UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH
Escola Superior d'Enginyeries Industrial,
Aeroespacial i Audiovisual de Terrassa

Proyecto de diseño de un modelo BIM para una instalación hospitalaria

Documento:

Memoria

Autor:

Christyan Gustavo Morante Pita

Director:

Jordi Cusidó Roura

Titulación:

Máster Universitario en Ingeniería Industrial

Convocatoria:

Prórroga, 2023

TRABAJO FIN DE ESTUDIOS

RESUMEN

En este proyecto se lleva a cabo el desarrollo de una herramienta que optimiza el proceso de iteración y pruebas de la fase de diseño de un proyecto BIM.

Dada la complejidad que supone el desarrollo de un proyecto BIM al completo, este trabajo tiene como objetivo mejorar la fase de diseño, optimizando y automatizando una tarea iterativa como lo es la búsqueda de la configuración de espacios óptima según los datos de entrada y restricciones que se pretenda establecer.

En concreto, se desarrolla una herramienta programada que optimiza la configuración de espacios, cuyos datos de dimensiones y superficies mínimas vienen predefinidos y son independientes del proyecto.

La herramienta se programa en código Python y toma los datos de entrada de archivos CSV. Los datos de salida se muestran visualmente y también se exportan a un fichero de Microsoft Excel (.xlsx).

Estos datos de salida se tratan mediante el software Dynamo para integrarlo a un modelo BIM trabajado en el programa Revit de diseño de edificaciones

El modelo preliminar obtenido es fruto de un posprocesado que se lleva a cabo por el autor del proyecto según el criterio que este opte por aplicar, de forma que se obtenga como resultado un modelo BIM de base sobre el que desarrollar un proyecto BIM en su totalidad y abarcando las diferentes fases de su ciclo de vida.

El modelo BIM de base que se obtiene debe validarse mediante la comprobación del cumplimiento de los requerimientos de áreas mínimas que se establecen en los datos de entrada.

En este proyecto, además, se ha llevado a cabo el análisis del estado del arte en materia de trabajo colaborativo, especialmente de las funcionalidades y ventajas que ofrecen varias de las opciones que están presentes en el mercado.

ABSTRACT

This Project involves the development of a tool that optimizes the iteration and testing process of the design phase of BIM methodology.

Due to the complexity of a full BIM Project Development, this work aims to improve the design phase by optimizing and automating an iterative task such as the research for an optimal layout based on predefined dimensions and constraints.

Specifically, a programmed tool is developed that optimizes space configuration, whose minimum dimension and surface data are predefined and independent of the project. The tool is programmed in Python code and takes input data from CSV files. The output data is displayed visually and also exported to a Microsoft Excel (.xlsx) file.

These output data are processed using Dynamo software to integrate it into a BIM model worked in Revit software. The preliminary model obtained is the result of post-processing carried out by the project author according to the criteria he chooses to apply, in order to obtain a BIM base model on which to develop a complete BIM project involving the different phases of its life cycle.

The obtained BIM base model must be validated by checking compliance with the minimum area requirements established in the input data. In addition, this project includes an analysis of the state of the art in collaborative work, especially the functionalities and advantages offered by different options available in the market.

ÍNDICE

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

1 INTRODUCCIÓN

1.1 OBJETO

El objeto del presente proyecto es desarrollar una herramienta que optimice las tareas de iteración y pruebas de la fase de diseño de un proyecto de construcción basado en la metodología BIM (Building Information Modeling).

Dada la magnitud de un proyecto desarrollado según la metodología BIM, se pretende trabajar una de las fases de las que engloba un proyecto de estas características. En concreto, se trabajará la fase de diseño, la cual se pretende optimizar mediante una herramienta programada y una serie de automatismos que incluye la interacción entre diferentes softwares.

Mediante la realización de este proyecto, se pretende demostrar las posibilidades de optimización de un proceso de la fase de diseño de la metodología BIM.

Para llevar a cabo el proyecto, se plantea como problema la configuración de espacios del ala de emergencias de un hospital. Los espacios y sus dimensiones vienen dados como datos de entrada.

Como resultado de este trabajo, se obtendrá un modelo BIM preliminar, el cual será fruto de, por una parte, un algoritmo que optimice la configuración de los diferentes espacios de una edificación, y por otra, de la interpretación de los datos de salidas del algoritmo y su transformación en un modelo BIM base.

El modelo BIM resultante deberá, además, estar posprocesado por el autor del proyecto, de forma que se valide la utilidad de la herramienta a desarrollar y demuestre las ventajas que esta supone. Este posprocesado se limitará a la configuración final de espacios y su integración en el modelo.

En el desarrollo del proyecto, también se analizará la futura integración de un proyecto BIM en el uso de herramientas de trabajo colaborativo en la nube, como es el caso de la plataforma Omniverse de tecnología Nvidia.

1.2 ALCANCE

En el alcance de este proyecto se incluye el desarrollo de los siguientes paquetes de trabajo:

- Análisis general de la metodología BIM y las posibilidades de trabajo colaborativo que existen en el mercado, enfocado en la plataforma Omniverse de Nvidia.
- Análisis de métodos de optimización de espacios y transformación de datos en proyectos BIM.
- Desarrollo de herramienta de optimización de espacios e integración en software de trabajo en BIM, que incluye:
    - Código fuente de herramienta de optimización.
    - Archivos base de entrada y salida de la herramienta programada.
    - Archivo de interpretación y traducción de datos optimizados a software de trabajo en metodología BIM.
- Modelo BIM preliminar posprocesado por el autor, incluyendo únicamente configuración definitiva de espacios mediante muros e identificación de los mismos.
- Memoria técnica con verificación de algoritmo y valoración económica del proyecto.

Quedan fuera del alcance de este proyecto los siguientes puntos:

- Estudio o diseño de requerimientos de edificación de complejos hospitalarios, incluyendo lo relativo a normativa específica de aplicación.
- Definición de número de espacios, tipos, dimensiones, áreas mínimas, accesos, rutas de circulación o evacuación.
- Desarrollo arquitectónico, estructural y de instalaciones del modelo BIM preliminar. No se incluirá la ubicación de accesos, definición detallada de muros, suelos, techos, particiones secundarias o de cualquier otro elemento adicional.
- Integración de modelo BIM en plataformas de trabajo colaborativo, incluyendo la plataforma Omniverse de Nvidia.

1.3 REQUISITOS

Los requisitos generales que deben asegurarse en el desarrollo del proyecto son los siguientes:

- El proyecto deberá desarrollarse tomando como datos de entrada el número de espacios a optimizar, el tipo, sus dimensiones de base y área mínima obligatoria.
- Se deberá utilizar software comercial ampliamente utilizado y en favor de la interoperabilidad de la información.
- Los entregables que requieran ejecución deben ser compatibles con equipos computadores comerciales y de una gama media o estándar.

Los requisitos específicos a cumplir son los siguientes:

- Datos de entrada mediante archivo CSV.
- Datos de salida mediante Excel.
- Uso de software para programación: Visual Studio Code.
- Lenguaje de programación: Python.
- Optimizador matemático de preferencia: Gurobi.
- Software de modelado y diseño BIM: Revit y Dynamo.
- Se requiere verificación de salas a nivel de superficies disponibles.

1.4 JUSTIFICACIÓN

La gestión de proyectos constructivos ha evolucionado de forma notoria en los últimos años gracias a la aplicación de nuevas metodologías como lo es el caso del BIM.

Esta metodología contempla la totalidad de las fases de un proyecto de construcción y permite el trabajo sobre un único modelo, el cual va evolucionando con el tiempo gracias al tratamiento de la información y datos que este contiene.

De cara a que un proyecto BIM se desarrolle según lo previsto, es necesario sentar unas bases sólidas, lo que requiere de una alta inversión de tiempo. Especialmente, la fase de diseño supone una gran cantidad de iteraciones debido a cambios de diseño, de criterios o de requerimientos. Cada iteración conlleva una gran cantidad de trabajo, lo que ralentiza en gran medida el proceso de diseño.

El presente proyecto se lleva a cabo con objeto de optimizar el proceso iterativo que supone el diseño de una edificación. En concreto, este trabajo se centra en facilitar la optimización de los espacios que componen una edificación gracias a una herramienta programada, y su transformación en un modelo BIM.

Mediante el presente proyecto, se pone en evidencia las ventajas que supone el aplicar automatismos para la optimización de diseño y la facilidad con la que el resultado de los mismos puede trasladarse a un modelo BIM con la programación adecuada y el flujo de información debido.

2 ESTADO DEL ARTE

Los objetivos del proyecto se centran en la metodología BIM o Building Information Modeling principalmente. Se expone en los siguientes párrafos un análisis del estado del arte de esta metodología.

Se analizará en qué consiste la metodología y porqué resulta tan relevante en el desarrollo de proyectos. Se estudiará cómo se introduce su aplicación en la legislación, los usos y aplicaciones que tiene y los estándares que se utilizan.

Adicionalmente, se analizarán las alternativas de trabajo colaborativo existentes en el mercado, especialmente la plataforma Omniverse del desarrollador Nvidia como parte del análisis que se debe realizar en el presente proyecto, a fin de considerar la futura implementación del modelo BIM resultante en esta plataforma.

2.1 QUÉ ES LA METODOLOGÍA BIM

La metodología BIM consiste en una forma innovadora de abordar y gestionar un proyecto constructivo y que comprende las diferentes fases de su ciclo de vida.

Esta metodología permite el trabajo colaborativo desde la fase de concepción o diseño hasta la gestión y el mantenimiento, pasando por las diferentes fases del proceso constructivo.

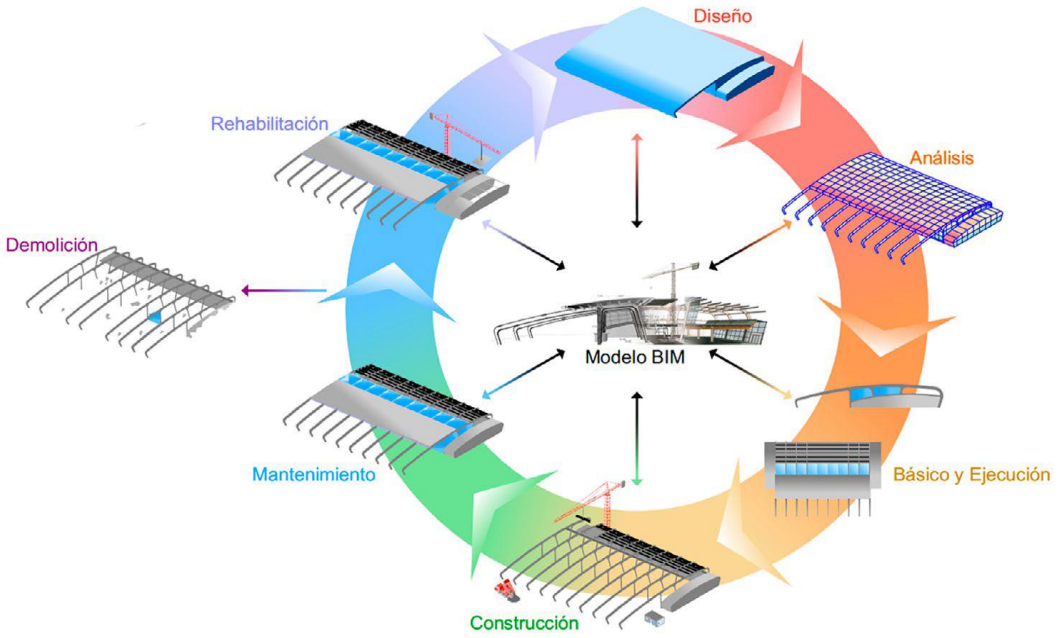

Ilustración 1. Ciclo de vida del Modelo BIM de un proyecto (Fuente: Buildingsmart.es).

El objetivo principal de esta metodología es la centralización de la información del proyecto en un modelo digital de información, el cual se compone de diferentes objetos inteligentes y que se crea desde que empieza el diseño. La información de los objetos es atribuida a través de los diferentes datos que se incrustan en el mismo y que corresponden con el ente que conforman.

El uso de esta metodología a lo largo de todo el ciclo de vida del bien construible supone la evolución de los tradicionales sistemas de diseño, los cuales eran basados en entornos planos o 2D. BIM evoluciona al incorporar información geométrica en 3D, datos de tiempos, costes asociados, datos ambientales e información o datos para el mantenimiento.

El mercado de herramientas software que permiten el trabajo aplicando esta metodología ha evolucionado con el tiempo, ofreciendo cada vez más herramientas y alternativas. Este hecho ha supuesto la necesidad de generación de estándares abiertos para el intercambio de datos entre diferentes agentes, procesos y softwares.

Un ejemplo de ello, y a su vez uno de los más extendidos, sería el estándar IFC (*Industry Foundation Classes*), el cual está regulado por la norma ISO 16739 *Industry Foundation Classes (IFC) for datasharing in the construction and facility management industries.*

Con objeto de impulsar la transformación digital del sector de la industria y la construcción, se han concebido entidades internacionales, las cuales generan diferentes guías, manuales, informes, casos a uso, entre otros informes, para ayudar a una transformación hacia el uso de la metodología BIM basada en estándares abiertos. Un ejemplo de entidad sería Buildingsmart que, además, también ha logrado crear una comunidad abierta que promueve el intercambio de conocimiento en beneficio del sector de la construcción y edificación.

2.2 IMPLANTACIÓN DEL BIM

La evolución que supone esta metodología ha revolucionado la forma de concebir proyectos constructivos de edificación e infraestructuras. Desde la mira institucional, la aplicación de esta metodología supone un facilitador de una política de construcción de edificaciones e infraestructura sostenible, eficiente en gasto público y de alta competencia.

Es por ello que, globalmente, diferentes administraciones públicas han ido requiriendo progresivamente la aplicación de esta metodología al promover nuevos proyectos.

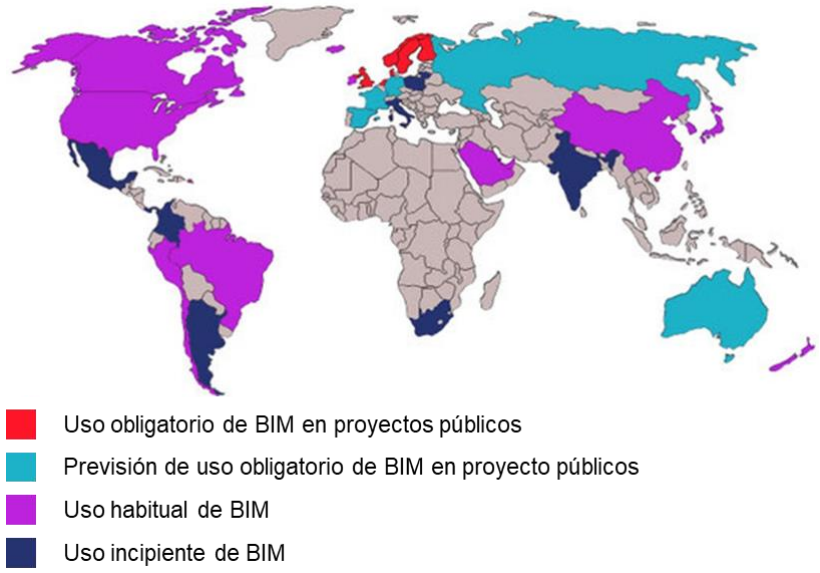

Ilustración 2. Mapa de implantación de BIM (Fuente: Buildingsmart.es).

En el caso de España, el Ministerio de Transportes, Movilidad y Agenta Urbana (MITMA) ostenta la Presidencia y Secretaría de la Comisión Interministerial para la incorporación de la Metodología BIM en la promoción de obras públicas (CBIM), sobre la que se trata en mayor profundidad posteriormente. También contempla la gestión de forma sostenible de los recursos invertidos y la economía circular, así como la innovación digital de la Agenda Urbana, habiendo incluido la metodología BIM en el documento de debate de la Estrategia de Movilidad segura, sostenible y conectada 2030, ya publicada.

Con ello, el MITMA pretende alinear la política de transportes e infraestructuras promovida en España con la Agenda 2030 de Naciones Unidas, o con las recientes Comunicaciones

de la Comisión Europea relacionada con la necesidad de digitalizar la construcción en los países miembros.

2.2.1 MARCO NORMATIVO ESPAÑOL

2.2.1.1 LEY DE CONTRATOS DEL SECTOR PÚBLICO

En el marco normativo español, tras la emisión de la Directiva 2014/24/UE, se introdujo en el ordenamiento español la Ley 9/2017, de 8 de noviembre, de Contratos del Sector Público, por la que se transponen al ordenamiento jurídico español las Directivas del Parlamento Europeo y del Consejo 2014/23/UE y 2014/24/UE, de 26 de febrero de 2014. En la disposición adicional decimoquinta sobre *Normas relativas a los medios de comunicación utilizables en los procedimientos regulados en esta Ley (9/2017)*, se indica lo siguiente en su apartado 6:

"Para contratos públicos de obras, de concesión de obras, de servicios y concursos de proyectos, y en contratos mixtos que combinen elementos de los mismos, los órganos de contratación podrán exigir el uso de herramientas electrónicas específicas, tales como herramientas de modelado digital de la información de la construcción (BIM) o herramientas similares".

Mediante la disposición indicada, las entidades públicas españolas tienen la capacidad de exigir el uso de la metodología BIM en proyectos para contratos, concesiones, licitaciones públicas y cualquier proceso de contratación sujeto a la mencionada Ley. Cabe aclarar que la Ley no exige la aplicación de BIM, sino que habilita a la entidad pública contratante a exigirlo a través de los correspondientes pliegos de prescripciones técnicas o administrativas en los contratos.

Con ello, el objeto de la mencionada Ley es el de impulsar una forma de trabajo eficiente y eficaz en el marco de los proyectos públicos, optimizando recursos y siempre considerando toda la información del ciclo de vida completo de los bienes construidos.

2.2.1.2 COMISIÓN INTERMINISTERIAL PARA LA INCORPORACIÓN DE LA METODOLOGÍA BIM EN LA CONTRATACIÓN PÚBLICA (REAL DECRETO 1515/2018)

Tras la introducción del uso de la metodología BIM en contratos públicos, se publica el Real Decreto 1515/2018, de 28 de diciembre, por el que se crea la Comisión Interministerial para la incorporación de la metodología BIM en la contratación pública, tal y como se ha introducido previamente.

El mencionado Real Decreto, indica que el uso de esta metodología aporta mayor eficacia en la inversión de fondos públicos para el desarrollo de infraestructura, pues implica una mejor gestión documental, mantenimiento a largo plazo de las instalaciones, información a agentes intervinientes en un mismo proyecto, entre otros procesos, todo ello gracias a que el BIM engloba el control y la gestión total de la información del proyecto, desde la fase de diseño, hasta la fase de desmantelamiento, pasando por la construcción y la operación.

El Real Decreto también indica que la Comisión Interministerial tiene un carácter temporal, puesto que su cometido será la implantación de la metodología BIM en la contratación pública. Cumplido este fin, dicha comisión cesará.

2.2.1.3 COMISIÓN BIM

Actualmente, el nombre oficial del órgano es *Comisión BIM* y, según la definición de su página web oficial (cbim.mitma.es) es:

"La Comisión BIM es una Comisión Interministerial, es decir un órgano colegiado, de carácter temporal, cuya finalidad es impulsar y garantizar la coordinación de la

Administración General del Estado (en adelante, AGE), y sus organismos públicos y entidades de derecho público vinculados o dependientes, en la implantación de la metodología BIM en la contratación pública".

La Comisión, de acuerdo a lo dispuesto en el Real Decreto 1515/2018 y según lo indicado en su página web oficial, se compone de:

- Presidente (perteneciente al Ministerio de Transportes, Movilidad y Agenda Urbana)
- Vicepresidente (perteneciente al Ministerio de Hacienda)
- Vocales
    - Cuatro representantes del Ministerio de Transportes, Movilidad y Agenda Urbana
    - Cuatro representantes del Ministerio de Hacienda
    - Dos representantes del Ministerio para la Transición Ecológica y el Reto Demográfico
    - Un representante de los Ministerios de Asuntos Exteriores, Unión Europea y Cooperación, del Interior, de Educación y Formación Profesional, de Industria, Comercio y Turismo, de Economía y Empresa, y de Ciencia, Innovación y Universidades.
- Secretario (perteneciente al Ministerio de Transportes, Movilidad y Agenda Urbana)

Las funciones de esta comisión son:

- Elaborar un Plan de incorporación de la Metodología BIM
- Seguir las medidas contenidas en el Plan
- Realizar acciones de información y formación de personal
- Representar al Reino de España en los distintos foros internacionales
- Recibir e intercambiar información entre los distintos ministerios.

2.2.2 APLICACIONES BIM

Como se ha introducido previamente, BIM consiste en una metodología orientada al avance tecnológico que pretende mejorar el rendimiento y la eficiencia de un proyecto a lo largo de su ciclo de vía.

Esta metodología se orienta al uso de objetos inteligentes, con información no solo geométrica sino también con datos de costes, tiempos, entre otros. Este hecho contrasta con las metodologías de trabajos tradicionales o planas, que se basan en formas 2D sin información adicional.

Las aplicaciones de BIM que más se han desarrollado hoy en día se centran en las fases de diseño y construcción, ya que ha supuesto un gran avance respecto a las formas de trabajo *lineales* y sin información relacionada con las que la industria constructora venía desarrollando sus proyectos.

Entre las diferentes aplicaciones en estas fases, se pueden hallar los siguientes ejemplos:

- **Visualización en 3D de proyectos**

  Los modelos de BIM se basan en objetos inteligentes con formas 3D, que permiten visualizar la forma o estado de la instalación proyectada. Además, este modelo 3D se puede exportar fácilmente a planos 2D, por lo que se trata de una aplicación compatible con los procesos seguidos hasta la actualidad. La visualización 3D permite comprender

el modelo de forma más rápida, intercambiar puntos de vista, realizar inspecciones visuales o simulaciones de forma más intuitiva.

- **Estimación de costes**

  Los objetos inteligentes de los que se compone un modelo BIM pueden relacionar datos económicos, lo que permite centralizar la información en el modelo propio modelo 3D sobre el que se trabaja. Esto facilita el trabajo a la hora de gestionar un presupuesto y permite estimar los costes asociados al proyecto minimizando errores.

- **Análisis de interfaces o colisiones entre elementos**

  Gracias a la visualización 3D o a la capacidad de simulaciones que permite el modelo BIM, se pueden detectar interferencias entre elementos de forma más rápida o automática, minimizando futuros retrabajos o modificaciones del proyecto en fases de ejecución.

- **Visualización de fases constructivas**

  La información contenida en el modelo BIM permite planificar los trabajos a partir del propio modelo. Pudiendo planificar las órdenes de trabajo conforme a lo que se plantee en el proyecto, evaluando temporalidades y visualizando en cada fase el estado de la construcción. Esta aplicación facilita la gestión de equipos, recursos, tiempos y el control del proyecto. A su vez, también se reduce el riesgo de imprevistos en fase de ejecución.

- **Realización de simulaciones**

  Esta aplicación permite obtener datos físicos, estadísticos y de cualquier tipo aplicable a un modelo BIM. En base a los datos introducidos en el modelo y a los modelos matemáticos que se apliquen al desarrollo de simulaciones, se pueden extraer modelos alternativos o estudios de temas específicos en tempranas fases del proyecto, permitiendo tomar decisiones relevantes en fases de diseño minimizando el riesgo de que en posteriores fases se detecten incompatibilidades.

- **Gestión de instalaciones**

  Tras la ejecución de la construcción o instalaciones, con los datos incorporados a los objetos inteligentes del modelo, se puede realizar un seguimiento de todos los puntos de la instalación. Permitiendo simular reparaciones, fallos o mejoras en las instalaciones que faciliten la toma de decisiones. Asimismo, permite tener un histórico de eventos acaecidos en la instalación, que a su vez facilitaría el mantenimiento preventivo, minimizando los fallos y, consecuentemente, recursos necesarios en reparaciones.

De esta última fase, resulta especialmente interesante la capacidad de prever eventos relacionados con el mantenimiento de una instalación, que a su vez hace que la calidad y cantidad de datos incorporados en el modelo BIM sea altamente importante para realizar una gestión adecuada.

Hasta ahora, los gestores de instalaciones o edificios utilizaban softwares de ayuda a la gestión como un *Building Management System* (BMS) o *Computarized Maintenance Management System (CMMS)*, los cuales se basaban en modelos estadísticos y datos históricos para su funcionamiento y su uso uso como apoyo a la gestión de instalaciones.

La información que proporcionaban a los gestores de instalaciones solía encontrarse disociada, modificada o con dificultades para su detección. Esto se debía a que los archivos eran manipulados por diferentes equipos y la información no estaba conectada, lo que impedía en muchos casos poner en coherencia la información real del edificio.

A su vez, la gran cantidad de información requería de un alto esfuerzo de gestión, ya que debían organizarse multitud de informes, archivos CAD, especificaciones, reportes de incidencias, etc. Esto suponía un nivel extra de dificultad al no existir interoperabilidad entre softwares o archivos, lo que muchas veces se traducía en incapacidad de acceder a la información, si más no a la información real y de calidad.

Con el tiempo, han surgido nuevas necesidades en los clientes poseedores de instalaciones, como el mantenimiento predictivo o las necesidades de optimizar el rendimiento energético o de costes de una instalación, que ha llevado a evolucionar la forma en que se gestiona un edificio o instalación.

La integración del BIM en este tipo de sistemas de ayuda a la gestión de instalaciones juega un papel importante ya que permite integrar la información en un único modelo, relacionándola y facilitando su gestión, lo que a su vez facilita el trabajo colaborativo entre los diferentes agentes involucrados en la gestión de un bien.

Las posibilidades que permiten la integración de BIM en la fase de operación y mantenimiento de instalaciones son múltiples, no obstante, se encuentran todavía en fase de desarrollo. Entre las líneas de investigación abiertas, se encuentran:

- Mantenimiento de la información seguro y de garantía
- Control de calidad
- Gestión y seguimiento de activos
- Gestión energética de instalaciones
- Sostenibilidad
- Gestión del espacio
- Gestión de casos de emergencia
- Planificación de mejoras y modernización de instalaciones
- Análisis de edificaciones en tiempo real
- Planificación de mantenimiento preventivo
- Procesos de auditoria

2.2.3 ESTÁNDAR PARA LA TRANSFERENCIA DE INFORMACIÓN

La implementación de BIM en fase de operación y mantenimiento depende de la aplicación que se vaya a realizar, por lo que se requiere definir un orden de desarrollo o nivel de detalle.

Con objeto de facilitar la transmisión, disponibilidad e integridad de la información, se han desarrollado diferentes estándares y especificaciones.

BuildingSMART, una entidad mundial de la industria de la construcción, ha desarrollado un formato estándar para la información: el *Industry Foundation Classes (IFC).* Este estándar es aplicable a la arquitectura, edificación y construcción, y ha sido utilizado en diferentes aplicaciones para el intercambio de información entre un modelo BIM y otros sistemas, como *Computerized Maintenance Management Systems (CMMS).*

El estándar *Construction Operations Building Information Exchange* (COBie), que se trata de un subconjunto del estándar IFC, sirve para el intercambio de información entre la fase de diseño y la fase de operación y mantenimiento, el cual se basa en información contenida en hojas de cálculo donde se definen multitud de datos pertenecientes al modelo BIM de proyecto.

2.2.4 INTEROPERABILIDAD

La diversidad de estándares utilizados en los softwares de apoyo a la gestión de instalaciones (BACnet, Modbus, ZigBee, …), supone una barrera a la integración de BIM en este tipo de sistemas durante la fase de operación y mantenimiento.

Se están llevando a cabo multitud de investigaciones para el desarrollo de estándares o protocolos que integren el intercambio de información con modelos BIM, todas basadas en el uso de librerías abiertas y equipos comerciales, que no impongan la necesidad de desarrollos específicos. Algunos de los desarrollos propuestos son:

- Integrar información visual BIM en un sistema GIS (*Geographic Information System*) usado para la gestión de instalaciones.
- Extracción de información de tareas de mantenimiento durante y tras su ejecución.
- Integración de datos BIM con sistema de radiofrecuencia para evaluar accesos a instalaciones.
- Extracción de datos de un *Building Automation System (BAS)* e incorporarla a un modelo BIM a través de una ruta de información.

Al integrar un modelo BIM con otro sistema deben considerarse las posibles interferencias que se pueden dar en la visualización de datos. Por ello, es importante definir adecuadamente los datos a extraer y su formato, especialmente cuando se incluyen visualizaciones en 3D.

Relacionado con la visualización de un modelo BIM, existen líneas de desarrollo del uso de realidad aumentada (AR). Esta tecnología permite explorar y manipular la información BIM de forma envolvente, mejorando también la accesibilidad y la usabilidad de la información.

Para llevar a cabo una aplicación adecuada en la gestión de instalaciones, se requiere un amplio trabajo en seleccionar la información que se desea visualizar y garantizar la mayor calidad de esta información.

El uso de este tipo de tecnología puede resultar ampliamente beneficiosos en tareas de campo relacionadas con la gestión de instalaciones.

2.3 TRABAJO COLABORATIVO APLICADO AL BIM

El trabajo colaborativo es una parte muy importante de la metodología BIM y se ha convertido en una práctica fundamental en la industria del diseño de edificaciones y construcciones.

La metodología BIM implica la creación y gestión de la información en formato digital de forma trazada y detallada, lo que permite a los integrantes del equipo de proyecto trabajar en conjunto de manera eficiente y colaborativa, de forma que puedan compartir información en tiempo real, resolver problemas en menor tiempo y minimizar los errores de, por ejemplo, incompatibilidades. Todas estas ventajas se acaban viendo reflejadas en el mejor desempeño con el que se desarrolla el proyecto y en el resultado de mayor calidad que se alcanza.

