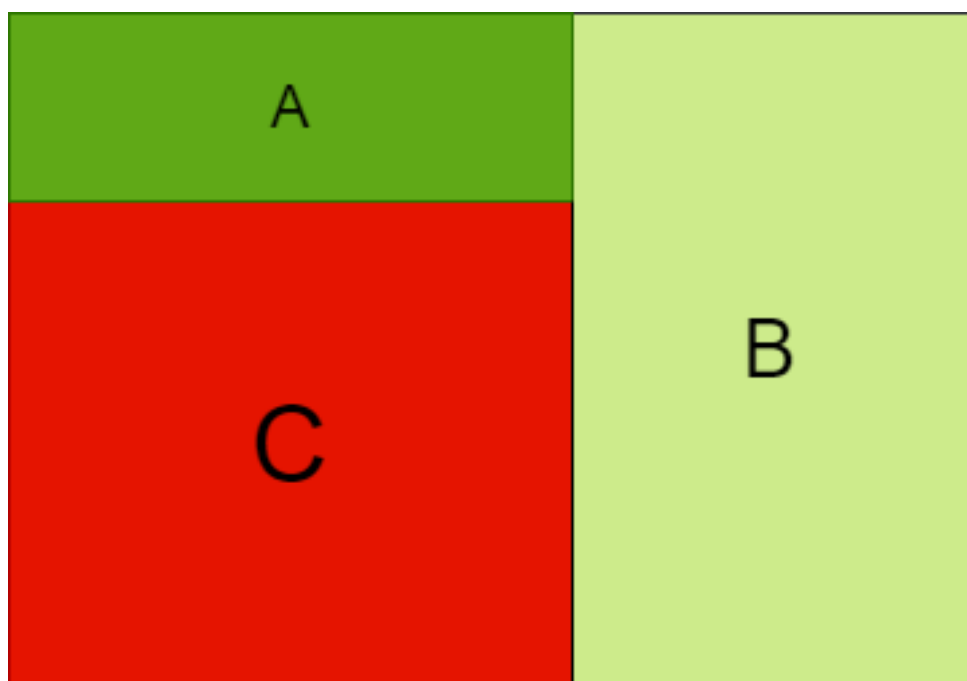


Figura 7.16: Grado de cumplimiento del aspect ratio deseable con dos colores

7.3.5. Representación de textos

Para la representación de los textos correspondientes a los nombres de las instalaciones, se calculará en cada caso la relación de aspecto en función de los resultados obtenidos en la solución, y se ajustará el tamaño y posición del mismo.

7.3.6. Representación gráfica del espacio vacío

No es algo que haya que calcular o venga definido. Se trata del espacio sobrante que puede quedar una vez se organicen todas las instalaciones de una planta. No obstante, se podrá resaltar mediante dos formas distintas:

- Color. Se escoge un color distinto para distinguir el espacio sin utilizar.
- Puntos. Fondo relleno de puntos para diferenciarlo del resto de elementos.

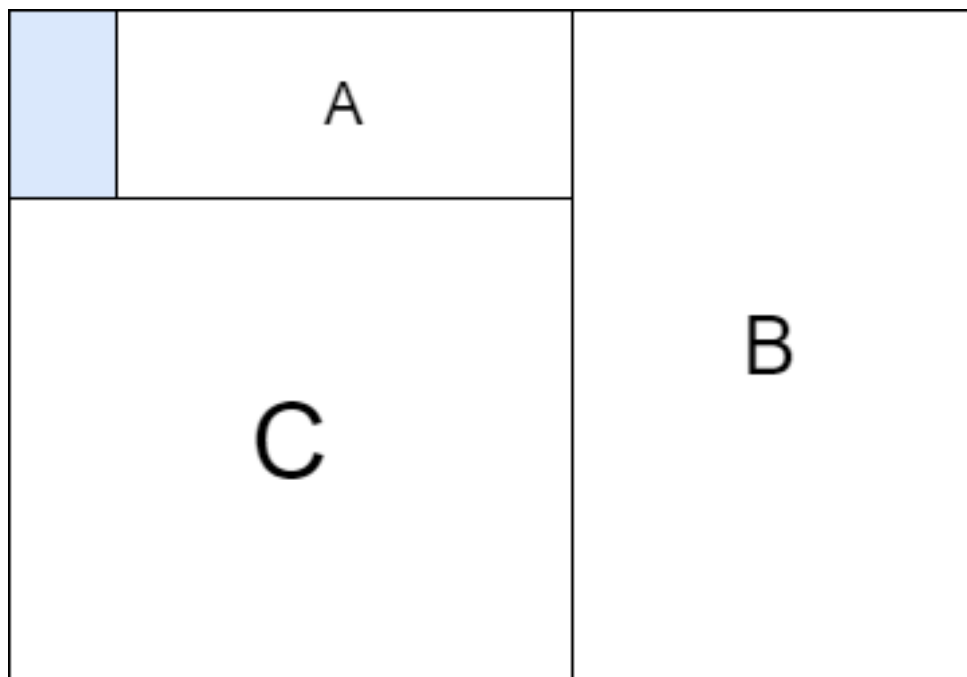


Figura 7.17: Resaltado del espacio vacío con color

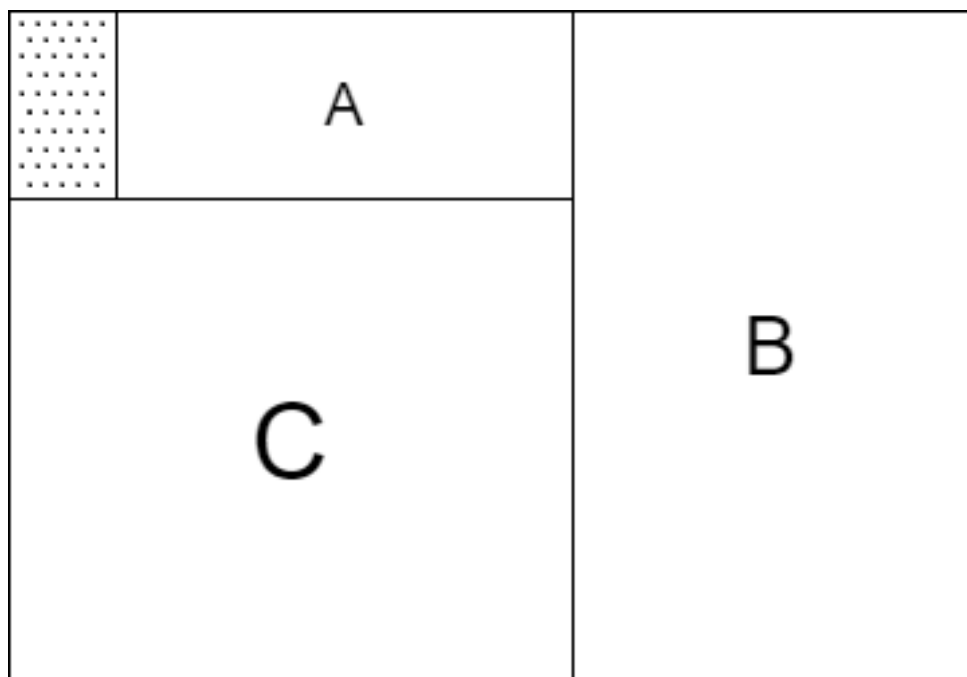


Figura 7.18: Resaltado del espacio vacío con puntos

7.4. Especificación de Requisitos

En esta sección, se especifican de forma técnica los requisitos necesarios para completar una descripción detallada de los servicios del sistema. Para facilitar su trazabilidad, estos requisitos serán expuestos en función de los casos de uso y descripciones de funcionalidad, información e interfaz.

7.4.1. Requisitos Funcionales

- RF-1. El sistema debe mostrar un listado de las soluciones generadas para la ejecución seleccionada.
- RF-2. El sistema debe mostrar información relativa a la ejecución que ha generado las soluciones.
- RF-3. El sistema debe permitir modificar los colores de visualización de cada aspecto por separado: instalaciones, textos, líneas delimitantes y flujos.
- RF-4. El sistema debe permitir escoger el modo de representación de los flujos: mediante el grosor de las líneas o mediante el color de las líneas.
- RF-5. El sistema debe permitir escoger el tipo de representación de los flujos: mediante Distancia de Manhattan o mediante Distancia Euclicídea.

- RF-6. El sistema debe permitir escoger el modo de representación de las adyacencias: mediante el grosor de las líneas o mediante el color de las líneas.
- RF-7. El sistema debe permitir escoger el tipo de representación de las adyacencias: mediante líneas de unión o mediante resaltado de los lados en común.
- RF-8. El sistema debe permitir configurar si quiere que se muestren, o no, los flujos y adyacencias de cada instalación por separado.
- RF-9. El sistema debe permitir almacenar una configuración personalizada de los parámetros, para poder recuperarla cuando le sea necesario.
- RF-10. El sistema debe ofrecer distintas opciones para representar los espacios vacíos.
- RF-11. El sistema debe ofrecer información acerca del grado de cumplimiento de la relación de aspecto.

7.4.2. Requisitos de Información

- RINF-1. Información sobre los problemas. El sistema necesita la información relativa al enunciado del problema cuya solución se va a representar.
- RINF-2. Información sobre las soluciones. El sistema necesita la información relativa a las soluciones generadas para su correcta representación.

7.4.3. Requisitos de Interfaz

- RI-1. El sistema deberá ofrecer una interfaz usable e intuitiva, la cual no requiera una gran curva de aprendizaje ni grandes conocimientos.

7.4.4. Requisitos No Funcionales

- RNF-1. El sistema deberá realizar de manera instantánea, tanto la representación de las soluciones generadas, como los cambios aplicados sobre las mismas.
- RNF-2. Las representaciones gráficas deben ser compatibles con los principales navegadores web.

Capítulo 8

Diseño del sistema

Una vez el problema ha sido especificado y la solución analizada, se procede al diseño del sistema comenzando por el modelo de los datos que se van a manejar.

8.1. Modelo de datos

Los datos serán extraídos de dos ficheros: el de parámetros de un problema y el de soluciones de un problema, ambos analizados en el punto 7.2 de este documento. No obstante, no toda la información contenida en ellos nos resulta relevante; en la tabla 8.1, se mostrarán tanto los datos de los que sí se hará uso como su procedencia.

Por otra parte, con el objetivo de dotar al sistema final con la mejor calidad, se van a aplicar algunas buenas praxis de la programación orientada a objetos. [6]

Concretamente, los datos que maneja el visualizador se gestionarán a través de un módulo de encapsulamiento, que hará las veces de intermediario entre los datos y el sistema. Esto facilitará su mantenimiento a lo largo del tiempo, ya que en caso de que se efectue cualquier modificación en el formato de los datos, solo será necesario realizar una adaptación de este módulo para que el sistema siga funcionando correctamente.

Procedencia	Nombre	Tipo	Descripción
Problema	Lista_Plantas	Lista	Listado de plantas, con información de su forma y dimensiones.
Problema	Lista_Instalaciones	Lista	Listado de las instalaciones, con información de su nombre, tipo, área y aspect ratio.
Problema	Matriz_Flujos	Matriz	Especificación del flujo entre instalaciones.
Problema	Matriz_Adyacencias	Matriz	Especificación de las preferencias de adyacencias entre instalaciones.
Soluciones	Cabecera_Evaluacion	Lista	Aspectos evaluados de la solución que varían según el algoritmo.
Soluciones	Lista_Soluciones	Lista	Listado de soluciones, con información de su evaluación y posición de cada una de las instalaciones.

Cuadro 8.1: Sumario de datos

8.2. Modelo arquitectónico

El principal objetivo del diseño arquitectónico es desarrollar una estructura de programa modular y representar las relaciones de control entre los módulos.

8.2.1. Clases

La clase Parámetros se utiliza para almacenar/recuperar preferencias del usuario sobre la representación de las soluciones. La clase se define de la siguiente forma:



Figura 8.1: Resaltado del espacio vacío con puntos

Dicha clase consta de los siguientes atributos:

- **UsuarioID.** Identificador del usuario que está logado en el sistema.
- **FlujoRepresentacion.** Forma en la que se va a representar el tipo de flujo escogido.
- **FlujoTipo.** Tipo de representación escogida para los flujos.
- **AdyRepresentacion.** Forma en la que se va a representar el tipo de adyacencia escogido.
- **AdyTipo.** Tipo de representación escogida para las adyacencias.
- **InstalacionesColor.** Color de las intalaciones.

- **TextosColor.** Color de los textos.
- **BordesColor.** Color de los bordes que delimitan las instalaciones.

Dicha clase consta de los siguientes métodos:

- **AlmacenarConfiguracion().** Función que almacena los parámetros indicados por el usuario para la representación de soluciones.
- **RecuperarConfiguracion().** Función que carga los parámetros de representación de soluciones previamente almacenados por el usuario.

En la siguiente imagen, se muestra como quedará el diagrama de clases completo una vez se incluya la clase Parametros.

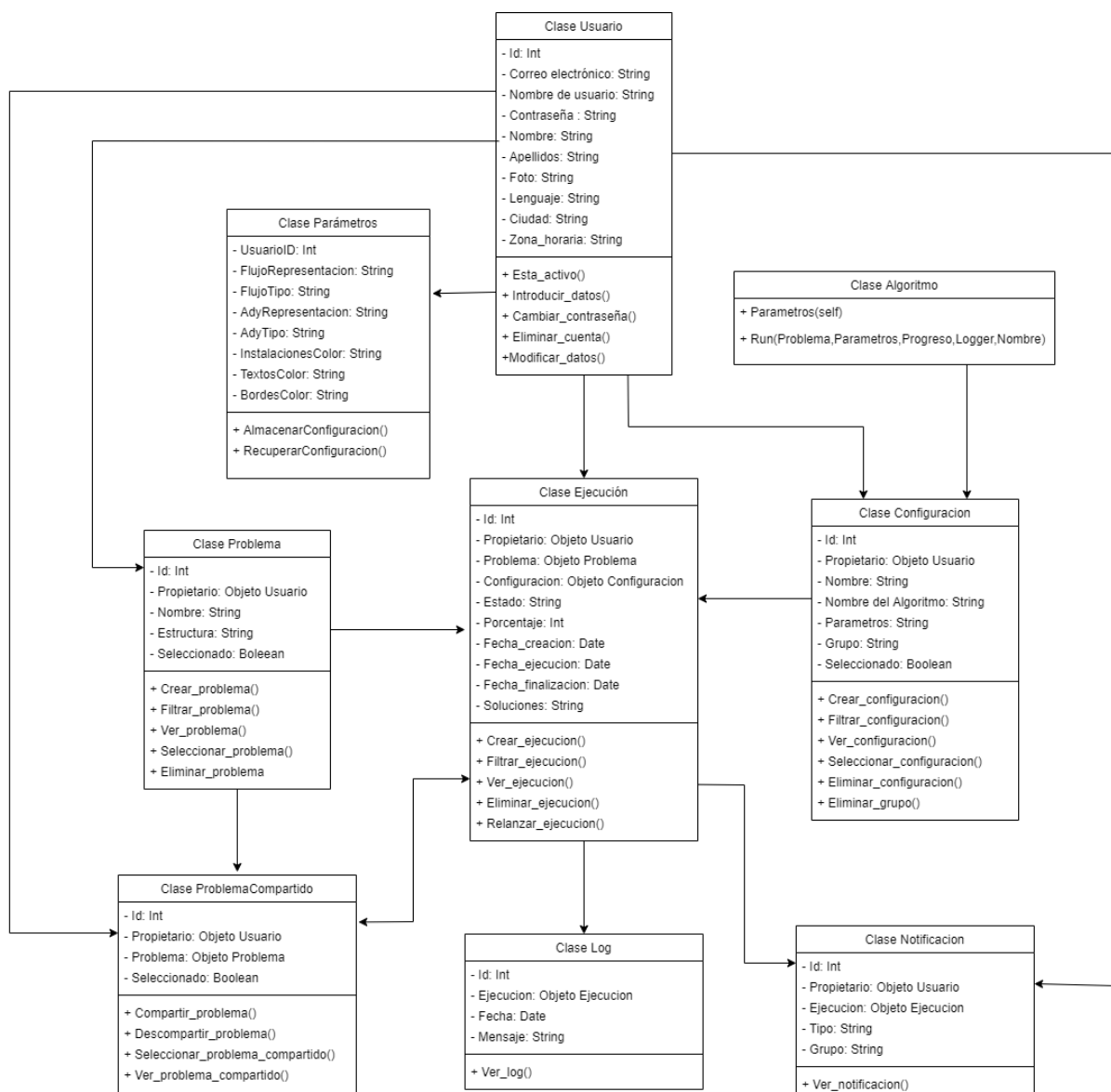


Figura 8.2: Diagrama de clases

8.2.2. Estructuración del código

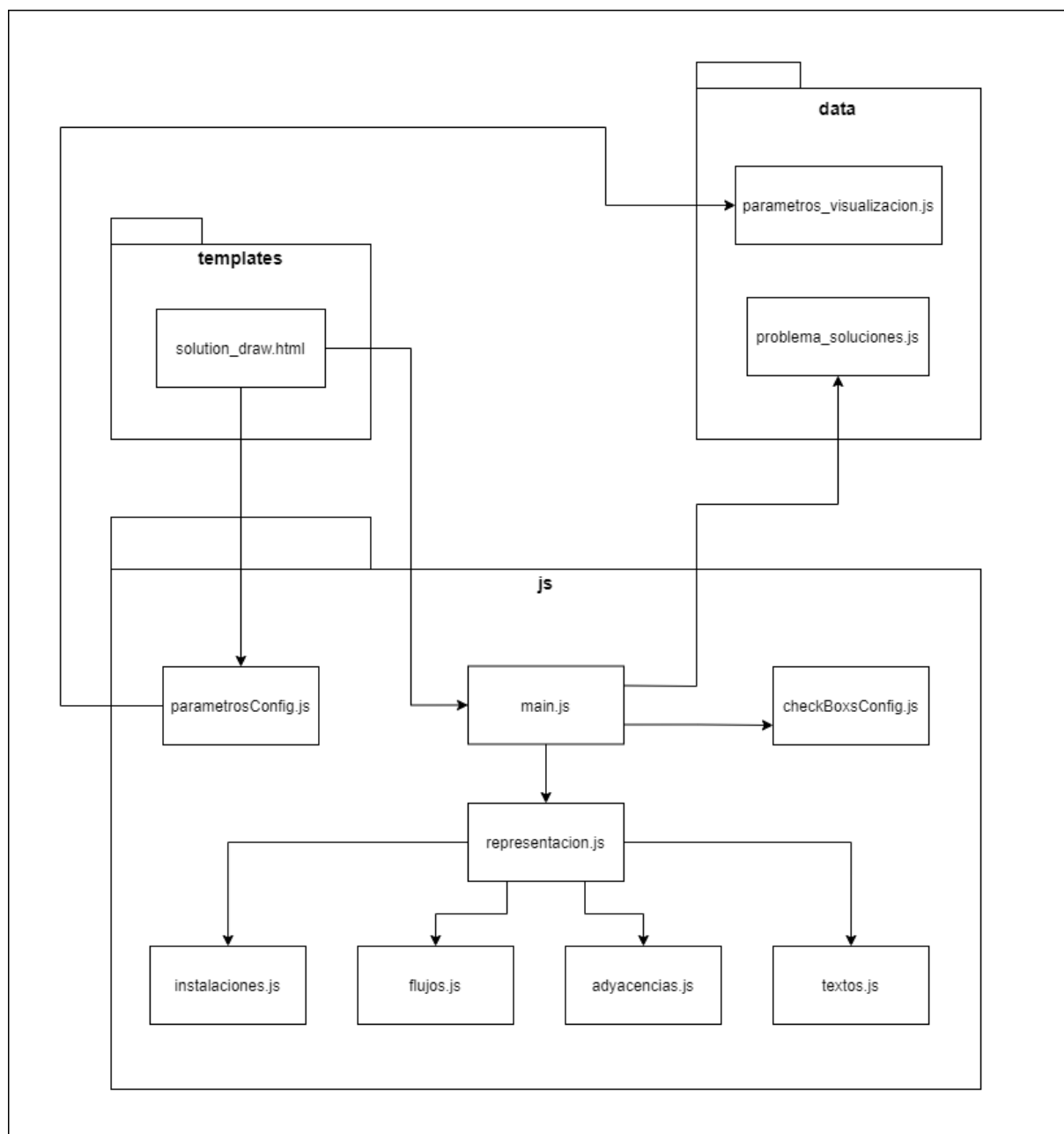


Figura 8.3: Esquema de estructuración del código

Como se puede observar en la figura 8.2, los archivos que componen nuestro módulo se encuentran repartidos en tres carpetas distintas:

Carpeta templates

Contiene todas las plantillas de la interfaz de la aplicación.

- **solution_draw.html.** Define los elementos que se muestran en la pantalla de visualización del detalle de la solución, se declaran las variables globales con las que se trabajará y se llama al archivo principal *main.js*.

Carpeta data

Contiene información relevante para el funcionamiento de la aplicación.

- **problema_soluciones.js.** Archivo que se recarga con la información del problema y soluciones generadas para cada caso en particular. Es llamado desde *main.js*.
- **parametros_visualizacion.js.** Archivo que se recarga con los parámetros de visualización almacenados por el usuario en cada momento.

Carpeta js

Contiene todos los archivos de JavaScript usados, tanto para elementos de diseño como para interacciones con el servidor.

- **main.js.** Contiene las configuraciones principales, como la carga de parámetros de visualización por defecto, inicialización de variables o creación de *CheckBoxes* en función del número y nombre de instalaciones para el problema en cuestión. Finaliza llamando al archivo de renderización de la planta *representacion.js*.
- **parametrosConfig.js.** Alberga las funciones que afectan a los parámetros de visualización, ya sea para aplicar cambios, restaurarlos por defecto, almacenar las preferencias del usuario o cargar unas preferencias previamente almacenadas. Interactúa directamente con el usuario desde *solution_draw.html*.
- **checkBoxsConfig.js.** Monta dinámicamente los *CheckBoxes* de selección de instalaciones y se definen las funciones para seleccionarlos y deseleccionarlos.
- **representacion.js.** Este archivo auna todas las operaciones o llamadas relacionadas con la creación de la imagen resultante. Se encuentra modulado para separar las operaciones que afectan directamente a las propias instalaciones, a los flujos, a las adyacencias y a los textos de la imagen.

- **instalaciones.js.** Contiene las funciones encargadas de dibujar las instalaciones en su posición adecuada en el espacio.
- **flujos.js.** Contiene todas las funciones encargadas de representar los flujos en sus distintas formas.
- **adyacencias.js.** Contiene todas las funciones encargadas de representar las adyacencias en sus distintas formas.
- **textos.js.** Contiene las funciones necesarias para dar formato, tamaño y color a los textos.

8.2.3. Flujos de datos

En base a los casos de uso analizados, se definirá el flujo de datos que deberá seguir el sistema, más tarde, en la implementación.

