Trabajo Fin de Grado
Ingeniería Civil

Data Analysis sobre Modelos BIM en formato IFC4.3

Autor: Jaime Zaforteza Quintanilla
Tutor: Blas González González
Cotutor: Francisco García Romero

Dpto. de Construcciones Arquitectónicas
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla

Sevilla, 2024

Trabajo Fin de Grado
Ingeniería Civil

Data Analysis sobre Modelos BIM en formato IFC4.3

Autor:

Jaime Zaforteza Quintanilla

Tutor:

Blas González González

Ingeniero de Caminos, Canales y Puertos

Profesor del Departamento de Construcciones Arquitectónicas I

Cotutor:

Francisco García Romero

Ingeniero de Caminos, Canales y Puertos

Dpto. de Construcciones Arquitectónicas I

Escuela Técnica Superior de Ingeniería

Universidad de Sevilla

Sevilla, 2024

Trabajo Fin de Grado:

Data Analysis sobre Modelos BIM en formato IFC4.3

Autor: Jaime Zaforteza Quintanilla

Tutor: Blas González González

Cotutor: Francisco García Romero

El tribunal nombrado para juzgar el Proyecto arriba indicado, compuesto por los siguientes miembros:

Presidente:

Vocales:

Secretario:

Acuerdan otorgarle la calificación de:

Sevilla, 2024

El Secretario del Tribunal

A mi familia

A mis maestros

Agradecimientos

A mi familia, que siempre ha creído en mí y me ha brindado su apoyo en cada etapa de este recorrido. Gracias por estar presentes y por ofrecerme el espacio y la confianza para crecer, aprender y superar los desafíos que he encontrado en el camino. A mi padre, tu atención y consejo han sido fundamentales para mantenerme enfocado y seguir adelante.

A mi pareja, Cuca.
Por tu apoyo constante y por estar siempre a mi lado, tanto en los momentos de éxito como en los más difíciles.

A mi tutor, Blas.
Por haberme ayudado a encontrar mi vocación, por sus esfuerzos en potenciar mis capacidades, por el trato profesional que he recibido.

Resumen

El presente Trabajo de Fin de Grado tiene como objetivo principal realizar un análisis detallado y exhaustivo de un modelo BIM utilizando el estándar abierto IFC4.3, desarrollado por BuildingSmart International. Esta nueva extensión del formato abierto es clave para la interoperabilidad en proyectos de infraestructuras civiles, permitiendo la integración de datos provenientes de diversas disciplinas, mejorando la colaboración entre los distintos agentes involucrados y presentando un modelo de libre uso en la industria.

El trabajo explora las mejoras de este estándar IFC4.3 en comparación con versiones anteriores, con un enfoque particular en su aplicación en proyectos de infraestructura como carreteras y otras obras lineales. Además, se evalúa su adaptación en el contexto del Plan BIM España, que promueve la adopción de la metodología BIM en la contratación pública mediante estándares abiertos IFC.

El análisis incluye el uso de software como Civil3D y BlenderBIM para gestionar estos proyectos, evaluando las mejoras que trae la versión IFC4.3. Asimismo, se identifican los desafíos técnicos en su implementación, proponiendo soluciones basadas en buenas prácticas y casos de estudio.

Además, se emplea la programación mediante Python y la librería IfcOpenShell para automatizar procesos de análisis y manipulación de modelos IFC, optimizando el flujo de trabajo y mejorando la precisión en la gestión de datos.

Los resultados obtenidos subrayan la relevancia de estándares abiertos como IFC4.3 para la transformación digital del sector de la construcción, destacando su impacto positivo en la eficiencia y sostenibilidad de los proyectos de ingeniería civil.

Palabras Clave:
BIM, IFC, IFC4.3, INFRAESTRUCTURAS, CIVIL3D, BLENDERBIM, BONSAI, IFCOPENSHELL, OPENBIM, PLAN BIM ESPAÑA, PYTHON, ANÁLISIS DATOS

Abstract

This Final Degree Project aims to conduct a detailed and exhaustive analysis of a BIM model using the open IFC4.3 standard, developed by BuildingSmart International. This new extension of the open format is key for interoperability in civil infrastructure projects, allowing for the integration of data from various disciplines, improving collaboration among the different stakeholders, and providing an open-use model for the industry.

The project explores the improvements of this IFC4.3 standard compared to previous versions, with a particular focus on its application in infrastructure projects such as roads and other linear works. Additionally, its adaptation is evaluated in the context of the Plan BIM Spain, which promotes the adoption of the BIM methodology in public procurement through open IFC standards.

The analysis includes the use of software such as Civil3D and BlenderBIM to manage these projects, assessing the improvements introduced by IFC4.3. Furthermore, the technical challenges in its implementation are identified, proposing solutions based on best practices and case studies.

Additionally, Python programming and the IfcOpenShell library are employed to automate processes of analysis and manipulation of IFC models, optimizing workflow and improving data management accuracy.

The results highlight the relevance of open standards like IFC4.3 for the digital transformation of the construction sector, emphasizing their positive impact on the efficiency and sustainability of civil engineering projects.

Keywords:

BIM, IFC, IFC4.3, INFRASTRUCTURES, CIVIL3D, BLENDERBIM, BONSAI, IFCOPENSHELL, OPENBIM, PLAN BIM ESPAÑA, PYTHON, DATA ANALYSIS

Índice

Índice de Tablas

Índice de Figuras

Notación

IFC	Industry Foundation Classes
BIM	Building Information Modelling
bSI	buildingSmart International
GIS	Sistemas de Información Geográfica
CIBIM	Comisión Interministerial de Building Information Modelling
CDE	Entorno Común de Datos
MVD	Model View Definition
XML	eXtensible Markup Language
JSON	JavaScript Object Notation
CAD	Dibujo Asistido por Ordenador
CSV	Código Seguro de Verificación
UI	Interfaz de Usuario
AEC	Arquitectura, Ingeniería y Construcción
PIM	Modelo de Información del Proyecto
EIR	Requisitos de Intercambio de Información
AIM	Modelo de Información del Activo
AIR	Requisitos de Información del Activo
TIN	Red Irregular de Triángulos
IGN	Instituto Geográfico Nacional
SAC	Subassembly Composer
SDT	Standarized Data Tool

1 OBJETO Y ALCANCE DEL TRABAJO

El presente documento, titulado *"Data Analysis de un Modelo BIM en formato IFC4.3"*, constituye el Trabajo de Fin de Grado (TFG) del alumno **D. Jaime Zaforteza Quintanilla**, enmarcado dentro del Grado en Ingeniería Civil de la Escuela Técnica Superior de Ingeniería de la Universidad de Sevilla. Este TFG responde a la necesidad de integrar conocimientos teóricos y prácticos adquiridos a lo largo de la formación universitaria, aplicándolos en un campo de vanguardia como es el Modelado de Información de Construcción (BIM, por sus siglas en inglés), particularmente en el contexto de la ingeniería civil. El objetivo principal de este trabajo es demostrar la competencia del alumno en el manejo de herramientas avanzadas de modelado digital, así como su capacidad para contribuir al desarrollo de soluciones innovadoras en el sector de la construcción.

El BIM ha transformado el paradigma de la construcción al introducir un enfoque digital que permite una gestión integral y colaborativa de los proyectos, desde la fase de diseño hasta la operación y mantenimiento de las infraestructuras. Esta metodología facilita la creación de un modelo digital que centraliza toda la información del proyecto, lo que reduce errores, optimiza procesos y mejora la toma de decisiones. El estándar IFC (Industry Foundation Classes), en su última actualización del estándar 4.3, se destaca como una herramienta esencial para la interoperabilidad en el ámbito del BIM Civil, permitiendo que diferentes sistemas y software se comuniquen de manera efectiva, sin perder la integridad de los datos.

Esta nueva versión IFC4.3 representa un avance significativo respecto a sus predecesoras, integrando mejoras específicas para el ámbito de las infraestructuras, como carreteras, puentes, túneles y otras obras lineales. Estas mejoras facilitan la representación digital precisa y detallada de los elementos constructivos, permitiendo una mayor eficiencia en la planificación, ejecución y mantenimiento de los proyectos de ingeniería civil. Además, el uso de IFC4.3 está alineado con las normativas internacionales, como la ISO 16739 [1], lo que asegura su aplicación global y su adopción como estándar en la industria de la construcción.

El desarrollo de este trabajo se enmarca en la línea de investigación sobre la aplicación de la metodología BIM en la ingeniería civil, liderada por el Departamento de Construcciones Arquitectónicas I de la Universidad de Sevilla. Esta línea de investigación se centra en explorar cómo la integración de la metodología BIM puede transformar los procesos constructivos, mejorando la eficiencia, sostenibilidad y calidad de las obras de ingeniería civil. El tutor de este trabajo es el ingeniero de caminos, canales y puertos, **D. Blas González González**, profesor del departamento y experto en la implementación de BIM en proyectos de infraestructura. Como cotutor, se cuenta con la colaboración de **D. Francisco García Romero**, ingeniero de caminos, canales y puertos, y *Laurea Magistrale* en Ingeniería Estructural, cuya experiencia como gestor BIM y profesional certificado en herramientas avanzadas de Autodesk aporta un valor añadido a la supervisión del proyecto.

El contexto en el que se desarrolla este trabajo es especialmente relevante, ya que el sector de la construcción se encuentra en un proceso de transformación digital acelerada. La adopción de tecnologías digitales, como el BIM, se ha convertido en una prioridad estratégica para mejorar la competitividad, reducir costos y garantizar la sostenibilidad de los proyectos de infraestructura. En España, el Plan BIM España, aprobado en 2023, marca un hito importante al establecer la obligatoriedad progresiva de la metodología BIM en la contratación pública, alineándose con los objetivos de sostenibilidad y eficiencia promovidos por la Agenda 2030 de la ONU.

El Plan BIM España no solo refuerza la necesidad de adoptar BIM en proyectos públicos, sino que también promueve el uso de estándares abiertos como el IFC, que garantizan la interoperabilidad y facilitan la colaboración entre las distintas partes involucradas en un proyecto. Este plan establece un marco claro para la implementación del BIM en todas las fases de los proyectos, desde el diseño hasta la gestión de activos, y destaca la importancia de formar a los profesionales en el manejo de estas tecnologías.

En este sentido, el presente TFG tiene un doble propósito: por un lado, realizar un análisis exhaustivo de un modelo BIM utilizando el estándar IFC4.3, y por otro, contribuir a la formación del alumno en competencias digitales avanzadas, necesarias para enfrentar los retos de un sector en plena transformación. La elección de este tema refleja la importancia de que los futuros ingenieros civiles no solo dominen los aspectos técnicos de la construcción, sino que también estén capacitados para integrar y gestionar tecnologías digitales en sus proyectos.

La relevancia de este trabajo también radica en su contribución al cuerpo de conocimiento existente sobre la aplicación de estándares abiertos en el BIM, un área que está en constante evolución. A medida que más países

adoptan normativas que exigen el uso de BIM en la construcción, la capacidad de trabajar con estándares como el IFC se convierte en una habilidad esencial para los ingenieros. Este TFG, por tanto, no solo busca cumplir con los requisitos académicos para la obtención del título de Grado en Ingeniería Civil, sino que también pretende aportar una visión práctica y aplicada de cómo el estándar IFC4.3 puede ser adoptado y utilizado en el flujo de trabajo de manejo de modelos BIM para mejorar la gestión y la interoperabilidad en proyectos de infraestructura.

Es importante destacar que este trabajo se inscribe en un contexto global de cambio, donde la digitalización no es solo una tendencia, sino una necesidad para garantizar la competitividad y la sostenibilidad del sector de la construcción. La metodología BIM y el uso de estándares abiertos como IFC4.3 son herramientas fundamentales en este proceso, y este TFG pretende ser una contribución significativa al entendimiento y aplicación de estas herramientas en el ámbito de la ingeniería civil.

1.1 Objetivos

El presente Trabajo de Fin de Grado (TFG) tiene como objetivo principal llevar a cabo un análisis detallado y exhaustivo de un modelo BIM en formato IFC4.3. Este formato es un estándar abierto desarrollado por BuildingSmart International, diseñado para facilitar la interoperabilidad y el intercambio de información en proyectos de construcción, especialmente en el ámbito de las infraestructuras civiles. A continuación, se detallan los objetivos principales y secundarios del trabajo.

Objetivos Principales

- **Evaluar las capacidades del estándar IFC4.3** para gestionar de manera eficiente proyectos de infraestructura civil, explorando su aplicabilidad en diferentes fases del ciclo de vida del proyecto, desde la planificación hasta la operación y mantenimiento de las infraestructuras.
- **Identificar las fortalezas y limitaciones del uso de IFC4.3** en la integración de datos procedentes de diversas disciplinas de ingeniería, con el objetivo de mejorar la colaboración y coordinación entre los distintos agentes involucrados en los proyectos de construcción.
- **Proporcionar recomendaciones prácticas** para optimizar el uso del estándar IFC4.3 en proyectos de infraestructura, contribuyendo al avance de la metodología BIM en el sector de la ingeniería civil.
- **Analizar la interoperabilidad del formato IFC4.3** con otras plataformas y software utilizados en la industria, destacando su impacto en la mejora de la eficiencia y la reducción de errores en el intercambio de información.

Objetivos Secundarios

1. **Explorar las mejoras específicas del estándar IFC4.3** en comparación con versiones anteriores, con énfasis en su aplicación a proyectos de infraestructura como carreteras, puentes, túneles y otras obras lineales.
2. **Evaluar la adaptación y adopción de IFC4.3** en el contexto del Plan BIM España, examinando cómo este estándar puede apoyar la estrategia nacional para la digitalización de la construcción y la implementación de la metodología BIM en proyectos públicos.
3. **Desarrollar un caso práctico** donde se aplique el estándar IFC4.3 en un proyecto de infraestructura civil, proporcionando un análisis detallado de los resultados obtenidos y su relevancia para la mejora de los procesos constructivos.
4. **Identificar y analizar los desafíos técnicos** que surgen en la implementación de IFC4.3, proponiendo soluciones basadas en las mejores prácticas documentadas en la literatura y en el estudio de casos de éxito.
5. **Contribuir a la formación en competencias digitales** para futuros ingenieros civiles, enfatizando la importancia de dominar estándares abiertos como IFC4.3 en el contexto de la transformación digital del sector de la construcción.
6. **Proponer mejoras y recomendaciones** para la estandarización y armonización de datos en proyectos de infraestructuras, basadas en el uso de IFC4.3 y otros estándares abiertos, con el fin de mejorar la

interoperabilidad y la colaboración entre las diferentes partes interesadas en un proyecto.

Estos objetivos, tanto principales como secundarios, orientan el desarrollo de este trabajo hacia una comprensión profunda y aplicada del estándar IFC4.3, con el propósito de mejorar la eficiencia, sostenibilidad y calidad de los proyectos de infraestructura civil a través de la metodología BIM.

1.2 Justificación

La necesidad de la investigación presentada en este Trabajo de Fin de Grado (TFG) surge de la transformación digital que está revolucionando el sector de la construcción, impulsada por la urgencia de mejorar la eficiencia, sostenibilidad y calidad de los proyectos de infraestructura. En este contexto, la metodología Building Information Modeling (BIM) se ha establecido como una herramienta crucial para gestionar de manera integral y eficiente la información de los proyectos a lo largo de todo su ciclo de vida, desde la planificación y diseño hasta la construcción, operación y mantenimiento.

El BIM no solo permite una visión detallada y colaborativa del proyecto, sino que también facilita la integración de diversas disciplinas y la comunicación entre todos los actores involucrados, reduciendo errores, optimizando recursos y mejorando la toma de decisiones. Sin embargo, para que el BIM cumpla plenamente con su potencial, es esencial que se utilicen estándares abiertos que aseguren la interoperabilidad entre los diferentes softwares y plataformas utilizados en la industria. Aquí es donde el estándar IFC4.3 (Industry Foundation Classes), desarrollado por BuildingSmart International, juega un papel fundamental.

El IFC4.3 representa una evolución significativa del estándar IFC, con mejoras y adaptaciones específicas para su aplicación en proyectos de infraestructuras civiles, como carreteras, puentes, túneles y otras obras lineales. Esta versión del estándar se enfoca en la integración de procesos y datos dentro del sector de las infraestructuras, permitiendo una gestión más eficaz y coherente de los proyectos a través de la interoperabilidad y la vinculación con sistemas de información geográfica (GIS). Estas capacidades son vitales para enfrentar los retos de complejidad y escala que presentan las infraestructuras modernas, y su uso es especialmente relevante en un entorno global cada vez más digitalizado.

En el contexto español, la justificación de este trabajo es aún más contundente dado el marco regulatorio reciente que impulsa la adopción del BIM en la contratación pública. El Plan BIM España establece la obligatoriedad progresiva de la metodología BIM en los contratos públicos, alineándose con los objetivos de la Agenda 2030 para el desarrollo sostenible, que buscan fomentar la innovación, la eficiencia energética y la reducción del impacto ambiental en el sector de la construcción. Este plan no solo destaca la importancia del BIM, sino que también subraya la necesidad de utilizar estándares abiertos como el IFC para garantizar la transparencia, interoperabilidad y eficiencia en la gestión de proyectos públicos.

La implementación de BIM y la adopción de estándares abiertos son, por tanto, elementos clave para asegurar que España avance hacia una construcción más digital, sostenible y competitiva. Sin embargo, a pesar del avance normativo, la integración efectiva del estándar IFC4.3 en los proyectos de infraestructura sigue siendo un desafío técnico y práctico que requiere de un análisis profundo y aplicado. Es en este contexto donde se enmarca la investigación de este TFG, cuyo objetivo, como se ha mencionado, es evaluar y demostrar las capacidades del estándar IFC en su primera versión aplicada a las Infraestructuras Civiles 4.3 para mejorar la gestión y la interoperabilidad en proyectos de ingeniería civil.

Además, la formación en competencias digitales, como el dominio del BIM y de estándares como IFC4.3, es indispensable para que los futuros ingenieros civiles estén preparados para los desafíos de un sector en plena transformación digital. Este trabajo no solo contribuye a la adquisición de estas competencias, sino que también pretende aportar conocimiento práctico y aplicado sobre la implementación de tecnologías avanzadas en el ámbito de las infraestructuras, posicionando a los profesionales en la vanguardia de la innovación y sostenibilidad en el sector de la construcción.

1.3 Estructura del Trabajo

La estructura del presente Trabajo de Fin de Grado ha sido considerada para abordar de forma coherente el análisis del estándar IFC4.3 dentro del contexto de la metodología BIM, aplicada a las infraestructuras civiles. La investigación se inicia con este capítulo de **Introducción** que establece el marco general del trabajo. En este apartado, se han definido los objetivos que se pretenden alcanzar, justificado la relevancia de la investigación en el contexto actual de la digitalización del sector de la construcción, y presentado una visión global de la estructura del documento.

A continuación, el trabajo avanza hacia una **Revisión Bibliográfica** en la que se exploran las fuentes más relevantes que sustentan la investigación. Este capítulo se centra especialmente en los documentos técnicos de BuildingSmart International, responsables del desarrollo del estándar IFC, y en las publicaciones relacionadas con la adopción de estándares abiertos en la construcción. La revisión bibliográfica proporciona el contexto teórico necesario, destacando la evolución del estándar IFC y su importancia en la gestión de proyectos de infraestructura civil.

El siguiente capítulo, **Plan BIM España**, se dedica a examinar en detalle este documento estratégico, que establece la obligatoriedad progresiva de la metodología BIM en la contratación pública. Este capítulo analiza las directrices y objetivos del plan, así como su impacto en la modernización del sector de la construcción en España. Se discuten las estrategias de implementación del plan y cómo el estándar IFC4.3 se integra en este marco normativo, subrayando su relevancia para la digitalización de las infraestructuras públicas.

Posteriormente, el trabajo se enfoca en el **Estado Actual del OpenBIM para Infraestructuras Civiles**, un capítulo que explora la adopción y aplicación del estándar IFC4.3 en el ámbito de las infraestructuras. Se examinan las innovaciones y mejoras introducidas por esta versión del estándar, destacando su capacidad para mejorar la interoperabilidad y la colaboración entre las diversas disciplinas involucradas en proyectos de infraestructura. Este análisis es fundamental para comprender cómo el IFC4.3 facilita la gestión de proyectos complejos y contribuye a la eficiencia y sostenibilidad en la construcción.

El capítulo titulado **El IFC Civil en Herramientas BIM de Diseño** aborda la integración del estándar IFC4.3 en las principales herramientas de software utilizadas en el diseño de infraestructuras civiles. Este apartado describe en detalle las capacidades técnicas de estas herramientas y cómo permiten gestionar y compartir información de manera eficiente en proyectos complejos. Se presentan casos de uso específicos que ilustran la aplicación práctica del estándar en el diseño y modelado de infraestructuras, proporcionando ejemplos concretos de su implementación.

El trabajo incluye también un **Estudio de Caso**, donde se aplica el estándar IFC4.3 en un proyecto real de infraestructura civil. Este capítulo describe el proceso de implementación del modelo BIM, desde su creación hasta su análisis y validación. El estudio de caso no solo demuestra las capacidades del estándar en un entorno real, sino que también identifica los desafíos y las soluciones prácticas para su aplicación efectiva en proyectos de infraestructura.

En el capítulo de **Metodología Códigos**, se describe detalladamente el enfoque metodológico empleado para el análisis de los datos del modelo BIM en formato IFC4.3. Este capítulo incluye una explicación de los procedimientos, algoritmos y herramientas utilizadas para codificar y decodificar la información del modelo. Además, se discuten los criterios utilizados para seleccionar las técnicas aplicadas, subrayando su relevancia para alcanzar los objetivos del trabajo y asegurar la validez de los resultados obtenidos.

Finalmente, el trabajo concluye con un capítulo de **Conclusiones y Futuras Líneas de Investigación**, en el que se resumen los principales hallazgos de la investigación. Se evalúan las fortalezas y limitaciones del estándar

IFC4.3, y se proponen recomendaciones para su implementación en futuros proyectos. Este capítulo también sugiere posibles líneas de investigación que podrían abordar los desafíos pendientes o explorar nuevas aplicaciones del estándar en el campo de la ingeniería civil.

2 REVISIÓN BIBLIOGRÁFICA

En este capítulo se procede a realizar una revisión bibliográfica centrada en las principales fuentes utilizadas para abordar los objetivos del presente Trabajo de Fin de Grado. En primer lugar, se ha investigado en profundidad la información proporcionada por BuildingSmart International, particularmente en lo referente a la actualización del estándar IFC a su versión 4.3. Esta nueva versión incluye importantes mejoras y adaptaciones para los nuevos ámbitos de uso en el sector de la construcción, los cuales se detallan en el capítulo 4. BuildingSmart se consolida, así como una fuente clave, proporcionando la base normativa y técnica necesaria para comprender la evolución y aplicación del estándar IFC en proyectos actuales.

Por otro lado, también se ha considerado de especial relevancia el material ofrecido por la Comisión Interministerial BIM, que desempeña un papel fundamental en la implementación de la metodología BIM en España. En este sentido, se ha revisado detalladamente el Plan BIM España, además de los datos relacionados con licitaciones públicas en las que se exige la presencia de BIM, como se analiza en el capítulo 3. La información proveniente de esta comisión no solo ofrece un panorama claro sobre la situación actual del BIM en el país, sino que también proporciona directrices y estrategias a seguir para la adopción y desarrollo de esta metodología en proyectos constructivos.

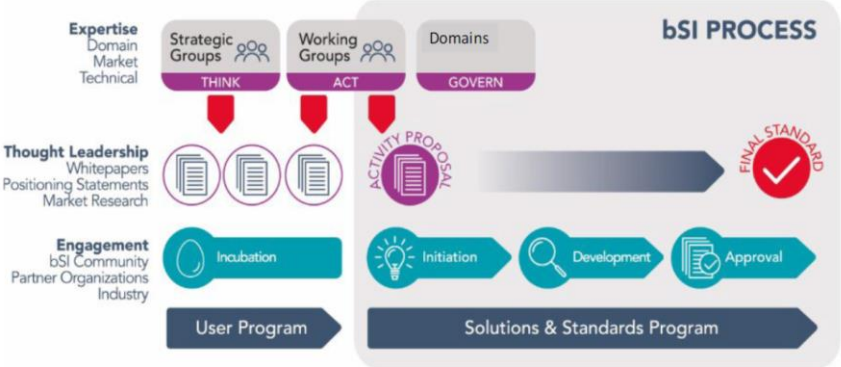
La revisión bibliográfica se ha acotado a estas dos fuentes principales: BuildingSmart International, como referencia normativa y técnica respecto al estándar IFC4.3, y la Comisión Interministerial BIM, como referente en la implementación y fomento del BIM en el contexto del territorio español. Estas fuentes son fundamentales para sustentar los análisis y propuestas desarrolladas en el presente trabajo.

2.1 BuildingSmart International

En el proceso de revisión bibliográfica, se ha examinado minuciosamente la documentación proporcionada por BuildingSmart International, que desempeña un papel crucial en el desarrollo y estandarización de procesos en el sector de la construcción mediante la metodología BIM. Particularmente, se ha focalizado la atención en el estándar IFC 4.3, que representa una evolución significativa respecto a versiones anteriores, integrando mejoras específicas para el ámbito de las infraestructuras.

El propósito del ámbito de las infraestructuras, tal como lo define BuildingSmart, es combinar, mejorar y desarrollar normas abiertas para datos inteligentes que faciliten la integración de procesos y datos dentro de este sector. Este enfoque no solo permite un intercambio de información más efectivo, sino que también refuerza la gestión integral del entorno construido, vinculando e integrando los modelos BIM con los sistemas de información geográfica (GIS). El estándar IFC 4.3 abarca una amplia gama de dominios dentro de las infraestructuras, incluyendo carreteras, puentes, túneles, puertos y vías navegables. Estas áreas han sido priorizadas en la hoja de ruta de BuildingSmart, que traza el camino para el desarrollo y aplicación de estos estándares [2].

Figura 1 – Proceso de la Building Smart International (Fuente: Building Smart)



EL PROCESO BUILDING SMART

BuildingSmart cuenta con un proceso formal [3] para el desarrollo de estándares y soluciones basados en el consenso. Este proceso tiene como objetivo garantizar que existan métodos claros y transparentes para trabajar en el desarrollo de soluciones y estándares que fomenten la participación y la construcción de consensos entre las partes interesadas.

El desarrollo de estos estándares se ejecuta formalmente a través de proyectos que siguen un proceso en tres fases: iniciación, desarrollo de la solución y aprobación.

1. **Fase de Iniciación**: En esta fase, se desarrollan propuestas de estándares que identifican claramente las necesidades de los usuarios y el concepto del estándar. Estas propuestas se documentan mediante casos de uso, un plan de trabajo, un plan de apoyo de las partes interesadas y un plan de ejecución detallado.
2. **Fase de Desarrollo**: Durante esta fase, el trabajo es llevado a cabo por un consorcio de estándares. Se realizan revisiones de expertos en áreas como los requisitos de usuarios, requisitos comerciales, arquitectura técnica y requisitos de implementación, todo ello para apoyar los objetivos de consenso y coherencia. Una vez que el trabajo de desarrollo se considera sustancialmente completo, el Dominio o Grupo patrocinador respalda los resultados, y el Comité Ejecutivo de Estándares revisa el Borrador de Estándar bSI.
3. **Fase de Aprobación**: Una vez que se alcanza esta fase, se requiere documentación que demuestre que se ha logrado el consenso entre las partes interesadas para su aprobación. Las reglas de votación y notificación que rigen el avance de los estándares en la escala de madurez están definidas por el Proceso de Estándares de buildingSMART. Este paso culmina con la designación del estándar como un Estándar Candidato de bSI, iniciando así la fase formal de aprobación.

Así la organización no solo asegura que los estándares desarrollados por buildingSMART cumplan con las necesidades actuales del sector, sino que también mantengan un alto nivel de rigor y calidad técnica, facilitando así su adopción global.

DOCUMENTACIÓN ESPECÍFICA REVISADA

Dentro de este marco, se ha revisado de manera exhaustiva la **hoja de ruta del InfraRoom** publicada por BuildingSmart en 2023, que establece las estrategias a seguir para la implementación global del estándar IFC 4.3. Este documento es fundamental para comprender la evolución futura de los estándares de infraestructuras, destacando la importancia de la interoperabilidad y la integración de datos a través de BIM y GIS [2].

Un libro blanco *"Whitepaper"* es un informe o guía que informa a los lectores de forma concisa sobre un tema complejo y presenta la filosofía del organismo emisor al respecto. Su objetivo es ayudar a los lectores a comprender una cuestión, resolver un problema o tomar una decisión. Un libro blanco es el primer documento que deben leer los investigadores para comprender mejor un concepto o idea central.

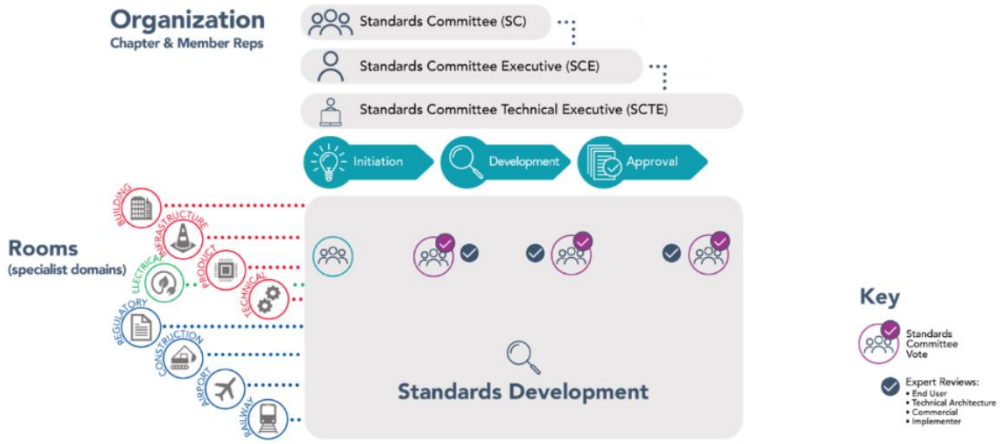
Asimismo, el **Libro Blanco *"Whitepaper"* del grupo de trabajo de la hoja de ruta de Infraestructuras** [4], publicado en 2020, se ha considerado una referencia clave. Este documento aborda los requisitos técnicos y de usuario para la implementación efectiva del estándar IFC en infraestructuras, subrayando la necesidad de mantener estándares abiertos y sostenibles.

Estos documentos proporcionan el fundamento normativo y estratégico sobre el cual se ha sustentado gran parte del presente trabajo a la hora de abarcar la adaptación del estándar IFC a la industria del diseño de infraestructuras, permitiendo así entender el enfoque actual y futuro de la integración de BIM en proyectos de infraestructuras.

Asimismo, estos no son los únicos documentos que deben ser considerados para obtener toda la información necesaria en relación con la aplicación del estándar abierto a proyectos de infraestructuras. Partiendo de los documentos generales mencionados, es esencial profundizar en cada uno de los ámbitos definidos por los laboratorios de investigación de los participantes en el proceso de BuildingSmart, denominados *"Rooms"*.

Durante los años 2018 y 2019, BuildingSmart llevó a cabo una serie de proyectos paralelos con el objetivo de ampliar el alcance del estándar IFC en diversos dominios de infraestructura. A raíz de una serie de talleres, las *"Rooms"* de Ferrocarriles e Infraestructuras establecieron una colaboración estrecha para armonizar estos proyectos, que culminó con la creación del proyecto al que se denominó *"Extensiones de Infraestructuras IFC"*.

Figura 2 – Proceso de desarrollo de los estándar OpenBIM (Fuente: Building Smart)



El proyecto de **Extensiones de Infraestructura de IFC** constituye un enfoque armonizado para la gestión de activos de infraestructura lineal que incluye carreteras, ferrocarriles, puentes, puertos y vías navegables. Aunque los proyectos individuales operan con diferentes marcos temporales e hitos, todos ellos abordan conceptos comunes que presentan una superposición significativa. Además de las áreas específicas de cada dominio, se creó un proyecto adicional denominado *"Esquema Común"*, cuya finalidad es coordinar estos proyectos separados, identificar y definir conceptos comunes como la estructura espacial, la geotecnia, el movimiento de tierras y las redes de servicios públicos, y garantizar así un alto nivel de armonización y coherencia en el desarrollo de las extensiones de cada dominio.

En particular, los proyectos relacionados con carreteras y puertos y vías navegables, tras haber completado la fase de definición de requisitos, avanzaron en el desarrollo de modelos conceptuales y esquemas preliminares de sus respectivas extensiones a lo largo de 2019. Paralelamente, el proyecto ferroviario elaboró tres informes fundamentales (análisis de requisitos, modelo conceptual y requisitos de datos) que fueron presentados como parte del paquete de estándares candidatos durante la Cumbre de Beijing, celebrada en octubre de 2019. El proyecto del esquema común de las infraestructuras también entregó propuestas de extensión para áreas clave, como la geotecnia, el movimiento de tierras, la estructura espacial y las envolventes cinemáticas.

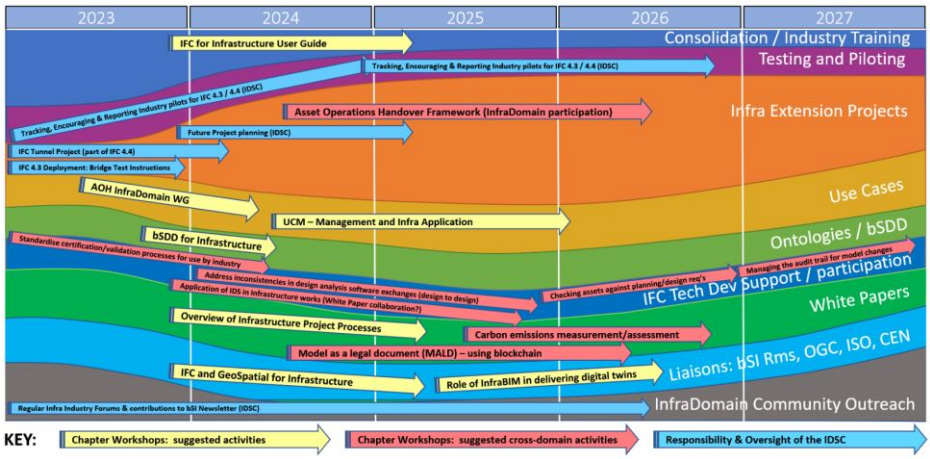
A mediados de 2019, se identificó la necesidad urgente de **armonizar** los trabajos en estos dominios, teniendo en cuenta los avances previos realizados, especialmente en el estándar candidato de puentes "IfcBridge" de la CFI. Como resultado, se alcanzó un consenso entre los distintos dominios de infraestructura, acordando que el objetivo común sería la entrega de una única propuesta de extensión del esquema de la CFI que incorporara de manera armonizada todas las extensiones propuestas (carreteras, ferrocarriles, puertos y vías navegables, y el esquema común).

Como resultado de la investigación de los procesos de armonización para cada uno de los ámbitos propuestos por BuildingSmart se obtuvo toda la documentación para efectuar una revisión bibliográfica que proporcione una visión en profundidad referente al nuevo estándar IFC para las infraestructuras.

En primer lugar, la documentación general del estándar previamente mencionada:

1) El libro Blanco del grupo de trabajo de la hoja de ruta de Infraestructuras "*buildingSMART InfraRoom Forethought Whitepaper"*, 2020.
2) La hoja de ruta a futuro del InfraRoom *"Infrastructure Domain Roadmap"*, 2023.

Figura 3 – Proyección del desarrollo del estándar IFC a lo largo de los años (Fuente: Building Smart)



Además de la documentación introducida para cada uno de los ámbitos de las infraestructuras contenidos:

- Ámbito de Puentes *"IfcBridge"*, previamente introducido en el IFC 4.2:
  1) Plan de Proyecto para Puentes *"Project Proposal for IFC Bridge"*, 2017.
  2) Análisis de Requisitos para Puentes *"Requirements Analysis for IFC Bridge"*, 2018.
  3) Modelo Conceptual para Puentes *"Conceptual Model for IFC Bridge"*, 2018.
  4) Estándar Candidato para Puentes IFC4.2 *"IFC Bridge Candidate Standard"*, 2020.
- Ámbito de Carreteras *"IfcRoad"*, únicamente se ha centrado el estudio de la fase 2 del proyecto:
  1) Estándar Candidato para Carreteras *"IFC Road Candidate Standard"*, 2020.
  2) Plan de Ejecución del Proyecto Fase 2 *"Project Execution Plan Phase 2"*, 2019.
  3) Modelo Conceptual para Carreteras *"Conceptual Model for IFC Road"*, 2020.
- Ámbito Ferroviario *"IfcRailway"*, siguiendo la dinámica de revisión del ámbito anterior únicamente se ha examinado la segunda fase de concepción del proyecto:
  1) Informe de la Fase 2 de IFC Rail *"IFC Rail Phase 2 Report"*
  2) Informe de Implementación y Validación *"IFC 4.3 Implementation & Validation Report"*
  3) Informe del Foro de Implementadores de IFC Rail *"IFC Rail Implementers Forum Report"*
  4) Informe de Validación de la Historia *"Storyline (SL) Validation Report Phase 2"*
  5) Informe de Conjunto de Propiedades *"Property Set Report"*
  6) Informe de Modelo Conceptual *"Conceptual Model Report"*
- Armonización de los ámbitos de infraestructuras para el estándar IFC4.3:
  1) Introducción a los Informes UML *"UML Model Report Part 1 – Introduction"*, 2020.
  2) Modelo UML del Esquema Común *"UML Model Report Part 2 - Common Schema"*, 2020.
  3) Modelo UML del Ámbito Ferroviario *"UML Model Report Part 4 – Railway"*, 2020.
  4) Modelo UML del Ámbito Carreteras *"UML Model Report Part 5 – Road"*, 2020.

2.2 Comisión Interministerial BIM

En el marco del presente Trabajo de Fin de Grado, se ha revisado en profundidad la documentación proporcionada por la Comisión Interministerial para la incorporación de la metodología BIM en la contratación pública. Esta revisión se ha centrado principalmente en la documentación derivada del Plan BIM España y los datos de licitaciones BIM presentados por el Observatorio de la comisión. Estas fuentes son las claves establecidas en la investigación para comprender la evolución y el estado actual de la adopción de BIM en la contratación pública en España, así como la implementación de los estándares openBIM en la industria.

El Plan BIM España es un documento clave aprobado por el consejo de ministros que establece la hoja de ruta para la incorporación gradual y obligatoria de la metodología BIM en la contratación pública, para conocer en profundidad su contenida en lo que refiere a los objetivos de este trabajo ver el capítulo 2.

La documentación tomada en relación con el Plan BIM España para la revisión bibliográfica:

1) *"Resumen Ejecutivo PLAN BIM España"*, 2023.

   Visión general del Plan BIM incluyendo objetivos, fases de implementación y la estrategia para la incorporación progresiva de BIM en la contratación pública en España.

2) *"PLAN BIM en la Contratación Pública"*, 2023.

   Documento detallado que describe las directrices y el calendario para la obligatoriedad de BIM en los contratos públicos, incluyendo los niveles de adopción BIM y los requisitos técnicos correspondientes.

3) *"Fundamentos BIM para la contratación pública"*, 2022.

El Observatorio BIM, creado en 2017, es otra fuente esencial revisada en este trabajo. Su función principal es monitorear y analizar el uso de la metodología BIM en las licitaciones públicas en España.

Asimismo, cabe destacar que no proporciona únicamente datos cuantitativos, sino que también ofrece un análisis cualitativo sobre las tendencias y desafíos en la implementación de BIM, permitiendo a los diferentes actores del sector público y privado ajustar sus estrategias y mejorar su preparación para la adopción de BIM. Esto es particularmente relevante en el contexto de la investigación en lo referente de la aplicación de metodologías BIM en el territorio español en la actualidad. En lo referente al observatorio se han tomado para la revisión bibliográfica:

1) *"Informe de Licitaciones BIM: Datos del Primer Semestre de 2022"*, 2022.
2) *"Informe de Licitaciones BIM: Datos del Segundo Semestre de 2022"*, 2022.
3) *"Informe de Licitaciones BIM: Datos del Tercer Semestre de 2022"*, 2022.
4) *"Informe de Licitaciones BIM: Datos del Cuarto Semestre de 2022"*, 2022.
5) *"Informe de Licitaciones BIM: Año 2022 Completo"*, 2022.
6) *"Informe de Licitaciones BIM: Datos del Primer Semestre de 2023"*, 2023.
7) *"Informe de Licitaciones BIM: Datos del Segundo Semestre de 2023"*, 2023.
8) *"Informe de Licitaciones BIM: Datos del Tercer Semestre de 2023"*, 2023.
9) *"Informe de Licitaciones BIM: Datos del Cuarto Semestre de 2023"*, 2023.
10) *"Informe de Licitaciones BIM: Año 2023 Completo"*, 2023.

3 PLAN BIM ESPAÑA

Dentro del marco de este trabajo, es imperativo abordar el Plan BIM España [5], una iniciativa estratégica aprobada por el Consejo de Ministros el 27 de junio de 2023, elaborada y promovida por el Ministerio de Transportes, Movilidad y Agenda Urbana junto con el Ministerio de Hacienda y Función Pública. Surge como una respuesta deliberada a los desafíos de eficiencia, sostenibilidad y calidad que enfrenta la industria de la construcción en el país. A través de la adopción coordinada de la metodología Building Information Modeling (BIM) [6], este plan busca transformar digitalmente el sector, optimizando los procesos constructivos y fomentando una gestión de proyectos más efectiva.

Coordinado por la Comisión Interministerial BIM (CIBIM), un órgano colegiado que incluye representantes de varios ministerios y entidades estatales, el plan asegura un enfoque integrado y multifacético. La CIBIM, establecida por el Real Decreto 1515/2018 y constituida en abril de 2019, juega un papel crucial en la coordinación, seguimiento y promoción de las iniciativas BIM en España.