Gracias a las nuevas herramientas que se han venido desarrollando en los últimos años y al uso de plataformas que permiten este tipo de gestión, tales como Omniverse, Autodes BIM360, Trimble Connect, entre otras, la mejora de la eficiencia en el desarrollo de proyectos de construcción gestionados mediante metodología BIM se ha visto potenciada.

2.3.1 PLATAFORMAS DE TRABAJO COLABORATIVO

En este apartado se realiza una explicación general de algunas de las herramientas de trabajo colaborativo que ofrece el mercado y que aplicarían a trabajos desarrollados en BIM.

Posteriormente, se realiza una comparación entre las opciones que ofrece cada una.

2.3.1.1 AUTODESK BIM360

Es una plataforma de gestión de proyectos basada en la nube y que permite a los miembros del equipo de trabajo colaborar en un entorno BIM centralizado. Esta plataforma ofrece herramientas para la gestión de documentos, la coordinación de modelos y sus versiones, la gestión de la calidad y aspectos como la seguridad en el lugar de trabajo.

Entre las ventajas de usar esta herramienta se encuentran:

- Integración: Esta plataforma está diseñada para integrarse con otros productos de la firma Autodesk, tales como Revit, AutoCAD, Navisworks, etc. Esto conlleva la posibilidad de realizar una integración más eficiente y mejor comunicada entre los diferentes miembros del equipo de proyecto.
- Información en la nube: Autodesk BIM360 aloja la información en la nube, de forma que permite a cualquier miembro del equipo acceder a la misma desde cualquier ubicación, requiriendo únicamente conexión a internet. Esto, además, facilita la gestión de la infraestructura tecnológica y las actualizaciones de software, que a su vez conlleva realizar un trabajo más eficiente.
- Interfaz intuitiva: Dispone de una interfaz de usuario intuitiva y fácil de manejar, que facilita a los usuarios su aprendizaje y aplicabilidad en proyectos.
- Gestión de documentos: La plataforma permite la gestión documental al completo, facilitando el control de versiones de archivos, evitando solapamientos, repetición de información innecesaria o pérdida de la misma.

Los inconvenientes que el uso de esta plataforma comportaría son:

- Inversión considerable: Se trata de una solución de pago, que incluye la suscripción de licencias, además de las relativas a los softwares de trabajo (AutoCAD, Revit, etc.) de Autodesk. El monto total puede ascender de forma considerable según el volumen de trabajadores y puede suponer un impedimento para algunos perfiles de empresas.
- Limitada personalización: A pesar de ser una herramienta fácil de utilizar, la personalización es limitada en comparación con otras opciones de mercado por lo que, en caso de necesitar opciones específicas, se requeriría de desarrollos propias o necesariamente el uso de otras herramientas
- Requerimientos de hardware: Además de conexión a internet, el ancho de banda con el que se trabaje debe poder soportar el trabajo de grandes cantidades de información. Además, el equipo con el que se trabaje también debe tener una alta capacidad de procesamiento para permitir el trabajo fluido y eficiente.
- Vinculación del software: Como se ha comentado, esta herramienta está diseñada para trabajar con programas del desarrollador Autodesk, por lo que se limita al uso de estas herramientas únicamente.

Como resumen, Autodesk BIM 360 se trata de una plataforma de trabajo colaborativo potente y fácil de utilizar que ofrece varias ventajas como la gestión documental completa e integración con otros productos del mismo desarrollador. Como aspectos limitativos se

destaca su coste y la vinculación tecnológica con Autodesk que podría suponer una restricción al uso de otro tipo de softwares.

2.3.1.2 TRIMBLE CONNECT

Se trata de una plataforma de colaboración BIM que permite a los usuarios de un equipo de trabajo compartir y colaborar en modelos y datos de proyectos en tiempo real. Incluye también opciones para coordinación de modelos, gestión de la información de proyecto y visualización en 3D.

Las ventajas que se destacan de esta herramienta son las siguientes:

- Integración: esta plataforma está diseñada para trabajar con soluciones de software de tecnología Trimble, como SketchUp, Tekla Structures u otras, permitiendo una integración más eficiente y mejor comunicación entre miembros del proyecto.
- Trabajo en la nube: La información se aloja en la nube, permitiendo el acceso a la misma desde cualquier localización y en cualquier momento a través de internet.
- Visualización de modelos en 3D: Permite la visualización de modelos en 3D en tiempo real y entre diferentes miembros del equipo, e incluso clientes. De esta forma, permite una mejor comprensión del proyecto.
- Gestión de tareas y procesos: Cuenta con herramientas de gestión de tareas y procesos facilitando el seguimiento de hitos y responsabilidades entre miembros del equipo.

Las desventajas de uso son las que se indican a continuación:

- Coste: se trata de una solución de pago, que puede suponer un impedimento según el uso que se pretenda hacer.
- Curva de aprendizaje: La interfaz de usuario puede resultar compleja para algunos usuarios, lo que implicaría una demora en la adopción de esta tecnología.
- Limitaciones en la personalización: La personalización de esta herramienta es limitada en comparación con otras alternativas de mercado.
- Requerimientos hardware: Además de conexión a internet, se precisa de equipos con suficiente capacidad de procesamientos para poder funcionar eficientemente.

2.3.1.3 REVIZTO

Esta plataforma de colaboración BIM permite a los miembros de un equipo de proyecto colaborar en tiempo real en modelos BIM del proyecto, planos u otros documentos. Esta plataforma integra opciones de coordinación de modelos, gestión de errores y problemas y comunicación entre usuarios.

Las ventajas que presenta esta herramienta son:

- Integración con Revit: Esta plataforma está pensada para trabajar con Revit, uno de los softwares de diseño de edificaciones más utilizados. Esta integración con el mencionado programa permite una colaboración más eficiente y con mejor comunicación entre miembros del equipo.
- Visualización 3D: Permite visualizar modelos en 3D en tiempo real entre el equipo de trabajo y clientes, facilitando la comprensión del modelo.
- Comunicación: Cuenta con una herramienta tipo chat o la posibilidad de incluir anotaciones, lo que facilita el flujo comunicativo entre los miembros del equipo de trabajo.

- Gestión documental: Permite la gestión documental completa, lo que mejora la colaboración en la documentación del proyecto.

Las desventajas destacables son las siguientes:

- Coste asociado: Al igual que otras herramientas, esta se trata de una solución de pago, que puede suponer una desventaja según el presupuesto disponible.
- Limitaciones de personalización: Al trabajar para Revit, la personalización es ajustada y no permite muchas opciones de personalización.
- Dependencia de Revit: Dado que está pensada para el trabajo con Revit, no se dispone de alternativas de softwares con los que trabajar.
- Requerimientos hardware: Requiere de una conexión a internet estable y un equipo con alta capacidad de procesamientos para funcionar de forma eficiente.

2.3.1.4 SOLIBRI

Se trata de una herramienta de revisión de modelos BIM que permite a los miembros del equipo de proyecto revisar modelos y detectar errores antes de la construcción. Incluye opciones de coordinación de modelos, detección de errores y gestión de la información.

Las principales ventajas son:

- Verificación de modelado: Permite analizar y verificar las reglas de modelas y diseño según estándares y especificaciones del proyecto.
- Detección de errores: Permite detectar conflictos y fallos de coordinación entre modelos en fase de diseño.
- Integración: Permite integrarse con varios softwares de modelado y herramientas BIM para un flujo de trabajo más eficiente.
- Informes: Proporciona informes detallados y personalizables sobre problemas, reglas de diseño u otra información relevante.

Las desventajas que presente son las siguientes:

- Interfaz poco intuitiva: Cuenta con una interfaz que puede resultar difícil de comprender a usuarios que no están habituados, lo que dificulta el aprendizaje de la herramienta.
- Limitaciones en la edición: No permite la realización de cambios directamente en el modelo BIM.
- Coste asociado: Tiene un precio elevado, lo que puede suponer un impedimento según el presupuesto disponible.
- Requerimientos hardware: Requiere de unas altas prestaciones en los equipos utilizados, lo que afectaría negativamente al presupuesto.

2.3.1.5 OMNIVERSE

Omniverse es una plataforma de colaboración y simulación en tiempo real que permite a equipos de trabajo crear, simular y colaborar en un entorno 3D virtual. Ha sido desarrollada por el tecnólogo Nvidia.

Se trata de una plataforma abierta y conectable que permite a los usuarios trabajar en diferentes aplicaciones como: Autodesk Revit, Autodesk Maya, Blender, Unreal Engine, entre varias otras, en un espacio 3D unificado.

Se permite que los usuarios importen, exporten y manipulen objetos y entorno 3D completos en tiempo real, facilitando la colaboración entre integrantes de equipos de trabajo de diferentes localizaciones y entre diferentes aplicaciones.

Omniverse también cuenta con módulos de simulación en tiempo real que permiten a los usuarios simular comportamientos de objetos en un entorno 3D, facilitando la toma de decisiones en etapas de diseño y producción de productos. Entre las simulaciones que se permiten se encuentran el movimiento, la iluminación, efectos físicos, etc. Al realizarse en tiempo real, permite llevar a cabo ajustes y mejoras en el diseño de forma instantánea.

Esta plataforma también es compatible con la inteligencia artificial, permitiendo a usuarios crear y entrenar modelos de IA en el mismo entorno 3D, simulando el comportamiento de objetos en función de los datos de entrada y la configuración que este tenga. Estas ventajas resultan verdaderamente útiles en etapas de diseño, pero también en fabricación, y es aplicable a multitud de campos como el diseño de edificaciones, la robótica, etc.

Esta plataforma de colaboración Omniverse puede ser una alternativa importante a tener en cuenta al aplicar la metodología BIM ya que sus funcionalidades encajan con los requerimientos que se asocian a dicha metodología. Además, las ventajas que ofrece pueden ser un complemento diferenciador respecto a otras alternativas.

Esta integración se permite gracias a la integración de Revit en Omniverse mediante un "conector" instalado en Revit como una extensión que permite la transferencia de modelos entre el software y la plataforma. Los requisitos para esta integración se analizan en un apartado posterior.

Las ventajas de Omniverse, en cuanto a la metodología BIM, son:

- Colaboración en tiempo real: Permite la colaboración en tiempo real entre diferentes miembros del equipo de trabajo, independientemente de la aplicación software que utilicen. Esto significa que los usuarios pueden trabajar en un entorno 3D unificado y hacer ajustes en tiempo real a través de los diferentes programas con los que accedan a Omniverse.
- Simulación en tiempo real: Las herramientas de Omniverse para llevar a cabo simulaciones en tiempo real permiten a los usuarios simular el comportamiento del modelo y sus diferentes objetos. Pueden realizarse diferentes tipos de pruebas en un único entorno, facilitando la detección de fallos y permitiendo ajustes en tiempo real.
- Integración con otras aplicaciones: Omniverse se integra con multitud de aplicaciones y softwares, lo que permite trabajar a usuarios de diferentes programas trabajar de forma unificada, facilitando la colaboración entre miembros del equipo de proyecto.
- Visualización avanzada: La plataforma ofrece capacidades de visualización avanzada, permitiendo a usuarios crear vistas y renderizaciones de muy alta calidad del modelo, lo que ayuda a mejorar y optimizar la comunicación hacia los clientes.
- Flexibilidad: Al tratarse de una plataforma abierta, Omniverse permite la integración con diferentes herramientas y tecnologías, haciendo al proyecto más flexible, que a su vez implica una mejor eficiencia.

Las desventajas que presenta esta plataforma son las siguientes:

- Requerimientos de hardware: La plataforma y las diferentes herramientas de simulación en tiempo real implican requerimientos de hardware muy especializados, tales como tarjetas gráficas avanzadas, mínimo de la familia RTX de Nvidia, y

procesadores potentes, que repercute directamente en el coste del uso de esa plataforma.

- Curva de aprendizaje: Omniverse y todo su repertorio de herramientas requieren de un proceso de aprendizaje para usuarios que no estén familiarizados con el software, lo que retrasa la adopción de esta tecnología y, consecuentemente, afecta al presupuesto.
- Interoperabilidad limitada: Si bien es cierto que se integra con varias aplicaciones ampliamente utilizadas en el mercado, puede haber limitaciones con algunos programas, lo que puede dificultar la colaboración entre equipos o puede requerir de desarrollos específicos que retrasarían la adopción de la tecnología y afectarían al coste.
- Costes adicionales: La plataforma Omniverse es de pago y puede llevar costes adicionales asociados a la integración entre softwares dependiendo del programa con el que se pretenda trabajar.

2.3.2 COMPARATIVA ENTRE ALTERNATIVAS

La tabla siguiente muestra un cuadro comparativo entre las plataformas y herramientas analizadas previamente:

Tabla 1. Tabla comparativa entre herramientas de trabajo colaborativo.

Herramienta	Visualización 3D	Simulación	Gestión documental	Coordinación de modelos	Comunicación de equipos	Integraciones
Autodesk BIM 360	Sí	No	Sí	Sí	Sí	Revit, Navisworks, AutoCAD, Civil 3D, etc.
Trimble Connect	Sí	No	Sí	Sí	Sí	Sketchup, Tekla Structures, etc.
Rezto	Sí	No	Sí	Sí	Sí	Revit, Navisworks, etc.
Solibri	Sí	No	No	Sí	Sí	Revit.
Omniverse	Sí	Sí	No	Sí	Sí	Revit, Autodesk Maya, 3ds Max, etc.

Como se observa, todas las herramientas de trabajo colaborativo permiten la visualización 3D, coordinación de modelos y comunicación entre equipos, lo que permite a ver los modelos en tiempo real y coordinar las diferentes disciplinas de diseño.

No todas las herramientas ofrecen simulaciones o gestión documental completa. Omniverse es la única que permite opciones simulaciones, pero no cuenta con gestión de la documentación completa. Solibri, a pesar de detectar errores, no se ha considerado que incluya simulaciones ya que se limita a detectar problemas entre coordinación de modelos, pero no cuenta con módulos para simulaciones lumínicas, movimientos, etc.

Todas las herramientas se integran con diferentes softwares. Por una parte, Autodesk BIM 360 y Trimble Connect se focalizan mucho en herramientas de diseño y software para la construcción, mientras que Solibri y Revizto se enfocan más en la coordinación y la detección de fallos.

Omniverse, por su parte, se integra más hacia diseño 3D, aplicable al diseño de edificaciones y trabajo con la metodología BIM.

Como resumen, se puede apreciar que cada herramienta tiene sus fortalezas y debilidades y que la elección de una dependerá de las necesidades específicas del proyecto.

En el caso del presente trabajo, donde se pretendía analizar la aplicabilidad de la plataforma Omniverse a un proyecto BIM, se concluye que esta puede ser integrable al 100% y que las opciones adicionales que permiten pueden suponer un aspecto diferenciador para el proyecto, ya que permitiría agrupar en una sola plataforma el trabajo de diferentes proyectistas de especialidades concretas, trabajando el mismo modelo y centralizando las simulaciones, pruebas, modificaciones y revisiones que se practiquen.

2.3.3 REQUISITOS PARA LA INTEGRACIÓN DE BIM Y OMNIVERSE

Como se ha informado previamente, la plataforma Omniverse es integrable en la metodología BIM por las diferentes ventajas que esta incluye en cuanto a la simulación y tratamiento de modelos BIM.

Esta se integraría en un proyecto BIM a través del uso del software de diseño Revit, gracias a una extensión del programa que permite la transferencia de datos en tiempo real con la plataforma Omniverse.

Una vez subido el modelo BIM a Omniverse, los diferentes miembros del equipo de proyecto pueden acceder al mismo a través de las aplicaciones vinculadas a Omniverse o las que la propia plataforma incluye. Mediante estas, se pueden llevar a cabo pruebas, simulaciones, modificaciones y ajustes que se centralizarían en el modelo BIM único, de forma que todos podrían disponer de la información actualizada en tiempo real.

Las simulaciones y el trabajo en tiempo real, así como las opciones de visualizaciones y renderizaciones que ofrece Omniverse, precisan de unas altas prestaciones de equipos para trabajar de forma fluida y eficiente.

Tras analizar diferentes fuentes de información sobre el uso de Omniverse, se incluyen a continuación algunas de las prestaciones de equipos más importantes y con algunos ejemplos de mercado:

- Procesador: Se requiere procesadores de alta gama, como podrían ser los Intel Core i7 o i9, o AMD Ryzen 7 o 9.
- Tarjeta gráfica: La renderización y visualización óptima que ofrece Omniverse, precisa de tarjetas gráficas de alta gama, recomendándose modelos como la RTX Serie 30 o superior, o una AMD Radeon RX 6000.
- Memoria RAM: Los procesos de trabajo en tiempo real que incluyen simulaciones y transferencia constante de gran cantidad de datos, precisan de, además de una banda ancha de internet, una significativa cantidad de memoria RAM, recomendándose al menos 16 GB de RAM, prefiriéndose disponer de 32 GB o superior.
- Almacenamiento: Se recomienda el uso de unidades de estado sólido (SSD) de alta velocidad para almacenar archivos, puesto que discos duros tradicionales (HDD) pueden ralentizar el funcionamiento y la eficiencia.

- Conexión a internet: Se precisa de una conectividad a internet con banda ancha de alta velocidad para permitir el trabajo fluido y sin interrupciones.

Cabe destacar que los requisitos indicados son orientativos y según información de referencia hallada tras analizar diferentes portales de información.

Además, las prestaciones que se exigen a los equipos pueden variar en función de la exigencia del proyecto. Los datos indicados de forma orientativa, son los exigibles a un proyecto BIM al completo y considerando todo el ciclo de vida del mismo, donde se analiza y trata gran cantidad de información.

Cabe destacar, además, que las prestaciones indicadas son también compatibles con los requerimientos del software Revit para un proyecto BIM completo.

Los equipos de mercado que actualmente cumplirían con las exigencias indicadas rondarían los 3.000 € - 4.000 €, lo que supondría una importante inversión de cara a considerar la solución de Omniverse.

3 METODOLOGÍA

Dada la alta complejidad que supone el desarrollo de un modelo BIM, este proyecto se centra en desarrollar y trabajar aspectos específicos dentro de la fase de diseño.

Como se ha indicado en la introducción del proyecto, este no incluye en su alcance la integración o el trabajo de la plataforma Omniverse ni de cualquier otra plataforma de trabajo colaborativo. Por lo que en la solución que se plantea y se desarrolla en este proyecto no se consideran esas alternativas.

En los apartados siguientes se expone la problemática planteada y la solución técnica que se propone para desarrollar este proyecto.

3.1 PLANTEAMIENTO DE LA SOLUCIÓN

Si bien es cierto que la correcta aplicación de la metodología BIM facilita los cambios sucesivos que se dan en el proceso de diseño y a lo largo de la vida de un proyecto de construcción, es necesario plantear unas bases sólidas y estructuradas en el modelo para la información mantenga un orden y cumpla con las funcionalidades previstas.

Las tomas de decisiones en fases iniciales del proyecto son cruciales para el futuro y es en esta etapa donde más iteraciones y cambios se suelen dar.

Centrando el objetivo en el desarrollo de una edificación, que en el caso específico de este proyecto se trataría de un complejo hospitalario, la definición de la organización de espacios (o *layout*) supone una importante tarea que conlleva un pormenorizado estudio e iteraciones del modelo hasta llegar a la solución definitiva que se apruebe.

Este proceso iterativo de prueba y error en el planteamiento del diseño de *layouts* supone una importante inversión en trabajo que, además, en muchos casos, no resulta apreciable de cara a cliente, el cual espera, generalmente, contar con la información visual, instantánea y ordenada que le permita tomar decisiones.

Sumando las necesidades del cliente, es decir, de disponer de la información completa y ordenada en el momento preciso, y la recurrente carga de trabajo que suponen los procesos iterativos, se hace evidente la necesidad de optimizar este proceso.

3.2 SOLUCIÓN TÉCNICA PLANTEADA

En base a la problemática expuesta, en este proyecto se plantea desarrollar una herramienta capaz de, en base a una información de entrada y a unas condiciones predefinidas, optimice la configuración de *layouts,* dando como información de salida una configuración de espacios en 2D que, a su vez, se transferirá a una herramienta de trabajo BIM, donde se puedan obtener modelos en 3D con la información necesaria para transmitir la información de forma rápida y ordenada, sin perder la trazabilidad de la información.

El proceso a seguir será el siguiente:

1. Definición de un algoritmo capaz de optimizar espacios. El algoritmo debe ser capaz de interpretar las dimensiones de los espacios y las condiciones de afinidad entre ellos. La salida de este algoritmo debe ser la información o configuración de espacios optimizada.

Ilustración 3. Diagrama de proceso - Optimización.

2. Desarrollo de proceso-interfaz entre datos de salida del algoritmo y entorno de trabajo donde se pueda gestionar la información y retrabajar, según proceda, estos datos.

Ilustración 4. Diagrama de proceso – Interpretación de datos y traslado a entorno BIM.

3. Obtención de modelo BIM con la configuración deseada.

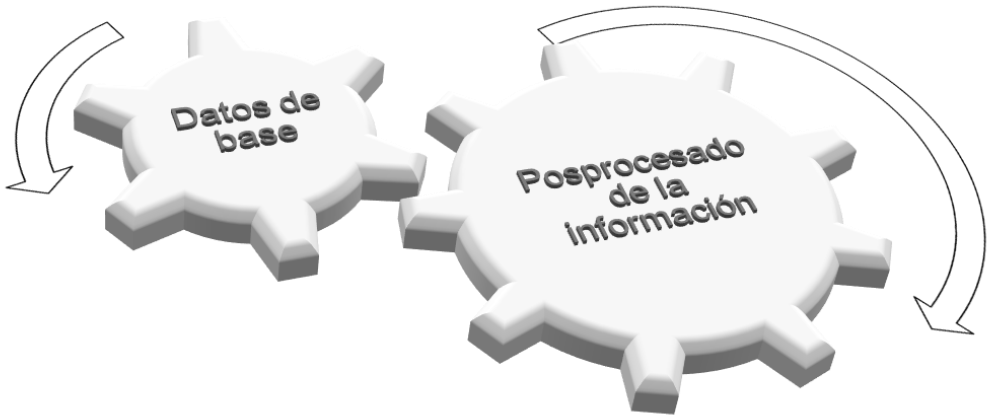

Ilustración 5. Diagrama – Posprocesado de la información.

4 PLANTEAMIENTO Y DECISIÓN SOBRE SOLUCIONES ALTERNATIVAS

Tomando como referencia la solución técnica planteada en el apartado previo, se define el planteamiento a llevar a cabo en la ejecución de este proyecto. Primeramente, se evaluarán las diferentes alternativas disponibles para su desarrollo y, posteriormente, se justificará la solución escogida finalmente.

La solución técnica previa comprende tres bloques principales:

- Algoritmo de optimización
- Interpretación de la información y obtención de modelo BIM preliminar
- Posprocesado de modelo BIM preliminar

Como resulta evidente, el trabajo de interpretación de la información y obtención de datos está muy relacionado con el posprocesado del modelo BIM preliminar que se obtenga, principalmente por el uso de software capaz de integrar esta información.

Si bien es cierto que la metodología BIM permite la interoperabilidad entre softwares de diferentes casas, para el desarrollo de este proyecto se trabajarán ambos puntos a la vez para reducir el número de programas a utilizar.

4.1 ALGORITMO DE OPTIMIZACIÓN

4.1.1 INVESTIGACIÓN TEÓRICA PREVIA

La optimización de espacios ha sido un tema gran interés para la ingeniería desde hace varios años. Resulta interesante para posteriormente optimizar tareas, operaciones, recorridos de evacuación, circulación de personas (trabajadores, pacientes, público, etc.), lo que, a su vez, se traduciría en la reducción de costes y recursos asociados a dichas tareas.

El objeto de la optimización de layouts es encontrar la disposición de espacios más eficiente dentro del espacio total al que se delimita. Para lograr ese objetivo, a lo largo del tiempo se han planteado diferentes modelos matemáticos de optimización de layouts, los cuales se han enfocado en diferentes tipos de problemas que permite resolver esta técnica, desde optimización de rutas de circulación de materiales, planificación de la producción o incluso asignación de elementos.

Con el avance de las tecnologías y el aumento de la capacidad de almacenar y procesar datos, se han incorporado técnicas de *'data mining'* y '*machine learning'* para mejorar la precisión y eficiencia de los modelos de optimización. Este tipo de técnicas permiten analizar mayores cantidades de datos y obtener información de alto interés para la toma de decisiones.

En los párrafos siguientes se exploran diferentes modelos matemáticos de optimización de espacios, desde modelos más sencillos y elementales hasta modelos más complejos.

4.1.1.1 PROBLEMA DE ASIGNACIÓN QUADRÁTICA (QAP – QUADRATIC ASSIGNMENT PROBLEM)

Esta metodología para optimización fue planteada por Koopmans y Beckmann en el año 1957 como un modelo matemático para un conjunto de actividades económicas indivisibles.

Esta técnica, por ejemplo, consiste en asignar N tareas a un número N de trabajadores, de mantea que se maximice la eficiencia de la asignación. El problema es una forma específica de optimización combinatoria, que se centra en la asignación óptima de elementos entre ambos conjuntos.

En sí, funciona de la siguiente manera:

- Se tienen N tareas y N trabajadores.
- Cada trabajador puede realizar una tarea a la vez.
- Se asigna una tarea a cada trabajador y se calcula el coste de la asignación.
- El coste de la asignación es la suma de los costes asociados con cada par de tareas y trabajadores asignados.
- El objetivo es encontrar la asignación que minimice el coste total.

Esta metodología se puede aplicar a la optimización de espacio ya que la asignación de elementos a posiciones en un layout puede considerarse una tarea de asignación. De este modo, el problema de asignación cuadrática puede utilizarse para asignar elementos a posiciones de manera que se minimice el coste de asignación.

En la optimización de layouts, se pretende encontrar la configuración óptima de espacios en un espacio físico, maximizando la eficiencia y reduciendo el tiempo de recorrido, de forma que se minimicen los costes asociados con el espacio y los recursos.

La formulación, de forma simplificada, consistiría en:

$$min \sum_{i=1}^{n} \sum_{j=1}^{n} c(i,j) \cdot x(i,j)$$

De forma que:

$$\sum_{i=1}^{n} x(i,j) = 1 \ \ para\ todo\ j$$

$$\sum_{j=1}^{n} x(i,j) = 1 \ \ para\ todo\ i$$

Donde:

$$n: número\ de\ elementos\ y\ posiciones\ en\ el\ layout.$$

$$c(i,j): coste\ asociado\ a\ esa\ posición.$$

$$x(i,j): posición\ en\ la\ que\ se\ ubica\ el\ elemento.$$

La solución óptima se logrará obtendrá encontrando la asignación de elementos a posiciones de menor coste.

Las variables que afectan al coste aumentarían en función de cómo se formule el coste. Para el caso de optimización de layouts se podrían considerar variables como la afinidad, la frecuencia de este tipo de recorrido, el coste de personas, etc.

4.1.1.2 PROGRAMACIÓN DE ENTEROS (MIP – MIXED-INTEGER PROGRAMMING)

Se trata de una técnica de optimización o factibilidad para resolver problemas donde se requiere tomar decisiones sobre variables discretas, es decir, que solo abarcan números enteros.

En la optimización para layouts, se puede optimizar buscando una función objetivo sujeta a un conjunto de restricciones. La función objetivo puede ser el coste total de la asignación de espacios a diferentes posiciones, mientras que las restricciones pueden ser limitaciones del espacio total del que se dispone, así como restricciones de distancia u otras especificas según el tipo de optimización a realizar.

Para aplicar esta optimización de aplican variables binarias, es decir, con valor 0 o 1, para representar si un elemento se ubica o no en una determinada posición.

La formulación simplificada es similar a la del caso anterior, considerando que la variable x definiría si un valor está en esa posición (asignado, por tanto, igual a 1) o no (no asignado, por tanto, igual a 0).

El objetivo sería el buscar la configuración que menor coste suponga, restringiendo el modelo a que cada espacio debe asignarse a una sola posición y cada posición debe contener un solo espacio.

Esta metodología presenta complejidades en aumentas la inclusión de restricciones, tales como dependencias entre espacios o afinidades.

4.1.1.3 PROBLEMA DE DISEÑO DE DISTRIBUCIÓN DE INSTALACIONES CON AREAS NO UNIFORMES (UAFLP – UNEQUAL AREA FACILITY LAYOUT PROBLEM)

Este método de optimización busca encontrar la ubicación óptima de instalaciones en un espacio físico determinado. La formulación básica de esta metodología incluye la minimización del costo total de la distribución de las instalaciones en el espacio, considerando factores como la proximidad, el flujo de trabajo u otras variables según la aplicación.

La principal ventaja de esta metodología de optimización es la capacidad de permitir optimizar espacios con áreas variables. Además, se adapta a una amplia gama de problemas de distribución, tales como situaciones donde las instalaciones tienen requisitos de espacios distintos y variables asociadas diferentes para cada espacio.

Asimismo, esta metodología, y las posteriores que se indican a continuación, se pueden integrar con nuevas metodologías de 'machine learning' y 'data mining'. En varios de los artículos de referencia que se han consultado, y en las referencias incluidas dentro de los mismo, se ejemplifica la aplicación de esta técnica combinando la interpretación de datos recabados en instalaciones reales, tales como la frecuencia, la cadencia, etc. Mediante estos datos, se pueden obtener optimizaciones más certeras, puesto que se pueden añadir restricciones que se adecúen mejor a la realidad y al comportamiento humano.

Esta metodología, pero, presenta una alta complejidad en la programación y requiere de unas prestaciones altas para poder procesar los cálculos, especialmente si se añaden más restricciones y la cantidad de datos a trabajar es alta.

4.1.1.4 ALGORITMO GENÉTICO (GA – GENETIC ALGORITHM)

Esta metodología, también aplicable a la optiización de layouts, se basa en el concepto de la evolución y selección natural de las especies. En continuas iteraciones, se crean poblaciones de posibles soluciones y se someten a operadores genéticos, tales como la selección, el cruce o la mutación, de forma que se creen nuevas soluciones. Estas soluciones se evaluan en función de la función objetivo (fitness function) y se seleccionan las que mejor resultado aporten. Este proceso se repite iterativamente hasta alcanzar una solución óptima.