Diagrama de secuencia Visualizar solución

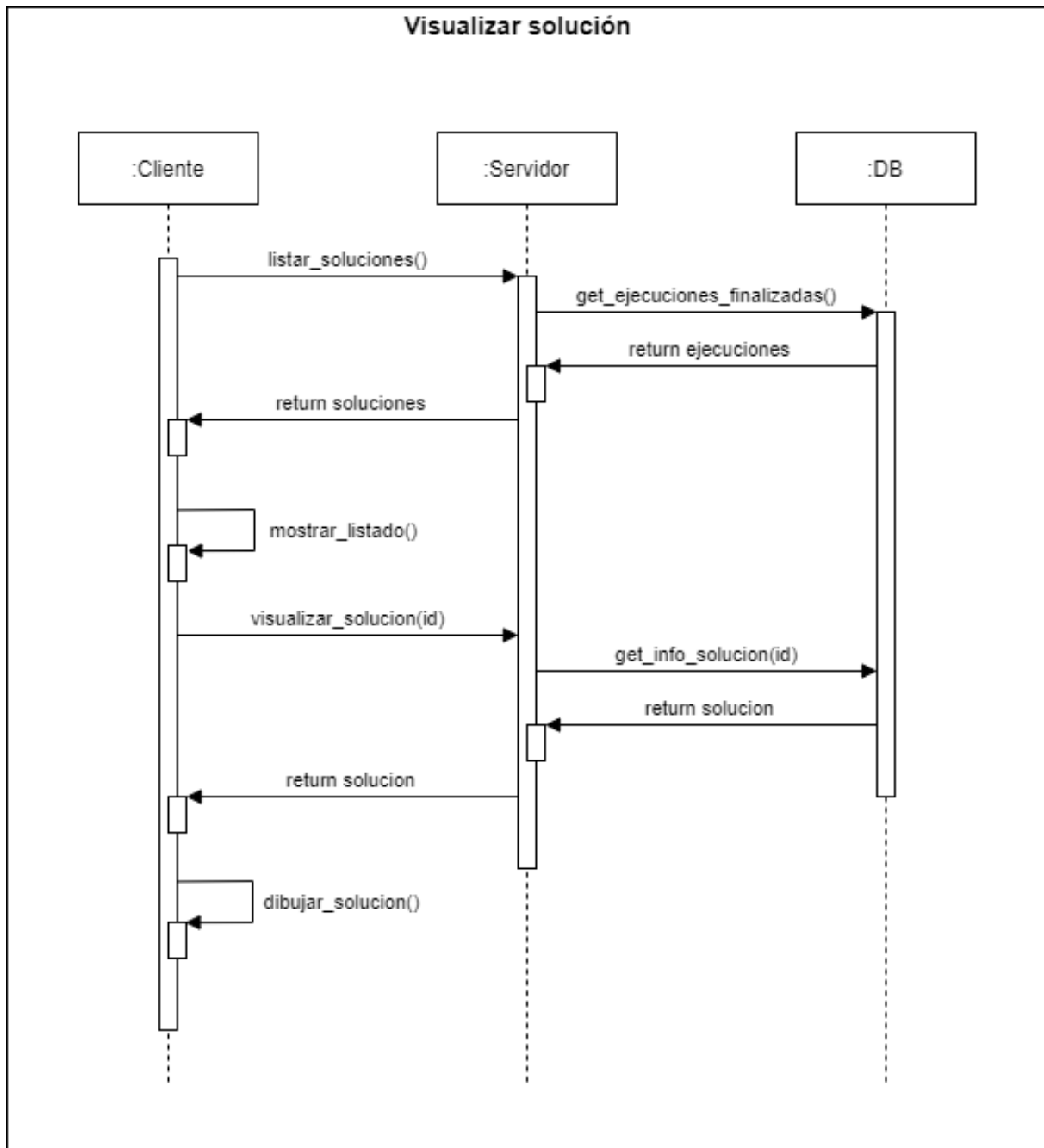


Figura 8.4: Diagrama de secuencia Visualizar solución

Diagrama de secuencia Cargar parámetros

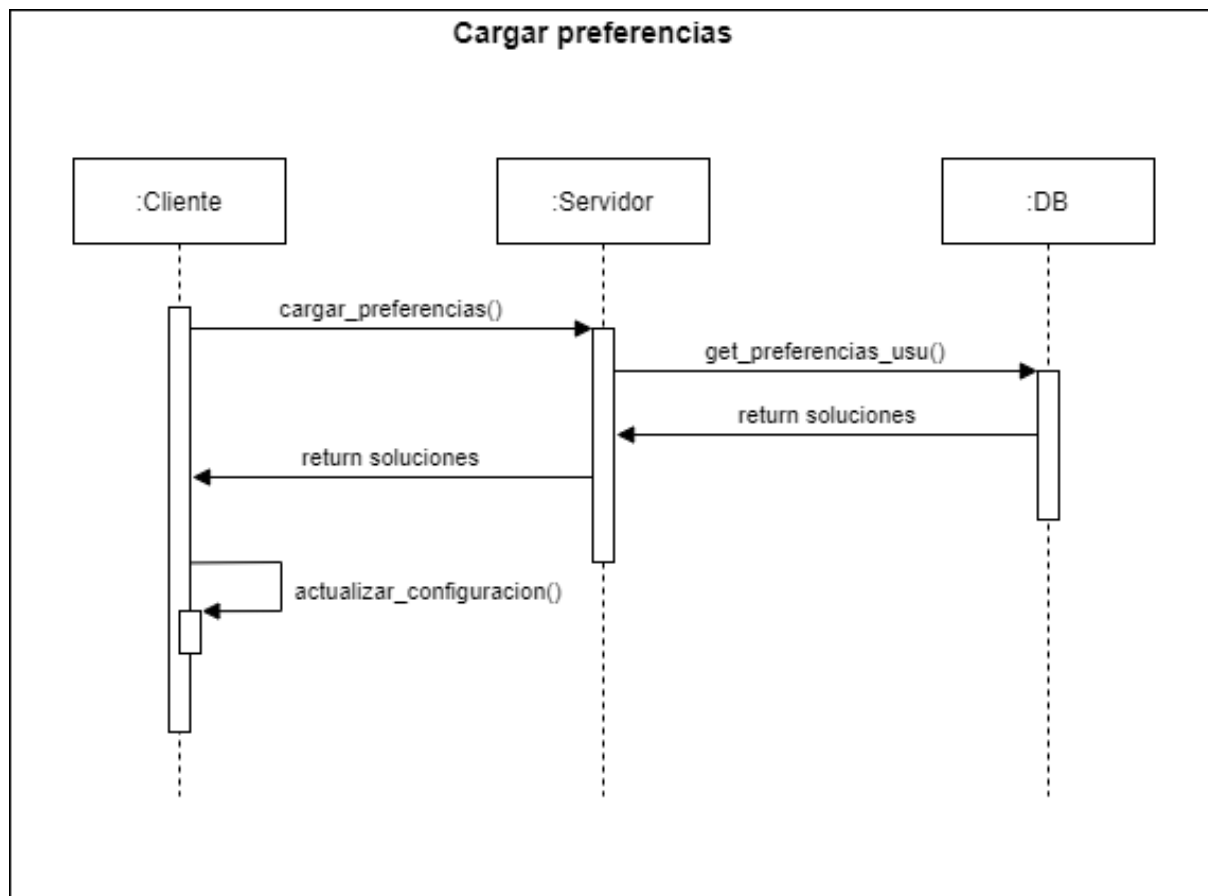


Figura 8.5: Diagrama de secuencia Cargar parámetros

Diagrama de secuencia Guardar parámetros

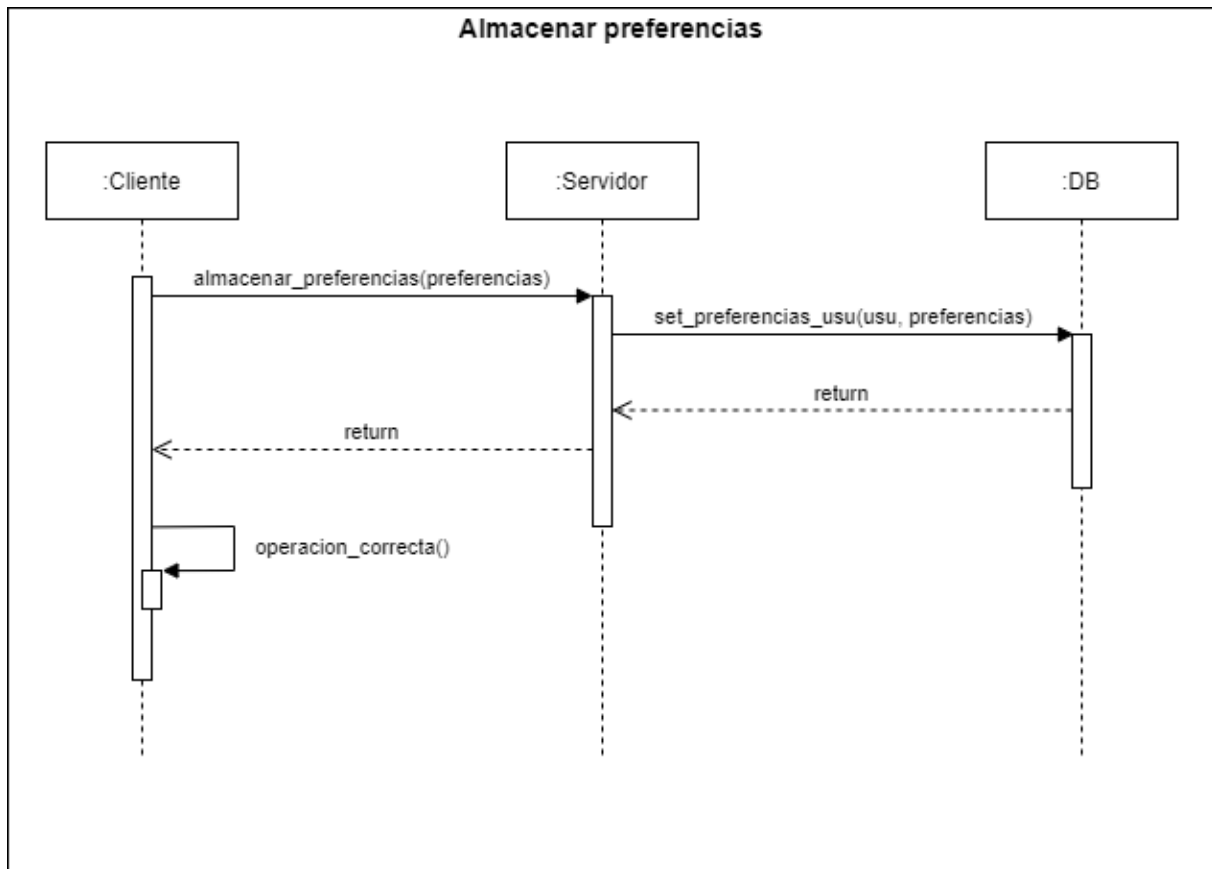


Figura 8.6: Diagrama de secuencia Guardar parámetros

8.3. Interfaz de usuario

La interfaz gráfica de la aplicación, consta de dos partes bien diferenciadas: una en la que se muestra la solución al problema de distribución en planta y un modal que contiene el menú de configuración de los parámetros de visualización.

The screenshot displays a web application interface titled "DISTRIBUCIÓN EN PLANTA". In the top right corner, there is a text input field labeled "username". Below the title, a horizontal navigation bar contains four buttons: "Problemas", "Configuraciones", "Ejecuciones", and "Soluciones". The "Soluciones" button is currently selected. Below this bar, there is a search input field and two buttons: "Volver" and "Configuración". The main content area features a layout diagram on the left with four labeled rectangular blocks: "A" (top-left), "B" (top-right), "C" (bottom-right), and "D" (bottom-left). To the right of the diagram is a section titled "Información de ejecución.." followed by five empty text input fields. At the bottom center of the interface, there are three small numbered buttons: "1", "2", and "3".

Figura 8.7: Interfaz. Vista previa del detalle de la solución

Configuración

Colores

Flujo

Adyacencias

Relación de aspecto

Guardar config.

Cargar config.

CheckBoxs

Reestablecer

Aceptar

Figura 8.8: Interfaz. Modal de configuración de parámetros de visualización

Capítulo 9

Pruebas

La realización de las pruebas de funcionamiento es una parte esencial en el ciclo de vida de cualquier desarrollo de software. En este capítulo, se detallará el plan de pruebas seguido desde la primera fase de desarrollo hasta las versiones finales del sistema.

9.1. Pruebas de caja blanca

Como actividad inherente al desarrollo, se han ido probando todos y cada uno de los métodos que componen los módulos en el momento de su codificación.

9.2. Pruebas de caja negra

En las últimas etapas del proyecto, comenzaron a realizarse pruebas para comprobar:

- Coherencia entre los datos introducidos y la salida generada.
- Correcta funcionalidad de la interfaz.
- Tiempo de respuesta óptimo.
- Respuesta del sistema ante la falta de datos.
- Comunicación con el back end.

9.3. Pruebas de integración

Las pruebas de integración han sido parte del propio desarrollo, ya que todos los módulos interactúan entre si.

9.4. Pruebas de representación

Una vez el sistema se encuentra completamente integrado y en su versión final, se procede a supervisar de manera detallada que la representación de cada uno de los elementos es la esperada.

9.4.1. Caso de prueba 01

Representación del flujo de materiales mediante el tipo *Distancia Euclídea*. Los datos usados corresponden a un problema y solución al azar. Los valores de la matriz de flujos son los siguientes:

	1	2	3	4	5	6	7
1	0	0	0	5	0	0	1
2	0	0	0	3	0	0	1
3	0	0	0	2	0	0	1
4	0	0	0	0	4	4	0
5	0	0	0	0	0	0	2
6	0	0	0	0	0	0	1
7	0	0	0	0	0	0	0

Cuadro 9.1: Ejemplo matriz de flujos

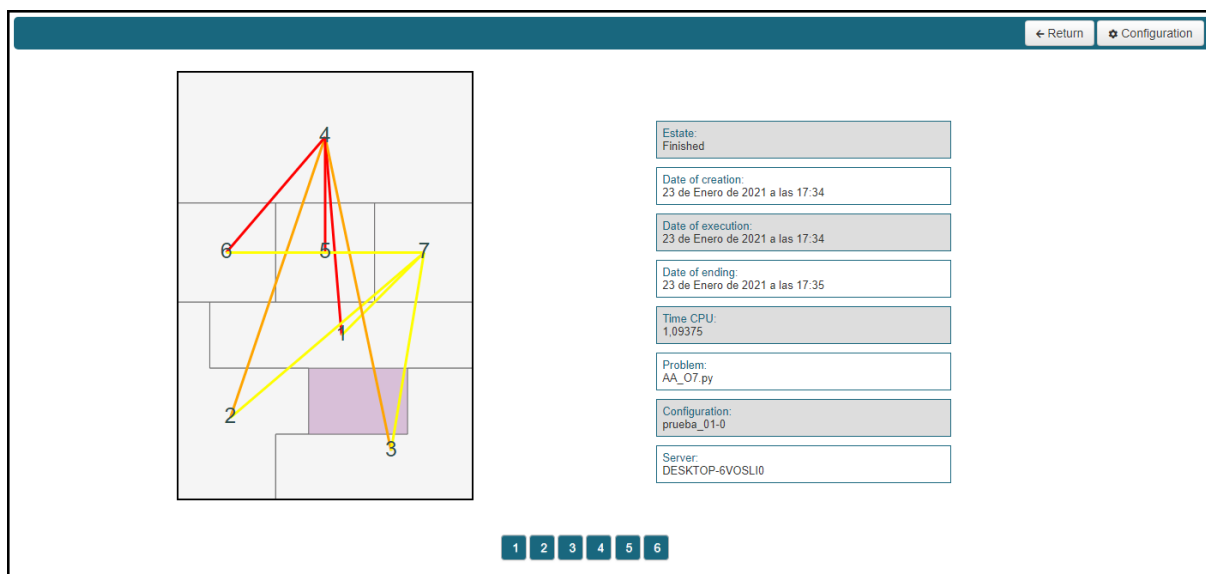


Figura 9.1: Flujo mediante distancia Euclídea y color de las líneas

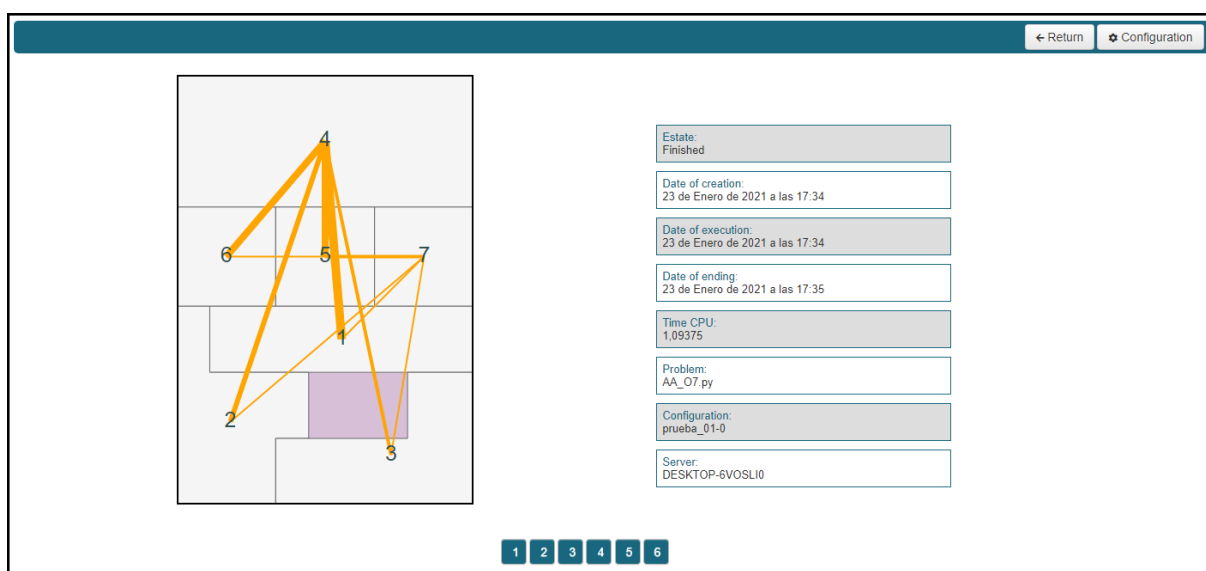


Figura 9.2: Flujo mediante distancia Euclídea y grosor de las líneas

9.4.2. Caso de prueba 02

Representación del flujo de materiales mediante el tipo *Distancia de Manhattan*. Los datos usados corresponden a un problema y solución al azar. Los valores de la matriz de flujos son los mismos que en la sección anterior.

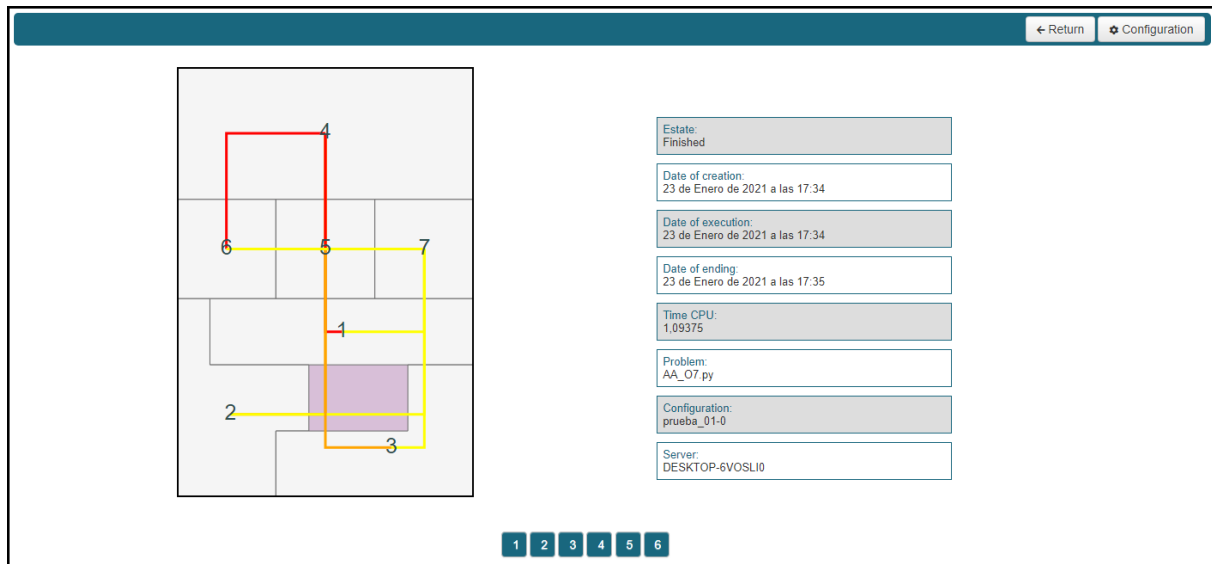


Figura 9.3: Flujo mediante distancia de Manhattan y color de las líneas

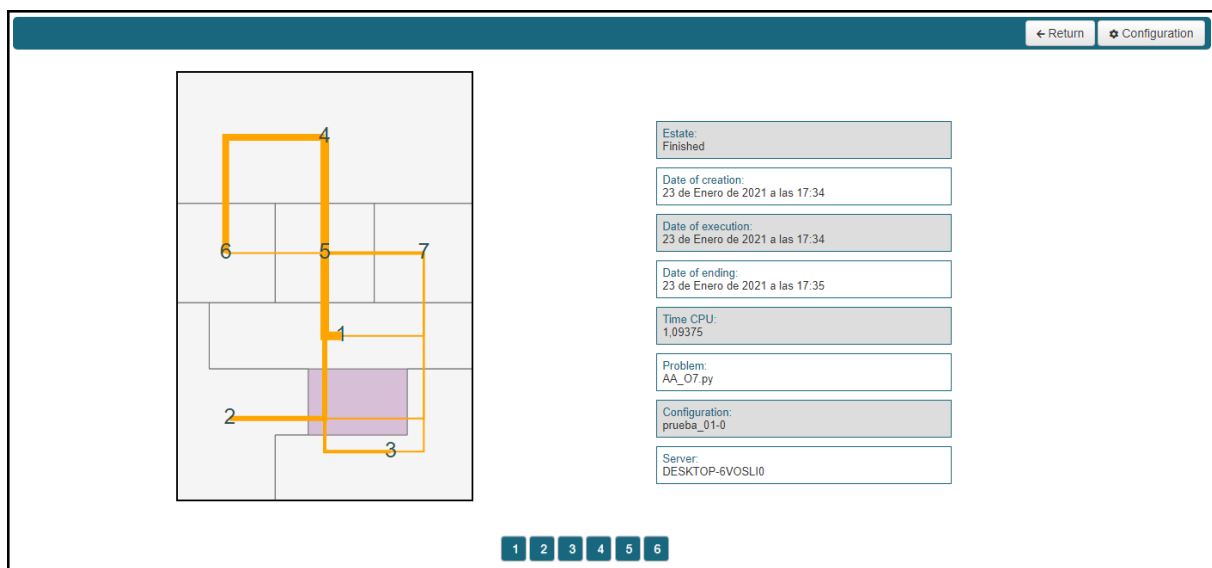


Figura 9.4: Flujo mediante distancia de Manhattan y grosor de las líneas

9.4.3. Caso de prueba 03

Representación del cumplimiento de adyacencias entre instalaciones mediante el tipo *Resaltar lado común*. Los datos usados corresponden a un problema y solución al azar. Los valores de la matriz de adyacencias son los siguientes:

	1	2	3	4	5	6	7
1	0	-8	0	4	0	0	0
2	0	0	0	2	0	0	0
3	0	0	0	1	0	0	0
4	0	0	0	0	3	3	0
5	0	0	0	0	-4	-4	1
6	0	0	-4	0	0	0	0
7	0	0	0	0	0	0	0

Cuadro 9.2: Ejemplo matriz de adyacencias

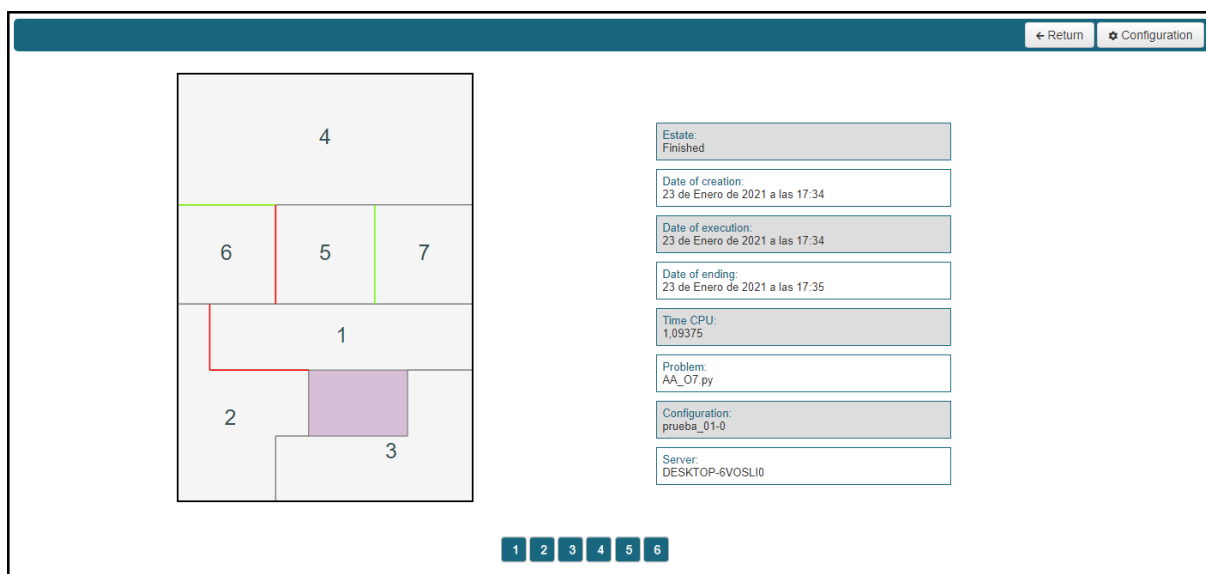


Figura 9.5: Adyacencia mediante resaltado de lado común y color de las líneas

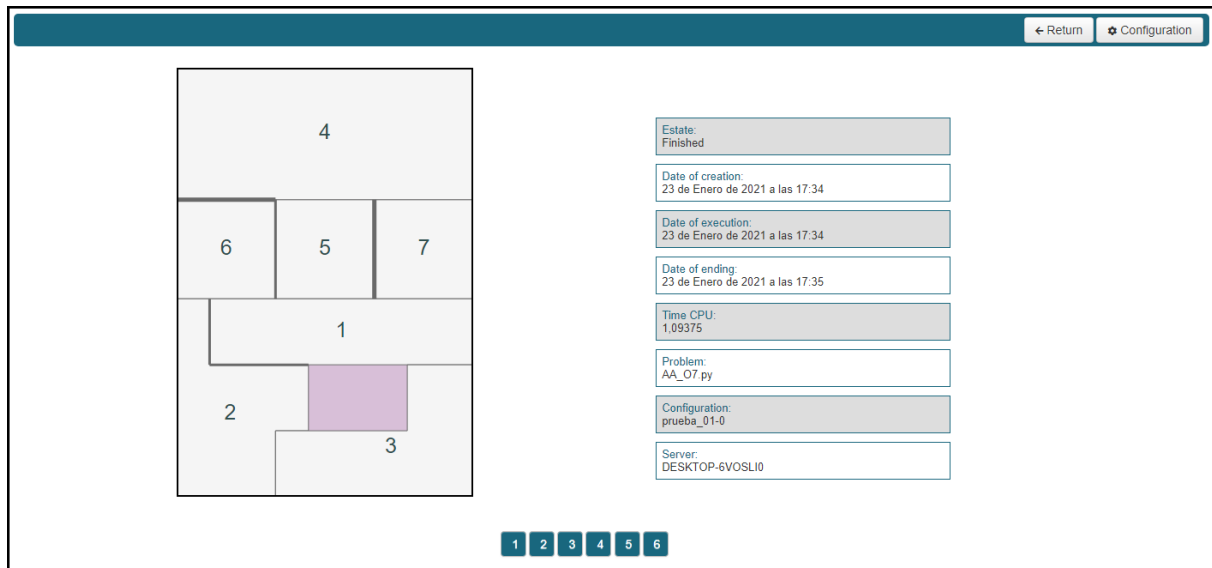


Figura 9.6: Adyacencia mediante resaltado de lado común y grosor de las líneas

9.4.4. Caso de prueba 04

Representación del cumplimiento de adyacencias entre instalaciones mediante el tipo *Línea de unión*. Los datos usados corresponden a un problema y solución al azar. Los valores de la matriz de adyacencias son los mismos que en la sección anterior.