Entre sus funciones, se encuentran:

- La elaboración de un Plan de incorporación de la metodología BIM en la contratación pública.
- Seguimiento de las medidas contenidas en dicho Plan.
- Realización de acciones de información y formación de personal.
- Representación del Reino de España y otros Órganos de la Administración General del Estado en foros internacionales sobre la metodología BIM.
- Recepción e Intercambio de información entre los distintos Ministerios y Órganos de la Administración General del Estado implicados sobre la Metodología BIM.

Figura 4 - Comisión Interministerial BIM (Fuente: Portal web Comisión Interministerial BIM)

Históricamente, el sector de la construcción ha experimentado retos significativos que limitaban su potencial de innovación y desarrollo. La necesidad de modernizar las prácticas constructivas, reducir los costos operativos y mejorar los estándares de calidad, impulsó a la creación del Plan BIM. Este enfoque estratégico no solo se alinea con las tendencias globales de digitalización en la construcción, sino que también establece a España como un participante activo en la promoción de prácticas constructivas avanzadas y sostenibles.

Los objetivos del Plan BIM son claros, promover la adopción generalizada de BIM tanto en la administración pública como en el sector privado, mejorar la eficiencia en el proceso de construcción y aumentar la competitividad de las empresas españolas. Al hacerlo, el plan no solo busca mejorar la calidad y eficiencia de los proyectos de construcción sino también facilitar la interoperabilidad y colaboración entre los diferentes actores del sector.

Queda por tanto definido que una piedra angular del plan es la promoción de estándares abiertos, como el Industry Foundation Classes (IFC), para el intercambio de información, lo que asegura la interoperabilidad y seguridad de los datos en los proyectos de construcción. Estos estándares son esenciales para garantizar el uso efectivo de la información a lo largo del ciclo de vida de un proyecto, desde su concepción.

3.1 Acciones y Medidas implantadas

Las acciones y medidas implementadas bajo el Plan BIM son diversas y abarcan desde la formación de profesionales en BIM, el desarrollo de estándares y guías, hasta la promoción de la interoperabilidad mediante el uso de formatos y protocolos comunes, como son las nuevas tendencias OpenBIM. Estas acciones están orientadas a facilitar la incorporación de BIM en proyectos de construcción, tanto públicos como privados, y a asegurar que los beneficios de esta metodología sean ampliamente reconocidos y aprovechados.

ESTRATEGIAS

Creación de la Oficina BIM: Se ha establecido una oficina central que coordina, sigue y comunica los avances del Plan BIM. Esta oficina actúa como enlace entre los diferentes ministerios y entidades, resolviendo dudas y fomentando el intercambio de experiencias y buenas prácticas.
Integración con Políticas Públicas: BIM se ha incorporado en políticas y estrategias nacionales relacionadas con la construcción y la digitalización, promoviendo su adopción más allá del ámbito de la contratación pública.

PERSONAS

Programas de Formación: Se han diseñado e implementado programas específicos para capacitar al personal de los órganos de contratación y a los agentes del sector privado en el uso de BIM. Estos programas abarcan desde fundamentos básicos hasta aspectos avanzados de la metodología.

PROCESOS

Modificación de Pliegos de Condiciones: Los pliegos de condiciones técnicas y administrativas se han adaptado para incluir requisitos específicos de BIM, asegurando que la información del proyecto se gestione adecuadamente a lo largo de todo su ciclo de vida.

TECNOLOGÍAS

Entorno Común de Datos (CDE): Se ha promovido el uso de plataformas CDE para facilitar el almacenamiento, gestión y compartición de la información de los proyectos, favoreciendo la colaboración entre los diferentes agentes involucrados.
Adopción de Estándares Abiertos: Se ha fomentado el uso de estándares abiertos, como IFC y BCF, para garantizar la interoperabilidad y la neutralidad tecnológica en el intercambio de información entre los distintos softwares de BIM utilizados por los participantes en los proyectos.

SEGUIMIENTO Y EVALUACIÓN

Para garantizar el éxito de la implantación de BIM en la contratación pública, se ha establecido un mecanismo de seguimiento y evaluación que permite monitorear el avance y realizar ajustes en el Plan BIM. Este seguimiento se lleva a cabo mediante el análisis de las licitaciones publicadas y el *feedback* de los participantes en los proyectos.

3.2 Uso Actual de la metodología BIM en España

Gracias al portal público de observatorio BIM [7] implantado por la CIBIM, es posible analizar la evolución del uso de la metodología BIM con datos de licitaciones públicas del sector recogidos desde el año 2017. El uso actual del modelado de información de la construcción (BIM) en España ha experimentado un crecimiento significativo y sostenido, especialmente evidente a través del análisis detallado de las licitaciones con requisitos BIM durante el cuarto trimestre de 2023 [8]. Este aumento refleja una transformación profunda hacia la

digitalización y la eficiencia en la gestión de proyectos de construcción en la contratación pública. Con un registro de 284 licitaciones y una inversión estimada de 785 millones de euros en el último trimestre del año, el 2023 marcó un hito en la implementación de técnicas BIM, cerrando con una inversión acumulada en licitaciones BIM de aproximadamente 3.697 millones de euros, lo que representa un crecimiento del 125% en comparación con el año anterior.

Figura 5 - Evolución tanto en inversión como de número de licitaciones (Fuente: CIBIM)

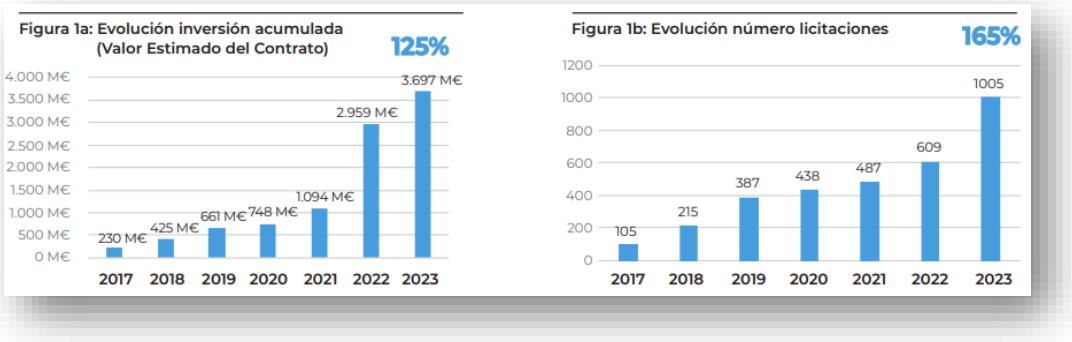
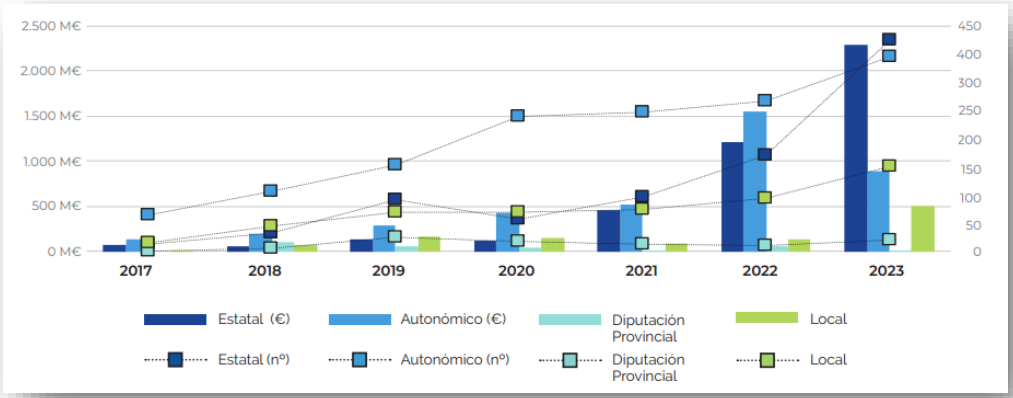
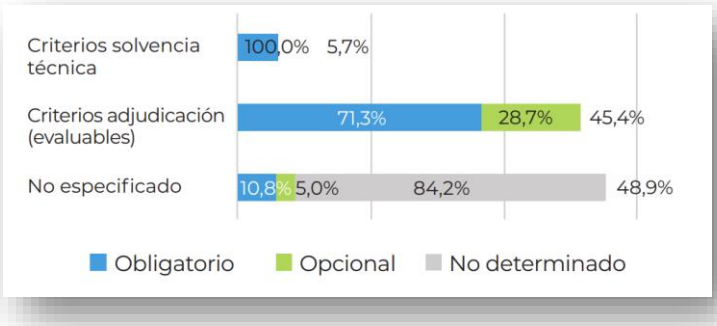


Figura 6 - Evolución en licitaciones BIM 2017 - 2023 (Fuente: CIBIM)



La inclusión de BIM en los Pliegos de Cláusulas Administrativas (PCA) varía, abarcando desde criterios de adjudicación y requisitos de solvencia técnica hasta mejoras técnicas. Esta variabilidad indica que los requisitos BIM pueden ser tanto obligatorios como opcionales, dependiendo de su especificación en los documentos de licitación, con una notable proporción de licitaciones haciendo obligatorio su cumplimiento.

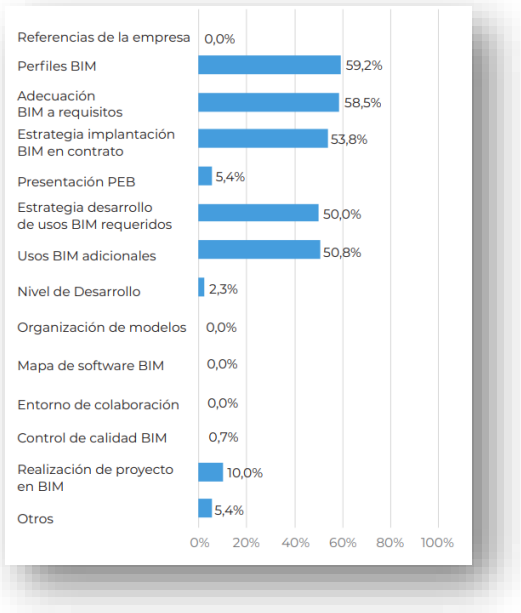
Figura 7 - Introducción BIM en Pliego de Cláusulas Administrativas (Fuente: CIBIM)



La presencia de BIM como criterio de adjudicación es especialmente prominente, lo que subraya su creciente importancia en los procesos de contratación pública. Generalmente, el uso de BIM aparece en los Pliegos de

Cláusulas Administrativas Particulares (PCAP) como criterio de adjudicación, un 45% de las veces, por ejemplo, como criterios cualitativos evaluables mediante fórmulas entre otros.

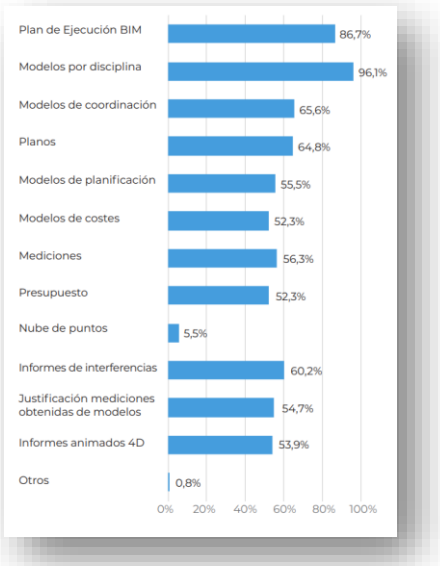
Figura 8 - Aspectos BIM valorados dentro de la puntuación técnica (Fuente: CIBIM)



La aplicación de BIM se extiende en diversidad de usos prácticos, incluyendo la coordinación 3D, visualización de los modelos, obtención de mediciones y presupuestos de forma automatizada y la ejecución de simulaciones constructivas, entre otros. Esta diversidad demuestra la profundidad con la que BIM se está integrando en los proyectos de construcción en España, con un énfasis creciente en la definición de requisitos específicos para cada proyecto. Esta tendencia refleja un nivel en avance de madurez en la implementación de BIM, evidenciado por la exigencia de entregables BIM específicos como modelos por disciplina y planes de ejecución BIM.

Además, el 21% de las ofertas de licitación que implican el uso de metodologías BIM, demandan la implementación de un control de calidad especializado del contenido BIM durante la fase de ejecución del contrato, principalmente en lo referente a la Estructura del modelo, Clasificación, Niveles de información entre otros.

Figura 9 - Porcentaje de licitaciones que solicita la elaboración de distintos entregables (Fuente: CIBIM)



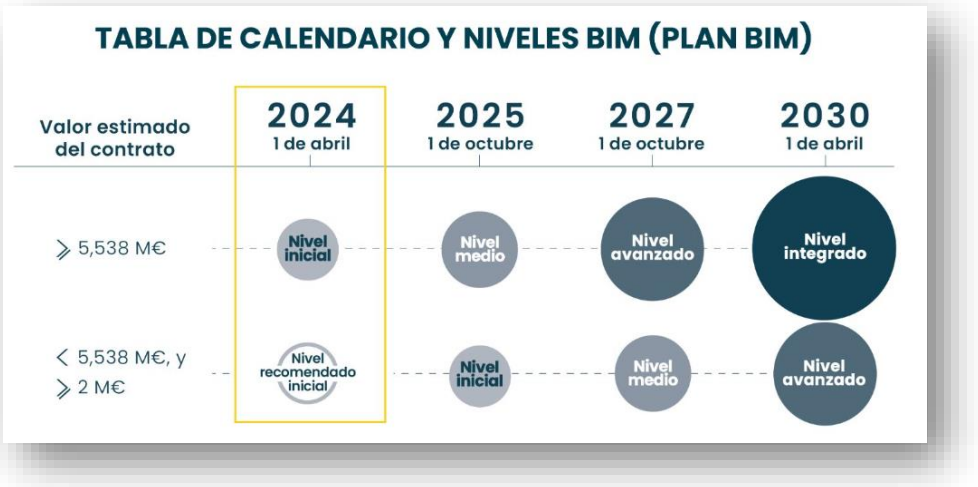
A nivel geográfico, la adopción de BIM muestra una distribución amplia a lo largo de España, con todas las comunidades autónomas igualando o incrementando el número de licitaciones con requisitos BIM en 2023. Esto indica una adopción generalizada del BIM en todo el país, con una inversión significativa en comunidades como Madrid, Cataluña y Andalucía, reflejando el papel central de estas regiones en el avance de la digitalización en la construcción.

Como se ha observado, se concluye que el uso de BIM en España está claramente en una trayectoria ascendente, respetando un compromiso creciente con la eficiencia, transparencia y sostenibilidad en el sector de la construcción. La implementación de BIM se está consolidando como un estándar en la contratación pública, reflejando el impacto de la CIBIM junto a su reciente Plan BIM, aportando las mejoras significativas en la gestión de estos. Aunque aún está en curso la evaluación cuantitativa completa de estos resultados en siguientes trimestres, los avances indican una dirección positiva hacia la consecución de los objetivos planteados.

3.2.1 Etapas del Plan BIM

Sin embargo, cabe destacar que la implementación del Plan BIM [9] también enfrenta varios desafíos, incluyendo la necesidad de superar la **resistencia al cambio** y la adaptación de las normativas existentes para facilitar una adopción más amplia de las técnicas BIM en el territorio español. A futuro, se espera que el Plan BIM continúe fomentando la integración profunda de BIM en todos los niveles de proyectos de construcción y consolide su posición como un estándar en la industria.

Figura 10 - Niveles del Plan de incorporación BIM (Fuente: Plan BIM España, CIBIM)



Acorde al Plan BIM, se establece un marco estratégico para el sistema de Contratación Pública de España [5] con el objetivo de generar una adopción progresiva de la metodología BIM dentro de la Administración General del Estado, sus organismos públicos y entidades de derecho público vinculados o dependientes. Este marco se articula a través de la definición de distintos niveles de aplicación BIM, que proporcionan una hoja de ruta para la integración gradual de esta metodología en la gestión de la información de los contratos públicos, abarcando desde una fase inicial de reconocimiento y exploración de BIM hasta su integración completa en los procesos.

La etapa inicial, denominada **PreBIM**, se caracteriza por una falta de estrategia BIM específica. En este nivel, los procesos de trabajo todavía no incorporan prácticas BIM, no se aplican herramientas BIM específicas, y no se requiere formación BIM específica para el personal involucrado. Esta fase representa el punto de partida para la mayoría de las entidades que comienzan su camino hacia la adopción de BIM, donde el conocimiento y la aplicación de BIM aún son ciertamente limitados o inexistentes.

Avanzando hacia el **Nivel Inicial** que entró en vigor a fecha 1 de abril del 2024, las entidades comienzan a considerar y establecer objetivos BIM básicos. Los procesos comienzan a incorporar elementos BIM de manera simple, enfocándose en etapas tempranas como el diseño y la documentación básica. Las herramientas BIM básicas son adoptadas para facilitar estas tareas, y se introduce una formación BIM elemental para el personal, sentando las bases para una mayor integración de BIM en los proyectos.

Continua el **Nivel Medio** con proyección de implantación para el último trimestre de 2025, marca un avance significativo en la implementación de BIM. Se definen objetivos BIM más claros y estrategias para su implementación, con procesos BIM estructurados que fomentan la colaboración entre disciplinas. Las herramientas BIM utilizadas en este nivel permiten el modelado de información y la coordinación entre los equipos de proyecto. Además, se imparte formación y certificación BIM intermedia para roles clave, asegurando que el personal tenga las competencias necesarias para trabajar eficazmente con BIM.

Al alcanzar el **Nivel Avanzado**, la estrategia BIM ya se integra plenamente en la gestión del proyecto, con objetivos y métricas claras que guían su implementación. Los procesos BIM están completamente integrados en todas las fases del proyecto, desde la concepción hasta la ejecución. La tecnología BIM en este nivel incluye herramientas avanzadas para análisis y simulación, permitiendo una optimización del diseño y la construcción. El personal involucrado posee una alta capacitación en BIM, con roles y responsabilidades claramente definidos.

Finalmente, el **Nivel Integrado** representa la culminación del proceso de adopción de BIM. En esta etapa, la estrategia BIM se alinea completamente con los objetivos de negocio de la entidad, maximizando el valor del BIM a lo largo del ciclo de vida completo del activo. Los procesos BIM se optimizan y automatizan para una integración total, utilizando tecnologías BIM avanzadas y emergentes. La cultura BIM está plenamente integrada en la organización, con todos los empleados capacitados y comprometidos con los principios BIM.

Este enfoque gradual y estructurado hacia la adopción de BIM permite a las entidades públicas avanzar de manera coherente y sostenible hacia la plena integración de esta metodología en sus proyectos de construcción y gestión de activos, asegurando que todos los niveles de la organización se alineen con los objetivos de innovación y eficiencia que BIM promueve.

4 ESTADO ACTUAL DEL OPENBIM PARA INFRAESTRUCTURAS CIVILES

En el presente capítulo se examina el estado actual del BIM en España, con un enfoque particular en el concepto de OpenBIM, y en consecuencia el uso de Industry Foundation Classes (IFC) como estándar abierto para la representación de modelos de construcción digital, que representa la base sobre la que asienta la filosofía de un BIM abierto para todos. La adopción de estas tecnologías promete mejorar la interoperabilidad y la colaboración a través de un marco común que facilita el intercambio de información entre los diferentes agentes implicados en la construcción.

El Plan BIM España representa un esfuerzo significativo en esta dirección, estableciendo directrices que buscan integrar eficazmente el BIM en las prácticas de construcción y gestión de infraestructuras del país. El análisis de este plan revela su potencial para estandarizar procesos y mejorar la calidad y eficiencia de los proyectos.

El IFC, como fundamento del enfoque OpenBIM, se discute en detalle, destacando su papel en la promoción de un entorno colaborativo y no propietario para el sector. La relevancia del IFC se extiende especialmente al ámbito de las infraestructuras civiles, donde la nueva actualización del estándar IFC4.3 trae consigo mejoras sustanciales que son vitales para la adaptación y evolución del BIM en proyectos de infraestructuras.

Este capítulo proporciona una visión holística de cómo el OpenBIM y el IFC están configurando el paisaje de la construcción digital en España, analizando tanto los avances logrados como, por último, los desafíos pendientes.

4.1 Estándar de intercambio IFC

Según la BuildingSmart [10] el estándar IFC, Industry Foundation Classes, consiste en una **descripción digital de proyectos constructivos** , incluyendo edificación y obra civil, en un formato de intercambio de datos BIM de carácter abierto y neutral, desarrollado y mantenido por la asociación BuildingSMART.

Éste estándar, contenido en normativa internacional ISO 16739 [11], dispone de un amplio uso en la industria de la construcción para facilitar la interoperabilidad entre diferente software especializado y proporcionar una gestión coordinada de proyectos.

Figura 11 - Asociación sin ánimo de lucro, BuildingSmart (Fuente: BuildingSmart)

El IFC se utiliza para describir, intercambiar y compartir información sobre proyectos de construcción, incluyendo datos geométricos, atributos de los elementos, relaciones espaciales y semánticas, entre otros. Esta estandarización de datos es fundamental para garantizar la integridad, precisión y coherencia de la información a lo largo de todo el ciclo de vida de un proyecto tanto de edificación como infraestructuras civiles.

4.1.1 Fundamentos del estándar IFC

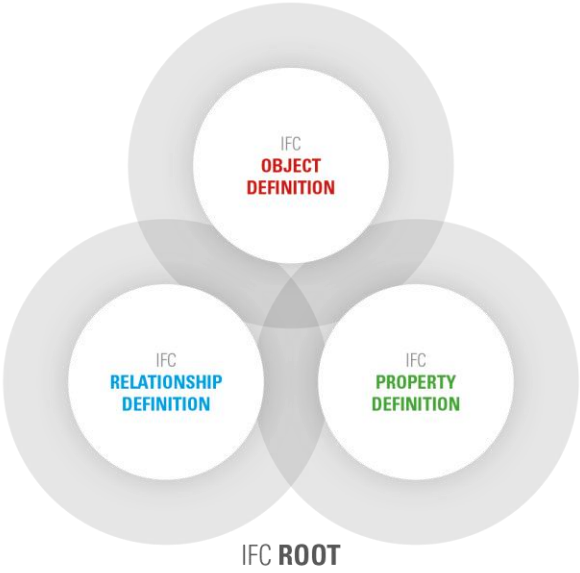
En lo respectivo a este capítulo, se ha acotado el término de arquitectura del estándar como referencia al formato genérico de IFC conocido como *STEP Physical File (SPF)*, como indica bSI [12], el formato IFC-SPF es el más usado en la práctica, está construido sobre el lenguaje representativo EXPRESS, definido en normativa estándar ISO 10303-21.

Los detalles técnicos del estándar IFC son de libre acceso, ofrecidos por bSI, es podible acceder a la documentación oficial de cada una de las versiones del estándar. Acorde a la fuente citada, se ha tomado como referencia principal para el presente trabajo de fin de grado la documentación de la nueva versión del estándar *IFC 4.3 ADD2* [13]. El acceso a esta documentación permite estudiar tanto la estructura como el significado de cada una de las componentes, como si de un diccionario IFC se tratase.

En términos de contenido un modelo IFC es una tecnología **orientada a objetos**, está formado por cientos de entidades ordenadas de forma jerárquica, generando así un gran entramado complejo entre todas ellas. De forma análoga a la teoría de color (RGB), la lógica con la que IFC genera este entramado y representa los distintos elementos descansa sobre tres conceptos fundamentales que componen la estructura que se define y acota cada uno de los elementos que forma parte del modelo abierto:

- **IfcObjectDefinition**: Define las distintas entidades (Objetos abstractos o tangibles).
- **IfcRleationship**: Define las relaciones entre las entidades existentes.
- **IfcPropertyDefinition**: Clasifica las propiedades asociadas a las entidades.

Figura 12 - Definición de una Entidad en el esquema IFC (Fuente: Elaboración Propia)



En consecuencia, cualquier elemento definido en un esquema IFC se puede reducir a un OBJETO que representa, dotado de unas PROPIEDADES que se le asocian y unas RELACIONES que ejercen de conectores entre ellos. Cabe destacar que estos pilares sobre los que se fundamenta la definición de un elemento están arraigados al concepto de *"IfcRoot"*.

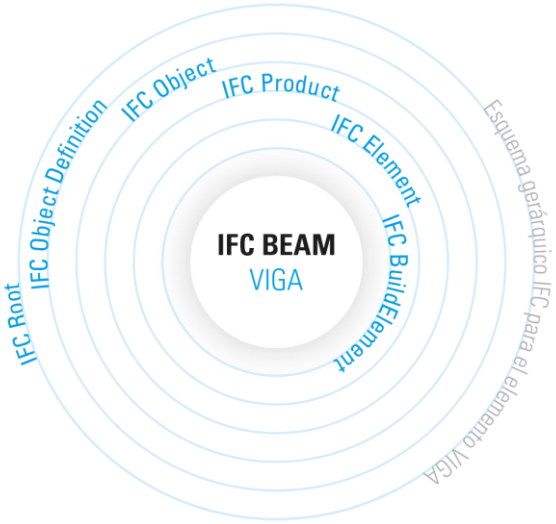
No es objeto de esta investigación profundizar excesivamente en la estructura del estándar IFC y sus términos informáticos, sino mostrar las implicaciones en el ámbito de las Infraestructuras. Será por ello por lo que no se tratará en profundidad la naturaleza y razón de ser de los términos de propiedades y relaciones de un IFC.

Asimismo, es procedente comprender la profundidad y complejidad intrínseca a la definición de la jerarquía de los objetos que componen un modelo, permitiendo así evaluar las nuevas implicaciones concebidas para el diseño de infraestructuras. Serán por tanto mencionados conceptos de relaciones y propiedades exponiendo parcialmente las ideas que interesan, para abordar el IFC en proyectos constructivos de infraestructuras, y no en su globalidad.

4.1.2 Jerarquía del IFC

El orden jerárquico que se ha establecido para cada uno de los conceptos que pertenecen al *"IFC schema"* permiten disponer de una estructura de datos ordenada y coherente con el contexto constructivo de cada una de las entidades. A modo de ejemplificación, se expone la descomposición del orden jerárquico que acota un elemento constructivo, en este caso uno de los elementos más comunes en la construcción, una viga conocida en el esquema como *"IfcBeam"*.

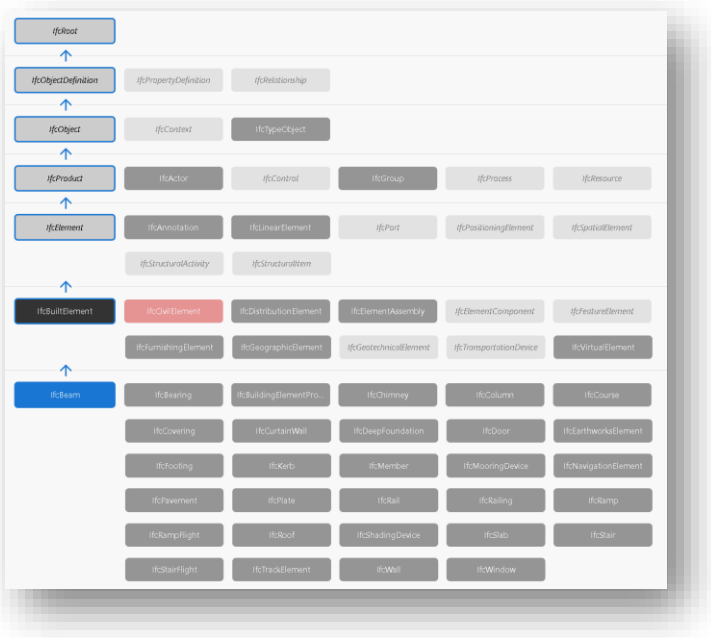
Figura 13 - Esquema jerárquico IFC para IfcBeam, elemento Viga (Fuente: Elaboración propia)



En la figura se pretende mostrar como cada entidad actúa como contenedor de la siguiente, sin ser exclusivamente el contenedor de esta, en este caso se representa el directorio dentro del entramado de la estructura del esquema IFC hasta llegar al elemento constructivo de viga.

La nueva actualización de la documentación *IFC 4.3 ADD2* [13], contiene dentro de cada instancia un esquema a modo de guía de navegación que facilita la comprensión del orden jerárquico de las entidades y todos sus participantes, permitiendo una comprensión dinámica de la etimología de la entidad IFC dentro del esquema.

Figura 14- Esquema Documentación elemento Viga (Fuente: Documentación IFC 4.3 ADD2, BuildingSmart)



Además, cabe destacar los tres elementos fundamentales de los que desciende la entidad constructiva objetivo del ejemplo, comentados anteriormente *"IfcObjectDefinition"*, *"IfcPropertyDefinition"* e "*IfcRelationship"*, todos descendientes de *"IfcRoot"*. Este caso, al ser un elemento constructivo tangible desciende de la entidad *"IfcObjectDefinition"* siendo este un objeto *"IfcObject"* que a su vez es catalogado como un producto *"IfcProduct"* seguido de un elemento *"IfcElement"* encargado de acotar su carácter físico y tangible, para finalmente calificarlo como un elemento constructivo *"IfcBuiltElement"*.

Asimismo, también está clara la directriz hasta el elemento viga y cómo las distintas entidades se contienen unas a otras.

4.1.3 Tipos Predefinidos

La importancia de este asunto reside en destacar que este no es el último nivel de profundidad en cuanto a la clasificación de elementos que el esquema IFC ofrece. Dentro de la sección *"Conceptos fundamentales y asunciones" de la* documentación IFC4.3 [13], es posible encontrar la definición de *Tipos predefinidos de Objeto.*

Muchas entidades de objetos tienen un atributo llamado *PredefinedType* (Tipos Predefinidos) siendo esta una enumeración específica de distinción más detallada para el elemento en cuestión. Este tipo predefinido proporciona esencialmente otro nivel de clasificación por herencia para diferenciar aún más los objetos sin necesidad de subentidades adicionales.

Estos tipos predefinidos son especificados bajo la selección del atributo que se desee dentro de la enumeración de tipos que se ha especificado para el elemento en cuestión, es decir los *PredefinedType* de la entidad.

En el caso de que se necesitase un valor personalizado, se usará la clase *IfcObjectType* para definir el nuevo *tipo*, mientras que en el atributo *PredefinedType* del objeto se establecerá como ***USERDEFINED***. Este valor e de vital importancia a la hora de dar de un sentido a los elementos que componen un modelo abierto IFC, la práctica de establecer tipos predefinidos enriquece en un nivel adicional de información el modelo y la acotación de los mismos, a pesar de que no se encuentre un tipo predefinido que satisfaga el concepto a describir, es de gran importancia.

Dentro de la instancia del elemento constructivo, previamente abordado de una viga *"IfcBeam"* en la documentación [13] es posible consultar su tipo predefinido, "*IfcBeamType"*. A través del acceso a su entrada descubrimos todos los tipos predefinidos que podremos usar para identificar elementos constructivos viga de forma más precisa en nuestro modelo.

Figura 15 - Tipos predefinidos del elemento viga (Fuente: Elaboración Propia)

Cabe destacar que la lista conjunta de tipos predefinidos no se encuentra en el término *"IfcBeamType"* en sí, sino que se puede consultar en el enumerador *"IfcBeamTypeEnum"* contenido dentro del primero mencionado.

4.1.4 Estructura Espacial

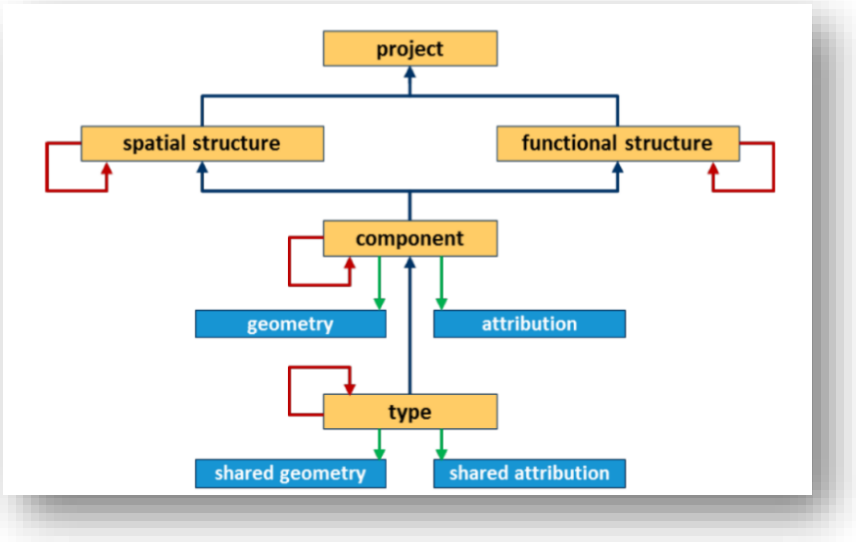
La estructura espacial de un IFC es un aspecto de especial interés a la hora de comprender la funcionalidad del estándar. Como dice Pszczolka [14], el sistema IFC usa una jerarquía espacial para organizar y posicionar sus entidades tangibles dentro de un proyecto constructivo. Este método para organizar y desglosar un proyecto en componentes más pequeños y manejables es conocido como *Estructura de Desglose Espacial*, de las siglas en inglés *SBS (Spatial Breakdown Structure*). Esto incluye la capacidad de vincular términos específicos, como elementos de carreteras, ferrocarriles y más, con una sección determinada del proyecto para mejorar la gestión y organización de los activos.

Estas ideas no precisan de mayor contextualización, puesto que carecen de carácter de novedad dentro del sector de la construcción. Así como la obra vertical siempre se ha clasificado por los niveles "IfcBuildingStore" que la componen es conocido que la obra lineal ha sido clasificada por los tramos longitudinales *"IfcFacilityPartCommon,SEGMENT"* que a su vez la componen, desglosando así el proyecto en medidas manejables para los equipos de ingenieros. La aplicación práctica de estos conceptos en el esquema IFC se aborda a lo largo de este capítulo.

De igual forma, es importante señalar que SBS difiere de otros tipos de estructuras de desglose, como la *Estructura de Desglose del Trabajo (EDT)* [15], que desglosa el proyecto en trozos más pequeños y manejables basados en el trabajo que hay que hacer, mientras que SBS divide el proyecto en partes más pequeñas y manejables en función de la ubicación. Esto permite a los equipos centrarse en zonas geográficas concretas, planificar y prepararse para los problemas relacionados con el trabajo en esas zonas.

Esto será de gran implicación a la hora de definir los proyectos de infraestructuras, como bien introduce la memoria del modelo conceptual para el dominio de carreteras de bSI [16] que será citado posteriormente.

Figura 16 - Estructura especial de un proyecto en IFC (Fuente: BuildingSmart International [16]***)***



El esquema espacial de un proyecto sienta sus bases en entidades responsables de la definición de espacios, es decir, contenedores espaciales. Esto conduce a la entidad padre de la que derivan el resto de las entidades responsables de acotar los espacios *"IfcSpatialStructureElement"*. E su entrada de la documentación oficial del estándar [17], se aborda el desglose canon que se sigue a la hora de definir los espacios en un proyecto.

Un *"IfcSpatialStructureElement"* es una categoría general que abarca todos los elementos espaciales utilizados para definir la estructura espacial de un proyecto. Esta estructura es fundamental para organizar y jerarquizar espacialmente los componentes de un proyecto.

Dentro de los elementos que conforman la estructura espacial de cualquier proyecto se incluyen:

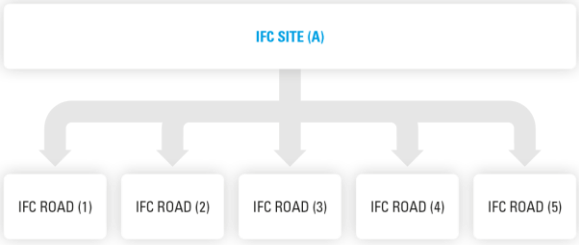
- **Emplazamiento**: Representado como **IfcSite**.
- **Infraestructura (Edificación u Obra Civil)**: Denominada **IfcFacility**, que puede desglosarse en:
  - **Edificio**: Como **IfcBuilding**.
  - **Puente**: Identificado como **IfcBridge**.
  - **Instalación Marítima (Puertos)**: Referida como **IfcMarineFacility**.
  - **Ferrocarril**: Denominado **IfcRailway**.
  - **Carretera**: Representada como **IfcRoad**.
- **Partes de la Infraestructura (Edificación u Obra Civil)**: Designadas como **IfcFacilityPart**, y que pueden incluir:
  - **Nivel o Planta**: Clasificado como **IfcBuildingStorey**.
  - **Segmento de la Instalación**: También bajo la denominación de **IfcFacilityPart**.
  - **Espacio**: Definido como **IfcSpace**.

Figura 17 - Descomposición espacial de un proyecto en IFC (Fuente: Elaboración Propia)

Este marco estructural de las distintas entidades espaciales permite una organización coherente y detallada de los componentes espaciales de cualquiera de las disciplinas de proyectos de infraestructuras mencionadas, facilitando su gestión y desarrollo.

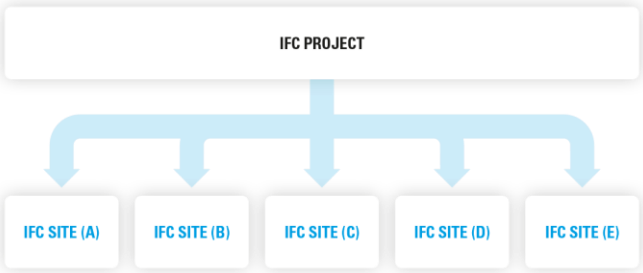
En este desglose cabe destacar se han enumerado las entidades espaciales introducidas en la nueva actualización del estándar dedicada a la gestión de proyectos de infraestructura, estos elementos serán abordados en detalle más adelante.

Figura 18 - Descomposición del proyecto en emplazamientos (Fuente: Elaboración Propia)



Además, bajo el concepto de jerarquía espacial, dentro de un mismo modelo es posible tener diversos emplazamientos, que a su vez se podrán descomponer en diversas instalaciones, véase edificaciones, carreteras, puentes, etc. Es decir, un proyecto no ha de disponer estrictamente de un único emplazamiento, es posible asignar varios emplazamientos distintos que a su vez se le aplica la misma filosofía a este, siendo posible su descomposición en variedad de instalaciones inclusive de la misma naturaleza.

Figura 19 - Descomposición de emplazamiento en instalaciones (Fuente: Elaboración Propia)

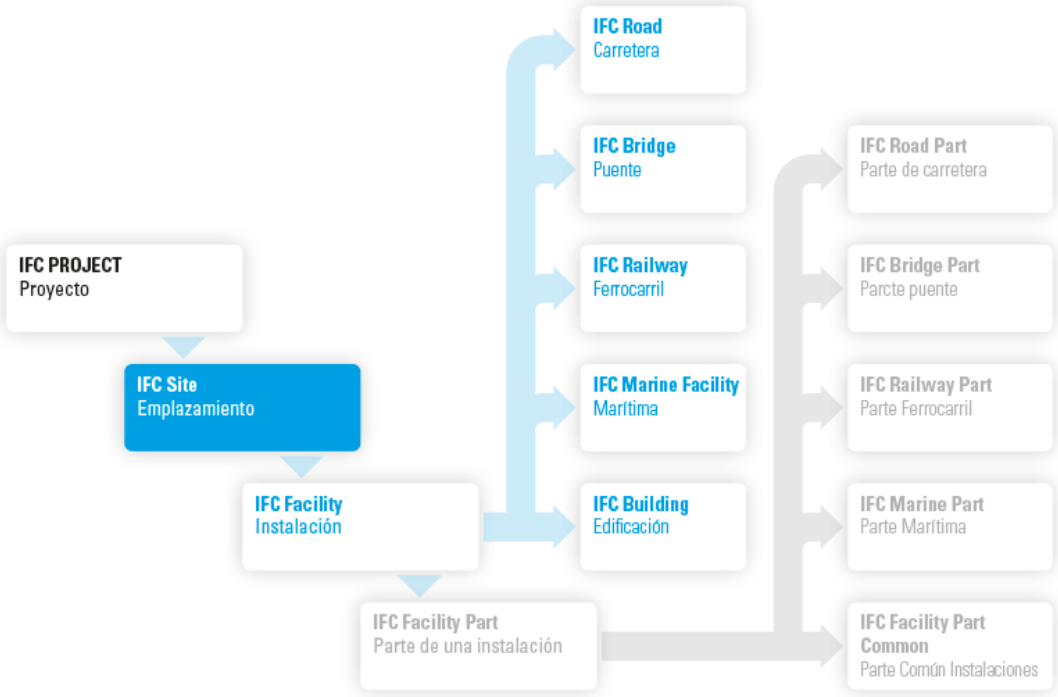


Cabe destacar que, dentro de la actualización del estándar, es posible dividir los activos construidos utilizando los nuevos objetos espaciales introducidos para los distintos dominios que se han propuesto. Ahora será de mayor viabilidad trabajar la estructura de los modelos abiertos para proyectos de infraestructuras mediante mejores objetos espaciales para los distintos dominios, véase carreteras, puentes, ferrocarriles e instalaciones marítimas. Las técnicas previamente mencionadas son aplicables al acotado de estas instalaciones y serán abordadas en detalle en el presente trabajo.

Además, de especial importancia será dotar a estas entidades espaciales de un tipo predefinido, acorde a los que ofrecen cada una de ellas. En la documentación oficial del estándar se posibilita el acceso a cada uno de los contenedores de enumeración como *"IfcFacilityPartCommonTypeEnum"* o *"IfcRailwayPartTypeEnum"*, entre otros, que contienen los tipos predefinidos considerados en el estándar. Estos serán abordados en detalle para la caracterización de los elementos espaciales de instalaciones dedicadas a infraestructuras.