La ventaja principal es que puede ser una técnica muy eficiente en la búsqueda de soluciones óptimas en problemas de distribución de espacios, adaptándose a objetivos diversos y varias restricciones. Otra ventaja es que la metodología tiene la capacidad de explorar diferentes soluciones, llegando a encontrar soluciones óptimas globales.

La principal desventaja de esta metodología es su alta complejidad de programación y las prestaciones en equipos que se requiere, añadiéndole el largo tiempo de ejecución que puede llevar si el número posible de soluciones es muy alto.

4.1.1.5 OPTIMIZACIÓN POR ENJAMBRE DE PARTÍCULAS (PSO – PARTICLE SWARM OPTIMIZATION)

Esta técnica de optimización se basa en imitar un enjambre de partículas que se mueven en el espacio en busca de la solución óptima. Cada partícula representa una posible solución del problema y se mueve a través del espacio delimitado por las restricciones en función de la propia experiencia y de la influencia del enjambre, es decir, del resto de partículas.

Se trata de un proceso iterativo, similar al explicado previamente, donde cada iteración actualiza las posiciones de cada partícula, evaluando en cada iteración también la función objetivo. El proceso de iteración se repite hasta alcanzar la solución que mejor se ajuste al objetivo.

Esta metodología tiene como ventaja su fácil implementación e integración de diferentes objetivos y restricciones, haciéndola una metodología versátil.

Además, se puede combinar con técnicas a la orden del día como el 'data mining' y el 'machine learning', de forma que, tomando históricos como referencia que se integren en el modelo matemático, se puedan alcanzar soluciones más acertadas según el caso y las condiciones de contorno que se asignen.

Una de las desventajas de esta metodología es que requiere de una alta capacidad de programación y en prestaciones del equipo para procesarlo, así como el tiempo en ejecutarse.

Asimismo, otra de las desventajas que presenta esta metodología es que puede quedarse atrapada en soluciones óptimas "locales" y no encontrar una solución óptima global, lo que haría entrar al algoritmo en bucle si no se definen adecuadamente las restricciones. Al añadir restricciones, la complejidad matemática del problema se intensifica.

4.1.2 CONCLUSIONES DE LA INVESTIGACION TÉORICA

Tras analizar las diferentes metodologías de optimización y varios artículos de investigación donde se han llevado a cabo, se evalúa la aplicabilidad de estos al objeto de este proyecto.

Si bien se han analizado alternativas complejas que son capaces de evaluar diferentes funciones objetivas, aplicar diferentes funciones límite y procesar grandes cantidades de datos, es necesario considerar que la aproximación que se pretende dar en este proyecto no busca abarcar soluciones altamente complejas y que tomen como referencia históricos basados en el procesado masivo de datos.

La solución propuesta para este proyecto pretende simplificar un proceso específico, el de toma de decisión, de forma que, una vez realizado el pertinente estudio, se transfieran unas variables totalizadoras a un modelo y que este las optimice.

Este proyecto no pretende realizar una investigación exhaustiva de las metodologías de optimización, sino que pretende integrar el resultado de las mismas en un modelo BIM a tomar como referencia o baso para el posterior retrabajo del mismo.

No obstante, cabe destacar que, en el desarrollo de este proyecto se ha llevado a cabo la programación de un modelo simple con optimización basada en PSO, tal y como se explica en el apartado 5.1.3 PROGRAMACIÓN DEL ALGORITMO.

Con lo expuesto, se ha optado por realizar una optimización de menor complejidad a algunas de las planteadas, aplicando, en este caso, una formulación aproximada al método de Programación de Enteros y al del Problema de Asignación Cuadrática.

4.1.3 DESARROLLO Y FORMULACIÓN

La formulación que se ha planteado para la optimización del modelo consiste en el encontrar la configuración óptima de espacios de forma que se minimice el coste.

4.1.3.1 FÓRMULA DE OPTIMIZACIÓN PRINCIPAL

La expresión matemática aplicable sería la siguiente:

$$min \sum_{i=1}^{n} \sum_{j=1}^{n} C_R(i,j) * D_{MH}(i,j)$$

Donde:

$$C_R: Coste\ de\ realizar\ la\ ruta\ entre\ i\ y\ j.$$

$$D_{MH}: Distancia\ Manhattan\ entre\ i\ y\ j.$$

$$i, j \ldots n: Espacios\ del\ modelo.$$

El objetivo del modelo es minimizar el coste total, encontrando, para cada combinación, el producto del coste de realizar una ruta entre dos espacios y la distancia Manhattan entre ellos. De esta forma, buscará de todas las combinaciones posibles, aquellas que supongan el menor coste total para el modelo.

4.1.3.1.1 COSTE DE REALIZAR UNA RUTA

Para definir el coste que supone realizar las rutas entre los diferentes espacios se utilizará el método de Tabla Relacional de Espacios. Este método se trata de una herramienta que se utiliza en el diseño de layouts, que permite identificar la relación entre varias áreas de un espacio. Las variables que definen esta relación entre cada espacio son las que el diseñador elija, como podría ser la proximidad, flujo de personas o materiales, importancia, etc.

En la siguiente figura se muestra el ejemplo de tabla relacional.

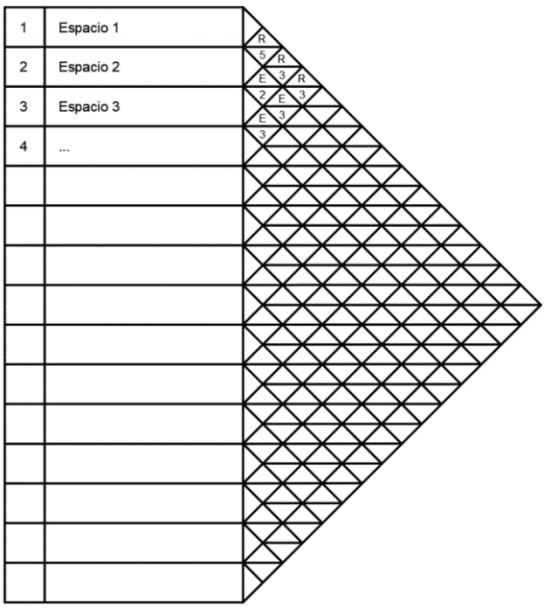

Ilustración 6. Plantilla tabla relacional.

En el caso específico de este proyecto, donde se plantea la optimización de espacios de una planta hospitalaria, para completar la tabla relacional se toman como referencia las siguientes premisas:

- **Tipo de relación**: consiste en la correlación que haya entre dos espacios. En el caso de que la correlación sea mayor, el factor aplicable será 1, ya que se asume que la importancia de que dos espacios estén juntos es mayor y, por tanto, debe suponer menor coste. Mientras que, si la correlación es menor, el coste debe ser mayor, ya que no interesa que esos dos espacios estén próximos o no sería de gran importancia.
- **Motivo o causa**: consiste en el móvil por el que una persona deba desplazarse entre dos espacios. Por ejemplo, se asume que el tránsito de personas entre dos espacios debido a una emergencia debe suponer menor coste que si el tránsito entre dos espacios no tiene relevancia.

De esta manera, la fórmula para calcular el coste es la siguiente:

$$C_R(i,j) = T_{Rel}(i,j) * M(i,j)$$

Donde:

$$T_{Rel}: Tipo\ de\ relación\ entre\ i\ y\ j.$$

$$M: Motivo\ por\ el\ que\ se\ realiza\ la\ ruta\ entre\ i\ y\ j.$$

4.1.3.1.2 DISTANCIA MANHATTAN ENTRE DOS ESPACIOS

La distancia Manhattan se calcula como la suma absoluta de las diferencias de las coordenadas X e Y entre dos puntos.

El motivo por el que se usa esta distancia y uno una distancia euclidiana (definida por el teorema de Pitágoras) es porque, a efectos de programación, la distancia Manhattan no involucra ecuaciones no lineares. Al introducir una raíz en el cálculo de la distancia euclidiana, se introduce en la formulación ecuaciones no lineares, que complicarían la ejecución del algoritmo una vez programado.

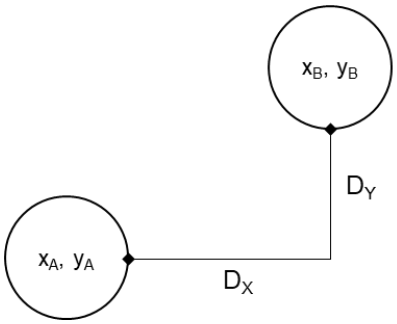

Ilustración 7. Esquema de distancia Manhattan.

La formulación sería:

$$D_{MH}(i,j) = D_X + D_Y = |x_B - x_A| + |y_B - y_A|$$

Donde:

$$D_X, D_Y: Distancias\ absolutas\ en\ eje\ x\ e\ y\ respectivamente.$$

$$x_i, y_i: Coordenadas\ x\ e\ y\ del\ centro\ de\ cada\ espacio.$$

4.1.3.1.3 LÍMITES Y RESTRICCIONES

Para aplicar la metodología propuesta adecuadamente, se hace preciso delimitar el modelo para adecuar los resultados obtenidos, de forma que se asegure que los datos resultantes sean correctos y que no supongan errores posteriores.

Para la definición de los límites y, consecuentemente, la posterior estructura general de programación, se ha tomado como referencia el trabajo "Dulle – Facility Layout Optimization and Evaluation Engine" de F. Dueñas Llera, adecuando las variables y restricciones, formulación, procesos, estructura en detalle de programación, funciones y cualquier otro condicionante al presente proyecto, y considerando los datos de salida deseados del algoritmo.

Límites de espacios máximos

En este caso, se establecen dos condiciones básicas que se expresan de la siguiente manera:

$$\sum_{i=1}^{n} x_i + \frac{A_i}{2} \leq xAnchoTotal$$

$$\sum_{i=1}^{n} y_i + \frac{L_i}{2} \leq yLargoTotal$$

Donde:

$$x_i, y_i: Coordenadas\ x\ e\ y\ del\ centro\ de\ cada\ espacio.$$

$$A_i, L_i: Dimensiones\ de\ ancho\ y\ longitud\ de\ cada\ espacio.$$

$$xAnchoTotal, yLargoTotal: Ancho\ total\ y\ longitud\ total\ de\ la\ parcela.$$

El objetivo de esta restricción es asegurar que el ancho o largo de un espacio no sobrepasa el espacio total disponible de parcela donde se asentarán los espacios optimizados.

Límites de espacios mínimos

Las expresiones que restringen los límites mínimos son las siguientes:

$$\sum_{i=1}^{n} x_i - \frac{A_i}{2} \geq 0$$

$$\sum_{i=1}^{n} y_i - \frac{L_i}{2} \geq 0$$

De este modo, se asegura que ningún espacio es igual a 0 o inferior, puesto que no tendría sentido.

Límites de superposición

El mecanismo para evitar que los espacios se superpongan entre ellos es crear una serie de variables de control en las que se evalúe la posición de un espacio respecto al otro.

Cabe destacar que el eje de coordenadas tomado como referencia contempla que el eje de las x asciende hacia la derecha y el eje de las y hacia abajo, tal y como se muestra en la figura siguiente:

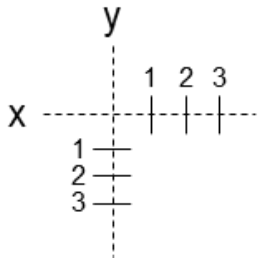

Ilustración 8. Eje de coordenadas de referencia.

Las expresiones de las variables de control son las especificadas a continuación.

Ubicación a derecha o izquierda:

$$\sum_{i=1}^{n}\sum_{j=1}^{n} C_1(i,j) = 1 \ si \ x_i + \frac{A_i}{2} \leq x_j - \frac{A_j}{2}$$

$$\sum_{i=1}^{n}\sum_{j=1}^{n} C_2(i,j) = 1 \ si \ x_j + \frac{A_j}{2} \leq x_i - \frac{A_i}{2}$$

La variable de control C1 determina si un espacio está ubicado a la izquierda del otro. En caso de que no sea así, debe determinarse si está a la derecha mediante la variable C2. En el esquema siguiente puede entenderse la aplicación:

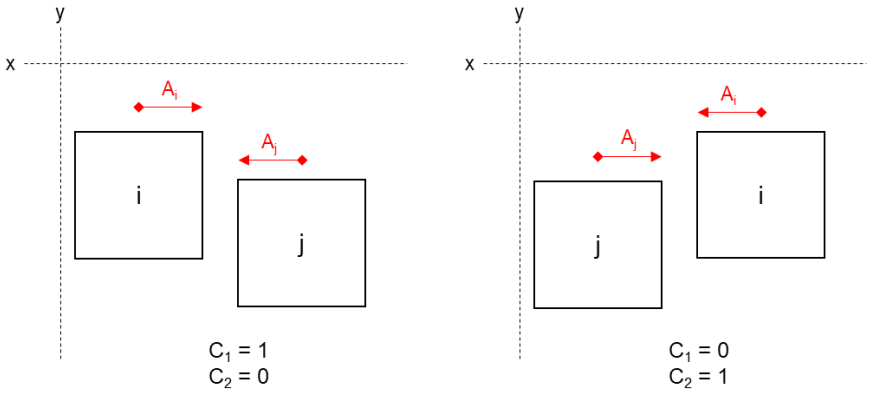

Ilustración 9. Diagrama relación izquierda - derecha.

Ubicación debajo o encima:

$$\sum_{i=1}^{n}\sum_{j=1}^{n} C_3(i,j) = 1 \ si \ y_i + \frac{L_i}{2} \leq y_j - \frac{L_j}{2}$$

$$\sum_{i=1}^{n}\sum_{j=1}^{n} C_3(i,j) = 1 \ si \ y_j + \frac{L_j}{2} \leq y_i - \frac{L_i}{2}$$

Similar al caso anterior, las variables de control C3 y C4 comprueban si un espacio respecto al otro está ubicado encima o debajo. Se incluye diagrama explicativo.

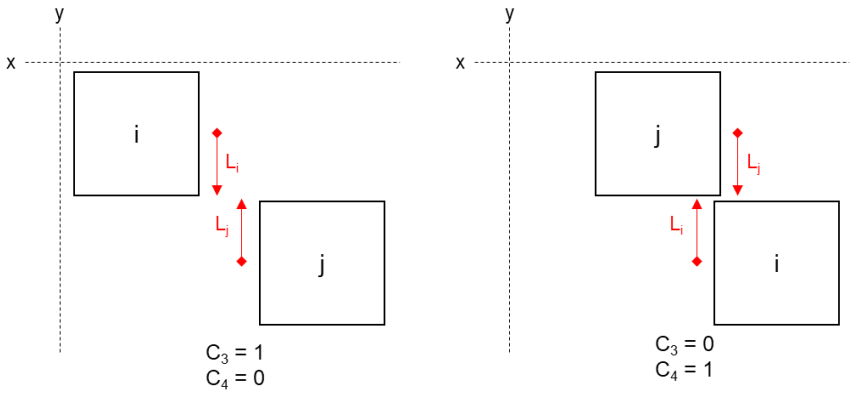

Ilustración 10. Diagrama relación encima - debajo.

Siempre debe cumplirse alguna relación de cada par para confirmar que no se superponen.

$$\sum_{i=1}^{n}\sum_{j=1}^{n} C_1(i,j) + C_2(i,j) + C_3(i,j) + C_4(i,j) \geq 1$$

Hay que recordar que la optimización principal buscará constantemente posicionar el centro de cada espacio y, a continuación, analizará esa ubicación según los límites que se definen. Estos límites, toman como referencia el centro hallado y lo evalúan añadiendo el ancho o largo para comprobar la interacción con otros espacios o el límite del espacio total.

4.2 TRATAMIENTO DE LA INFORMACIÓN Y OBTENCIÓN DE MODELO BIM PRELIMINAR

Partiendo de la formulación expuesta en el apartado previo, se plantea la forma en que esa se aplicará a un modelo de optimización que, posteriormente, debe ser capaz de extraer los resultados de forma matemática y visual. Finalmente, se debe encontrar la manera de trasladar esa información a un entorno BIM donde se pueda trabajar el modelo finalmente obtenido.

Primeramente, se analizan los lenguajes de programación que facilitarían integrar en código la formulación planteada. Asimismo, se plantean los softwares de referencia que se utilizarán para llevar a cabo la aplicación y cómo interaccionarán entre ellos.

4.2.1 SOFTWARES Y HERRAMIENTAS

A continuación, se justifican las herramientas utilizadas en el desarrollo de este proyecto.

4.2.1.1 LENGUAJE DE PROGRAMACIÓN 'PYTHON'

Se escoge Python como lenguaje de programación debido a que se trata de un lenguaje de uso fácil, que ofrece una gran flexibilidad y capacidad para manejar grandes cantidades de datos.

Cuenta con numerosas bibliotecas especializadas en, entre otros, el campo de la optimización y el tratamiento de datos. Algunos ejemplos serían NumPy, SciPy, y Pandas.

Gracias a estas bibliotecas, la implementación de un algoritmo de optimización con formulación como la expuesta previamente resulta más sencilla. Incluso, una formulación basada en expresiones más complejas (PSO, GA, etc.) también sería integrable en este lenguaje.

Python se trata de un código abierto, lo que permite a usuarios acceder y modificar código fuente de bibliotecas y adaptarlos a necesidades específicas. Muchas de estas

modificaciones se encuentran compartidas en plataformas como Github, entre otras, lo que facilita el acceso a herramientas a utilizar en Python y, sobre todo, al aprendizaje de este lenguaje si no se tiene conocimientos previos de este lenguaje.

Otra ventaja que ofrece este lenguaje es la existencia de librerías para visualizar datos y exportar a otros formatos que permiten la interoperabilidad entre softwares, como podría ser el formato separado por comas, *CSV.*

Por último, este lenguaje se integra fácilmente con motores de optimización, lo que permitirá llevar a cabo grandes cantidades de operaciones matemáticas de forma eficiente, como es el caso que se expone en el siguiente apartado.

4.2.1.2 VISUAL STUDIO CODE

Para trabajar en el lenguaje Python, se requiere del uso de un entorno de edición de código fuente. En este caso, se escoge Visual Studio Code por tratarse de una herramienta abierta y gratuita.

Entre otras ventajas, este programa cuenta con una interfaz muy intuitiva a usuarios con un nivel de programación bajo y permite personalizar el entorno completo.

Cuenta también con herramientas de depuración de código, que facilitan la generación de tramas de código fuente. También se integra con servicios de desarrollo como GitHub y otras plataformas.

4.2.1.3 GUROBI

El software Gurobi consiste en un motor de optimización matemática que permite resolver problemas de diferentes tipos y complejidad (optimización lineal, no lineal, enteros mixtos, etc.). Permite resolver problemas complejos en un tiempo reducido.

Este software se integra fácilmente en Python gracias a las bibliotecas de Python, como gurobipy, que permite acceder y utilizar todas las funciones de Gurobi en un entorno IDE donde se trabaje el código Python.

Gurobi, además, ofrece una gran cantidad de funcionalidades y herramientas de visualización que permiten al usuario mostrar y analizar los resultados.

Cabe destacar, además, que este software de optimización se encuentra disponible de forma gratuita a estudiantes universitarios, lo que ha facilitado la elección del motor de optimización.

4.2.1.4 DYNAMO

El software Dynamo se trata de una herramienta de diseño paramétrico basada en nodos, que se utiliza de forma integrada en el diseño arquitectónico y de construcción. Permite crear diseños complejos y adaptados a las necesidades del proyecto. Su uso se facilita ya que se trata de una herramienta de programación visual que permite al usuario crear y modificar modelos paramétricos de forma interactiva.

Esta herramienta se integra fácilmente con otros softwares con los que normalmente viene preinstalado en formato de extensión, como sería el caso de Revit, AutoCAD, entre otros.

Esta herramienta servirá de interfaz entre los datos extraídos entre el algoritmo de optimización y el entorno BIM ya que, gracias a que permite la programación lineal, entre sus funciones múltiples presenta la lectura de datos, interpretación y, en base a ellos, la generación de modelos 3D, por ejemplo.

Otra de las ventajas es su fácil aprendizaje si no se dispone de conocimientos previos gracias a la interfaz de uso gráfica que resulta más intuitiva y fácil.

4.2.1.5 REVIT

Se trata de un software de diseño de edificación y construcción en modelos 3D. Está desarrollado por Autodesk y una de las principales ventajas y principal razón por la que se ha seleccionado es porque se trata de un programa ampliamente utilizado en la industria dedicada a la construcción o proyectos constructivos.

Además, se encuentra disponible una versión gratuita a estudiantes universitarios.

El software Revit integra el software Dynamo comentado previamente, permitiendo el diseño paramétrico a partir de dicha herramienta. Asimismo, permite un diseño integrado y que aplica la metodología BIM, la cual se pretende trabajar en este proyecto.

4.2.1.6 MICROSOFT EXCEL

Es un software de Microsoft que permite crear y operar hojas de cálculos, realizando operaciones de diferente complejidad. También permite el tratamiento de datos, extracción de gráficas, uso de tablas dinámicas, etc.

Entre las funcionalidades que tiene, permite el tratamiento de datos en formato *Comma Separated Values (CSV)*, que se trata de un formato de archivos que se utiliza para almacenar datos en formato de tablas. Los valores se almacenan de forma que cada columna está separada por comas.

Este formato sirve como intercambio entre los diferentes softwares a utilizar en este proyecto debido a su facilidad de uso y que se permite su edición y creación mediante herramientas como Microsoft Excel o a partir de lenguaje Python.

Asimismo, también se utilizarán tablas Excel para realizar cálculos o almacenar datos.

La principal ventaja del uso de este software y los formatos que permite trabajas es que, dado que Dynamo incluye nodos programables, se puede leer o grabar datos desde esta herramienta o, incluso, directamente en el entorno de Revit.

4.2.2 FLUJO DE INFORMACIÓN

A continuación, se expone un diagrama de flujo de información donde se plasma de forma visual la interacción entre los diferentes softwares y herramientas a utilizar en este proyecto.

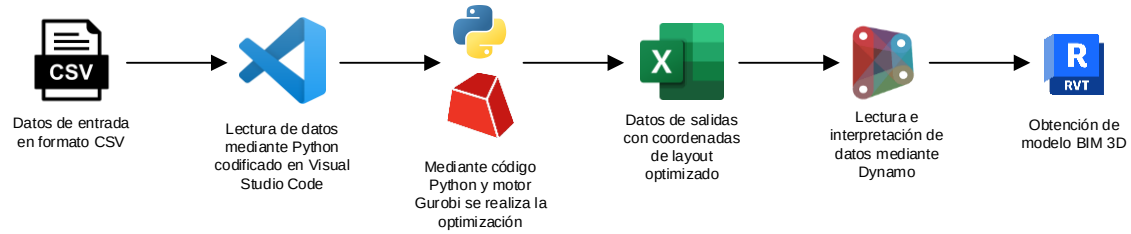

Ilustración 11. Flujo de información y datos.

5 DESARROLLO DE LA SOLUCIÓN ESCOGIDAS

El presente apartado se enfoca en el desarrollo de la solución escogida para abordar el problema planteado que da pie a este proyecto.

Habiendo definido previamente la metodología y las bases del proceso a seguir, ahora se pretende exponer el desarrollo y la implementación de la solución elegida, así como la obtención de resultados.

Al igual que en el planteamiento de la solución, este apartado se divide en los dos grandes bloques que marcan el proyecto. Por una parte, se lleva a cabo la realización de un algoritmo de optimización que dé como fruto una configuración de espacios optimizada.

Por otro lado, se transforma la información obtenida en un modelo BIM que servirá de base para futuro trabajos de desarrollo BIM.

5.1 DESARROLLO DE ALGORITMO DE OPTIMIZACIÓN

5.1.1 CÁLCULO DE COSTES DE RUTAS

Siguiendo la formulación planteada previamente, la creación del algoritmo parte de la definición del coste de hacer las rutas entre diferentes espacios. Para ello, se ha seguido el método de tabla relacional.

5.1.1.1 DATOS DE ENTRADA DEL PROBLEMA

Los datos de entrada que se han considerado son los aportados como base del problema. En este caso, los datos han correspondido con:

- Tipo de instalación y dimensión total: ala de emergencias de un hospital, de 100x70 m.
- Espacios: los espacios considerados y sus dimensiones han sido tomados como dato de entrada. En este proyecto no se ha incluido en su alcance evaluar el tipo de espacios y las dimensiones que estos requieren para el ala de emergencias de un hospital. Se considera que esta definición forma parte de un estudio previo y de unas decisiones que no están entre los objetivos que pretende lograr este proyecto.

  Los espacios en cuestión son los siguientes:

Tabla 2. Datos de entrada - Espacios.

ID	Nombre de espacio	Ancho (m)	Largo (m)	Área (m^2)
1	Recepción principal	5	3	15
2	Área de consultas	20	30	600
3	Sala de espera principal	15	10	150
4	Oficina central	4	7	28
5	Vestuarios y sala de personal	12	10	120
6	Ecografías	13	11	143
7	Emergencias médicas	10	12	120

ID	Nombre de espacio	Ancho (m)	Largo (m)	Área (m^2)
8	Sala de espera Rx	18	15	270
9	Inyecciones, vacunas y extracciones	11	20	220
10	Enfermería	15	18	270
11	Radiología	10	15	150
12	Recepción de rehabilitación	5	4	20
13	Sala de espera rehabilitación	10	12	120
14	Rehabilitación	15	15	225
15	Ultrasonidos	12	10	120

5.1.1.2 GENERACIÓN DE TABLA RELACIONAL

Para llevar a cabo la tabla relacional de los espacios indicados, se han tomado como referencia los criterios indicados en el planteamiento.

Los criterios se han desarrollado de la siguiente manera:

- **Tipo de relación**: se han definido cinco niveles de correlación según la vinculación que existiría entre un espacio y otro.

Tabla 3. Criterio 'tipo de relación'.

Código	Factor	Criterio
A	1	Absolutamente importante
E	2	Especialmente importante
I	3	Importante
O	4	Ordinaria
U	5	Sin importancia

- **Motivo o causa**: para definir este criterio se establecen también cinco niveles determinados según el tipo de recorrido que haría un paciente dentro del ala de emergencias. Los tipos de recorrido han sido extraídos de estudios de referencia que se han consultado y que figuran en la bibliografía del presente proyecto. Estos recorridos han sido simplificados según el criterio propio, intentando aportar el mayor coste a los recorridos que se interpretan 'más complejos' y menor coste a los que requieren mayor simplicidad.
    - Emergencia médica: un paciente que entra al ala debido a una emergencia en el que puede existir riesgo a la integridad física o mental.

- Visitas médicas o enfermería: consiste en el recorrido para desplazarse a la zona de consultas médicas o de enfermería, donde se realicen exploraciones o curas.
- Movimiento entre espacios: se trata del movimiento natural entre espacios de forma necesaria, bien sea por pacientes o por el personal de la instalación.
- Pruebas, extracciones y especialidades: consiste en el itinerario a seguir para llegar a zonas donde se realicen pruebas (ecografías, radiografías, etc.), extracciones de sangre o fluidos o visitas a personal especializado.
- Sin importancia o no deseado: aquellos recorridos que no interesa llevar a cabo o que no suponen ningún beneficio para la optimización.

Los valores asignados son los siguientes:

Tabla 4. Criterio 'motivo o causa'.

Código	Factor	Criterio
1	1	Emergencia médica
2	2	Visita médica o enfermería
3	3	Movimiento entre espacios
4	4	Pruebas, extracciones y especialidades
5	5	Sin importancia o no deseado

Con los criterios indicados, se lleva a cabo la tabla relacional aplicando el criterio propio, llevando a cabo la tarea propia del equipo diseñador del proyecto.

El resultado obtenido es el siguiente:

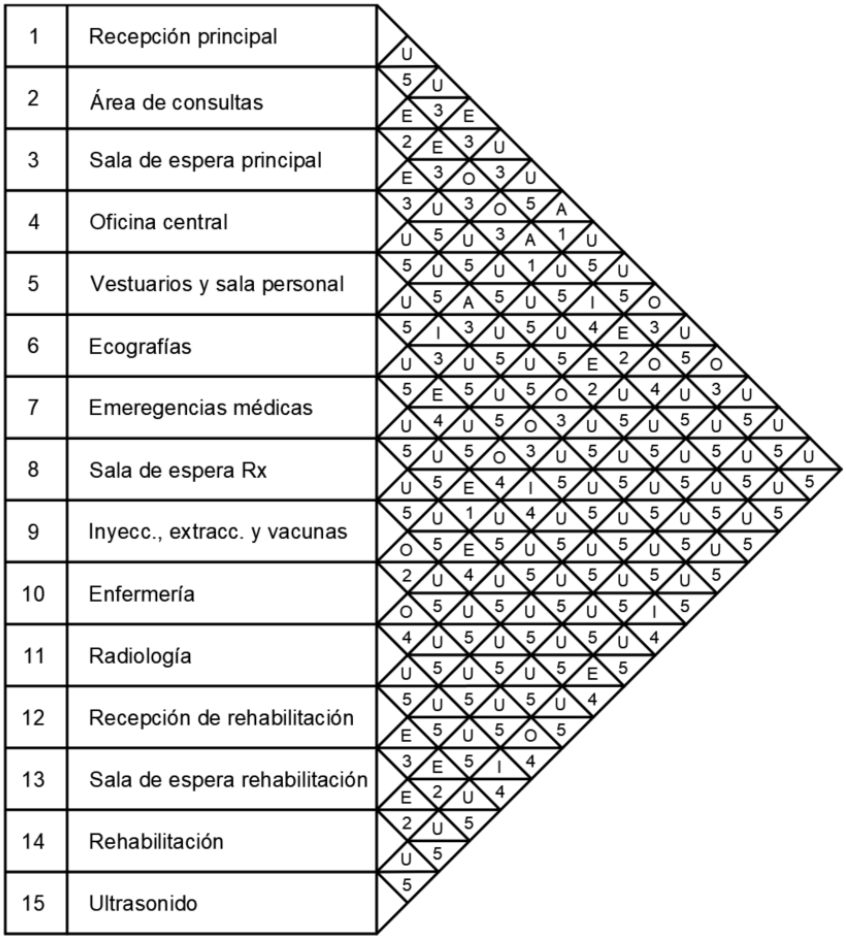

Ilustración 12. Tabla relacional de proyecto.