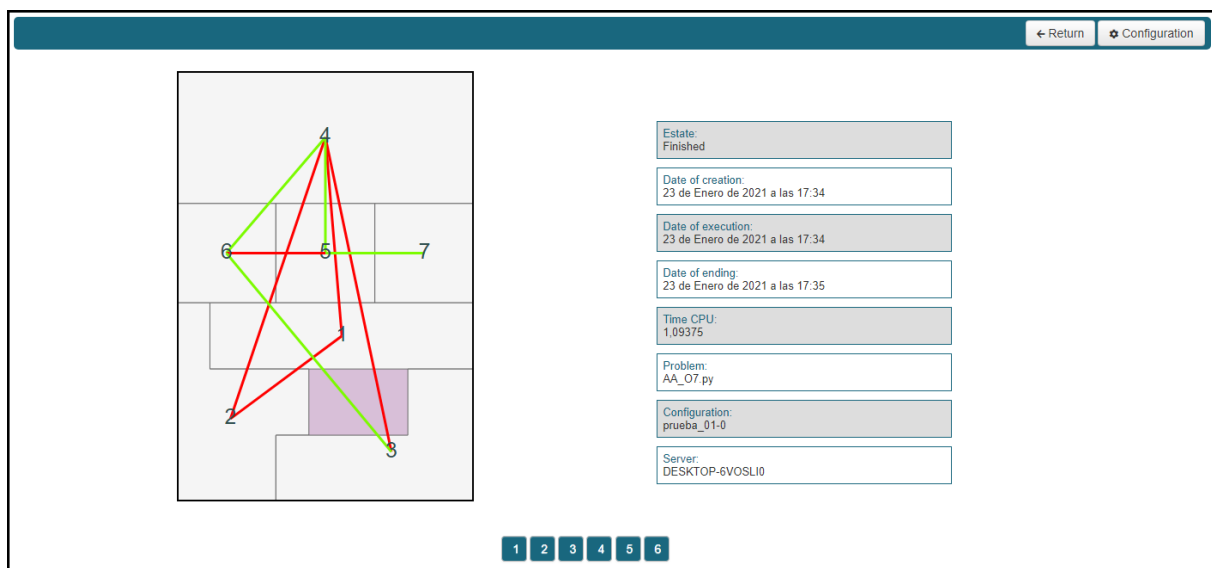


Figura 9.7: Adyacencia mediante línea de unión y color de las líneas

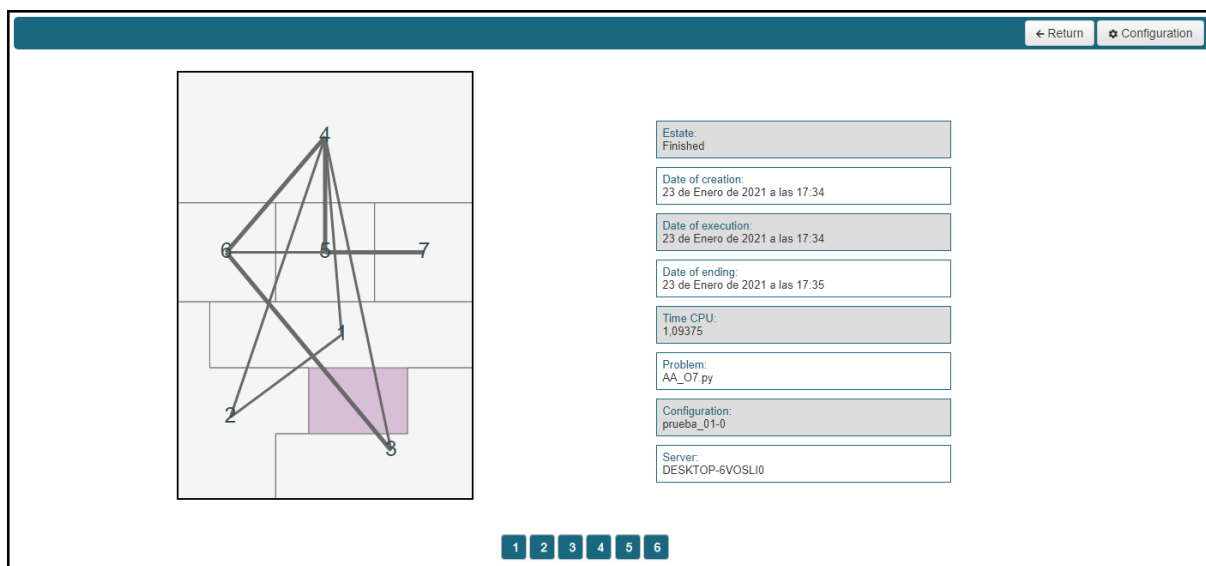


Figura 9.8: Adyacencia mediante línea de unión y grosor de las líneas

9.4.5. Caso de prueba 05

Representación de la relación de aspecto mediante un solo color. Los datos usados corresponden a un problema y solución al azar.

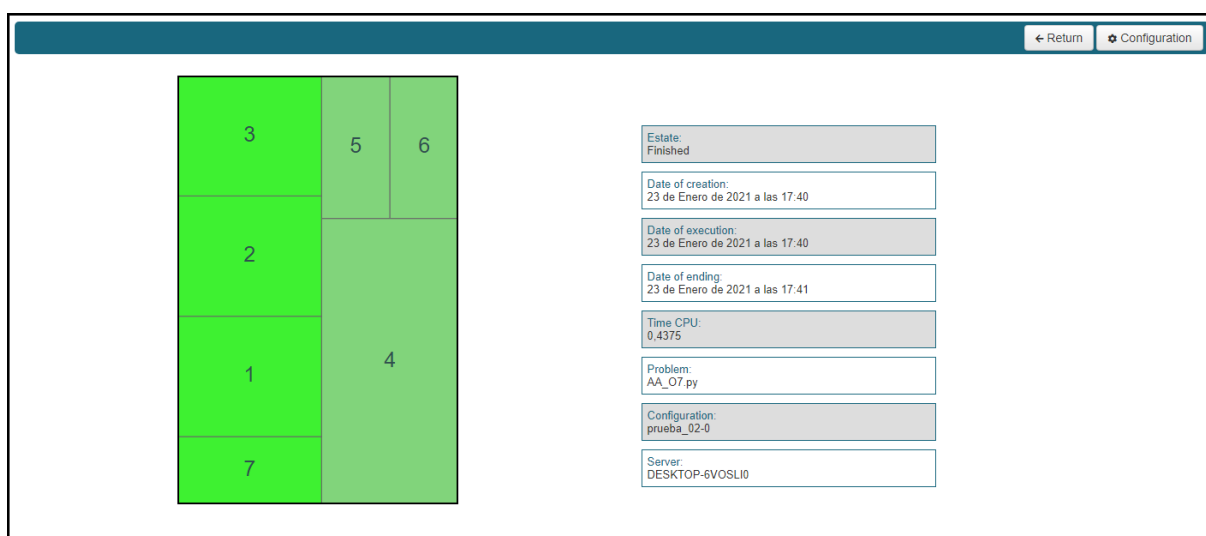


Figura 9.9: Relación de aspecto mediante un color

9.4.6. Caso de prueba 06

Representación de los espacios vacíos con fondo liso o con estampado.

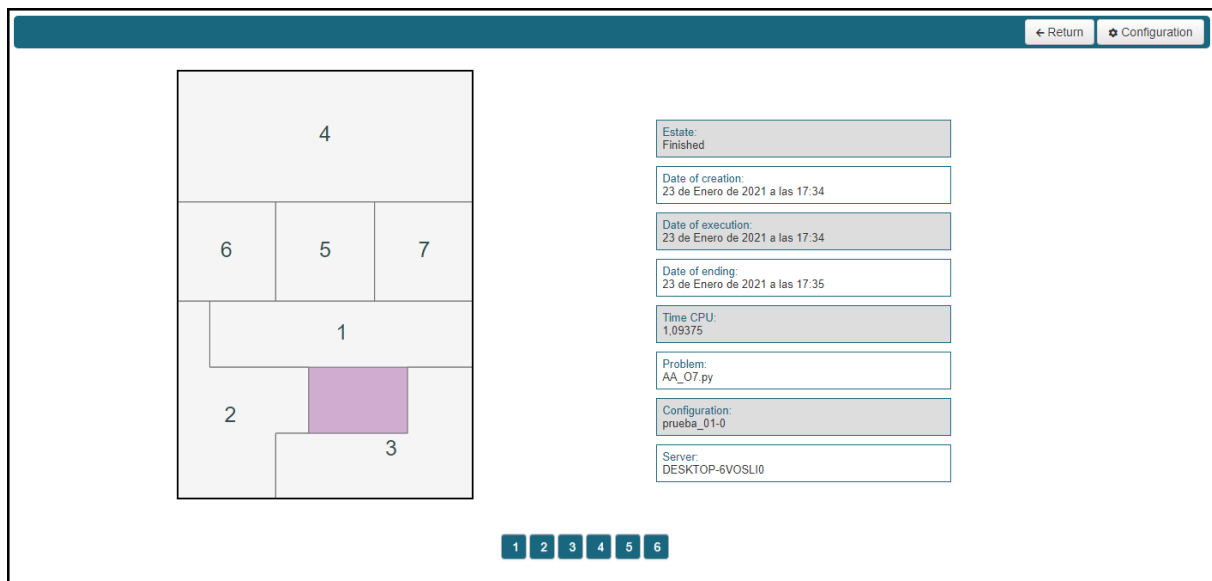


Figura 9.10: Espacio vacío mediante fondo liso

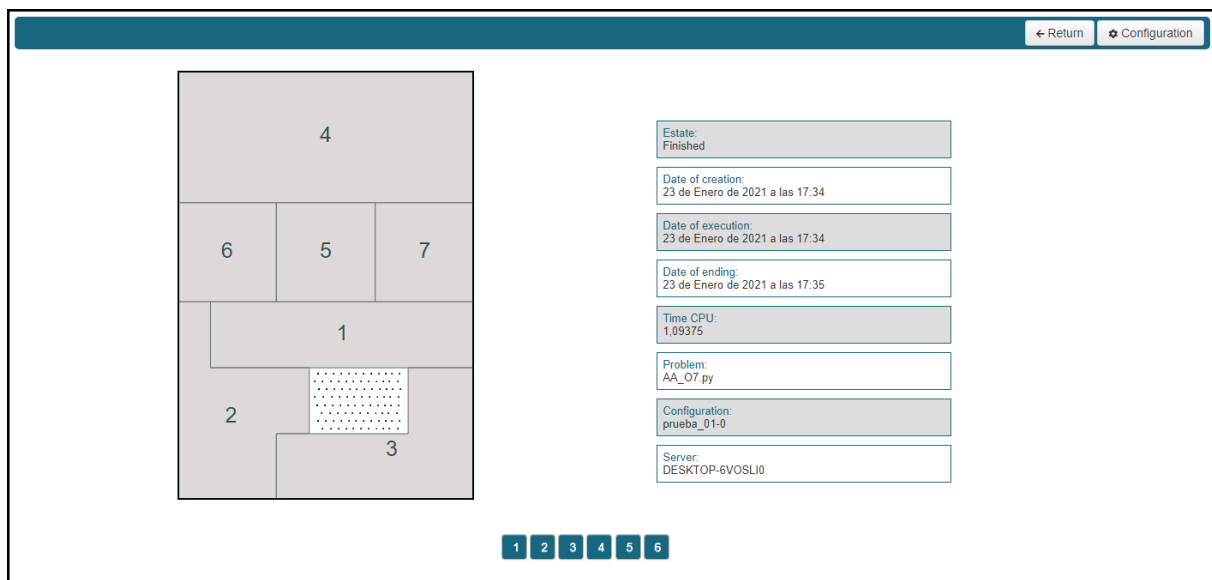


Figura 9.11: Espacio vacío mediante estampado de puntos

9.4.7. Caso de prueba 07

En la siguiente imagen, podemos observar que en el modal tenemos las opciones de cargar una configuración previamente guardada o de guardar una nueva. En el primer caso, se hace una llamada al servidor para comprobar si ese usuario tiene alguna configuración de parámetros guardada, y si es así, se actualiza el modal con esos datos. En el segundo caso, se guardan en la base de datos las preferencias de visualización de ese usuario para que pueda recuperarlas en otro momento.

The image shows a modal window titled "Configuración" with a close button (X) in the top right corner. The modal contains several configuration options:

- Top section: Four input fields for "Fondo de las instalaciones:", "Líneas delimitantes:", "Nombres de las instalaciones:", and "Líneas de flujo:", each with a small square icon.
- "Flujo" section: A dropdown menu for "Calcular el flujo según:" set to "Distancia Euclídea", and another dropdown for "Representar el flujo mediante:" set to "Grosor de las líneas".
- "Adyacencias" section: A dropdown menu for "Representar la adyacencia:" set to "Resaltando lados en común", and another dropdown for "Las líneas de adyacencia variarán su:" set to "Color".
- Bottom section: Two buttons, "Cargar config." (blue) and "Guardar config." (green), both with document icons.
- Footer: Three buttons: "Cerrar" (white with X), "Reset" (orange), and "Aplicar cambios" (white with checkmark).

Figura 9.12: Opciones de Almacenar/Cargar configuración

9.4.8. Caso de prueba 08

En la figura 9.13 se muestra como se selecciona que sean representados los flujos y adyacencias de ciertas instalaciones y en la figura 9.14 se aprecia que el resultado es el esperado.

Figura 9.13: Seleccionar instalaciones

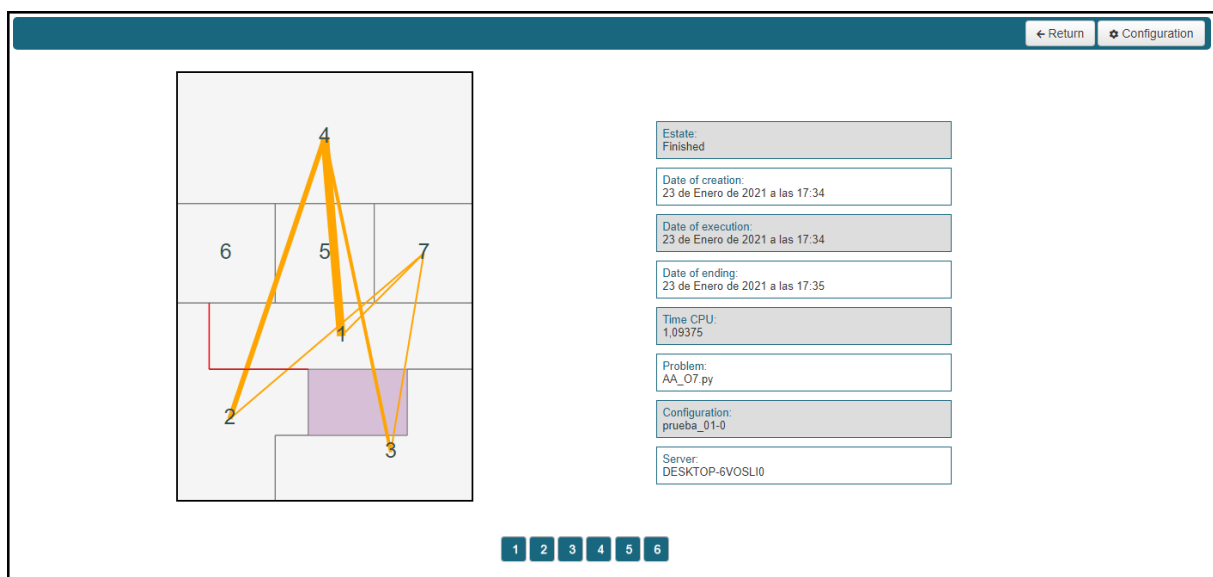


Figura 9.14: Mostrar flujos y adyacencias de las instalaciones seleccionadas

9.4.9. Caso de prueba 09

En el modal se muestra como se pueden modificar los colores en los que se representan varios elementos.



Figura 9.15: Modificar colores de los distintos elementos

9.5. Matriz de cumplimiento

En el cuadro 9.3 se comprobará que las pruebas realizadas cubren todos los requisitos funcionales definidos en el capítulo 7 de este documento.

Requisito	Caso de prueba
RF-1	01, 02, 03, 04, 05, 06, 08
RF-2	01, 02, 03, 04, 05, 06, 08
RF-3	09
RF-4	01, 02
RF-5	01, 02
RF-6	03, 04
RF-7	03, 04
RF-8	08
RF-9	07
RF-10	06
RF-11	05

Cuadro 9.3: Matriz de cumplimiento de requisitos

Capítulo 10

Conclusiones

Durante la realización del presente proyecto he ampliado mis conocimientos sobre el ciclo de vida de desarrollo de un software, comprobando que es de vital importancia realizar una buena planificación temporal del trabajo para abordar las fases de estudio, análisis, diseño e implementación y pruebas de forma ordenada y coherente, sin perder nunca de vista los objetivos marcados.

Es cierto que se han presentado más dificultades de las previstas, especialmente, en lo respectivo a la puesta en marcha de la aplicación. No obstante, los objetivos han sido cumplidos de manera positiva y el resultado ha sido satisfactorio.

Capítulo 11

Futuras mejoras

Una vez concluido, y para hacer más sencilla una posible ampliación del presente proyecto, se mencionarán algunas ideas que podrían desarrollarse en un futuro para añadir más valor a la aplicación:

- Migrar la aplicación a una versión más reciente de python para prevenir errores de compatibilidades.
- Actualizar los estilos de la aplicación web para poder usar elementos actuales.
- Hacer la aplicación completamente *responsive*.
- A la hora de comprobar la relación de aspecto, no se ha contemplado la posibilidad de que el cuadrado/rectángulo inscrito en la instalación esté rotado.

Capítulo 12

Planificación del proyecto

El plan de trabajo y calendario que ha sido seguido durante el desarrollo del proyecto ha sido el que se desglosa a continuación:

Tareas	Tiempo(h)
Aprendizaje y familiarización con las tecnologías a usar en el proyecto.	40
· <i>HTML5 y CSS3</i>	5
· <i>Javascript</i>	15
· <i>SVG</i>	10
· <i>D3.js</i>	10
Investigación de la aplicación que genera las soluciones, puesta en marcha y adaptación.	65
Desarrollo e implementación de la solución	110
· <i>Estudio de variables</i>	10
· <i>Diseño</i>	45
· <i>Integración en la aplicación web</i>	10
· <i>Programación de funcionalidades</i>	45
Realización de pruebas	25
Documentación	60
Total	300

Cuadro 12.1: Distribución temporal

Además, se ha utilizado un diagrama de Gantt para distribuir temporalmente las fases del proyecto:

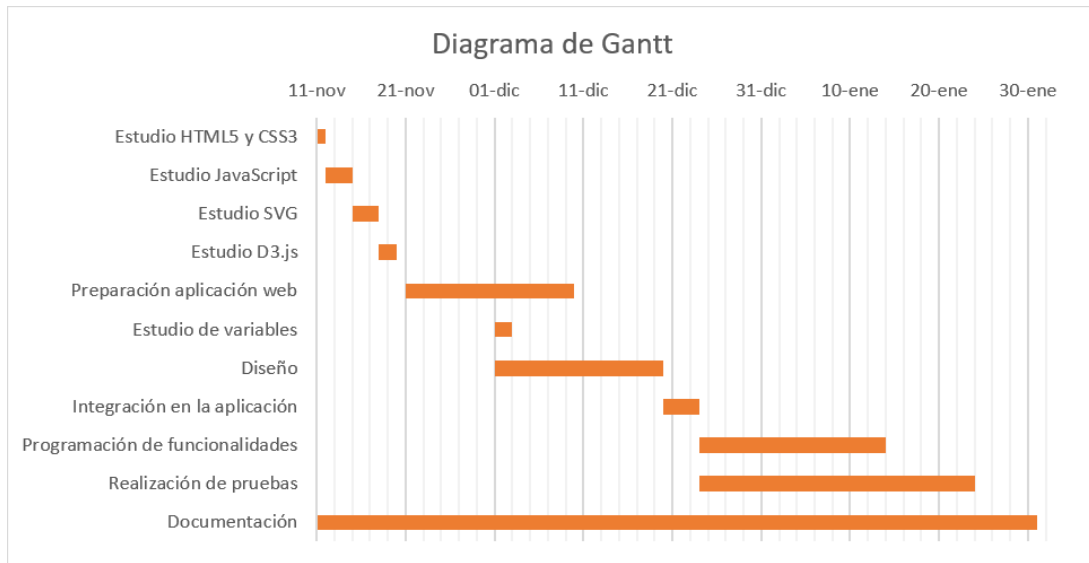


Figura 12.1: Diagrama de Gantt

Bibliografía

- [1] J. Rumbaugh, G. Booch, and I. Jacobson. Unified modeling language, 2015. URL https://es.wikipedia.org/wiki/Lenguaje_unificado_de_modelado.
- [2] Miguel Ángel Cid García. *Sistema Web para la experimentación en diseño de distribuciones en planta*. Proyecto de fin de carrera, Escuela Politécnica Superior de la Universidad de Córdoba, 2013.
- [3] Distancia euclídea, . URL https://www.ecured.cu/Distancia_eucl%C3%ADdea.
- [4] Geometría del taxista, . URL https://es.wikipedia.org/wiki/Geometr%C3%ADa_del_taxista.
- [5] Francisco Raso Lucena. *Visualizador de resultados de algoritmos genéticos para distribución en planta*. Proyecto de fin de carrera, Escuela Politécnica Superior de la Universidad de Córdoba, 2012.
- [6] Poo en javascript, 2019. URL <https://davidinformatico.com/poo-en-javascript/>.

Apéndice A

Manual de usuario

A.1. Introducción

Este manual, está destinado al usuario final para orientarlo en la instalación, manejo y desinstalación de la aplicación. Si se desea información detallada del funcionamiento o diseño se recomienda consultar el manual técnico.

A.2. Instalación

A continuación, se detallarán los pasos a seguir para la instalación del producto.

A.2.1. Crear entorno Virtual

Haciendo uso de Python=2.7 y el sistema de gestión de paquetes *pip*, instalar *virtualenv*:

- `pip install virtualenv`

Una vez instalado, crear el entorno virtual:

- `virtualenv NOMBRE_ENTORNO`

Cuando el entorno se haya creado, accedemos a la carpeta `NOMBRE_ENTORNO` y lanzamos el entorno:

- `./Scripts/activate`

A.2.2. Instalación de paquetes

Los paquetes a instalar dentro del entorno son los siguientes:

- `pip install Django==1.5.12`
- `pip install backports.functools-lru-cache==1.5`
- `pip install confusable-homoglyphs==3.2.0`
- `pip install cycler==0.10.0`
- `pip install django-email-as-username==1.6.7`
- `pip install django-modeltranslation==0.6.1`
- `pip install django-recaptcha==0.0.6`
- `pip install django-registration==0.7`
- `pip install kiwisolver==1.0.1`
- `pip install matplotlib==2.2.3`
- `pip install numpy==1.16.1`
- `pip install Pillow==5.4.1`
- `pip install pyparsing==2.3.1`
- `pip install python-dateutil==2.8.0`
- `pip install pytz==2018.9`
- `pip install six==1.12.0`
- `pip install subprocess32==3.5.3`

A.2.3. Descargar aplicación

Descargar la carpeta con la aplicación web y copiarla en `/NOMBRE_ENTORNO/`.

- `https://dl.dropbox.com/s/vsx1cd3f5emr16i/distlant.tar?dl=1`

A.2.4. Configuración y sincronización de la base de datos

Para llevar a cabo este paso, será necesario que instale en su equipo (fuera del entorno virtual) el gestor PostgreSQL.

- <https://www.postgresql.org/download/>

Cuando haya finalizado la instalación, deberá lanzar su entorno virtual e instalar un paquete para que se comuniquen con su gestor de bases de datos:

- `pip install psycopg2`

A continuación, cree una nueva base de datos desde PostgreSQL y en el archivo `/NOMBRE_ENTORNO/distlant/settings.py`, configure la conexión a la base de datos con los mismos parámetros.

- `'ENGINE': 'django.db.backends.postgresql_psycopg2',`
- `'NAME': 'DB_NAME',`
- `'USER': 'USER_NAME',`
- `'PASSWORD': 'PWD',`
- `'HOST': 'localhost',`
- `'PORT': '5432',`

Cuando finalice este procedimiento, sincronizar la base de datos y crear un superusuario:

- `python manage.py syncdb`

A.2.5. Lanzar el proyecto

Por último, solo será necesario lanzar el proyecto:

- `python manage.py runserver`

A.2.6. Posibles errores

Si durante este proceso ha surgido algún tipo de error, descargar la siguiente carpeta e instalar los archivos en el entorno virtual:

- <https://dl.dropbox.com/s/n3kpot0tbinm7u9/solveErrors.tar?dl=1>

Para instalar tan solo será necesario:

- `pip install NOMBRE_ARCHIVO`

A.3. Entorno virtual preconfigurado

Si así se desea, se puede descargar directamente un entorno virtual ya configurado:

- <https://dl.dropbox.com/s/xib2un5p1jhyy46/venv.tar?dl=1>

En este caso, solo será necesario seguir los siguientes pasos del proceso de instalación:

- Usar Python=2.7 en el sistema local.
- Lanzar el entorno.
- Configurar y sincronizar la base de datos.
- Lanzar el proyecto.

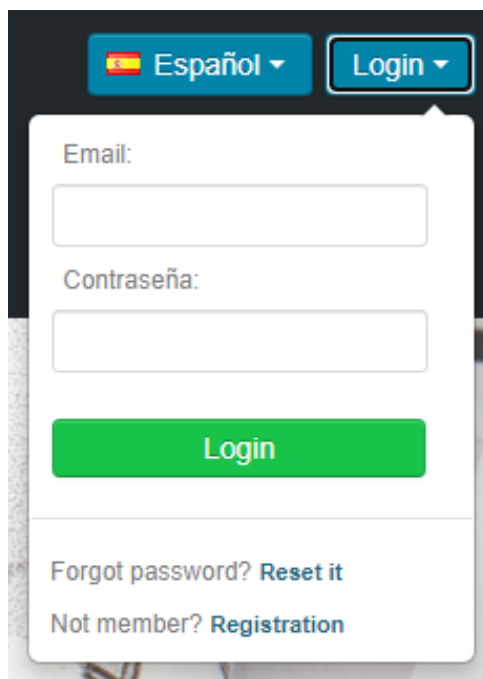
A.4. Uso de la aplicación web

Una vez la aplicación ha sido lanzada el funcionamiento es sencillo e intuitivo.

A.4.1. Pantalla de Inicio



Figura A.1: Manual de usuario. Inicio



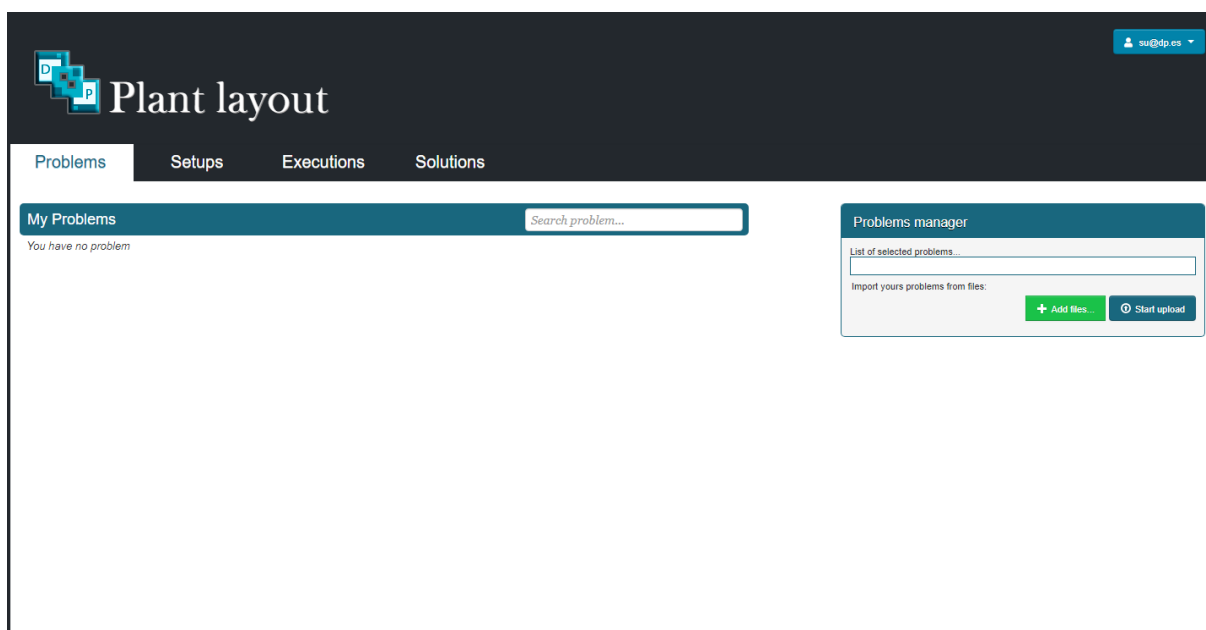
The image shows a login form with a dark background. At the top, there is a language selector with a Spanish flag and the text 'Español', and a 'Login' button. Below these are two input fields: 'Email:' and 'Contraseña:'. A green 'Login' button is positioned below the password field. At the bottom, there are two links: 'Forgot password? Reset it' and 'Not member? Registration'.