Las posibilidades de descomposición espacial y definición que nos ofrece para las infraestructuras serán estudiadas a continuación materializando así la aplicación del estándar en su versión más reciente, siendo este la máxima expresión del estándar IFC en referencia a las Infraestructuras y sus componentes.

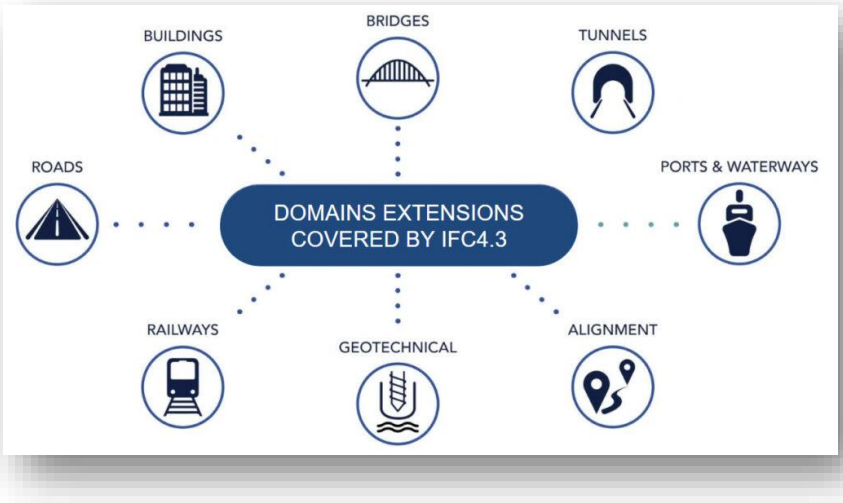
Figura 20 - Descomposición espacial con las nuevas entidades introducidas (Fuente: Elaboración Propia)



4.2 Aplicación del IFC para Infraestructuras Civiles

En lo respectivo al uso del estándar IFC en el ámbito de las infraestructuras civiles la, actualización IFC 4.3 ha sido un hito importante para la industria de la construcción y la gestión de infraestructuras, ofreciendo una plataforma estandarizada para el intercambio de modelos de información de construcción (BIM) en formatos abiertos. Aprobada como estándar final ISO este 4 de enero de 2024, IFC 4.3 se ha convertido en una norma internacionalmente acreditada, marcando un avance significativo en la adopción y uso de estándares abiertos en la industria.

Figura 21 - Extensiones de Dominio introducidas en IFC4.3 (Fuente: BuildingSmart International)



IFC 4.3 se centra, entre otros, en la extensión de los beneficios del IFC a lo que se cómo obra lineal, es decir infraestructuras que se extienden a través del paisaje en su dimensión longitudinal, como las carreteras y ferrocarriles. El alcance completo del programa establecido para IFC 4.3 incluye los dominios en proyectos de carreteras, ferrocarriles, puertos y canales, puentes y los elementos comunes de infraestructura entre ellos (movimiento de tierras, geotecnia, etc.), apoyando una amplia gama de casos de uso con el esquema IFC.

Figura 22 - Dominios disponibles en IFC4.3 (Fuente: BuildingSmart International)

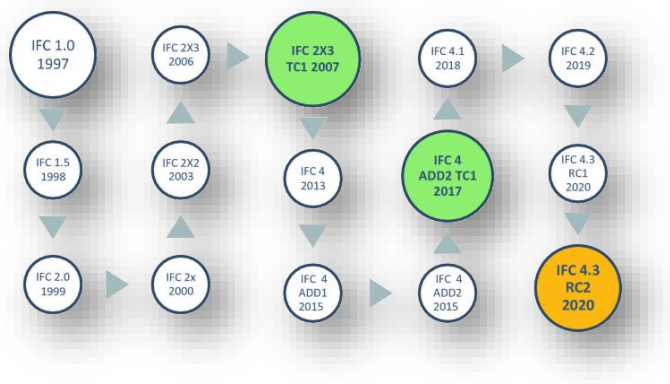
Este estándar representa el culmen de alrededor de más de 7 años de trabajo de proyecto concentrado y múltiples proyectos individuales dentro de la comunidad de bSI. Con una fuerte participación y liderazgo por parte de los clientes, especialmente los gubernamentales y los de infraestructuras, y una contribución de los participantes de la industria AEC de todo tipo, junto con el apoyo cercano y la colaboración de los proveedores de software, IFC 4.3 ha sido sometido a pruebas exhaustivas, convirtiéndolo en la versión IFC más probada jamás emitida.

Como se ha comentado, la versión IFC 4.3 impulsa considerablemente la digitalización de infraestructuras, incorporando entidades en el esquema abierto IFC y planeando extensiones adicionales, como las entidades de túnel, que se publicarán en un futuro como IFC 4.4.

4.2.1 Innovaciones y Mejoras

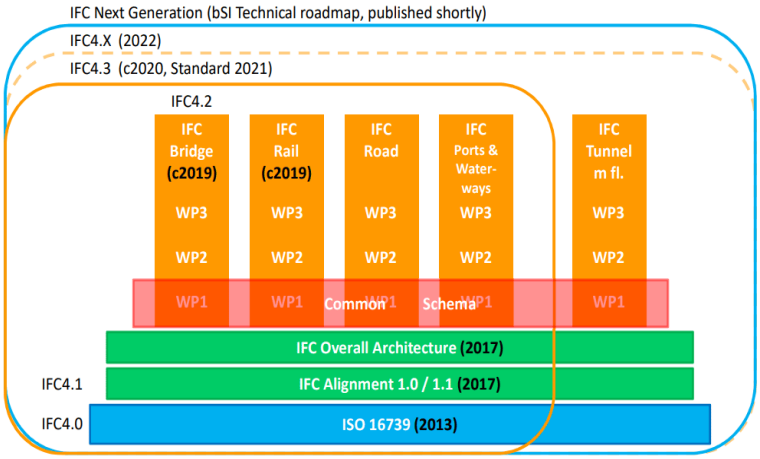
Para comprender el nuevo estándar IFC4.3, es necesario antes describir el camino que se trazó hasta alcanzar las mejoras que incorpora [18]. Desde la nueva versión IFC4 (2013) frente a IFC2x3 (2007), se fueron introduciendo las capacidades necesarias para soportar proyectos de infraestructura de cara a la posibilidad de incorporar la máxima expresión de los nuevos dominios en un futuro. En ese nuevo estándar IFC4 se introdujeron mejoras en la consistencia de los esquemas, optimización de archivos, soporte para cálculos energéticos y simulaciones avanzadas, estableciendo las bases para un intercambio de información más eficiente y preciso.

Figura 23 – Evolución de IFC (Fuente: IFC for Infrastructures: New Open Standards for Intelligent Data [19]*)*



Esta actualización introdujo significativas mejoras con el objetivo de potenciar la coherencia en todo el esquema, optimizar los archivos tras la incorporación de conjuntos de datos y facilitar la exportación e importación fluida de modelos IFC entre diferentes softwares para su desarrollo continuo. Esta versión se centró en la corrección de problemas técnicos identificados en IFC2x3, permitiendo un intercambio paramétrico más detallado de la forma, material y tipo de elementos constructivos. Con la inclusión de cantidades base estandarizadas (Qto) en la especificación IFC, se eliminó la variabilidad de conjuntos de propiedades (PropertySet) cuantitativas según el software origen, apoyando así cálculos energéticos y simulaciones avanzadas. Se mejoró la interoperabilidad BIM a GIS, se simplificaron las definiciones de tiempos de tarea, calendarios laborales y programaciones para trabajos abordando así con mayor facilidad las famosas dimensiones del modelado digital de la construcción BIM 4D, y se optimizó el uso de conjuntos de datos en 5D al vincular los valores de costo directamente a los ítems, agilizando así los modelos.

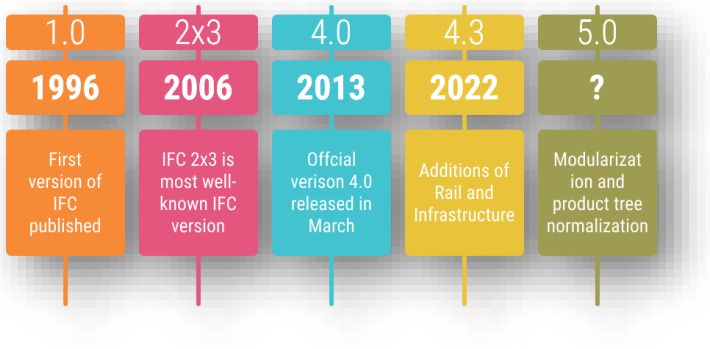
Figura 24 – Evolución de la concepción del IFC aplicado a Infraestructuras (Fuente: Building Smart)



Además, la actualización facilita el uso de idiomas nacionales y clasificaciones mediante la conexión de la definición de propiedades IFC con el Diccionario de Datos de buildingSMART (bSdd) y mejora significativa del rendimiento de modelos con curvas al introducir soporte para superficies y trazos b-spline, marcando el fin de la dependencia en geometrías poligonales para representar curvaturas.

Asimismo, no es hasta la versión IFC4.3 que representa un salto cualitativo en la evolución del esquema IFC, enfocándose especialmente en la ampliación de su aplicación a la infraestructura, incluyendo puentes, ferrocarriles, carreteras y el resto de los dominios que se han comentado. Este avance se caracteriza por el trabajo directo del equipo de desarrollo en las extensiones específicas para la más reciente versión 4.3, evidenciando un esfuerzo por abarcar una gama más amplia de aplicaciones en el sector de la construcción y la ingeniería civil.

Figura 25 – Versiones de IFC a lo largo del tiempo (Fuente: Plannerly)



Anteriormente, la falta de soporte para entidades dedicadas a infraestructuras limitaba severamente la modelización detallada de elementos como carreteras y ferrocarriles. A fecha del presente trabajo, la última actualización IFC4.3 introduce entidades específicas para estos elementos, lo que permite una representación más precisa y detallada de proyectos de infraestructura a gran escala. Una novedad significativa de la versión 4.3 es la depreciación de la entidad constructiva genérica *"IfcBuildingElementProxy"*, lo que refleja la confianza de buildingSMART en haber logrado una cobertura exhaustiva de todos los elementos constructivos dentro de las clases IFC disponibles. Esta decisión subraya el compromiso de IFC por proporcionar una representación más precisa y detallada de los componentes constructivos, eliminando la necesidad de *"proxies"* o elementos genéricos no específicos que anteriormente servían para llenar las lagunas en la representación de elementos complejos o no estándar.

Asimismo, cabe destacar que el elemento que hereda en el esquema la atribución de una entidad constructiva genérica será el de *"IfcBuiltElement"* siendo esta una entidad de alto nivel jerárquico que contiene el desglose del resto de elementos constructivos que componen el esquema. Esta entidad, al ser de carácter más global no está concebida para designar elementos constructivos en su detalle y por ello no contiene la posibilidad de definir tipos predefinidos que permitan al usuario establecer una caracterización personalizada *"USERDEFINED"* del elemento en cuestión. Es por tanto concluido que consiste en una mala práctica la asignación de la entidad genérica constructiva *"IfcBuiltElement"* a cualquier elemento del proyecto. Se recomienda en sustitución a esta práctica encontrar algún elemento constructivo acotado en el esquema, que incorpora ya una cantidad considerable, y aplicar un tipo predefinido personalizado por el usuario con el término que se encuentre necesario.

No obstante, cabe destacar que fue la versión anterior 4.2 de IFC cuando se avanzó considerablemente en la dirección de incluir la infraestructura dentro del espectro de aplicación de IFC. Con la adición del primer dominio de infraestructura dedicado a puentes mediante *"IfcBridge"*, fue cuando se iniciaron los dominios específicos dedicados a las infraestructuras dentro del esquema. Siendo así esta incorporación la que marcó el inicio de una expansión más amplia hacia la inclusión de diferentes tipos de infraestructura en el esquema IFC, preparando el camino para las extensiones más ambiciosas presentadas en la versión 4.3 objeto de este trabajo.

Es procedente comentar, que todos los cambios introducidos por las distintas actualizaciones del estándar están contenidos y enumerados de forma concreta y ordenada en el **Anejo F** [20] de la nueva documentación oficial de IFC4.3. Es por tanto recomendable para aquel que quiera conocer las nuevas posibilidades que nos ofrece desde un ámbito técnico la revisión de dicha documentación.

Figura 26 – Proceso de armonización para la concepción de un estándar IFC (Fuente: Elaboración Propia)

En definitiva, las innovaciones y mejoras introducidas por IFC4.3 en el ámbito de las infraestructuras civiles no solo superan las limitaciones previas en términos de representación geométrica e interoperabilidad, sino que también abren nuevas posibilidades para el diseño, análisis y gestión de proyectos de infraestructura a gran escala. La inclusión de entidades específicas para infraestructuras y la mejora en la precisión geométrica representan un salto cualitativo en la modelización BIM, promoviendo una mayor eficiencia y colaboración entre los diversos agentes involucrados en la construcción y mantenimiento de infraestructuras civiles.

Figura 27 - Novedades introducidas en la actualización IFC4.3 (Fuente: BuildingSmart International)

Domain	Extensions	Type Entities and Occurrences
Shared Infrastructure	Signalling	IfcSign__[Type]
		IfcSignal__[Type]
	Earthworks	IfcEarthworksCut
		IfcEarthworksElement
		IfcEarthworksFill
		IfcReinforcedSoil
	Above-ground works	IfcPavement__[Type]
		IfcCourse__[Type]
		IfcImpactProtectionDevice__[Type]
		IfcPlant
	Geotechnical	IfcGeotechnicalAssembly
		IfcGeotechnicalElement
		IfcGeotechnicalStratum
		IfcGeomodel
		IfcGeoslice
		IfcBorehole
		IfcSolidStratum
		IfcVoidStratum
		IfcWaterStratum
Railways	Spatial	IfcRailway
	Alignment	IfcAlignment2DCant,
		IfcAlignment2DCantSegLine
		IfcAlignment2DCantSegment
		IfcAlignment2DCantSegTransition
		IfcAlignment2DVerSegTransition
		IfcAxisLateralInclination
		IfcLinearAxisWithInclination
	Linear placement considering railway cant	IfcLinearPlacementWithInclination
	Create swept area solid	IfcDirectrixCurveSweptAreaSolid
		IfcDirectrixDistanceSweptAreaSolid
		IfcInclinedReferenceSweptAreaSolid
	Built element (track)	IfcRail[Type], IfcTrackElement[Type
	Distribution element	IfcDistributionBoard[Type],
		IfcElectricFlowTreatmentDevice[Type],
		IfcMobileTelecommunicationsAppliance[Type]
Roads	Spatial	IfcRoad
	Create sectioned surfaces	IfcSectionedSurface
	Create of profile definitions using widths and slopes	IfcOpenCrossProfileDef
	Association between a profile definition and its definition sources such as superelevation and width events	IfcRelAssociatesProfileDe
	Placement entity	IfcLinearSpanPlacement
	Built element type and occurrence entities	IfcKerb[Type]
Ports and Waterways	Spatial	IfcMarineFacility
	Element occurrence and type entities covering mooring	IfcMooringDevice[Type]
	Navigation element	IfcNavigationElement[Type]
	Distribution element occurrence and type entities covering Conveyor Systems	IfcConveyorSegment[Type]
	Liquid transport systems	IfcLiquidTerminal[Type]

4.2.2 Casos de Uso IFC 4.3 en Ingeniería Civil

Es de interés para este estudio, acotar la profesión a la que se accede a través del grado y las áreas de afección de dicha profesión. En España el grado de Ingeniería Civil, que dependiendo de la Universidad pueden tener diversas denominaciones (Tecnología en Ingeniería Civil, Ingeniería Civil y Territorial, Ingeniería de Tecnologías de Caminos), que permite acceder a la profesión de Ingeniero Técnico de Obras públicas. [21]

Ésta, es una profesión que distingue entre las distintas especialidades, construcciones civiles, hidrología y transporte y servicios urbanos, queda definida según:

- La Ley 12/1986, que estipula sobre la regulación de las atribuciones profesionales de los ARQUITECTOS e INGENIEROS TÉCNICOS, posteriormente modificada por la Ley 33/1992.
- Profesión Regulada de la Unión Europea para Ingeniero Técnico de Obras Públicas en su correspondiente especialidad.
- Orden CIN/307/2009, de 9 de febrero, por la que se establecen los requisitos para la verificación de los títulos universitarios oficiales que habiliten para el ejercicio de la profesión de Ingeniero Técnico de Obras Públicas.

Es por tanto comprendido que la Ingeniería Técnica de Obras Públicas asume una profesión con competencias parciales en función de la especialidad cursada (sólo podrá ejercer profesionalmente; o en construcciones civiles, o en hidrología, o en transportes y servicios urbanos) y que sólo puede desarrollar, según la jurisprudencia, trabajos de menor envergadura y complejidad frente a la profesión de Ingeniería de caminos canales y puertos.

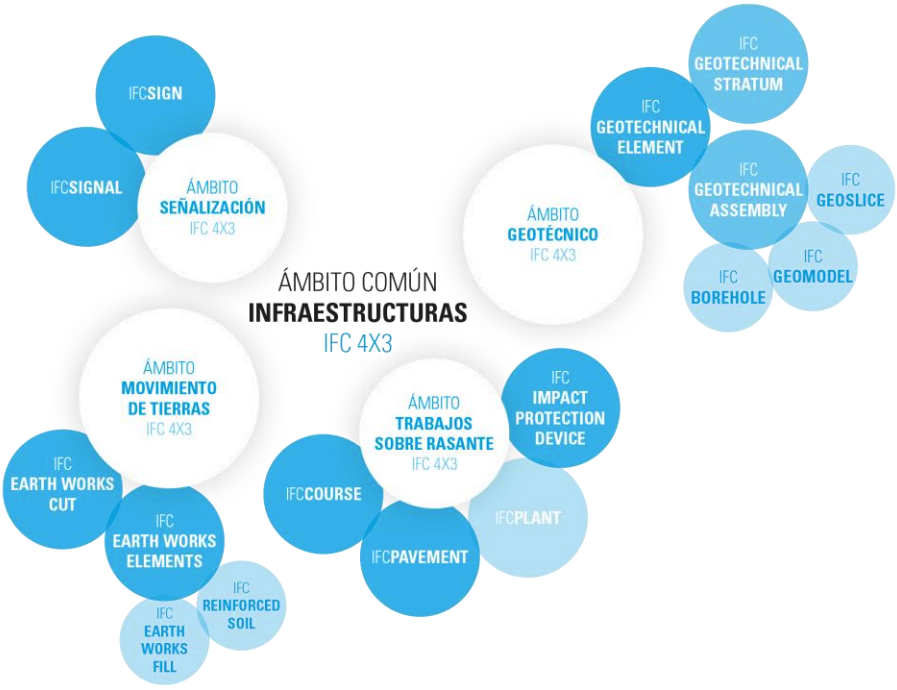
En relación con lo anterior, las **áreas de actuación profesional de la Ingeniería Civil** [22], entre otras, comprenden las **involucradas en los nuevos dominios que abarca el estándar abierto IFC4.3**. Queda por tanto justificada la imperante necesidad de los ingenieros civiles de adaptación hacia el entorno de tecnología BIM en la construcción, en gran medida como consecuencia del reciente Plan BIM España abordado con anterioridad.

4.2.2.1 Dominio Común de Infraestructuras

Como se ha introducido previamente, los elementos dentro de este esquema común a todas las infraestructuras responden a las necesidades que se identificaron dentro del estudio en cada proyecto de dominio que ha desarrollado las extensiones propuestas para áreas comunes específicas [23], véase estructuras/elementos espaciales, geotécnicos, movimientos de tierras, etc.

Asimismo, durante el proceso de armonización entre las distintas disciplinas en el marco único del estándar se identificaron muchos otros elementos compartidos que se reúnen en este paquete del esquema común. Así pues, todos ellos se tratan colectivamente como elementos de infraestructura compartidos. Dentro de las propuestas para el dominio de Infraestructura Compartida, podemos encontrar distintas instancias agrupadas en distintas extensiones (ver *Figura 27* - ***Novedades introducidas en la actualización IFC4.3 (Fuente: BuildingSmart International)***).

Figura 28 – Entidades del Ámbito de elementos comunes en Infraestructuras (Fuente: Elaboración Propia)



PAQUETE DE SEÑALIZACIÓN

Acorde a IfcSign [13], una señal es un aviso expuesto que da información o instrucciones de forma escrita, simbólica o de otro tipo. Los rótulos son pasivos y su forma más común es un panel pictórico.

Tabla 1 - Tipos predefinidos contenidos en IfcSignTypeEnum (Fuente: Documentación Oficial IFC4.3)

IfcSignTypeEnum	
MARKER	Tipo de señal formada por un poste vertical (posiblemente con algunas letras o símbolos) que suele utilizarse para delimitar la distancia o la ubicación de algún equipo.
MIRROR	Tipo de señal que proporciona información a través de una superficie de espejo reflectante.
PICTORAL	Tipo de signo formado por una placa plana con algunas imágenes escritas o simbólicas sobre ella.

Según IfcSignal [13], consiste en un dispositivo activo que transmite información o instrucciones a los usuarios, mediante una señal acústica, visual o una combinación de ambas. La principal diferencia con respecto a un IfcSign es que es una señal activa y, por tanto, un subtipo de IfcFlowTerminal que suele requerir conexiones de alimentación y datos para su funcionamiento.

Una instancia de IfcSignal representa un dispositivo de señalización singular en una unidad ensamblada más grande o en un sistema conectado, como un marco individual dentro de una señal ferroviaria, una sola unidad luminosa en un sistema de semáforos o una señal acústica o luminosa montada en una boya de navegación.

Tabla 2 - Tipos predefinidos contenidos en IfcSignalTypeEnum (Fuente: Documentación Oficial IFC4.3)

IfcSignalTypeEnum	
AUDIO	Tipo de señal formado por un dispositivo activo que transmite información emitiendo una señal acústica, como un pitido, un timbre, una bocina o un sonido explosivo.
MIXED	Tipo de señal formado por un dispositivo activo que transmite información de forma visual y sonora.
VISUAL	Tipo de señal formado por un dispositivo activo que transmite información de manera visual, como una luz, un grupo de luces o formas mecánicas móviles.

PAQUETE DE MOVIMIENTO DE TIERRAS

Siguiendo las directrices que indica IfcEarthworksElement [13], esta entidad engloba los tipos de elementos constructivo generados por las actividades de movimiento de tierras para realizar el desmonte, elevar el nivel del terreno en general o reforzar o estabilizar el suelo mediante algún método mecánico o químico.

Como es posible consultar en la figura anterior esquemática sobre dominio de elementos comunes a las infraestructuras, esta es la entidad parental en la que se descomponen dos elementos más:

IfcEarthworksFill [13], elemento constructivo generado por actividades de Movimiento de Tierras, representa el material que se coloca en un área determinada para elevar o nivelar el terreno en general.

Tabla 3 - Tipos predefinidos contenidos en IfcEarthworksFillTypeEnum (Fuente: Documentación Oficial IFC4.3)

IfcEarthworksFillTypeEnum	
BACKFILL	Relleno detrás de muros de contención u otras estructuras como muelles, detrás de estribos y puentes.
COUNTERWEIGHT	Terraplén construido a un lado de la estructura principal de la carretera para reducir el asentamiento de ésta.
EMBANKMENT	Elemento de movimiento de tierras de tipo predominantemente longitudinal, sin otro tipo particular asignado en función de su función en el firme o en la subrasante. - NOTA: Definición de la norma ISO 6707-1: Sección de movimiento de tierras, a menudo formada por corte o relleno, en la que el nivel del terreno acabado está por encima o por debajo del nivel del terreno original y cuya longitud suele superar ampliamente su anchura.
SLOPEFILL	Relleno del talud lateral colindante con la estructura de la carretera o relleno del talud posterior.
SUBGRADE	Tipo de elemento de movimiento de tierras que forma la estructura por debajo del pavimento y por encima del suelo natural. - NOTA, Definición de ISO 6707-1: Parte superior del suelo, natural o construido, que soporta las cargas transmitidas por la estructura suprayacente de una carretera, pista de aterrizaje o superficie dura similar. - NOTA, Definición de la AIPCR: Capa superior del suelo natural sobre la que se construye el pavimento.
SUBGRADEBED	Parte superior del suelo, natural o construido, que soporta las cargas transmitidas por la estructura suprayacente de una carretera, pista o superficie dura similar.
TRANSITIONSECTION	Sección de la subrasante para garantizar la consistencia de la rigidez y evitar asentamientos desiguales. La sección de transición puede aparecer, por ejemplo, entre: terraplén y estribo de puente; terraplén y estructura transversal; desmonte y túnel; terraplén y desmonte.

Según IfcReinforcedSoil [13], abarca el concepto de suelo reforzado o estabilizado por algún método mecánico o químico.

Tabla 4 - Tipos predefinidos contenidos en IfcReinforcedSoilTypeEnum (Fuente: Documentación Oficial IFC4.3)

IfcReinforcedSoilTypeEnum	
DINAMICALLY COMPACTED	Método que consiste en utilizar una apisonadora dinámica que suele dejar caer libremente un pesado martillo desde la altura, compactando el suelo y mejorando rápidamente la capacidad portante de los cimientos.
GROUTED	Método de inyección de lechada curable en las grietas o poros de una cimentación geotécnica para mejorar sus propiedades físicas y mecánicas.
REPLACED	Excavar el suelo blando en un rango determinado por debajo del terreno de cimentación y, a continuación, rellenar la zona con materiales de alta resistencia, baja compresibilidad y no corrosivos.
ROLLERCOMPACTED	Un tipo de método de compactación que adopta maquinaria de laminación, la laminación repetida y la vibración compactan el suelo de cimentación, aumentando la resistencia y disminuyendo la compresibilidad.
SUBCHARGEPRELOADED	Método que aplica carga a los cimientos para descargar el agua de los poros, y los cimientos se consolidan para mejorar la resistencia de los cimientos. Se descarga cuando la capacidad de carga alcanza el nivel requerido.
VERTICALLYDRAINED	Método para establecer medidas de drenaje vertical en los cimientos, de forma que se descargue el agua de los poros del suelo y se mejore la resistencia de los cimientos.

Según, IfcEarthWorksCut [13] Consiste en el vacío resultante de la modificación del terreno existente o de la estructura de la carretera por excavación o por otros medios de eliminación de material.

Tabla 5 - Tipos predefinidos contenidos en IfcEarthworksCutTypeEnum (Fuente: Documentación Oficial IFC4.3)

IfcEarthworksCutTypeEnum	
BASE_EXCAVATION	Excavación para sótanos de edificios, estribos de puentes o estructuras similares parcial o totalmente por debajo del nivel del suelo.
CUT	Excavación en la que el suelo o la roca situados por debajo de la capa superior del suelo se cortan a la profundidad necesaria para la construcción de instalaciones como carreteras y vías férreas. El material extraído puede utilizarse como relleno (IfcEarthworksElement) para terraplenes o para formar una superficie nivelada sobre la que construir.
DREDGING	Excavación bajo nivel freático para recuperar material o crear una mayor profundidad de agua.
EXCAVATION	Tipo genérico de excavación cuando no se especifica un tipo más preciso.
OVEREXCAVATION	Excavación que va más allá de la profundidad requerida para la construcción, con el fin de sustituir material inadecuado.
PAVEMENTMILLING	Retirada de material caducado de la parte superior del pavimento para sustituirlo por material nuevo.
STEPEXCAVATION	Retirada de la parte blanda del talud de la carretera existente, donde se excava en escalones, cuando se ensancha una carretera.
TOPSOILREMOVAL	Excavación en la que se corta o retira la capa superior del suelo que contiene material orgánico. La tierra vegetal extraída puede utilizarse como relleno (elemento del movimiento de tierras), por ejemplo, en caso de plantación.

TRENCH	Excavación cuya longitud supera ampliamente la profundidad y la anchura. La zanja se excava normalmente para cimientos en franjas o para servicios enterrados como drenaje o cableado.

PAQUETE DE TRABAJOS SOBRE RASANTE

Según IfcPavement [13], tipo de elemento construido en una carretera u otra zona pavimentada para proporcionar una superficie uniforme que soporte las cargas de vehículos o peatones, normalmente compuesto por varias capas. Acorde a la definición de ISO 6707-1: carretera, pista de aterrizaje o construcción similar por encima de la subrasante.

Tabla 6 - Tipos predefinidos contenidos en IfcPavementTypeEnum (Fuente: Documentación Oficial IFC4.3)

IfcPavementTypeEnum	
RIGID	Pavimento construido sustancialmente con hormigón de cemento.
FLEXIBLE	Pavimento con un revestimiento bituminoso y con una capa de base con o sin ligante hidrocarbonado.

Según IfcCourse [13], consiste en un elemento constructivo cuya longitud supera con creces su grosor y a menudo también su anchura, generalmente de un solo material colocado in situ sobre otro elemento construido horizontal o casi horizontal. Un elemento constructivo de capa *"course*, se distingue de un elemento de movimiento de tierras en que una capa es un material granular graduado (que puede estar ligado o no) que generalmente se procesa de alguna manera, mientras que los elementos de movimiento de tierras son estructuras basadas en tierra que pueden formarse mediante la retirada y el transporte de material del suelo en general.

Estructuralmente, una capa (IfcCourse) no tiene capacidad para soportar cargas en un tramo abierto, ni para ser retirada o reemplazada como una sola unidad. Ejemplos de capas incluyen:

- Capas de áridos graduados
- Capas de arena graduada
- Material ligado con cemento (CBM)
- Capas de asfalto

Tabla 7 - Tipos predefinidos contenidos en IfcCourseTypeEnum (Fuente: Documentación Oficial IFC4.3)

IfcCourseTypeEnum	
ARMOUR	Capa de áridos cuya función principal es proteger contra la erosión del material subyacente por el agua, por ejemplo, escollera. - NOTA, Definición de la norma ISO 21650: capa protectora de un rompeolas, malecón u otras estructuras de montículos de escombros compuesta por unidades de blindaje.
BALLASTBED	Capa compuesta de piedras rotas bajo las traviesas.
CORE	Una capa principal es la estructura interna masiva de estructuras agregadas.
FILTER	Capa intermedia cuya función principal es impedir el paso de materiales finos.
PAVEMENT	Capa perteneciente a una estructura de pavimento que forma una zona pavimentada o carretera.
PROTECTION	Capa cuya función principal es proporcionar protección contra la erosión y la socavación.

Según IfcImpactProtectionDevice [13], consiste en un dispositivo de protección contra impactos es un componente utilizado para proteger otros elementos construidos de daños cinéticos. Actualmente existen diferentes tipos de dispositivos de protección contra impactos:

- Un amortiguador de vibraciones utilizado para minimizar los efectos de las vibraciones en una estructura disipando la energía cinética. El amortiguador puede ser pasivo (elástico, de fricción, de inercia) o activo (en un sistema que utiliza sensores y actuadores).

- Un aislante de vibraciones es un dispositivo utilizado para minimizar los efectos de la transmisibilidad de las vibraciones en una estructura.
- Dispositivos de impacto que disipan la energía cinética de los elementos que impactan (como vehículos) mediante deformación o mecánica elástica.

Tabla 8 - Tipos predefinidos contenidos en IfcImpactProtectionDeviceTypeEnum (Fuente: Documentación Oficial IFC4.3)

IfcImpactProtectionDeviceTypeEnum	
BUMPER	Un parachoques es un objeto situado al final de la vía que impide el paso. Puede estar fijado a los raíles o al panel de la vía, o también puede ser un elemento natural (por ejemplo, roca, arena).
CRASHCUSHION	Definición de la norma EN1317-1:2010: dispositivo de absorción de energía del vehículo de carretera instalado delante de uno o varios peligros para reducir la gravedad del impacto. Definición de ISO6707-1: dispositivo de absorción de energía instalado delante de un objeto rígido para reducir la gravedad del impacto de un vehículo, (Barrera de impacto, US).
DAMPINGSYSTEM	Elemento elástico insertado entre la superestructura (vía y placa sobre vía en placa o lecho de balasto con balasto insertado) y la estructura del túnel (suelo del túnel). Algunos de los elementos elásticos tienen un efecto de desacoplamiento parcial entre la superestructura y el subsuelo debido a las vibraciones. Como sistemas de suspensión pueden utilizarse tanto muelles helicoidales como bloques de elastómero o tiras de elastómero.
FENDER	Dispositivo pasivo o activo formado por un amortiguador y un panel de impacto que se monta en el muelle para proteger del impacto de los buques.

Cabe destacar que la entidad *"IfcPlant"* propuesta en la memoria de dominio común a las infraestructuras [23], pero no aparece en la última actualización del IFC4.3 [13].

PAQUETE GEOTÉCNICO

Acorde a IfcGeotechnicalElement [13], esta entidad consiste en un supertipo abstracto para entidades geotécnicas, de tal forma que será el que contiene el resto de entidades geotécnicas que se presentan.

En el caso de IfcGeotechnicalAssembly [13], implica la representación del concepto abstracto de un modelo geológico y geotécnico, normalmente una interpretación, pero a veces creado directamente a partir de mediciones de penetración en el suelo.

El uso de un conjunto *"assembly"* es opcional, pero puede contener la metodología y la información sobre la incertidumbre. Dichos conjuntos incluirán tipos de entidad IfcGeotechnicalStratum y pueden incluir otros tipos de entidad como IfcPile, IfcSlab o IfcSensor para representar los equipos de recubrimiento, revestimiento o registro presentes. IfcBorehole o IfcGeoslice pueden tener una realidad física como peligro de construcción además de ser el soporte de los resultados interpretados. Los riesgos geológicos pueden asociarse a cualquier IfcGeotechnicalAssembly o IfcGeotechnicalStratum.

- Según IfcGeomodel [13], representación del concepto de modelo geológico y geotécnico volumétrico, generalmente una interpretación, pero a veces creado directamente a partir de mediciones de penetración en el suelo.
- Según IfcBorehole [13], consiste en una representación del concepto de un modelo geológico y geotécnico lineal, normalmente una interpretación, pero a veces creado directamente a partir de mediciones de penetración en el terreno.
- Según IfcGeoslice [13], consiste en una representación del concepto de modelo geológico y geotécnico plano seccional, generalmente una interpretación, pero a veces creado directamente a partir de mediciones de penetración en el suelo.

Por último, en lo referente al paquete geotécnico, el conjunto puede contener uno o varios estratos y otros elementos, como el recubrimiento y el revestimiento. Por lo que se toma en cuenta los subtipos contenidos de IfcGeotechnicalStratum que tendrán representaciones de forma hechas a partir de tubos rectos o doblados que reflejen el diámetro de perforación.

Según IfcGeotechnicalStratum [13], consiste en una representación del concepto de un rasgo geológico discreto casi homogéneo identificado con una forma de superficie superior sólida irregular.

Tabla 9 - Tipos predefinidos contenidos en IfcGeotechnicalStratumTypeEnum (Fuente: Documentación Oficial IFC4.3)

IfcGeotechnicalStratumTypeEnum	
SOLID	Representación del concepto de un rasgo geológico o superficial sólido discreto casi homogéneo identificado, incluidas discontinuidades como fallas, fracturas, límites e interfaces que no se modelan explícitamente.
VOID	Representación del concepto de un elemento geológico discreto relleno de aire, incluidas las cuevas y otros vacíos.
WATER	Representación del concepto de un elemento geológico o superficial discreto relleno de agua, incluidos lagos, ríos y mares.

Un estrato se representa como una entidad discreta, especializada (subtipificada) a partir de IfcElement. Un estrato puede dividirse en entidades más pequeñas si las propiedades varían a lo largo del estrato o, alternativamente, las propiedades pueden describirse con rangos numéricos limitados.

Un estrato puede llevar información sobre la forma física y su interpretación como Elemento Geológico (GML). Las representaciones de forma utilizadas deben corresponder al subtipo de IfcGeotechnicalAssembly en el que se encuentre.

4.2.2.2 Dominio de Ferrocarriles

Como se ha comentado a lo largo del capítulo, una de las facetas de mayor importancia introducidas en los nuevos dominios de infraestructura consiste en la jerarquización espacial de los nuevos elementos definidos. En el caso de una estructura de instalación ferroviaria *"IfcRailway"*, al igual que el resto consiste en una instalación *"IfcFacility"* que puede ser subdividida en elementos parciales de la instalación *"IfcFacilityPart"*.

Figura 29 - Entidades del Ámbito de Ferrocarriles (Fuente: Elaboración Propia)

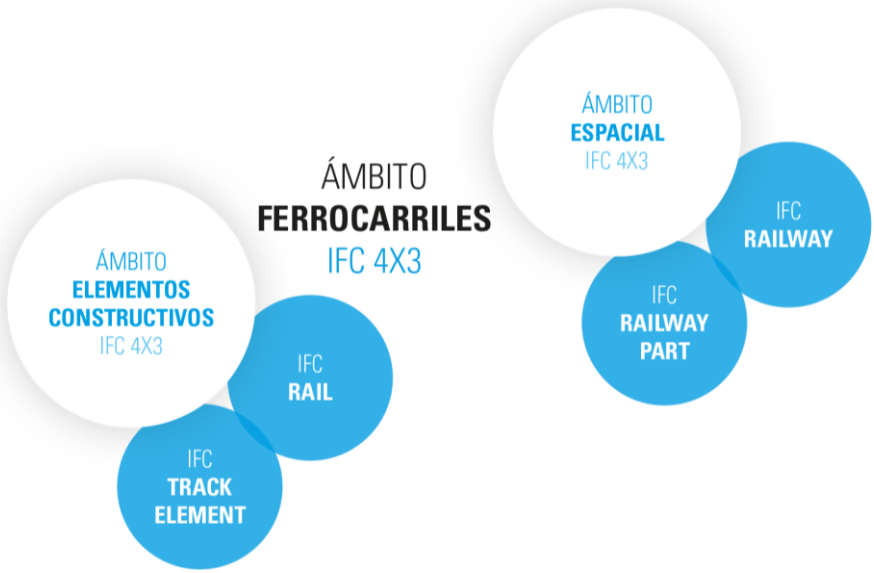
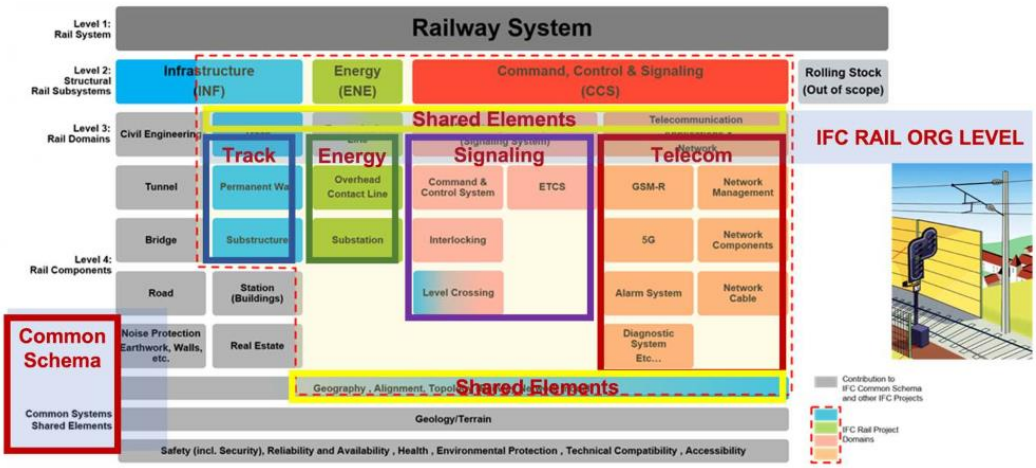


Figura 30 – Desglose de elementos IFC que componen un sistema ferroviario (Fuente: Building Smart)



Según IfcRailway [13], es un elemento de estructura espacial que sirve de ruta de un lugar a otro para el paso guiado de vehículos de ruedas sobre raíles. Actúa como un elemento básico de estructura espacial que ayuda a dividir un proyecto ferroviario en partes manejables, tramo de Obra Lineal.

Nota: El concepto está descrito acorde a la definición según ISO 6706: 2017: sistema de transporte nacional o regional para el paso guiado de vehículos de ruedas sobre raíles.

Además, acorde IfcRailwayPart [13], este consiste en una descomposición por partes de la estructura espacial IfcRailway. Destaca en sus tipos predefinidos:

IfcRailwayPartTypeEnum	
ABOVETRACK	Elemento de la estructura espacial que contiene elementos situados por encima o sobre la vía, por ejemplo, líneas catenarias y sistemas de suspensión.
DILATIONTRACK	Pista de Dilatación, no hay descripción disponible.
LINESIDE	Elemento de la estructura espacial que contiene elementos del ferrocarril que no están en o sobre las vías, por lo que se denomina "lado línea".
LINESIDEPART	Elemento de la estructura espacial que divide aún más una parte lateral de la línea. Puede utilizarse para dividir las partes laterales de la línea en volúmenes más manejables, con fines de ingeniería.
PLAINTRACK	Elemento de estructura espacial que divide una vía. No contiene ningún panel de desvío ni de dilatación.
SUBSTRUCTURE	Elemento de estructura espacial que contiene elementos situados por debajo de la vía, por ejemplo, la plataforma de movimiento de tierras, la subrasante preparada y el terraplén. Puede estar por encima o por debajo del nivel del suelo acabado.
TRACK	Elemento de la estructura espacial que contiene elementos relacionados con la vía, por ejemplo, carriles y traviesas.
TRACKPART	Elemento de la estructura espacial que sirve para dividir aún más una vía, para fines que no entran en estas categorías: vía plana, vía de desvío, vía de dilatación.
TURNOUTTRACK	Elemento de estructura espacial para dividir aún más una vía. Contiene desvíos y no contiene ningún panel de vía llana o de dilatación.

A continuación, se ofrecen algunas sugerencias sobre cómo pueden utilizarse las partes para organizar espacialmente una línea ferroviaria.