5.1.1.3 COSTES RESULTANTES

Con la tabla relacional planteada y los factores definidos, se aplica la fórmula para el cálculo del coste de cada ruta entre espacios:

$$C_R(i,j) = T_{Rel}(i,j) * M(i,j)$$

Los resultados de cada ruta se plasman en la siguiente matriz, la cual servirá como dato de entrada al algoritmo de optimización junto a los datos de los espacios.

Tabla 5. Coste de rutas entre espacios.

ID	Nombre de espacio	Recepción principal	Área de consultas	Sala de espera principal	Oficina central	Vestuarios y sala personal	Ecografías	Emegencias médicas	Sala de espera Rx	Inyecciones, vacunas y extracciones	Enfermería	Radiología	Recepción de rehabilitación	Sala de espera rehabilitación	Rehabilitación	Ultrasonidos
1	**Recepción principal**	0	25	15	6	15	25	1	25	25	12	25	12	25	25	25
2	**Área de consultas**	25	0	4	6	12	12	1	25	12	4	16	25	25	25	25
3	**Sala de espera principal**	15	4	0	6	25	25	25	25	25	4	25	25	25	25	25
4	**Oficina central**	6	6	6	0	25	25	3	25	25	12	25	25	25	25	25
5	**Vestuarios y sala personal**	15	12	25	25	0	25	9	25	25	12	25	25	25	25	25
6	**Ecografías**	25	12	25	25	25	0	25	8	25	16	12	25	25	25	12
7	**Emegencias médicas**	1	1	25	3	9	25	0	25	25	2	25	25	25	25	25
8	**Sala de espera Rx**	25	25	25	25	25	8	25	0	25	25	8	25	25	25	8
9	**Inyecciones, vacunas y extracciones**	25	12	25	25	25	25	25	25	0	8	25	25	25	25	25
10	**Enfermería**	12	4	4	12	12	16	2	25	8	0	16	25	25	25	16
11	**Radiología**	25	16	25	25	25	12	25	8	25	16	0	25	25	25	12
12	**Recepción de rehabilitación**	12	25	25	25	25	25	25	25	25	25	25	0	6	4	25
13	**Sala de espera rehabilitación**	25	25	25	25	25	25	25	25	25	25	25	6	0	4	25
14	**Rehabilitación**	25	25	25	25	25	25	25	25	25	25	25	4	4	0	25
15	**Ultrasonidos**	25	25	25	25	25	12	25	8	25	16	12	25	25	25	0

5.1.2 ARCHIVOS DE ENTRADA

Con los datos de entrada del algoritmo conseguidos, se deben generar los archivos *CSV* que leerá el código Python.

Los archivos en cuestión contendrán, respectivamente, la información de espacios (número de ID, ancho, largo) y los costes de rutas conforme a la matriz generada previamente.

Los archivos, procesados a través de Microsoft Excel, tienen el siguiente aspecto una vez exportados a *CSV.*

	A
1	DIMENSIONES,ANCHO,LARGO
2	1,5,3
3	2,20,30
4	3,15,10
5	4,4,7
6	5,12,10
7	6,13,11
8	7,10,12
9	8,18,15
10	9,11,20
11	10,15,18
12	11,10,15
13	12,5,4
14	13,10,12
15	14,15,15
16	15,12,10

	A
1	DATOS RUTAS,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15
2	1,0,25,15,6,15,25,1,25,25,12,25,12,25,25,25
3	2,25,0,4,6,12,12,1,25,12,4,16,25,25,25,25
4	3,15,4,0,6,25,25,25,25,25,4,25,25,25,25,25
5	4,6,6,6,0,25,25,3,25,25,12,25,25,25,25,25
6	5,15,12,25,25,0,25,9,25,25,12,25,25,25,25,25
7	6,25,12,25,25,25,0,25,8,25,16,12,25,25,25,12
8	7,1,1,25,3,9,25,0,25,25,2,25,25,25,25,25
9	8,25,25,25,25,25,8,25,0,25,25,8,25,25,25,8
10	9,25,12,25,25,25,25,25,25,0,8,25,25,25,25,25
11	10,12,4,4,12,12,16,2,25,8,0,16,25,25,25,16
12	11,25,16,25,25,25,12,25,8,25,16,0,25,25,25,12
13	12,12,25,25,25,25,25,25,25,25,25,25,0,6,4,25
14	13,25,25,25,25,25,25,25,25,25,25,25,25,6,0,4,25
15	14,25,25,25,25,25,25,25,25,25,25,25,25,4,4,0,25
16	15,25,25,25,25,25,25,12,25,8,25,16,12,25,25,25,0
17	DIMENSIONES PARCELA,100,,70,,,,,,,,,,,,

Ilustración 13. Archivos CSV de entrada de datos al algoritmo.

Los archivos se nombran de la siguiente manera:

- Dimensiones.CSV
- Rutas.CSV

5.1.3 PROGRAMACIÓN DEL ALGORITMO

La programación del algoritmo se lleva a cabo en el software Visual Studio Code, versión 1.77.3, configuración de usuario, para Windows 11.

Asimismo, la versión de Python utilizada es la 3.10.

En el caso del software Gurobi, la versión utilizada es la 10.0.1., descargada de forma gratuita con licencia de estudiante tras la correspondiente acreditación de estudios en curso.

Las condiciones bajo las que se desarrolla el código que se explica más adelante, son las siguientes:

- Se analizan espacios con dimensiones no variables.
- Se delimita el espacio máximo que pueden ocupar los diferentes departamentos que compondrán el layout.
- Se asume que el algoritmo buscará la mejor solución matemática. No entra dentro de las capacidades del algoritmo evaluar la funcionalidad del resultado.

El código resultante de la implementación de la formulación planteada es el que se explica a continuación.

Tabla 6. Código Python analizado.

Código Python	Explicación de código

`import tkinter.ttk as ttk`
`from tkinter import *`
`from gurobipy import *`
`from gurobipy import GRB, Model, abs_, quicksum`
`from threading import Thread`
`from tkinter import Frame`
`from tkinter.font import Font`
`from openpyxl import Workbook, load_workbook`
`import CSV`
	Importación de librerias:
	- Tkinter.ttk: para crear salidas de datos de forma visual.
	- gurobipy: librería para acceder a los procesos de optimización del SW Gurobi. Previamente hay que instarlo y activarlo.
	- openpyxl: manipular datos de Excel.
	- CSV: manipular datos de CSV.

`roota = Tk()`
`roota.geometry("1000x500")`
`roota.configure(background="white", bg='white')`
`roota.title("Algoritmo de optimización - Christyan Morante")`
`nb = ttk.Notebook(roota)`
`first = ttk.Frame(roota)`
`second= ttk.Frame(roota)`
`nb.add(first, text='Configuración optimizada')`
`nb.add(second, text='Resultados')`
`nb.pack(expand="1" , fill="both")`
	Creación de ventanas para mostrar resultados posteriormente.

`class LeeDimensiones():`
`  def __init__ (self, filename):`
`  with open (filename, 'r', encoding='utf-8') as f_input:`
`  CSV_input = CSV.reader(f_input)`
`  self.dimensiones = list(CSV_input)`
`  def GetAncho (self):`
`  return self.dimensiones[row][1]`
`  def GetLargo (self,row):`
`  return self.dimensiones[row][2]`
	Función para extraer datos de espacios de archivo CSV.

`class LeeRutas():`
`  def __init__ (self, filename):`
`  with open (filename, 'r', encoding='utf-8') as f_input:`
`  csv_input = csv.reader(f_input)`
`  self.rutas = list(csv_input)`
`  def GetRutas (self, row, column):`
`  return self.rutas[row-1][column-1]`
	Función para extraer datos de coste de rutas de archivo CSV.

`model = Model('Opt')`
`CR = {}`
`MRutas = {}`
`Anch = {}`
`Larg = {}`
`x = {}`
`y = {}`
`c1 = {}`
`c2 = {}`
`c3 = {}`
`c4 = {}`
`Dx = {}`
`Dy = {}`
`Dxabs = {}`
`Dyabs = {}`
`Dij = {}`
	Definición del modelo matemático y declaración de variables.

`InfoRutas = LeeRutas('Rutas.csv')` `rows = len(InfoRutas.rutas)`	Llamada a la función de leer datos de coste de rutas.

Código Python	Explicación de código
`NumEspacios = int(rows-2)`	

`for row in range (1, NumEspacios+1):`
`  for column in range (1, NumEspacios+1):`
`  val = int (InfoRutas.rutas[row][column])`
`  MRutas[row,column] = val`
`for row in range(1,NumEspacios+1):`
`  for column in range(1,NumEspacios+1):`
`  CR[row,column] = MRutas[row,column]`
	Variables para lectura temporal de datos de rutas.

`xMAXIM = int(InfoRutas.rutas[rows-1][1])` `yMAXIM = int(InfoRutas.rutas[rows-1][3])`	Variables para lectura de tamaños máximos de la edificación.
`InfoDimensiones = LeeDimensiones('Dimensiones.csv')`	Llamada a la función leer espacios.

`for row in range(1,NumEspacios+1):`
`  Anch[row] = float(InfoDimensiones.dimensiones[row][1])`
`  Larg[row] = float(InfoDimensiones.dimensiones[row][2])`
	Variables para lectura temporal de dimensiones de espacios.

`for row in range(1,NumEspacios+1):`
`  x[row]= model.addVar(vtype=GRB.CONTINUOUS, name='x_{}'.format(row))`
`  y[row]= model.addVar(vtype=GRB.CONTINUOUS, name='y_{}'.format(row))`
	Variables de coordenadas X e Y temporales donde ubicar espacios en el modelo.

`for row in range(1,NumEspacios+1):`
`  for column in range (1,NumEspacios+1):`
`  Dij[row,column] = model.addVar(vtype=GRB.CONTINUOUS, lb=-10000 , ub =1000, name='d_{}{}'.format(row,column))`
`  Dx[row,column] = model.addVar(vtype=GRB.CONTINUOUS, lb=-10000 ,ub =1000,name='d_x{}{}'.format(row,column))`
`  Dy[row,column] = model.addVar(vtype=GRB.CONTINUOUS, lb=-10000,ub =1000,name='d_y{}{}'.format(row,column))`
`  Dxabs[row,column] = model.addVar(vtype=GRB.CONTINUOUS, lb=-10000,ub = 1000,name='d_x{}{}'.format(row,column))`
`  Dyabs[row,column] = model.addVar(vtype=GRB.CONTINUOUS, lb=-10000 ,ub =1000,name='d_y{}{}'.format(row,column))`
	Variables para cálculo de distancias en el modelo.

`for row in range (1,NumEspacios+1):`
`  for column in range (1,NumEspacios+1):`
`  if column>row:`
`  c1[row,column] = model.addVar(vtype=GRB.BINARY, name="s_1{}{}".format(row,column))`
`  c2[row,column] = model.addVar(vtype=GRB.BINARY, name="s_2{}{}".format(row,column))`
`  c3[row,column] = model.addVar(vtype=GRB.BINARY, name="s_3{}{}".format(row,column))`
`  c4[row,column] = model.addVar(vtype=GRB.BINARY, name="s_4{}{}".format(row,column))`
	Variables de control en el modelo.

`model.update()`	Actualización y repetición de funciones.

`for row in range(1,NumEspacios+1):`
`  model.addConstr( x[row]+(Anch[row]/2) <= xMAXIM)`
`  model.addConstr( x[row]-(Anch[row]/2) >= 0)`
`  model.addConstr( y[row]+(Larg[row]/2) <= yMAXIM)`
`  model.addConstr( y[row]-(Larg[row]/2) >= 0)`
	Operaciones matemáticas de límites superiores e inferiores.

Código Python	Explicación de código

`for row in range (1,NumEspacios+1):`
`  for column in range (1,NumEspacios+1):`
`  if(column>row):`
`  model.addConstr (Dx[row,column] == (x[row] - x[column]))`
`  model.addConstr (Dy[row,column] == (y[row] - y[column]))`
`  model.addConstr (Dxabs[row,column] == abs_(Dx[row,column]))`
`  model.addConstr (Dyabs[row,column] == abs_(Dy[row,column]))`
`  model.addConstr (Dij[row,column] == Dxabs[row,column] + Dyabs[row,column])`
`  model.addConstr((c1[row,column] == 1) >> (x[row] + (Anch[row]/2) <= x[column] -(Anch[column]/2)))`
`  model.addConstr((c2[row,column] == 1) >> ( x[column]+(Anch[column]/2) <= x[row]-(Anch[row]/2)))`
`  model.addConstr((c3[row,column] == 1) >> ( y[column]+(Larg[column]/2) <= y[row]-(Larg[row]/2)))`
`  model.addConstr((c4[row,column] == 1) >> ( y[row]+(Larg[row]/2) <= y[column]-(Larg[column]/2)))`
`  model.addConstr(c1[row,column] + c2[row,column] + c3[row,column] + c4[row,column] >=1)`
	Operaciones matemáticas de ubicación de espacios y comprobación de superposición.

`obj = quicksum ( Dij[row,column] * CR[row,column]`
`  for row in range (1,NumEspacios+1)`
`  for column in range (1,NumEspacios+1)`
`  if column>row`
	Obtención del coste de cada par en cada iteración.

`model.setObjective(obj, GRB.MINIMIZE)`	Función objetivo para optimizador de Gurobi.

`model.Params.TimeLimit = 1800`
`model.Params.MipFocus=1`
`model.Params.BestObjStop=1500`
	Parametrización de las operaciones.

`model.optimize()`
`obj = model.ObjVal`
`obj = round(obj,4)`
`x_1 = model.getVarByName("x_1")`
`x_1 = round(x_1.X , 4)`
`y_1 = model.getVarByName("y_1")`
`y_1 = round (y_1.X , 4)`
`x_2 = model.getVarByName("x_2")`
`x_2 = round (x_2.X , 4)`
`y_2 = model.getVarByName("y_2")`
`y_2 = round (y_2.X , 4)`
`d_12 = model.getVarByName("d_12")`
`d_12 = round (d_12.X , 4)`
	Obtención de datos optimizados. Esta estructura se repite para el caso de hasta 15 espacios, aunque es ampliable al número de espacios deseado.

`label_results_L0= Label(second, text = " COORDENADAS DE CADA ESPACIO:", font = ("Arial", 10), bg="white").place(relx=0.01,rely=0.05)`
`label_results_D0= Label(second, text = " DISTANCIAS ENTRE ESPACIOS (m):", font = ("Arial", 10), bg="white").place(relx=0.27,rely=0.01)`
`label_results_T0= Label(second, text = " COSTE DEL LAYOUT OPTIMIZAADO ES: " + str(obj) , font = ("Arial", 10), bg="white").place(relx=0.01,rely=0.01)`
	Presentación de resultados en formato texto. Se informa de coste total, ubicación en coordenadas de cada espacio y distancia entre pares de espacios. Esta estructura se repite para el caso de hasta 15 espacios, aunque es ampliable al número de espacios deseado.

Código Python	Explicación de código

`label_results_L1= Label(second, text = "Espacio 1 (X = " + str(x_1) + " , Y = " + str(y_1) + ")", bg="white").place(relx=0.05,rely=0.09)`
`label_results_L2= Label(second, text = "Espacio 2 (X = " + str(x_2) + " , Y = " + str(y_2) + ")", bg="white").place(relx=0.05,rely=0.12)`
`label_results_D1= Label(second, text = "Distancia entre 1 y 2 es " + str(d_12) , bg="white").place(relx=0.3,rely=0.05)`	

`label_results_graph= Label(first, text = " CONFIGURACIÓN ÓPTIMA:", bg="white", font=("Arial",10)).place(relx=0.01,rely=0.01)`
`AlturaCanvas= yMAXIM*7`
`AnchoCanvas = xMAXIM*7`
`canvas3 = Canvas(first, height= AlturaCanvas, width= AnchoCanvas, bg = "white")`
`canvas3.place(relx= 0.05 , rely = 0.05)`
	Generación de pestaña para presentar resultados gráficos.

`wb2 = load_workbook('Coordenadas_Dynamo.xlsx')` `ws = wb2['Coordinates']`	Acceso a archivo donde almacenar coordenadas optimizadas.

`Dept1_rectangle = canvas3.create_rectangle( 7*(x_1 - (Anch[1]/2)), 7*(y_1 + (Larg[1]/2)), 7*(x_1 + (Anch[1]/2)), 7*(y_1 - (Larg[1]/2)), fill="white")`
`Dept1_lable = canvas3.create_text( 7*x_1, 7*y_1 , text = "1" , fill = "black")`
`ws['A1'] = x_1-(Anch[1]/2)`
`ws['B1'] = y_1-(Larg[1]/2)`
`ws['A2'] = x_1+(Anch[1]/2)`
`ws['B2'] = y_1-(Larg[1]/2)`
`ws['A3'] = x_1+(Anch[1]/2)`
`ws['B3'] = y_1+(Larg[1]/2)`
`ws['A4'] = x_1-(Anch[1]/2)`
`ws['B4'] = y_1+(Larg[1]/2)`
	Generación gráfica de espacio en ubicación optimizada y almacenamiento de datos en archivo Excel. Esta estructura se repite para el caso de hasta 15 espacios, aunque es ampliable al número de espacios deseado.

`wb2.save('Coordenadas_Dynamo.xlsx')` `roota.mainloop()`	Guardado de Excel con datos optimizados y cierre.

El código completo se encuentra en el anejo n.º 1 de esta memoria.

Alternativa experimental

En el desarrollo de la programación, se trató de explorar la optimización basada en la metodología PSO definida en apartados previos. Se realizaron pruebas simples con modelos muy acotados de optimización donde la única restricción planteada era el uso de la menor área posible, tomando como referencia siempre el área que ocuparía cada espacio.

Al igual que en el modelo programado desarrollado finalmente en el proyecto, los datos de las dimensiones de los espacios se extraían de un archivo externo tipo CSV. A continuación, se incluye la explicación del código experimental que se logró para la optimización con PSO:

Código Python	Explicación de código

`import numpy as np`
`from pyswarm import pso`
`import csv`
`import matplotlib.pyplot as plt`
	Importación de librerias:
	- numpy: operaciones matemáticas
	- pyswarm: implementa algormitmo PSO.
	- CSV: manipular datos de CSV.
	- matplotlib: visualización de datos.

`def read_spaces(filename):`
`  with open(filename, 'r') as f:`
`  reader = csv.reader(f)`
`  spaces = []`
`  next(reader)`
`  for row in reader:`
`  spaces.append([float(row[0]), float(row[1])])`
`  return spaces`
	Definición de función de leer datos de espacios.

`def objective(x, spaces):`
`  total_area = np.sum(np.prod(x.reshape(-1, 1), axis=0) * np.array(spaces))`
`  return -total_area`
	Definición de función de cálculo de área.

`def optimize_spaces(spaces):`	Definición de la función de optimización.
`  lb = np.zeros(len(spaces))` `  ub = np.ones(len(spaces))`	Definición de límites.
`  xopt, fopt = pso(objective, lb, ub, args=(spaces,), maxiter=300)`	Función de optimización mediante PSO.

`  rectangles = []`
`  for i, s in enumerate(spaces):`
`  w = xopt[i] * s[0]`
`  h = xopt[i] * s[1]`
`  rectangles.append([i+1, s[0], s[1], w, h])`
	Función de transformar resultado en rectángulos.

`  no_overlapping_rectangles = adjust_positions(rectangles)` `  return no_overlapping_rectangles`	Llamada a la función de ubicación de espacios y de superposición.

`def adjust_positions(rectangles):`
`  for i in range(len(rectangles)):`
`  for j in range(i + 1, len(rectangles)):`
`  if is_overlapping(rectangles[i], rectangles[j]):`
`  rectangles[j][1] = rectangles[i][1] + rectangles[i][3]`
`  return rectangles`
	Función de ubicación de espacios y de superposición. Itera las posiciones y, si se superponen (*) Llamada a variable de comprobación de superposición. (*) las mueve.

`def is_overlapping(rect1, rect2):`
`  x1, y1, w1, h1 = rect1[1:]`
`  x2, y2, w2, h2 = rect2[1:]`
`  return x2 < x1 + w1 and x2 + w2 > x1 and y2 < y1 + h1 and y2 + h2 > y1`
	Función de comprobación si dos rectángulos se superponen.

`if __name__ == '__main__':`
`  spaces = read_spaces('Dimensiones.csv')`
`  packed_opt_rectangles = optimize_spaces(spaces)`
`  W = max([r[1] + r[3] for r in packed_opt_rectangles])`
`  H = max([r[2] + r[4] for r in packed_opt_rectangles])`
	Operación principal.
	Llamada a función de leer datos.
	Llamada a función de optimizar.
	Obtención de datos.

`  fig, ax = plt.subplots()` `  for r in packed_opt_rectangles:`	Generación de rectángulos para salida gráfica.

Código Python	Explicación de código

`ax.add_patch(plt.Rectangle((r[1], r[2]), r[3], r[4], fill=False))`
`ax.annotate(str(r[0]), (r[1]+r[3]/2, r[2]+r[4]/2), color='r', weight='bold', fontsize=12, ha='center', va='center')`
`plt.xlim(-10, W+10)`
`plt.ylim(-10, H+10)`
`plt.show()`
	Genera salida gráfica.

En el anejo n.º 1 de esta memoria se ha incluido el código de optimización basado en PSO explicado que se ha logrado abordar en este proyecto pero que no se ha logrado desarrollar al completo.

No se ha desarrollado al completo este algoritmo basado en PSO debido a la alta complejidad matemática, los altos requerimientos de conocimiento de programación y las demandas de prestaciones computacionales. Considerando el alcance del proyecto y las bases sobre las que se ha planteado, se ha optado por abandonar esta metodología en pro de desarrollar una alternativa que simplifique el resultado final para con los objetivos propuestos.

5.1.4 OBTENCIÓN DE RESULTADOS

Como se ha indicado en el planteamiento de la solución, el resultante del algoritmo de optimización debía basarse en un archivo Excel (extensión .xlsx) con las coordenadas de los espacios optimizados.

Adicionalmente, en el código desarrollado se ha creído conveniente mostrar los resultados de forma gráfica y numérica a través de una ventana generada también en Python.

El archivo Excel con los datos de salida de las coordenadas es simple pero funcional.

La información contenida corresponde a los vértices de cada rectángulo, por tanto, se obtendrán cuatro nodos por cada espacio. La primera columna alberga la coordenada X, mientras que la segunda la Y.

Cabe recordar que se ha trabajado en unidades de metros (m), tomando como eje de coordenadas el punto (0,0). Asimismo, se toma la convención de que, la configuración mostrada en 2D, sea tomada desde la vista inferior del modelo, de forma que lo visualizado en dos dimensiones en la salida del optimizador, corresponda con la vista inferior del futuro modelo BIM en 3D. Se trata de una asunción realizad en este proyecto de forma deliberada.

La siguiente figura muestra el aspecto con resultados obtenidos. El nombre del archivo es Coordenadas_Dynamo.xlsx.

	A	B
1	70	38
2	75	38
3	75	41
4	70	41
5	80	5
6	100	5
7	100	35
8	80	35
9	50	55
10	65	55

Ilustración 14. Archivos Excel de salida de datos del algoritmo.

Asimismo, la información gráfica generada en Python es la siguiente:

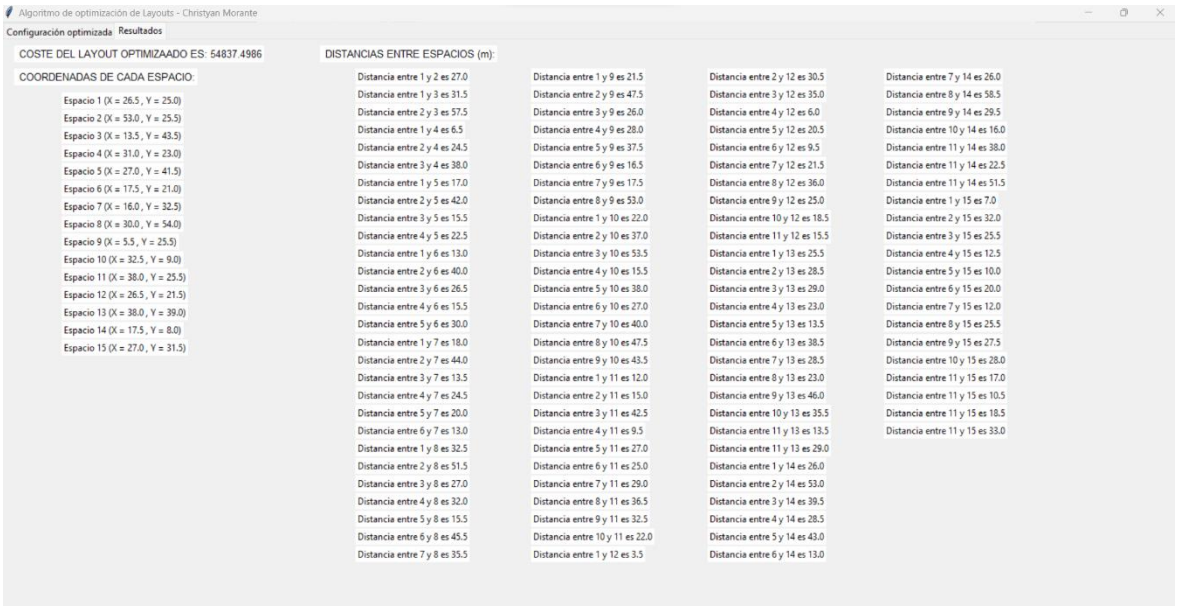

Ilustración 15. Salida numérica de datos del algoritmo.

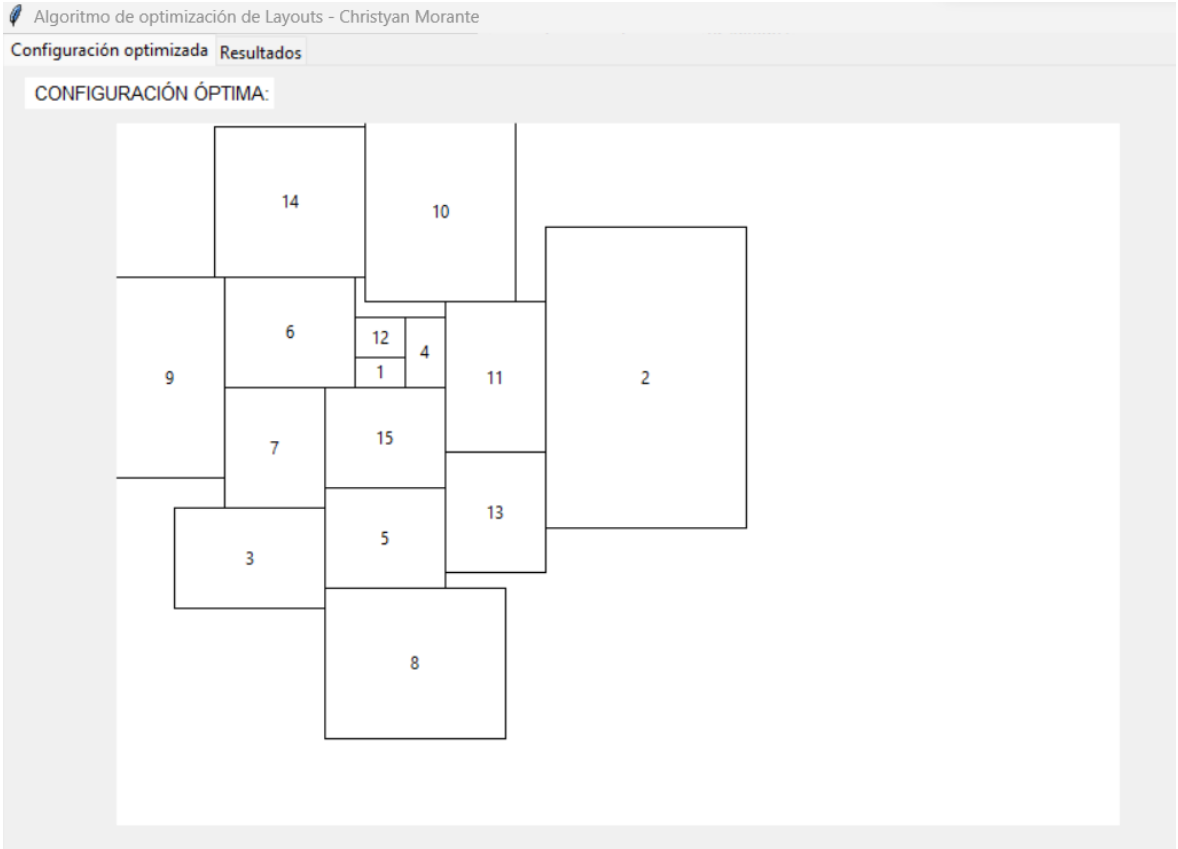

Ilustración 16. Salida gráfica de datos del algoritmo.