Figura A.2: Manual de usuario. Login

Como podemos observar, nada más lanzar la aplicación nos encontramos con la pantalla principal y el *login* para poder acceder al contenido.

A.4.2. Home - Problemas

Una vez estamos registrados y logados, se nos abrirá la pestaña de problemas, ya que es el primer paso a realizar para obtener una solución.



The image shows the 'Home - Problemas' screen of the application. The header is dark blue with the 'Plant layout' logo and a user profile icon labeled 'su@dp.es'. Below the header is a navigation bar with tabs: 'Problems', 'Setups', 'Executions', and 'Solutions'. The 'Problems' tab is active. The main content area is divided into two sections. On the left, 'My Problems' shows a search bar and the message 'You have no problem'. On the right, 'Problems manager' shows a list of selected problems and a section for importing problems from files, with buttons for '+ Add files...' and 'Start upload'.

Figura A.3: Manual de usuario. Problemas

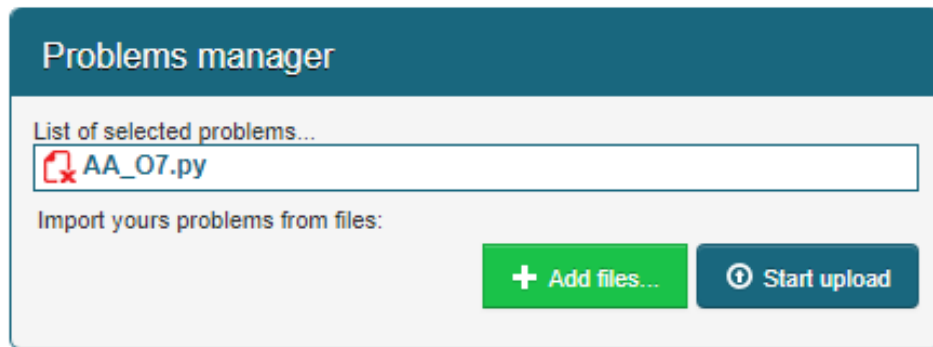


Figura A.4: Manual de usuario. Subir problema desde fichero

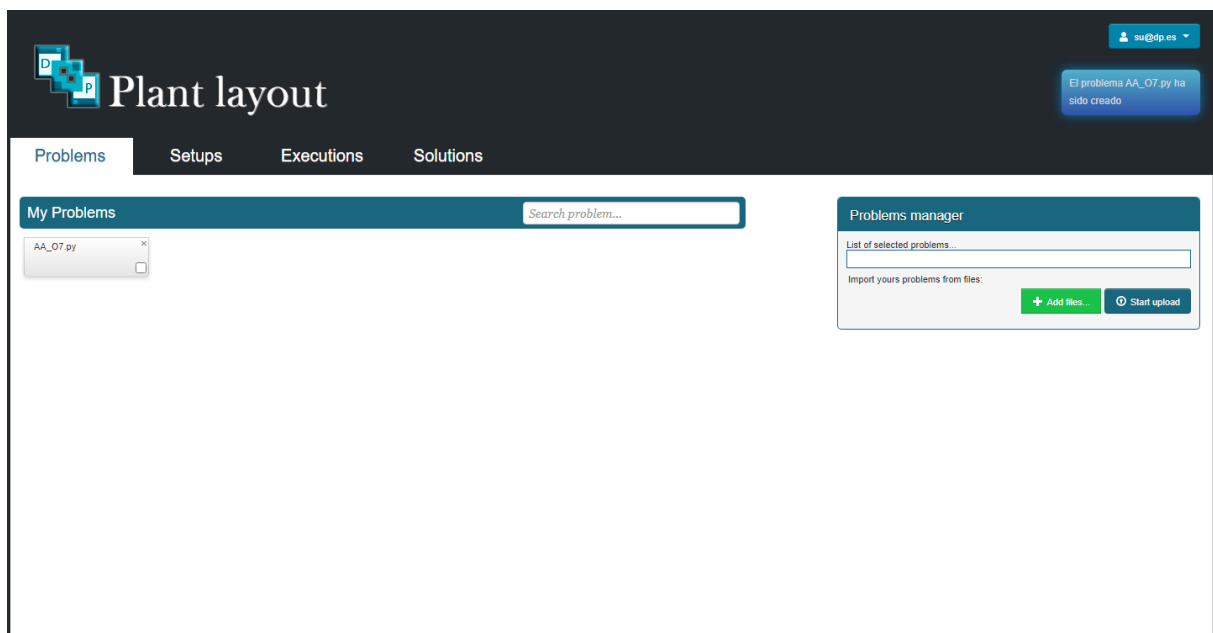


Figura A.5: Manual de usuario. Problema creado

Como se observa en las imágenes, para cargar nuestros problemas de distribución en planta disponemos de un gestor que interactúa directamente con nuestro explorador de archivos.

A.4.3. Configuraciones

Por otro lado, podemos crear una configuración en la que elegiremos con que algoritmo queremos resolver el problema en cuestión y varios aspectos más.

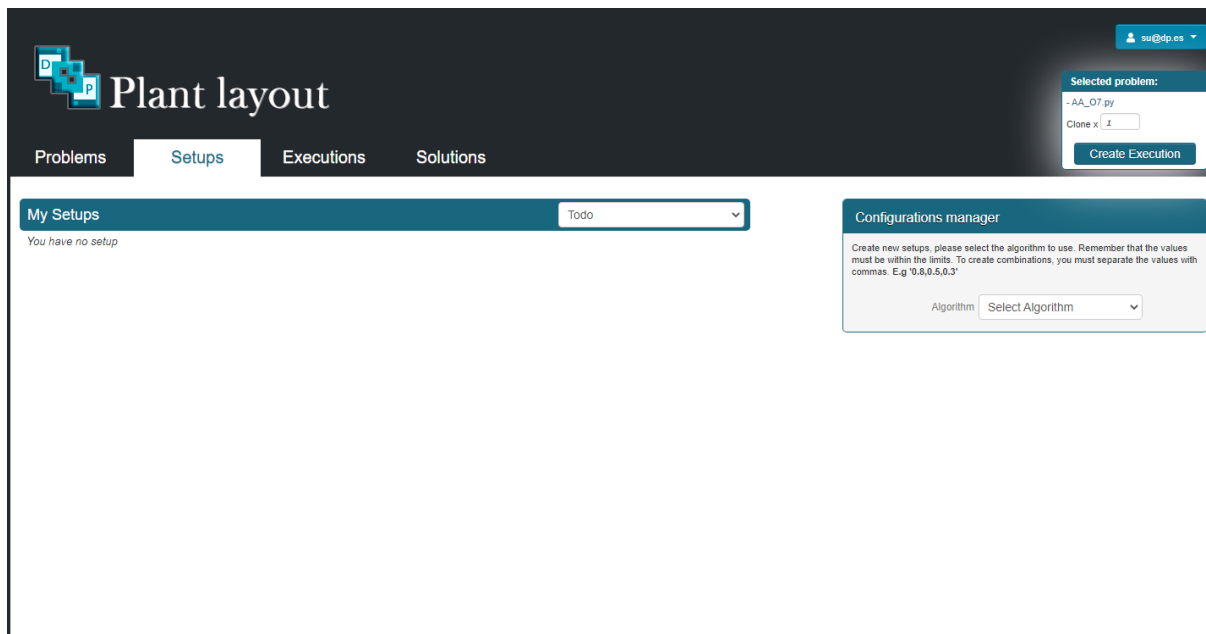


Figura A.6: Manual de usuario. Configuraciones

Configurations manager

Create new setups, please select the algorithm to use. Remember that the values must be within the limits. To create combinations, you must separate the values with commas. E.g '0.8,0.5,0.3'

Algorithm	<input type="text" value="cid"/>
Name Group	<input type="text" value="prueba_01"/>
crossover_method	<div><div>only</div><div>double</div><div>random</div></div>
crossover_prob	<input type="text" value="0.7"/>
elitism_num	<input type="text" value="1"/>
fitness_method	<div><div>mffc</div><div>tlc</div><div>cost</div><div>flow</div></div>
mutation_prob	<input type="text" value="0.2"/>
number_of_generations	<input type="text" value="10"/>
number_of_solutions	<input type="text" value="10"/>
population_size	<input type="text" value="100"/>
selection_method	<div><div>roulette</div><div>sigma_scaling</div><div>hierarchy</div><div>tournament</div></div>

Figura A.7: Manual de usuario. Gestor de configuraciones

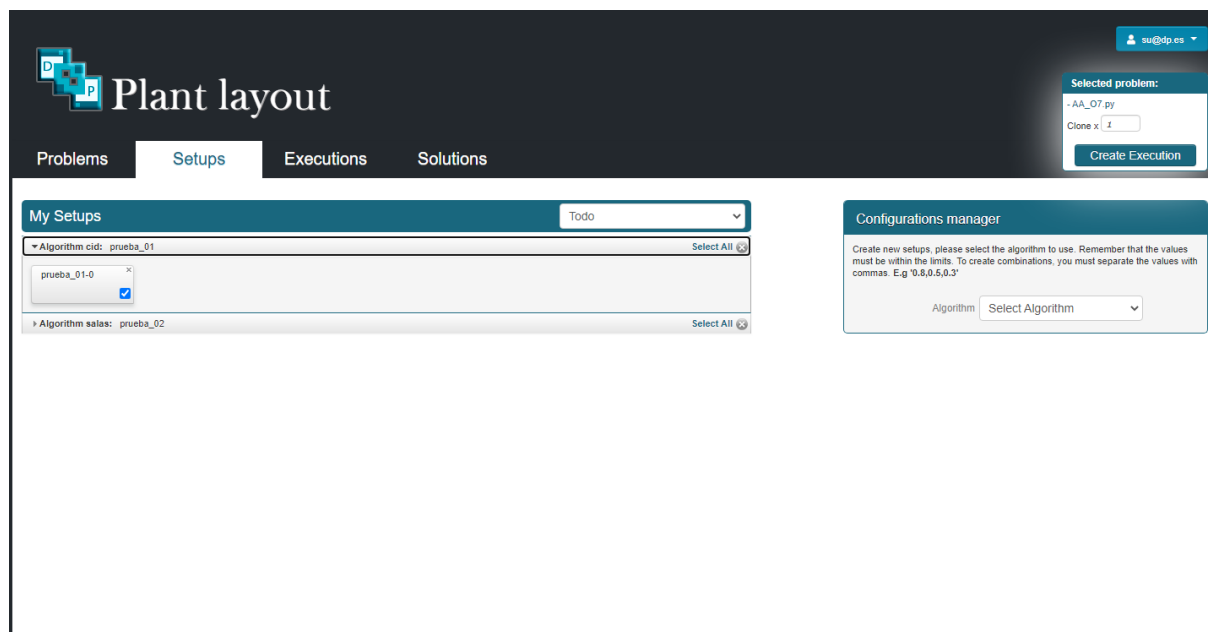


Figura A.8: Manual de usuario. Configuraciones creadas

Teniendo creada y seleccionada la configuración, y habiendo seleccionado el problema creado en la pestaña anterior, veremos que arriba a la derecha nos da la opción de iniciar una nueva ejecución.

A.4.4. Ejecuciones

En esta pestaña, podemos ver el proceso en el que se encuentran las ejecuciones. Si no se ha ejecutado el lanzador, permanecerán en espera como es el caso de la imagen.

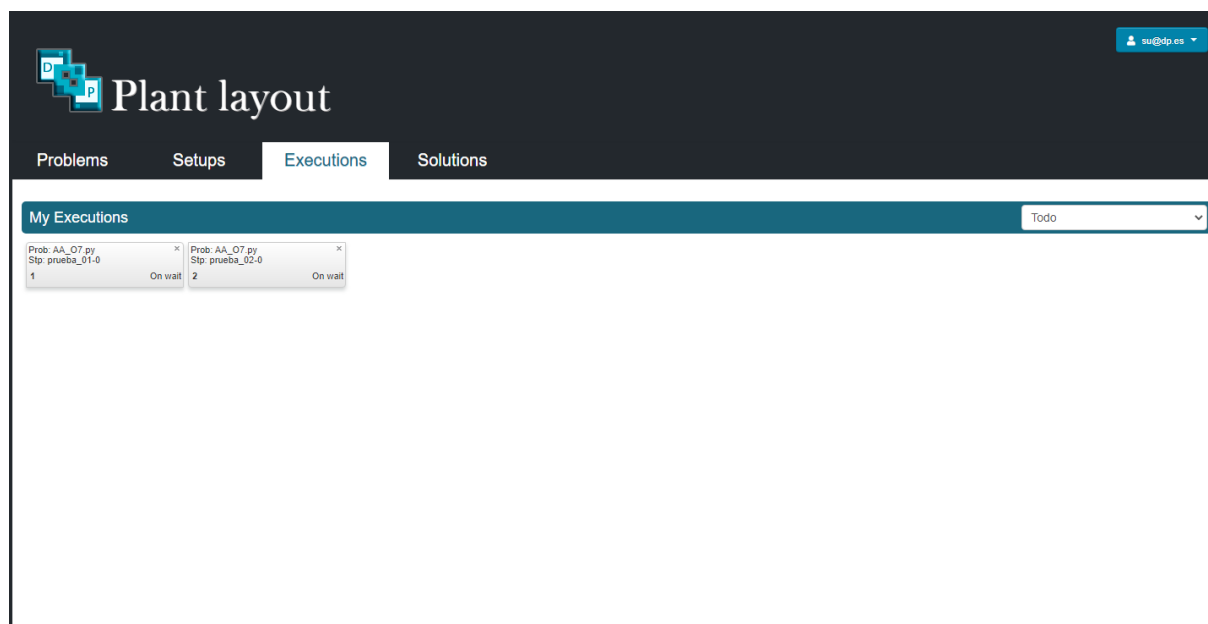


Figura A.9: Manual de usuario. Ejecuciones en espera

A.4.5. Soluciones

Por último, tenemos la pestaña de soluciones, las cuales aparecerán en caso de que la ejecución se haya completado satisfactoriamente.

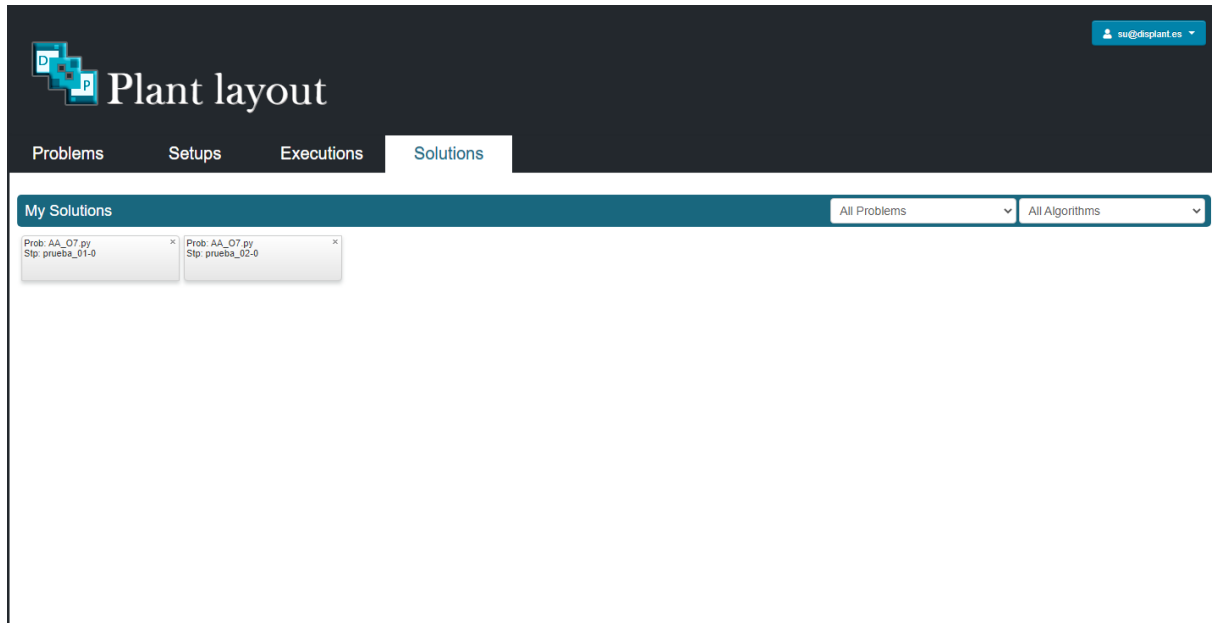


Figura A.10: Manual de usuario. Soluciones

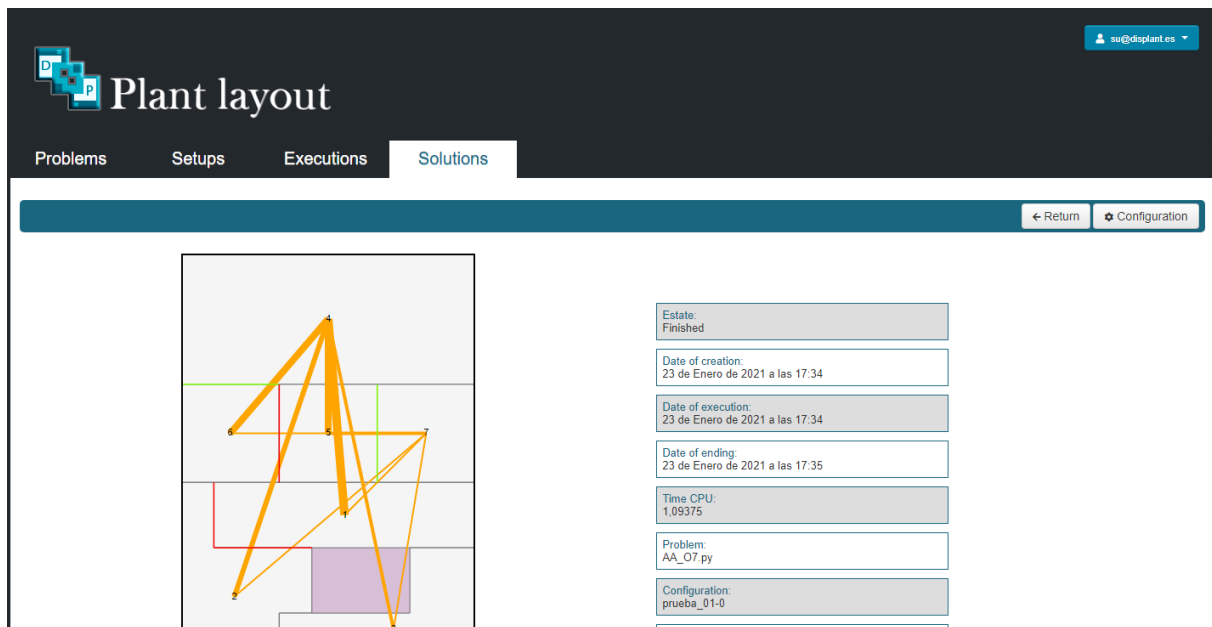


Figura A.11: Manual de usuario. Visualización de la solución

Si pulsamos en alguna de las soluciones generadas podemos ver todos los detalles.

Figura A.12: Manual de usuario. Menú de configuración de parámetros de visualización

Y en caso de que queramos modificar cualquiera de los elementos de la planta (flujos, adyacencias, colores, relación de aspecto, etc) tan solo tenemos que acudir al botón superior que dice 'Configuración', y realizar los cambios que consideremos oportunos. Desde este mismo menú, se dará la opción de almacenar una configuración de los parámetros para poder recuperarla cuando sea necesario.

A.5. Uso del lanzador

Para usar el lanzador, habrá que seguir los siguientes pasos:

- En el archivo `runner.py`, comprobar que los datos de las cadenas de conexión a la base de datos coinciden con los indicados en `settings.py`.
- Abrir una nueva instancia de la consola.
- Lanzar en ella el entorno virtual.
- Acceder desde la consola a `'distlant/dp'`.
- Ejecutar el lanzador (`python runner.py`).

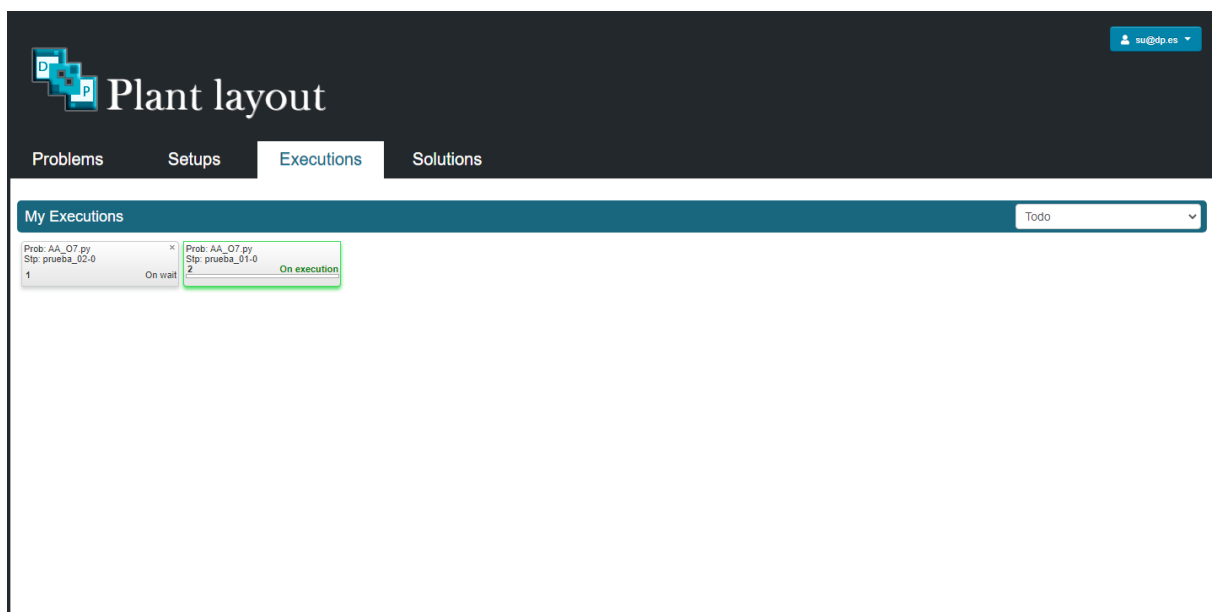


Figura A.13: Organización del código

Cuando el lanzador es ejecutado, observamos que las ejecuciones pasan del estado 'en espera' al estado 'en ejecución'.

A.6. Desinstalación

Si se desea desinstalar la aplicación, será suficiente con eliminar la carpeta del entorno virtual. Adicionalmente, se puede acceder a PostgreSQL, a través de su gestor gráfico pgAdmin, y eliminar la base de datos creada para la aplicación.

Apéndice B

Manual de código

B.1. Introducción

Este manual de código, pertenece al proyecto *Módulo web para la representación de distribuciones en planta con gráficos vectoriales*.

El lenguaje principalmente utilizado ha sido JavaScript, con él se ha diseñado toda la lógica de representación de los elementos de la planta, y se ha podido mostrar gráficamente el resultado en formato SVG, haciendo uso de la librería d3.js y sus métodos. En menor medida, se ha hecho uso de Python para comunicaciones con el servidor y para adaptar el propio módulo de visualización a la aplicación ya existente, la cual, está desarrollada con el framework Django. Finalmente, se ha hecho uso de HTML5 y CSS3 para todos los aspectos relativos a la interfaz gráfica del sistema.

B.2. Organización del código

En este apartado, se muestra a modo de esquema la forma en la que se encuentra organizado todo el código perteneciente al módulo de visualización.

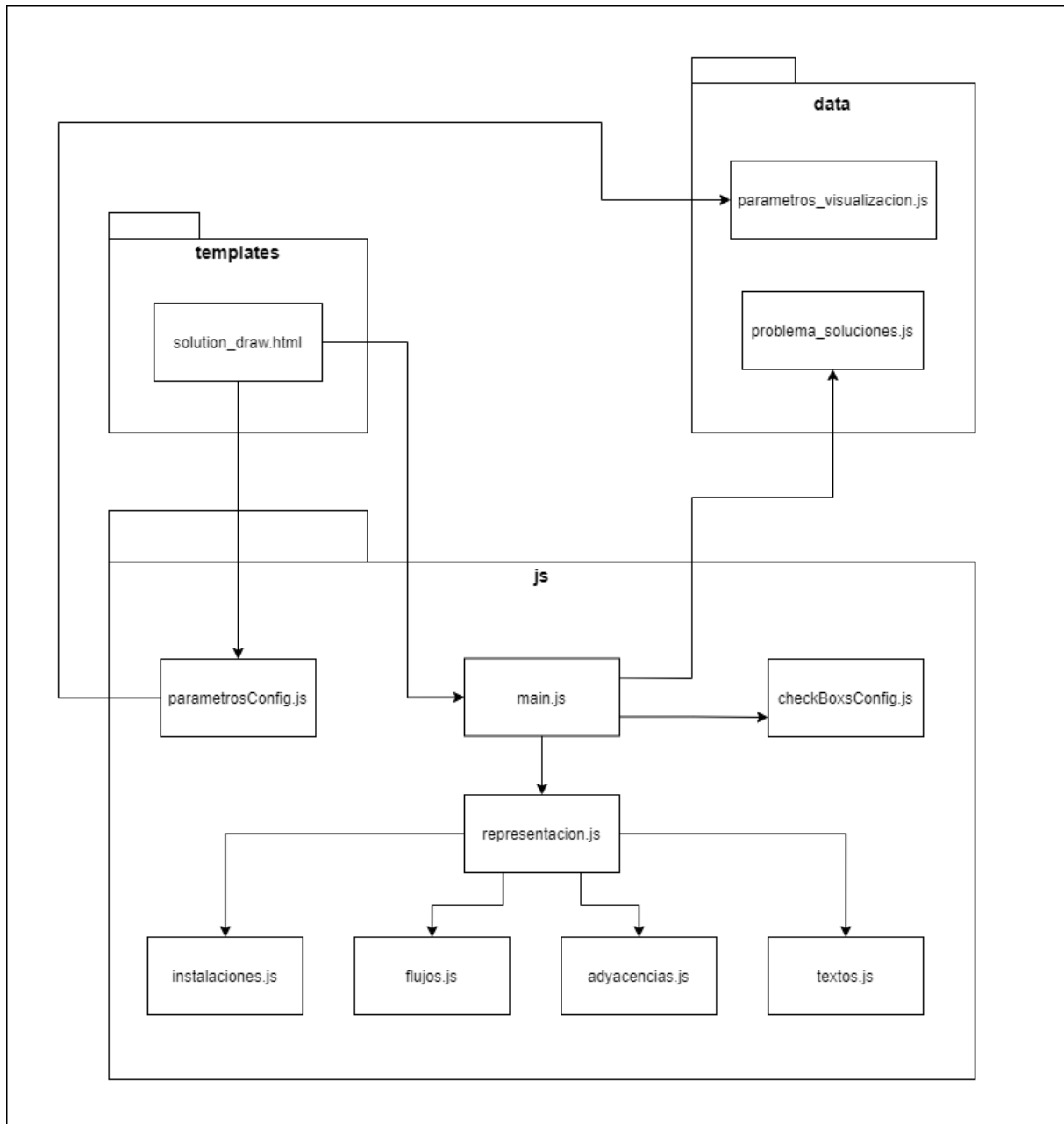


Figura B.1: Organización del código

Carpeta templates

Contiene todas las plantillas de la interfaz de la aplicación.