La línea ferroviaria puede estructurarse espacialmente a través de diferentes componentes clave. En primer lugar, se encuentra la **Vía (Track)**, que abarca no solo las vías planas, sino también aquellas de desvío y de dilatación, así como otras partes relacionadas con la infraestructura de la vía. Además, el **Lado de la vía (Line-Side)** incluye todos los elementos adyacentes a la vía principal, lo que permite una gestión integral del entorno ferroviario.

En lo que respecta a la **Subestructura (Substructure)**, se refiere al suelo construido que sustenta la vía, mientras que los **Elementos por encima de la vía (Abovetrack)** engloban todas las instalaciones o estructuras situadas por encima de la línea ferroviaria. La figura siguiente ilustra cómo estas partes pueden organizarse verticalmente dentro de un proyecto ferroviario.

PAQUETE DE ELEMENTOS CONSTRUCTIVOS (VÍA FÉRREA)

Según IfcRail [13], un raíl es un elemento constructivo predominantemente lineal que tiene un perfil de sección especial. El raíl se distingue de otros elementos de construcción con formas geométricas similares (por ejemplo, una viga o un travesaño) en que su función principal es garantizar el guiado de vehículos u otros tipos de maquinaria.

Tabla 10 - Tipos predefinidos contenidos en IfcRailTypeEnum (Fuente: Documentación Oficial IFC4.3)

IfcRailTypeEnum	
BLADE	Una cuchilla es un carril mecanizado, a menudo de sección especial, pero fijo y/o unido en el extremo del talón a un carril para proporcionar continuidad de apoyo a la rueda. Los dos carriles de aguja de un juego son los dos carriles interiores. Un carril de aguja se describe como derecho o izquierdo en función de si forma parte de un semijuego de agujas derecho o izquierdo. - Nota, definición de la norma EN 13232-1-2004.
CHECKRAIL	Un contracarril es un carril colocado cerca de la cara del ancho de vía de un carril de rodadura que participa en el guiado lateral de la rueda y evita el descarrilamiento en vías curvas de radio pequeño y en aparatos de vía. - Nota, definición de la norma EN 13481-1.
GUARDRAIL	Un guardarraíl es un carril que limita el riesgo de descarrilamiento de un tren, normalmente sin carga.
RACKRAIL	Un carril cremallera es un módulo de construcción para mejorar las prestaciones de tracción y rotura.
RAIL	Un raíl es una barra de sección especial (generalmente de acero) que asegura el guiado de la rueda de un material rodante u otras maquinarias pesadas. En ferrocarril, dos raíles se combinan para formar una vía.
STOCKRAIL	Un carril stock es un carril fijo mecanizado, que garantiza la continuidad en la vía principal o divergente con el interruptor en posición abierta. La parte mecanizada del carril original soporta su carril de cambio en la posición cerrada, dando continuidad a la línea a través de este carril de cambio. Los dos rieles originales en un conjunto de interruptores son los dos rieles exteriores. Un riel original se describe como derecho o izquierdo según sea parte de un medio juego de interruptores derecho o izquierdo. - Nota, definición de la norma EN 13232-1-2004.

Según IfcTrackElement [13], un elemento de vía es un elemento construido que se utiliza específicamente en el ámbito de la vía ferroviaria.

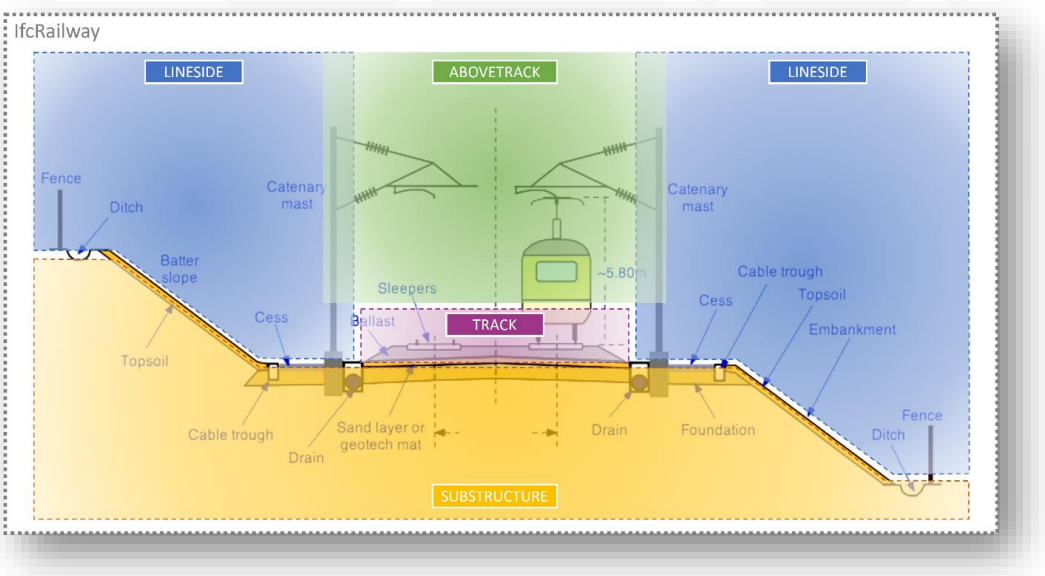
Tabla 11 - Tipos predefinidos contenidos en IfcTrackElementTypeEnum (Fuente: Documentación Oficial IFC4.3)

IfcTrackElementTypeEnum	
BLOCKINGDEVICE	Dispositivo compuesto de componentes neumáticos, mecánicos o eléctricos que provoca el frenado de un tren en caso de emergencia.
DERAILER	Dispositivo fijo que, colocado sobre el carril, hace descarrilar las ruedas de un vehículo y sirve para proteger una línea convergente. - Nota, definición de la norma IEC 60050-821.

FROG	Una rana es una disposición que asegura la intersección de dos bordes de rodadura opuestos de desvíos o cruces en rombo y que tiene una uve de cruce y dos carriles de aletas. - Nota: definición de la norma EN 13232-1-2004.
HALF_SET_OF_BLADES	Un medio juego de hojas consta de una contraaguja y su contraaguja completa con pequeños herrajes. Se encuentra a la derecha o a la izquierda según lo vea un observador situado en el centro de la vía mirando el talón de la aguja desde la punta de la aguja. - Nota: definición de la norma EN 13232-1-2004.
SLEEPER	Una traviesa es un elemento de la vía que soporta los carriles de rodadura, los guardarraíles y los contracarriles, normalmente en ángulo recto con su eje.
SPEEDREGULATOR	Dispositivo compuesto de componentes neumáticos, mecánicos o eléctricos que provoca la detención de un tren en caso de emergencia.
TRACKENDOFALIGNMENT	Un extremo de alineación de vía es una instalación funcional especial, como un punto de cambio de ancho de vía o un punto de carga de vagones de transporte.
VEHICLESTOP	Instalación fija situada al final de la vía que detiene cualquier movimiento de vehículos (por ejemplo, parachoques, joroba de arena, etc.).

ORGANIZACIÓN VERTICAL

Figura 31 - Ejemplo de uso de IfcRailwayPart para organizar verticalmente los elementos de una línea ferroviaria (Fuente: Documentación Oficial IFC4.3)



La figura muestra un ejemplo práctico de la clasificación de un proyecto ferroviario en elementos espaciales parciales de la instalación *"**IfcRailwayPart**"* para la **organización vertical** de los elementos en una línea ferroviaria. En situaciones más complejas, la parte inferior de la vía puede diferenciarse entre:

- **IfcRailwayPart.SUBSTRUCTURE,** para suelo construido
- **IfcFacilityPartCommon.BELOWGROUND** para suelo no construido.

Además, es importante considerar que otros escenarios podrían incluir líneas de ferrocarril sobre puentes, en túneles, en cruces con otras líneas, dentro de estaciones principales, o incluso en zonas con equipos de radiocomunicación situados en terrenos elevados.

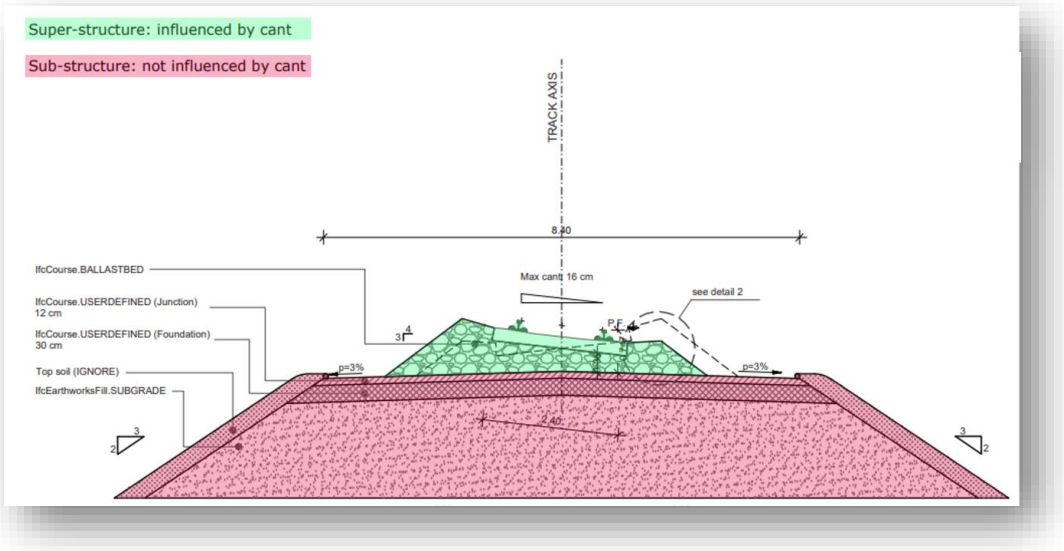
Por ejemplo, otros casos avanzados podrían ser:

- Línea de ferrocarril en un puente
- Línea de ferrocarril en un túnel (uno o varios tubos)
- Cruce de líneas de ferrocarril (dos o más líneas que se cruzan)
- Línea de ferrocarril en el contexto de un edificio de estación principal
- Línea de ferrocarril con sus equipos de radiocomunicación de apoyo (por ejemplo, equipos GSM-R, postes de antena) en una colina cercana.

ORGANIZACIÓN VERTICAL DETALLADA

Si el caso de uso lo requiere, la parte de la estructura espacial concebida para la subestructura de la línea ferroviaria *"IfcRailwayPart.SUBSTRUCTURE"* puede contener algunos elementos constructivos como *"IfcCourse"* o *"IfcEarthworksFill"* para distinguir las distintas capas de subestructura. En la figura siguiente se muestra un ejemplo.

Figura 32 - Ejemplo de capas y elementos que puede contener la parte SUBESTRUCTURA (Fuente: Documentación Oficial IFC4.3)



ORGANIZACIÓN LONGITUDINAL

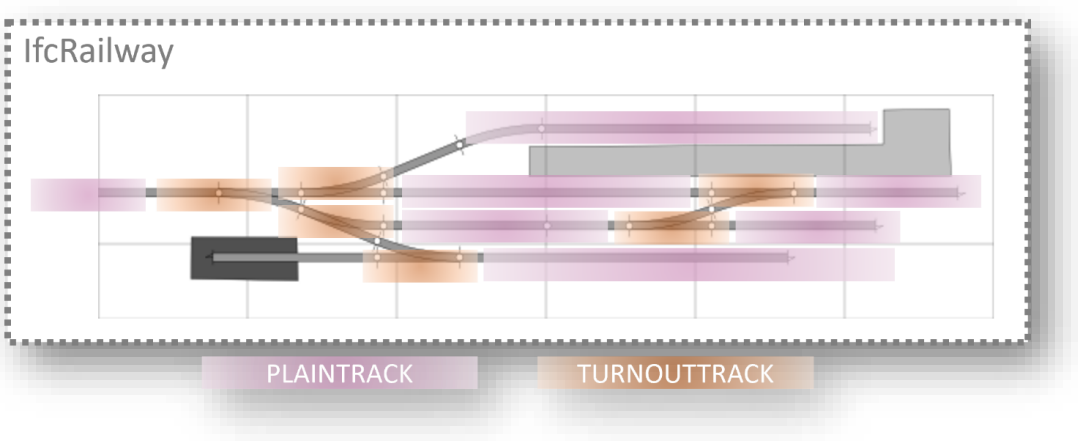
Las vías también pueden tener una organización longitudinal basada en las especificidades de la vía:

- PLAINTRACK para las vías de la línea de llanura.
- TURNOUTTRACK para la zona de desvíos.

Como se representa en el ejemplo de la figura siguiente. Otras partes, no incluidas en el ejemplo, pueden ser:

- DILATATIONTRACK para el área de los paneles de dilatación.
- TRACKPART para la organización longitudinal genérica de partes de la vía.

Figura 33 - Ejemplo de IfcRailwayPart para organizar longitudinalmente los elementos de una línea ferroviaria (Fuente: Documentación Oficial IFC4.3)



4.2.2.3 Dominio de Carreteras

En primer lugar, vamos a estudiar a modo de revisión, las aplicaciones actuales de las nuevas herramientas descritas sobre los proyectos de infraestructura viaria.

La documentación aportada por Building Smart en las memorias para el proyecto de carreteras [16] (IfcRoad), de la bSI InfrastructureRoom", nos ofrece un anejo [24] de ejemplos prácticos sobre el uso de las distintas clases definidas para el dominio.

Según la documentación, el anejo muestra ejemplos de cómo pueden utilizarse los nuevos conceptos propuestos junto con los conceptos IFC existentes para modelar diversas situaciones en el diseño de carreteras. No representan una forma estándar de utilizar estos conceptos, ya que puede haber varias formas, cada una de ellas útil en algún caso de uso. Asimismo, el hecho de presentar varias formas alternativas de modelar una misma situación (por ejemplo, mediante representaciones geométricas alternativas) no implica que todas las aplicaciones de software deban ser implementadas. Se utilizarán definiciones de vistas de modelos (MVD) para especificar qué alternativas deben admitirse en cada caso.

Figura 34 - Entidades del Ámbito de Carreteras (Fuente: Elaboración Propia)

Según IfcRoad [25], consiste en uta construida sobre tierra para permitir el desplazamiento de un lugar a otro, incluidas autopistas, calles, carriles bici y peatonales, pero excluidos los ferrocarriles. Como tipo de instalación, la carretera proporciona el elemento básico en la jerarquía de la estructura del proyecto para los componentes de un proyecto de carretera (es decir, cualquier empresa como el diseño, la construcción o el mantenimiento).

Nota: Definición de ISO 6707-1: Vía principalmente para vehículos.

Nota: Definición de la AIPCR: Línea de comunicación (vía transitada) que utiliza una base estabilizada distinta de carriles o pistas de aterrizaje, destinada principalmente al uso de vehículos motorizados de carretera que circulan sobre su propia rueda.

Acorde a IfcRoadPart [25], este consiste en una descomposición por partes de la estructura espacial IfcRoad. Destacando en sus tipos predefinidos para acotar la descomposición de la instalación de carretera:

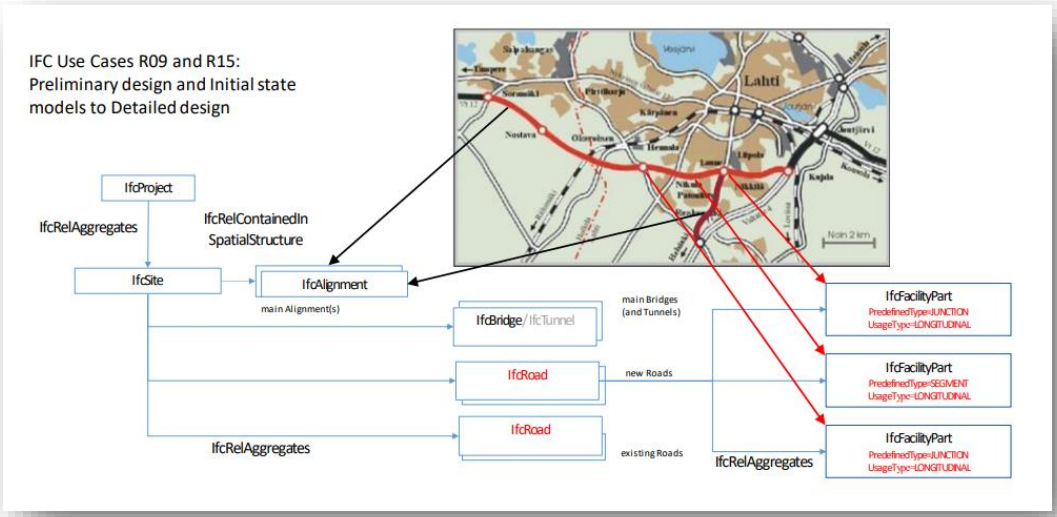
Tabla 12 - Tipos predefinidos contenidos en IfcRoadPartTypeEnum (Fuente: Documentación Oficial IFC4.3)

IfcRoadPartTypeEnum	
BICYCLECROSSING	Paso a nivel designado sobre una carretera para ciclistas.
BUS_STOP	Parte lateral de la carretera destinada a la parada de autobuses que les permite salir de los carriles de circulación y esperar durante breves periodos.
CARRIAGEWAY	Parte lateral unitaria de la carretera construida para el tráfico. La calzada puede comprender varios tipos de carriles de circulación y apartaderos, así como isletas, y en caso de carretera de doble calzada están separados por una mediana.
CENTRALISLAND	El centro de una glorieta no destinado al tráfico, puede estar pintado o levantado.
CENTRALRESERVE	Espacio lateral que separa dos calzadas de una misma carretera o que separa los carriles de circulación y la acera.
HARDSHOULDER	Tipo de arcén asfaltado que permite la circulación segura de vehículos en apuros.
INTERSECTION	Cruce a nivel en el que confluyen o se cruzan dos o más carreteras. Las intersecciones pueden clasificarse a su vez por el número de segmentos de carretera, los controles de tráfico y/o el diseño de los carriles.
LAYBY	Parte lateral de la carretera donde los vehículos pueden desviarse de la corriente ordinaria de tráfico.
PARKINGBAY	Parte lateral de la carretera destinada al estacionamiento de vehículos.
PASSINGBAY	Parte lateral de una calzada, ensanchada para permitir el paso de otro vehículo.
PEDESTRIAN CROSSING	Paso a nivel designado sobre una carretera para peatones.
RAILWAYCROSSING	Paso a nivel entre la carretera y el ferrocarril.
REFUGEISLAND	Plataforma elevada o zona protegida situada en la calzada para dividir los flujos de tráfico y proporcionar una zona de seguridad para los peatones.
ROADSEGMENT	Segmento longitudinal y lineal de una carretera, definido por características uniformes o como segmento de transición (por ejemplo, cambio de número de carriles).
ROADSIDE	Parte lateral de la carretera situada a lo largo de la misma, contigua a los bordes exteriores de los arcenes. Concepto general que comprende las zonas situadas fuera de la plataforma de la calzada no destinadas a los vehículos.
ROADSIDEPART	Concepto general que engloba varias partes del borde de la carretera.
ROUNDABOUT	Tipo de cruce a nivel en el que los flujos de tráfico se dirigen alrededor de un círculo.
SHOULDER	Parte lateral de la carretera adyacente a la calzada y normalmente al mismo nivel que ésta; no está destinada al tráfico rodado, pero puede utilizarse en caso de emergencia.
SIDEWALK	Senda peatonal a lo largo del lateral de una carretera. Puede presentar cambios moderados de pendiente (elevación) y normalmente está separada de la sección para vehículos por un bordillo. Puede haber una mediana o arcén entre la acera y los carriles de circulación.

SOFTSHOULDER	Tipo de arcén que no está pavimentado.
TOLLPLAZA	Parte de una carretera donde se cobra peaje por el uso de una vía de peaje, túnel o puente.
TRAFFICISLAND	Zona central o subsidiaria elevada o señalizada en la calzada, generalmente en un cruce de carreteras o en un paso a nivel, conformada y colocada de manera que dirija el movimiento del tráfico y/o proporcione refugio a los peatones.
TRAFFICLANE	Parte lateral de la calzada destinada al tráfico de vehículos para un fin determinado.

ESTRUCTURA ESPACIAL

Figura 35 - Esquema especial Anteproyecto (Fuente: BuildingSmart International)



Definición de la estructura jerárquica correspondiente a la fase de anteproyecto de un viario, en el cual podemos ver relación con puente/túnel que complementan la vía. Claro ejemplo sobre organización espacial básica a gran escala que incorpora esta nueva concepción de las infraestructuras IFC.

Figura 36 - Intersección de calles iguales (Fuente: BuildingSmart International)

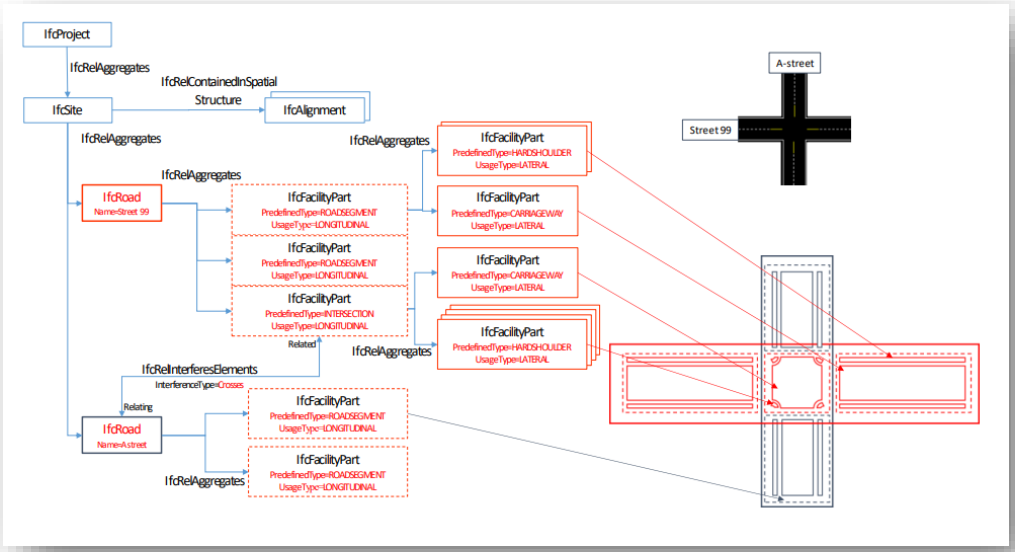


Figura 37 - Cruce de dos carreteras sin empalme, separación en desnivel (Fuente: BuildingSmart International)

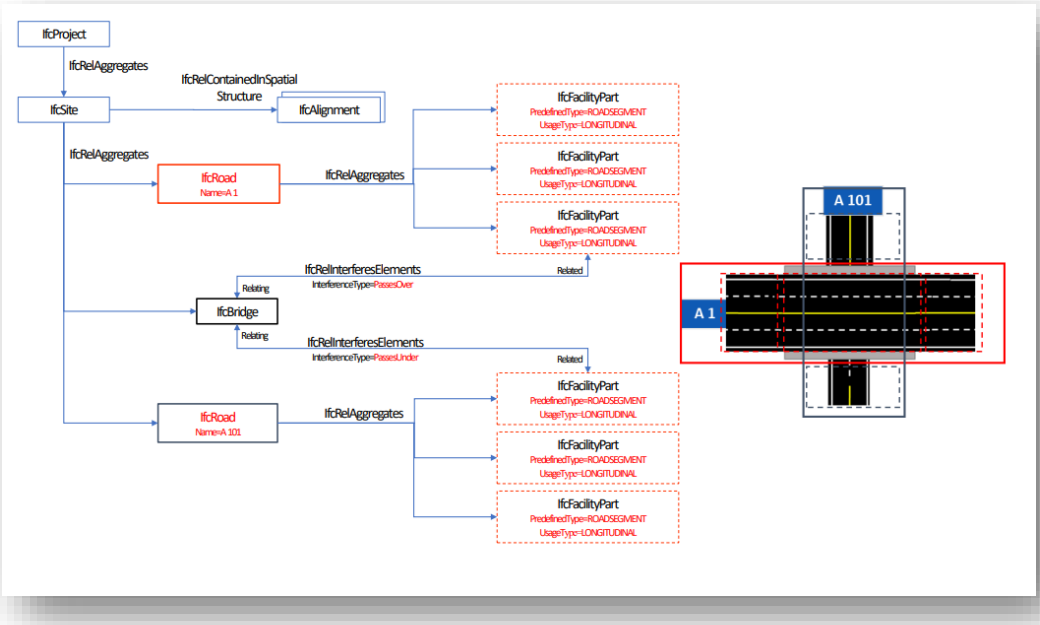


Figura 38 - Paso a nivel ferroviario (Fuente: BuildingSmart International)

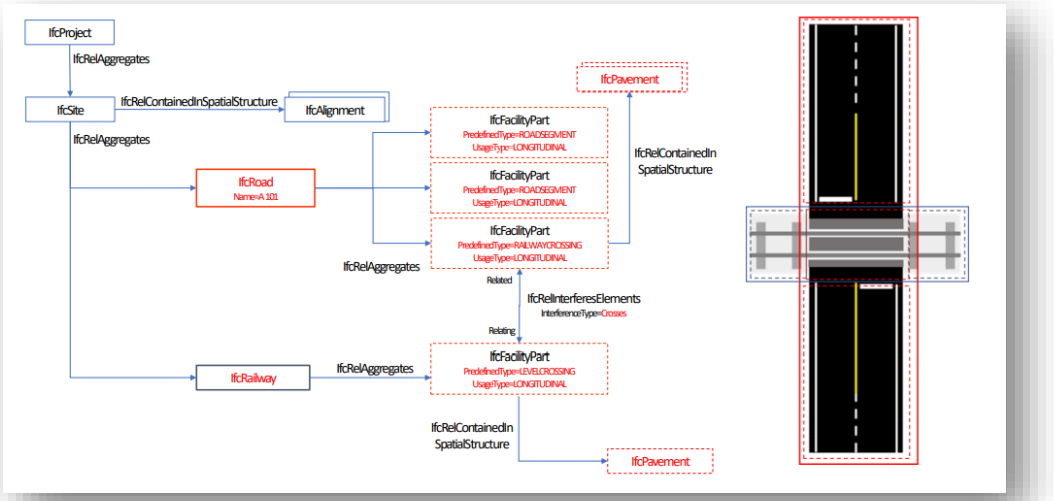
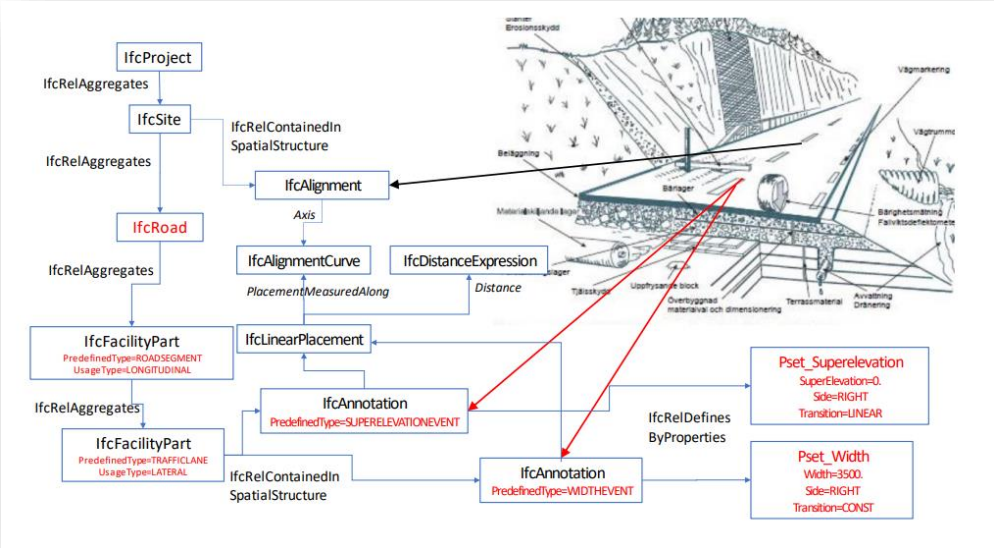


Figura 39 - Especificación del peralte y la anchura de un carril de circulación (Fuente: BuildingSmart International)



GEOMETRÍA DEL PAVIMENTADO

Los ejemplos de la documentación [24] muestran 5 formas alternativas de modelar el caso de un tramo de carretera con pavimento de dos capas.

Pavimento no descompuesto en partes (capas)

- Representación geométrica utilizando IfcSectionedSolidHorizontal, asociación de materiales mediante IfcMaterialProfileSet
- Representación geométrica mediante la propuesta IfcSectionedSurface e IfcOpenCrossProfileDef, asociación de materiales mediante IfcMaterialProfileSet.

Pavimento descompuesto en 2 partes (distintas capas como IfcCourse)

- Representación geométrica mediante IfcSectionedSolidHorizontal, asociación de materiales mediante un único IfcMaterial para cada IfcCourse.
- Representación geométrica mediante la propuesta IfcSectionedSurface e IfcOpenCrossProfileDef, asociación de materiales utilizando un único IfcMaterial para cada IfcCourse
- Representación geométrica mediante IfcGeometricCurveSet, asociación de materiales utilizando un único IfcMaterial para cada IfcCourse

Asimismo, cabe destacar que también son posibles otras representaciones geométricas, por ejemplo, utilizando superficies TIN o BREP.

Figura 40 - Pavimento con perfiles de material y representación del cuerpo (sólido) (Fuente: BuildingSmart International)

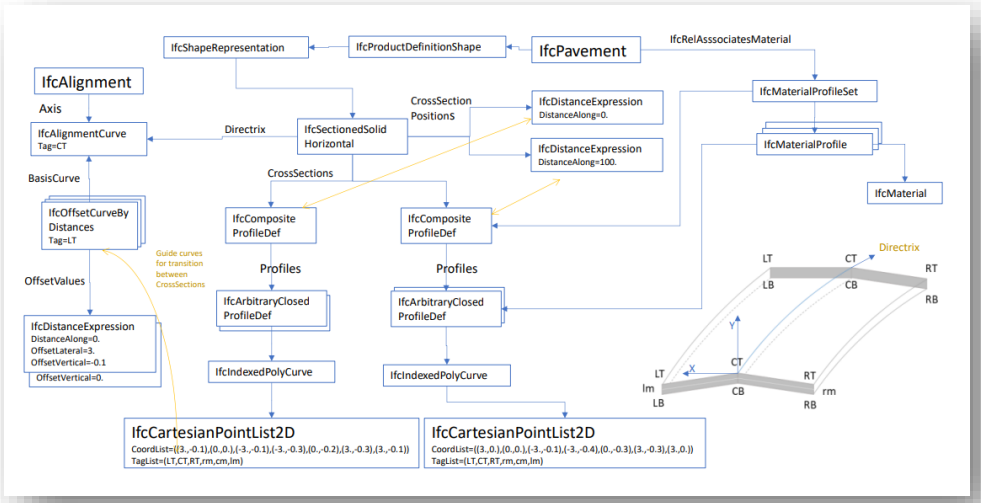


Figura 41- Pavimento con perfiles de material y representación de superficies (Fuente: BuildingSmart International)

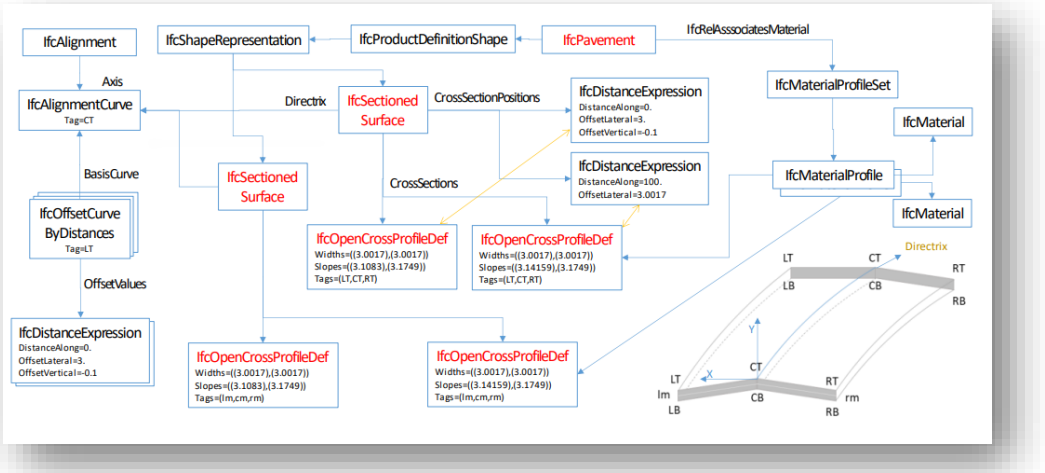
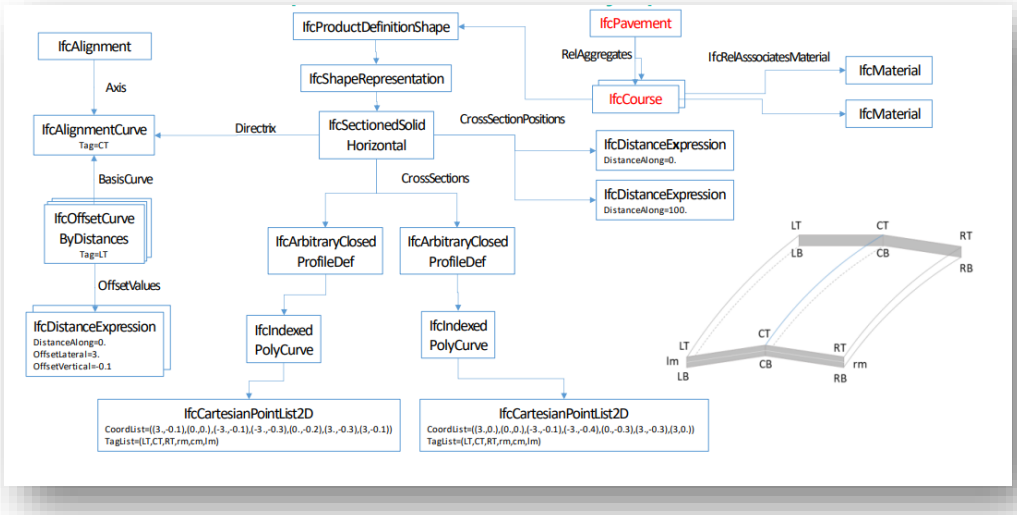


Figura 42 - Pavimento descompuesto en capas con representación del cuerpo (sólido) (Fuente: BuildingSmart International)



Además, también se podrá definir el pavimento descompuesto en capas mediante representación de superficie y con alineación.

MOVIMIENTO DE TIERRAS Y ANOTACIONES

Figura 43 - Representación de Movimiento de Tierras (Fuente: BuildingSmart International)

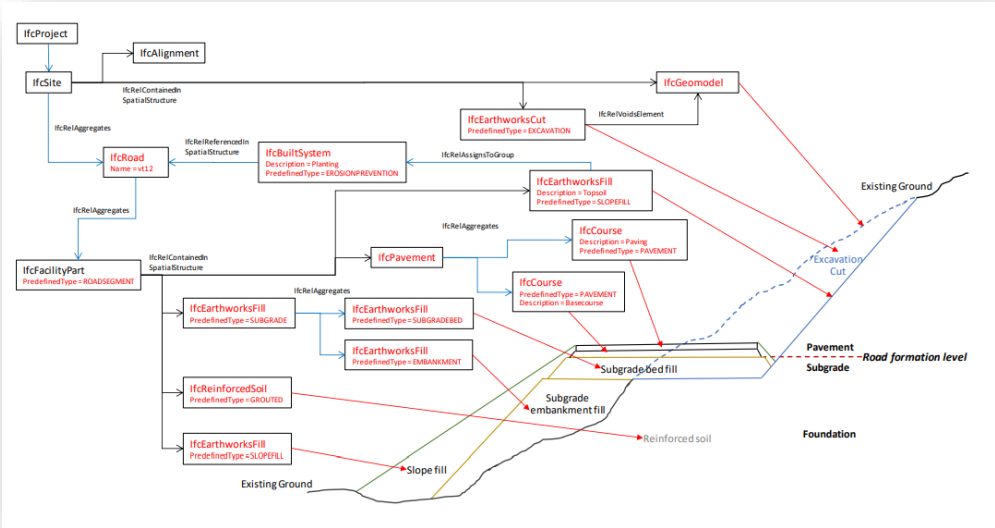


Figura 44 - Anotaciones geotécnicas del proyecto (Fuente: BuildingSmart International)

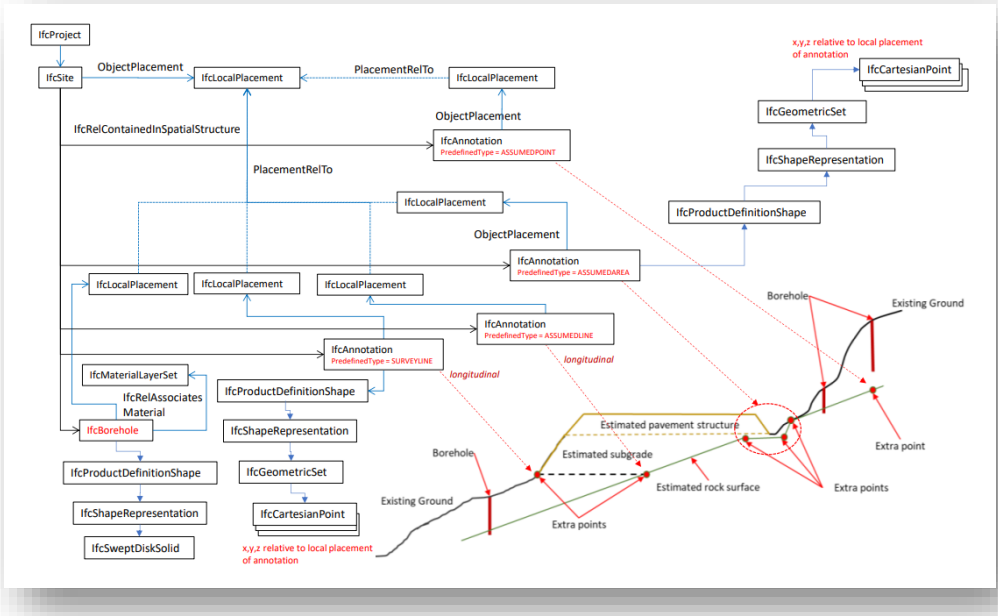


Figura 45 - Ejemplo de Guardarraíl (Fuente: BuildingSmart International)

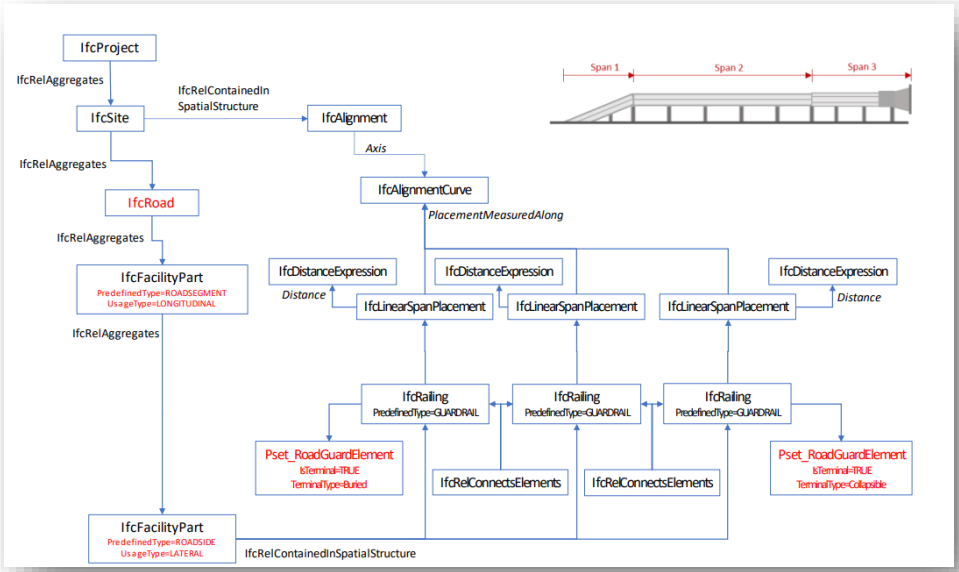
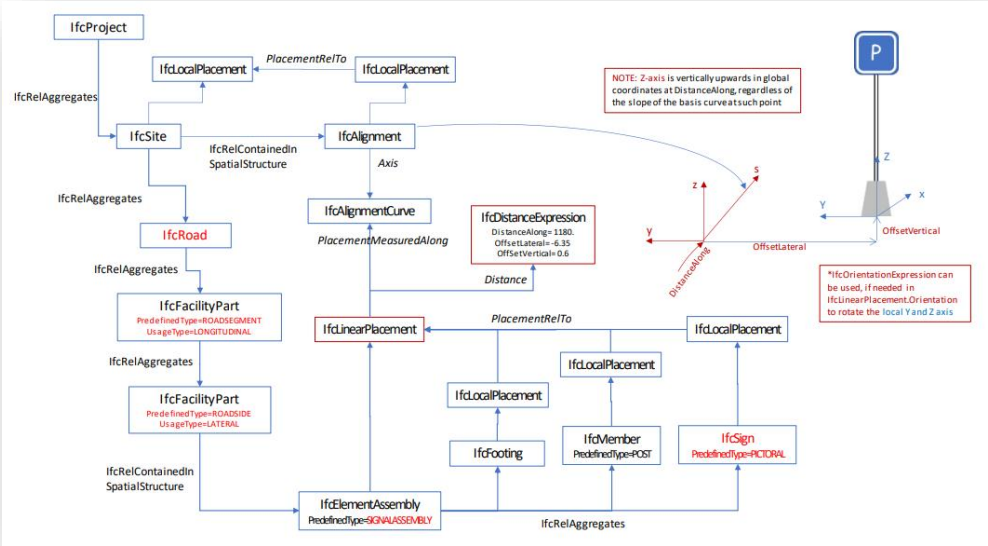


Figura 46 - Ejemplo de Señalización (Fuente: BuildingSmart International)



4.2.2.4 Dominio de Puentes

El dominio de puentes *"IfcBridge"* fue el primero de los dominios de infraestructuras en incorporarse al esquema IFC en la actualización IFC4.2 [26]. Asimismo, también se emitieron memorias sobre el desarrollo del proyecto [27] que serán tomadas como referencia.