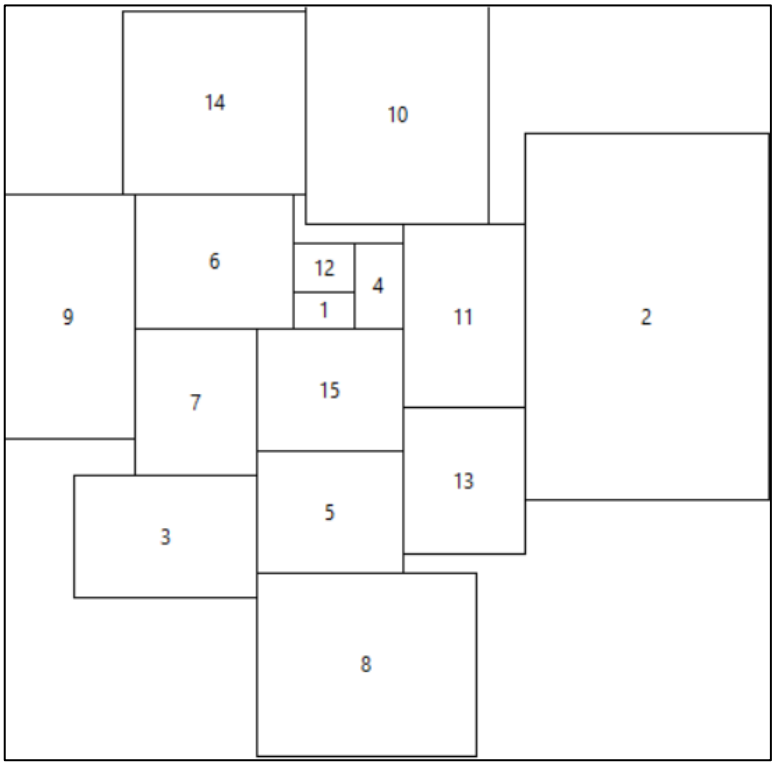

Ilustración 17. Detalle de salida gráfica de datos del algoritmo.

Alternativa experimental

Como se ha explicado en el apartado previo, se llevó a cabo el desarrollo experimental de un algoritmo basado en optimización PSO, el cual se abandonó por la complejidad y los altos requerimientos que suponía.

La salida de datos que se logró generar está basada en un visor de Python, tal y como se puede apreciar en la siguiente imagen, la cual incluye una configuración resultante de optimización de espacios.

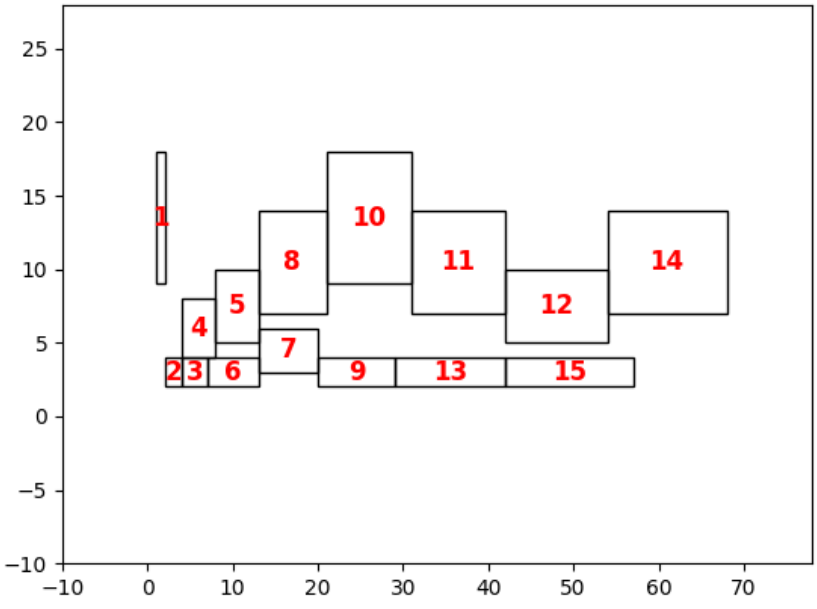

Ilustración 18. Ejemplo de optimización en PSO (elaboración propia).

Tal y como se ha explicado en la definición de esta metodología, el proceso se vuelve más complejo al integrar restricciones.

En el caso de este proyecto, se introdujo la variable de '*afinidad*' de forma que el modelo debía optimizar el espacio considerando la afinidad entre ellos y el menor uso de la superficie.

La adición de esta restricción entraba en contradicción con el mecanismo que evitaba que los espacios se superpusieran. El mencionado mecanismo, evaluaba si dos espacios estaban superpuestos y, en caso de que lo estuvieran, desplazaba uno de ellos a la derecha, cosa que, se deduce, afectaría a la restricción de afinidad.

Finalmente, dada la complejidad del modelo matemático, se descartó esta vía para la optimización considerando que los objetivos del proyecto no pretendían abarcar una optimización exhaustiva considerando que se asumía el posterior retrabajo de la información en el modelo BIM.

5.1.5 EVALUACIÓN DE RESULTADOS

Con objeto de evaluar el resultado que se obtiene del algoritmo de optimización, se llevan a cabo diferentes pruebas.

Todas ellas partirán de los datos de entrada del problema a resolver en este proyecto.

De esta forma, se plantean los casos de optimización de 5, 10 y 15 espacios, siendo el último el caso planteado como problema en el proyecto.

La tabla siguiente resume los espacios a analizar:

Tabla 7. Espacios de pruebas.

	CASO 1		**CASO 2**		**CASO 3** *(Problema de proyecto)*	
ID	**Ancho (m)**	**Largo (m)**	**Ancho (m)**	**Largo (m)**	**Ancho (m)**	**Largo (m)**
1	5	3	5	3	5	3
2	20	30	20	30	20	30
3	15	10	15	10	15	10
4	4	7	4	7	4	7
5	12	10	12	10	12	10
6			13	11	13	11
7			10	12	10	12
8			18	15	18	15
9			11	20	11	20
10			15	18	15	18
11					10	15
12					5	4
13					10	12
14					15	15
15					12	10

Se deben considerar las siguientes premisas en la realización de las pruebas:

- No se pretende obtener la configuración óptima a nivel funcional, sino con las condiciones matemáticas empleadas.
- No se tiene en cuenta el uso de cada espacio. Esta variable queda intrínseca en los datos de entrada de coste de rutas.
- La configuración obtenida no se trata de un diseño definitivo a construir, sino sienta unas bases para el desarrollo de un posterior trabajo más complejo.

Las prestaciones del equipo que utiliza el programa son las mostradas en la siguiente captura del terminal de ejecución de Visual Studio Code:

```
CPU model: Intel(R) Core(TM) i7-8750H CPU @ 2.20GHz, instruction set [SSE2|AVX|AVX2]
Thread count: 6 physical cores, 12 logical processors, using up to 12 threads
```

Ilustración 19. Datos del equipo utilizado para procesar el algoritmo.

CASO 1:

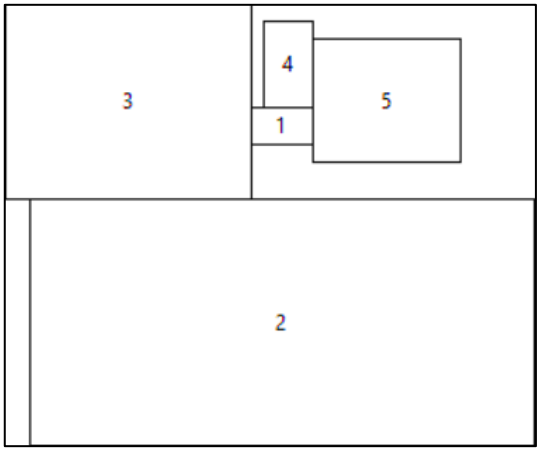

Ilustración 20. Configuración optimizada – Prueba 1.

Los datos obtenidos son los siguientes:

- Coste total: 2.273,00
- Centros de espacios:
  - Espacio 1: (30, 10)
  - Espacio 2: (30, 26)
  - Espacio 3: (17.5, 8)
  - Espacio 4: (30.5, 5)
  - Espacio 5: (38.5, 8)

Según el terminal de Visual Studio Code, se han analizado 108 nodos y se han realizado 1609 iteraciones, alcanzando el gap de 0,00% en menos de 1 segundo.

CASO 2:

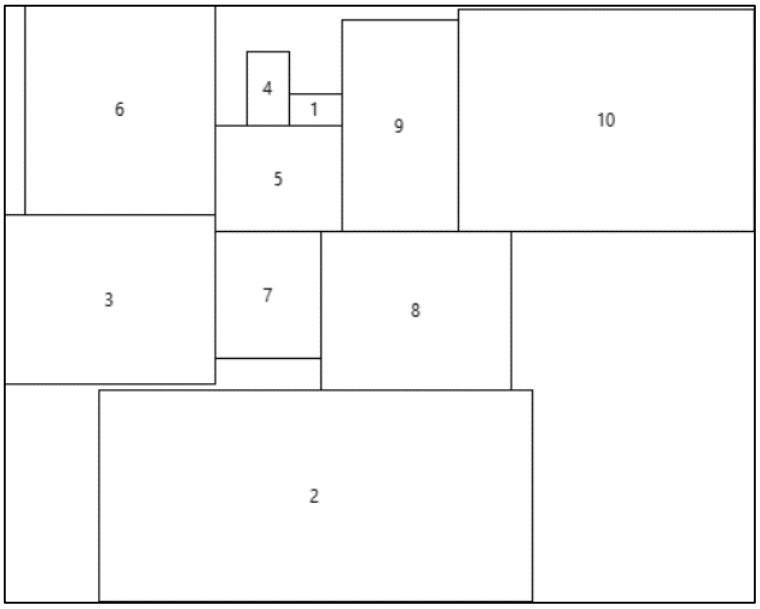

Ilustración 21. Configuración optimizada – Prueba 2.

Los datos obtenidos son los siguientes:

- Coste total: 20.116,99
- Centros de espacios:
  - Espacio 1: (30, 10)
  - Espacio 2: (30, 46.5)
  - Espacio 3: (10.5, 28)
  - Espacio 4: (25.5, 8)
  - Espacio 5: (26.5, 16.5)
  - Espacio 6: (11.5, 10)
  - Espacio 7: (25.5, 27.5)
  - Espacio 8: (39.5, 29)
  - Espacio 9: (38, 11.5)
  - Espacio 10: (57.5, 11)

Según el terminal de Visual Studio Code, se han analizado 1.756.377 nodos y se han realizado 41.365.158 iteraciones. Después de varias pruebas, las variaciones del resultado eran prácticamente nulas pasados aproximadamente 1800 segundos, asimilándose bastante a la configuración definitiva. El gap alcanzado tras este periodo de tiempo es del 14,19%.

CASO 3:

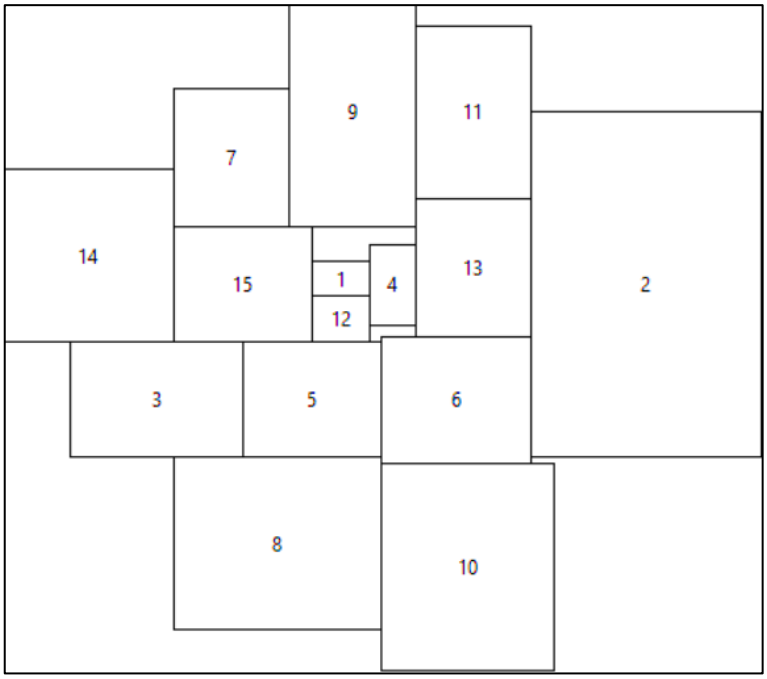

Ilustración 22. Configuración optimizada – Problema de proyecto.

Los datos obtenidos son los siguientes:

- Coste total: 54.876,99
- Centros de espacios:
    - Espacio 1: (29.5, 24.5)
    - Espacio 2: (56, 25)
    - Espacio 3: (13.5, 35)
    - Espacio 4: (34, 25)
    - Espacio 5: (27, 35)
    - Espacio 6: (39.5, 35)
    - Espacio 7: (20, 14)
    - Espacio 8: (24, 47.5)
    - Espacio 9: (30.5, 10)
    - Espacio 10: (40.5, 49.5)
    - Espacio 11: (41, 10)
    - Espacio 12: (29.5, 28)
    - Espacio 13: (41, 23.5)
    - Espacio 14: (7.5, 22.5)
    - Espacio 15: (21, 25)

Según el terminal de Visual Studio Code, se han analizado 938.341 nodos y se han realizado 49.468.287 iteraciones. Tras aproximadamente 3600 segundos, las variaciones

del gap resultaban prácticamente nulas, por lo que el resultado era prácticamente igual al obtenido finalmente. El gap alcanzado tras este tiempo es del 51,71%.

Con las pruebas realizadas, se aprecia que para configuraciones más simples el programa halla la solución matemática en menor tiempo, mientras que para configuraciones de más espacios el tiempo aumenta debido al aumento de posibles soluciones que evalúa.

También es apreciable que, en configuraciones más exigentes, difícilmente se alcanza la solución óptima global de gap igual a 0,00%. Esto es debido a varios factores como la configuración matemática propuesta, a la estructura de programación realizada, a las prestaciones y rendimiento del equipo, entre otras.

No obstante, se ha demostrado que los resultados hallados con considerablemente buenos y dan cumplimiento al objeto de este desarrollo: sentar una configuración de espacios lo más óptima posible para conseguir un modelo BIM posteriormente.

Como líneas de desarrollo y mejora, en base a lo analizado, se puede investigar una propuesta matemática más compleja o el uso de mejores equipos.

5.2 DESARROLLO DE MODELO BIM

Con la simulación realizada, los datos de los nodos que componen cada espacio se almacenan en un archivo Excel, nombrado *Coordenadas_Dynamo.xlsx*, tal y como se había comentado previamente.

En el caso del problema del proyecto, los datos obtenidos son los indicados en la siguiente tabla. Cabe recordar que la tabla de salida almacenada todos los datos en las dos primeras columnas, pero para favorecer la presentación de este documento se ha dispuesto de la siguiente forma:

Tabla 8. Datos de salida de configuración óptima.

X	Y	X	Y	X	Y	X	Y	X	Y
27	23	32	21,5	15	8	33	40,5	36	17,5
32	23	36	21,5	25	8	48	40,5	46	17,5
32	26	36	28,5	25	20	48	58,5	46	29,5
27	26	32	28,5	15	20	33	58,5	36	29,5
46	10	21	30	15	40	36	2,5	0	15
66	10	33	30	33	40	46	2,5	15	15
66	40	33	40	33	55	46	17,5	15	30
46	40	21	40	15	55	36	17,5	0	30
6	30	33	29,5	25	0	27	26	15	20
21	30	46	29,5	36	0	32	26	27	20
21	40	46	40,5	36	20	32	30	27	30
6	40	33	40,5	25	20	27	30	15	30

Cabe destacar que, de cara a abordar la siguiente fase, en la que se utilizan estas coordenadas para dar pie a un modelo BIM en formato 3D, se toman como referencia los puntos generados y la configuración obtenida en dos dimensiones como una vista desde el interior del suelo. Se trata de una convención tomada de forma independiente en este proyecto.

5.2.1 PROGRAMACIÓN VISUAL EN DYNAMO

Siguiendo el flujo de información expuesto en apartados anteriores, se programa en Dynamo el algoritmo para exportar los datos de salida del optimizador. Estos datos, interpretados por Dynamo, servirán de base para generar un modelo BIM preliminar que, posteriormente, se retrabajará adecuando lo obtenido en el optimizador según considere el equipo de diseño.

La versión de Dynamo utilizada es la incluida en la versión 2023 de Revit. Según la información extraída del software, se utiliza la siguiente versión, indicada la versión Core y la de Revit:

- Versión Dynamo Core: 2.13.1.3887
- Versión Dynamo Revit: 2.13.1.3891

La programación en Dynamo se realiza de forma visual, a partir de nodos que se unen y transfieren datos entre ellos.

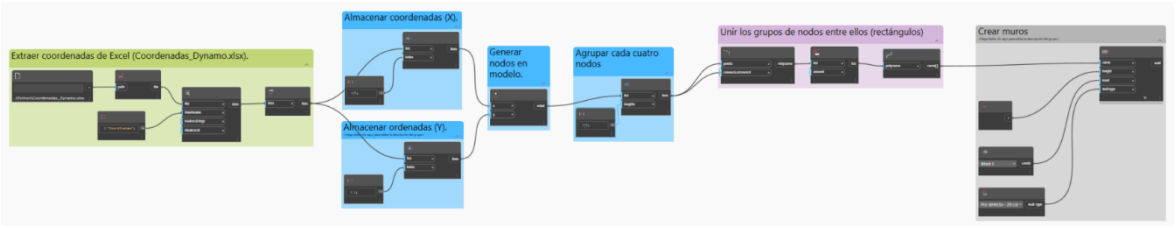

Ilustración 23. Vista de programación el Dynamo.

Los nodos se han agrupado para facilitar la explicación de la programación. A continuación, se exponen los grupos configurados.

5.2.1.1 EXTRAER COORDENADAS

Se usa el nodo *File Path* para indicar al programa la ruta donde se encuentra el archivo que debe leer. Este nodo se vincula al nodo *File From Path* que crea un objeto en Dynamo con la información dada.

Se vincula la salida al nodo *Data.ImportExcel* en el nodo 'file'. También, mediante un nodo *Code Block* se indica el nombre de la hoja a leer. El nodo *Code Block* se programa con el dato en tipo *string* e indica el nombre de la hoja "Coordinates". Se vincula en la entrada 'sheetName' del nodo *Data.ImportExcel.*

Este nodo lee los datos de las filas y los agrupa. Sin embargo, en el caso del presente proyecto, se desea leer los datos por columnas. Por ello, la salida de este nodo se lleva a la entrada del nodo *List.Transpose.* De este modo, se obtienen dos listados, uno de las coordenadas y otro de las ordenadas.

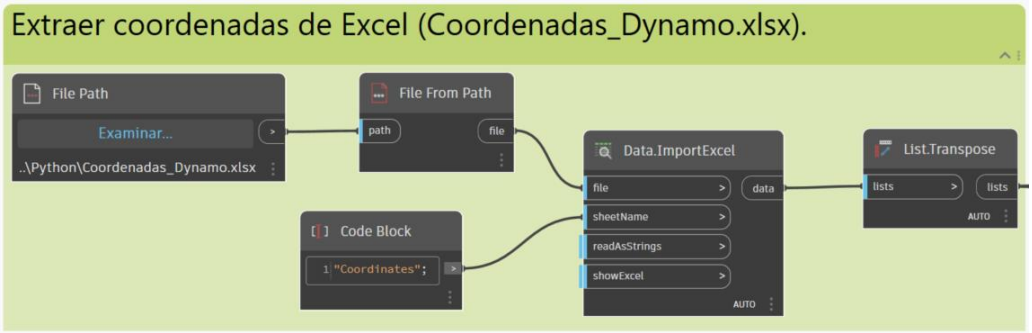

Ilustración 24. Programación en Dynamo. Extracción de coordenadas.

5.2.1.2 GENERACIÓN DE PUNTOS Y AGRUPACIÓN

Con las coordenadas disponibles en listas, se independizan las listas en dos salidas diferentes. Para ello, se utiliza el nodo *List.GetItemAtIndex*, el cual, indicándole el número de lista mediante un nodo *Code Block,* lee la lista del nodo previo.

Se generan dos grupos independiente de igual estructura, uno para los valores de X y otro para los valores de Y. Esto es necesario realizarlo porque el nodo *Point.ByCoordinates*, que permite la generación de los puntos en el espacio tridimensional, requiere de dos entradas independientes. Dado que no se especifican datos de altura, es decir, eje Z, los nodos se generarán con un valor de Z igual a 0.

Se crean los puntos en el espacio según las coordenadas especificadas, tal y como se aprecia en la siguiente imagen:

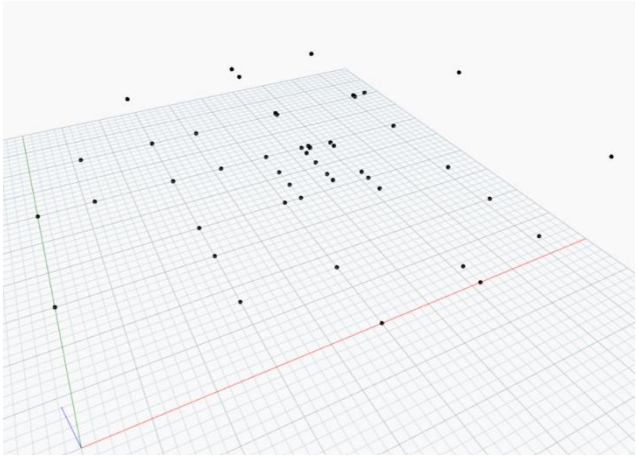

Ilustración 25. Puntos generados en espacio tridimensional.

Tras ello, se requiere agrupar los puntos de cuatro en cuatro pues en el archivo Excel original, estos se ordenan por cada rectángulo generado por el código en Python.

Para ello, se lleva la salida del nodo *Point.ByCoordinates* al nodo *List.Chop*, al cual se le indica, mediante un nodo *Code Block,* que debe agrupar el listado de puntos de cuatro en cuatro.

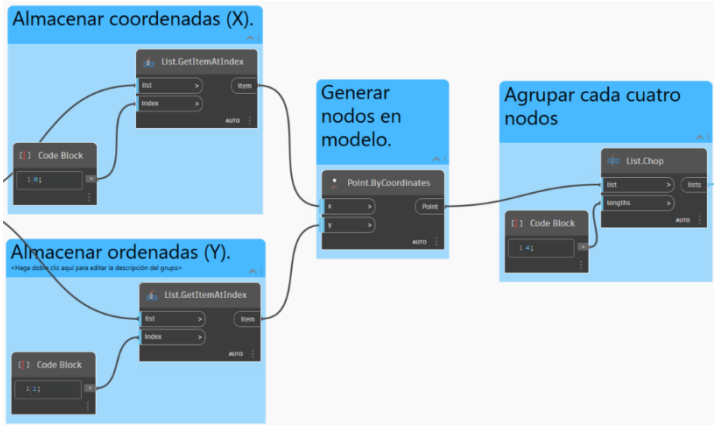

Ilustración 26. Programación en Dynamo. Generación de puntos y agrupación.

5.2.1.3 UNIÓN DE PUNTOS Y GENERACIÓN DE LINEAS

Con los puntos agrupados en listas, se recogen los puntos y las listas en el nodo *PolyCurve.ByPoints*. Cabe destacar que la salida del nodo *List.Chop* sirve como las dos entradas que requiere el nodo *Point.ByCoordinates.*

La salida del último nodo continúa agrupando las líneas de cuatro en cuatro. Ahora se requiere extraer todas las líneas y generar una única. Para ello, antes, se utiliza el nodo

List.Flatten, el cual suprime un nivel de los listados quedando, en este caso, todas las curvas en el mismo nivel de grupo.

Posteriormente, se pasa al nodo *PolyCurve.Curves*, que genera una curva según las "policurvas" de entrada. Con ello, se obtienen las líneas que seguirán las paredes del modelo.

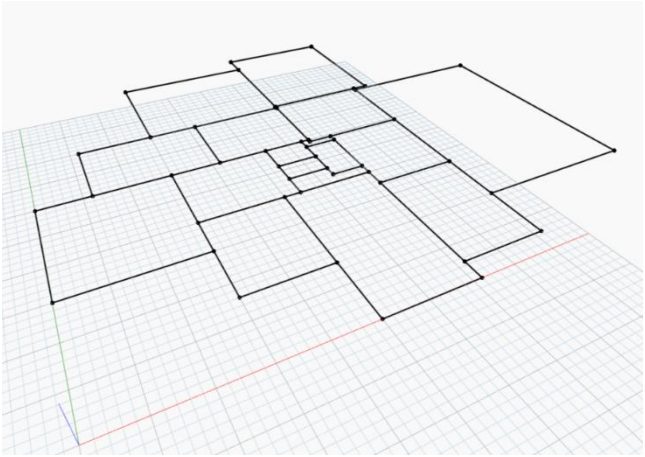

Ilustración 27. Líneas generadas en espacio tridimensional.

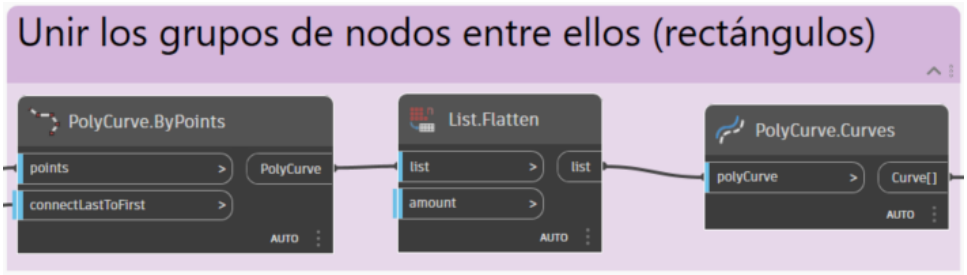

Ilustración 28. Programación en Dynamo. Unión de puntos y generación de líneas.

5.2.1.4 CREACIÓN DE MUROS

Con el dato de curvas generado, se lleva al nodo *Wall.ByCurveAndHeight*, mediante el cual se generará en el modelo de Revit las paredes que configuremos.

Este nodo requiere, además de las curvas, las variables de altura de las paredes, que se introducirá mediante un nodo *Number*; el nivel al que asociar las paredes, que se indicará mediante el nodo *Levels*, el cual detecta los niveles creados en el modelo de Revit para utilizarlo en la programación de Dynamo; y finalmente, el tipo de muro, que se indica mediante el nodo *Wall Types,* el cual lee las familias de muros que están configuradas en el archivo de Revit y permite su uso en Dynamo.

En este caso, se han establecido los siguientes valores de forma predeterminada para obtener la versión preliminar del modelo BIM:

- Altura: 3,50 m
- Nivel 1 (nivel de suelo)
- Tipo de muro: Muro por defecto – 20 cm de grosor.

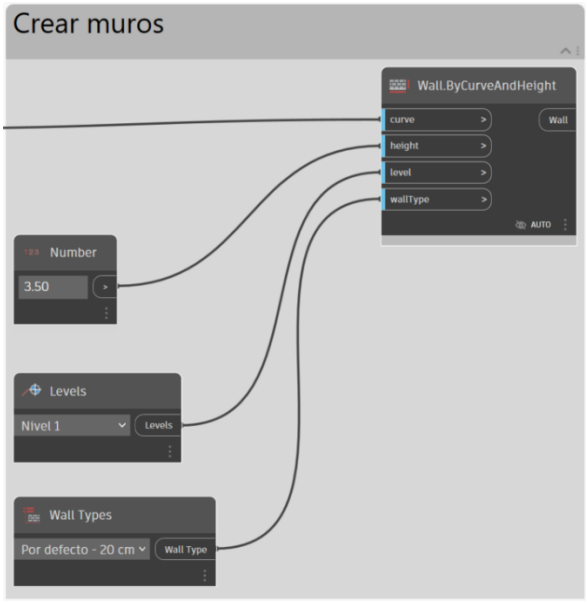

Ilustración 29. Programación en Dynamo. Creación de muros.

El resultado de ejecutar este grupo se aprecia en Revit.

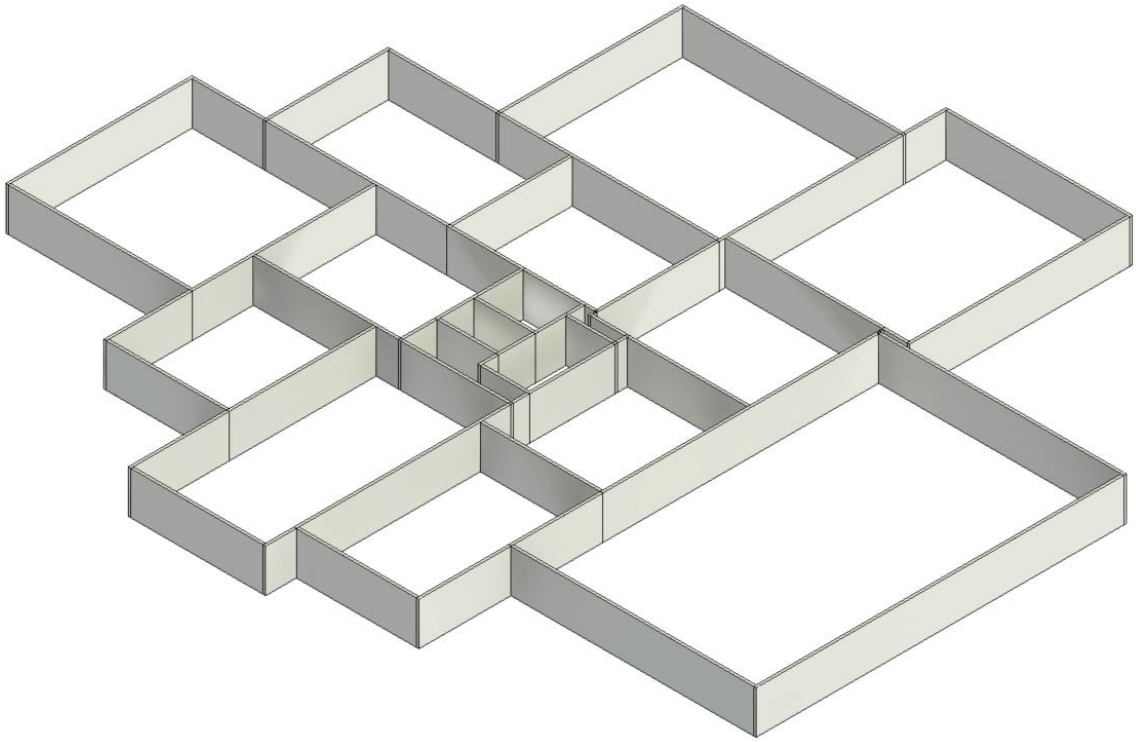

Ilustración 30. Muros del modelo BIM preliminar. Vista en 3D superior.