- **solution_draw.html.** Define los elementos que se muestran en la pantalla de visualización del detalle de la solución, se declaran las variables globales con las que se trabajará y se llama al archivo principal *main.js*.

Carpeta data

Contiene información relevante para el funcionamiento de la aplicación.

- **problema_soluciones.js.** Archivo que se recarga con la información del problema y soluciones generadas para cada caso en particular. Es llamado desde *main.js*.
- **parametros_visualizacion.js.** Archivo que se recarga con los parámetros de visualización almacenados por el usuario en cada momento.

Carpeta js

Contiene todos los archivos de JavaScript usados, tanto para elementos de diseño como para interacciones con el servidor.

- **main.js.** Contiene las configuraciones principales, como la carga de parámetros de visualización por defecto, inicialización de variables o creación de *CheckBoxes* en función del número y nombre de instalaciones para el problema en cuestión. Finaliza llamando al archivo de renderización de la planta *representacion.js*.
- **parametrosConfig.js.** Alberga las funciones que afectan a los parámetros de visualización, ya sea para aplicar cambios, restaurarlos por defecto, almacenar las preferencias del usuario o cargar unas preferencias previamente almacenadas. Interactúa directamente con el usuario desde *solution_draw.html*.
- **checkBoxsConfig.js.** Monta dinámicamente los *CheckBoxes* de selección de instalaciones y se definen las funciones para seleccionarlos y deseleccionarlos.
- **representacion.js.** Este archivo auna todas las operaciones o llamadas relacionadas con la creación de la imagen resultante. Se encuentra modulado para separar las operaciones que afectan directamente a las propias instalaciones, a los flujos, a las adyacencias y a los textos de la imagen.
- **instalaciones.js.** Contiene las funciones encargadas de dibujar las instalaciones en su posición adecuada en el espacio.

- **flujos.js.** Contiene todas las funciones encargadas de representar los flujos en sus distintas formas.
- **adyacencias.js.** Contiene todas las funciones encargadas de representar las adyacencias en sus distintas formas.
- **textos.js.** Contiene las funciones necesarias para dar formato, tamaño y color a los textos.

B.3. Ficheros

Aunque el visualizador de soluciones esté programado completamente desde cero, se encuentra integrado en una aplicación existente en la que se han hecho modificaciones, de modo que, para no dar lugar a confusión, se mostrarán por un lado todos los ficheros nuevos, y por otro, los existentes que hayan sido modificados parcialmente. En este segundo caso, se mostrará solo la parte del fichero que ha sido modificada excepto en las plantillas (.html) y en runner.py, ya que los cambios se encuentran dispersados por todo el código.

B.3.1. Ficheros existentes modificados

settings.py

```
1  """
2  Django settings for distlant project.
3  """
4
5  DATABASES = {
6      'default': {
7          'ENGINE'   : 'django.db.backends.postgresql_psycopg2',
8          'NAME'     : 'DB_DistPlantaNEW',
9          'USER'     : 'postgres',
10         'PASSWORD': 'admin',
11         'HOST'     : 'localhost',
12         'PORT'     : '5432',
13     }
14 }
```

/app/models.py

```
1  """
2  Modelo de datos de la aplicacion
3  """
4
5  from django.db import models
6  from django.contrib.auth.models import User
7
8  class Parametros(models.Model):
9      """
10     Tabla Parametros
11     """
12     Usuario = models.ForeignKey(User)
13     FlujoRepresentacion = models.CharField(max_length=20)
14     FlujoTipo = models.CharField(max_length=20)
15     FlujoColor = models.CharField(max_length=20)
16     AdyRepresentacion = models.CharField(max_length=20)
17     AdyTipo = models.CharField(max_length=20)
18     AspectRatio = models.CharField(max_length=20)
19     EspacioVacio = models.CharField(max_length=20)
20     InstalacionesColor = models.CharField(max_length=10)
21     TextosColor = models.CharField(max_length=10)
22     BordosColor = models.CharField(max_length=10)
```

/app/urls.py

```

1  """
2  Direcccionamiento de la parte app
3  """
4
5  from django.conf.urls import patterns, url
6  from django.contrib import admin
7
8  admin.autodiscover()
9
10 #(?P<username>\w+)/
11
12 urlpatterns = patterns('app',
13     url(r'^$', 'problems.views.problems'),
14     url(r'^problems/$', 'problems.views.problems'),
15     url(r'^problemslist/$', 'problems.views.problemslist'),
16     url(r'^problemshare/$', 'problems.views.problemshare'),
17     url(r'^problems/(?P<problem_id>\d+)/$',
18         'problems.views.problemdetail'),
19
20     url(r'^setup/$', 'setup.views.setup'),
21     url(r'^setuplist/$', 'setup.views.setuplist'),
22     url(r'^setupgroup/$', 'setup.views.setupgroup'),
23     url(r'^setup/(?P<setup_id>\d+)/$',
24         'setup.views.setupdetail'),
25
26     url(r'^executions/$', 'executions.views.executions'),
27     url(r'^executionlist/$', 'executions.views.executionlist'),
28
29     url(r'^solutions/$', 'executions.views.solutions'),
30     url(r'^solutionlist/$', 'executions.views.solutionlist'),
31     url(r'^solutiondraw/$', 'executions.views.solutiondraw'),
32     url(r'^solutionSaveConf/$', 'executions.views.saveconfig'),
33     url(r'^solutionLoadConf/$', 'executions.views.loadconfig'),
34 )

```

/app/executions/views.py

```

1  """
2  Vista de ejecucion y soluciones, donde se muestra las
3  ejecuciones con los estados:
4      - En espera - 0
5      - En ejecucion - 1
6      - Fallido - 2
7      - Terminado - 3
8  """
9
10 import ast
11 import json
12
13 import os
14
15 from django.shortcuts import render_to_response, render
16 from django.http import HttpResponseRedirect, HttpResponse
17 from django.template import RequestContext
18 from django.contrib.auth.decorators import login_required
19 from django.contrib.auth.models import User
20
21 from app.models import Execution, Problem, Setup, Log, Parametros
22
23 import fldata
24 from draw import layoutdrawer
25 from StringIO import StringIO
26
27
28 @login_required
29 def solutionlist(request):
30     """
31     Vista de ejecucion donde se obtienes la lista de ejecuciones
32     si resueltos
33     """
34
35     if request.is_ajax():
36         if request.method == 'POST':
37             # Visualizar solucion
38             if "info" in request.POST:
39                 execut = Execution.objects.filter(user=request.user).get(
40                     id=request.POST.get('info'))
41                 datasolution = fldata.Solutions(structure=ast.literal_eval(
42                     execut.solutions))
43
44                 problem = Problem.objects.filter(owner=request.user).get(
45                     id=execut.problem_id)
46                 dataproblem = fldata.Problem(structure=ast.literal_eval(
47                     problem.structure), problem_name=problem.name)
48
49                 #####
50
51                 #Creacion de archivo en forma de funciones .js con los datos y las
52                 soluciones
53                 strProblema = str(dataproblem).replace("#", "//").replace("(", "[").
54                     replace(")", "]")

```

```

48     strProblema = strProblema.replace('[""', '["/*").replace('""', '/*/')
49     strSolucion = str(datasolution).replace("#", "//").replace("(", "[").
        replace(")", "]").replace("None", "null")
50
51     directory = os.path.dirname(__file__) # Get current directory
52     file_path = os.path.join(directory, '../..', 'static/data/', '
        problema_soluciones.js')
53
54     file = open(file_path, 'w')
55     file.write("function getEnunciado () {\n\n"
56               + "var datos = "
57               + strProblema + ";\n\n"
58               + "return datos; \n}\n\n\n"
59               + "function getSoluciones () {\n\n"
60               + "var soluciones = "
61               + strSolucion + ";\n\n"
62               + "return soluciones; \n}")
63     file.close()
64     #####
65
66     if len(datasolution.solutions) ==0:
67         numsol = 0
68
69     else:
70         numsol = range(len(datasolution.solutions))
71
72     return render_to_response('app/solutions_list.html', {
73         'lsc': execut,
74         'sol': datasolution,
75         'numsol': numsol,
76         'info':1,
77         'prob':dataprobem
78     },RequestContext(request))
79
80     # Eliminar solucion
81     elif "delete" in request.POST:
82         delete_exec = Execution.objects.get(user=request.user,
83             id=request.POST.get('delete'))
84         delete_exec.delete()
85
86     if request.POST.get('selected1') == 'all':
87         if request.POST.get('selected2') == 'all':
88             listexec = Execution.objects.filter( user=request.user,
89                 state=3)
89         else:
90             listalg = Setup.objects.values_list('id', flat=True).filter(
91                 user=request.user,
92                 algorithm_name=request.POST.get('selected2'))
93             listexec = Execution.objects.filter( user=request.user,
94                 setup__in = listalg,
95                 state=3)
96
97
98     else:
99         if request.POST.get('selected2') == 'all':

```

```

100         listexec = Execution.objects.filter( user=request.user,
101                                             problem = request.POST.get('selected1'),
102                                             state=3)
103     else:
104         listalg = Setup.objects.values_list('id', flat=True).filter(
105             user=request.user,
106             algorithm_name=request.POST.get('selected2'))
107         listexec = Execution.objects.filter( user=request.user,
108                                             problem = request.POST.get('selected1'),
109                                             setup__in = listalg,
110                                             state=3)
111
112     return render_to_response('app/solutions_list.html', {'listexec': listexec},
113                             context_instance=RequestContext(request))
114
115     else:
116         return HttpResponseRedirect('/app/solutions/')
117
118 @login_required
119 def saveconfig(request):
120     """
121     Almacenar configuracion de parametros de visualizacion
122     """
123     if request.is_ajax() and request.method == 'POST':
124         if Parametros.objects.filter(Usuario = request.user):
125             edit_parametros = Parametros.objects.get(Usuario = request.user)
126             edit_parametros.FlujoRepresentacion = request.POST.get('
127                 FlujoRepresentacion')
128             edit_parametros.FlujoTipo = request.POST.get('FlujoTipo')
129             edit_parametros.FlujoColor = request.POST.get('FlujoColor')
130             edit_parametros.AdyRepresentacion = request.POST.get('AdyRepresentacion')
131             edit_parametros.AdyTipo = request.POST.get('AdyTipo')
132             edit_parametros.AspectRatio = request.POST.get('AspectRatio')
133             edit_parametros.EspacioVacio = request.POST.get('EspacioVacio')
134             edit_parametros.InstalacionesColor = request.POST.get('InstalacionesColor
135                 ')
136             edit_parametros.TextosColor = request.POST.get('TextosColor')
137             edit_parametros.BordesColor = request.POST.get('BordesColor')
138
139             edit_parametros.save()
140
141             return HttpResponse('')
142
143     else:
144         new_parametros = Parametros.objects.create(
145             Usuario = request.user,
146             FlujoRepresentacion = request.POST.get('FlujoRepresentacion'),
147             FlujoTipo = request.POST.get('FlujoTipo'),
148             FlujoColor = request.POST.get('FlujoColor'),
149             AdyRepresentacion = request.POST.get('AdyRepresentacion'),
150             AdyTipo = request.POST.get('AdyTipo'),
151             AspectRatio = request.POST.get('AspectRatio'),
152             EspacioVacio = request.POST.get('EspacioVacio'),
153             InstalacionesColor = request.POST.get('InstalacionesColor'),
154             TextosColor = request.POST.get('TextosColor'),
155             BordesColor = request.POST.get('BordesColor'),

```

```

154
155         return HttpResponse('')
156
157     else:
158         return HttpResponse('')
159
160 @login_required
161 def loadconfig(request):
162     """
163     Cargar configuracion de parametros de visualizacion
164     """
165     if request.is_ajax() and request.method == 'GET':
166         if Parametros.objects.filter(Usuario = request.user):
167             load_parametros = Parametros.objects.get(Usuario = request.user)
168
169             #####
170
171             #Creacion de archivo en forma de funcion .js con los datos
172             directory = os.path.dirname(__file__) # Get current directory
173             file_path = os.path.join(directory, '../..', 'static/data/', '
174                 parametros_visualizacion.js')
175
176             file = open(file_path, 'w')
177             file.write("function getParametros () {\n\n"
178                 + "var mParametros = {\n"
179                 + "'UsuarioID' : " + str(request.user.id) + ",\n"
180                 + "'FlujoRepresentacion' : '" +
181                     load_parametros.FlujoRepresentacion + "',\n"
182                 + "'FlujoTipo' : '" + load_parametros.FlujoTipo + "',\n"
183                 + "'FlujoColor' : '" + load_parametros.FlujoColor + "',\n"
184                 + "'AdyRepresentacion' : '" +
185                     load_parametros.AdyRepresentacion + "',\n"
186                 + "'AdyTipo' : '" + load_parametros.AdyTipo + "',\n"
187                 + "'AspectRatio' : '" + load_parametros.AspectRatio + "',\n"
188                 + "'EspacioVacio' : '" + load_parametros.EspacioVacio + "',\n"
189                 + "'InstalacionesColor' : '" +
190                     load_parametros.InstalacionesColor + "',\n"
191                 + "'TextosColor' : '" + load_parametros.TextosColor + "',\n"
192                 + "'BordesColor' : '" + load_parametros.BordesColor + "';\n\n"
193                 + "}"
194                 + "return mParametros; \n}")
195             file.close()
196             #####
197
198         return HttpResponse('')
199
200     else:
201         return HttpResponse('')

```

runner.py

```
1  """
2  Runner que mira si existen ejecuciones en espera y la lanza el algoritmo
3  """
4
5  import sys, logging, time
6  import threading, uuid
7  import psycpg2, ast, collections
8  import psycpg2.extras
9  import fldata
10
11 class MySQLHandler(logging.Handler, threading.Thread):
12     """
13     Log handler storing messages on a MySQL database
14     """
15     def __init__(self, handler_id):
16         logging.Handler.__init__(self)
17         threading.Thread.__init__(self)
18         self.setDaemon(True)
19
20         self.handler_id = str(handler_id)
21
22         self.buffer = []
23         self.send_event = threading.Event()
24         self.closing_event = threading.Event()
25         self.start()
26
27
28     def emit(self, record):
29         """
30         Register a log message to be sent
31         """
32         # Note: emit is always inside acquire() release()
33         self.buffer.append(record)
34         self.send_event.set()
35
36
37     def flush(self):
38         """
39         Send event to send messages if stored on buffer
40         """
41         self.send_event.set()
42
43
44     def close(self):
45         """
46         Flushes and chains to the parent class' close().
47         """
48         self.closing_event.set()
49         self.flush()
50         logging.Handler.close(self)
51
52
53     def run(self):
54         """
```

```

55     Thread code to send log messages when communication gets possible
56     """
57     while not self.closing_event.is_set() or self.buffer:
58         time.sleep(5)
59         self.send_event.wait()
60         self.send_event.clear()
61
62         # Copy and format messages safely with handler lock
63         self.acquire()
64         msgs = collections.deque([self.format(record) for record in self.buffer])
65         self.buffer = []
66         self.release()
67
68         # Background message sending
69         while msgs:
70             try:
71                 dbc = psycopg2.connect(host="localhost", port="5432", dbname="
72                     DB_DistPlantaNEW", user="postgres", password="admin")
73                 cursor = dbc.cursor()
74                 msg = msgs[0]
75
76                 sql = """INSERT INTO app_log (execut_id, "message") """
77                 sql += """VALUES({0}, '{1}')


---



```

```
110         if progress >= 0 and progress <= 1:
111             dbc = psycopg2.connect(host="localhost", port="5432", dbname=
112                                     "DB_DistPlantaNEW", user="postgres", password="admin")
113             cursor = dbc.cursor()
114
115             sql = "UPDATE app_execution "
116             sql += "SET perc={0} "
117             sql += "WHERE id={1}".format(progress*100, self.task_id)
118             cursor.execute(sql)
119
120             time_last_progress_report = time.time()
121
122             dbc.commit()
123             cursor.close()
124             dbc.close()
125
126             sended = True
127
128         except:
129             print >> sys.stderr, "Error al insertar progreso en la BBDD"
130             time.sleep(5)
131
132     class Runner():
133         """
134         Lanzador de ejecuciones
135         """
136         def start(self):
137             """
138             Parte principal del lanzador que se encarga:
139             - Conectarse a la BBDD
140             - Hacer peticiones a la BBDD de forma iterativa
141             - Seleccionar la primera ejecucion de la BBDD,sino espera
142             - Cambiar el estado a "En ejecucion"
143             - Seleccionar el problema
144             - Seleccionar la configuracion
145             - Ejecutar el logger y el progressbar en hilos independientes
146             - Lanzar el algoritmo con los datos guardados en cada clase
147             - Guardar la solucion si se ha producido en la BBDD
148             dbc = psycopg2.connect(host="localhost", port="5432", dbname="
149                                     DB_DistPlantaNEW", user="postgres", password="admin")
150             """
151         while True:
152             # Open database connection
153             dbc = psycopg2.connect(host="localhost", port="5432", dbname="
154                                     DB_DistPlantaNEW", user="postgres", password="admin")
155
156             # prepare a cursor object using cursor() method with DictCursor
157             cursor = dbc.cursor(cursor_factory=psycopg2.extras.DictCursor)
158
159             # execute SQL query using execute() method.
160             sql = "SELECT id, user_id, problem_id, setup_id "
161             sql += "FROM app_execution "
162             sql += "WHERE state=0 LIMIT 1 FOR UPDATE"
163             cursor.execute(sql)
```

```

163
164     # Fetch a single row using fetchone() method.
165     execution = cursor.fetchone()
166
167     if execution:
168         # Actualiza el estado de la ejecucion
169         sended = False
170
171         while not sended:
172             try:
173                 # Execute the SQL command and Commit
174                 sql = "UPDATE app_execution "
175                 sql += "SET state=1, perc=0, date_exec=CURRENT_TIMESTAMP "
176                 sql += "WHERE id={0}".format(execution['id'])
177                 cursor.execute(sql)
178
179                 dbc.commit()
180                 sended = True
181
182             except:
183                 print ">> sys.stderr, \"No se ha actualizado la ejecucion\""
184                 # Rollback in case there is any error
185                 dbc.rollback()
186                 time.sleep(5)
187
188         cursor.close()
189         dbc.close()
190
191         # Logging to Database
192         logger = logging.getLogger(str(execution['id']))
193         logger.setLevel(logging.DEBUG)
194         formatter = logging.Formatter('%(asctime)s-->[%%(levelname)s] %(message)s')
195         handler = MySQLHandler(execution['id'])
196         handler.setFormatter(formatter)
197         logger.addHandler(handler)
198
199         # Progress
200         progress_sender = SendProgress(execution['id'])
201         callback = progress_sender.run
202
203         #Consulta el problema de la ejecucion
204         sended = False
205
206         while not sended:
207             try:
208                 dbc = psycopg2.connect(host="localhost", port="5432", dbname=
209                     "DB_DistPlantaNEW", user="postgres", password="admin")
210                 cursor = dbc.cursor(cursor_factory=psycopg2.extras.DictCursor
211                     )
212
213                 sql = "SELECT name, structure "
214                 sql += "FROM app_problem "
215                 sql += "WHERE id={0}".format(execution['problem_id'])
216                 cursor.execute(sql)

```

```
216             problem = cursor.fetchone()
217             cursor.close()
218             dbc.close()
219             sended = True
220
221         except:
222             print >> sys.stderr, "No se ha consultado el problema"
223             time.sleep(5)
224
225     #Consulta la configuracion de la ejecucion
226     sended = False
227
228     while not sended:
229         try:
230             dbc = psycopg2.connect(host="localhost", port="5432", dbname=
231                                     "DB_DistPlantaNEW", user="postgres", password="admin")
232             cursor = dbc.cursor(cursor_factory=psycopg2.extras.DictCursor
233                                 )
234
235             sql = "SELECT algorithm_name, parameters "
236             sql += "FROM app_setup "
237             sql += "WHERE id={0}".format(execution['setup_id'])
238             cursor.execute(sql)
239
240             setup = cursor.fetchone()
241             cursor.close()
242             dbc.close()
243             sended = True
244
245         except:
246             print >> sys.stderr, "No se ha consultado la configuracion"
247             time.sleep(5)
248
249     # Inserta la notificacion del estado de la ejecucion
250     sended = False
251
252     while not sended:
253         try:
254             dbc = psycopg2.connect(host="localhost", port="5432", dbname=
255                                     "DB_DistPlantaNEW", user="postgres", password="admin")
256             cursor = dbc.cursor()
257
258             sql = """INSERT INTO app_notification (user_id, execut_id, "
259                     type", "group") """
260             sql += """VALUES({0}, {1}, 1, '{2}')""".format(execution['
261                     user_id'], execution['id'], uuid.uuid1())
262             cursor.execute(sql)
263
264             dbc.commit()
265             cursor.close()
266             dbc.close()
267             sended = True
268
269         except:
270             print >> sys.stderr, "No se ha consultado la configuracion"
```

```

267         time.sleep(5)
268
269     # Traduccion de los datos al formato fldata
270     data_problem = fldata.Problem(structure=ast.literal_eval(problem['
271                                     structure']),
272                                   problem_name=problem['name']
273                                   )
274
275     data_parameter = fldata.Parameters(structure=(setup['algorithm_name',
276                                                         ], 0,
277                                                         ast.literal_eval(setup['parameters
278                                                         ']), ()))
279
280     solution = None
281
282     try:
283         algorithm = __import__("algorithms." + setup['algorithm_name'],
284                                fromlist = ["*"]).Algorithm()
285         solution = algorithm.run(data_problem, data_parameter, callback,
286                                logger)
287
288     except Exception:
289         logger.exception('Algorithm ' + setup['algorithm_name'] + ' has
290                           failed:')
291
292     # Estado de error si se ha cometido algun error.
293     sended = False
294
295     while not sended:
296         try:
297             dbc = psycopg2.connect(host="localhost", port="5432",
298                                    dbname="DB_DistPlantaNEW", user="postgres", password=
299                                    "admin")
300             cursor = dbc.cursor()
301
302             sql = "UPDATE app_execution "
303             sql += "SET state=2, date_finish=CURRENT_TIMESTAMP "
304             sql += "WHERE id={0}".format(execution['id'])
305             cursor.execute(sql)
306
307             sql2 = """INSERT INTO app_notification (user_id,
308                                                         execut_id, "type", "group") """
309             sql2 += """VALUES({0}, {1}, 2, '{2}'))""".format(execution
310                                                                ['user_id'], execution['id'], uuid.uuid1())
311             cursor.execute(sql2)
312
313             dbc.commit()
314             cursor.close()
315             dbc.close()
316             sended = True
317
318         except:
319             print >> sys.stderr, "No se ha puesto en FALLIDO"
320             time.sleep(5)

```

```
313         finally:
314             logger.info("Terminado el algoritmo")
315             handler.close()
316             handler.join()
317
318     # Sending solution...
319     if solution:
320         logger.info('Sending solution...')
321
322         # Inserta la solucion en la ejecucion
323         sended = False
324
325         while not sended:
326             try:
327                 dbc = psycopg2.connect(host="localhost", port="5432",
328                                         dbname="DB_DistPlantaNEW", user="postgres", password=
329                                             "admin")
330                 cursor = dbc.cursor()
331
332                 txtSolucion = str(solution)
333                 txtID = str(execution['id'])
334
335                 cursor.execute("UPDATE app_execution "+
336                                "SET state=3, perc=100,
337                                    date_finish=CURRENT_TIMESTAMP,
338                                    solutions = %(txtSolucion)s "+
339                                    "WHERE id = %(txtID)s",
340                                {'txtSolucion':txtSolucion, 'txtID':txtID}
341                                )
342
343                 dbc.commit()
344                 cursor.close()
345                 dbc.close()
346                 sended = True
347
348             except:
349                 print >> sys.stderr, "No se ha enviado la solucion "
350                 dbc.rollback()
351                 time.sleep(5)
352
353         # Inserta la notificacion del estado de la ejecucio
354         sended = False
355
356         while not sended:
357             try:
358                 dbc = psycopg2.connect(host="localhost", port="5432",
359                                         dbname="DB_DistPlantaNEW", user="postgres", password=
360                                             "admin")
361                 cursor = dbc.cursor()
362
363                 sql = """INSERT INTO app_notification (user_id,
364                                                            execut_id, "type", "group") """
365                 sql += """VALUES({0}, {1}, 3, '{2}'))""".format(execution[
366                                                                     'user_id'], execution['id'], uuid.uuid1())
367                 cursor.execute(sql)
```

```

361         dbc.commit()
362         cursor.close()
363         dbc.close()
364         send = True
365
366     except:
367         print >> sys.stderr, "No se ha enviado la solucion "
368         dbc.rollback()
369         time.sleep(5)
370
371     else:
372         send = False
373
374     while not send:
375         try:
376             dbc = psycopg2.connect(host="localhost", port="5432",
377                                     dbname="DB_DistPlantaNEW", user="postgres", password=
378                                     "admin")
379             cursor = dbc.cursor()
380
381             sql = "UPDATE app_execution "
382             sql += "SET state=2, date_finish=CURRENT_TIMESTAMP "
383             sql += "WHERE id={0}".format(execution['id'])
384             cursor.execute(sql)
385
386             dbc.commit()
387             cursor.close()
388             dbc.close()
389             send = True
390
391         except:
392             print >> sys.stderr, "No se ha puesto en FALLIDO"
393             dbc.rollback()
394             time.sleep(5)
395
396     else:
397         dbc.commit()
398         cursor.close()
399         dbc.close()
400
401         print "wait"
402         time.sleep(2)
403
404 def main():
405     """
406     Main
407     """
408     runner = Runner()
409     runner.start()
410
411 # If the program is run directly
412 if __name__ == '__main__':
413     sys.exit(main())