Este dominio apenas tuvo que tomar acción en lo respectivo a incorporar elementos constructivos, se añadieron diversos tipos predefinidos a elementos constructivos ya existentes como son las Vigas *"IfcBeam"*, vistos al inicio del capítulo, o los muros *"IfcWall"*, en los que se añadió el *PredefinedType* para muros de contención (RETAININGWALL) por poner un ejemplo.

Todos los cambios en las versiones IFC4 pueden ser consultados en el **Anejo F** *"Change Logs",* de la documentación oficial [13].

Dentro de los cambios propuestos [27] encontramos:

Elementos Espaciales:

- IfcFacility – subtipo de SpatialStructureElement
- IfcFacilityPart - subtipo de SpatialStructureElement
- IfcBridge - subtipo de Facility
- IfcBridgePart - subtipo de FacilityPart

Elementos constructivos:

- IfcBearing - subtipo de IfcBuildingElement
- IfcDeepFoudation - (abstracto) subtipo de IfcBuildingElement
- IfcCaissonFoundation - subtipo de IfcDeepFoudation
- IfcVibrationDamper - subtipo de IfcElementComponent
- IfcTendonConduit - subtipo de IfcReinforcingElement

En lo respectivo a Sistemas, se ampliaron los conceptos para agrupar objetos con el fin de describir un sistema o un grupo de activos. En este módulo no se proponen nuevos conceptos tipos predefinidos para el concepto existente de IfcBuildingSystem

- REINFORCING
- PRESTRESSING

Figura 47 - Jerarquía especial de infraestructura IfcBridge (Fuente: BuildingSmart - IfcBridge Conceptual Model)

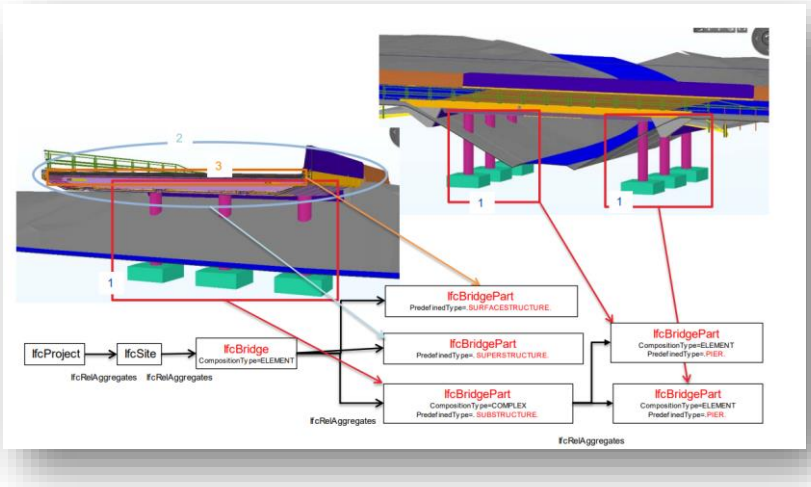


Figura 48 - Contenedores espaciales dentro del dominio IfcBridge (Fuente: BuildingSmart - IfcBridge Conceptual Model)

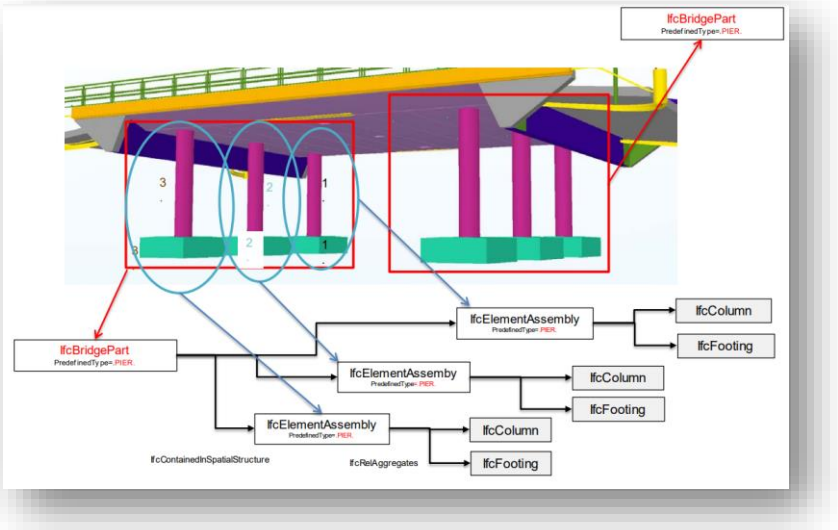


Figura 49 - Ejemplo de apoyos para infraestructura de puente (Fuente: BuildingSmart - IfcBridge Conceptual Model)

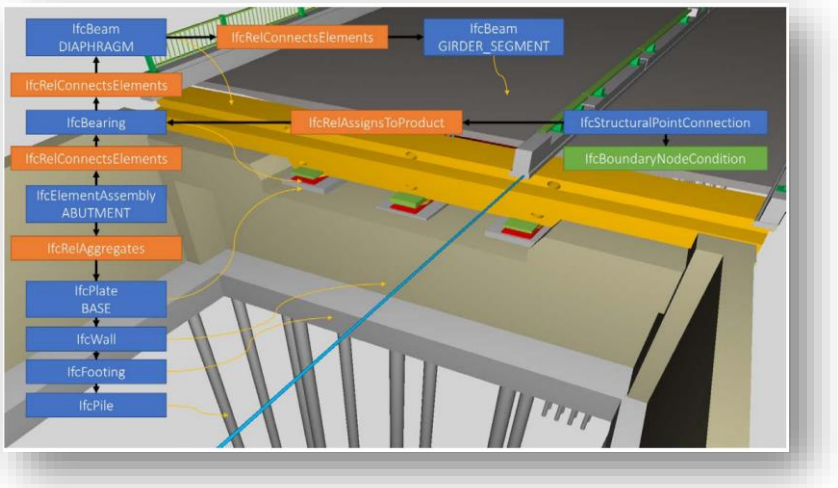


Figura 50 - Ejemplo de elementos para infraestructura de puente (Fuente: BuildingSmart - IfcBridge Conceptual Model)

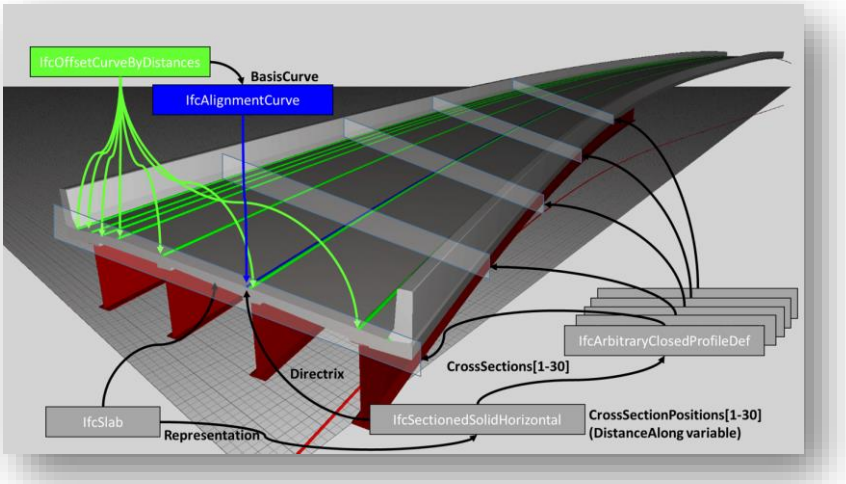
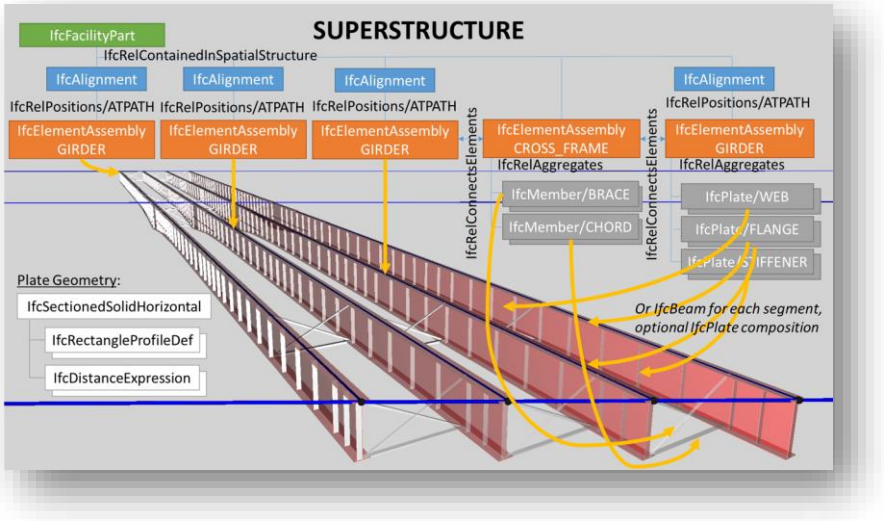


Figura 51 - Ejemplo de desglose espacial para infraestructura de puente (Fuente: BuildingSmart - IfcBridge Conceptual Model)



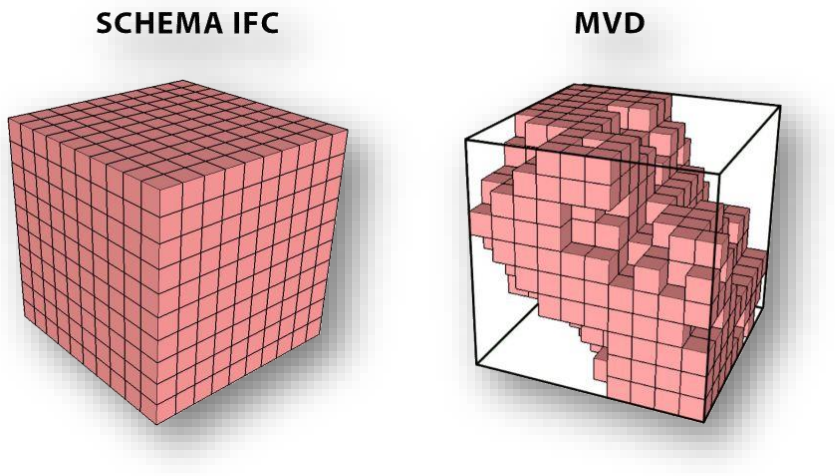
4.2.3 Herramientas y Software Compatibles

Se examinan las herramientas y soluciones de software compatibles con la versión IFC 4.3, poniendo especial énfasis en aquellas relevantes para las infraestructuras civiles. La compatibilidad con IFC 4.3 es un indicador clave de la capacidad del software para participar en flujos de trabajo de modelado de información de construcción (BIM) multidisciplinarios y de infraestructura civil, abarcando desde el diseño hasta el análisis y la gestión de proyectos.

La certificación de buildingSMART juega un papel crucial en este contexto, determinando la eficacia con la que el software maneja la exportación o importación de modelos propietarios al formato IFC, definida por la Model View Definition (MVD). La MVD especifica subconjuntos del esquema IFC destinados a satisfacer los requisitos de intercambio de información de diferentes disciplinas y usos del modelo. La certificación indica que el software ha superado la mayoría de los escenarios de prueba, aunque no garantiza una transferencia de modelos sin pérdidas o disturbios menores.

Con IFC2x3, se inició con una MVD oficial para la colaboración multidisciplinaria, la "Coordination View", pero a lo largo de los años, se han creado numerosos "add-ins" MVD no oficiales. La "Coordination View" original tenía como objetivo apoyar la coordinación entre diferentes disciplinas, pero se ha extendido para mejorar el soporte al traslado de diseño de archivos IFC entre softwares. Esto llevó a la creación de la "Coordination View 2.0", ahora estándar para la definición de exportación. [18]

Figura 52 - Representación del MVD (Fuente: BIM Manager, M. Baldwing [6]***)***



Para IFC4, se buscó separar completamente los propósitos de coordinación y transferencia de diseño, creando dos MVD: la "Reference View" para coordinación y la "Design Transfer" para transferencia de diseño. La certificación se ha hecho más estricta, requiriendo plena conformidad con la MVD correspondiente. Esto ha llevado a resultados de exportación de IFC más homogéneos y, por ende, a modelos IFC de mayor calidad, aunque también ha complicado la obtención de certificados para los desarrolladores de software.

En cuanto a software compatible con IFC 4.3, es esencial identificar aquellos que no solo han logrado la certificación de buildingSMART, sino que también demuestran un fuerte enfoque en el diseño, análisis, y gestión de proyectos de infraestructura.

Identificar y evaluar estas herramientas requiere una investigación detallada de sus certificaciones, capacidades específicas para manejar proyectos de infraestructura civil y la facilidad de integración en flujos de trabajo BIM colaborativos. Por parte de buildingSMART International, disponen de una lista pública con las certificaciones oficiales otorgadas a cada casa de Software en función de las implicaciones que hayan demostrado asumir con el estándar abierto, además se podrá acceder en algunos casos a los reportes de revisión que se llevaron a cabo para otorgar la certificación.[28]

4.2.4 LandXML

LandXML (Land eXtensible Markup Language) consiste en un formato de datos basado en XML ampliamente utilizado en la industria de la ingeniería civil y la topografía para el intercambio de información detallada sobre proyectos de infraestructura [29]. Este estándar se desarrolló para permitir la interoperabilidad entre diferentes sistemas de software, facilitando la transferencia y reutilización de datos geoespaciales y de ingeniería en proyectos de diseño, construcción y gestión de infraestructuras como carreteras, puentes, parcelas de terreno, y redes de servicios públicos. Este es uno de los estándares más famosos dentro de la industria, por lo que procede abarcar su contenido para finalmente ejercer una comparativa en retrospectiva de lo que ofrece en la actualidad la versión IFC dedicada a las infraestructuras

LandXML fue introducido a principios de los años 2000 como una solución para los desafíos de interoperabilidad que enfrentaban los ingenieros civiles y topógrafos al intentar compartir y reutilizar datos entre diferentes plataformas de software. Antes de LandXML, el intercambio de datos entre distintos sistemas requería conversiones manuales, que eran propensas a errores y a la pérdida de información. LandXML surgió para estandarizar este proceso, proporcionando un formato de archivo abierto y basado en XML que podría ser utilizado por cualquier software compatible, facilitando así la colaboración y la eficiencia en los proyectos de infraestructura.

Figura 53 – Logotipo del portal web oficial del estándar LANDXML (Fuente: LandXML.org)

Entre los principales objetivos clave de LandXML se puede deducir:

- Estandarización del Intercambio de Datos: Crear un estándar que permita a los profesionales compartir datos de diseño, topografía y construcción de manera uniforme y sin pérdida de información.
- Interoperabilidad: Facilitar la interoperabilidad entre diferentes aplicaciones de software en la ingeniería civil, eliminando la necesidad de procesos de conversión complejos y propensos a errores.
- Eficiencia en el Proceso de Diseño y Construcción: Mejorar la eficiencia en los procesos de diseño y construcción al permitir un flujo de datos más fluido y coherente entre las distintas fases del proyecto.

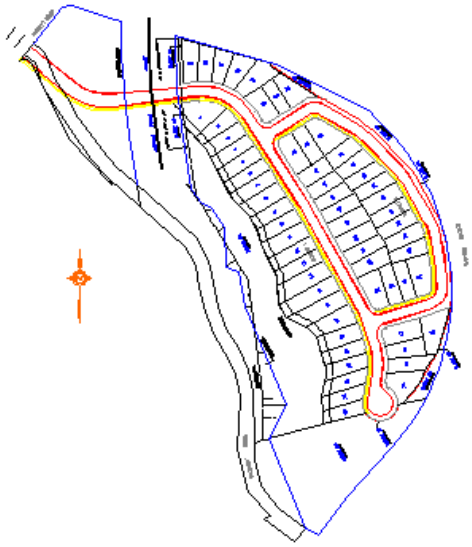
Además, una vez se conocen los objetivos es de especial importancia tener en cuenta que es lo que caracteriza principalmente el estándar LandXML:

1. Basado en XML: LandXML utiliza el lenguaje XML (eXtensible Markup Language) para representar datos topográficos y de ingeniería. XML es un formato de texto que es tanto legible por humanos como procesable por máquinas, lo que facilita su adopción y uso en diferentes sistemas.
2. Amplio Alcance:
    - Superficies: LandXML puede representar modelos digitales de terreno (DTM) incluyendo puntos, líneas de ruptura, triángulos, y otras entidades necesarias para definir la topografía de un área.
    - Alineaciones y Carreteras: Incluye detalles sobre alineaciones horizontales y verticales, así como secciones transversales, que son cruciales para el diseño de carreteras y otras infraestructuras lineales.
    - Parcelas: LandXML permite la definición y descripción de parcelas de terreno, incluyendo sus límites y propiedades asociadas.
    - Redes de Servicios Públicos: Puede manejar datos relacionados con la infraestructura subterránea, como tuberías de agua, redes eléctricas, y sistemas de alcantarillado.
3. Interoperabilidad: LandXML está diseñado para ser utilizado por diferentes tipos de software de diseño asistido por computadora (CAD), sistemas de información geográfica (GIS), y herramientas de

ingeniería civil, lo que facilita la colaboración entre diferentes disciplinas y sectores de la industria.

4. Compatibilidad con Modelos 2D y 3D: Aunque LandXML es más conocido por su capacidad de manejar datos de diseño en 2D, también puede representar modelos 3D, lo que lo hace adecuado para proyectos de infraestructura que requieren detalles tridimensionales.

Figura 54 – Importación de información de un LandXML (Fuente: Autodesk)



LandXML COMO ESTÁNDAR OPENBIM

Asimismo, dentro del enfoque colaborativo para el diseño de infraestructuras openBIM, basado en estándares abiertos que permiten la interoperabilidad y la libre elección de herramientas. En este contexto, LandXML desempeña un papel importante como un estándar que se alinea con los principios de openBIM, especialmente en el ámbito de las infraestructuras.

Relación entre LandXML y openBIM:

- **Interoperabilidad y Colaboración**: LandXML se alinea con el enfoque openBIM al facilitar la interoperabilidad entre diferentes sistemas y disciplinas. Aunque openBIM abarca una gama más amplia de estándares, incluyendo IFC (Industry Foundation Classes) para modelado de información de construcción, LandXML ofrece una solución específica y eficiente para el intercambio de datos geoespaciales y de diseño civil.
- **Compatibilidad con BIM**: Aunque LandXML no es un estándar BIM en sí mismo, puede integrarse en flujos de trabajo BIM como una herramienta complementaria para el manejo de datos específicos de infraestructuras. Por ejemplo, los datos topográficos y de alineaciones generados en LandXML pueden ser importados en modelos BIM para su uso en la planificación y diseño de proyectos más amplios [5].
- **Flujo de Datos Continuo**: Dentro de un entorno openBIM, LandXML puede ser utilizado en las primeras etapas del proyecto, donde la precisión topográfica y la definición de alineaciones son cruciales. Estos datos pueden luego integrarse con modelos IFC en las etapas posteriores del diseño, construcción y operación, asegurando un flujo de datos continuo y coherente.

En adición, es de especial importancia remarcar que LandXML es compuesto por unos Usos Específicos en función de qué Proyecto de Infraestructura se trate, entre los que se contemplan:

Diseño de Carreteras y Ferrocarriles: LandXML es ampliamente utilizado en el diseño de infraestructuras lineales como carreteras y ferrocarriles. Permite la definición precisa de alineaciones horizontales y verticales, así como la creación de secciones transversales, que son esenciales para la planificación y construcción de estas infraestructuras.

Modelos de Terreno y Análisis Topográfico: Los modelos digitales de terreno (DTM) creados y compartidos

en formato LandXML son fundamentales para proyectos de construcción en los que la topografía del terreno juega un papel crucial. Estos datos permiten a los ingenieros realizar análisis precisos del terreno y planificar las obras con mayor eficiencia.

Planificación de Parcelas y Desarrollo Urbano: En el contexto del desarrollo urbano, LandXML se utiliza para definir y gestionar parcelas de terreno, facilitando la planificación y el diseño urbano. Esto es particularmente útil en proyectos de desarrollo de grandes extensiones de terreno donde la organización de parcelas y la infraestructura asociada son críticas.

LandXML vs. IFC

Con la actualización IFC 4.3, tal y como se ha presentado en este capítulo, se ha ampliado el estándar considerablemente para cubrir de manera más integral infraestructuras como carreteras, ferrocarriles y puentes, abarcando todo el ciclo de vida de estos proyectos. Es por tanto procedente establecer una comparativa con LandXML de tal forma que se clarifique las acotaciones de uso de cada uno y las principales características que los diferencian, a pesar de compartir múltiples objetivos.

Tabla 13 - Diferencias entre LandXML e IFC4.3 (Fuente: Elaboración Propia)

	LandXML	**IFC 4.3**
Enfoque y Alcance	Diseñado principalmente para el intercambio de datos en las fases de diseño y topografía de proyectos de infraestructura. Proporciona una solución eficiente y específica para estas tareas.	Tiene un enfoque más amplio, cubriendo todas las etapas del ciclo de vida de un proyecto de infraestructura en un entorno BIM, desde el diseño y la construcción hasta la operación y el mantenimiento.
Estructura de Datos	Diseñado principalmente para el intercambio de datos en las fases de diseño y topografía de proyectos de infraestructura. Proporciona una solución eficiente y específica para estas tareas.	Tiene un enfoque más amplio, cubriendo todas las etapas del ciclo de vida de un proyecto de infraestructura en un entorno BIM, desde el diseño y la construcción hasta la operación y el mantenimiento.
Integración en Flujos de Trabajo BIM	Diseñado principalmente para el intercambio de datos en las fases de diseño y topografía de proyectos de infraestructura. Proporciona una solución eficiente y específica para estas tareas.	Tiene un enfoque más amplio, cubriendo todas las etapas del ciclo de vida de un proyecto de infraestructura en un entorno BIM, desde el diseño y la construcción hasta la operación y el mantenimiento.

Compatibilidad y Flujo de Trabajo Combinado

En muchos proyectos de infraestructura, es común que LandXML e IFC se utilicen de manera complementaria. Por ejemplo, los datos de levantamiento topográfico y alineaciones viales pueden ser capturados y procesados inicialmente en LandXML. Luego, estos datos pueden ser importados en un modelo BIM basado en IFC para su integración con otros elementos de infraestructura, como edificios, puentes y sistemas de servicios públicos.

Beneficios del Uso Combinado:

- **Precisión y Detalle en las Etapas Iniciales**: LandXML proporciona un alto nivel de detalle y precisión en las fases de diseño y planificación, lo que es esencial para la correcta ejecución de las fases posteriores en un proyecto de infraestructura.
- **Integración Multidisciplinaria**: IFC 4.3 permite la integración de estos datos detallados en un modelo BIM más amplio, facilitando la colaboración entre diferentes disciplinas y asegurando que todos los aspectos del proyecto estén coordinados.
- **Flujo de Datos Coherente**: La combinación de LandXML e IFC en un flujo de trabajo coherente asegura que los datos recopilados y procesados en las etapas iniciales se mantengan consistentes y

accesibles a lo largo de todo el ciclo de vida del proyecto.

En definitiva, **LandXML** presenta ser un estándar poderoso y específico para el intercambio de datos en proyectos de infraestructura, especialmente útil en las etapas de diseño y topografía. En el contexto del openBIM, LandXML desempeña un papel crucial al proporcionar una solución interoperable para el manejo de datos geoespaciales y de alineaciones viales, que luego pueden integrarse en flujos de trabajo BIM más amplios utilizando estándares como **IFC 4.3**. La compatibilidad y el uso combinado de ambos estándares ofrecen una solución robusta y eficiente para la gestión de proyectos de infraestructura, asegurando precisión, colaboración y coherencia en todas las fases del ciclo de vida del proyecto.

4.2.5 Desafíos y Limitaciones

A pesar de las prometedoras actualizaciones que presenta el estándar IFC 4.3, su implementación en el sector de infraestructuras civiles enfrenta varios desafíos significativos. Uno de los obstáculos más notorios es la curva de aprendizaje asociada con esta nueva versión. Las entidades de diseño e ingeniería requieren una capacitación extensa para aprovechar plenamente las capacidades mejoradas de IFC 4.3, lo cual puede traducirse en una inversión inicial de tiempo y recursos.

Además, la compatibilidad con versiones anteriores sigue siendo un punto crítico. Muchos proyectos en infraestructura aún utilizan versiones anteriores de IFC, y la transición a IFC 4.3 puede implicar desafíos en la integración de datos históricos, lo que potencialmente podría llevar a problemas de interoperabilidad entre diferentes versiones del estándar.

Otro factor limitante es la necesidad de mayor desarrollo en áreas específicas del estándar. Si bien IFC 4.3 ha ampliado su alcance para incluir mejor definición en tipos de infraestructura como carreteras y puentes, aún existen segmentos como instalaciones subterráneas, presas y otros tipos de infraestructuras civiles que requieren una mayor especificación y detalle en el estándar para ser plenamente funcionales en un entorno BIM.

RESISTENCIA A IMPLANTACIÓN EN EL MERCADO

La implementación del estándar IFC 4.3 en el mercado ha enfrentado una cierta resistencia, particularmente por parte de las casas de software. Esto se debe principalmente a que, al no ser los autores directos de esta tecnología, la adaptación a estándares como IFC 4.3 implica para ellos un costo significativo en términos de recursos y tiempo. Las empresas de software, por naturaleza, tienden a ser reacias al cambio cuando este no está impulsado desde dentro. Sin embargo, esta resistencia puede ser superada rápidamente cuando los clientes clave, como son las administraciones públicas, comienzan a exigir el uso de estándares OpenBIM en proyectos, tal como se establece en el Plan BIM España que se ha tratado. En este escenario, las casas de software no tienen otra opción más que adaptarse a los requisitos del cliente. Como resultado, una vez que se formalicen estas exigencias desde el sector público, la adopción del estándar será inmediata.

Además, este cambio en la dinámica tiene implicaciones para los actores del mercado que tradicionalmente han dominado bajo estructuras cercanas oligopolios en el desarrollo de software BIM. Estas grandes empresas, que anteriormente controlaban el acceso a las tecnologías mediante formatos cerrados, se verán forzadas a abrirse y adoptar estándares abiertos como IFC para mantener su relevancia en un mercado donde la demanda de interoperabilidad será cada vez mayor. Con la presión de un cliente como la administración pública que exige la implementación de estándares OpenBIM que democraticen la tecnología de los modelos BIM, las barreras impuestas por estas estructuras sufrirán una dilución, acelerando así la adopción del IFC 4.3 en todo el sector.

En consecuencia, se percibe un cambio hacia una mayor **democratización del acceso a la tecnología**, ya que los estándares abiertos permiten a los usuarios elegir herramientas más diversas sin sacrificar la interoperabilidad. Esto reduce la dependencia de los grandes proveedores y promueve la aparición de competidores que ofrecen soluciones más accesibles, basadas en estos estándares.

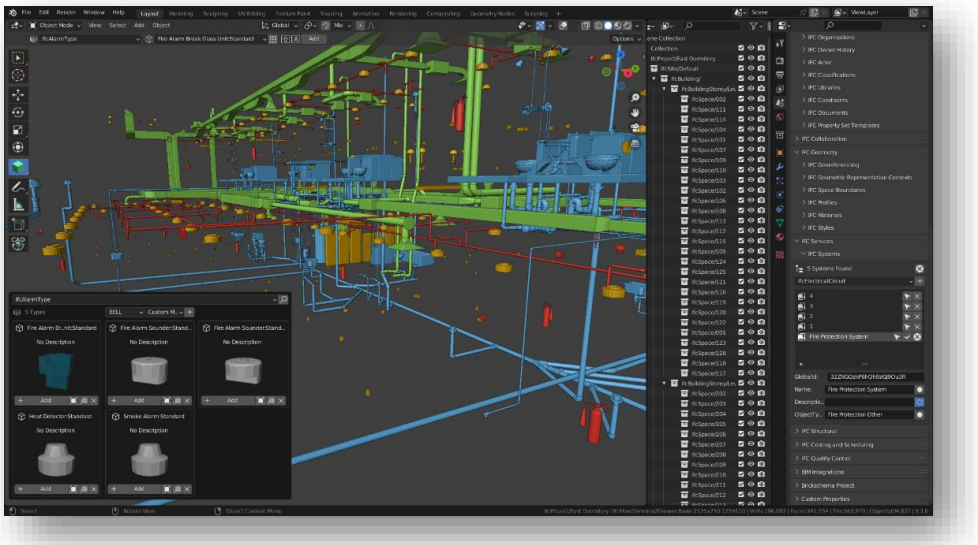
En segundo lugar, **la competitividad en el mercado aumentará**. Las grandes empresas de software se verán obligadas a mejorar sus capacidades de exportación y compatibilidad con estándares abiertos para no perder relevancia. Al no poder retener a los clientes exclusivamente dentro de su ecosistema cerrado, estas empresas tendrán que centrar sus esfuerzos en mejorar la calidad, innovación y servicios de soporte de sus productos, lo que, a su vez, beneficiará a los usuarios finales.

Por último, se espera que **la innovación en el desarrollo de nuevas herramientas y flujos de trabajo** se acelere, ya que la adopción de estándares abiertos reduce las barreras de entrada al mercado y facilita la colaboración entre diferentes actores. Los profesionales y desarrolladores tendrán más libertad para adaptar y personalizar soluciones de construcción digital a sus necesidades, lo que podría desencadenar un ciclo de innovación tecnológica en el sector AEC (Arquitectura, Ingeniería y Construcción).

SOFTWARE OPENSOURCE Y MODELADO NATIVO EN IFC

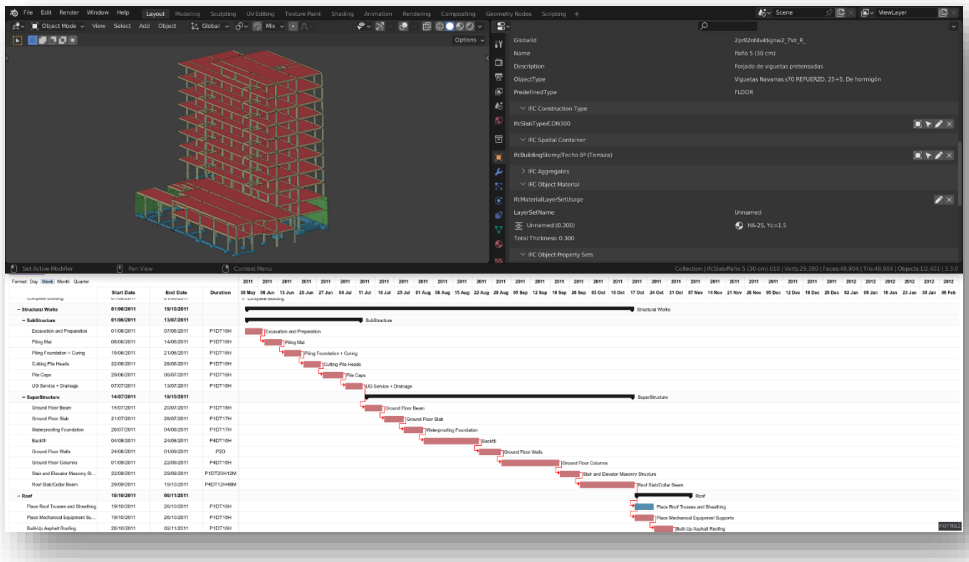
En este contexto, el desarrollo de software de modelado de código abierto *"Open Source"*, como BlenderBIM [30], que trabaja bajo una dinámica de edición nativa del estándar IFC, surge como una solución prometedora para superar algunas de las limitaciones y monopolios que dominan el mercado de software propietario. BlenderBIM es un ejemplo destacado de esta iniciativa, ya que busca ofrecer una plataforma de modelado totalmente compatible con IFC, sin depender de las soluciones comerciales tradicionales. Esto representa una oportunidad significativa para aquellos en el sector de infraestructuras que desean adoptar prácticas OpenBIM sin las restricciones impuestas por las licencias y los costos de software comercial.

Figura 55 – Software de Libre Uso (OpenSource) de modelado nativo IFC (Fuente: BlenderBIM)



El desarrollo continuo y la mejora de herramientas como BlenderBIM no solo puede democratizar el acceso a tecnologías BIM basadas en IFC, sino que también puede facilitar una adopción más amplia y efectiva del estándar IFC4.3. Sin embargo, como muchas herramientas de código abierto, su evolución y capacidad para transformarse en una opción viable en el mercado profesional depende en gran medida del apoyo y contribución de una comunidad activa que contribuya al refinamiento y mejora de la plataforma. De este modo, BlenderBIM puede satisfacer las necesidades complejas del modelado de infraestructuras.

Figura 56 - Software de Libre Uso (OpenSource) de modelado nativo IFC (Fuente: BlenderBIM)



Una prueba concreta de la creciente importancia de este tipo de software es el reciente "Tutorial BlenderBIM"[31], publicado por la administración pública del País Vasco, a través de **Euskal Trenbide Sarea (ETS)**, Red Ferroviaria Vasca. Este manual de uso resalta la viabilidad de BlenderBIM como herramienta de modelado IFC en proyectos, **demostrando que incluso ciertas administraciones ya han comenzado a explorar y recomendar su uso en flujos de trabajo BIM**. Esto sugiere que, conforme siga evolucionando y recibiendo apoyo, BlenderBIM se continuará consolidándose como una alternativa altamente viable para el sector de infraestructuras, adaptándose a las necesidades crecientes que se han presentado.

5 EL IFC CIVIL EN HERRAMIENTAS BIM DE DISEÑO

En este capítulo se abordará el proceso de exportación de modelos de infraestructura civil utilizando el estándar IFC 4.3, un formato que representa la primera vez que se define específicamente para infraestructuras civiles, marcando un avance significativo en la interoperabilidad y colaboración dentro de la industria de la construcción. Se explorará en detalle la utilización de Autodesk Civil 3D, una herramienta fundamental dentro del entorno Autodesk y ampliamente reconocida en la ingeniería civil y el diseño de infraestructuras. La elección de Civil 3D se justifica por su disponibilidad a través de las licencias educativas que ofrece la escuela, lo que permite a estudiantes y profesionales en formación acceder a herramientas de vanguardia para el diseño y la documentación de proyectos. Además, se profundizará en las funcionalidades del complemento recientemente desarrollado por Autodesk para IFC 4.3, destacando los pasos necesarios para su configuración y personalización en el contexto de exportación de modelos desde Civil 3D.

En futuras líneas de investigación y expansión se podrá plantear la implementación del estándar IFC aplicado a las infraestructuras en otras plataformas relevantes del mercado, como *Revit* de Autodesk, *OpenROAD* de Bentley, *ISTRAM*, y *Allplan ROAD*, entre otros. Estas herramientas representan alternativas potentes para la gestión y modelado de infraestructuras, y su exploración en relación con IFC 4.3 será clave para asegurar una integración completa y eficiente del estándar en diversos entornos de diseño y construcción.

Además, se tratará de forma introductoria el software de diseño BIM de modelos en IFC nativo, que se ha decidido denominar *herramientas de diseño openBIM, BlenderBIM*. Explorando así su estado actual y potencialidades en lo referente a su contribución al ecosistema de herramientas openBIM.

5.1 La herramienta BIM de diseño, *Civil3D*

Dentro del entorno Autodesk, AutoCAD Civil 3D es una herramienta fundamental en el dominio de la ingeniería civil y el diseño de infraestructuras, brindando un conjunto de funcionalidades diseñadas para potenciar la eficiencia en el diseño, análisis, y documentación de proyectos de ingeniería civil. Encarnando, junto a Revit, un pilar crucial en el flujo de trabajo del Modelado de Información para la Construcción (BIM) de Autodesk, Civil 3D facilita la toma de decisiones informadas mediante un enfoque dinámico e iterativo, lo cual permite a los ingenieros visualizar proyectos dentro de un contexto real, evaluar alternativas de diseño, y simular el desempeño de infraestructuras antes de su construcción física.

Civil3D destaca por su habilidad y herramientas para generar y manipular modelos digitales detallados del terreno sobre los que también se nos permite diseñar proyectos de infraestructura como carreteras, puentes, canales, presas, y sistemas de drenaje, además de administrar grandes cantidades de datos geoespaciales. Esta herramienta integra análisis geotécnicos, hidrológicos, y estructurales, ofreciendo una visión comprensiva del proyecto y su entorno. Su interoperabilidad con otras herramientas de software promueve una colaboración efectiva entre los diversos participantes de un proyecto, incluyendo arquitectos, diseñadores y contratistas, asegurando la coherencia y precisión de los datos a lo largo del ciclo de vida del proyecto.

La capacidad de exportación a IFC4.3 representa una evolución significativa en Civil 3D, mejorando la interoperabilidad con otro software de diseño. Sin embargo, es importante destacar que la herramienta de exportación a IFC4.3 consiste en un complemento externo y no se encuentra integrado directamente en la interfaz. Este complemento se puede descargar desde la plataforma de Autodesk para posteriormente proceder con su instalación en Civil 3D, lo que permitirá a los usuarios exportar sus modelos de diseño a este formato estándar de la industria, facilitando así el intercambio y la colaboración entre diversas plataformas de diseño, además de promover el uso de un estándar abierto de libre uso para la presentación de nuestro proyecto. Este enfoque permite una mayor flexibilidad y actualización continua de las herramientas de exportación del software, asegurando que se mantengan alineadas con los últimos estándares y prácticas en la industria.

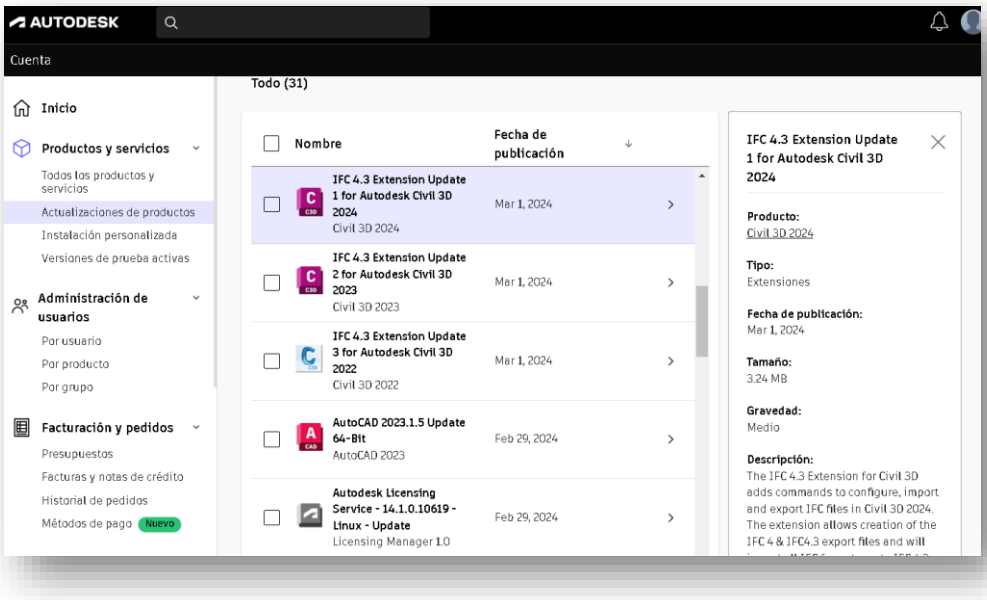
5.2 Configuración Inicial para el complemento IFC 4.3

Descarga e Instalación del Complemento:

Visite la página de centro de descargas de Autodesk [32] donde puede gestionar sus productos y busque el complemento de exportación IFC4.3 para Civil 3D dentro de la pestaña de actualizaciones que se muestra en la siguiente figura.

Descargue e instale el complemento siguiendo las instrucciones proporcionadas por Autodesk. Este proceso puede requerir derechos de administrador en su máquina.

Figura 57 - Portal de Descarga de Productos y Servicios Autodesk (Fuente: Autodesk)

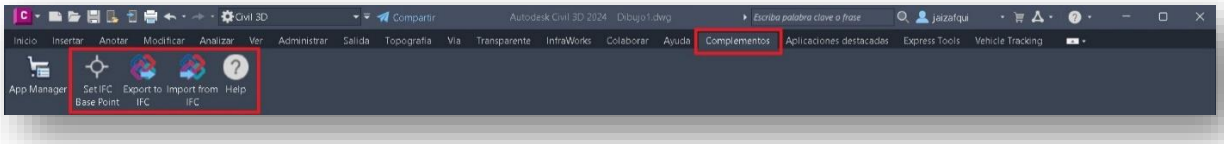


Como se observa en la figura, dispone de distintas opciones para las versiones correspondientes a Civil 3D desde el año 2022 en adelante. Deberá escoger la que se adapte a la versión con la que esté trabajando en su proyecto e instalarla. En este caso nosotros usaremos la última versión de 2024.

Activación del Complemento:

Una vez instalado, inicie Civil 3D. Puede que necesite reiniciar Civil 3D si ya estaba abierto durante la instalación. Verifique que el complemento está activo accediendo a la pestaña de complementos o extensiones dentro de la interfaz del software. Debería ver opciones relacionadas con la exportación a IFC disponibles en la interfaz de usuario.