A continuación, se muestra la vista inferior del modelo que, como se informó previamente, debe ser la que corresponda con la visualización extraída del optimizador por convención tomada deliberadamente en este proyecto.

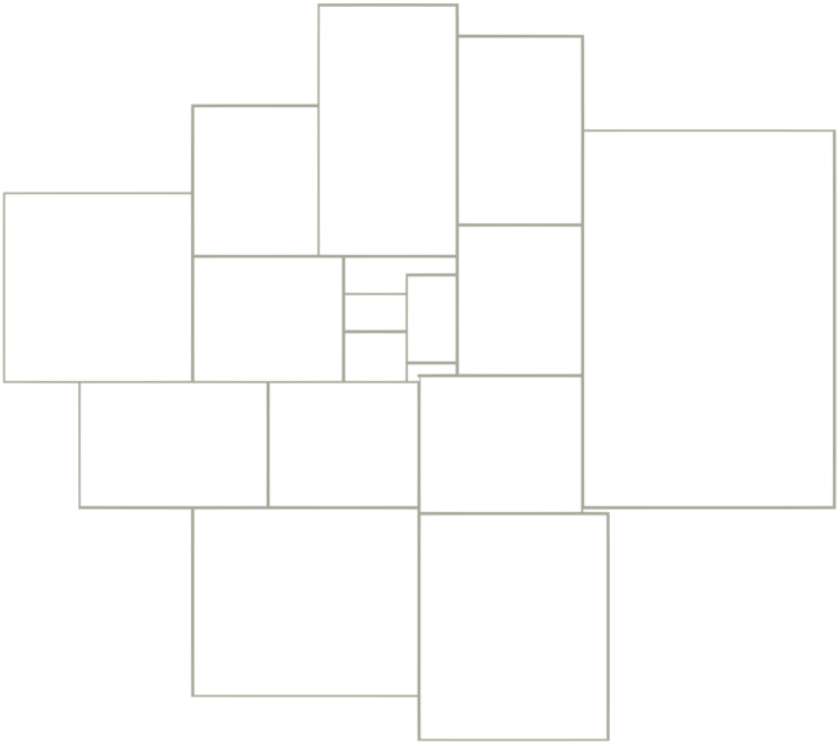

Ilustración 31. Muros del modelo BIM preliminar. Vista inferior.

5.2.2 POSPROCESADO DEL MODELO BIM

Una vez obtenido el primero modelo BIM preliminar, se procede al retrabajo del mismo para optimizarlo según las necesidades de la edificación que, en el caso específico de este proyecto, se trata del ala de emergencia de un hospital.

Primero de todo, se analizan los espacios y la configuración obtenida. Cabe recordar que el modelo obtenido está optimizado matemáticamente, y que no responde a consideraciones funcionales que se tengan. Dependerá de la formulación escogida el considerar ciertos aspectos. En cualquier caso, bajo consideración propia, se recomendaría revisar la configuración y realizar los cambios que se estimen necesarios.

5.2.2.1 RECONFIGURACIÓN DE ESPACIOS

Se realizan varias iteraciones para llegar al modelo que se considere definitivo.

Iteración 1:

La configuración obtenida del optimizador, se revisa y se modifica conforme a criterios que no han sido considerados en la formulación del problema.

Estos criterios han sido tomados deliberadamente por el autor según su criterio de funcionalidad y basado en la experiencia propia del uso de instalaciones hospitalarias. En ningún caso se ha llevado a cabo un estudio pormenorizado de la configuración de espacios que requiere un hospital dado que no forma parte del alcance del proyecto ni sus objetivos, habiendo asumido estos como datos de entrada.

Al realizar el posprocesado de la configuración, se ha mantenido la premisa incluida en el algoritmo del optimizador de mantener las dimensiones de los espacios fijas. Quedando la configuración de la siguiente manera:

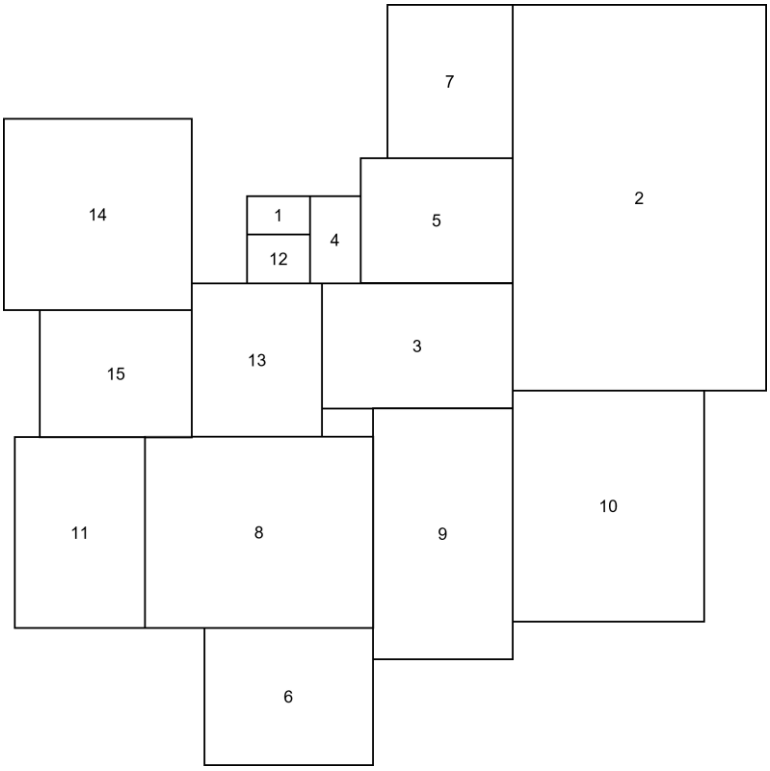

Ilustración 32. Configuración posprocesada en proyecto- Iteración 1.

Iteración 2:

En esta iteración se pretenden integrar los pasillos que deben incluirse en el diseño final. Para dimensionarlos, se ha tenido en cuenta lo dispuesto en el Código Técnico de la Edificación (CTE), Documento Básico de Seguridad en caso de Incendio (DB-SI). En concreto, se considera lo indicado en la tabla *4.1 Dimensionado de los elementos de evacuación*, que en su nota n.º 3 indica lo siguiente:

4.2 Cálculo

1 El dimensionado de los elementos de evacuación debe realizarse conforme a lo que se indica en la tabla 4.1.

Tabla 4.1 Dimensionado de los elementos de la evacuación

Tipo de elemento	Dimensionado
Puertas y pasos	$A \geq P / 200^{(1)} \geq 0{,}80\ m^{(2)}$ La anchura de toda hoja de puerta no debe ser menor que 0,60 m, ni exceder de 1,23 m.
Pasillos y rampas	$A \geq P / 200 \geq 1{,}00\ m^{(3)(4)(5)}$

(1) La anchura de cálculo de una puerta de salida del recinto de una *escalera protegida* a planta de *salida del edificio* debe ser al menos igual al 80% de la anchura de cálculo de la escalera.

(2) En *uso hospitalario* A ≥ 1,05 m, incluso en puertas de habitación.

(3) En *uso hospitalario* A ≥ 2,20 m (≥ 2,10 m en el paso a través de puertas).

Ilustración 33. Extracto de la tabla 4.1 del CTE DB-SI.

Por estar del lado de la seguridad, se han considerado pasillos de 2,50 m en la configuración siguiente:

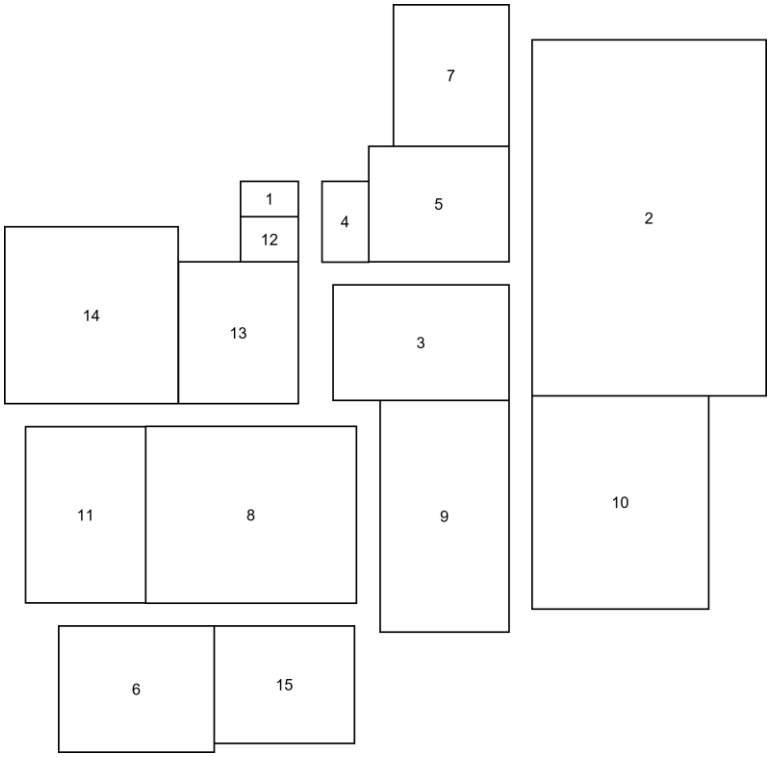

Ilustración 34. Configuración posprocesada en proyecto- Iteración 2.

Iteración 3:

Finalmente, para la iteración 3, primero se traslada la configuración al modelo de Revit ya que lo que se va a pretender es suavizar los contornos sin afectar negativamente al área del espacio, es decir, que si el área se modifica sea creciente.

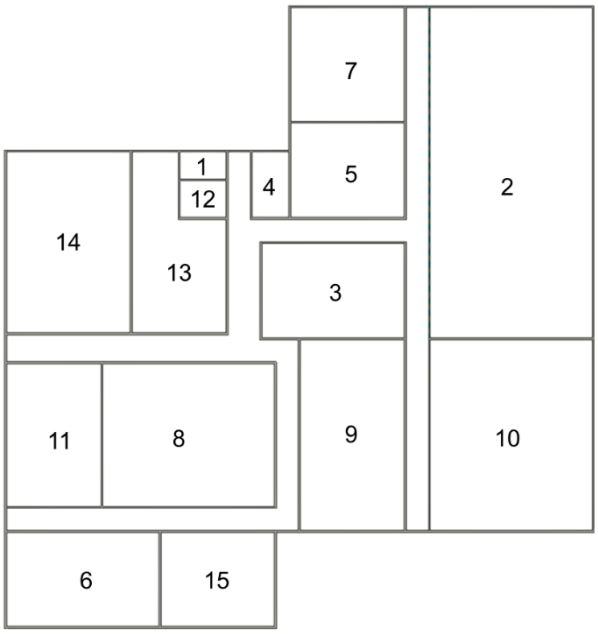

Ilustración 35. Configuración posprocesada en proyecto- Iteración 3.

Tras ello, se eliminan los muros de las zonas que se consideran que deberían quedar diáfanas por su funcionalidad, que en este caso se ha optado por eliminar los muros de las salas de espera.

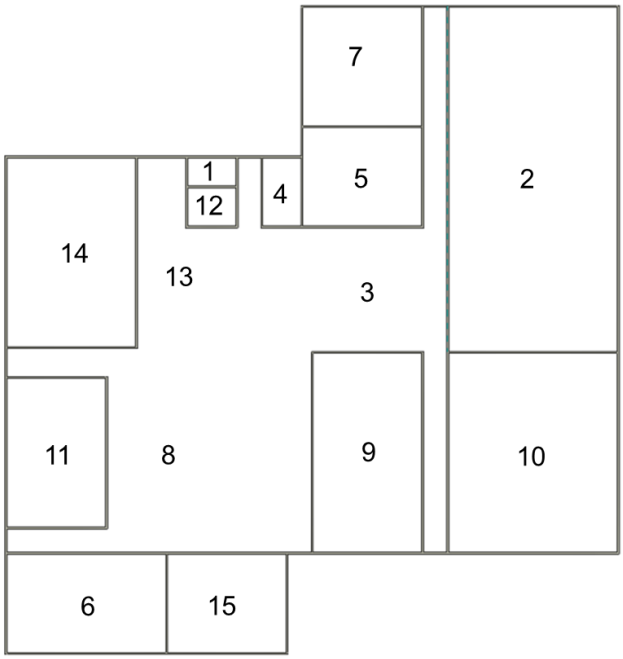

Ilustración 36. Configuración posprocesada en proyecto- Iteración 3, espacios diafanos.

5.2.2.2 CREACIÓN DE ESPACIOS

Con la configuración posprocesada, se plantea el modelo BIM de espacios que quedará como definitivo del presente proyecto.

Para finalizarlo, se definen los límites de las habitación o espacios que contiene el modelo. También se corrigen aspectos de detalle como son las uniones entre muros, de forma que no queden configuraciones erráticas. En el siguiente esquema se ejemplifica este proceso.

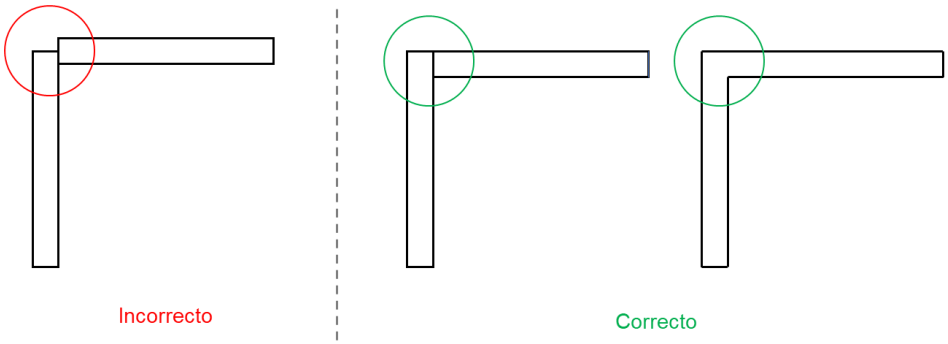

Ilustración 37. Esquema de uniones entre muros.

Los espacios se crean mediante la herramienta *Habitaciones* de Revit y, tras la asignación, se genera una vista en planta vista desde la parte superior con las dependencias finales identificadas.

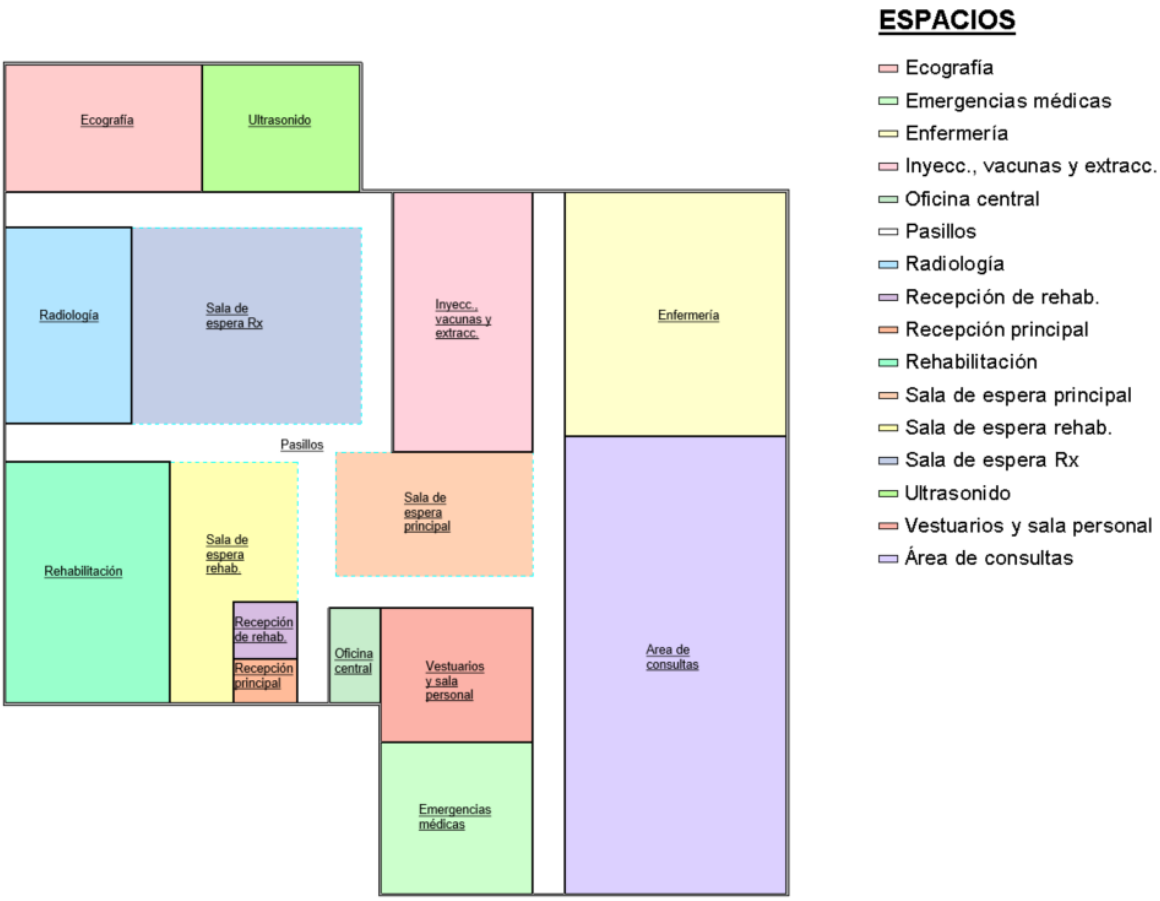

Ilustración 38. Configuración final en modelo BIM. Vista en planta.

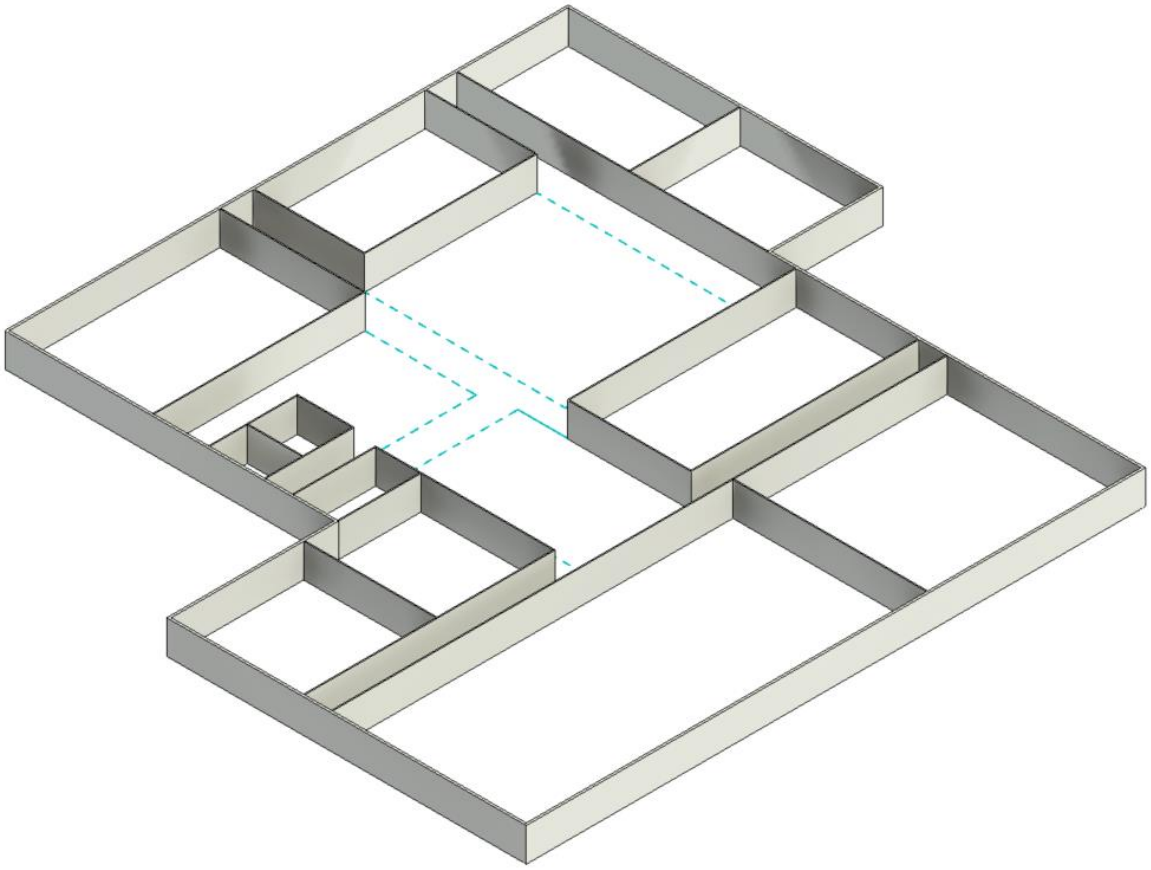

Ilustración 39. Configuración final en modelo BIM. Vista en 3D.

En el alcance del presente proyecto no se ha incluido el definir la ubicación de accesos ni mobiliario o cualquier elemento arquitectónico adicional. Con la configuración de espacios optimizada se da por finalizado el modelo BIM preliminar objeto de este proyecto.

5.2.2.3 COMPROBACIONES DE REQUERIMIENTOS DE ESPACIOS

Tras definir la configuración final, se debe validar que el planteamiento es el adecuado para con los requisitos establecidos en los datos de entrada del problema de proyecto.

En este caso, se toma como referencia el área que debía ocupar cada espacio con las dimensiones proporcionadas.

En la siguiente figura se observa la salida de Revit del listado de habitaciones con la superficie final de cada una.

<Listado de espacios>

A	B	C
Número	Nombre	Área
2	Área de consultas	622.00 m²
10	Enfermería	331.39 m²
9	Inyecc., vacunas y extracc.	221.82 m²
15	Ultrasonido	122.27 m²
6	Ecografía	151.97 m²
11	Radiología	150.71 m²
14	Rehabilitación	242.22 m²
12	Recepción de rehab.	21.56 m²
1	Recepción principal	16.42 m²
4	Oficina central	29.03 m²
5	Vestuarios y sala personal	123.24 m²
7	Emergencias médicas	140.42 m²
8	Sala de espera Rx	276.43 m²
13	Sala de espera rehab.	147.91 m²
3	Sala de espera principal	150.58 m²
16	Pasillos	430.28 m²
Total general: 16		3178.22 m²

Ilustración 40. Listado de áreas finales.

En la tabla siguiente se incluye la comparativa de las áreas requeridas en los datos de entrada y las definitiva. En ella, se puede observar que se ha cumplido el requisito en todos los casos.

Tabla 9. Tabla de comprobación de cumplimiento de áreas mínimas.

ID	Nombre de espacio	Área mínima (m^2)	Área final (m^2)	CUMPLE/NO CUMPLE
1	Recepción principal	15	16,42	**CUMPLE**
2	Área de consultas	600	622,00	**CUMPLE**
3	Sala de espera principal	150	150,58	**CUMPLE**
4	Oficina central	28	29,03	**CUMPLE**
5	Vestuarios y sala de personal	120	123,24	**CUMPLE**
6	Ecografías	143	151,97	**CUMPLE**
7	Emergencias médicas	120	140,42	**CUMPLE**
8	Sala de espera Rx	270	276,43	**CUMPLE**
9	Inyecciones, vacunas y extracciones	220	221,82	**CUMPLE**
10	Enfermería	270	331,39	**CUMPLE**

ID	Nombre de espacio	Área mínima (m^2)	Área final (m^2)	CUMPLE/NO CUMPLE
11	Radiología	150	150,71	**CUMPLE**
12	Recepción de rehabilitación	20	21,56	**CUMPLE**
13	Sala de espera rehabilitación	120	147,91	**CUMPLE**
14	Rehabilitación	225	242,22	**CUMPLE**
15	Ultrasonidos	120	122,27	**CUMPLE**

6 RESUMEN DEL PRESUPUESTO

Conforme a los objetivos del proyecto y a su alcance, el presupuesto por la ejecución del mismo, considerando los gastos generales y el beneficio industrial, sin IVA asciende a **VEINTIDÓS MIL CIENTO OCHENTA Y OCHO EUROS CON DOS CÉNTIMOS (22.188,02 €).**

Tabla 10. Resumen del presupuesto.

Concepto	Importe
Presupuesto de Ejecución Material (PEM)	18.645,40 €
Presupuesto de Ejecución de Contrato (PEC) sin IVA - 13 % Gastos generales - 6% Beneficio Industrial	22.188,02 €
Presupuesto de Ejecución de Contrato (PEC) con IVA	26.847,50 €

El detalle del presente presupuesto de desarrolla en el documento PRESUPUESTO.

7 ANÁLISIS Y VALORACIÓN DE LAS IMPLICACIONES AMBIENTALES Y SOCIALES

Las implicaciones ambientales y sociales que aborda un proyecto BIM son fundamentales para asegurar el desarrollo sostenible y responsable de la industria de la construcción.

Es importante que un proyecto de este estilo considere aspectos tan importantes como los siguientes:

- La eficiencia energética
- El uso sostenible de materiales
- La reducción de emisiones de gases de efecto invernadero
- La gestión adecuada del agua
- La gestión sostenible de residuos
- El impacto social que incumbe un proyecto BIM (empleabilidad, seguridad y calidad de vida, preservación del patrimonio, etc.)
- Accesibilidad e igualdad de condiciones

Por supuesto que, en el caso de desarrollar un proyecto BIM en su totalidad, los puntos a considerar se adaptarían y evolucionarían de forma específica según las necesidades y el impacto de cada proyecto.

En las líneas abordadas en este trabajo se influye directamente sobre el aspecto de 'eficiencia energética' ya que, mediante el cumplimiento de los objetivos del proyecto se propone la optimización de tareas, que a su vez implica en una reducción de tiempo y recursos necesarios para llevar a cabo un mismo trabajo, lo que implica un uso más sostenible de la energía.

Asimismo, centrando el foco en el problema planteado en el proyecto, el llevar a cabo una optimización de espacios adecuada en un complejo hospitalario afecta también directamente en la seguridad y calidad de vida de las personas, considerando también aspectos de accesibilidad e igualdad de condiciones para todos los usuarios.

En conclusión, se destaca la contribución que realiza el desarrollo de este proyecto en cuanto a la eficiencia energética al reducir tiempos de trabajo y, dentro del propio desarrollo de un proyecto BIM de una instalación hospitalaria, se destaca, entre otros, el impacto directo sobre la calidad de vida de las personas, su accesibilidad y seguridad.

8 CONCLUSIONES

Tras la realización del presente proyecto, se concluye que la herramienta de optimización y automatización desarrollada cumple con los objetivos del proyecto y con los requisitos que se habían impuesto.

Se ha logrado desarrollar una herramienta de optimización matemática de espacios, considerando unos datos de entrada concretos, dando como resultado una configuración optimizada matemáticamente.

La formulación matemática adoptada ha sido la necesaria para dar cumplimiento a los objetivos del proyecto, si bien se tiene conciencia de que esta puede mejorarse según las necesidades de otros proyectos o el número de condicionantes adicionales que se pretendan considerar.

El flujo de información tratado con el fin de integrar los datos optimizados en un modelo BIM cumple con la idea de partida, llegando al detalle y alcance especificado al inicio del proyecto, llegando al modelo objetivo que se pretendía, donde se sientan las bases de diseño para un futuro desarrollo más detallado de la edificación.

El modelo BIM resultante, tras el posprocesado del autor del proyecto, da cumplimiento a los requerimientos de superficies mínima que se establecían en los datos de entrada, quedando patente que la metodología seguida y que se propone en este trabajo es correcta y aplicable a un proyecto de estas características.

Se destaca también que el modelo conseguido es integrable en varias de las alternativas de mercado analizadas para el trabajo colaborativo. En concreto, el modelo BIM obtenido se puede trabajar en la plataforma Omniverse dado que el software utilizado y con el que se ha desarrollado el modelo BIM es compatible con esta plataforma.

Para finalizar, se consideran demostradas las ventajas que ofrece la automatización de procesos en la metodología BIM, especialmente en la fase de diseño donde mayores iteraciones se llevan a cabo.

Trabajos futuros

A continuación, se plantean varios trabajos a llevar a cabo como continuación del proyecto abordado:

- **Desarrollo matemático del algoritmo de optimización**

  Si bien la solución planteada en este proyecto cumple con los requisitos y objetivos del mismo, se propone llevar a cabo un estudio pormenorizado en el que se desarrolle una siguiente versión del algoritmo de optimización empleando modelos matemáticos más avanzados, como el basado en PSO y que se abordó de forma experimental en este proyecto.

  Se deberán incluir restricciones adicionales propias de las instalaciones hospitalarias, considerando también la normativa de edificación aplicable a las mismas y al proceso constructivo que se pretenda seguir (Código Técnico de la Edificación, Código Estructural, etc.).

  El estudio deberá incluir también el minado de datos y algoritmos de interpretación de los mismos, realizando un nuevo estudio de comportamiento de pacientes y las rutas propias que se dan en hospitales, o tomando como referencia estudios ya realizados como los que se incluyen en la bibliografía de este proyecto y que se han tomado como base del mismo.

- **Procesado detallado del modelo BIM**

  Como se ha comentado, el modelo BIM obtenido se toma como base para un futuro desarrollo que dé como fruto un modelo a construir y, posteriormente, un modelo que sirva para la explotación y mantenimiento de la instalación.

  Se propone desarrollar el modelo BIM obtenido a lo largo de las diferentes fases del proyecto, pasando por la obtención del proyecto de diseño definitivo, el proyecto de ejecución, la inclusión de la información en el modelo BIM relativa a las instalaciones, y llegando a utilizar la misma en la fase de operación y mantenimiento.

  El nivel de detalle de las diferentes fases se adaptará a las necesidades de la misma y se tomarán como referencia estándares aplicables del momento.