```

base.html

```
1  {% load admin_static %}
2  {% load url from future %}
3  {% load i18n %}
4  {% load static %}
5  {% get_static_prefix as STATIC %}
6  {% get_current_language as LANGUAGE_CODE %}
7  {% get_available_languages as LANGUAGES %}
8  {% get_current_language_bidi as LANGUAGE_BIDI %}
9
10 <link rel="icon" type="image/png" href="{% STATIC %}/img/favicon.png" />
11
12 <!DOCTYPE html
13     PUBLIC
14     "-//W3C//DTD XHTML 1.0 Transitional//EN"
15     "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
16
17 <html lang="es">
18     <head>
19         <title>
20             {% block title %}
21             {% endblock %}
22         </title>
23
24         <script src="http://code.jquery.com/jquery-1.9.0.js" type="text/javascript">
25             </script>
26         <script src="http://code.jquery.com/ui/1.10.0/jquery-ui.js" type="text/javascript">
27             </script>
28         <script src="{% STATIC %}js/jquery/bootstrap.min.js"></script>
29         <script src="{% STATIC %}js/jquery/gen_validatorv4.js" type="text/javascript">
30             </script>
31         <link href="http://code.jquery.com/ui/1.10.0/themes/base/jquery-ui.css" rel="
32             stylesheet" type="text/css"/>
33
34         <!-- MYCSS -->
35         <link href="{% STATIC %}css/base.css" rel="stylesheet" type="text/css"/>
36         <link href="{% STATIC %}css/style.css" rel="stylesheet" type="text/css"/>
37
38         <!-- BOOTSTRAP -->
39         <link href="{% STATIC %}css/bootstrap/bootstrap-responsive.min.css" rel="
40             stylesheet" type="text/css"/>
41         <link href="{% STATIC %}css/bootstrap/bootstrap.min.css" rel="stylesheet" type="
42             text/css"/>
43         <link href="{% STATIC %}css/bootstrap/dropdown.tip.css" rel="stylesheet" type="
44             text/css"/>
45
46         <!-- JQUERY TEMPLATES -->
47         <script src="http://ajax.aspnetcdn.com/ajax/jquery.templates/beta1/
48             jquery.tmpl.min.js"></script>
49
50         <!-- FIXING CSRF IN DJANGO FOR JQUERY -->
51         <script src="{% STATIC %}js/jquery/jquery_fix_csrf.js"></script>
52
53         <!-- ICONS -->
54         <link rel="stylesheet" href="https://use.fontawesome.com/releases/v5.8.1/css/
```

```

all.css" integrity="sha384-50oBUHEmvpQ+1lW4y57PTFmhCaXpOML5d60M1M7uH2+
nqUivzIebhnd0JK28anvf" crossorigin="anonymous">
47
48 {%block static%}
49 {%endblock%}
50
51 <script Language="JavaScript">
52     $('>.dropdown-toggle').dropdown()
53 </script>
54 </head>
55
56
57 <body>
58     <!-- CABECERA -->
59     <div class="headerLM">
60         {% if user.is_authenticated %}
61             <!-- Menú usuario -->
62             <div class="row" style="padding: 10 20 0 0">
63                 <div class="pull-right">
64                     <div class="navbar" style="margin-top:5px; margin-bottom: 0;" >
65                         <ul class="pull-right">
66                             <li class="dropdown" style="list-style:none; ">
67                                 <button class="profile-button btn btn-info dropdown-toggle"
68                                     data-toggle="dropdown" style="background:#0183AA;">
69                                     <i class="fa fa-user"></i>&nbsp;  
70                                     <span>{{ user.username }}</span>&nbsp;  
71                                     <span class="caret"></span>
72                                 </button>
73                                 <ul id="menu1" role="menu" class="dropdown-menu drop
74                                     dropdown-tip">
75                                     <li><a href="/accounts/profile/">{% trans 'Edit Profile' %}>
76                                         </a></li>
77                                     <li><a href="/accounts/logout/">{% trans "Log out" %}></a> </
78                                         li>
79                                 </ul>
80                             </li>
81                         </ul>
82                     </div>
83                 </div>
84             </div>
85         {% else %}
86             <!-- Botones -->
87             <div class="row" style="padding: 10 20 0 0">
88                 <div class="pull-right">
89                     <div class="navbar menu_button" style="margin-top: -2px;">
90                         <ul class="pull-right">
91                             <li class="dropdown" style="list-style:none">
92                                 <button class="btn btn-info dropdown-toggle" data-toggle="
93                                     dropdown" style="background:#0183AA">
94                                     {% trans "Login" %}
95                                     <span class="caret"></span>
96                                 </button>
97                                 <ul id="menu2" style="min-width:212px" role="menu"
98                                     aria-labelledby="drop5" class="dropdown-menu drop
99                                     dropdown-tip">

```

```

94         <form id="login_form" method="post" action="{% url '
           views.index' %}">
95     {% csrf_token %}
96     <p>
97         <label for="id_username">Email:</label>
98         <input id="id_username" type="text" name="username"
           maxlength="75">
99     </p>
100    <p>
101        <label for="id_password">{% trans "Password" %}</label>
102        <input type="password" name="password" id="id_password">
103    </p>
104
105    <input type="submit" class="btn btn-success" style="
           padding:3px 71px; margin:20 3px 10 5px; background:#19
           C249" value="{% trans 'Login' %}" />
106 </form>
107
108         <div class="divider"></div>
109         <div style="padding: 3px 0px 0px 5px;">
110
111             <p>{% trans "Forgot password" %}? <a href="/accounts/password/
           reset/">{% trans "Reset it" %}</a>
112             </p>
113             <p>{% trans "Not member" %}? <a href="/accounts/register/">{%
           trans "Registration" %}</a>
114             </p>
115         </div>
116     </ul>
117 </li>
118 </ul>
119 <ul class="pull-right">
120     <li class="dropdown" style="list-style:none; margin-right: -19px"
121     >
122         <button id="{{LANGUAGE_CODE}}" style="
           padding-left:30px;background:#0183AA" class="btn btn-info
           dropdown-toggle" data-toggle="dropdown">
123             {% ifequal LANGUAGE_CODE "es" %}
124             {% trans "Spanish" %}
125             {% else %}
126             {% trans "English" %}
127             {% endifequal %}
128         <span class="caret"></span></button>
129         <ul class="dropdown-menu language_ul drop dropdown-tip">
130             {% for lang in LANGUAGES %}
131                 <li>
132                     <a id="{{ lang.0 }}" href="#" onclick="document.setLang({{
           lang.1 }}.submit();return false;">
133                         <form class="form_lang" name="setLang{{ lang.1 }}"
           action="/i18n/setlang/" method="POST">
134                             {% csrf_token %}
135                             <input type="hidden" name="language" value="
           {{ lang.0 }}" />
136                             <span>{{ lang.1 }}</span>
137                         </form>
138                     </a>
139                 </li>

```

```

137
138             {% endfor %}
139         </ul>
140     </li>
141 </ul>
142 </div>
143     </div>
144 </div>
145 {% endif %}
146
147 <!-- Logo -->
148 <div class="row" style="padding: 0 0 10 40">
149     <div style="margin: 0 auto; width:100%; margin-top:-44px;
150         height:104px; align-content:left">
151         
153         <div>
154             <h1 style="padding:44px 0 0 0; color:#ffffff; font:52px
155                 Baskerville; text-shadow:1px 1px 0px rgb(173, 168, 167)">
156                 {% trans "Plant layout" %}
157             </h1>
158         </div>
159     </div>
160 </div> <!-- /CABECERA -->
161
162 <!-- CONTENIDO -->
163 <div class="content">
164     {% block header_content %}
165     {% endblock %}
166
167     {% if not user.is_authenticated %}
168     <div class="block_content" style="background-image: url('/static/img/
169         plano.jpg'); width: 100%; padding-top: 100px; text-align: center;
170         background-repeat: no-repeat; background-size: cover">
171     <div class="container" style="width: 380px; padding: 15px;
172         background-color: #23282d; opacity: 70%">
173     <p style="text-align: justify; font-size: 18px; color: #fff;
174         font-style: italic">Esta herramienta en línea te ayudará,
175         gracias a sus algoritmos y su eficaz visualizador grá
176         fico, a realizar distribuciones en planta de forma rápida
177         y obteniendo los mejores resultados.</p>
178     </div>
179     </div>
180     {%else%}
181     <div class="block_content1">
182         {% block content %}
183         {% endblock %}
184     </div>
185     {%endif%}
186 </div> <!-- /CONTENIDO -->
187
188 <!-- PIE -->
189 <div class="footerLM">

```

```
183         </div> <!-- /PIE -->
184     </body>
185
186 </html>
```


index.html

```

1  {% extends "base.html" %}
2  {% load url from future %}
3  {% load i18n %}
4  {% load static %}
5  {% get_static_prefix as STATIC %}
6
7  {% block title %}
8      {% trans "DP - Index" %}
9  {% endblock %}
10
11 {%block static%}
12     <script src="{% STATIC %}js/comercial.js" type="text/javascript"></script>
13     <link href="{% STATIC %}css/comercial.css" rel="stylesheet" type="text/css"/>
14
15     <style type="text/css">
16         .content .header_content{margin:0 auto; padding: 10px; width:960px;}
17         .content .bg_content {padding-top: 20px;}
18         .content .block_content {margin: 0 auto; margin-bottom:20px}
19     </style>
20 {%endblock%}
21
22 {% block logo %}
23 {% endblock %}
24
25 {% block header_content %}
26 {% endblock %}
27
28 {% block content %}
29 {% endblock %}
30
31 {% block notific %}
32     <div class="div_notify">
33         {% if form.errors %}
34             <div class="notify" id="red">
35
36                 <p> {% trans "Your username and password didn't match. Please try again" %}.
37                 </p>
38             </div>
39         {% endif %}
40     </div>
41 {% endblock %}

```

problems.html

```

1  {% extends "app/base.html" %}
2  {% load url from future %}
3  {% load i18n %}
4  {% load static %}
5  {% load util%}
6  {% get_static_prefix as STATIC %}
7
8  {% block title %}
9      {% trans "Problems" %}
10 {% endblock %}
11
12 {% block static-more %}
13     <script src="{ { STATIC }}js/problems.js" type="text/javascript"></script>
14     <script src="{ { STATIC }}js/jquery/jquery.MultiFile.js" type="text/javascript">
15         </script>
16 {% endblock %}
17
18 {% block menu-app %}
19     <a class="brand menu-active" href="/app/">{% trans "Problems" %}</a>
20     <a class="brand" href="/app/setup/">{% trans "Setups" %}</a>
21     <a class="brand" href="/app/executions/">{% trans "Executions" %}</a>
22     <a class="brand" href="/app/solutions/">{% trans "Solutions" %}</a>
23 {% endblock %}
24
25 {% block content_app %}
26     <!-- MY PROBLEMS -->
27     <div id="content-app" class="content-app" style="overflow: auto;">
28         <div class="title-page" style="height: 34px; padding: 4px 4px 0 0;
29             border-radius: 5px">
30             <div class="name"><h2>{% trans "My Problems" %}</h2></div>
31             <div class="search"><input type="text" id="search_problem" placeholder="
32                 {% trans "Search problem..." %}" /></div>
33         </div>
34         <div id="listproblems" class="btn-group btn-group-vertical content-problem"
35             style="border: 0; background-color: #fff; padding-top: 5px">
36             {% include 'app/problems_list.html' %}
37         </div>
38     </div>
39
40     <!-- PROBLEMS MANAGER -->
41     <div id="info-app" class="info-app">
42         {% include 'app/problem_detail.html' %}
43     </div>
44
45     <div id="dialog-delete" title="{% trans "Delete problem?" %}" style="height: 125px
46         !important;">
47         <p>
48             <span class="ui-icon ui-icon-alert" style="float: left; margin: 0 7px
49                 20px 0;"></span>
50             {% trans "This problem will be permanently deleted and cannot be
51                 recovered. Are you sure?" %}
52         </p>
53         <div class="ui-dialog-buttonpane ui-widget-content ui-helper-clearfix">
54             <div class="ui-dialog-buttonset">

```

```

48         <button id="deleteproblem" role="button" aria-disabled="false"
           onclick="delete_pbl(this.name)"
49         class="ui-button ui-widget ui-state-default ui-corner-all
           ui-button-text-only btn btn-danger coloredred">
50             <span class="ui-button-text">{% trans "Delete" %}</span>
51         </button>
52         <button id="cancel" type="button" style="padding: 0px 1px;margin: 0
           -10px 0 0;" class="ui-button ui-widget ui-state-default
           ui-corner-all ui-button-text-only btn btn-warning coloryellow"
           role="button" aria-disabled="false">
53             <span class="ui-button-text">{% trans "Cancel" %}</span>
54         </button>
55     </div>
56 </div>
57 </div>
58
59 <div id="dialog-link" title="{% trans "Unlink problem?" %}" style="height: 125px
!important;">
60     <p>
61         <span class="ui-icon ui-icon-alert" style="float: left; margin: 0 7px
           20px 0;"></span>
62         {% trans " This problem will be permanently unlink and cannot be
           recovered. Are you sure? " %}
63     </p>
64     <div class="ui-dialog-buttonpane ui-widget-content ui-helper-clearfix">
65         <div class="ui-dialog-buttonset">
66             <button id="unlinkproblem" role="button" aria-disabled="false"
           onclick="unlink_pbl(this.name)"
67             class="ui-button ui-widget ui-state-default ui-corner-all
           ui-button-text-only btn btn-danger coloredred">
68                 <span class="ui-button-text">{% trans "Delete" %}</span>
69             </button>
70             <button id="cancelink" type="button" style="padding: 0px 1px;margin:
           0 -10px 0 0;" class="ui-button ui-widget ui-state-default
           ui-corner-all ui-button-text-only btn btn-warning coloryellow"
           role="button" aria-disabled="false">
71                 <span class="ui-button-text">{% trans "Cancel" %}</span>
72             </button>
73         </div>
74     </div>
75 </div>
76
77 {% if errors %}
78     <div class="div_notify">
79         {% for error in errors %}
80         <div class="notify" id="{error.0}">
81             <p>{{error.1}}</p>
82         </div>
83         {% endfor %}
84     </div>
85 {% endif %}
86
87 {% endblock %}

```

setup.html

```

1  {% extends "app/base.html" %}
2  {% load url from future %}
3  {% load i18n %}
4  {% load static %}
5  {% load util%}
6  {% get_static_prefix as STATIC %}
7
8  {% block title %}
9      {% trans "Setup" %}
10 {% endblock %}
11
12 {%block static-more%}
13     <script src="{{ STATIC }}js/setup.js" type="text/javascript"></script>
14
15 {%endblock%}
16
17 {% block menu-app%}
18     <a class="brand" href="/app/">{% trans "Problems" %}</a>
19     <a class="brand menu-active" href="/app/setup">{% trans "Setups" %}</a>
20     <a class="brand" href="/app/executions/">{% trans "Executions" %}</a>
21     <a class="brand" href="/app/solutions/">{% trans "Solutions" %}</a>
22
23 {% endblock%}
24
25 {% block content_app %}
26
27     <!-- MY SETUPS -->
28     <div id="content-app" class="content-app " style="overflow: auto;">
29         <div class="title-page" style="height:34px; padding: 4px 4px 0 0; border-radius:
30             5px">
31             <div class="name"><h2>{% trans "My Setups" %}</h2></div>
32
33             <div class="search" style="margin-left: 100px;">
34                 <select id="para_algorithm" name="para_algorithm" onchange="filtrar_stp(
35                     this.options[this.selectedIndex].value)" style="margin: 0;">
36                     <option value="todos">{% trans "All" %}</option>
37                     {% for key in listfilter|sort %}
38                     <option value="{{key}}" >{{key}}</option>
39                     {% endfor %}
40                 </select>
41             </div>
42         </div>
43
44         <div id="listsetup" class="content-setup" style="border: 0; background-color: #
45             fff; padding-top: 5px">
46             {% include 'app/setup_list.html' %}
47         </div>
48     </div>
49
50     <!-- CONFIGURATIONS MANAGER -->
51     <div id="info-app" class="info-app">
52         {% include 'app/setup_detail.html' %}
53     </div>

```

```

52
53 <div id="dialog-delete" title="{% trans "Delete setup?" %}" style="height: 125px !
    important;">
54 <input id="setupdelete" name="setupdelete" style="display:none">
55 <p>
56 <span class="ui-icon ui-icon-alert" style="float: left; margin: 0 7px 20px 0;
    "></span>
57 {% trans " This setup will be permanently deleted and cannot be recovered.
    Are you sure? " %}
58 </p>
59 <div class="ui-dialog-buttonpane ui-widget-content ui-helper-clearfix">
60 <div class="ui-dialog-buttonset">
61 <button id="deletesetup" onclick="delete_stp(this.name)" role="button"
    aria-disabled="false"
62 class="ui-button ui-widget ui-state-default ui-corner-all
    ui-button-text-only btn btn-danger colorred" >
63 <span class="ui-button-text">{% trans "Delete" %}</span>
64 </button>
65 <button id="cancel" type="button" style="padding: 0px 1px;margin: 0 -10px 0
    0;" role="button" aria-disabled="false"
66 class="ui-button ui-widget ui-state-default ui-corner-all
    ui-button-text-only btn btn-warning coloryellow" >
67 <span class="ui-button-text">{% trans "Cancel" %}</span>
68 </button>
69 </div>
70 </div>
71 </div>
72
73 <div id="dialog-delete-group" title="{% trans "Delete groups setups?" %}" style="
    height: 125px !important;">
74 <p>
75 <span class="ui-icon ui-icon-alert" style="float: left; margin: 0 7px 20px 0;
    "></span>
76 {% trans " These setups will be permanently deleted and cannot be recovered.
    Are you sure? " %}
77 </p>
78 <div class="ui-dialog-buttonpane ui-widget-content ui-helper-clearfix">
79 <div class="ui-dialog-buttonset">
80 <button id="deletegroup" onclick="delete_groupstp(this.name)" role="button"
    aria-disabled="false"
81 class="ui-button ui-widget ui-state-default ui-corner-all
    ui-button-text-only btn btn-danger colorred" >
82 <span class="ui-button-text">{% trans "Delete" %}</span>
83 </button>
84
85 <button id="cancelgroup" type="button" style="padding: 0px 1px;margin: 0
    -10px 0 0;" role="button" aria-disabled="false"
86 class="ui-button ui-widget ui-state-default ui-corner-all
    ui-button-text-only btn btn-warning coloryellow" >
87 <span class="ui-button-text">{% trans "Cancel" %}</span>
88 </button>
89 </div>
90 </div>
91
92 </div>
93

```

```
94 <div class="div_notify">
95
96     {%if listproblemsharetemp or listproblemtemp%}
97     <div id="draggable" class="listproblem" style="border-radius: 5px">
98         <div class="title-menuproblem">{% trans "Selected problem" %}</div>
99
100         {% if listproblemtemp %}
101         {% for pt in listproblemtemp %}
102             <div style="margin-left: 5px; color:#19677e"><span>- {{ pt.name|leng:19
103                 }}</span></div>
104         {% endfor %}
105         {% endif %}
106
107         <form method="post" action="" enctype="multipart/form-data" style="margin:5px
108             0 0 5px">
109             {% csrf_token %}
110             Clone x <input name="num_clones" style="width:initial" type="text" value="1
111                 " size="3">
112             <button class="btn colorblue btn-info dropdown-toggle temp " title="{%
113                 trans "Run executions" %}"
114                 type="submit" name="addexecution" >
115                 {% trans "Create Execution" %}
116             </button>
117         </form>
118     </div>
119
120     {% endif %}
121
122     {%if errors%}
123     {%for error in errors%}
124         <div class="notify" id="{{error.0}}">
125             <p>{{error.1}}</p>
126         </div>
127     {%endfor %}
128     {%endif %}
129 </div>
130 {% endblock %}
```

executions.html

```

1  {% extends "app/base.html" %}
2  {% load url from future %}
3  {% load i18n %}
4  {% load static %}
5  {% load util%}
6  {% get_static_prefix as STATIC %}
7
8  {% block title %}
9      {% trans "Executions" %}
10 {% endblock %}
11
12 {%block static-more%}
13     <script src="{ { STATIC }}js/execution.js" type="text/javascript"></script>
14 {%endblock%}
15
16 {% block menu-app%}
17     <a class="brand" href="/app/">{% trans "Problems" %}</a>
18     <a class="brand" href="/app/setup/">{% trans "Setups" %}</a>
19     <a class="brand menu-active" href="/app/executions/">{% trans "Executions" %}</a>
20     <a class="brand" href="/app/solutions/">{% trans "Solutions" %}</a>
21 {% endblock%}
22
23 {% block content_app %}
24
25 <div id="content-app" class="content-app content-main" >
26
27     <!-- MY EXECUTIONS TITTLE -->
28     <div class="title-page" style="height:34px; padding: 4px 4px 0 0; border-radius:
29         5px">
30         <div class="name"><h2>{% trans "My Executions" %}</h2></div>
31         <div class="search" style="margin-left: 325px;">
32             <select id="para_algorithm" name="para_algorithm" onchange="refresh_exec()"
33                 style="margin: 0;">
34                 <option value="all"> {% trans "All" %}</option>
35                 {% for key in listgroup|sort %}
36                 <option value="{{key.id}}" >{{key.name}}</option>
37                 {% endfor %}
38             </select>
39         </div>
40     </div>
41
42     <!-- MY EXECUTIONS CONTENT -->
43     <div id="sortable" class="btn-group btn-group-vertical content-execution" style="
44         background-color: #fff;border: 0; padding-top: 5px">
45         {% include 'app/executions_list.html' %}
46     </div>
47 </div>
48
49 <div id="dialog-delete" title="{% trans "Delete execution?" %}" style="height: 125px
50     !important">
51     <p>
52         <span class="ui-icon ui-icon-alert" style="float: left; margin: 0 7px 20px 0;">
53         </span>
54         {% trans " This execution will be permanently deleted and cannot be recovered.

```

```

    Are you sure? " %}
50 </p>
51 <div class="ui-dialog-buttonpane ui-widget-content ui-helper-clearfix">
52   <div class="ui-dialog-buttonset">
53     <button id="deletexec" onclick="delete_exec(this.name)" role="button"
54       aria-disabled="false"
55       class="ui-button ui-widget ui-state-default ui-corner-all
56         ui-button-text-only btn btn-danger colorred" >
57       <span class="ui-button-text">{% trans "Delete"%}</span>
58     </button>
59     <button id="cancel" type="button" style="padding: 0px 1px;margin: 0 -10px 0
60       0;" class="ui-button ui-widget ui-state-default ui-corner-all
61       ui-button-text-only btn btn-warning coloryellow" role="button"
62       aria-disabled="false">
63       <span class="ui-button-text">{% trans "Cancel"%}</span>
64     </button>
65   </div>
66 </div>
67 <div id="dialog-run" title="{% trans "Run execution?"%}" style="height: 104px !
68   important">
69   <p>
70     <span class="ui-icon ui-icon-alert" style="float: left; margin: 0 7px 20px 0;
71       "></span>
72     {% trans " This execution will be run again. Are you sure?. Are you sure? "
73       %}
74   </p>
75   <div class="ui-dialog-buttonpane ui-widget-content ui-helper-clearfix">
76     <div class="ui-dialog-buttonset">
77       <button id="runexec" onclick="run_exec(this.name)" role="button"
78         aria-disabled="false"
79       class="ui-button ui-widget ui-state-default ui-corner-all
80         ui-button-text-only btn btn-danger colorred">
81       <span class="ui-button-text">{% trans "Run again"%}</span>
82     </button>
83     <button id="cancelrun" type="button" style="padding: 0px 1px;margin: 0
84       -10px 0 0;" class="ui-button ui-widget ui-state-default ui-corner-all
85       ui-button-text-only btn btn-warning coloryellow" role="button"
86       aria-disabled="false">
87       <span class="ui-button-text">{% trans "Cancel"%}</span>
88     </button>
89   </div>
90 </div>
91 {% endblock %}

```


solutions.html

```

1  {% extends "app/base.html" %}
2  {% load url from future %}
3  {% load i18n %}
4  {% load static %}
5  {% load util%}
6  {% get_static_prefix as STATIC %}
7
8  {% block title %}
9      {% trans "Solutions" %}
10 {% endblock %}
11
12 {%block static-more%}
13     <script src="{ { STATIC }}js/solution.js" type="text/javascript"></script>
14
15     <!-- D3.js -->
16     <script type="text/javascript" src="{ { STATIC }}js/d3/d3.js"></script>
17
18     <!-- Scripts manejo D3.js -->
19     <script type="text/javascript" src="{ { STATIC }}js/visualizador/textos.js">
20         </script>
21     <script type="text/javascript" src="{ { STATIC }}js/visualizador/adyacencias.js">
22         </script>
23     <script type="text/javascript" src="{ { STATIC }}js/visualizador/flujos.js">
24         </script>
25     <script type="text/javascript" src="{ { STATIC }}js/visualizador/instalaciones.js"
26         ></script>
27     <script type="text/javascript" src="{ { STATIC }}js/visualizador/representacion.js
28         "></script>
29     <script type="text/javascript" src="{ { STATIC }}js/visualizador/parametrosConfig.js"></script>
30     <script type="text/javascript" src="{ { STATIC }}js/visualizador/checkboxsConfig.js"></script>
31     <script type="text/javascript" src="{ { STATIC }}data/parametros_visualizacion.js"
32         ></script>
33     <script type="text/javascript" src="{ { STATIC }}data/problema_soluciones.js">
34         </script>
35     <script type="text/javascript" src="{ { STATIC }}js/visualizador/main.js"></script>
36     >
37
38     <link rel="stylesheet" href="{ { STATIC }}plugins/selectmaster/css/
39         bootstrap-select.css" />
40     <script type="text/javascript" src="{ { STATIC }}plugins/selectmaster/js/
41         bootstrap-select.js"></script>
42
43     <link media="screen" rel="stylesheet" type="text/css" href="{ { STATIC }}plugins/
44         colorpicker/bootstrap-colorpicker.min.css" />
45     <script src="{ { STATIC }}plugins/colorpicker/bootstrap-colorpicker.min.js" type="
46         text/javascript"></script>
47
48     <script src="{ { STATIC }}plugins/dialog/js/bootstrap-dialog.min.js" type="text/
49         javascript"></script>
50
51     <style>
52         .backgroundDots {