Figura 58 - Complemento de Exportación en la interfaz de Civil 3D (Fuente: Elaboración Propia)



5.3 Exportación IFC4.3 en *Civil3D*

La exportación de modelos desde Autodesk Civil 3D al formato IFC4.3 es un proceso que permite a los usuarios de Civil 3D participar en flujos de trabajo siguiendo la metodología BIM interoperables, compartiendo información de diseño con precisión entre distintas plataformas de software. Para facilitar este proceso, Autodesk

ofrece un complemento de exportación IFC4.3 que se debe instalar sobre Civil 3D. A continuación, se describe todo el proceso de configuración inicial y los pasos que se han de seguir para realizar una exportación efectiva, incluyendo las opciones y configuraciones específicas que deben considerarse, que serán comentadas en detalle para aprovechar todas las funcionalidades que nos ofrece la herramienta.

5.3.1 Proceso de Exportación

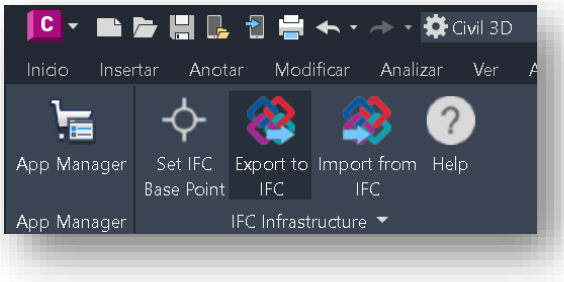
Preparación del Modelo:

Antes de exportar, asegúrese de que su modelo está preparado y organizado correctamente dentro de Civil 3D. Esto incluye la revisión de la estructura de los objetos dentro del modelo, la limpieza de datos innecesarios y la verificación de la integridad del diseño que deseemos exportar.

Exportación Genérica:

Tal y como se explica en el portal de ayuda Autodesk para Civil 3D [33], existen distintas vías de acceso a la herramienta de exportación IFC, en primer lugar, la más intuitiva y general es a través de la pestaña correspondiente a los complementos mostrada en la figura anterior. Aquí, seleccionará "Exportar a IFC" para comenzar el proceso, de tal forma que obtendremos un modelo IFC en su última versión 4x3 de todo nuestro proyecto contenido en Civil 3D. De forma alternativa, el complemento también nos permite ejecutar la exportación a partir de la consola de Civil 3D, los comandos disponibles se muestran en una tabla al final del capítulo.

Figura 59 – Herramienta de Exportación IFC del complemento IFC4.3 de Civil3D (Fuente: Elaboración Propia)



Por otro lado, también podemos realizar una selección específica de elementos a Exportar. Tal que, puede elegir exportar todo el proyecto dentro de Civil 3D o solo partes específicas de este. Esta selección dependerá de las necesidades de su proyecto y de los requisitos de los colaboradores que utilizarán el archivo IFC.

Mapeo de Categorías y Propiedades:

Además, el complemento nos permite configurar de forma exhaustiva los mapas de categorías de objetos y las propiedades de Civil 3D de forma personalizada a sus equivalentes en IFC [33]. Este paso es crucial para asegurar que la información del modelo se traduzca de la forma que el especialista encuentre correctamente oportuna para el flujo de trabajo que desee en el formato IFC4x3. En el siguiente punto abordamos en profundidad el uso de estas herramientas para un control más técnico de nuestra exportación.

Exportación:

Una vez configuradas todas las opciones, proceda con la exportación. El proceso puede tomar algunos minutos, dependiendo del tamaño y la complejidad del modelo.

Después de la exportación, es recomendable revisar el archivo IFC utilizando herramientas de visualización IFC o software de modelado BIM que admita IFC, para asegurarse de que el modelo se ha exportado correctamente y cumple con las expectativas como veremos posteriormente.

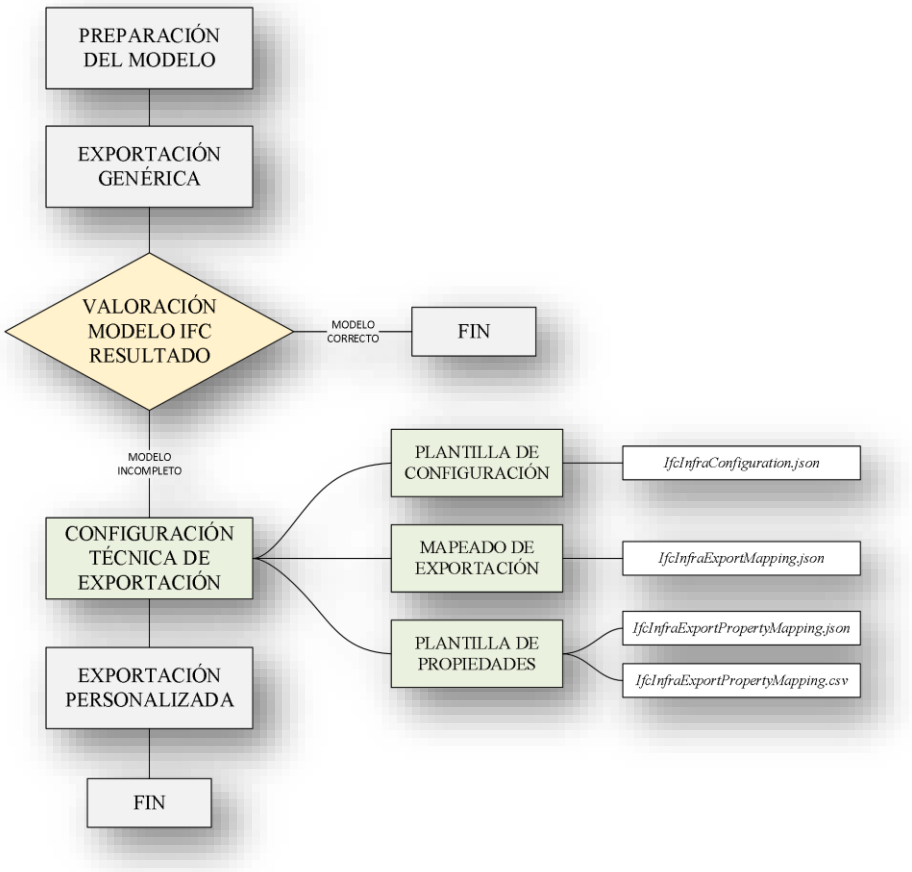
Asimismo, cabe destacar que todas las funciones pertinentes al complemento de exportación se pueden usar llamando su comando correspondiente desde la consola de Civil 3D como mostramos en la siguiente tabla todos los comandos disponibles [33]:

Tabla 14 - Lista de Comandos para el Complemento de Exportación IFC4.3 (Fuente: Elaboración Propia)

Comando	Descripción
IFCInfraExport	Exporta el proyecto actual a un archivo IFC.
IFCInfraExportSelected	Exporta las entidades de un conjunto seleccionado a un archivo IFC.
IFCInfraSavePropertyTemplates	Exporta un archivo de definición de propiedades IFC y un mapa de parámetros Civil 3D asociado.
IFCInfraHelp	Muestra la documentación en línea del complemento y otros recursos.
IFCInfraImport	Importa un archivo IFC en Civil 3D. Más información.
IFCInfraSaveConfig	Crea un archivo de configuración (.json) en la carpeta de dibujos actual.
IFCInfraSaveMappingConfig	Crea un archivo de configuración de mapeado (.json) en la carpeta de dibujos actual.
IFCInfraSetProjectBasePoint	Especifica un punto base para el proyecto.

La exportación de modelos desde Civil 3D a IFC4.3 abre nuevas posibilidades para la colaboración en proyectos de construcción e ingeniería civil. Siguiendo estos pasos y recomendaciones, los usuarios pueden maximizar la interoperabilidad de sus modelos, facilitando una colaboración efectiva y eficiente entre todas las partes involucradas en un proyecto.

Figura 60 - Diagrama de Exportación IFC4.3 en Civil 3D (Fuente: Elaboración Propia)



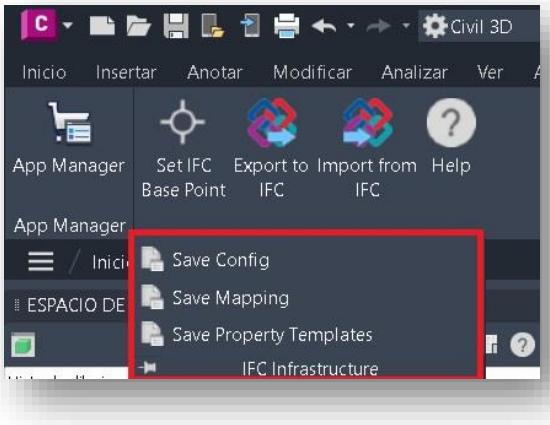
5.3.2 Configuración personalizada de la Exportación

Como se ha visto en la tabla de comandos disponibles para la extensión de exportación en IFC4.3 para Civil 3D, disponemos de varias opciones para configurar nuestra exportación. Actualmente la extensión no dispone de una UI (Interfaz de Uso) que nos permita configurar los aspectos técnicos de la exportación. No obstante, lo que sí nos permite es generar unos archivos *JSON,* un formato de texto sencillo para el intercambio de datos, para que el usuario manipule los datos contenidos en estos y posteriormente se puedan cargar en el proceso de exportación para generar un modelo equivalente IFC personalizado que cumpla las necesidades de nuestro proyecto.

En primer lugar, se han de generar las plantillas de configuración que posteriormente podremos modificar. Podemos hacerlo a través de la interfaz de Civil 3D, dentro de la pestaña de complementos disponemos de las opciones de exportado IFC, es ahí donde podemos abrir un desplegable como el de la figura que nos ofrece las siguientes funcionalidades [33]:

- ***Save Config***, genera un archivo *IfcInfraConfiguration.json* que define elementos que contendrá nuestro archivo IFC y como lo gestiona la exportación.
- ***Save Mapping***, genera un archivo *IfcInfraExportMapping.json* que nos define las reglas de asignación de los objetos que se exportan a entidades IFC.
- ***Save Property Templates***, en este caso nos generará por defecto dos archivos tanto en formato JSON como CSV. Tal que finalmente obtenemos un archivo *IFCInfraExportPropertyMapping.csv* y *IFCInfraExportPropertyMapping.json*.

Figura 61 - Funcionalidades para la Configuración de la Exportación (Fuente: Elaboración Propia)



Además, de forma alternativa, como se ha introducido anteriormente, se pueden ejecutar estas funciones a partir de los comandos de la tabla anterior. En ese caso, no se nos abrirá una ventana emergente para seleccionar el directorio donde se desea guardar el archivo, sino que se guardará por defecto en la ubicación del proyecto, si el dibujo no está guardado, se almacenará en *C:\Program Data\Autodesk\Civil3D*.

5.3.2.1 Plantilla de Configuración

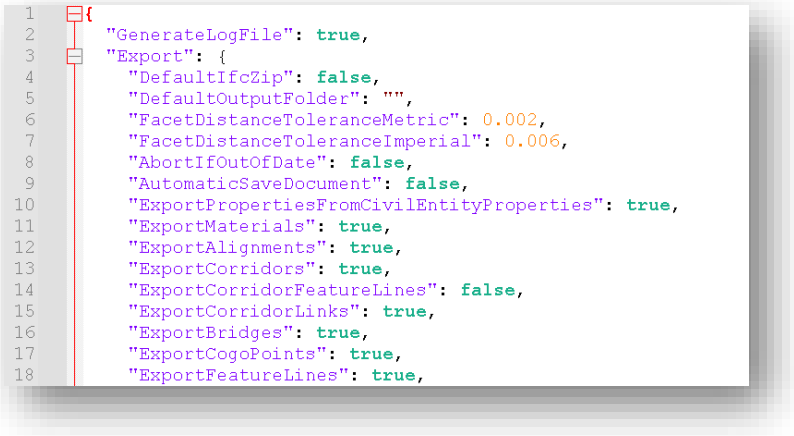
El archivo JSON de configuración *IfcInfraConfiguration.json* para la exportación IFC4.3 en Civil 3D sirve como un archivo de configuración personalizable que define cómo se realizará la exportación de datos desde AutoCAD Civil 3D a un archivo IFC. Este archivo es crucial para ajustar el proceso de exportación a las necesidades específicas de un proyecto, permitiendo a los usuarios especificar detalles sobre cómo se deben exportar diferentes tipos de entidades y datos.

A continuación, se profundiza en la estructura este archivo y en su manipulación para poder llevar a cabo un ajuste de la exportación personalizado a las necesidades específicas del proyecto.

Estructura General del archivo JSON

El archivo se divide en varias secciones principales, cada una relacionada con diferentes aspectos de la exportación de IFC, incluidas las opciones de exportación (Export), importación (Import), y reporte de progreso (Progress Reporting). Dentro de estas secciones, puedes encontrar configuraciones específicas que pueden ser ajustadas.

Figura 62 - Fragmento de IfcInfraConfiguration.json en Notepad++ (Fuente: Elaboración Propia)



Dentro de la sección de Exportación "Export", disponemos de las funcionalidades:

- Opciones Genéricas sobre la exportación: Define si se prioriza la extensión ifczip, tolerancias para la faceta/tesselación de la geometría en métricas e imperiales, y si abortar la exportación si algún alineamiento o corredor está desactualizado.
- Objetos, Propiedades y Materiales: Especifica si se exportan propiedades de entidades civiles, materiales, alineamientos, carreteras (corridors), líneas de características, puntos COGO, entre muchos otros.
- Geometría y Capas: Controla la exportación de sólidos, puntos CAD, polilíneas 3D, mallas polifacéticas, y si solo las capas visibles deben ser consideradas para la exportación.
- Atributos Específicos: Ajusta la precisión de los puntos base, si se exporta el sistema de coordenadas proyectadas sin un código EPSG, y la versión de IFC especificada para la exportación.

Para la sección de Importación "Import" podremos editar:

- Opciones de Importación: Incluye configuraciones como la tolerancia de desviación para la creación de sólidos y si productos huérfanos (elementos no relacionados explícitamente con la estructura espacial o composición de elementos) deben ser reconocidos e incluidos en la importación.

Además, también se podrá definir un reporte de progreso, indicando:

- Definir entre los distintos Modos de Reporte de Progreso: Define cómo se reporta el progreso de la exportación/importación, ya sea en la línea de comandos, un panel específico, o sin reporte.

Cómo Manipular el Archivo

Para ajustar la configuración de exportación a las necesidades de tu proyecto, puedes editar directamente este archivo *JSON*, modificando los valores según lo requieras. Por ejemplo, si necesitas exportar solo alineaciones y carreteras, asegúrate de que las opciones *ExportAlignments* y *ExportCorridors* estén configuradas como el booleano *true*, mientras que las demás entidades que no necesitas exportar pueden estar configuradas como *false*. También es posible ajustar tolerancias, configuraciones específicas de proyecto y los atributos de autor que se asignarán al proyecto según sea necesario.

Una vez hayas realizado los cambios deseados en el archivo *IfcInfraConfiguration.json*, guárdalo en el mismo directorio que tu archivo de dibujo de Civil 3D. Luego, al iniciar la exportación IFC desde AutoCAD Civil 3D,

el software utilizará las configuraciones definidas en este archivo para personalizar la exportación de acuerdo con tus especificaciones.

5.3.2.2 Mapeado de Exportación

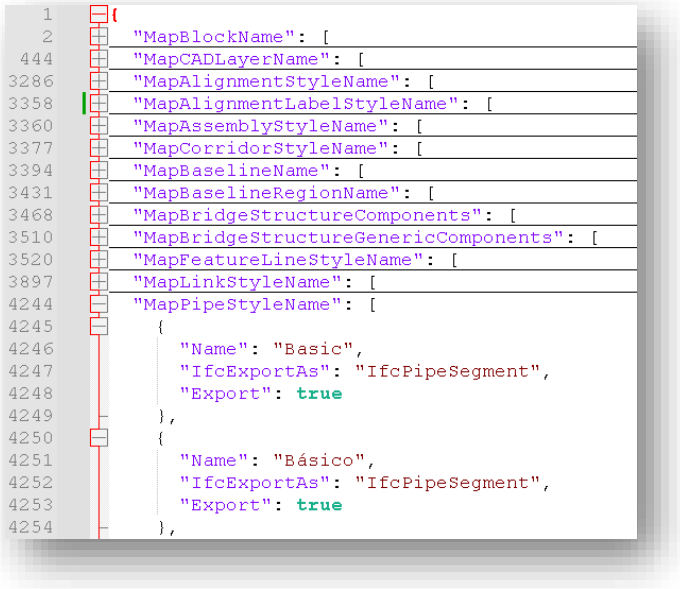
El mapeado de la exportación de proyectos de AutoCAD Civil 3D hacia el formato IFC4.3 se gestiona a través del archivo JSON, *IfcInfraExportMapping.json*, que permite especificar cómo se deben mapear los objetos del modelo Civil 3D a entidades IFC, es decir, contiene las reglas de asignación de objetos de AutoCAD y Civil 3D a clasificación y tipos IFC.

Estructura General del archivo JSON

La estructura general de este archivo es relativamente simple y se compone de secciones de mapeado con múltiples entradas que especifican cómo se deben exportar los diferentes objetos de Civil 3D. Cada entrada en el archivo de mapeo contiene varios campos clave:

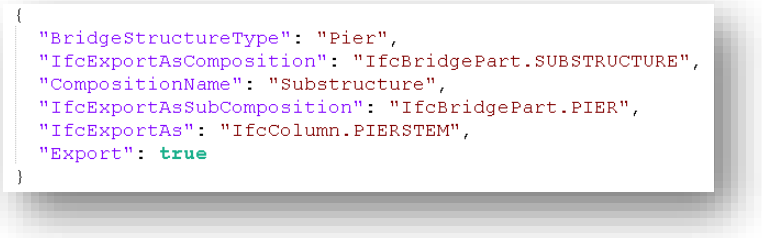
- **Name**: El nombre identificador del objeto en Civil 3D. Puede ser el nombre de un bloque, una capa, o cualquier otro elemento que se quiera mapear a una entidad IFC.
- **IfcExportAs**: Define la clasificación y el tipo IFC al cual el objeto deberá ser exportado. Este campo permite especificar la categoría IFC y, opcionalmente, un subtipo, asegurando que el objeto se interprete correctamente en el contexto del modelo IFC. Tiene el formato de indicar la clase "IfcClassification" o indicar también el tipo predefinido de esta "IfcClassification.IfcType"
- **Export**: Un valor booleano (true/false) que indica si el objeto debe ser incluido en la exportación IFC. Esto permite excluir ciertos objetos del modelo exportado si es necesario.

Figura 63 - Fragmento de IfcInfraExportMapping.json desde Notepad++ (Fuente: Elaboración Propia)



En el fragmento *JSON* mostrado en la figura se puede apreciar cómo las entradas que especifican las clases de IFC correspondientes se encuentran contenidas en distintas secciones de mapeado, para los bloques de AutoCAD (*MapBlockName*), las capas dentro del proyecto (*MapCADLayerName*), etc.

Figura 64 - Entrada de Mapeo para un objeto dentro de una Estructura Espacial (Fuente: Elaboración Propia)



Como se aprecia en la figura anterior, existen algunas entradas en la plantilla de mapeado, que no se indica su funcionamiento en la ayuda de Autodesk. Su funcionamiento es desconocido, aunque, sirviéndonos de la lógica, coincide con la idea de descomposición espacial de las componentes estructurales dentro de una instalación como viene a ser en este caso la de un puente, en concreto, las pilas.

La configuración del mapeo de entidades IFC equivalentes a las de nuestro proyecto nos permite una personalización detallada del proceso de exportación, asegurando que los modelos IFC generados cumplen con los requisitos específicos.

Cómo Manipular el Archivo

La manipulación requiere conocimientos básicos de edición de archivos JSON, los cuales se pueden editar con cualquier editor de texto plano o un editor de código que soporte resaltado de sintaxis JSON, como Notepad++ o Microsoft Visual Studio, para facilitar la visualización de la estructura.

Para modificar el archivo de manera efectiva, se recomienda navegar por las entradas existentes en el archivo y modifique los valores de ***Name/Code***, ***IfcExportAs***, y ***Export*** según sea necesario. Puede añadir nuevas reglas de mapeo copiando y editando las entradas existentes o eliminar reglas eliminando entradas completas del archivo.

Después de realizar las modificaciones deseadas, guarde el archivo. Asegúrese de mantener el formato JSON correcto, ya que errores de sintaxis pueden impedir que el archivo se procese correctamente durante la exportación.

5.3.2.3 Plantilla de Propiedades

Al guardar la plantilla de propiedades obtendremos un archivo JSON denominado *IFCInfraExportPropertyMapping.json* y en adición otro archivo CSV denominado *IFCInfraExportPropertyMapping.csv*, ambos representan la misma información y juegan un papel crucial en la personalización del proceso de exportación desde AutoCAD Civil 3D hacia el formato IFC 4.3. Mientras el archivo JSON consiste en una estructura similar al resto, en este caso también se dispone de un CSV que permitirá analizar la configuración de la plantilla de forma más intuitiva desde una hoja de cálculo de Excel.

Estructura del Archivo

IFCInfraExportPropertyMapping.json está organizado jerárquicamente, comenzando con un arreglo de *PropertySetTemplates*. Cada uno de estos conjuntos representa una categoría de propiedades que se aplicará a las entidades de IFC durante la exportación.

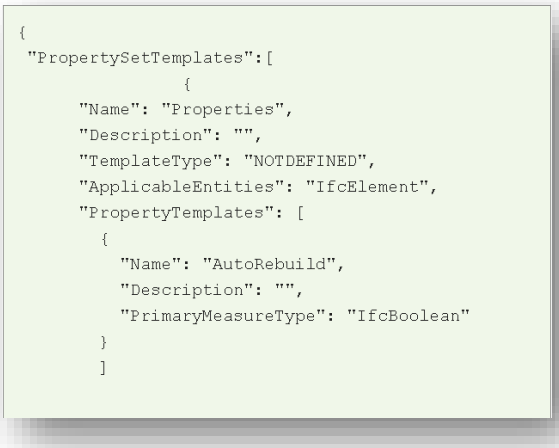
PropertySetTemplates: Un arreglo que contiene múltiples conjuntos de propiedades. Cada conjunto se relaciona con diferentes aspectos de la información del proyecto, como resumen, propiedades generales, visualización 3D, y más.

Dentro de cada *PropertySetTemplate*, encontramos los siguientes campos:

- **Name**: El nombre del conjunto de propiedades, que identifica su propósito o categoría.
- **Description**: Una descripción opcional del conjunto de propiedades.
- **TemplateType**: El tipo de plantilla, que en la mayoría de los casos está definido como "NOTDEFINED".
- **ApplicableEntities**: Especifica a qué entidades IFC aplican estas propiedades.

- **PropertyTemplates**: Un arreglo que contiene las propiedades individuales dentro del conjunto. Cada propiedad tiene:
    - **Name**: El nombre de la propiedad.
    - **Description**: Una descripción de la propiedad.
    - **PrimaryMeasureType**: El tipo de medida o dato para la propiedad, que define el tipo de valor esperado (por ejemplo, IfcLabel, IfcBoolean, IfcReal).

Figura 65 - Ejemplo de sintaxis de asignación para un arreglo de propiedades en formato JSON (Fuente: Autodesk)



```
{
 "PropertySetTemplates":[
                {
      "Name": "Properties",
      "Description": "",
      "TemplateType": "NOTDEFINED",
      "ApplicableEntities": "IfcElement",
      "PropertyTemplates": [
        {
          "Name": "AutoRebuild",
          "Description": "",
          "PrimaryMeasureType": "IfcBoolean"
        }
        ]
```

Además, el archivo *IFCInfraExportPropertyMapping.csv* complementa la configuración realizada en el archivo JSON, proporcionando un método estructurado para el mapeo detallado de propiedades. La estructura del archivo CSV se organiza en seis columnas esenciales:

Figura 66 - Fragmento del mapeo de propiedades CSV en una hoja de cálculo (Fuente: Elaboración Propia)

IfcPropertySet Name	IfcProperty Name	Active	Source	Group	Civil 3D Name
Summary	Title	TRUE	DrawingProperties	Summary	Title
Summary	Subject	TRUE	DrawingProperties	Summary	Subject
Summary	Author	TRUE	DrawingProperties	Summary	Author
Summary	Keywords	TRUE	DrawingProperties	Summary	Keywords
Summary	Comments	TRUE	DrawingProperties	Summary	Comments
Summary	HyperlinkBase	TRUE	DrawingProperties	Summary	HyperlinkBase
Properties	Convergence	TRUE	General	Properties	Convergence
Properties	DescriptionFormat	TRUE	General	Properties	DescriptionFormat
Properties	Easting	TRUE	General	Properties	Easting
Properties	Elevation	TRUE	General	Properties	Elevation
Properties	ElevationOverride	TRUE	General	Properties	ElevationOverride
Properties	FullDescription	TRUE	General	Properties	FullDescription
Properties	FullDescriptionOverride	TRUE	General	Properties	FullDescriptionOverride
Properties	GridEasting	TRUE	General	Properties	GridEasting
Properties	GridNorthing	TRUE	General	Properties	GridNorthing
Properties	IsCheckedOut	TRUE	General	Properties	IsCheckedOut
Properties	IsLabelDragged	TRUE	General	Properties	IsLabelDragged
Properties	IsLabelPinned	TRUE	General	Properties	IsLabelPinned
Properties	IsLabelVisible	TRUE	General	Properties	IsLabelVisible
Properties	IsLocked	TRUE	General	Properties	IsLocked
Properties	IsMovable	TRUE	General	Properties	IsMovable

- **IFCPropertySet Name**: Especifica el nombre del conjunto de propiedades en el archivo IFC, agrupando propiedades bajo un mismo contexto.
- **IFCProperty Name**: Define el nombre específico de la propiedad dentro del conjunto en el archivo IFC.
- **Active**: Determina si la propiedad se exportará o no, con valores de True (verdadero) o False (falso).

- **Source**: Indica el origen del objeto desde el cual se define el parámetro, pudiendo ser Properties, PartData, General, o PropertySet.
- **Group**: Señala el grupo de Civil 3D al cual pertenece el valor de la propiedad.
- **Civil 3D Name**: Especifica el nombre exacto de la propiedad tal como se define en Civil 3D.

Este archivo CSV permite una definición clara y estructurada de cómo se deben mapear y exportar las propiedades específicas de Civil 3D al formato IFC, complementando la información proporcionada en el archivo JSON y ofreciendo una granularidad adicional en el proceso de exportación.

Cómo manipular el Archivo

La manipulación del archivo *IFCInfraExportPropertyMapping.json* permite ajustar la exportación de datos desde Civil 3D a IFC para cumplir con los requisitos específicos del proyecto. A continuación, se detallan las consideraciones para su edición:

Al igual que en los casos anteriores, se ha de utilizar un editor de texto plano o un editor JSON específico para realizar cambios en el archivo. Estos cambios pueden incluir la adición, modificación o eliminación de conjuntos de propiedades y sus propiedades individuales. Para modificar una propiedad existente o agregar una nueva, debes ajustar los campos ***Name***, ***Description***, y ***PrimaryMeasureType*** según sea necesario. Estos cambios permiten controlar qué información se exporta y cómo se presenta en el archivo IFC final.

Será de gran importancia el paso de asegurar una configuración correcta de las plantillas de propiedades a través del archivo ***IfcInfraConfiguration.json*** en Civil 3D, es crucial entender cómo utilizar los parámetros ***PropertyTemplatePaths*** y ***PropertyManagementPaths*** dentro de la configuración. Estos parámetros son esenciales para la integración de plantillas personalizadas de mapeo de propiedades en el proceso de exportación IFC. Para incorporar esta información de manera efectiva, hemos de tener en cuenta la Integración con ***IfcInfraConfiguration.json***, expresando las rutas de las plantillas.

Para utilizar las plantillas personalizadas creadas o editadas para las propiedades (***IFCInfraExportPropertyMapping.json y .csv***), debes especificar sus rutas mediante los parámetros ***PropertyTemplatePaths*** para las plantillas de propiedades y ***PropertyManagementPaths*** para la gestión de estas. La estructura de este archivo de configuración es la siguiente:

Figura 67 - Fragmento de exportación en IfcInfraConfiguration.json (Fuente: Autodesk)

```
{
  "Export": {
    "PropertyTemplatePaths": [
      "./TfNSW_Properties.ifc"
    ],
    "PropertyManagementPaths": [
      "./TfNSW_Properties.csv"
    ]
  }
}
```

- **PropertyTemplatePaths**: Especifica la ruta de la plantilla de propiedades que se usarán durante la exportación IFC. Aquí puedes incluir el archivo **IFCInfraExportPropertyMapping.json** modificado, asegurándote de que la ruta sea correcta en relación con la ubicación del archivo **IfcInfraConfiguration.json**.
- **PropertyManagementPaths**: Define la ruta de los archivos relacionados con la gestión de propiedades, que pueden incluir definiciones adicionales de mapeo o configuraciones específicas del proyecto.

Hemos de asegurar que las rutas especificadas en **PropertyTemplatePaths** y **PropertyManagementPaths**

reflejen correctamente la ubicación de tus archivos personalizados. Puedes usar rutas relativas (como en el ejemplo) o rutas absolutas, según tu estructura de proyecto. Mientras que **PropertyTemplatePaths** apuntaría a archivos como **IFCInfraExportPropertyMapping.json**, **PropertyManagementPaths** podría referirse a archivos CSV que contienen definiciones de gestión de propiedades específicas del proyecto. La especificación correcta de estos parámetros garantiza que durante la exportación IFC, Civil 3D utilice tu configuración personalizada. Esto permite un control detallado sobre qué propiedades se exportan y cómo se organizan dentro del archivo IFC resultante.

Por último, hay que destacar que en ausencia de definiciones completas en **IFCInfraExportPropertyMapping.json**, el proceso de exportación intentará realizar estimaciones inteligentes para llenar los huecos, basándose en la información disponible. Sin embargo, para obtener los mejores resultados, es recomendable proporcionar definiciones de propiedad tan completas como sea posible.

5.4 Importación IFC en Civil3D

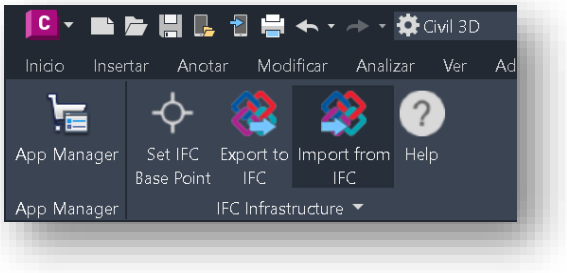
Siguiendo la situación de obligatoriedad de los estándares IFC como parte del Plan BIM España, tanto la capacidad de exportar como importar correctamente proyectos en formato IFC4.3 se convierte en una competencia esencial para los agentes participantes en el sector de la construcción de infraestructuras. Esta sección se dedica a los procedimientos de importación y las configuraciones necesarias para una integración exitosa de estos modelos dentro de la plataforma AutoCAD Civil3D.

Nos centraremos en los aspectos prácticos de la importación, conocido el proceso de instalación y configuración del complemento IFC4.3 (IFC Infra) para Civil3D, que se ha visto al inicio del capítulo. Se pretende transformar los archivos IFC en modelos funcionales dentro de Civil3D, entendiendo las limitaciones y particularidades que ofrece el complemento. A su vez se pretende gestionar los parámetros de importación, para adaptar los modelos abiertos a las necesidades específicas de cada proyecto.

5.4.1 Proceso de Importación

La importación de un archivo IFC en Autodesk Civil 3D [33] comienza con la selección de la función adecuada dentro del software. Para iniciar este proceso, el usuario debe dirigirse a la ficha Complementos, que ha sido mostrada al inicio del capítulo, en la Figura 22. Aquí, dentro del grupo Infraestructura IFC, encontrará la opción Importar desde IFC. Al seleccionar esta opción, se abre un cuadro de diálogo que le permite navegar por los archivos disponibles en su sistema.

Figura 68 - Herramienta de Importación IFC del complemento IFC4.3 de Civil3D (Fuente: Elaboración Propia)



En el cuadro de diálogo Abrir, el usuario debe localizar el archivo IFC que desea importar. Una vez encontrado, debe seleccionarlo y proceder haciendo clic en Abrir. Este acto inicia la importación del modelo IFC al entorno de Civil 3D.

Durante la primera importación de un archivo IFC en un dibujo específico de Civil 3D, el software automáticamente crea parámetros de proyecto IFC. Estos parámetros se llenan con valores que serán utilizados en futuras exportaciones. Los parámetros comunes se almacenan dentro del archivo como propiedades de dibujo personalizadas y se guardan en el archivo ya conocido *IfcInfraConfiguration.json*, facilitando su acceso y reutilización en proyectos futuros.

Un consejo útil para los usuarios es consultar estos parámetros comunes utilizando el comando *"dwgprops"* dentro de Civil 3D. Este comando permite visualizar y modificar las propiedades del dibujo, lo que ayuda a asegurar que la configuración del IFC esté correctamente ajustada para el proyecto actual y cualquier interacción futura con otros archivos IFC o aplicaciones BIM.

5.5 Limitaciones del Complemento

Aunque el complemento de exportación IFC 4.3 para Autodesk Civil 3D representa un avance significativo en la interoperabilidad del flujo de trabajo BIM, es importante reconocer ciertas limitaciones inherentes a su uso que pueden afectar la precisión y eficiencia del proceso de exportación.

Una de las principales limitaciones es la falta de un refinado detallado del modelo IFC directamente desde Civil 3D. El complemento no permite un control exhaustivo sobre aspectos específicos de la exportación, lo que puede resultar en modelos IFC que no capturan completamente la complejidad o las particularidades del diseño original. Debido a esto, es probable que sea necesario recurrir a software externo especializado en la edición y refinamiento de archivos IFC. Estas herramientas externas permiten ajustar y optimizar el modelo exportado, garantizando que se cumplan los requisitos específicos del proyecto y se maximice la utilidad del archivo IFC en entornos colaborativos.

Otro aspecto crítico es la ausencia de una interfaz de usuario (UI) intuitiva y fácil de usar para la configuración del complemento. En su lugar, los usuarios deben manipular manualmente archivos JSON, como el IfcInfraConfiguration.json y otros archivos de mapeo de exportación. Este enfoque puede ser considerado deficiente, especialmente para usuarios que no tienen experiencia en la edición de archivos de configuración o que no están familiarizados con la estructura y sintaxis del formato JSON. La falta de una UI limita la accesibilidad y usabilidad del complemento, haciendo que el proceso de personalización y ajuste de la exportación sea menos intuitivo y más propenso a errores.

La necesidad de editar manualmente estos archivos JSON no solo aumenta la carga de trabajo, sino que también introduce un mayor riesgo de cometer errores que podrían comprometer la integridad del modelo exportado. Para usuarios que no están familiarizados con la edición de estos archivos, la curva de aprendizaje puede ser empinada, lo que podría retrasar el progreso de los proyectos o llevar a la creación de archivos IFC que no cumplen con los estándares necesarios.

Todo ello se podrá contrastar con el caso de estudio que se presenta en el capítulo 6, en el cual se examina una situación real de exportación utilizando este complemento. En dicho capítulo, se han explorado las dificultades que presenta la dinámica de trabajo que plantea la extensión, proporcionando una visión práctica de las limitaciones mencionadas y cómo estas impactan en un escenario de proyecto real. Este análisis permitirá evaluar con mayor detalle las implicaciones de las limitaciones del complemento y la necesidad potencial de herramientas adicionales para lograr resultados óptimos en la exportación a IFC 4.3.

5.6 Herramientas openBIM de diseño

En el contexto de la adopción de estándares OpenBIM, el uso de herramientas de diseño que soporten estos estándares se ha vuelto fundamental para garantizar la interoperabilidad y la eficiencia en los proyectos de infraestructura. Una de las herramientas más destacadas en este ámbito es BlenderBIM [30], un complemento de código abierto para el software de modelado 3D Blender, que permite la creación, edición y gestión de modelos BIM en formato IFC.

BlenderBIM se posiciona como una solución robusta y flexible dentro del ecosistema OpenBIM, ofreciendo a los profesionales del diseño una herramienta que no solo facilita la conformidad con el estándar IFC, sino que también potencia la capacidad de los usuarios para realizar modificaciones detalladas y precisas en sus modelos. Esta herramienta es especialmente relevante para aquellos que buscan un mayor control sobre el proceso de modelado y una integración más profunda de los estándares abiertos en sus proyectos.

Figura 69 – Uso de BlenderBIM para proyectos de edificación (Fuente: blender3darchitect.com)

El enfoque de BlenderBIM en la interoperabilidad y su naturaleza de código abierto lo convierten en una opción atractiva para proyectos que requieren una alta flexibilidad y la posibilidad de personalización mediante scripts y automatizaciones. A lo largo de este trabajo, se discutirá el potencial de BlenderBIM en el contexto de casos de estudio específicos y metodologías de código, donde se explorará cómo esta herramienta puede ser utilizada para mejorar la eficiencia y precisión en el modelado de infraestructuras civiles.

Sin embargo, en este apartado, se introduce BlenderBIM como un ejemplo representativo de las herramientas OpenBIM de diseño que están emergiendo en el mercado y que están comenzando a cambiar la forma en que los profesionales abordan el diseño y la gestión de modelos BIM. La creciente adopción de herramientas como BlenderBIM refleja un movimiento hacia la apertura y la transparencia en la industria AEC (Arquitectura, Ingeniería y Construcción), permitiendo a los equipos de proyecto colaborar de manera más efectiva y mantener la integridad de los datos a lo largo de todo el ciclo de vida del proyecto.

Para conocer en detalle la aplicación del software OpenBIM BlenderBIM ver epígrafe 6.2.5 y el capítulo 7

6 CASO DE ESTUDIO

El presente capítulo tiene como objetivo la aplicación práctica de los temas tratados en la investigación central de este trabajo. En consecuencia, se estudia desde un enfoque práctico la figura del IFC dentro de los flujos de BIM que puedan plantearse. En este caso, se planteará la simulación de dos situaciones que requieren el uso del formato abierto, abordando así los aspectos que determinan si el IFC aporta valor dentro de proyectos profesionales de infraestructuras. Cabe destacar que no se pretende incidir en la fase de diseño y/o modelado de un proyecto de ingeniería sino en la transformación de información, en conjunto a su representación gráfica, de un proyecto terminado a un formato abierto.

La primera casuística aborda el punto de partida de un entorno cerrado, es decir, un modelo desarrollado dentro de la herramienta de diseño BIM, introducida con anterioridad en el presente proyecto, Civil3D del entorno de software Autodesk. El objetivo de este punto de partida es transformar el modelo cerrado en un formato IFC, intercambiable y operable de forma abierta por las distintas partes interesadas del proyecto. Este proceso implicará la recepción, limpieza, reestructuración, ampliación y conversión, asegurando que todos los datos relevantes estén presentes y sean accesibles a través del estándar IFC4.3 resultante.

En segundo lugar, se revisa la situación opuesta: esta vez se planteará como punto de inicio la recepción de un proyecto real en formato abierto IFC con el que se deberá trabajar para adaptar el modelo a los flujos de trabajo internos del receptor, véase una empresa constructora. Esta casuística permitirá analizar cómo gestionar un modelo que ya cumple con el estándar IFC desde su recepción y cómo adaptarlo a las necesidades específicas de la fase constructiva del proyecto.

Con todo ello, no se pretende más que transportar las ideas tratadas en el proyecto a un plano práctico de proyectos de infraestructuras y estudiar si realmente, acorde a la normativa aprobada por el Plan BIM, es viable el trabajo con el formato abierto IFC y si las herramientas con las que estamos acostumbrados a trabajar lo permiten. Esta investigación nos permitirá obtener unas conclusiones sobre la utilidad y uso del formato IFC en el sector de la Ingeniería Civil actualmente, donde se remarca que está contemplado por la legislación que propone el *Plan BIM España*.

6.1 Metodología Propuesta de Trabajo

Se propone una simulación del proceso de recepción y adaptación de los modelos digitales de un proyecto de infraestructuras BIM. Esta situación surge cuando una entidad constructora, adjudicataria principal de la ejecución de un proyecto que ha aplicado la metodología BIM en su fase de diseño, recibe los modelos BIM por parte del equipo de diseño. Como bien se ha citado en la introducción, la simulación pretende recrear la recepción de estos modelos por el equipo de obra, permitiéndonos entender las dinámicas y desafíos involucrados en la aplicación práctica de BIM en proyectos de infraestructura, de especial relevancia en consecuencia de la aprobación del *Plan BIM España*.

Como hipótesis principal, uno de los requisitos establecidos por parte del equipo BIM de obra es la necesidad de convertir y manejar todos los modelos del proyecto en formato BIM abierto, es decir, *OpenBIM*. En este contexto, aparece la implementación de *IFC4.3* dedicado a infraestructuras.

El estudio práctico abarca una simulación que abordará cómo el Técnico BIM, guiado por las indicaciones y directrices del BIM Manager, recibe los modelos nativos iniciales, diseñados durante la fase de redacción de proyectos, y procede a su adaptación para cumplir con las necesidades específicas de la fase constructiva. Esto incluye; la limpieza, reestructuración, ampliación y conversión a *OpenBIM* de los modelos recibidos, obteniendo como resultado el cumplimiento de las exigencias del equipo, asegurando que todos los datos relevantes estén presentes y sean accesibles a través del estándar *IFC4.3.* Por tanto, se realizarán las modificaciones necesarias para optimizar los modelos para su uso en la construcción, incorporando detalles adicionales y asegurando que todos los modelos se alineen con los requisitos técnicos.

MODELOS Y REQUISITOS BIM

Para entender en profundidad los modelos y requisitos BIM, es esencial desglosar los siguientes términos: PIM, EIR, AIM y AIR, según se describen en la norma UNE-EN ISO 19650-1:2019 [34]:

PIM (Project Information Model)

El Modelo de Información del Proyecto (PIM) es un conjunto de información estructurada y no estructurada relacionada con la fase de desarrollo de un proyecto. El PIM apoya el desarrollo del proyecto y contribuye al AIM (Modelo de Información del Activo) para facilitar las actividades de gestión de activos. Puede contener información como la geometría del proyecto, ubicación de equipos, requisitos de funcionamiento en la etapa de diseño, método de construcción, programación, costos y detalles durante la etapa de construcción del proyecto. Además, debe almacenarse con fines de archivo y auditoría a largo plazo.