  Asimismo, se explorará la integración con nuevas tecnologías, como la realidad aumentada, para la visualización del modelo en fase de operación y mantenimiento, e incluso durante la fase de ejecución e instalaciones.

- **Integración del modelo en Omniverse**

  El presente proyecto ha dado como fruto un modelo BIM de base desarrollado en Revit, es decir, compatible con la integración en Omniverse. Tras analizar las ventajas del mismo, se propone llevar a cabo la integración en esta plataforma para evaluar las opciones que esta ofrece sobre un proyecto BIM.

  Se deberán realizar simulaciones de todo tipo, desde físicas hasta de comportamiento de usuarios basándose en bases de datos e históricos, con el objetivo de llegar a hacer, por ejemplo y de forma no limitativa, predicciones de ocupación, evaluaciones sobre rutas de evacuación, etc.

  La integración en Omniverse se realizará, como mínimo, para la fase de diseño, aunque se explorará la aplicación en futuras fases de las que comprende un proyecto BIM.

9 REFERENCIAS

Bibliografía

Alavi, H., & Forcada Matheu, N. (2022). Building information modeling for facility managers. TDX (Tesis Doctorals En Xarxa). https://www.tdx.cat/handle/10803/675747

An Introduction to Particle Swarm Optimization (PSO) Algorithm. (n.d.). Retrieved May 1, 2023, from https://www.analyticsvidhya.com/blog/2021/10/an-introduction-to-particle-swarm-optimization-algorithm/

Arnolds, I. V., & Gartner, D. (2018). Improving hospital layout planning through clinical pathway mining. Annals of Operations Research, 263(1–2), 453–477. https://doi.org/10.1007/S10479-017-2485-4/TABLES/8

Burkard, R., Dell'Amico, M., & Martello, S. (2012). Assignment Problems. Assignment Problems. https://doi.org/10.1137/1.9781611972238

Dueñas, F., Director, L., Jeffrey, & Madrid, H. (2019). "Dulle-Facility Layout Optimization and Evaluation Engine."

Halawa, F., Chalil Madathil, S., & Khasawneh, M. T. (2021). Integrated framework of process mining and simulation–optimization for pod structured clinical layout design. Expert Systems with Applications, 185, 115696. https://doi.org/10.1016/J.ESWA.2021.115696

Kennedy, J., & Eberhart, R. (n.d.). Particle swarm optimization. Proceedings of ICNN'95 - International Conference on Neural Networks, 4, 1942–1948. https://doi.org/10.1109/ICNN.1995.488968

Koopmans, T. C., & Beckmann, M. J. (n.d.). Assignment Problems and the Location of Economic Activities. Retrieved May 1, 2023, from https://elischolar.library.yale.edu/cowles-discussion-paper-series

Lawler, E. L. (1963). The Quadratic Assignment Problem. https://Doi.Org/10.1287/Mnsc.9.4.586, 9(4). https://doi.org/10.1287/MNSC.9.4.586

Lin, Z., & Yingjie, Z. (2019). Solving the Facility Layout Problem with Genetic Algorithm. 2019 IEEE 6th International Conference on Industrial Engineering and Applications, ICIEA 2019, 164–168. https://doi.org/10.1109/IEA.2019.8715148

Montreuil, B. (1991). A Modelling Framework for Integrating Layout Design and flow Network Design. 95–115. https://doi.org/10.1007/978-3-642-84356-3_8

Rismanchian, F., & Lee, Y. H. (2016). Process Mining–Based Method of Designing and Optimizing the Layouts of Emergency Departments in Hospitals. Https://Doi.Org/10.1177/1937586716674471, 10(4), 105–120. https://doi.org/10.1177/1937586716674471

Webgrafía

Autodesk | Software de diseño, ingeniería y construcción en 3D. (n.d.). Retrieved May 1, 2023, from https://www.autodesk.es/

GitHub: Let's build from here · GitHub. (n.d.). Retrieved May 1, 2023, from https://github.com/

IFC en español - BuildingSMART Spanish Chapter. (n.d.). Retrieved May 1, 2023, from https://www.buildingsmart.es/recursos/ifc-en-espa%C3%B1ol/

Learn - Dynamo BIM. (n.d.). Retrieved May 1, 2023, from https://dynamobim.org/

Portada | Comisión Interministerial BIM. (n.d.). Retrieved May 1, 2023, from https://cbim.mitma.es/

Python para principiantes - Training | Microsoft Learn. (n.d.). Retrieved May 1, 2023, from https://learn.microsoft.com/es-es/training/paths/beginner-python/?source=learn

The Leader in Decision Intelligence Technology - Gurobi Optimization. (n.d.). Retrieved May 1, 2023, from https://www.gurobi.com/

Visual Studio: IDE y Editor de código para desarrolladores de software y Teams. (n.d.). Retrieved May 1, 2023, from https://visualstudio.microsoft.com/es/

Welcome to Python.org. (n.d.). Retrieved May 1, 2023, from https://www.python.org/