```

```

40         background: url("{ STATIC }img/dots.png");
41         background-repeat: repeat;
42     }
43 </style>
44
45 {%endblock%}
46
47 {% block menu-app%}
48     <a class="brand" href="/app/">{% trans "Problems"%}</a>
49     <a class="brand" href="/app/setup/">{% trans "Setups"%}</a>
50     <a class="brand" href="/app/executions/">{% trans "Executions"%}</a>
51     <a class="brand menu-active" href="/app/solutions/">{% trans "Solutions"%}</a>
52 {% endblock%}
53
54 {% block content_app %}
55
56 <div id="content-app" class="content-app content-main">
57
58     <div id="solutionsHead" class="title-page" style="height:34px; padding: 4px 4px 0
59         0; border-radius: 5px">
60
61         <div class="name"><h2 id="headText" style="display: block">{% trans "My Solutions
62             " %}</h2></div>
63
64         <!-- div SEARCH -->
65         <div id="divSearch" class="search" style="margin-left: 325px; display: block">
66             <select id="para_problem" name="para_problem" onchange="refresh_sol()" style="
67                 margin: 0;">
68                 <option value="all">{% trans "All Problems" %}</option>
69                 {% for key in listprob|sort %}
70                 <option value="{{key.id}}" >{{key.name}}</option>
71                 {% endfor %}
72             </select>
73
74             <select id="para_algorithm" name="para_algorithm" onchange="refresh_sol()"
75                 style="margin: 0;">
76                 <option value="all">{% trans "All Algorithms" %}</option>
77                 {% for key in listalg|sort %}
78                 <option value="{{key}}" >{{key}}</option>
79                 {% endfor %}
80             </select>
81         </div>
82
83         <!-- div BUTTONS -->
84         <div id="divButtons" class="search" style="margin-left: 325px; display: none">
85             <button class="btn btn-secondary" type="button" onclick="refresh_sol()">
86                 <i class="fa fa-arrow-left"></i>&nbsp;{% trans "Return" %}
87             </button>
88             <button class="btn btn-secondary" id="btnConfig" type="button" onclick="
89                 javascript:Configuracion()">
90                 <i class="fa fa-cog"></i>&nbsp;{% trans "Configuration" %}
91             </button>
92         </div>
93
94 </div>
95 <div id="soluciones" class="btn-group btn-group-vertical content-execution" style="
96     border: 0; background-color: #fff; padding-top: 5px">

```

```

90         {% include 'app/solutions_list.html' %}
91     </div>
92 </div>
93
94 <div id="dialog-delete" title="{% trans 'Delete execution?' %}" style="height: 125px
    !important">
95     <p>
96         <span class="ui-icon ui-icon-alert" style="float: left; margin: 0 7px 20px 0;">
          </span>
97         {% trans " This execution with solution will be permanently deleted and cannot
            be recovered. Are you sure? " %}
98     </p>
99     <div class="ui-dialog-buttonpane ui-widget-content ui-helper-clearfix">
100         <div class="ui-dialog-buttonset">
101             <button id="deletexec" onclick="delete_sol(this.name)" role="button"
                aria-disabled="false"
102             class="ui-button ui-widget ui-state-default ui-corner-all
                ui-button-text-only btn btn-danger colorred" >
103                 <span class="ui-button-text">{% trans "Delete" %}</span>
104             </button>
105             <button id="cancel" type="button" style="padding: 0px 1px;margin: 0 -10px 0
                0;" class="ui-button ui-widget ui-state-default ui-corner-all
                ui-button-text-only btn btn-warning coloryellow" role="button"
                aria-disabled="false">
106                 <span class="ui-button-text">{% trans "Cancel" %}</span>
107             </button>
108         </div>
109     </div>
110 </div>
111
112 {% endblock %}

```

solutions_list.html

```

1  {% load url from future %}
2  {% load i18n %}
3  {% load static %}
4  {% load util%}
5  {% get_static_prefix as STATIC %}
6
7  <!--Info de la solucion seleccionada-->
8  {% if info%}
9  <div class="div_solution" style="background:#fff">
10
11      <div class="row">
12          <div class="" id="ImgPlanta" style="display:inline-block; margin:10px 200px">
13              {% include 'app/solution_drawLM.html' %}
14          </div>
15
16          <div class="info_solution">
17              <div style="float:left">
18                  <div class="row-exec row-exec-even">
19                      <div class="title-col">{% trans "Estate: "%}</div>
20                      <div>{% trans "Finished"%}</div>
21                  </div>
22                  <div class="row-exec">
23                      <div class="title-col">{% trans "Date of creation: "%}</div>
24                      <div>{{lsc.date_create}}</div>
25                  </div>
26                  <div class="row-exec row-exec-even">
27                      <div class="title-col">{% trans "Date of execution: "%}</div>
28                      <div>{{lsc.date_exec}}</div>
29                  </div>
30                  <div class="row-exec">
31                      <div class="title-col">{% trans "Date of ending: "%}</div>
32                      <div>{{lsc.date_finish}}</div>
33                  </div>
34                  <div class="row-exec row-exec-even">
35                      <div class="title-col">{% trans "Time CPU: "%}</div>
36                      <div>{{sol.cpu_time}}</div>
37                  </div>
38                  <div class="row-exec ">
39                      <div class="title-col">{% trans "Problem: "%}</div>
40                      <div class="">{{lsc.problem.name}} </div>
41                  </div>
42                  <div class="row-exec row-exec-even">
43                      <div class="title-col">{% trans "Configuration: "%}</div>
44                      <div class="">{{lsc.setup.name}}</div>
45                  </div>
46                  <div class="row-exec ">
47                      <div class="title-col">{% trans "Server: "%}</div>
48                      <div class="">{{sol.server}}</div>
49
50                  <script>
51                      var _usuario = '{{lsc.problem.owner_id}}'
52                  </script>
53              </div>
54          </div>

```

```

55         </div>
56     </div>
57
58     <div style="width: 258px; margin: 30px auto;">
59         {%if not numsol|length %}
60             {%for i in numsol%}
61                 <button class="btn btnnum colorblue" onclick="mainSVG(null, '{{i}}')">
62                     {{forloop.counter}}</button>
63             {%endfor%}
64         {%endif%}
65     </div>
66
67     <!--Listado de soluciones-->
68     {% elif listexec%}
69         {% for le in listexec %}
70             <div class="wid_btn borderbtn btn ui-button ui-button-text-only" style="height:
71                 48px;">
72                 <div class="selectsolut" onclick="mainSVG('{{le.id}}',null)">
73                     <span id="problemname" >Prob: {{le.problem.name|leng:21}}</span>
74                     <span style="display:block; margin-bottom: 7px">Stp: {{le.setup.name|leng:22
75                         }}</span>
76                 </div>
77                 <div onclick="popup_dlt_sol('{{ le.id }}')" class="deleteE" title="{% trans "
78                     Delete" %}"></div>
79             </div>
80         {% endfor %}
81
82     <!--Listado de soluciones vacío-->
83     {% else%}
84         <span style="margin-left: 8px; font-size: 13px; font-style: italic">{% trans "
85             You have no executions"%}</span>
86     {% endif%}

```

B.3.2. Ficheros nuevos

solution_draw.html

```

1  {% load url from future %}
2  {% load i18n %}
3  {% load static %}
4  {% load util%}
5  {% get_static_prefix as STATIC %}
6
7  <div>
8      <div class="modal hide fade" id="modalConfig" style="border-radius: 6px">
9          <div class="modal-header" style="background: #19677e; border-radius: 5px 5px
10             0 0">
11              <button type="button" class="close" data-dismiss="modal" aria-hidden="
12                 true" style="color:#fff; opacity:.8">&times;</button>
13              <p style="color:#fff; font-size: 28px; font-weight: bold; text-align:
14                 center">Configuración</p>
15          </div>
16
17          <div class="modal-body" style="padding-bottom: 5px">
18              <form id="myForm">
19                  <div style="border: 1.8px solid #0F5062; border-radius: 5px;
20                     margin-bottom: 10px">
21                      <p style="font-size: 18px; font-weight: bold">Colores</p>
22                      <p>
23                          <span style="font-size: 12px">Fondo de las instalaciones:
24                              </span>
25                          <input type="color" id="color_instalaciones" value="#F5F5F5"
26                              style="width: 20px">
27
28                          <span style="font-size: 12px; margin-left: 120px">Líneas
29                              delimitantes:</span>
30                          <input type="color" id="color_bordes" value="#696969" style="
31                              width: 20px">
32                      </p>
33                      <p>
34                          <span style="font-size: 12px">Nombres de las instalaciones:
35                              </span>
36                          <input type="color" id="color_textos" value="#2F4F4F" style="
37                              width: 20px">
38
39                          <span style="font-size: 12px; margin-left: 105.5px">Líneas de
40                              flujo:</span>
41                          <input type="color" id="color_flujo" value="#FFA500" style="
42                              width: 20px">
43                      </p>
44                  </div>
45
46                  <div style="border: 1.8px solid #0F5062; border-radius: 5px;
47                     margin-bottom: 10px">
48                      <p style="font-size: 18px; font-weight: bold">Flujo</p>
49                      <p style="font-size: 12px">Calcular el flujo según:
50                          <select id="tipo_flujo">
51                              <option value="DistanciaEuclidea">Distancia Euclídea</
52                                  option>

```

```

39         <option value="DistanciaManhattan">Distancia Manhattan</
        option>
40         <option value="NoMostrar">No mostrar</option>
41     </select>
42 </p>
43
44 <p style="font-size: 12px">Representar el flujo mediante:
45     <select id="representacion_flujo">
46         <option value="GrosorLinea">Grosor de las líneas</option>
47         <option value="ColorLinea">Color de las líneas</option>
48     </select>
49 </p>
50 </div>
51
52 <div style="border: 1.8px solid #0F5062; border-radius: 5px;
    margin-bottom: 10px">
53 <p style="font-size: 18px; font-weight: bold">Adyacencias</p>
54 <p style="font-size: 12px">Representar la adyacencia:
55     <select id="tipo_adyacencia">
56         <option value="LadoComun">Resaltando lados en común</
        option>
57         <option value="LineaUnion">Con una línea de unión</option>
58         <option value="NoMostrar">No mostrar</option>
59     </select>
60 </p>
61
62 <p style="font-size: 12px">Las líneas de adyacencia variarán su:
63     <select id="representacion_adyacencia">
64         <option value="ColorLinea">Color</option>
65         <option value="GrosorLinea">Grosor</option>
66     </select>
67 </p>
68 </div>
69
70 <div style="border: 1.8px solid #0F5062; border-radius: 5px;
    margin-bottom: 25px">
71 <p style="font-size: 18px; font-weight: bold">Otros</p>
72 <p style="font-size: 12px">Representar relación de aspecto:
73     <select id="aspect_ratio">
74         <option value="NoMostrar">No mostrar</option>
75         <option value="1Color">Con un solo color</option>
76         <option value="2Colores">Con distinto color</option>
77     </select>
78 </p>
79
80 <p style="font-size: 12px">Resaltar espacio vacío mediante:
81     <select id="espacio_vacio">
82         <option value="FondoLiso">Fondo liso</option>
83         <option value="FondoEstampado">Fondo estampado</option>
84     </select>
85 </p>
86 </div>
87
88 <div style="text-align: center; margin-bottom: 25px">
89     <button class="btn btn-info" type="button" onclick="

```

```

        javascript:CargarConfiguracion() ">
90         <i class="fa fa-upload"></i>&nbsp;Cargar config.
91     </button>
92     <button class="btn btn-success" type="button" onclick="
        javascript:GuardarConfiguracion() ">
93         <i class="fa fa-save"></i>&nbsp;Guardar config.
94     </button>
95 </div>
96
97 <div id="seleccionDeInstalaciones" style="border: 1.8px solid #0
    F5062; border-radius: 5px; margin-bottom: 25px; padding-bottom: 5
    px">
98 <p style="font-size: 18px; font-weight: bold">Mostrar adyacencias
    y flujos de la instalación:</p>
99 <p>
100     <a href="javascript:MarcarTodo()">Marcar todos</a> |
101     <a href="javascript:DesmarcarTodo()">Desmarcar todos</a>
102 </p>
103 </div>
104 </form>
105 </div>
106
107 <div class="modal-footer" style="background: #19677e; border-top: 0;
    border-radius: 0 0 5px 5px">
108     <button class="btn pull-left" type="button" data-dismiss="modal">
109         <i class="fa fa-times"></i>&nbsp;Cerrar
110     </button>
111     <button class="btn btn-warning" type="button" onclick="
        javascript:setParametrosPorDefecto() ">
112         <i class="fa fa-redo"></i>&nbsp;Reset
113     </button>
114     <button class="btn" type="button" onclick="javascript:setParametros() ">
115         <i class="fa fa-check"></i>&nbsp;Aplicar cambios
116     </button>
117 </div>
118 </div>
119 </div>
120
121 <!-- Dibujo de la solucion -->
122 <div id="render">
123     <script>
124         /***** VARIABLES GLOBALES *****/
125         //Parámetros de representación
126         var _parametros;
127
128         //Datos del enunciado y solución generada
129         //var _plantas;
130         var _instalacionesList;
131         var _flujos;
132         var _adyacencias;
133         //var _problemaNombre;
134         //var _cabeceraEvaluacion;
135         var _instalaciones;
136
137         //Dimensiones del canvas
138         var _svgWidth;

```



```
139     var _svgHeight;
140
141     //Numero de la solución
142     var _solNum;
143
144     //Escala
145     var _svgScale;
146
147     //Mapeo de posición de las instalaciones
148     //entre el enunciado y la solución
149     var _arrMapeo = [];
150
151     //
152     var _configCheckBoxs = 0;
153     /* ***** */
154
155
156     $('document').ready(function(){
157         $("#headText").css('display', 'none');
158         $("#divSearch").css('display', 'none');
159         $("#divButtons").css('display', 'block');
160
161         mainSVG(null,0);
162
163         $(".my-colorpicker2").colorpicker();
164     });
165
166
167     function Configuracion() {
168         $("#modalConfig").modal("show");
169     }
170
171 </script>
172 </div>
```

main.js

```

1  function mainSVG(showList, showSol) {
2
3      // Mostrar listado de soluciones
4      if (showList !== null) {
5          index1 = document.getElementById("para_problem").showList
6          index2 = document.getElementById("para_algorithm").showList
7
8          $('#soluciones').load('/app/solutionlist/',
9                                {'info':showList,
10                                 'selected1':index1,
11                                 'selected2':index2});
12      }
13
14
15      // Mostrar representacion de una solucion
16      else {
17          _solNum = showSol;
18          // Obtencion del enunciado del problema y las soluciones generadas
19          var datos = getEnunciado();
20          var soluciones = getSoluciones();
21
22          // Parametros por defecto
23          _parametros = { 'ColorInstalaciones' : '#F5F5F5',
24                          'ColorTextos' : '#0E1C1C',
25                          'ColorBordes' : '#696969',
26                          'RepresentacionFlujo' : 'GrosorLinea',
27                          'TipoFlujo' : 'DistanciaEuclidea',
28                          'ColorFlujoPorGrosor' : '#FFA500',
29                          'RepresentacionAdyacencia' : 'ColorLinea',
30                          'TipoAdyacencia' : 'LadoComun',
31                          'AspectRatio' : 'NoMostrar',
32                          'EspacioVacio' : 'FondoLiso',
33                      };
34
35
36          // ***** Envoltura de datos *****
37          _instalacionesList = datos[2];
38          _flujos = datos[3]['Flow'];
39          _adyacencias = datos[3]['Proximity'];
40          _instalaciones = soluciones[3];
41          // *****
42
43          // Anchura, altura y proporcion de la planta
44          var plantaWidth = 0;
45          var plantaHeight = 0;
46
47          // Algunas soluciones no respetan las dimensiones del enunciado, por tanto
48          // las
49          // dimensiones de la planta hay que obtenerlas directamente de la solución
50          // generada
51          for(var i=0; i<_instalaciones[_solNum][4].length; i++) {
52              for(var j=0; j<_instalaciones[_solNum][4][i][3].length; j++) {

```

```

53         if(_instalaciones[_solNum][4][i][3][j][0] > plantaWidth) {
54             plantaWidth = _instalaciones[_solNum][4][i][3][j][0];
55         }
56
57         if(_instalaciones[_solNum][4][i][3][j][1] > plantaHeight) {
58             plantaHeight = _instalaciones[_solNum][4][i][3][j][1];
59         }
60     }
61 }
62
63
64 // Mapeo Problema->Solución de las intalaciones
65 _arrMapeo = [];
66 for(var i=0; i<_instalacionesList.length; i++) {
67     for(var j=0; j<_instalaciones[_solNum][4].length; j++) {
68         if(_instalacionesList[i][0] == _instalaciones[_solNum][4][j][0]) {
69             var mapeo = {
70                 p1: i, // Posición de la inst. en las tablas (flujo, ady)
71                 name: _instalacionesList[i][0], // Nombre de la instalación
72                 p2: j // Posición de la inst. en la solución generada
73             };
74             _arrMapeo.push(mapeo);
75         }
76     }
77 }
78
79
80 // Mapeo Solución->Problema de las intalaciones
81 _arrMapeoInv = [];
82 for(var i=0; i<_instalaciones[_solNum][4].length; i++) {
83     for(var j=0; j<_instalacionesList.length; j++) {
84         if(_instalaciones[_solNum][4][i][0] == _instalacionesList[j][0]) {
85             var mapeo = {
86                 p1: i, // Posición de la inst. en las tablas (flujo, ady)
87                 name: _instalaciones[_solNum][4][i][0], // Nombre de la
                        instalación
88                 p2: j // Posición de la inst. en la solución generada
89             };
90             _arrMapeoInv.push(mapeo);
91         }
92     }
93 }
94
95
96 // Se toma como referencia el lado mas grande
97 if(plantaWidth >= plantaHeight) {
98
99     var proporcion = plantaWidth / plantaHeight;
100
101     // Anchura y altura del SVG
102     _svgWidth = 500;
103     _svgHeight = _svgWidth / proporcion;
104
105 } else {
106
107     var proporcion = plantaHeight / plantaWidth;

```

```
108
109         // Anchura y altura del SVG
110         _svgHeight = 500;
111         _svgWidth = _svgHeight / proporcion;
112     }
113
114
115     // Creacion de la escala
116     _svgScale = d3.scaleLinear()
117         .domain([0, plantaWidth]) // Dominio de entrada
118         .range([0, _svgWidth]); // Rango de salida
119
120
121     // Configurar checkboxes
122     if(_configCheckBoxes == 0) {
123         configurarCheckBoxes();
124     }
125
126
127     // Si se habia creado un SVG con anterioridad lo eliminamos
128     d3.select("svg").remove();
129     var svg = crearSvg(_svgWidth, _svgHeight);
130     representar(svg, _instalaciones[_solNum][4]);
131 }
132 }
```

checkboxsConfig.js

```
1  // Checkboxes para marcar las instalaciones de las que se quieren conocer sus flujos/
   adjacencias
2  function configurarCheckBoxes() {
3      _configCheckBoxes = 1;
4
5      for(var i=0; i<_instalacionesList.length; i++) {
6          var checkBox = document.createElement("input");
7          checkBox.setAttribute("type", "checkbox");
8          checkBox.setAttribute("id", "checkBox" + i);
9          checkBox.setAttribute("checked", 1);
10
11         var instalacion = document.createElement("p");
12         var texto = document.createTextNode(_instalacionesList[i][0]);
13         instalacion.appendChild(texto);
14         instalacion.appendChild(checkBox);
15
16         document.getElementById("seleccionDeInstalaciones").appendChild(instalacion);
17     }
18 }
19
20 // Marcar/Desmarcar checkboxes
21 function MarcarTodo() {
22     for(var i=0; i<_instalaciones[0][4].length; i++) {
23         document.getElementById("checkBox" + i).checked = 1;
24     }
25 }
26
27 function DesmarcarTodo() {
28     for(var i=0; i<_instalaciones[0][4].length; i++) {
29         document.getElementById("checkBox" + i).checked = 0;
30     }
31 }
```

representacion.js

```
1  function crearSvg(mSvgWidth, height_svg) {
2
3      var svg = d3.select("#render")
4          .append("svg")
5          .attr("width", mSvgWidth)
6          .attr("height", height_svg);
7
8      return svg;
9  }
10
11 function dibujarFondo(svg, mSvgWidth, height_svg) {
12
13     var dato = [1];
14     var marco = svg.selectAll("fondos")
15         .data(dato)
16         .enter()
17         .append("rect");
18
19     marco.attr("x", 0)
20         .attr("y", 0)
21         .attr("width", mSvgWidth)
22         .attr("height", height_svg)
23         //.attr("class", "backgroundDots");
24         .attr("fill", "#D8BFD8"); // Thistle
25 }
26
27
28 function crearMarco(svg, mSvgWidth, height_svg) {
29
30     var dato = [1];
31     var marco = svg.selectAll("marcos")
32         .data(dato)
33         .enter()
34         .append("rect");
35
36     marco.attr("x", 0)
37         .attr("y", 0)
38         .attr("width", mSvgWidth)
39         .attr("height", height_svg)
40         .attr("fill", "Transparent")
41         .attr("stroke", "Black")
42         .attr("stroke-width", "4");
43 }
44
45
46 function representar(svg, mInstalaciones) {
47
48     // Fondo para resaltar espacios vacíos
49     dibujarFondo(svg, _svgWidth, _svgHeight);
50
51     // Representación de las instalaciones
52     representarInstalaciones(svg, mInstalaciones, _svgScale, _arrMapeoInv);
53
54     // Representación de los flujos
```

```
55     if(_flujos != null) {
56         representarFlujos(svg, _flujos, mInstalaciones, _svgScale, _arrMapeo);
57     }
58
59     // Representación de las adyacencias
60     if(_adyacencias != null) {
61         representarAdyacencias(svg, _adyacencias, mInstalaciones, _svgScale,
62                                 _arrMapeo);
63     }
64
65     // Creación y ajuste de los textos
66     darFormatoTextos(svg, mInstalaciones, _svgScale);
67
68     // Creación de un marco
69     crearMarco(svg, _svgWidth, _svgHeight);
70 }
```

instalaciones.js

```

1  function representarInstalaciones(svg, mInstalaciones, mSvgScale, mArrMapeo) {
2
3      // Relación de aspecto
4      if(_parametros['AspectRatio'] == '1Color' || _parametros['AspectRatio'] == '2
        Colores') {
5          var instalacionDraw;
6          var positivo, negativo;
7
8          // Lados del cuadrado inscrito
9          var sup, izq, dch, inf;
10         var i_2;
11         var base, altura, RA;
12
13         for(var i=0; i<mInstalaciones.length; i++) {
14             instalacionDraw = [];
15             positivo = negativo = 0;
16             sup = izq = dch = inf = 0;
17             base = altura = RA = 0;
18
19             i_2 = mArrMapeo[i].p2;
20
21             for(var j=0; j<mInstalaciones[i][3].length; j++) {
22                 instalacionDraw[j] = {"x":mInstalaciones[i][3][j][0], "y":
                    mInstalaciones[i][3][j][1]};
23             }
24
25             // 4 lados
26             if(instalacionDraw.length == 4) {
27                 // Inicializamos
28                 izq = instalacionDraw[0]['x'];
29                 dch = instalacionDraw[0]['x'];
30                 sup = instalacionDraw[0]['y'];
31                 inf = instalacionDraw[0]['y'];
32
33                 // Obtenemos el cuadrado inscrito
34                 for(var k=0; k<instalacionDraw.length; k++) {
35                     if (instalacionDraw[k]['x'] < izq) {izq = instalacionDraw[k]['x'];}
36                     if (instalacionDraw[k]['x'] > dch) {dch = instalacionDraw[k]['x'];}
37                     if (instalacionDraw[k]['y'] < sup) {sup = instalacionDraw[k]['y'];}
38                     if (instalacionDraw[k]['y'] > inf) {inf = instalacionDraw[k]['y'];}
39                 }
40
41                 base = dch - izq;
42                 altura = inf - sup;
43
44                 if(base >= altura) {
45                     var min = altura;
46                     var max = base;
47
48                 } else {

```

```

49         var max = altura;
50         var min = base;
51     }
52
53     // Comprobamos área
54     if(_instalacionesList[i_2][2] > (base*altura)) {
55         negativo++;
56
57     } else {
58         positivo++;
59     }
60
61     // Comprobamos lado
62     if(_instalacionesList[i_2][3] > min) {
63         negativo++;
64
65     } else {
66         positivo++;
67     }
68
69     // Comprobamos relacion de aspecto
70     RA = max / min;
71     if(_instalacionesList[i_2][5] <= RA <= _instalacionesList[i_2][6]) {
72         positivo++;
73
74     } else if(RA < _instalacionesList[i_2][4] || RA > _instalacionesList[
75         i_2][7]) {
76         negativo++;
77     }
78
79     var color = _parametros['ColorInstalaciones']
80
81     switch(_parametros['AspectRatio']) {
82         case '2Colores':
83             if(positivo - negativo == 3) {
84                 color = '#3EF131'
85
86             } else if(positivo - negativo == 2) {
87                 color = '#5DE053'
88
89             } else if(positivo - negativo == 1) {
90                 color = '#81D47B'
91
92             } else if(positivo - negativo == 0 && positivo != 0) {
93                 color = '#A8D4A5'
94
95             } else if(positivo - negativo == -1) {
96                 color = '#D57F7F'
97
98             } else if(positivo - negativo == -2) {
99                 color = '#E14A4A'
100
101             } else if(positivo - negativo == -3) {
102                 color = '#E72525'
103             }

```

```
104
105         break;
106
107         case '1Color':
108             if(positivo - negativo == 3) {
109                 color = '#2CE42C'
110
111             } else if(positivo - negativo == 2) {
112                 color = '#3BDF3B'
113
114             } else if(positivo - negativo == 1) {
115                 color = '#50D950'
116
117             } else if(positivo - negativo == 0 && positivo != 0) {
118                 color = '#73D373'
119
120             } else if(positivo - negativo == -1) {
121                 color = '#8FCD8F'
122
123             } else if(positivo - negativo == -2) {
124                 color = '#A8C6A8'
125
126             } else if(positivo - negativo == -3) {
127                 color = '#B7BEB7'
128             }
129
130         break;
131
132         default:
133     }
134
135     svg.selectAll("instalacion_"+i)
136         .data([instalacionDraw])
137         .enter().append("polygon")
138         .attr("points",function(d) {
139             return d.map(function(d) {
140                 return [mSvgScale(d.x),mSvgScale(d.y)].join(",");
141             }).join(" ");
142         })
143         .attr("fill", color)
144         .attr("stroke", _parametros['ColorBordes'])
145         .text("prueba")
146         .attr("stroke-width", "1");
147     }
148 }
149
150 // NoMostar
151 else {
152     var instalacionDraw;
153
154     for(var i=0; i<mInstalaciones.length; i++) {
155         instalacionDraw = [];
156
157         for(var j=0; j<mInstalaciones[i][3].length; j++) {
158             instalacionDraw[j] = {"x":mInstalaciones[i][3][j][0], "y":
159                                     mInstalaciones[i][3][j][1]};
```

```
159         }
160
161         svg.selectAll("instalacion_"+i)
162             .data([instalacionDraw])
163             .enter().append("polygon")
164             .attr("points",function(d) {
165                 return d.map(function(d) {
166                     return [mSvgScale(d.x),mSvgScale(d.y)].join(",");
167                 }).join(" ");
168             })
169             .attr("fill", _parametros['ColorInstalaciones'])
170             .attr("stroke", _parametros['ColorBordes'])
171             .text("prueba")
172             .attr("stroke-width", "1");
173     }
174 }
175 }
```

flujos.js

```

1  function representarFlujos(svg, mFlujos, mInstalaciones, mSvgScale, mArrMapeo) {
2
3      // Creación de escala para el grosor de las líneas
4      var maximo = 0;
5
6      for(var i=0; i<mFlujos.length; i++) {
7          for(var j=0; j<mFlujos[i].length; j++) {
8              if(mFlujos[i][j] > maximo) {
9                  maximo = mFlujos[i][j];
10             }
11         }
12     }
13
14     var grosorScale = d3.scaleLinear()
15         .domain([0, maximo])
16         .range([0, 10]);
17
18
19     // 1er uso: almacenar los puntos que forman cada instalación
20     // 2ndo uso: almacenar temporalmente los valores de cada flujo antes de volcarlos
21     // a la matriz de flujos
22     var aux = new Array();
23
24     // Matriz de flujos: cada línea equivale a un elemento del array anterior
25     var matrizFlujos = new Array();
26
27     // Contador del número de filas totales en la matriz de flujos
28     var numFila = 0;
29
30     var instalacionOrigen = new Array();
31     var instalacionDestino = new Array();
32
33     var puntoOrigen;
34     var puntoDestino;
35
36     // Menu para representar según los parámetros seleccionados
37     switch(_parametros['TipoFlujo']) {
38         // DISTANCIA EUCLÍDEA
39         case 'DistanciaEuclidea':
40
41             for(var i=0; i<mFlujos.length; i++) {
42
43                 for(var j=0; j<mFlujos[i].length; j++) {
44
45                     // Creación de matriz con los puntos de inicio y fin de los
46                     // flujos
47                     if((mFlujos[i][j] != 0) && (document.getElementById("checkBox" +
48                         i).checked == 1)) {
49                         // "i" y "j" apuntan a las posiciones de las intalaciones según
50                         // la definición del problema
51                         // "i_2" y "j_2" apuntan a las posiciones de las intalaciones
52                         // en la solución generada
53                         var i_2, j_2;