EIR (Exchange Information Requirements)

Los Requisitos de Intercambio de Información (EIR) especifican los aspectos de gestión, comerciales y técnicos de la producción de información del proyecto. Los aspectos de gestión y comerciales incluyen los estándares de información y los métodos y procedimientos de producción que el equipo de desarrollo implementará. Los aspectos técnicos deben detallar la información necesaria para cumplir con los PIR (Requisitos de Información del Proyecto). Los EIR deben alinearse con eventos desencadenantes que marcan la finalización de hitos del proyecto y deben identificarse en cualquier adjudicación relacionada. Además, pueden subdividirse y transmitirse a lo largo de la cadena de suministro.

AIM (Asset Information Model)

El Modelo de Información del Activo (AIM) soporta los procesos de gestión de activos estratégicos y diarios establecidos por el adjudicador. Este modelo puede proporcionar información al inicio del proceso de desarrollo del proyecto, conteniendo registros de equipos, tarifas de mantenimiento, fechas de instalación y mantenimiento, información sobre derechos de propiedad, y otros datos valiosos para la gestión sistemática del activo. El AIM se actualiza continuamente a lo largo del ciclo de vida del activo, incorporando información relevante del PIM al inicio de un proyecto y viceversa al finalizar.

AIR (Asset Information Requirements)

Los Requisitos de Información del Activo (AIR) establecen los aspectos de gestión, comerciales y técnicos de la producción de información de los activos. Los aspectos de gestión y comerciales deben incluir el estándar de información y los métodos y procedimientos de producción. Los aspectos técnicos deben especificar la información detallada necesaria para responder a los OIR (Requisitos de Información de la Organización) relacionados con los activos. Los AIR se preparan en respuesta a eventos desencadenantes durante la operación del activo y pueden subdividirse y transmitirse a lo largo de la cadena de suministro, enriquecidos con requisitos adicionales si es necesario.

Figura 70 - Ciclo de vida de la gestión de información genérica de proyectos y activos (Fuente: UNE-EN ISO 19650-1)

6.1.1 Rol del Ingeniero de Construcción Digital (Especialista BIM) y Director de Construcción Digital (BIM Manager)

Acorde al flujo de trabajo planteado en el escenario práctico simulado, se define la figura del **Ingeniero de Construcción Digital (Especialista BIM)**, desempeñada por el autor del presente trabajo de fin de grado, quien será responsable de la manipulación técnica, actualización y adaptación de los modelos BIM recibidos y coordinados por el **Director de Construcción Digital (BIM Manager)** del equipo, figura desempeñada por el tutor, el ICCP D. Blas González González. El director de Construcción Digital establecerá las estrategias para la implementación de la metodología BIM dentro del proyecto, asegurando que todos los aspectos del modelo y su documentación cumplan con los estándares establecidos por la normativa ISO 19650. Así pues, ambos roles formarán, en su conjunto, una representación simplificada, a modo de propuesta, del equipo BIM en la fase de construcción del proyecto.

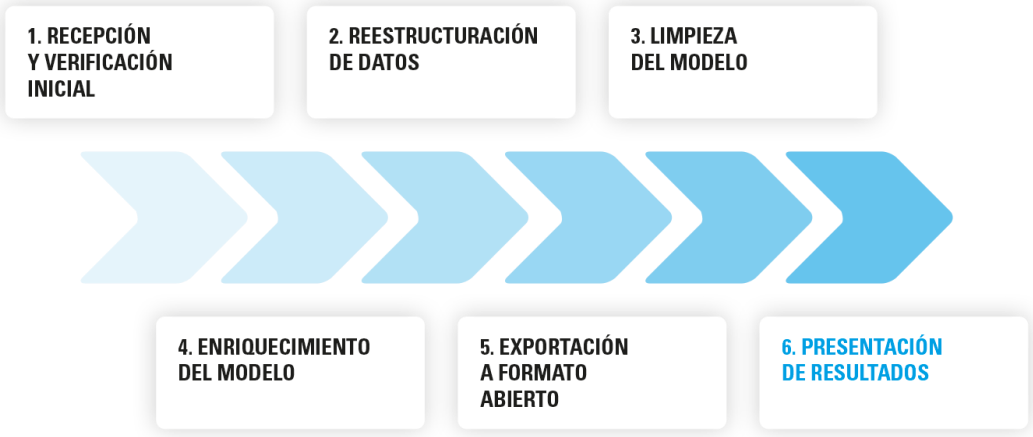
6.1.2 Procesos generales de recepción y adaptación de modelos

Uno de los objetivos principales de simular un equipo de trabajo BIM es demostrar cómo el Ingeniero de Construcción Digital, bajo las directrices de su Director de Construcción Digital, adapta los modelos recibidos para un trabajo mediante formato abierto OpenBIM, obteniendo así los resultados y conclusiones sobre la aplicación práctica del concepto OpenBIM en proyectos de ingeniería y emitir un juicio de valor sobre la viabilidad actual de su presencia en el sector de las infraestructuras.

En el primero de los casos, se plantea que una vez el Director de Construcción Digital ha recibido los modelos nativos iniciales de diseño del proyecto, asignará al Ingeniero de Construcción Digital la tarea de limpieza, reestructuración, ampliación y exportación a un formato abierto, documentando los procesos que se han llevado a cabo para obtener el resultado planteado. Se tratará en detalle los procesos que ha definido el Director de Construcción Digital y el Ingeniero de Construcción Digital ha llevado a cabo.

Continuando con el segundo caso de estudio, se plantea la recepción por el Director de Construcción Digital de los modelos BIM de un proyecto de infraestructuras en formato abierto pertenecientes a una licitación pública de ejecución material, es decir, de construcción de un proyecto de infraestructuras. En este caso, se establece como punto de inicio el formato abierto, generando así precedentes de estudio sobre la viabilidad y el uso de estos formatos abiertos en la redacción de proyectos contemplada por el Plan BIM. El Director de Construcción Digital enviará los modelos pertinentes al Ingeniero de Construcción Digital para que este los analice, de tal forma que los limpie, reestructure, amplíe y explore las posibilidades que nos ofrece este formato a la hora de extraer información valiosa para la construcción. El Técnico BIM, ya sea en el primer o en el segundo caso, seguirá una serie de pasos generales para la recepción y adaptación de los modelos:

Figura 71 – Procesos acotados del flujo de trabajo establecido para el caso de estudio (Fuente: Elaboración Propia)



1. **Recepción y Verificación Inicial:** El Técnico BIM recibe los modelos y verifica su integridad y conformidad con los requisitos iniciales del proyecto.
2. **Reestructuración de Datos:** Ajuste de la información dentro del modelo BIM utilizado (en Civil3D o el propio IFC), asegurando que toda la información y los datos contenidos en el modelo sean accesibles de forma eficiente y estructurada. Propuesta de una nueva estructura del CDE del proyecto, en lo que corresponde a modelos BIM, para una organización óptima de los ficheros.
3. **Limpieza del Modelo:** Depuración, cuando así lo requiera, del modelo para eliminar posibles redundancias o errores, acompañada de una documentación exhaustiva de su contenido dentro de los formatos BIM correspondientes. Este paso asegura que toda la información incluida sea precisa y cumpla con los estándares de la ISO 19650.
4. **Enriquecimiento del Modelo:** Progresivo llenado del modelo con datos e información relevante de las distintas disciplinas del proyecto, asegurando que los modelos contengan todos los datos necesarios, enriqueciendo así la olvidada I (*information*) del acrónimo BIM, para su utilización en la fase constructiva.
5. **Exportación a Formato Abierto:** Exportación, únicamente en el primero de los casos, de toda la información obtenida, tanto gráfica como alfanumérica, a un formato abierto (OpenBIM), permitiendo así su acceso y utilización por todas las partes interesadas del proyecto.
6. **Presentación del AIM**: Análisis de los formatos abiertos obtenidos como resultado de la exportación mediante técnicas de desarrollo propio para conocer el contenido informativo de los modelos que se definen como resultado del proceso.

Estos pasos propuestos pretenden asegurar que los modelos BIM, ya sea en su transición desde un formato cerrado a uno abierto o partiendo desde un formato abierto, sean útiles, precisos y cumplan con los estándares necesarios para su uso efectivo en proyectos de infraestructuras.

6.2 Caso de Estudio – Modelo Nativo, PIM

El primer caso de estudio aborda la recepción inicial de los modelos BIM nativos, que constituyen la base desde la cual el equipo BIM de obra partirá para la fase constructiva del proyecto. Este hito es fundamental, ya que establece el marco de referencia sobre el que se ha de comenzar a planificar todas las adaptaciones y mejoras que se realizarán posteriormente. Aquí, se establece el foco en recibir y revisar estos modelos para asegurar que se ajusten a los requisitos específicos del equipo y que estén listos para su adaptación y uso en las etapas siguientes.

Se incorpora por tanto el Modelo Nativo Ferroviario concebido como parte del Trabajo de Fin de Máster redactado por el ICCP D. David Pérez Viera, enfocado en la *"Modelización BIM de una línea ferroviaria de alta velocidad utilizando Dynamo para la dirección de obra"* [35]. Este modelo representa un tramo significativo de infraestructura ferroviaria diseñado con la intención de optimizar la planificación y ejecución de la construcción, además de facilitar la gestión y mantenimiento futuros de la línea.

Asimismo, cabe recordar que, en acuerdo a los objetivos de los casos de estudio explicados anteriormente, no se pretende realizar un nuevo modelado o diseño del proyecto, sino más bien gestionar, ampliar y exportar toda la información contenida en los modelos existentes. Así, este enfoque permite abordar la problemática común de recibir modelos externos al equipo de trabajo y adaptarlos a los propósitos específicos del proyecto, proponiendo soluciones BIM eficientes para esta situación habitual en el sector.

6.2.1 Línea Ferroviaria de Alta Velocidad

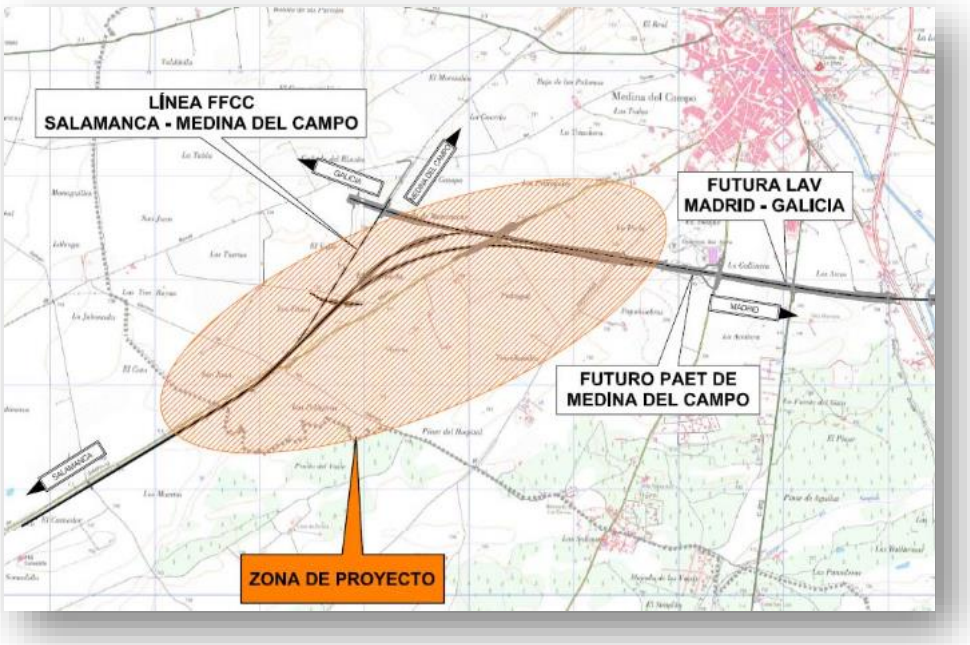
El proyecto aborda la construcción de una plataforma ferroviaria que facilitará la conexión entre la línea Medina del Campo a Salamanca y la Línea de Alta Velocidad Madrid-Galicia, específicamente en el área del Puesto de Adelantamiento y Estacionamiento de Trenes (PAET) en Medina del Campo, Valladolid. Este desarrollo es crucial para mejorar la infraestructura ferroviaria y la eficiencia del servicio en esta región.

Tramos Ejecutados:

1. Plataforma de Alta Velocidad de Doble Vía:
   - Longitud: Aproximadamente 3,54 km.
   - Comienza extendiendo los trazados de proyectos en desarrollo, como la LAV Salamanca-Medina del Campo, y concluye donde se divergen las plataformas hacia las márgenes de la LAV Madrid-Galicia. Este tramo incluye un paso pérgola que cruza por encima de la línea ferroviaria existente Salamanca - Medina del Campo.
2. Ramal Norte de Alta Velocidad de Vía Única:
   - Longitud: Cerca de 3,35 km.
   - Facilita la conexión de trenes desde Madrid hacia Salamanca por el norte. Dentro de este ramal, se incluye la construcción de un viaducto de 110 metros sobre el arroyo de la Golosa y una pérgola de 100 metros para superar la LAV Madrid-Galicia.
3. Ramal Sur de Alta Velocidad de Vía Única:
   - Longitud: Aproximadamente 400 metros.
   - Este ramal ya está construido, y el proyecto actual incluye la creación de un segmento que enlace la Plataforma de Doble Vía con el trazado existente del ramal Sur hacia el PAET de la Línea de Alta Velocidad Madrid-Galicia.
4. Rectificación de Vía de Ancho Ibérico:
   - Longitud: 1,56 km.
   - Se requiere una rectificación de la línea de FFCC Salamanca – Medina del Campo para alinearla con los nuevos ramales de conexión al PAET y la plataforma de Vía Doble. Esta rectificación se extiende desde los PPKK 6+315 al 7+875.

Acorde al autor del proyecto referenciado, está diseñado no solo para mejorar la conectividad entre importantes líneas ferroviarias sino también para asegurar que la infraestructura resultante sea capaz de manejar eficientemente el tráfico ferroviario de alta velocidad y mejorar las operaciones y la seguridad en la región. Además, las construcciones incluidas en el proyecto, como viaductos y pérgolas, son fundamentales para superar las barreras geográficas y otras infraestructuras existentes, garantizando una integración fluida con el paisaje y las estructuras ya presentes.

Figura 72 - Esquema de ejes de proyecto (Fuente: TFM David Pérez Viera)



6.2.2 Recepción y Verificación Inicial

El autor propone una metodología para la creación de modelos BIM, fundamentada en la norma UNE-EN ISO 19650-1 [34], siguiendo sus directrices para la organización de la información sobre obras de construcción, permitiendo así una segmentación y gestión eficaz de los datos en diferentes modelos.

Los modelos nativos pertenecen al entorno Autodesk, específicamente a Civil3D. Esto implica que se trabajará inicialmente con el software AutoCAD Civil3D, como se ha indicado en el capítulo pertinente. Este es el software principal de diseño BIM de infraestructuras civiles disponible en la Escuela Técnica Superior de Ingeniería, por lo tanto, se trabaja la recepción de los modelos nativos en este formato.

Una problemática habitual al recibir modelos ejecutados por terceros es la pérdida de referencias externas a otros documentos y el desorden en la organización del proyecto, al haber sido concebida por otro profesional. Esto se debe a que diferentes usuarios pueden seguir prácticas y estándares variados, lo que puede resultar en inconsistencias y dificultades para gestionar el modelo. Aquí es donde la filosofía de trabajo con **Accesos Directos** en Civil3D, conocidos como *Shortcuts*, cobra especial importancia.

ACCESOS DIRECTOS (CIVIL 3D)

Los *Shortcuts* en Civil3D son herramientas esenciales para mantener la integridad y accesibilidad de las referencias dentro del proyecto. Estos accesos directos permiten crear rutas a los datos almacenados en diferentes archivos de dibujo, facilitando la compartición y actualización de datos entre múltiples usuarios y archivos. Esto asegura que cualquier cambio realizado en un archivo de referencia se refleje automáticamente en todos los dibujos que lo utilizan, manteniendo así la coherencia y precisión de la información a lo largo del proyecto. De esta forma, se plantea un trabajo orientado al disco de almacenamiento principal del equipo, al que comúnmente se le denomina *Disco C* (Ruta: C:\). Siguiendo esa dinámica de trabajo, se consigue trabajar con las mismas rutas dentro de la carpeta del proyecto dispuesta en la ruta del disco C, así las referencias se mantendrán vigentes para cualquier equipo que desee trabajar con los modelos.

No obstante, la correcta estructura de *Shortcuts* implica la creación de una jerarquía organizada de accesos directos que refleje la estructura del proyecto. Esto incluye, la concepción de una carpeta contenedor de datos y la categorización de datos en secciones como topografía, trazado, obras lineales y estructuras, lo que facilita la localización y gestión de la información. La implementación de *Shortcuts* también ayuda a reducir el tamaño de los archivos de dibujo, ya que los datos no se duplican, sino que se referencian desde una ubicación central.

La experiencia de trabajar con *Shortcuts* en Civil3D ofrece múltiples beneficios:

1. **Integridad de Datos:** Mantiene la coherencia y precisión de los datos a lo largo del proyecto, ya que cualquier modificación en un archivo de referencia se actualiza automáticamente en todos los dibujos relacionados.
2. **Eficiencia en la Gestión:** Facilita la gestión del proyecto al permitir un acceso rápido y organizado a la información necesaria, reduciendo el tiempo y esfuerzo requerido para buscar y actualizar datos.
3. **Reducción de Errores:** Minimiza los errores y redundancias al evitar la duplicación de datos y asegurar que todos los usuarios trabajen con la información más actualizada.
4. **Colaboración Mejorada:** Mejora la colaboración entre los miembros del equipo al proporcionar una estructura clara y accesible de los datos contenidos en los distintos archivos del proyecto, permitiendo que distintos usuarios trabajen simultáneamente sin conflictos.

Además de utilizar *Shortcuts*, es crucial **documentar** en profundidad los modelos con los que se trabaja. Esto implica registrar todos los cambios y actualizaciones realizados, así como describir cada uno de los elementos del modelo en el software. La documentación exhaustiva asegura que toda la información

relevante esté disponible y organizada, facilitando la gestión del proyecto y reduciendo el riesgo de errores y futuros malentendidos.

6.2.2.1 Modelos Nativos Recibidos

Los Modelos BIM Desarrollados en el proyecto que se recibe consiste en:

1. **Modelo BIM de la Cartografía**: Este modelo es esencial para proporcionar una base precisa de la superficie topográfica natural sobre la cual se desarrollarán todas las demás fases del proyecto. Es la fundación sobre la cual se planifican y ejecutan las obras, asegurando que todos los cálculos y modificaciones posteriores se basen en datos precisos y actualizados del terreno.
2. **Modelos BIM de Ejes**: Estos modelos incluyen toda la información geométrica necesaria y los parámetros de diseño de las alineaciones tanto en planta como en alzado (rasantes de diseño). Es crucial para la planificación de la ruta que seguirá la infraestructura ferroviaria, permitiendo ajustes precisos en la alineación y garantizando que el trazado cumpla con todas las normativas y requisitos de diseño. Cada uno de los diseños de las alineaciones está acotado en distintos modelos de Civil3d.
3. **Modelos BIM de la Obra Lineal**: Aquí se incluyen los objetos de Obra Lineal para cada trazado además de los elementos constructivos como desmontes, terraplenes, capas de forma, subbalasto, balasto, cunetas y muros. Además, este modelo abarca los raíles y la extrusión del paso pérgola, elementos fundamentales en la construcción de la infraestructura ferroviaria presente. Estos modelos, uno por cada variante de eje análogamente a los modelos BIM de Trazado, son un contenedor complejo que sintetiza múltiples aspectos de la construcción física de la línea ferroviaria.
4. **Modelo BIM de Estructuras Puntuales**: Se concentra en estructuras específicas y críticas, como el Viaducto sobre el Arroyo de la Golosa. Este modelo es conceptual y se debe importar desde InfraWorks a Civil3D, lo que muestra la integración de diferentes herramientas y técnicas BIM para lograr una representación precisa y funcional de estructuras complejas dentro del proyecto general. Asimismo, cabe destacar que la representación de la estructura en cuestión se ve añadida al modelo BIM de la Obra Lineal del trazado al que pertenece.

El BIM Manager del equipo ha recibido estos modelos como entregables iniciales, esta recepción es documentada y verificada para asegurar que todos los modelos cumplan con los estándares necesarios y que estén listos para ser utilizados en las fases subsiguientes. Para ilustrar de manera efectiva el proceso de recepción de los modelos BIM, en la siguiente figura se ilustran los diferentes modelos BIM que son entregados al BIM Manager y cómo cada uno de estos modelos se integra dentro del marco general del proyecto.

Figura 73 – Modelos BIM Nativos del proyecto Ferroviario (Fuente: TFM David Pérez Viera)

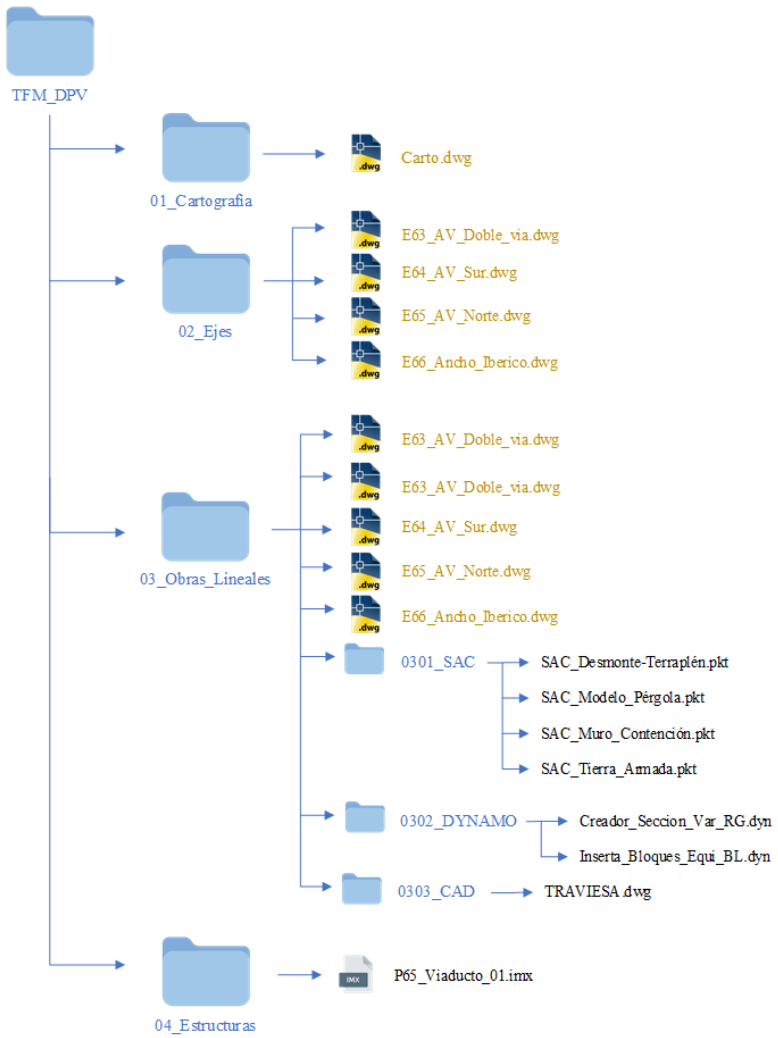
ID	Nombre	Extensión	Programa	Modelo BIM
001	**TFM_DPV_T_Carto**	.dwg	C3D	Superficie de terreno natural subyacente
002	**TFM_DPV_E63_AV_Doble_via**	.dwg	C3D	Trazado en planta y en alzado
003	**TFM_DPV_E64_AV_Sur**	.dwg	C3D	Trazado en planta y en alzado
004	**TFM_DPV_E65_AV_Norte**	.dwg	C3D	Trazado en planta y en alzado
005	**TFM_DPV_E66_Ancho_Iberico**	.dwg	C3D	Trazado en planta y en alzado
006	**TFM_DPV_Exx_Conjunto**	.dwg	C3D	Trazado en planta y en alzado
007	**TFM_DPV_O63_AV_Doble_via**	.dwg	C3D	Superficie de obra lineal y secciones
008	**TFM_DPV_O64_AV_Sur**	.dwg	C3D	Superficie de obra lineal y secciones
009	**TFM_DPV_O65_AV_Norte**	.dwg	C3D	Superficie de obra lineal y secciones
010	**TFM_DPV_O66_Ancho_Iberico**	.dwg	C3D	Superficie de obra lineal y secciones
011	**TFM_DPV_Oxx_Conjunto**	.dwg	C3D	Superficie de obra lineal y secciones
012	**TFM_DPV_Sxx_Solidos**	.dwg	C3D	Sólido de obra lineal
013	**TFM_DPV_SAC_Desmonte-Terraplén**	.pkt	SAC	Subensamblaje de obra lineal
014	**TFM_DPV_SAC_Modelo_Pérgola**	.pkt	SAC	Subensamblaje de obra lineal
015	**TFM_DPV_SAC_Muro_hormigón**	.pkt	SAC	Subensamblaje de obra lineal
016	**TFM_DPV_SAC_Tierra_Armada**	.pkt	SAC	Subensamblaje de obra lineal
017	**TFM_DPV_P65_Viaducto_01**	.imx	IW	Modelo de estructura
018	**TFM_DPV_PAET_MEDINA_DEL_CAMPO**	.imx	IW	Modelo conjunto IW

6.2.3 Reestructuración de Datos y Limpieza del Modelo

El proceso de adaptación y enriquecimiento se centra en la utilización de los Modelos del proyecto Ferroviario en formato Nativo que se ha presentado anteriormente, para someterlos a una reestructuración y documentación en detalle para seguir con especificaciones particulares de nuestro proyecto.

La primera intervención tras la recepción y validación del modelo implica una reestructuración de los datos contenidos dentro del modelo. Así se muestra en la siguiente figura el Entorno Común de Datos, CDE en adelante, propuesto por el autor del proyecto, resultando en la estructura de carpetas y ficheros de la siguiente figura. Se propone por tanto una reestructuración de dicho CDE, de tal forma que se han modificado los modelos centrales contenedores de información y gestionado a partir de una metodología de trabajo basada en accesos directos a los distintos ficheros como datos externos dentro de la interfaz de la herramienta BIM Civil3D. Esto se lleva a cabo mediante la creación de nuevas referencias que son implementadas por el técnico BIM siguiendo las directrices del BIM Manager.

Figura 74 – Estructura de datos del Modelo Nativo recibido (Fuente: Elaboración Propia)



Este paso permite documentar y organizar de forma efectiva los distintos modelos recibidos para asegurar que toda la información y los datos contenidos en el modelo sean accesibles de forma eficiente y estructurada.

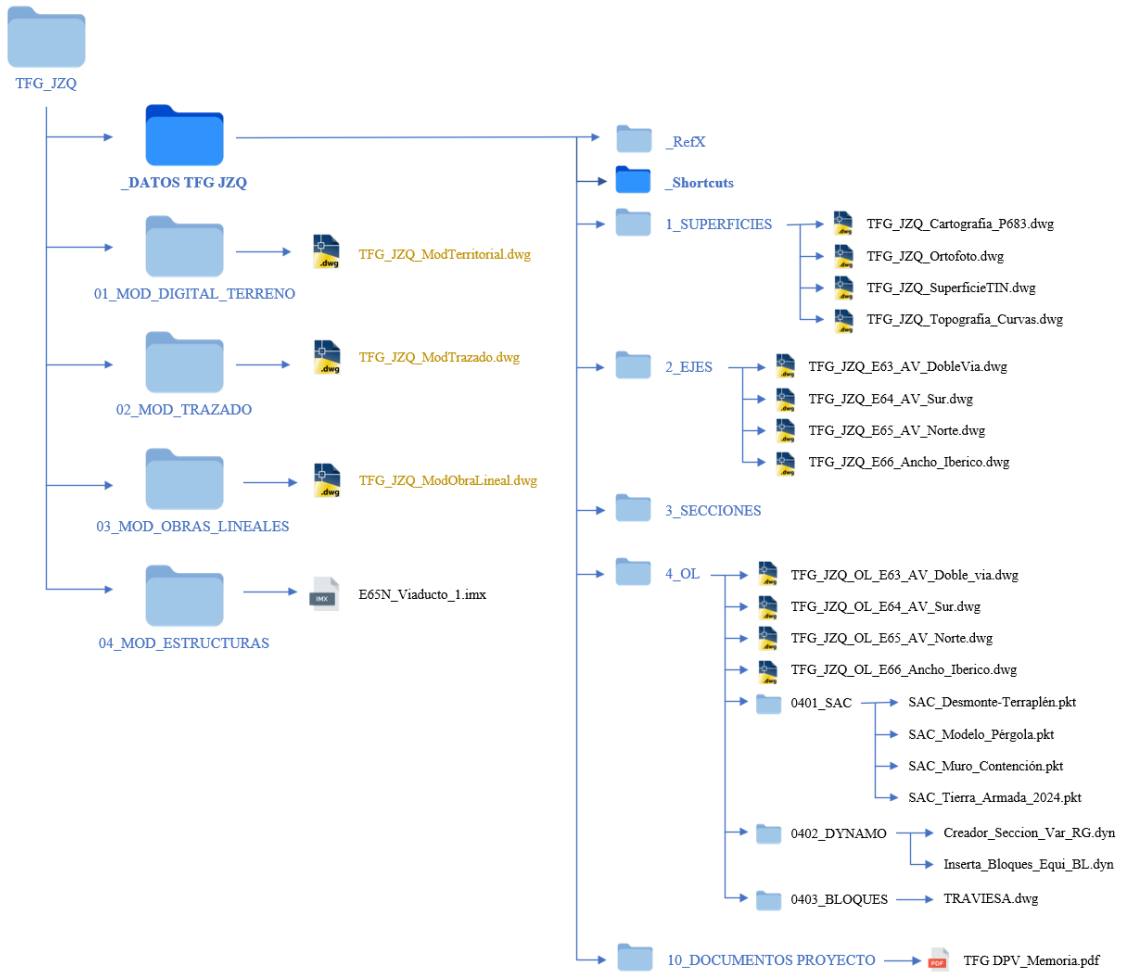
De tal forma que, la propuesta de una nueva estructura del CDE del proyecto para una organización óptima de los ficheros, incluye la creación de carpetas específicas para distintos aspectos del modelo ferroviario.

Se propone trasladar los distintos modelos de cada trazado, además de los formatos auxiliares al diseño de Obras Lineales, a una carpeta de datos central la cual se encargará de custodiar toda la información relevante para los modelos centrales. Entonces los modelos centrales separados por sus disciplinas correspondientes serán:

- La carpeta de Cartografía será renombrada como **Modelo Digital del Terreno**, que contiene todos los datos topográficos y geográficos relevantes.
- La carpeta de Ejes es ahora concebida de nuevo como **Modelo de Trazado**, incluyendo todas las referencias y detalles de los trazados de la línea ferroviaria.
- Las Obras Lineales son almacenadas en la carpeta **Modelo de Obras Lineales**, que se llena con los elementos de las Obras Lineales de cada trazado propuesto, siendo así el modelo central del proyecto. A partir de esos elementos, han sido generadas las distintas superficies que nos ofrecerán los datos detallados y específicos sobre las construcciones lineales implicadas.
- La carpeta de Estructuras se renombra como **Modelo de Estructuras**, destinada a albergar los datos estructurales pertinentes.

Conocida la nueva definición de la estructura en carpetas de los modelos, le sigue una figura de la misma pudiendo así apreciar de forma gráfica las distintas rutas de acceso a la información, destacándose la carpeta de *_DATOS TFG JZQ* como protagonista de la reestructuración del modelo, actuará como contenedora de la información que será accedida en los distintos modelos centrales a partir de sus accesos directos (_Shortcuts) generados en Civil3d como se verá posteriormente.

Figura 75 – Estructura de Datos Propuesta basada en _Shortcuts (Fuente: Elaboración Propia)



Una vez definida la nueva estructura de ficheros y modelos BIM, la siguiente fase del proceso involucra una depuración meticulosa del modelo para eliminar cualquier redundancia o errores, **acompañada de una documentación exhaustiva de su contenido**. Este paso no solo ayuda a clarificar la estructura y el contenido del modelo, sino que también asegura que toda la información incluida sea precisa y cumpla con los estándares de la ISO 19650.

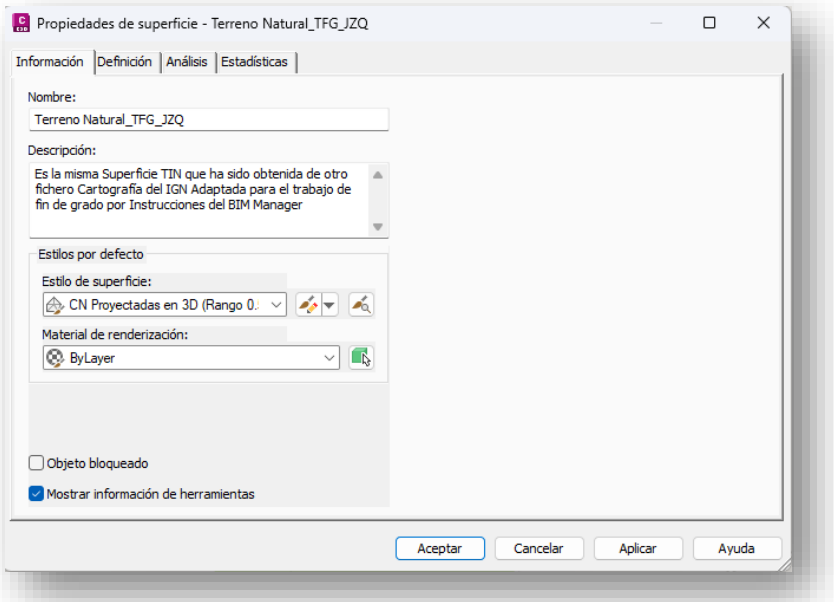
MODELO DIGITAL DEL TERRENO

Se comienza por el primer modelo del que el resto accede a su información, el modelo digital del terreno del proyecto. Como se ha comentado a lo largo de este capítulo la reestructuración consiste en una verificación de la información que contiene la modelo seguida de una reconcepción de los elementos de este para el presente trabajo de fin de grado. Así pues, será añadida la nomenclatura de *"_TFG_JZQ"* en cada uno de los elementos documentados.

Figura 76 – Modelo Digital del Terreno reestructurado (Fuente: Elaboración Propia)

En la figura se puede apreciar el Modelo Digital del Terreno que verificamos era correcto, ya disponía de una triangulación corregida y meticulosamente depurada de los datos obtenidos del terreno existente del Instituto Geográfico Nacional, IGN. Por tanto, se procede a su documentación en detalle y creamos un acceso directo a nuestra base de datos del elemento de la superficie correspondiente.

Figura 77 – Propiedades de la Superficie (Fuente: Elaboración Propia)

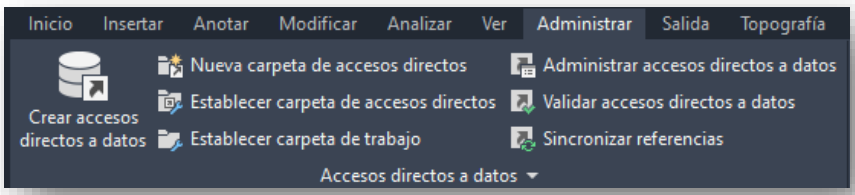


En la figura se puede apreciar el estilo de documentación del modelo que se sigue. Se adopta la definición de

misma superficie TIN (red irregular de triángulos) obtenida a partir del archivo de cartografía del IGN.

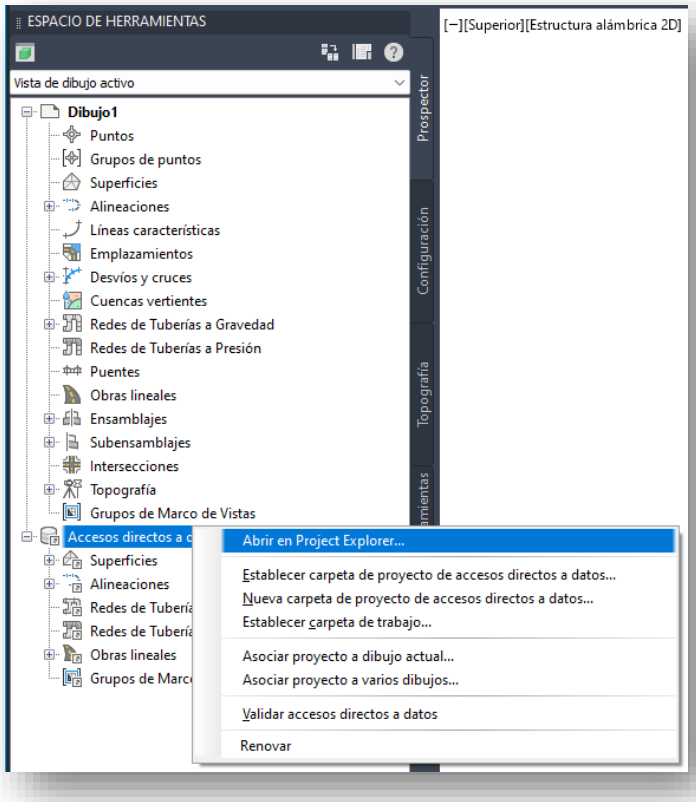
Una vez se ha terminado el trabajo con el elemento se establece el Acceso Directo a la Base de datos del proyecto. Pero antes de definir la superficie como nuestro primer Acceso Directo, ha de ser configurada la carpeta de Datos del proyecto. Los controles de Accesos Directos a Datos se encuentran en la pestaña de Administrar, será el primer panel de esta pestaña en el se pueda encontrar todas las funcionalidades pertinentes. Asimismo, de forma análoga se puede acceder a estos controles a partir del prospector de Civil3d, si accionamos un clic derecho sobre la base de datos del dibujo actual, nos permitirá realizar una serie de operaciones.

Figura 78 – Panel de Accesos Directos a Datos de la Paleta de Herramientas de Civil3D (Fuente: Elaboración Propia)



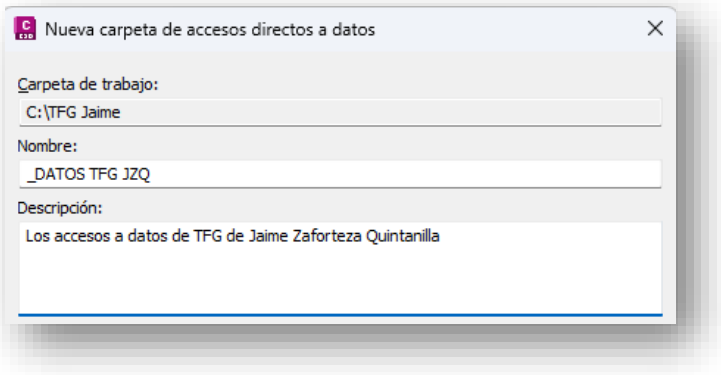
Es dentro de las opciones de Accesos Directos a Datos, que en primer lugar deberemos de crear nuestra carpeta de accesos directos del proyecto que comenzamos a definir.

Figura 79 – Controles de Accesos Directos a Datos a partir del Prospector del dibujo (Fuente: Elaboración Propia)



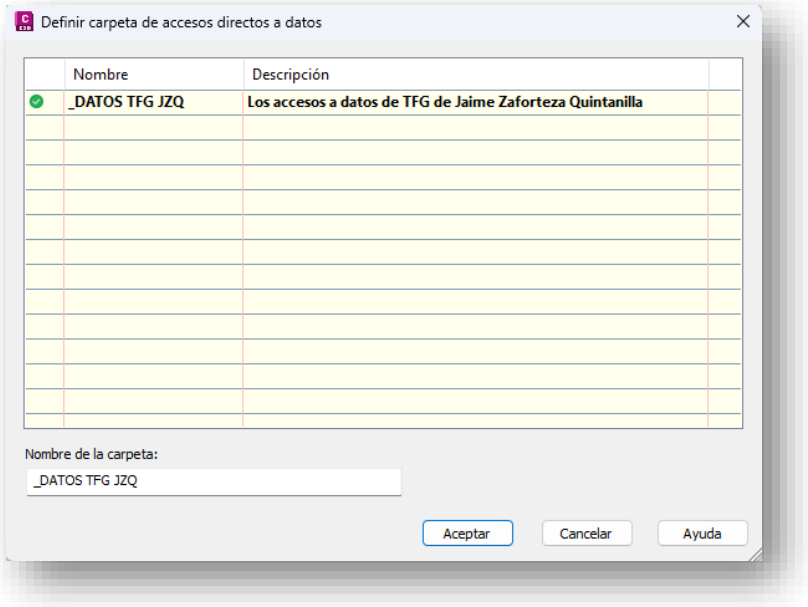
Accionando entonces el control para establecer una Nueva Carpeta de Proyecto a accesos directos, se nos mostrará un formulario para crear y documentar la nueva carpeta en la ruta que nosotros deseemos. Cabe recordar que como se ha matizado con anterioridad vamos a trabajar contra el disco duro central de la máquina. Es decir, nuestra carpeta de proyecto estará en la ruta *"C:\TFG JAIME"* para que cualquiera pueda usar la carpeta en esa misma ruta sin perder posteriormente las referencias.