ANEJO N. º 1 PROGRAMACIÓN RESULTANTE

1 CÓDIGO RESULTADO DE PROYECTO

```
import tkinter.ttk as ttk
from tkinter import *
from gurobipy import *
from gurobipy import GRB, Model, abs_, quicksum
from threading import Thread
from tkinter import Frame
from tkinter.font import Font
from openpyxl import Workbook, load_workbook
import csv

roota = Tk()
roota.geometry("1000x500")
roota.configure(background="white", bg='white')
roota.title("Algoritmo de optimización - Christyan Morante")

nb = ttk.Notebook(roota)
first = ttk.Frame(roota)
second= ttk.Frame(roota)
nb.add(first, text='Configuración optimizada')
nb.add(second, text='Resultados')
nb.pack(expand="1" , fill="both")

class LeeDimensiones():
    def __init__ (self, filename):
        with open (filename, 'r', encoding='utf-8') as f_input:
            csv_input = csv.reader(f_input)
            self.dimensiones = list(csv_input)
    def GetAncho (self):
        return self.dimensiones[row][1]
    def GetLargo (self,row):
        return self.dimensiones[row][2]

class LeeRutas():
    def __init__ (self, filename):
        with open (filename, 'r', encoding='utf-8') as f_input:
            csv_input = csv.reader(f_input)
            self.rutas = list(csv_input)
    def GetRutas (self, row , column):
        return self.rutas[row-1][column-1]

model = Model('Opt')
CR = {}
MRutas = {}
Anch = {}
Larg = {}
x = {}
y = {}
c1 = {}
c2 = {}
c3 = {}
c4 = {}
Dx = {}
Dy = {}
Dxabs = {}
Dyabs = {}
Dij = {}

InfoRutas = LeeRutas('Rutas.csv')
rows = len(InfoRutas.rutas)
NumEspacios = int(rows-2)

for row in range(1,NumEspacios+1):
    for column in range(1,NumEspacios+1):
        val = int(InfoRutas.rutas[row][column])
```

```
        MRutas[row,column] = val

for row in range(1,NumEspacios+1):
    for column in range(1,NumEspacios+1):
        CR[row,column] = MRutas[row,column]

xMAXIM = int(InfoRutas.rutas[rows-1][1])
yMAXIM = int(InfoRutas.rutas[rows-1][3])

InfoDimensiones = LeeDimensiones('Dimensiones.csv')

for row in range(1,NumEspacios+1):
    Anch[row] = float(InfoDimensiones.dimensiones[row][1])
    Larg[row] = float(InfoDimensiones.dimensiones[row][2])

for row in range(1,NumEspacios+1):
    x[row]= model.addVar(vtype=GRB.CONTINUOUS, name='x_{}'.format(row))
    y[row]= model.addVar(vtype=GRB.CONTINUOUS, name='y_{}'.format(row))

for row in range(1,NumEspacios+1):
    for column in range (1,NumEspacios+1):
        Dij[row,column] = model.addVar(vtype=GRB.CONTINUOUS, lb=-10000 , ub =1000,
name='d_{}{}'.format(row,column))
        Dx[row,column] = model.addVar(vtype=GRB.CONTINUOUS, lb=-10000 ,ub
=1000,name='d_x{}{}'.format(row,column))
        Dy[row,column] = model.addVar(vtype=GRB.CONTINUOUS, lb=-10000,ub
=1000,name='d_y{}{}'.format(row,column))
        Dxabs[row,column] = model.addVar(vtype=GRB.CONTINUOUS, lb=-10000,ub =
1000,name='d_x{}{}'.format(row,column))
        Dyabs[row,column] = model.addVar(vtype=GRB.CONTINUOUS, lb=-10000 ,ub
=1000,name='d_y{}{}'.format(row,column))

for row in range (1,NumEspacios+1):
    for column in range (1,NumEspacios+1):
        if column>row:
            c1[row,column] = model.addVar(vtype=GRB.BINARY,
name="s_1{}{}".format(row,column))
            c2[row,column] = model.addVar(vtype=GRB.BINARY,
name="s_2{}{}".format(row,column))
            c3[row,column] = model.addVar(vtype=GRB.BINARY,
name="s_3{}{}".format(row,column))
            c4[row,column] = model.addVar(vtype=GRB.BINARY,
name="s_4{}{}".format(row,column))

model.update()

for row in range(1,NumEspacios+1):
    model.addConstr( x[row]+(Anch[row]/2) <= xMAXIM)
    model.addConstr( x[row]-(Anch[row]/2) >= 0)
    model.addConstr( y[row]+(Larg[row]/2) <= yMAXIM)
    model.addConstr( y[row]-(Larg[row]/2) >= 0)

for row in range (1,NumEspacios+1):
    for column in range (1,NumEspacios+1):
        if(column>row):
            model.addConstr (Dx[row,column] == (x[row] - x[column]))
            model.addConstr (Dy[row,column] == (y[row] - y[column]))
            model.addConstr (Dxabs[row,column] == abs_(Dx[row,column]))
            model.addConstr (Dyabs[row,column] == abs_(Dy[row,column]))
            model.addConstr (Dij[row,column] == Dxabs[row,column] + Dyabs[row,column])
            model.addConstr((c1[row,column] == 1) >> (x[row] + (Anch[row]/2) <=
x[column] -(Anch[column]/2)))
            model.addConstr((c2[row,column] == 1) >> ( x[column]+(Anch[column]/2) <=
x[row]-(Anch[row]/2)))
```

```
            model.addConstr((c3[row,column] == 1) >> ( y[column]+(Larg[column]/2) <=
y[row]-(Larg[row]/2)))
            model.addConstr((c4[row,column] == 1) >> ( y[row]+(Larg[row]/2) <=
y[column]-(Larg[column]/2)))
            model.addConstr(c1[row,column] + c2[row,column] + c3[row,column] +
c4[row,column] >=1)

model.addConstr (x[1]==30)
model.addConstr (y[1]==10)

obj = quicksum ( Dij[row,column] * CR[row,column]
    for row in range (1,NumEspacios+1)
    for column in range (1,NumEspacios+1)
    if column>row
)
model.setObjective(obj, GRB.MINIMIZE)

model.Params.TimeLimit = 1800
model.Params.MipFocus=1
model.Params.BestObjStop=1500

model.optimize()
obj = model.ObjVal
obj = round(obj,4)
x_1 = model.getVarByName("x_1")
x_1 = round(x_1.X , 4)
y_1 = model.getVarByName("y_1")
y_1 = round (y_1.X , 4)
x_2 = model.getVarByName("x_2")
x_2 = round (x_2.X , 4)
y_2 = model.getVarByName("y_2")
y_2 = round (y_2.X , 4)
d_12 = model.getVarByName("d_12")
d_12 = round (d_12.X , 4)

if NumEspacios>2:
    x_3 = model.getVarByName("x_3")
    x_3 = round (x_3.X , 4)
    y_3 = model.getVarByName("y_3")
    y_3 = round (y_3.X , 4)
    d_13 = model.getVarByName("d_13")
    d_13 = round (d_13.X , 4)
    d_23 = model.getVarByName("d_23")
    d_23 = round (d_23.X , 4)

    if NumEspacios>3:
        x_4 = model.getVarByName("x_4")
        x_4 = round (x_4.X , 4)
        y_4 = model.getVarByName("y_4")
        y_4 = round (y_4.X , 4)
        d_14 = model.getVarByName("d_14")
        d_14 = round (d_14.X , 4)
        d_24 = model.getVarByName("d_24")
        d_24 = round (d_24.X , 4)
        d_34 = model.getVarByName("d_34")
        d_34 = round (d_34.X , 4)

        if NumEspacios>4:
            x_5 = model.getVarByName("x_5")
            x_5 = round (x_5.X , 4)
            y_5 = model.getVarByName("y_5")
            y_5 = round (y_5.X , 4)
            d_15 = model.getVarByName("d_15")
            d_15 = round (d_15.X , 4)
```

```
d_25 = model.getVarByName("d_25")
d_25 = round (d_25.X , 4)
d_35 = model.getVarByName("d_35")
d_35 = round (d_35.X , 4)
d_45 = model.getVarByName("d_45")
d_45 = round (d_45.X , 4)

if NumEspacios>5:
    x_6 = model.getVarByName("x_6")
    x_6 = round (x_6.X , 4)
    y_6 = model.getVarByName("y_6")
    y_6 = round (y_6.X , 4)
    d_16 = model.getVarByName("d_16")
    d_16 = round (d_16.X , 4)
    d_26 = model.getVarByName("d_26")
    d_26 = round (d_26.X , 4)
    d_36 = model.getVarByName("d_36")
    d_36 = round (d_36.X , 4)
    d_46 = model.getVarByName("d_46")
    d_46 = round (d_46.X , 4)
    d_56 = model.getVarByName("d_56")
    d_56 = round (d_56.X , 4)

    if NumEspacios>6:
        x_7 = model.getVarByName("x_7")
        x_7 = round (x_7.X , 4)
        y_7 = model.getVarByName("y_7")
        y_7 = round (y_7.X , 4)
        d_17 = model.getVarByName("d_17")
        d_17 = round (d_17.X , 4)
        d_27 = model.getVarByName("d_27")
        d_27 = round (d_27.X , 4)
        d_37 = model.getVarByName("d_37")
        d_37 = round (d_37.X , 4)
        d_47 = model.getVarByName("d_47")
        d_47 = round (d_47.X , 4)
        d_57 = model.getVarByName("d_57")
        d_57 = round (d_57.X , 4)
        d_67 = model.getVarByName("d_67")
        d_67 = round (d_67.X , 4)

        if NumEspacios>7:
            x_8 = model.getVarByName("x_8")
            x_8 = round (x_8.X , 4)
            y_8 = model.getVarByName("y_8")
            y_8 = round (y_8.X , 4)
            d_18 = model.getVarByName("d_18")
            d_18 = round (d_18.X , 4)
            d_28 = model.getVarByName("d_28")
            d_28 = round (d_28.X , 4)
            d_38 = model.getVarByName("d_38")
            d_38 = round (d_38.X , 4)
            d_48 = model.getVarByName("d_48")
            d_48 = round (d_48.X , 4)
            d_58 = model.getVarByName("d_58")
            d_58 = round (d_58.X , 4)
            d_68 = model.getVarByName("d_68")
            d_68 = round (d_68.X , 4)
            d_78 = model.getVarByName("d_78")
            d_78 = round (d_78.X , 4)

            if NumEspacios>8:
                x_9 = model.getVarByName("x_9")
                x_9 = round (x_9.X , 4)
```

```
y_9 = model.getVarByName("y_9")
y_9 = round (y_9.X , 4)
d_19 = model.getVarByName("d_19")
d_19 = round (d_19.X , 4)
d_29 = model.getVarByName("d_29")
d_29 = round (d_29.X , 4)
d_39 = model.getVarByName("d_39")
d_39 = round (d_39.X , 4)
d_49 = model.getVarByName("d_49")
d_49 = round (d_49.X , 4)
d_59 = model.getVarByName("d_59")
d_59 = round (d_59.X , 4)
d_69 = model.getVarByName("d_69")
d_69 = round (d_69.X , 4)
d_79 = model.getVarByName("d_79")
d_79 = round (d_79.X , 4)
d_89 = model.getVarByName("d_89")
d_89 = round (d_89.X , 4)

if NumEspacios>9:
    x_10 = model.getVarByName("x_10")
    x_10 = round (x_10.X , 4)
    y_10 = model.getVarByName("y_10")
    y_10 = round (y_10.X , 4)
    d_110 = model.getVarByName("d_110")
    d_110= round (d_110.X , 4)
    d_210 = model.getVarByName("d_210")
    d_210 = round (d_210.X , 4)
    d_310 = model.getVarByName("d_310")
    d_310 = round (d_310.X , 4)
    d_410 = model.getVarByName("d_410")
    d_410 = round (d_410.X , 4)
    d_510 = model.getVarByName("d_510")
    d_510 = round (d_510.X , 4)
    d_610 = model.getVarByName("d_610")
    d_610 = round (d_610.X , 4)
    d_710 = model.getVarByName("d_710")
    d_710 = round (d_710.X , 4)
    d_810 = model.getVarByName("d_810")
    d_810 = round (d_810.X , 4)
    d_910 = model.getVarByName("d_910")
    d_910 = round (d_910.X , 4)

    if NumEspacios>10:
        x_11 = model.getVarByName("x_11")
        x_11 = round (x_11.X , 4)
        y_11 = model.getVarByName("y_11")
        y_11 = round (y_11.X , 4)
        d_111 = model.getVarByName("d_111")
        d_111= round (d_111.X , 4)
        d_211 = model.getVarByName("d_211")
        d_211 = round (d_211.X , 4)
        d_311 = model.getVarByName("d_311")
        d_311 = round (d_311.X , 4)
        d_411 = model.getVarByName("d_411")
        d_411 = round (d_411.X , 4)
        d_511 = model.getVarByName("d_511")
        d_511 = round (d_511.X , 4)
        d_611 = model.getVarByName("d_611")
        d_611 = round (d_611.X , 4)
        d_711 = model.getVarByName("d_711")
        d_711 = round (d_711.X , 4)
        d_811 = model.getVarByName("d_811")
        d_811 = round (d_811.X , 4)
```

```
d_911 = model.getVarByName("d_911")
d_911 = round (d_911.X , 4)
d_1011 = model.getVarByName("d_1011")
d_1011 = round (d_1011.X , 4)

if NumEspacios>11:
    x_12 = model.getVarByName("x_12")
    x_12 = round (x_12.X , 4)
    y_12 = model.getVarByName("y_12")
    y_12 = round (y_12.X , 4)
    d_112 = model.getVarByName("d_112")
    d_112= round (d_112.X , 4)
    d_212 = model.getVarByName("d_212")
    d_212 = round (d_212.X , 4)
    d_312 = model.getVarByName("d_312")
    d_312 = round (d_312.X , 4)
    d_412 = model.getVarByName("d_412")
    d_412 = round (d_412.X , 4)
    d_512 = model.getVarByName("d_512")
    d_512 = round (d_512.X , 4)
    d_612 = model.getVarByName("d_612")
    d_612 = round (d_612.X , 4)
    d_712 = model.getVarByName("d_712")
    d_712 = round (d_712.X , 4)
    d_812 = model.getVarByName("d_812")
    d_812 = round (d_812.X , 4)
    d_912 = model.getVarByName("d_912")
    d_912 = round (d_912.X , 4)
    d_1012 = model.getVarByName("d_1012")
    d_1012 = round (d_1012.X , 4)
    d_1112 = model.getVarByName("d_1112")
    d_1112 = round (d_1112.X , 4)

    if NumEspacios>12:
        x_13 = model.getVarByName("x_13")
        x_13 = round (x_13.X , 4)
        y_13 = model.getVarByName("y_13")
        y_13 = round (y_13.X , 4)
        d_113 = model.getVarByName("d_113")
        d_113= round (d_113.X , 4)
        d_213 = model.getVarByName("d_213")
        d_213 = round (d_213.X , 4)
        d_313 = model.getVarByName("d_313")
        d_313 = round (d_313.X , 4)
        d_413 = model.getVarByName("d_413")
        d_413 = round (d_413.X , 4)
        d_513 = model.getVarByName("d_513")
        d_513 = round (d_513.X , 4)
        d_613 = model.getVarByName("d_613")
        d_613 = round (d_613.X , 4)
        d_713 = model.getVarByName("d_713")
        d_713 = round (d_713.X , 4)
        d_813 = model.getVarByName("d_813")
        d_813 = round (d_813.X , 4)
        d_913 = model.getVarByName("d_913")
        d_913 = round (d_913.X , 4)
        d_1013 = model.getVarByName("d_1013")
        d_1013 = round (d_1013.X , 4)
        d_1113 = model.getVarByName("d_1113")
        d_1113 = round (d_1113.X , 4)
        d_1213 = model.getVarByName("d_1213")
        d_1213 = round (d_1213.X , 4)

        if NumEspacios>13:
```

```
x_14 = model.getVarByName("x_14")
x_14 = round (x_14.X , 4)
y_14 = model.getVarByName("y_14")
y_14 = round (y_14.X , 4)
d_114 = model.getVarByName("d_114")
d_114= round (d_114.X , 4)
d_214 = model.getVarByName("d_214")
d_214 = round (d_214.X , 4)
d_314 = model.getVarByName("d_314")
d_314 = round (d_314.X , 4)
d_414 = model.getVarByName("d_414")
d_414 = round (d_414.X , 4)
d_514 = model.getVarByName("d_514")
d_514 = round (d_514.X , 4)
d_614 = model.getVarByName("d_614")
d_614 = round (d_614.X , 4)
d_714 = model.getVarByName("d_714")
d_714 = round (d_714.X , 4)
d_814 = model.getVarByName("d_814")
d_814 = round (d_814.X , 4)
d_914 = model.getVarByName("d_914")
d_914 = round (d_914.X , 4)
d_1014 = model.getVarByName("d_1014")
d_1014 = round (d_1014.X , 4)
d_1114 = model.getVarByName("d_1114")
d_1114 = round (d_1114.X , 4)
d_1214 = model.getVarByName("d_1214")
d_1214 = round (d_1214.X , 4)
d_1314 = model.getVarByName("d_1314")
d_1314 = round (d_1314.X , 4)

if NumEspacios>14:
    x_15 = model.getVarByName("x_15")
    x_15 = round (x_15.X , 4)
    y_15 = model.getVarByName("y_15")
    y_15 = round (y_15.X , 4)
    d_115 = model.getVarByName("d_115")
    d_115= round (d_115.X , 4)
    d_215 = model.getVarByName("d_215")
    d_215 = round (d_215.X , 4)
    d_315 = model.getVarByName("d_315")
    d_315 = round (d_315.X , 4)
    d_415 = model.getVarByName("d_415")
    d_415 = round (d_415.X , 4)
    d_515 = model.getVarByName("d_515")
    d_515 = round (d_515.X , 4)
    d_615 = model.getVarByName("d_615")
    d_615 = round (d_615.X , 4)
    d_715 = model.getVarByName("d_715")
    d_715 = round (d_715.X , 4)
    d_815 = model.getVarByName("d_815")
    d_815 = round (d_815.X , 4)
    d_915 = model.getVarByName("d_915")
    d_915 = round (d_915.X , 4)
    d_1015 =
```

```
model.getVarByName("d_1015")
```

```
    d_1015 = round (d_1015.X , 4)
    d_1115 =
```

```
model.getVarByName("d_1115")
```

```
    d_1115 = round (d_1115.X , 4)
    d_1215 =
```

```
model.getVarByName("d_1215")
```

```
    d_1215 = round (d_1215.X , 4)
```

```
                                                    d_1315 =
model.getVarByName("d_1315")
                                                    d_1315 = round (d_1315.X , 4)
                                                    d_1415 =
model.getVarByName("d_1415")
                                                    d_1415 = round (d_1415.X , 4)

label_results_L0= Label(second, text = " COORDENADAS DE CADA ESPACIO:", font = ("Arial",
10), bg="white").place(relx=0.01,rely=0.05)
label_results_D0= Label(second, text = " DISTANCIAS ENTRE ESPACIOS (m):", font =
("Arial", 10), bg="white").place(relx=0.27,rely=0.01)
label_results_T0= Label(second, text = " COSTE DEL LAYOUT OPTIMIZAADO ES: " + str(obj) ,
font = ("Arial", 10), bg="white").place(relx=0.01,rely=0.01)

label_results_L1= Label(second, text = "Espacio 1 (X = " + str(x_1) + " , Y = " +
str(y_1) + ")", bg="white").place(relx=0.05,rely=0.09)
label_results_L2= Label(second, text = "Espacio 2 (X = " + str(x_2) + " , Y = " +
str(y_2) + ")", bg="white").place(relx=0.05,rely=0.12)
label_results_D1= Label(second, text = "Distancia entre 1 y 2 es " + str(d_12) ,
bg="white").place(relx=0.3,rely=0.05)

if NumEspacios>2:
    label_results_L3= Label(second, text = "Espacio 3 (X = " + str(x_3) + " , Y = " +
str(y_3) + ")", bg="white").place(relx=0.05,rely=0.15)
    label_results_D2= Label(second, text = "Distancia entre 1 y 3 es " + str(d_13) ,
bg="white").place(relx=0.3,rely=0.08)
    label_results_D3= Label(second, text = "Distancia entre 2 y 3 es " + str(d_23) ,
bg="white").place(relx=0.3,rely=0.11)

    if NumEspacios>3:
        label_results_L4= Label(second, text = "Espacio 4 (X = " + str(x_4) + " , Y = "
+ str(y_4) + ")", bg="white").place(relx=0.05,rely=0.18)
        label_results_D4= Label(second, text = "Distancia entre 1 y 4 es " + str(d_14) ,
bg="white").place(relx=0.3,rely=0.14)
        label_results_D5= Label(second, text = "Distancia entre 2 y 4 es " + str(d_24) ,
bg="white").place(relx=0.3,rely=0.17)
        label_results_D6= Label(second, text = "Distancia entre 3 y 4 es " + str(d_34) ,
bg="white").place(relx=0.3,rely=0.20)

        if NumEspacios>4:
            label_results_L5= Label(second, text = "Espacio 5 (X = " + str(x_5) + " , Y
= " + str(y_5) + ")", bg="white").place(relx=0.05,rely=0.21)
            label_results_D7= Label(second, text = "Distancia entre 1 y 5 es " +
str(d_15) , bg="white").place(relx=0.3,rely=0.23)
            label_results_D8= Label(second, text = "Distancia entre 2 y 5 es " +
str(d_25) , bg="white").place(relx=0.3,rely=0.26)
            label_results_D9= Label(second, text = "Distancia entre 3 y 5 es " +
str(d_35) , bg="white").place(relx=0.3,rely=0.29)
            label_results_D10= Label(second, text = "Distancia entre 4 y 5 es " +
str(d_45) , bg="white").place(relx=0.3,rely=0.32)

            if NumEspacios>5:
                label_results_L6= Label(second, text = "Espacio 6 (X = " + str(x_6) +
" , Y = " + str(y_6) + ")", bg="white").place(relx=0.05,rely=0.24)
                label_results_D11= Label(second, text = "Distancia entre 1 y 6 es " +
str(d_16) , bg="white").place(relx=0.3,rely=0.35)
                label_results_D12= Label(second, text = "Distancia entre 2 y 6 es " +
str(d_26) , bg="white").place(relx=0.3,rely=0.38)
                label_results_D13= Label(second, text = "Distancia entre 3 y 6 es " +
str(d_36) , bg="white").place(relx=0.3,rely=0.41)
                label_results_D14= Label(second, text = "Distancia entre 4 y 6 es " +
str(d_46) , bg="white").place(relx=0.3,rely=0.44)
                label_results_D15= Label(second, text = "Distancia entre 5 y 6 es " +
str(d_56) , bg="white").place(relx=0.3,rely=0.47)
```

```
                if NumEspacios>6:
                    label_results_L7= Label(second, text = "Espacio 7 (X = " + str(x_7)
+ " , Y = " + str(y_7) + ")", bg="white").place(relx=0.05,rely=0.27)
                    label_results_D16= Label(second, text = "Distancia entre 1 y 7 es "
+ str(d_17) , bg="white").place(relx=0.3,rely=0.50)
                    label_results_D17= Label(second, text = "Distancia entre 2 y 7 es "
+ str(d_27) , bg="white").place(relx=0.3,rely=0.53)
                    label_results_D18= Label(second, text = "Distancia entre 3 y 7 es "
+ str(d_37) , bg="white").place(relx=0.3,rely=0.56)
                    label_results_D19= Label(second, text = "Distancia entre 4 y 7 es "
+ str(d_47) , bg="white").place(relx=0.3,rely=0.59)
                    label_results_D20= Label(second, text = "Distancia entre 5 y 7 es "
+ str(d_57) , bg="white").place(relx=0.3,rely=0.62)
                    label_results_D21= Label(second, text = "Distancia entre 6 y 7 es "
+ str(d_67) , bg="white").place(relx=0.3,rely=0.65)

                    if NumEspacios>7:
                        label_results_L8= Label(second, text = "Espacio 8 (X = " +
str(x_8) + " , Y = " + str(y_8) + ")", bg="white").place(relx=0.05,rely=0.30)
                        label_results_D22= Label(second, text = "Distancia entre 1 y 8
es " + str(d_18) , bg="white").place(relx=0.3,rely=0.68)
                        label_results_D23= Label(second, text = "Distancia entre 2 y 8
es " + str(d_28) , bg="white").place(relx=0.3,rely=0.71)
                        label_results_D24= Label(second, text = "Distancia entre 3 y 8
es " + str(d_38) , bg="white").place(relx=0.3,rely=0.74)
                        label_results_D25= Label(second, text = "Distancia entre 4 y 8
es " + str(d_48) , bg="white").place(relx=0.3,rely=0.77)
                        label_results_D26= Label(second, text = "Distancia entre 5 y 8
es " + str(d_58) , bg="white").place(relx=0.3,rely=0.80)
                        label_results_D27= Label(second, text = "Distancia entre 6 y 8
es " + str(d_68) , bg="white").place(relx=0.3,rely=0.83)
                        label_results_D28= Label(second, text = "Distancia entre 7 y 8
es " + str(d_78) , bg="white").place(relx=0.3,rely=0.86)

                        if NumEspacios>8:
                            label_results_L9= Label(second, text = "Espacio 9 (X = " +
str(x_9) + " , Y = " + str(y_9) + ")", bg="white").place(relx=0.05,rely=0.33)
                            label_results_D29= Label(second, text = "Distancia entre 1 y
9 es " + str(d_19) , bg="white").place(relx=0.45,rely=0.05)
                            label_results_D30= Label(second, text = "Distancia entre 2 y
9 es " + str(d_29) , bg="white").place(relx=0.45,rely=0.08)
                            label_results_D31= Label(second, text = "Distancia entre 3 y
9 es " + str(d_39) , bg="white").place(relx=0.45,rely=0.11)
                            label_results_D32= Label(second, text = "Distancia entre 4 y
9 es " + str(d_49) , bg="white").place(relx=0.45,rely=0.14)
                            label_results_D33= Label(second, text = "Distancia entre 5 y
9 es " + str(d_59) , bg="white").place(relx=0.45,rely=0.17)
                            label_results_D34= Label(second, text = "Distancia entre 6 y
9 es " + str(d_69) , bg="white").place(relx=0.45,rely=0.20)
                            label_results_D35= Label(second, text = "Distancia entre 7 y
9 es " + str(d_79) , bg="white").place(relx=0.45,rely=0.23)
                            label_results_D36= Label(second, text = "Distancia entre 8 y
9 es " + str(d_89) , bg="white").place(relx=0.45,rely=0.26)

                            if NumEspacios>9:
                                label_results_L10= Label(second, text = "Espacio 10 (X =
" + str(x_10) + " , Y = " + str(y_10) + ")", bg="white").place(relx=0.05,rely=0.36)
                                label_results_D37= Label(second, text = "Distancia entre
1 y 10 es " + str(d_110) , bg="white").place(relx=0.45,rely=0.29)
                                label_results_D38= Label(second, text = "Distancia entre
2 y 10 es " + str(d_210) , bg="white").place(relx=0.45,rely=0.32)
                                label_results_D39= Label(second, text = "Distancia entre
3 y 10 es " + str(d_310) , bg="white").place(relx=0.45,rely=0.35)
```

```
                                label_results_D40= Label(second, text = "Distancia entre
4 y 10 es " + str(d_410) , bg="white").place(relx=0.45,rely=0.38)
                                label_results_D41= Label(second, text = "Distancia entre
5 y 10 es " + str(d_510) , bg="white").place(relx=0.45,rely=0.41)
                                label_results_D42= Label(second, text = "Distancia entre
6 y 10 es " + str(d_610) , bg="white").place(relx=0.45,rely=0.44)
                                label_results_D43= Label(second, text = "Distancia entre
7 y 10 es " + str(d_710) , bg="white").place(relx=0.45,rely=0.47)
                                label_results_D44= Label(second, text = "Distancia entre
8 y 10 es " + str(d_810) , bg="white").place(relx=0.45,rely=0.50)
                                label_results_D45= Label(second, text = "Distancia entre
9 y 10 es " + str(d_910) , bg="white").place(relx=0.45,rely=0.53)

                                if NumEspacios>10:
                                    label_results_L10= Label(second, text = "Espacio 11
(X = " + str(x_11) + " , Y = " + str(y_11) + ")", bg="white").place(relx=0.05,rely=0.39)
                                    label_results_D46= Label(second, text = "Distancia
entre 1 y 11 es " + str(d_111) , bg="white").place(relx=0.45,rely=0.56)
                                    label_results_D47= Label(second, text = "Distancia
entre 2 y 11 es " + str(d_211) , bg="white").place(relx=0.45,rely=0.59)
                                    label_results_D48= Label(second, text = "Distancia
entre 3 y 11 es " + str(d_311) , bg="white").place(relx=0.45,rely=0.62)
                                    label_results_D49= Label(second, text = "Distancia
entre 4 y 11 es " + str(d_411) , bg="white").place(relx=0.45,rely=0.65)
                                    label_results_D50= Label(second, text = "Distancia
entre 5 y 11 es " + str(d_511) , bg="white").place(relx=0.45,rely=0.68)
                                    label_results_D51= Label(second, text = "Distancia
entre 6 y 11 es " + str(d_611) , bg="white").place(relx=0.45,rely=0.71)
                                    label_results_D52= Label(second, text = "Distancia
entre 7 y 11 es " + str(d_711) , bg="white").place(relx=0.45,rely=0.74)
                                    label_results_D53= Label(second, text = "Distancia
entre 8 y 11 es " + str(d_811) , bg="white").place(relx=0.45,rely=0.77)
                                    label_results_D54= Label(second, text = "Distancia
entre 9 y 11 es " + str(d_911) , bg="white").place(relx=0.45,rely=0.80)
                                    label_results_D55= Label(second, text = "Distancia
entre 10 y 11 es " + str(d_1011) , bg="white").place(relx=0.45,rely=0.83)

                                    if NumEspacios>11:
                                        label_results_L12= Label(second, text = "Espacio
12 (X = " + str(x_12) + " , Y = " + str(y_12) + ")",
bg="white").place(relx=0.05,rely=0.42)
                                        label_results_D56= Label(second, text =
"Distancia entre 1 y 12 es " + str(d_112) , bg="white").place(relx=0.45,rely=0.86)
                                        label_results_D57= Label(second, text =
"Distancia entre 2 y 12 es " + str(d_212) , bg="white").place(relx=0.6,rely=0.05)
                                        label_results_D58= Label(second, text =
"Distancia entre 3 y 12 es " + str(d_312) , bg="white").place(relx=0.6,rely=0.08)
                                        label_results_D59= Label(second, text =
"Distancia entre 4 y 12 es " + str(d_412) , bg="white").place(relx=0.6,rely=0.11)
                                        label_results_D60= Label(second, text =
"Distancia entre 5 y 12 es " + str(d_512) , bg="white").place(relx=0.6,rely=0.14)
                                        label_results_D61= Label(second, text =
"Distancia entre 6 y 12 es " + str(d_612) , bg="white").place(relx=0.6,rely=0.17)
                                        label_results_D62= Label(second, text =
"Distancia entre 7 y 12 es " + str(d_712) , bg="white").place(relx=0.6,rely=0.20)
                                        label_results_D63= Label(second, text =
"Distancia entre 8 y 12 es " + str(d_812) , bg="white").place(relx=0.6,rely=0.23)
                                        label_results_D64= Label(second, text =
"Distancia entre 9 y 12 es " + str(d_912) , bg="white").place(relx=0.6,rely=0.26)
                                        label_results_D65= Label(second, text =
"Distancia entre 10 y 12 es " + str(d_1012) , bg="white").place(relx=0.6,rely=0.29)
                                        label_results_D66= Label(second, text =
"Distancia entre 11 y 12 es " + str(d_1112) , bg="white").place(relx=0.6,rely=0.32)
```

```
                                    if NumEspacios>12:
                                        label_results_L13= Label(second, text =
"Espacio 13 (X = " + str(x_13) + " , Y = " + str(y_13) + ")",
bg="white").place(relx=0.05,rely=0.45)
                                        label_results_D67= Label(second, text =
"Distancia entre 1 y 13 es " + str(d_113) , bg="white").place(relx=0.6,rely=0.35)
                                        label_results_D68= Label(second, text =
"Distancia entre 2 y 13 es " + str(d_213) , bg="white").place(relx=0.6,rely=0.38)
                                        label_results_D69= Label(second, text =
"Distancia entre 3 y 13 es " + str(d_313) , bg="white").place(relx=0.6,rely=0.41)
                                        label_results_D70= Label(second, text =
"Distancia entre 4 y 13 es " + str(d_413) , bg="white").place(relx=0.6,rely=0.44)
                                        label_results_D71= Label(second, text =
"Distancia entre 5 y 13 es " + str(d_513) , bg="white").place(relx=0.6,rely=0.47)
                                        label_results_D72= Label(second, text =
"Distancia entre 6 y 13 es " + str(d_613) , bg="white").place(relx=0.6,rely=0.50)
                                        label_results_D73= Label(second, text =
"Distancia entre 7 y 13 es " + str(d_713) , bg="white").place(relx=0.6,rely=0.53)
                                        label_results_D74= Label(second, text =
"Distancia entre 8 y 13 es " + str(d_813) , bg="white").place(relx=0.6,rely=0.56)
                                        label_results_D75= Label(second, text =
"Distancia entre 9 y 13 es " + str(d_913) , bg="white").place(relx=0.6,rely=0.59)
                                        label_results_D76= Label(second, text =
"Distancia entre 10 y 13 es " + str(d_1013) , bg="white").place(relx=0.6,rely=0.62)
                                        label_results_D77= Label(second, text =
"Distancia entre 11 y 13 es " + str(d_1113) , bg="white").place(relx=0.6,rely=0.65)
                                        label_results_D78= Label(second, text =
"Distancia entre 11 y 13 es " + str(d_1213) , bg="white").place(relx=0.6,rely=0.68)

                                        if NumEspacios>13:
                                            label_results_L14= Label(second, text =
"Espacio 14 (X = " + str(x_14) + " , Y = " + str(y_14) + ")",
bg="white").place(relx=0.05,rely=0.48)
                                            label_results_D79= Label(second, text =
"Distancia entre 1 y 14 es " + str(d_114) , bg="white").place(relx=0.6,rely=0.71)
                                            label_results_D80= Label(second, text =
"Distancia entre 2 y 14 es " + str(d_214) , bg="white").place(relx=0.6,rely=0.74)
                                            label_results_D81= Label(second, text =
"Distancia entre 3 y 14 es " + str(d_314) , bg="white").place(relx=0.6,rely=0.77)
                                            label_results_D82= Label(second, text =
"Distancia entre 4 y 14 es " + str(d_414) , bg="white").place(relx=0.6,rely=0.80)
                                            label_results_D83= Label(second, text =
"Distancia entre 5 y 14 es " + str(d_514) , bg="white").place(relx=0.6,rely=0.83)
                                            label_results_D84= Label(second, text =
"Distancia entre 6 y 14 es " + str(d_614) , bg="white").place(relx=0.6,rely=0.86)
                                            label_results_D85= Label(second, text =
"Distancia entre 7 y 14 es " + str(d_714) , bg="white").place(relx=0.75,rely=0.05)
                                            label_results_D86= Label(second, text =
"Distancia entre 8 y 14 es " + str(d_814) , bg="white").place(relx=0.75,rely=0.08)
                                            label_results_D87= Label(second, text =
"Distancia entre 9 y 14 es " + str(d_914) , bg="white").place(relx=0.75,rely=0.11)
                                            label_results_D88= Label(second, text =
"Distancia entre 10 y 14 es " + str(d_1014) , bg="white").place(relx=0.75,rely=0.14)
                                            label_results_D89= Label(second, text =
"Distancia entre 11 y 14 es " + str(d_1114) , bg="white").place(relx=0.75,rely=0.17)
                                            label_results_D90= Label(second, text =
"Distancia entre 11 y 14 es " + str(d_1214) , bg="white").place(relx=0.75,rely=0.20)
                                            label_results_D91= Label(second, text =
"Distancia entre 11 y 14 es " + str(d_1314) , bg="white").place(relx=0.75,rely=0.23)

                                            if NumEspacios>14:
                                                label_results_L15= Label(second,
text = "Espacio 15 (X = " + str(x_15) + " , Y = " + str(y_15) + ")",
bg="white").place(relx=0.05,rely=0.51)
```

```
                                                    label_results_D92= Label(second,
text = "Distancia entre 1 y 15 es " + str(d_115) ,
bg="white").place(relx=0.75,rely=0.26)
                                                    label_results_D93= Label(second,
text = "Distancia entre 2 y 15 es " + str(d_215) ,
bg="white").place(relx=0.75,rely=0.29)
                                                    label_results_D94= Label(second,
text = "Distancia entre 3 y 15 es " + str(d_315) ,
bg="white").place(relx=0.75,rely=0.32)
                                                    label_results_D95= Label(second,
text = "Distancia entre 4 y 15 es " + str(d_415) ,
bg="white").place(relx=0.75,rely=0.35)
                                                    label_results_D96= Label(second,
text = "Distancia entre 5 y 15 es " + str(d_515) ,
bg="white").place(relx=0.75,rely=0.38)
                                                    label_results_D97= Label(second,
text = "Distancia entre 6 y 15 es " + str(d_615) ,
bg="white").place(relx=0.75,rely=0.41)
                                                    label_results_D98= Label(second,
text = "Distancia entre 7 y 15 es " + str(d_715) ,
bg="white").place(relx=0.75,rely=0.44)
                                                    label_results_D99= Label(second,
text = "Distancia entre 8 y 15 es " + str(d_815) ,
bg="white").place(relx=0.75,rely=0.47)
                                                    label_results_D100= Label(second,
text = "Distancia entre 9 y 15 es " + str(d_915) ,
bg="white").place(relx=0.75,rely=0.50)
                                                    label_results_D101= Label(second,
text = "Distancia entre 10 y 15 es " + str(d_1015) ,
bg="white").place(relx=0.75,rely=0.53)
                                                    label_results_D102= Label(second,
text = "Distancia entre 11 y 15 es " + str(d_1115) ,
bg="white").place(relx=0.75,rely=0.56)
                                                    label_results_D103= Label(second,
text = "Distancia entre 11 y 15 es " + str(d_1215) ,
bg="white").place(relx=0.75,rely=0.59)
                                                    label_results_D104= Label(second,
text = "Distancia entre 11 y 15 es " + str(d_1315) ,
bg="white").place(relx=0.75,rely=0.62)
                                                    label_results_D105= Label(second,
text = "Distancia entre 11 y 15 es " + str(d_1415) ,
bg="white").place(relx=0.75,rely=0.65)

label_results_graph= Label(first, text = " CONFIGURACIÓN ÓPTIMA:", bg="white",
font=("Arial",10)).place(relx=0.01,rely=0.01)
AlturaCanvas= yMAXIM*7
AnchoCanvas = xMAXIM*7
canvas3 = Canvas(first, height= AlturaCanvas, width= AnchoCanvas, bg = "white")
canvas3.place(relx= 0.05 , rely = 0.05)

wb2 = load_workbook('Coordenadas_Dynamo.xlsx')
ws = wb2['Coordinates']

Dept1_rectangle = canvas3.create_rectangle( 7*(x_1 - (Anch[1]/2)), 7*(y_1 +
(Larg[1]/2)), 7*(x_1 + (Anch[1]/2)), 7*(y_1 - (Larg[1]/2)), fill="white")
Dept1_lable = canvas3.create_text( 7*x_1, 7*y_1 , text = "1" , fill = "black")
ws['A1'] = x_1-(Anch[1]/2)
ws['B1'] = y_1-(Larg[1]/2)
ws['A2'] = x_1+(Anch[1]/2)
ws['B2'] = y_1-(Larg[1]/2)
ws['A3'] = x_1+(Anch[1]/2)
ws['B3'] = y_1+(Larg[1]/2)
ws['A4'] = x_1-(Anch[1]/2)
ws['B4'] = y_1+(Larg[1]/2)
```

```
Dept2_rectangle = canvas3.create_rectangle( 7*(x_2 - (Anch[2]/2)), 7*(y_2 +
(Larg[2]/2)), 7*(x_2 + (Anch[2]/2)), 7*(y_2 - (Larg[2]/2)), fill="white")
Dept2_lable = canvas3.create_text( 7*x_2, 7*y_2 , text = "2" , fill = "black")
ws['A5'] = x_2-(Anch[2]/2)
ws['B5'] = y_2-(Larg[2]/2)
ws['A6'] = x_2+(Anch[2]/2)
ws['B6'] = y_2-(Larg[2]/2)
ws['A7'] = x_2+(Anch[2]/2)
ws['B7'] = y_2+(Larg[2]/2)
ws['A8'] = x_2-(Anch[2]/2)
ws['B8'] = y_2+(Larg[2]/2)

if NumEspacios>2:
    Dept3_rectangle = canvas3.create_rectangle( 7*(x_3 - (Anch[3]/2)), 7*(y_3 +
(Larg[3]/2)), 7*(x_3 + (Anch[3]/2)), 7*(y_3 - (Larg[3]/2)), fill="white")
    Dept3_lable = canvas3.create_text( 7*x_3, 7*y_3 , text = "3" , fill = "black")
    ws['A9'] = x_3-(Anch[3]/2)
    ws['B9'] = y_3-(Larg[3]/2)
    ws['A10'] = x_3+(Anch[3]/2)
    ws['B10'] = y_3-(Larg[3]/2)
    ws['A11'] = x_3+(Anch[3]/2)
    ws['B11'] = y_3+(Larg[3]/2)
    ws['A12'] = x_3-(Anch[3]/2)
    ws['B12'] = y_3+(Larg[3]/2)

    if NumEspacios>3:
        Dept4_rectangle = canvas3.create_rectangle(7*(x_4 - (Anch[4]/2)), 7*(y_4 +
(Larg[4]/2)), 7*(x_4 + (Anch[4]/2)), 7*(y_4 - (Larg[4]/2)), fill="white")
        Dept4_lable = canvas3.create_text( 7*x_4, 7*y_4 , text = "4" , fill = "black")
        ws['A13'] = x_4-(Anch[4]/2)
        ws['B13'] = y_4-(Larg[4]/2)
        ws['A14'] = x_4+(Anch[4]/2)
        ws['B14'] = y_4-(Larg[4]/2)
        ws['A15'] = x_4+(Anch[4]/2)
        ws['B15'] = y_4+(Larg[4]/2)
        ws['A16'] = x_4-(Anch[4]/2)
        ws['B16'] = y_4+(Larg[4]/2)

        if NumEspacios>4:
            Dept5_rectangle = canvas3.create_rectangle( 7*(x_5 - (Anch[5]/2)), 7*(y_5 +
(Larg[5]/2)) , 7*(x_5 + (Anch[5]/2)) , 7*(y_5 - (Larg[5]/2)), fill="white")
            Dept5_lable = canvas3.create_text( 7*x_5, 7*y_5 , text = "5" , fill =
"black")
            ws['A17'] = x_5-(Anch[5]/2)
            ws['B17'] = y_5-(Larg[5]/2)
            ws['A18'] = x_5+(Anch[5]/2)
            ws['B18'] = y_5-(Larg[5]/2)
            ws['A19'] = x_5+(Anch[5]/2)
            ws['B19'] = y_5+(Larg[5]/2)
            ws['A20'] = x_5-(Anch[5]/2)
            ws['B20'] = y_5+(Larg[5]/2)

            if NumEspacios>5:
                Dept6_rectangle = canvas3.create_rectangle( 7*(x_6 - (Anch[6]/2)),
7*(y_6 + (Larg[6]/2)) , 7*(x_6 + (Anch[6]/2)) , 7*(y_6 - (Larg[6]/2)), fill="white")
                Dept6_lable = canvas3.create_text( 7*x_6, 7*y_6 , text = "6" , fill =
"black")
                ws['A21'] = x_6-(Anch[6]/2)
                ws['B21'] = y_6-(Larg[6]/2)
                ws['A22'] = x_6+(Anch[6]/2)
                ws['B22'] = y_6-(Larg[6]/2)
                ws['A23'] = x_6+(Anch[6]/2)
                ws['B23'] = y_6+(Larg[6]/2)
```

```
                ws['A24'] = x_6-(Anch[6]/2)
                ws['B24'] = y_6+(Larg[6]/2)

                if NumEspacios>6:
                    Dept7_rectangle = canvas3.create_rectangle( 7*(x_7 - (Anch[7]/2)),
7*(y_7 + (Larg[7]/2)) , 7*(x_7 + (Anch[7]/2)) , 7*(y_7 - (Larg[7]/2)), fill="white")
                    Dept7_lable = canvas3.create_text( 7*x_7, 7*y_7 , text = "7" , fill
= "black")
                    ws['A25'] = x_7-(Anch[7]/2)
                    ws['B25'] = y_7-(Larg[7]/2)
                    ws['A26'] = x_7+(Anch[7]/2)
                    ws['B26'] = y_7-(Larg[7]/2)
                    ws['A27'] = x_7+(Anch[7]/2)
                    ws['B27'] = y_7+(Larg[7]/2)
                    ws['A28'] = x_7-(Anch[7]/2)
                    ws['B28'] = y_7+(Larg[7]/2)

                    if NumEspacios>7:
                        Dept8_rectangle = canvas3.create_rectangle( 7*(x_8 -
(Anch[8]/2)), 7*(y_8 + (Larg[8]/2)) , 7*(x_8 + (Anch[8]/2)) , 7*(y_8 - (Larg[8]/2)),
fill="white")
                        Dept8_lable = canvas3.create_text( 7*x_8, 7*y_8 , text = "8" ,
fill = "black")
                        ws['A29'] = x_8-(Anch[8]/2)
                        ws['B29'] = y_8-(Larg[8]/2)
                        ws['A30'] = x_8+(Anch[8]/2)
                        ws['B30'] = y_8-(Larg[8]/2)
                        ws['A31'] = x_8+(Anch[8]/2)
                        ws['B31'] = y_8+(Larg[8]/2)
                        ws['A32'] = x_8-(Anch[8]/2)
                        ws['B32'] = y_8+(Larg[8]/2)

                        if NumEspacios>8:
                            Dept9_rectangle = canvas3.create_rectangle( 7*(x_9 -
(Anch[9]/2)), 7*(y_9 + (Larg[9]/2)) , 7*(x_9 + (Anch[9]/2)) , 7*(y_9 - (Larg[9]/2)),
fill="white")
                            Dept9_lable = canvas3.create_text( 7*x_9, 7*y_9 , text =
"9" , fill = "black")
                            ws['A33'] = x_9-(Anch[9]/2)
                            ws['B33'] = y_9-(Larg[9]/2)
                            ws['A34'] = x_9+(Anch[9]/2)
                            ws['B34'] = y_9-(Larg[9]/2)
                            ws['A35'] = x_9+(Anch[9]/2)
                            ws['B35'] = y_9+(Larg[9]/2)
                            ws['A36'] = x_9-(Anch[9]/2)
                            ws['B36'] = y_9+(Larg[9]/2)

                            if NumEspacios>9:
                                Dept10_rectangle = canvas3.create_rectangle( 7*(x_10 -
(Anch[10]/2)), 7*(y_10 + (Larg[10]/2)) , 7*(x_10 + (Anch[10]/2)) , 7*(y_10 -
(Larg[10]/2)), fill="white")
                                Dept10_lable = canvas3.create_text( 7*x_10, 7*y_10 ,
text = "10" , fill = "black")
                                ws['A37'] = x_10-(Anch[10]/2)
                                ws['B37'] = y_10-(Larg[10]/2)
                                ws['A38'] = x_10+(Anch[10]/2)
                                ws['B38'] = y_10-(Larg[10]/2)
                                ws['A39'] = x_10+(Anch[10]/2)
                                ws['B39'] = y_10+(Larg[10]/2)
                                ws['A40'] = x_10-(Anch[10]/2)
                                ws['B40'] = y_10+(Larg[10]/2)

                                if NumEspacios>10:
```

```
                                    Dept11_rectangle = canvas3.create_rectangle( 7*(x_11
- (Anch[11]/2)), 7*(y_11 + (Larg[11]/2)) , 7*(x_11 + (Anch[11]/2)) , 7*(y_11 -
(Larg[11]/2)), fill="white")
                                    Dept11_lable = canvas3.create_text( 7*x_11, 7*y_11 ,
text = "11" , fill = "black")
                                    ws['A41'] = x_11-(Anch[11]/2)
                                    ws['B41'] = y_11-(Larg[11]/2)
                                    ws['A42'] = x_11+(Anch[11]/2)
                                    ws['B42'] = y_11-(Larg[11]/2)
                                    ws['A43'] = x_11+(Anch[11]/2)
                                    ws['B43'] = y_11+(Larg[11]/2)
                                    ws['A44'] = x_11-(Anch[11]/2)
                                    ws['B44'] = y_11+(Larg[11]/2)

                                    if NumEspacios>11:
                                        Dept12_rectangle =
canvas3.create_rectangle( 7*(x_12 - (Anch[12]/2)), 7*(y_12 + (Larg[12]/2)) , 7*(x_12 +
(Anch[12]/2)) , 7*(y_12 - (Larg[12]/2)), fill="white")
                                        Dept12_lable = canvas3.create_text( 7*x_12,
7*y_12 , text = "12" , fill = "black")
                                        ws['A45'] = x_12-(Anch[12]/2)
                                        ws['B45'] = y_12-(Larg[12]/2)
                                        ws['A46'] = x_12+(Anch[12]/2)
                                        ws['B46'] = y_12-(Larg[12]/2)
                                        ws['A47'] = x_12+(Anch[12]/2)
                                        ws['B47'] = y_12+(Larg[12]/2)
                                        ws['A48'] = x_12-(Anch[12]/2)
                                        ws['B48'] = y_12+(Larg[12]/2)

                                        if NumEspacios>12:
                                            Dept13_rectangle =
canvas3.create_rectangle( 7*(x_13 - (Anch[13]/2)), 7*(y_13 + (Larg[13]/2)) , 7*(x_13 +
(Anch[13]/2)) , 7*(y_13 - (Larg[13]/2)), fill="white")
                                            Dept13_lable = canvas3.create_text( 7*x_13,
7*y_13 , text = "13" , fill = "black")
                                            ws['A49'] = x_13-(Anch[13]/2)
                                            ws['B49'] = y_13-(Larg[13]/2)
                                            ws['A50'] = x_13+(Anch[13]/2)
                                            ws['B50'] = y_13-(Larg[13]/2)
                                            ws['A51'] = x_13+(Anch[13]/2)
                                            ws['B51'] = y_13+(Larg[13]/2)
                                            ws['A52'] = x_13-(Anch[13]/2)
                                            ws['B52'] = y_13+(Larg[13]/2)

                                            if NumEspacios>13:
                                                Dept14_rectangle =
canvas3.create_rectangle( 7*(x_14 - (Anch[14]/2)), 7*(y_14 + (Larg[14]/2)) , 7*(x_14 +
(Anch[14]/2)) , 7*(y_14 - (Larg[14]/2)), fill="white")
                                                Dept14_lable =
canvas3.create_text( 7*x_14, 7*y_14 , text = "14" , fill = "black")
                                                ws['A53'] = x_14-(Anch[14]/2)
                                                ws['B53'] = y_14-(Larg[14]/2)
                                                ws['A54'] = x_14+(Anch[14]/2)
                                                ws['B54'] = y_14-(Larg[14]/2)
                                                ws['A55'] = x_14+(Anch[14]/2)
                                                ws['B55'] = y_14+(Larg[14]/2)
                                                ws['A56'] = x_14-(Anch[14]/2)
                                                ws['B56'] = y_14+(Larg[14]/2)

                                                if NumEspacios>14:
                                                    Dept15_rectangle =
canvas3.create_rectangle( 7*(x_15 - (Anch[15]/2)), 7*(y_15 + (Larg[15]/2)) , 7*(x_15 +
(Anch[15]/2)) , 7*(y_15 - (Larg[15]/2)), fill="white")
```

```
                                                    Dept15_lable =
canvas3.create_text( 7*x_15, 7*y_15 , text = "15" , fill = "black")
                                                    ws['A57'] = x_15-(Anch[15]/2)
                                                    ws['B57'] = y_15-(Larg[15]/2)
                                                    ws['A58'] = x_15+(Anch[15]/2)
                                                    ws['B58'] = y_15-(Larg[15]/2)
                                                    ws['A59'] = x_15+(Anch[15]/2)
                                                    ws['B59'] = y_15+(Larg[15]/2)
                                                    ws['A60'] = x_15-(Anch[15]/2)
                                                    ws['B60'] = y_15+(Larg[15]/2)

wb2.save('Coordenadas_Dynamo.xlsx')
roota.mainloop()
```

2 CÓDIGO EXPERIMENTAL DE PROYECTO

```
import numpy as np
from pyswarm import pso
import csv
import matplotlib.pyplot as plt

def read_spaces(filename):
    with open(filename, 'r') as f:
        reader = csv.reader(f)
        spaces = []
        next(reader)
        for row in reader:
            spaces.append([float(row[0]), float(row[1])])
        return spaces

def objective(x, spaces):
    total_area = np.sum(np.prod(x.reshape(-1, 1), axis=0) * np.array(spaces))
    return -total_area

def optimize_spaces(spaces):
    lb = np.zeros(len(spaces))
    ub = np.ones(len(spaces))
    xopt, fopt = pso(objective, lb, ub, args=(spaces,), maxiter=300)
    rectangles = []
    for i, s in enumerate(spaces):
        w = xopt[i] * s[0]
        h = xopt[i] * s[1]
        rectangles.append([i+1, s[0], s[1], w, h])
    no_overlapping_rectangles = adjust_positions(rectangles)
    return no_overlapping_rectangles

def adjust_positions(rectangles):
    for i in range(len(rectangles)):
        for j in range(i + 1, len(rectangles)):
            if is_overlapping(rectangles[i], rectangles[j]):
                rectangles[j][1] = rectangles[i][1] + rectangles[i][3]
    return rectangles

def is_overlapping(rect1, rect2):
    x1, y1, w1, h1 = rect1[1:]
    x2, y2, w2, h2 = rect2[1:]
    return x2 < x1 + w1 and x2 + w2 > x1 and y2 < y1 + h1 and y2 + h2 > y1

if __name__ == '__main__':
    spaces = read_spaces('Dimensiones.csv')
    packed_opt_rectangles = optimize_spaces(spaces)
    W = max([r[1] + r[3] for r in packed_opt_rectangles])
    H = max([r[2] + r[4] for r in packed_opt_rectangles])
    fig, ax = plt.subplots()
    for r in packed_opt_rectangles:
        ax.add_patch(plt.Rectangle((r[1], r[2]), r[3], r[4], fill=False))
        ax.annotate(str(r[0]), (r[1]+r[3]/2, r[2]+r[4]/2), color='r', weight='bold',
fontsize=12, ha='center', va='center')
    plt.xlim(-10, W+10)
    plt.ylim(-10, H+10)
    plt.show()
```