```

```

50
51         i_2 = mArrMapeo[i].p2;
52         j_2 = mArrMapeo[j].p2;
53
54         // ORIGEN
55         for(var k=0; k<mInstalaciones[i_2][3].length; k++) {
56             aux[0] = mInstalaciones[i_2][3][k][0];
57             aux[1] = mInstalaciones[i_2][3][k][1];
58
59             instalacionOrigen[k] = aux;
60             aux = [];
61         }
62         puntoOrigen = d3.polygonCentroid(instalacionOrigen);
63         instalacionOrigen = [];
64
65         // DESTINO
66         for(var l=0; l<mInstalaciones[j_2][3].length; l++) {
67             aux[0] = mInstalaciones[j_2][3][l][0];
68             aux[1] = mInstalaciones[j_2][3][l][1];
69
70             instalacionDestino[l] = aux;
71             aux = [];
72         }
73         puntoDestino = d3.polygonCentroid(instalacionDestino);
74         instalacionDestino = [];
75
76
77         // Volcamos datos en la matriz de flujos
78         // x1
79         aux[0] = mSvgScale(puntoOrigen[0]);
80         // x2
81         aux[1] = mSvgScale(puntoOrigen[1]);
82         // x3
83         aux[2] = mSvgScale(puntoDestino[0]);
84         // x4
85         aux[3] = mSvgScale(puntoDestino[1]);
86
87
88         switch(_parametros['RepresentacionFlujo']) {
89
90             case 'GrosorLinea':
91                 // stroke
92                 aux[4] = _parametros['ColorFlujoPorGrosor'];
93                 // stroke-width
94                 aux[5] = grosorScale(mFlujos[i][j]);
95
96                 break;
97
98
99             case 'ColorLinea':
100
101                 // stroke
102                 if( mFlujos[i][j] > ((maximo/3)*2) ) {
103                     aux[4] = "Red";
104
105                     } else if( (mFlujos[i][j] <= ((maximo/3)*2)) && (

```

```

106         mFlujos[i][j] > (maximo/3)) ) {
107             aux[4] = "Orange";
108         } else { aux[4] = "Yellow"; }
109
110         // stroke-width
111         aux[5] = 3;
112         break;
113     }
114
115     // Volcamos todo el contenido del array a una fila de la
116     // matriz y vaciamos aux
117     matrizFlujos[numFila] = aux;
118     aux = [];
119
120     // Incrementamos el contador, para seguir almacenando en las
121     // siguientes filas
122     numFila = numFila + 1;
123 }
124
125 // Representación de distancia Euclídea
126 var rFlujos = svg.selectAll("flujosDistanciaEuclidea")
127     .data(matrizFlujos)
128     .enter()
129     .append("line");
130
131 rFlujos.attr("x1", function(d) { return d[0]; })
132     .attr("y1", function(d) { return d[1]; })
133     .attr("x2", function(d) { return d[2]; })
134     .attr("y2", function(d) { return d[3]; })
135     .attr("stroke", function(d) { return d[4]; })
136     .attr("stroke-width", function(d) { return d[5]; });
137 break;
138
139 // DISTANCIA DE MANHATTAN
140 case 'DistanciaManhattan':
141
142     for(var i=0; i<mFlujos.length; i++) {
143
144         for(var j=0; j<mFlujos[i].length; j++) {
145
146             if((mFlujos[i][j] != 0) && (document.getElementById("checkBox" +
147                 i).checked == 1)) {
148                 // "i" y "j" apuntan a las posiciones de las instalaciones segú
149                 // n la definición del problema
150                 // "i_2" y "j_2" apuntan a las posiciones de las instalaciones
151                 // en la solución generada
152                 var i_2, j_2;
153
154                 i_2 = mArrMapeo[i].p2;
155                 j_2 = mArrMapeo[j].p2;
156
157                 // ORIGEN

```

```

156         for(var k=0; k<mInstalaciones[i_2][3].length; k++) {
157             aux[0] = mInstalaciones[i_2][3][k][0];
158             aux[1] = mInstalaciones[i_2][3][k][1];
159
160             instalacionOrigen[k] = aux;
161             aux = [];
162         }
163         puntoOrigen = d3.polygonCentroid(instalacionOrigen);
164         instalacionOrigen = [];
165
166         // DESTINO
167         for(var l=0; l<mInstalaciones[j_2][3].length; l++) {
168             aux[0] = mInstalaciones[j_2][3][l][0];
169             aux[1] = mInstalaciones[j_2][3][l][1];
170
171             instalacionDestino[l] = aux;
172             aux = [];
173         }
174         puntoDestino = d3.polygonCentroid(instalacionDestino);
175         instalacionDestino = [];
176
177         // x1
178         var x1 = mSvgScale(puntoOrigen[0]);
179         // y1
180         var y1 = mSvgScale(puntoOrigen[1]);
181         // x3
182         var x3 = mSvgScale(puntoDestino[0]);
183         // y3
184         var y3 = mSvgScale(puntoDestino[1]);
185         // x2
186         var x2 = x3;
187         // y2
188         var y2 = y1;
189
190
191         // Puntos de la polilínea
192         aux[0] = x1 + "," + y1 + "," + x2 + "," + y2 + "," + x3 + ","
193             + y3;
194
195         switch(_parametros['RepresentacionFlujo']) {
196
197             case 'GrosorLinea':
198                 // stroke
199                 aux[1] = _parametros['ColorFlujoPorGrosor'];
200                 // stroke-width
201                 aux[2] = grosorScale(mFlujos[i][j]);
202
203                 break;
204
205             case 'ColorLinea':
206                 // stroke
207                 if( mFlujos[i][j] > ((maximo/3)*2) ) {
208                     aux[1] = "Red";
209
210                 } else if( (mFlujos[i][j] <= ((maximo/3)*2)) && (

```

```
211         mFlujos[i][j] > (maximo/3)) ) {
212             aux[1] = "Orange";
213         } else { aux[1] = "Yellow"; }
214
215         // stroke-width
216         aux[2] = 3;
217
218         break;
219     }
220
221     // Volcamos todo el contenido del array a una fila de la
222     // matriz y vaciamos aux
223     matrizFlujos[numFila] = aux;
224     aux = [];
225
226     // Incrementamos el contador, para seguir almacenando en las
227     // siguientes filas
228     numFila = numFila + 1;
229 }
230
231 // Representación de distancia Manhattan
232 var rFlujos = svg.selectAll("flujosDistanciaManhattan")
233     .data(matrizFlujos)
234     .enter()
235     .append("polyline");
236
237 rFlujos.attr("points", function(d) { return d[0]; })
238     .attr("fill", "none")
239     .attr("stroke", function(d) { return d[1]; })
240     .attr("stroke-width", function(d) { return d[2]; });
241
242 break;
243
244 default:
245 }
246 }
247 }
```


adyacencias.js

```

1  function trunc(x, posiciones = 0) {
2      var s = x.toString();
3      var l = s.length;
4      var decimalLength = s.indexOf('.') + 1;
5      var numStr = s.substr(0, decimalLength + posiciones);
6
7      return Number(numStr);
8  }
9
10
11 function representarAdyacencias(svg, mAdyacencias, mInstalaciones, mSvgScale,
    mArrMapeo) {
12
13     // Array para almacenar temporalmente los valores de cada adyacencia antes de
        volcarlos a la matriz de adyacencias
14     var aux = new Array();
15     // Matriz de adyacencias: cada línea equivale a un elemento del array anterior
16     var matrizAdyacencias = new Array();
17     // Contador del número de filas totales en la matriz de adyacencias
18     var numFila = 0;
19
20
21     // Menu para representar según los parámetros seleccionados
22     switch(_parametros['TipoAdyacencia']) {
23
24         // RESALTAR LADO EN COMÚN
25         case 'LadoComun':
26
27             for(var i=0; i<mAdyacencias.length; i++) {
28
29                 for(var j=0; j<mAdyacencias[i].length; j++) {
30
31                     if((mAdyacencias[i][j] != 0)
32                         && (document.getElementById("checkBox" + i).checked == 1)
33                         && (i != j)
34                     )
35                     {
36                         // "i" y "j" apuntan a las posiciones de las instalaciones según
                            la definición del problema
37                         // "i_2" y "j_2" apuntan a las posiciones de las instalaciones
                            en la solución generada
38                         var i_2, j_2;
39
40                         i_2 = mArrMapeo[i].p2;
41                         j_2 = mArrMapeo[j].p2;
42
43                         var long_i, long_j;
44
45                         long_i = mInstalaciones[i_2][3].length-1;
46                         long_j = mInstalaciones[j_2][3].length-1;
47
48
49                         for(var k=0; k<mInstalaciones[i_2][3].length; k++) {
50                             for(var l=0; l<mInstalaciones[j_2][3].length; l++) {

```

```

51
52 // Adyacencia vertical
53 if((trunc(mInstalaciones[i_2][3][k][0], 2) == trunc(
54     mInstalaciones[j_2][3][l][0], 2)) // Vertical
55 ) {
56     if((mInstalaciones[i_2][3][k][1] < mInstalaciones
57         [i_2][3][k+1][1]) // Izquierda
58         && (mInstalaciones[i_2][3][k][1] <
59             mInstalaciones[j_2][3][l][1]) // Limit sup
60         && (mInstalaciones[i_2][3][k+1][1] >
61             mInstalaciones[j_2][3][l+1][1]) ) // Limit
62             inf
63         {
64             // x1
65             aux[0] = mInstalaciones[i_2][3][k][0];
66             // y1
67             aux[1] = Math.max(mInstalaciones[i_2][3][k
68                 ][1], mInstalaciones[j_2][3][l+1][1]);
69             // x2
70             aux[2] = aux[0];
71             // y2
72             aux[3] = Math.min(mInstalaciones[i_2][3][k
73                 +1][1], mInstalaciones[j_2][3][l][1]);
74         }
75
76     if((mInstalaciones[i_2][3][k][1] > mInstalaciones
77         [i_2][3][k+1][1]) // Derecha
78         && (mInstalaciones[i_2][3][k][1] >
79             mInstalaciones[j_2][3][l][1]) // Limit inf
80         && (mInstalaciones[i_2][3][k+1][1] <
81             mInstalaciones[j_2][3][l+1][1]) ) // Limit
82             sup
83         {
84             // x1
85             aux[0] = mInstalaciones[i_2][3][k][0];
86             // y1
87             aux[1] = Math.max(mInstalaciones[i_2][3][k
88                 +1][1], mInstalaciones[j_2][3][l][1]);
89             // x2
90             aux[2] = aux[0];
91             // y2
92             aux[3] = Math.min(mInstalaciones[i_2][3][k
93                 ][1], mInstalaciones[j_2][3][l+1][1]);
94         }
95     }
96 }
97
98 // Adyacencia horizontal
99 if((trunc(mInstalaciones[i_2][3][k][1], 2) == trunc(
100     mInstalaciones[j_2][3][l][1], 2)) ) // Horizontal
101 {
102     if((k == 0) // Superior
103         && (mInstalaciones[i_2][3][k][0] <
104             mInstalaciones[j_2][3][l+1][0]) // Limit
105             izq
106         && (mInstalaciones[i_2][3][long_i][0] >
107             mInstalaciones[j_2][3][l][0]) ) // Limit

```

```

90         dch
91     {
92         // x1
93         aux[0] = Math.max(mInstalaciones[i_2][3][k]
94             ][0], mInstalaciones[j_2][3][l+1][0]);
95         // y1
96         aux[1] = mInstalaciones[i_2][3][k][1];
97         // x2
98         aux[2] = Math.min(mInstalaciones[i_2][3][
99             long_i][0], mInstalaciones[j_2][3][l][0])
100         ;
101         // y2
102         aux[3] = aux[1];
103     }
104
105     if((k < (long_i-1)) // Más de 4 lados
106         && (mInstalaciones[i_2][3][k][0] >
107             mInstalaciones[i_2][3][k+1][0]) //
108             Superior
109         && (mInstalaciones[i_2][3][k+1][0] <
110             mInstalaciones[j_2][3][l+1][0]) // Limit
111             izq
112         && (mInstalaciones[i_2][3][k][0] >
113             mInstalaciones[j_2][3][l][0]) ) // Limit
114             dch
115     {
116         // x1
117         aux[0] = Math.max(mInstalaciones[i_2][3][k
118             +1][0], mInstalaciones[j_2][3][l][0]);
119         // y1
120         aux[1] = mInstalaciones[i_2][3][k][1];
121         // x2
122         aux[2] = Math.min(mInstalaciones[i_2][3][k
123             ][0], mInstalaciones[j_2][3][l+1][0]);
124         // y2
125         aux[3] = aux[1];
126     }
127
128     if((l == 0) // Inferior
129         && (mInstalaciones[i_2][3][k][0] <
130             mInstalaciones[j_2][3][long_j][0]) //
131             Limit izq
132         && (mInstalaciones[i_2][3][k+1][0] >
133             mInstalaciones[j_2][3][l][0]) ) // Limit
134             dch
135     {
136         // x1
137         aux[0] = Math.max(mInstalaciones[i_2][3][k
138             ][0], mInstalaciones[j_2][3][l][0]);
139         // y1
140         aux[1] = mInstalaciones[i_2][3][k][1];
141         // x2
142         aux[2] = Math.min(mInstalaciones[i_2][3][k
143             +1][0], mInstalaciones[j_2][3][long_j
144             ][0]);
145         // y2

```

```

127         aux[3] = aux[1];
128     }
129
130     if((l < (long_j-1)) // Más de 4 lados
131         && (mInstalaciones[i_2][3][k][0] <
132             mInstalaciones[i_2][3][k+1][0]) //
133             Inferior
134         && (mInstalaciones[i_2][3][k][0] <
135             mInstalaciones[j_2][3][l][0]) // Limit izq
136         && (mInstalaciones[i_2][3][k+1][0] >
137             mInstalaciones[j_2][3][l+1][0]) ) // Limit
138             dch
139     {
140         // x1
141         aux[0] = Math.max(mInstalaciones[i_2][3][k
142             ][0], mInstalaciones[j_2][3][l+1][0]);
143         // y1
144         aux[1] = mInstalaciones[i_2][3][k][1];
145         // x2
146         aux[2] = Math.min(mInstalaciones[i_2][3][k
147             +1][0], mInstalaciones[j_2][3][l][0]);
148         // y2
149         aux[3] = aux[1];
150     }
151 }
152
153 //
154 if(aux.length > 0) {
155     switch(_parametros['RepresentacionAdyacencia']) {
156         // DISTINTO COLOR DE LAS LÍNEAS
157         case 'ColorLinea':
158
159             // stroke
160             if(mAdyacencias[i][j] > 0) { aux[4] = "
161                 LawnGreen"; }
162             else { aux[4] = "Red"; }
163
164             // stroke-width
165             aux[5] = 1.5;
166
167             break;
168
169             // DISTINTO GROSOR DE LAS LÍNEAS
170             case 'GrosorLinea':
171
172                 // stroke
173                 aux[4] = _parametros['ColorBordes'];
174
175                 // stroke-width
176                 if(mAdyacencias[i][j] > 0) { aux[5] = 5;
177                     }
178                 else { aux[5] = 3; }
179
180                 break;

```

```

174         }
175
176
177         // Se vuelca todo el contenido del array a una
            fila de la matriz y vaciamos aux
178         matrizAdyacencias[numFila] = aux;
179         aux = [];
180
181         // Se incrementa el contador, para seguir
            almacenando en las siguientes filas
182         numFila++;
183     }
184
185     //
186     if(l == long_j-1) {
187         l++;
188     }
189 }
190
191 //
192 if(k == long_i-1) {
193     k++;
194 }
195 }
196 }
197 }
198 }
199
200 // Representación
201 var rAdyacencias = svg.selectAll("adyacenciasLadoComun")
202     .data(matrizAdyacencias)
203     // .data([...new Set(matrizAdyacencias)])
204     .enter()
205     .append("line");
206
207 rAdyacencias.attr("x1", function(d) { return mSvgScale(d[0]); })
208     .attr("y1", function(d) { return mSvgScale(d[1]); })
209     .attr("x2", function(d) { return mSvgScale(d[2]); })
210     .attr("y2", function(d) { return mSvgScale(d[3]); })
211     .attr("stroke", function(d) { return d[4]; })
212     .attr("stroke-width", function(d) { return d[5]; });
213
214 break;
215
216
217 //UNIR CON UNA LÍNEA COMO EN LA DISTANCIA EUCLÍDEA
218 case 'LineaUnion':
219     var instalacionOrigen = new Array();
220     var instalacionDestino = new Array();
221
222     var puntoOrigen;
223     var puntoDestino;
224
225     for(var i=0; i<mAdyacencias.length; i++) {
226
227         for(var j=0; j<mAdyacencias[i].length; j++) {

```

```
228
229     if((mAdyacencias[i][j] != 0) && (document.getElementById("
230         checkBox" + i).checked == 1)) {
231         // "i" y "j" apuntan a las posiciones de las instalaciones segú
232             n la definición del problema
233         // "i_2" y "j_2" apuntan a las posiciones de las instalaciones
234             en la solución generada
235         var i_2, j_2;
236
237         i_2 = mArrMapeo[i].p2;
238         j_2 = mArrMapeo[j].p2;
239
240         // ORIGEN
241         for(var k=0; k<mInstalaciones[i_2][3].length; k++) {
242             aux[0] = mInstalaciones[i_2][3][k][0];
243             aux[1] = mInstalaciones[i_2][3][k][1];
244
245             instalacionOrigen[k] = aux;
246             aux = [];
247         }
248         puntoOrigen = d3.polygonCentroid(instalacionOrigen);
249         instalacionOrigen = [];
250
251         // DESTINO
252         for(var l=0; l<mInstalaciones[j_2][3].length; l++) {
253             aux[0] = mInstalaciones[j_2][3][l][0];
254             aux[1] = mInstalaciones[j_2][3][l][1];
255
256             instalacionDestino[l] = aux;
257             aux = [];
258         }
259         puntoDestino = d3.polygonCentroid(instalacionDestino);
260         instalacionDestino = [];
261
262         // Volcamos datos en la matriz de flujos *****
263         // x1
264         aux[0] = mSvgScale(puntoOrigen[0]);
265         // x2
266         aux[1] = mSvgScale(puntoOrigen[1]);
267         // x3
268         aux[2] = mSvgScale(puntoDestino[0]);
269         // x4
270         aux[3] = mSvgScale(puntoDestino[1]);
271
272         //var flag = 0;
273         var flag, long_i, long_j;
274
275         flag = 0;
276         long_i = mInstalaciones[i_2][3].length-1;
277         long_j = mInstalaciones[j_2][3].length-1;
278
279         for(var k=0; k<mInstalaciones[i_2][3].length; k++) {
280             for(var l=0; l<mInstalaciones[j_2][3].length; l++) {
```

```

281         // Adyacencia vertical
282         if((trunc(mInstalaciones[i_2][3][k][0], 2) == trunc(
283             mInstalaciones[j_2][3][l][0], 2)) // Vertical
284         ) {
285             if((mInstalaciones[i_2][3][k][1] < mInstalaciones
286                 [i_2][3][k+1][1]) // Izquierda
287                 && (mInstalaciones[i_2][3][k][1] <
288                     mInstalaciones[j_2][3][l][1]) // Limit sup
289                 && (mInstalaciones[i_2][3][k+1][1] >
290                     mInstalaciones[j_2][3][l+1][1]) ) // Limit
291                     inf
292             {
293                 flag = 1;
294             }
295
296             if((mInstalaciones[i_2][3][k][1] > mInstalaciones
297                 [i_2][3][k+1][1]) // Derecha
298                 && (mInstalaciones[i_2][3][k][1] >
299                     mInstalaciones[j_2][3][l][1]) // Limit inf
300                 && (mInstalaciones[i_2][3][k+1][1] <
301                     mInstalaciones[j_2][3][l+1][1]) ) // Limit
302                     sup
303             {
304                 flag = 1;
305             }
306         }
307
308         // Adyacencia horizontal
309         if((trunc(mInstalaciones[i_2][3][k][1], 2) == trunc(
310             mInstalaciones[j_2][3][l][1], 2)) ) // Horizontal
311         {
312             if((k == 0) // Superior
313                 && (mInstalaciones[i_2][3][k][0] <
314                     mInstalaciones[j_2][3][l+1][0]) // Limit
315                     izq
316                 && (mInstalaciones[i_2][3][long_i][0] >
317                     mInstalaciones[j_2][3][l][0]) ) // Limit
318                     dch
319             {
320                 flag = 1;
321             }
322
323             if((k < (long_i-1)) // Más de 4 lados
324                 && (mInstalaciones[i_2][3][k][0] >
325                     mInstalaciones[i_2][3][k+1][0]) //
326                     Superior
327                 && (mInstalaciones[i_2][3][k+1][0] <
328                     mInstalaciones[j_2][3][l+1][0]) // Limit
329                     izq
330                 && (mInstalaciones[i_2][3][k][0] >
331                     mInstalaciones[j_2][3][l][0]) ) // Limit
332                     dch
333             {
334                 flag = 1;
335             }
336         }

```

```

317         if((l == 0) // Inferior
318             && (mInstalaciones[i_2][3][k][0] <
                 mInstalaciones[j_2][3][long_j][0]) //
                 Limit izq
319             && (mInstalaciones[i_2][3][k+1][0] >
                 mInstalaciones[j_2][3][l][0]) ) // Limit
                 dch
320         {
321             flag = 1;
322         }
323
324         if((l < (long_j-1)) // Más de 4 lados
325             && (mInstalaciones[i_2][3][k][0] <
                 mInstalaciones[i_2][3][k+1][0]) //
                 Inferior
326             && (mInstalaciones[i_2][3][k][0] <
                 mInstalaciones[j_2][3][l][0]) // Limit izq
327             && (mInstalaciones[i_2][3][k+1][0] >
                 mInstalaciones[j_2][3][l+1][0]) ) // Limit
                 dch
328         {
329             flag = 1;
330         }
331     }
332
333     //
334     if(l == long_j-1) {
335         l++;
336     }
337 }
338
339 //
340 if(k == long_i-1) {
341     k++;
342 }
343 }
344
345
346 switch(_parametros['RepresentacionAdyacencia']) {
347     // DISTINTO COLOR DE LAS LÍNEAS
348     case 'ColorLinea':
349
350         // stroke
351         // Si es >0 y tienen ALGÚN lado común
352         if( ( (mAdyacencias[i][j] > 0)
353             && (flag == 1)
354             )
355
356             // 0 es <0 y no tienen NINGÚN lado común
357             ||
358             ( (mAdyacencias[i][j] < 0)
359             && (flag == 0)
360             )
361
362             ) { aux[4] = "LawnGreen"; }
363

```

```

364         else { aux[4] = "Red"; }
365
366
367         // stroke-width
368         aux[5] = 3;
369
370         break;
371
372
373         // DISTINTO GROSOR DE LAS LÍNEAS
374         case 'GrosorLinea':
375
376             // stroke
377             aux[4] = _parametros['ColorBordes'];
378
379             // stroke-width
380             // Si es >0 y tienen ALGÚN lado común
381             if( ( mAdyacencias[i][j] > 0)
382                 && (flag == 1)
383             )
384
385                 // 0 es <0 y no tienen NINGÚN lado común
386                 ||
387                 ( mAdyacencias[i][j] < 0)
388                 && (flag == 0)
389             )
390
391                 { aux[5] = 5; }
392
393             else { aux[5] = 3; }
394
395             break;
396         }
397
398         // Se vuelca todo el contenido del array a una fila de la
399         // matriz y vaciamos aux
400         matrizAdyacencias[numFila] = aux;
401         aux = [];
402
403         // Se incrementa el contador, para seguir almacenando en las
404         // siguientes filas
405         numFila++;
406     }
407
408     // Representación
409     var rAdyacencias = svg.selectAll("adyacenciasLineaUnion")
410         .data(matrizAdyacencias)
411         .enter()
412         .append("line");
413
414     rAdyacencias.attr("x1", function(d) { return d[0]; })
415         .attr("y1", function(d) { return d[1]; })
416         .attr("x2", function(d) { return d[2]; })
417         .attr("y2", function(d) { return d[3]; })

```

```
418             .attr("stroke", function(d) { return d[4]; })
419             .attr("stroke-width", function(d) { return d[5]; });
420
421         break;
422     }
423
424 }
```

textos.js

```
1 function darFormatoTextos(svg, mInstalaciones, mSvgScale) {
2     // Representación de los nombres de las instalaciones
3     var nombres = svg.selectAll("nombresInstalaciones")
4         .data(mInstalaciones)
5         .enter()
6         .append("text");
7
8     nombres.text(function(d) { return d[0]; })
9         .attr("x", function(d) { return mSvgScale(d3.polygonCentroid(d[3])[0]); })
10        .attr("y", function(d) { return mSvgScale(d3.polygonCentroid(d[3])[1]); })
11        .attr("text-anchor", "middle") // Centrar horizontalmente
12        .attr("alignment-baseline", "middle") // Centrar verticalmente
13        .attr("font-family", "sans-serif")
14        .attr("fill", _parametros['ColorTextos'])
15        .attr("lengthAdjust", "spacingAndGlyphs")
16        .style("font-size", 25)
17 }
```