Figura 80 – Formulario para Nueva Carpeta de accesos directos a datos (Fuente: Elaboración Propia)



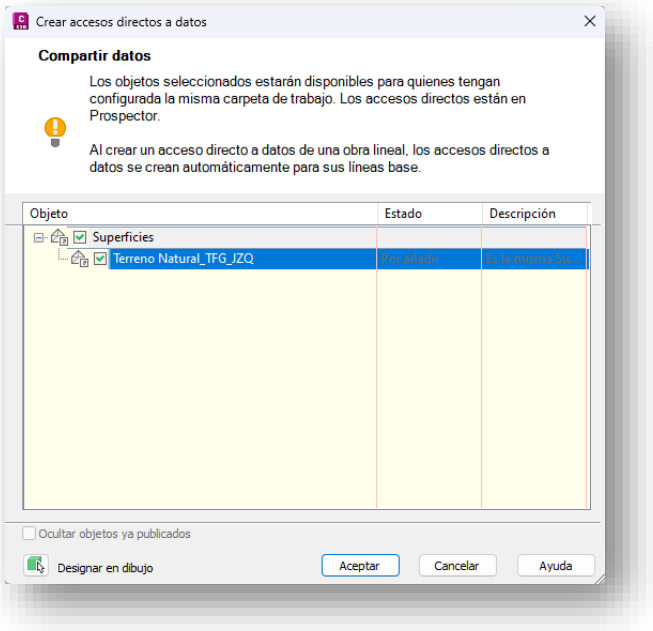
A continuación, se debe establecer la carpeta de accesos directos, una vez la hayamos creado nos debería de aparecer en el formulario que acciona dicha funcionalidad.

Figura 81 – Formulario de Definición de la carpeta de accesos directos del proyecto (Fuente: Elaboración Propia)



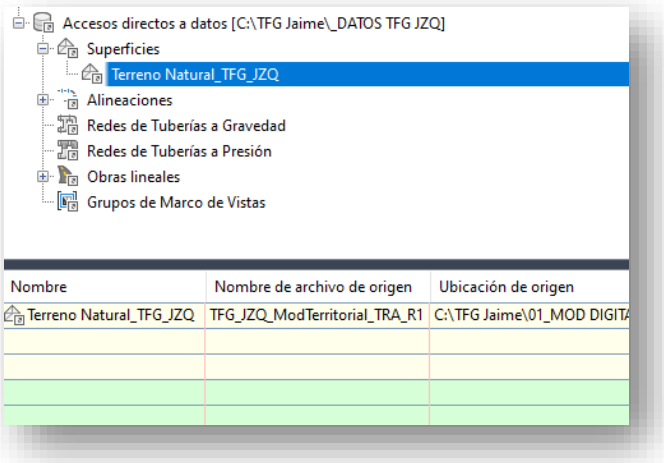
Una vez está la carpeta de trabajo generada y seleccionada dentro del proyecto con el que se desea estar trabajando podemos generar accesos directos de los elementos que dispongamos en nuestro dibujo actual. Hemos de accionar la funcionalidad del control de *Crear accesos directos a datos* en la barra de herramientas del *ribbon* o la opción de *Abrir Project explorer* en el menú desplegable al accionar un clic derecho sobre la base de datos cargada del proyecto. Ambas opciones muestran el formulario que se presenta en la siguiente figura.

Figura 82 – Formulario de Creación de Accesos Directos a Datos (Fuente: Elaboración Propia)



Como se puede apreciar en la figura en esta ocasión al estar trabajando con el modelo digital del terreno, únicamente aparece la opción de compartir un acceso directo a los datos de la superficie generada. Se debe seleccionar la superficie del modelo que se desee almacenar en la base de datos directos accesible posteriormente. La selección puede realizarse mediante la activación de la casilla, tanto de todas las superficies como de la única superficie que disponemos en el modelo, con un "check".

Figura 83 – Prospector de Acceso directo a datos de la carpeta de trabajo definida (Fuente: Elaboración Propia)



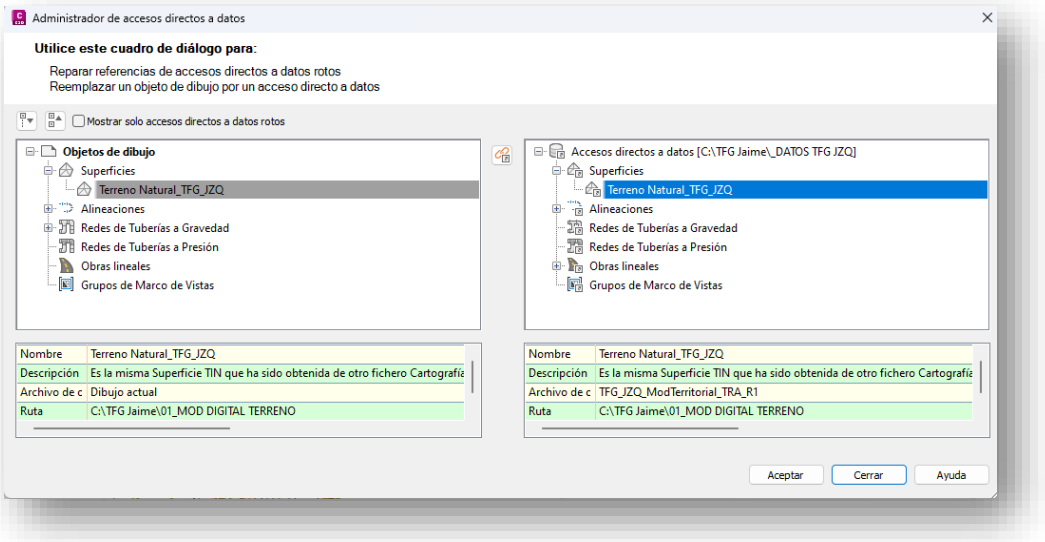
Una vez generado el acceso directo a la superficie del modelo esta debe de aparecer dentro del desplegable de la base de datos de accesos directos que puede consultarse en el prospector del dibujo. Como puede apreciarse en la figura anterior en el grupo de superficies de acceso directo a datos ya nos aparece *Terreno Natural_TFG_JZQ* siendo esa la superficie compartida.

Cuando se selecciona un elemento registrado en la carpeta de accesos directos a datos en el prospector se despliega su información contenida en la ventana inferior del mismo prospector, así puede consultarse la información del elemento original al que estamos accediendo y cobra sentido la documentación del modelo.

Asimismo, cabe mostrar que Civil3D brinda herramientas que permiten la consulta de estos datos contenidos en los elementos de accesos directos con una mayor versatilidad. La opción de *Administrar accesos directos a datos*

que se puede encontrar en la paleta de herramientas es la encargada de ello.

Figura 84 – Formulario de Administración de accesos directos a datos del dibujo (Fuente: Elaboración Propia)



A partir del formulario que muestra, es posible contrastar la información sobre todos los objetos de Civil3D del dibujo con el que se trabaja, inclusive las referencias generadas sobre este, posibilitando así la capacidad de verificar la información. Como se puede apreciar en esta ocasión ambos objetos seleccionados son el mismo, así lo demuestran también sus metadatos. Esta herramienta nos será de gran ayuda para mostrar los objetos del dibujo con el que se trabaje en conjunto con la base de datos de accesos directos de la que se dispone.

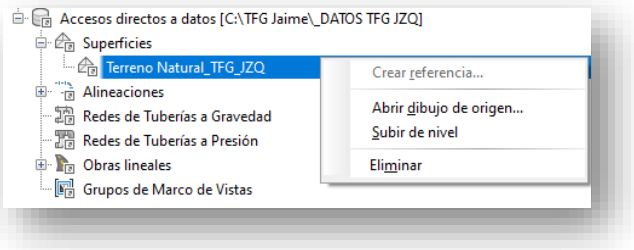
MODELO DE TRAZADO

De forma análoga se ha procedido a reestructurar cada uno de los modelos de los ejes, al ser una dinámica similar a la del modelo de superficies anterior y la misma técnica para cada modelo de cada tramo del trazado, como ejemplificación únicamente se mostrará uno de los modelos de los datos originales de los ejes del trazado.

Por lo tanto, se toma contacto con el modelo que contiene el trazado de la *Paforma de Alta Velocidad de Doble Vía* que recibe la nomenclatura de *TFG_JZQ_E63_AV_Doblevia.dwg*, tal y como se puede consultar en la figura del nuevo CDE propuesto.

En primer lugar, para situar geográficamente el eje propuesto se genera el acceso directo a la superficie del modelo digital del terreno. Se procede de tal forma que se acciona un clic derecho sobre el objeto en el desplegable de accesos directos a datos de las superficies del prospector. Entonces aparece la opción de *Crear referencia…*

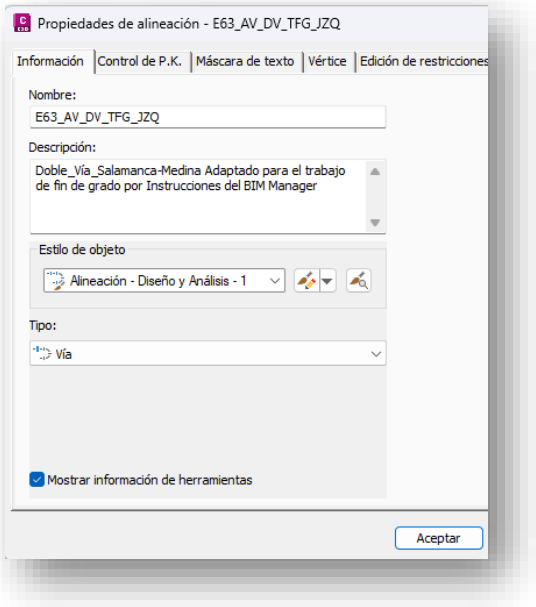
Figura 85 – Desplegable para generar referencia de acceso directo a datos (Fuente: Elaboración Propia)



Una vez generada la referencia a la superficie esta podrá ser visualizada en el dibujo, asimismo se continúa reestructurando la información de los ejes concebidos por el autor del proyecto de línea ferroviaria. Él diseño del trazado, como se ha comentado previamente, no es objeto del presente proyecto por lo que, al igual que para la superficie del terreno, se aplica la nomenclatura apropiada para la adaptación y se documentan los objetos del

trazado.

Figura 86 – Propiedades de alineación del trazado E63 AV Doblevía (Fuente: Elaboración Propia)

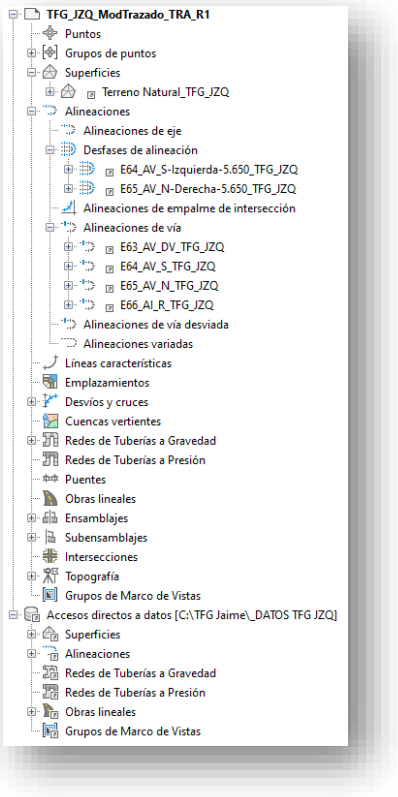


Una vez terminada la edición de cada uno de los datos de los dibujos de trazado de ejes recibidos, se genera los accesos directos a datos de cada uno de los objetos procediendo con la misma dinámica que la planteada para el objeto de superficie abordado con anterioridad.

Por lo tanto, una vez que se ha gestionado toda la información del trazado en la base de datos generada, se puede continuar con la concepción del modelo BIM central correspondiente a la disciplina de Trazado. Este consiste en un modelo vacío en que se unirán todos los objetos de trazado recibidos y tratados de los que se dispone en la base de acceso directo a datos.

Figura 87 – Modelo de Trazado (Fuente: Elaboración Propia)

Figura 88 – Prospector del Modelo de Trazado (Fuente: Elaboración Propia)



En la figura se puede apreciar el prospector de los objetos presentes en el modelo, todo consiste en referencias externas actuando así el modelo central como un contenedor de información externa no editable.

MODELO DE OBRAS LINEALES

Para el modelo de Obras Lineales, se debe profundizar en el concepto de los objetos de Obras Lineales de Civil3D, dado que no son elementos sencillos de generar. Estos objetos representan las infraestructuras lineales, tales como carreteras, ferrocarriles, canales. Su creación y gestión requieren una comprensión detallada de varios de los componentes y procesos específicos de Civil3D.

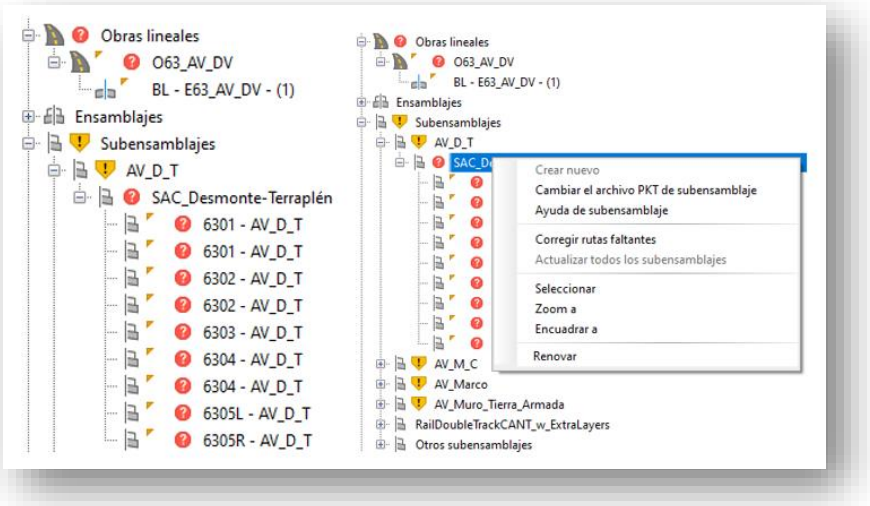
En primer lugar, se ha conservado cada uno de los modelos de Obra Lineal individual para cada tramo del trazado diseñado, tal como se ilustra en la figura del nuevo CDE propuesto. La reestructuración de estos modelos individuales implica la regeneración de todas las Obras Lineales con las referencias adecuadas a sus ensamblajes, los cuales, a su vez, necesitan de los subensamblajes generados por el autor en *Subassembly Composer*, SAC en adelante.

OBRA LINEAL (CIVIL 3D)

Los objetos de Obra Lineal en Civil3D son complejos y multifacéticos. Estos objetos se construyen a partir de componentes clave que incluyen alineaciones, perfiles, ensamblajes, subensamblajes, desfases de eje y superficies entre otros. Las alineaciones definen el trazado horizontal de la obra, mientras que los perfiles representan el trazado vertical. Los ensamblajes, o secciones tipo, determinan la geometría transversal de la infraestructura, especificando cómo deben ser las dimensiones y la forma de la obra en cada sección. Los subensamblajes son componentes específicos que se usan para construir los ensamblajes y pueden incluir elementos como carriles, arcenes, cunetas y otros. Además, los desfases de eje son ajustes necesarios en el trazado para cumplir con requisitos específicos de diseño, y las superficies son modelos digitales del terreno que sirven como referencia para el diseño de la obra.

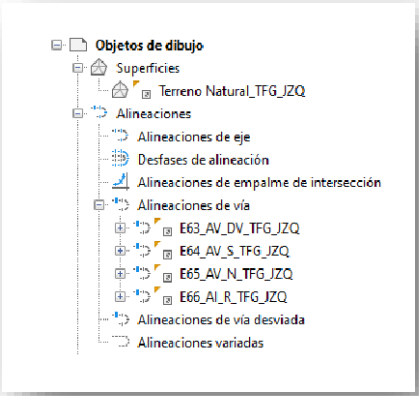
Un aspecto fundamental en la creación de estos componentes es el uso de la herramienta Subassembly Composer. Esta herramienta permite la creación de subensamblajes personalizados que pueden adaptarse a las necesidades específicas del proyecto. En el caso de este proyecto, se ha adaptado el subensamblaje `TIERRA_ARMADA_2024.PKT` para ajustarse a los requisitos del diseño. Todos los subensamblajes modificados se encuentran organizados en la carpeta `0401_SAC` en formato `.pkt`, lo cual facilita su localización y uso en el modelo.

Figura 89 – Reestructuración de los Subensamblajes de los Modelos Nativos recibidos (Fuente: Elaboración Propia)



Como se aprecia en la figura que muestra el desglose de los elementos pertinentes a la obra lineal del prospector, para la correcta reestructuración del modelo, es esencial referenciar todos los subensamblajes que no se encuentren en el modelo existente desde la carpeta `0401_SAC`, a partir de la acción de *Cambiar el archivo PKT de subensamblaje*. Una vez que estos subensamblajes han sido referenciados adecuadamente, se procede a generar accesos directos a la superficie del modelo digital del terreno y a los ejes pertinentes al tramo del trazado en el que se va a realizar la obra.

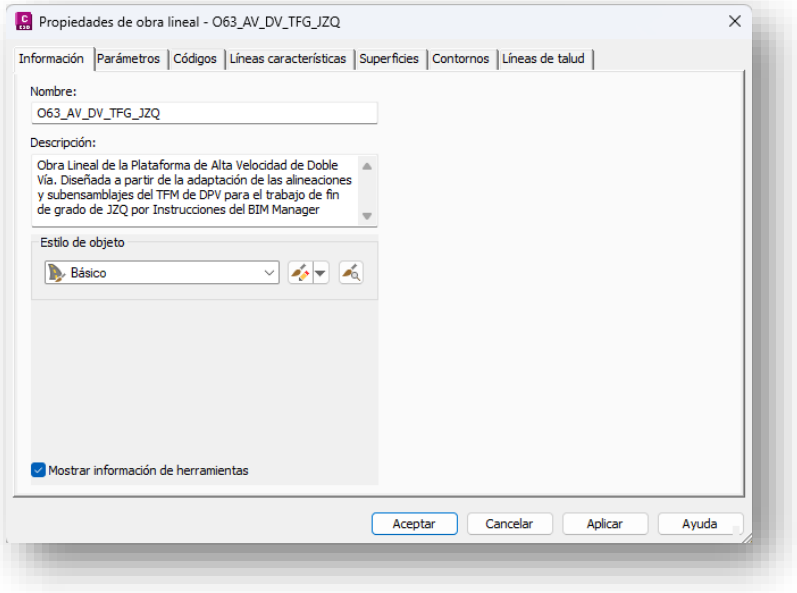
Figura 90 – Administrador de accesos directos a datos del modelo de Obra Lineal para E63 (Fuente: Elaboración Propia)



Este paso muestra la eficiencia de la metodología de acceso directo a datos, ya que garantiza que todos los componentes del modelo estén correctamente conectados y actualizados según las condiciones actuales del proyecto.

Entonces, se comienza a reestructurar la información que corresponde con los objetos de Obra Lineal de los modelos, siguiendo la misma dinámica hasta ahora vista.

Figura 91 –Propiedades de la Obra Lineal (Fuente: Elaboración Propia)



Después de generar los accesos directos necesarios, se deben actualizar los objetivos de la Obra Lineal. Esto implica revisar y ajustar los componentes del modelo para asegurar que todos los elementos estén alineados con los objetivos del proyecto y las especificaciones técnicas requeridas.

Figura 92 – Parámetros de la Obra Lineal (Fuente: Elaboración Propia)

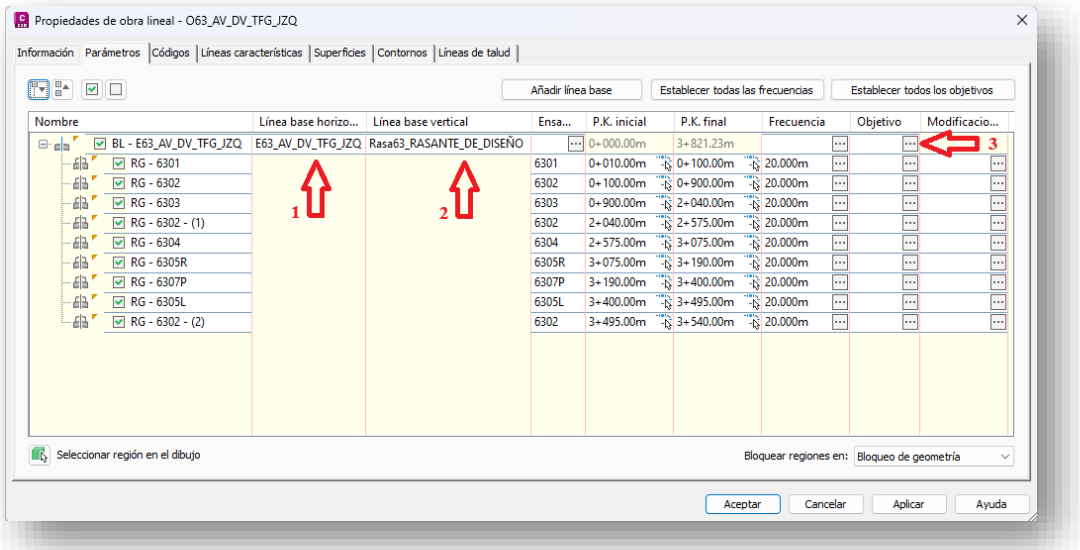
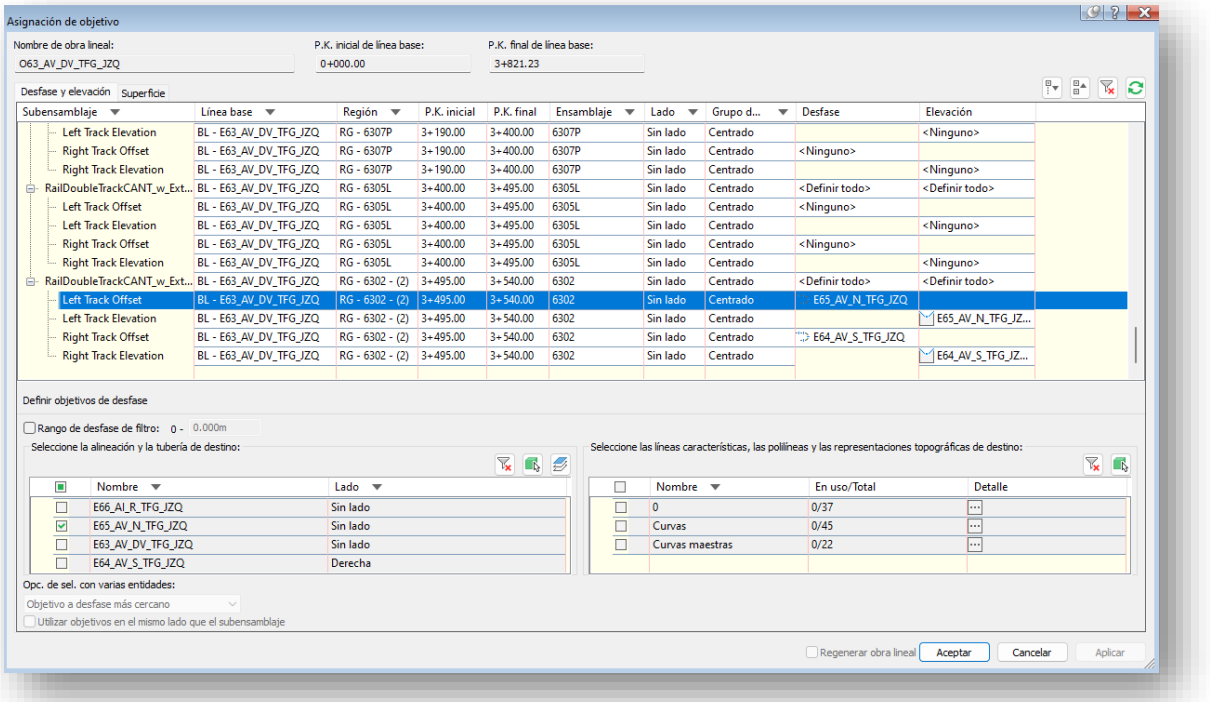
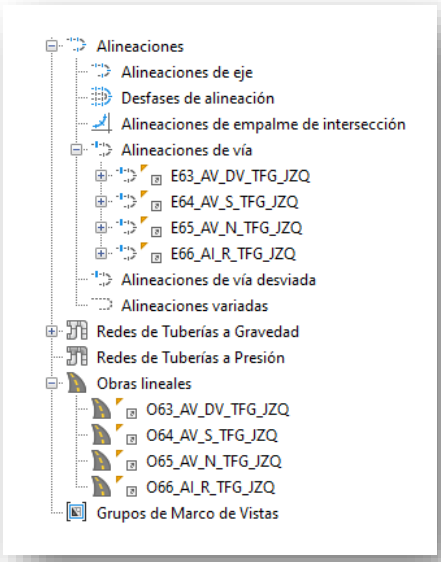


Figura 93 – Reasignación de los Objetivos de la Obra Lineal (Fuente: Elaboración Propia)



Una vez que tanto la línea base horizontal, línea base vertical y todos los objetivos de la obra lineal han sido actualizados a los accesos directos a datos de los ejes que han sido regenerados, se procede a regenerar en el modelo dicha obra lineal y crear el acceso directo.

Figura 94 - Administrador de accesos directos a datos del modelo de Obra Lineal (Fuente: Elaboración Propia)



Será con los accesos directos a cada una de las Obras Lineales del trazado que generaremos los sólidos finales en el modelo de Obra Lineal.

Figura 95 – Propiedades de Obra Lineal, Generación de Superficies (Fuente: Elaboración Propia)

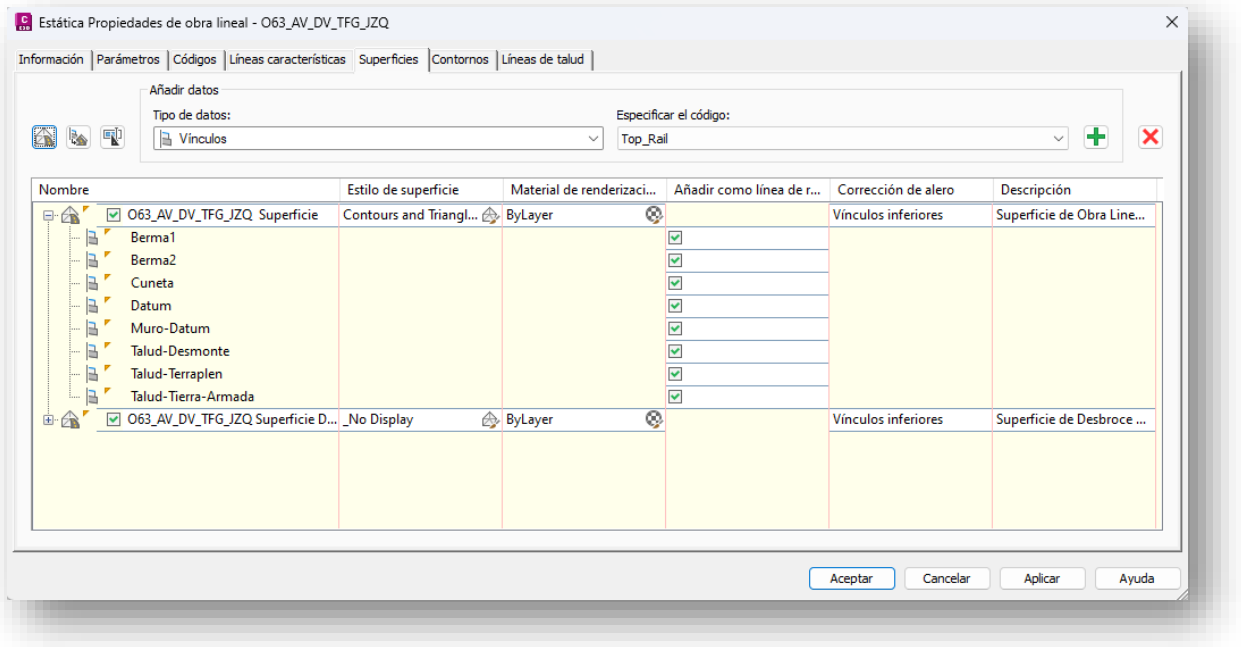
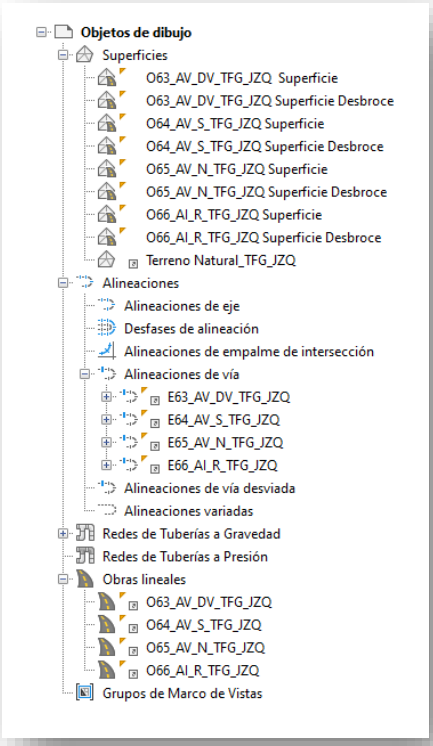


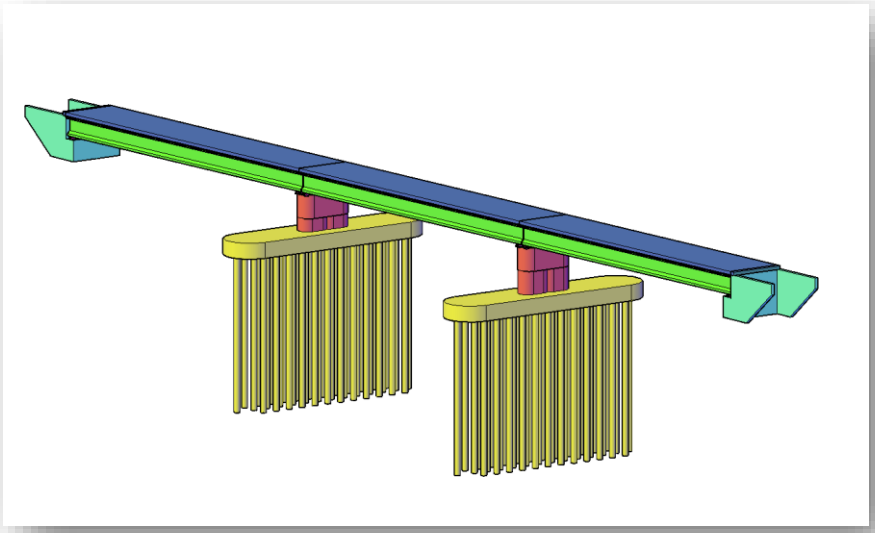
Figura 96 – Objetos del modelo de Obras Lineales (Fuente: Elaboración Propia)



MODELO DE ESTRUCTURA

Por último, el modelo de estructura ha sido el único que no ha sido alterado en materia de documentación. Consiste en la importación de los datos del software de diseño de infraestructuras BIM Infraworks (*.imx*), también de la casa Autodesk. Este archivo es importado en Civil3D, de tal forma que se genera un nuevo objeto en nuestro modelo de naturaleza de *Puentes* de tal forma que disponemos de la estructura diseñada en Civil3D.

Figura 97 – Modelo de Estructura en Civil3D (Fuente: Elaboración Propia)



Como se puede apreciar en la siguiente figura, el modelo estructural importado desde InfraWorks a Civil3D, se genera como objeto de *Puente* en el prospector. En la parte inferior de la interfaz es posible consultar cada uno de los elementos sólidos 3D que componen el objeto estructural. Se muestra así el desglose jerárquico que han recibido los estructurales constructivos definidos por el autor.

Figura 98 – Prospector del Objeto de Puente (Fuente: Elaboración Propia)

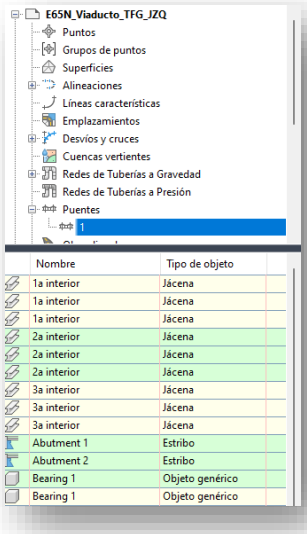
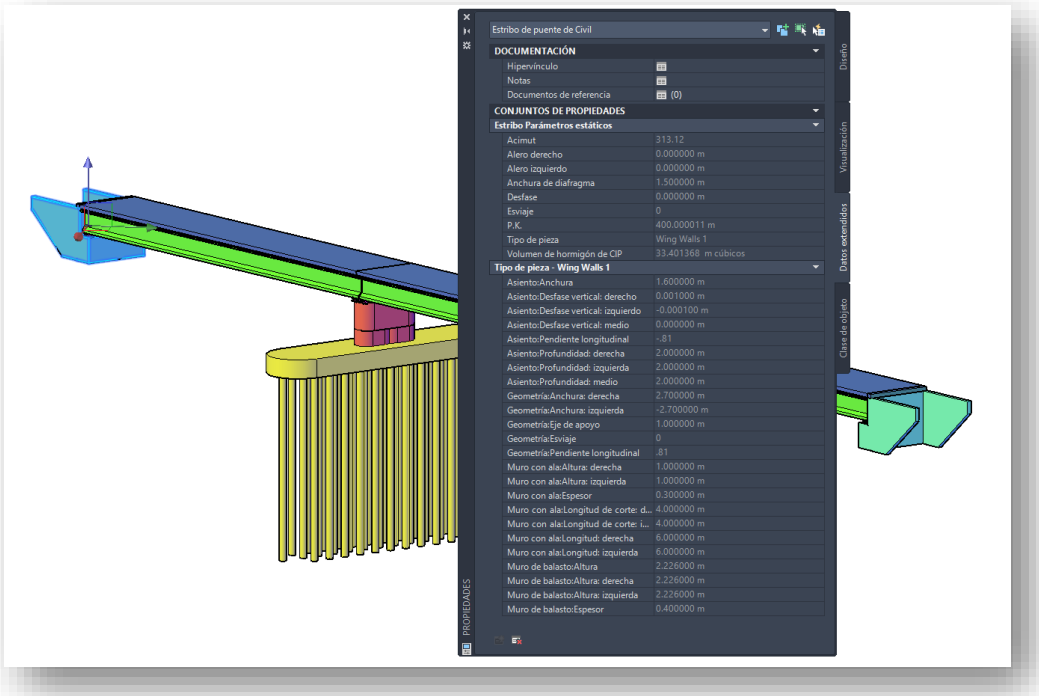


Figura 99 – Conjuntos de Propiedades implícitos a los elementos de la Estructura (Fuente: Elaboración Propia)



Asimismo, se ha analizado que cada uno de los elementos previamente mencionados dispone de una serie de conjuntos de propiedades (*PropertySets)* asignados que acotan los valores geométricos de diseño y estructurales.

La organización de datos, la claridad de la información y su documentación son una de las claves de la metodología BIM. En ocasiones, se producen malentendidos de la filosofía BIM, considerando únicamente representaciones gráficas o "maquetas digitales". Sin embargo, estas representaciones son consecuencia de la adecuada gestión y estructuración de la información, un modelo sin información no será definido como un modelo BIM. El verdadero valor de esta metodología radica en la integración y gestión de la información, un modelo debe ser una base de datos viva, conteniendo los detalles exigidos sobre cada componente del proyecto, desde materiales y especificaciones técnicas hasta cronogramas y costos.

6.2.4 Enriquecimiento del Modelo

Una vez los modelos nativos PIM del proyecto han sido recibidos, verificados, limpiados y reestructurados bajo instrucciones del BIM Manager, se procede al enriquecimiento de la información que ha de contener. Este es un paso clave a la hora de desarrollar el modelo, tras la investigación de las vías de enriquecimiento acorde a lo que afirma Ramos [36], se concluye que puede realizarse sobre el formato nativo del diseño, que en el presente estudio es *.dwg*, o sobre el propio formato abierto (.ifc) resultado de la exportación del nativo.

Por lo tanto, en alineación con los objetivos de investigación de este trabajo de fin de grado en el caso de estudio se evaluarán ambas alternativas como propuestas de procesos de incorporación de datos sobre el modelo. Se plantea por tanto que mediante un flujo automatizado se incorpore toda la infraestructura de propiedades necesarias para acotar los modelos.

¿QUÉ INFORMACIÓN ES NECESARIA?

Ha sido ampliamente comentada la imperante necesidad de aportar agrupaciones de datos, organizados en conjuntos de propiedades, que protagonicen nuestros modelos. No obstante, apenas se ha abordado la cuestión de qué información es realmente necesaria y/o de especial importancia a la hora de definir un modelo. Acorde la normativa [34], la responsabilidad de definición de los requisitos de información del proyecto recaerá siempre sobre el adjudicador de este.

En lo relativo a esta cuestión, se ha tomado un proyecto de fecha actual, como referencia de requisitos de información que solicita el adjudicador, en este caso la administración pública, para los modelos a entregar. En la licitación pública del proyecto de "*OBRA CIVIL Y SUPERESTRUCTURA DE LA METROPOLITANO DE GRANADA. TRAMO ARMILLA-CHURRIANA DE LA VEGA"* consultada en la plataforma de contratación pública de la junta de Andalucía [37] se pudo acceder al pliego de prescripciones técnicas particulares, PPTP en adelante, para la ejecución del proyecto. En él, se puede encontrar un *Anejo BIM* al que han denominado *"Requerimientos BIM (EIR)"*.

En el anejo mencionado se pueden encontrar los requisitos de aplicación de la metodología BIM en el proyecto entre los cuales encontramos la entrega de toda la documentación pertinente en formato abierto IFC. Asimismo, la cuestión de mayor interés respecto a la pregunta planteada se encuentra en lo que la Agencia de Obra Pública de la Junta de Andalucía, AOPJA en adelante, ha definido como los *Niveles de Información No Gráfica* de los modelos.

PPTP – Anejo 1º (4.2.2 Niveles de Información No Gráfica)

La información no gráfica de los elementos de los modelos (metadatos) estará estructurada en torno a una agrupación de propiedades (set de propiedades), definida por la AOPJA. Las propiedades y set de propiedades de los elementos que compondrán los diferentes tarán organizados de forma homogénea y estandarizada. No se admitirán elementos en los modelos que no contengan la estructura de set de propiedades definida por la AOPJA y que a continuación se indica:

[…]

Estos niveles y estructura organizativa de atributos entorno a sets de propiedades de la AOPJA (PSET JAND) serán plenamente visibles y operables en formatos OpenBIM (IFC).

Este fragmento del anejo de requisitos BIM que pertenece a la licitación mencionada, sigue una plantilla definida por la AOPJA para la redacción de sus pliegos, tal y como se puede consultar en su portal en línea [38] que aborda la cuestión de la implantación BIM. En él es posible encontrar un documento de los *requerimientos BIM tipo* que se exigen para pliégos técnicos de licitaciones, que es idéntico al anejo del PPTP.

Figura 100 – Set de Propiedades definida por la AOPJA [1 de 2] (Fuente: Agencia de Obra Pública de la Junta de Andalucía)

SET DE PROPIEDADES DE AOPJA		
IDENTIFICADOR DEL PARÁMETRO	**TIPO CAMPO**	**VALOR POSIBLE**
01_JAND_IDENTIFICACION		
01_01_JAND_PROYECTO	texto	Código de proyecto
01_02_JAND_LOCALIZADOR	texto	Código de localizacion del elemento
01_03_JAND_CLASIFICACION	texto	Código de clasificación del elemento
01_04_JAND_DISCIPLINA	texto	Código de disciplina según el PEB
01_05_JAND_COD_PRESUP	texto	Código del presupuesto
01_06_JAND_DISCIPLINA	texto	Código de disciplina según el PEB
01_07_JAND_SUBDISCIPLINA	texto	Código de subdisciplina según el PEB
01_08_JAND_TRAMO	texto	Código del tramo o subdivisión del proyecto
01_0N_JAND_XXXXXXXXX	texto	Se deberá terminar de configurar y consensuar entre los agentes antes de la entrega del PEB por el adjudicatario
02_JAND_CANTIDADES		
02_01_JAND_UNIDAD	ud	Valor
02_02_JAND_LONGITUD	m	Valor
02_03_JAND_ESPESOR	m	Valor
02_04_JAND_AREA	m2	Valor
02_05_JAND_VOLUMEN	m3	Valor
02_0N_JAND_XXXXXXXX		Se deberá terminar de configurar y consensuar entre los agentes antes de la entrega del PEB por el adjudicatario
03_JAND_PROYECTO		
03_01_JAND_FASE	texto	Código de la fase de obra a la que hace referencia el elemento
03_02_JAND_PLANOS	url*	URL a la ubicación en el CDE de los planos
03_03_JAND_PPTP	url*	URL a la ubicación en el CDE del articulo del PPTP
03_04_JAND_CAP_PRESUP	texto	Código del capítulo del presupuesto en el que se encuentra el elemento
03_05_JAND_SUBCAP_PRESUP	texto	Código del subcapítulo del presupuesto en el que se encuentra el elemento

Cabe destacar que este anejo lo denominan, erróneamente como Requerimientos de Intercambio de Información (EIR) cuando se define claramente la solicitud de los Modelos de Información del Proyecto (PIM) de la fase de ejecución del proyecto y otro Modelo de Información del Activo (AIM) al finalizar el proyecto, donde acotan a su vez los requisitos del modelo del activo "As Built", es decir, Requerimientos de Información del Activo (AIR) para la fase de operación y mantenimiento.

Queda por tanto enmarcada la necesidad, que se reitera a lo largo de esta investigación, de la inclusión de la información sobre los modelos, es más, ahora se ha evidenciado qué información se exige en un entorno real de ámbito profesional y podemos añadir a la simulación de nuestro caso de estudio unos EIR/AIR definidos por la Junta de Andalucía.

Se plantea entonces, el estudio de enriquecimiento de los modelos reestructurados como, la inserción de todos los sets de propiedades y las propiedades que estos incluyen definidas por la AOPJA.

No obstante, acorde al alcance del trabajo de fin de grado, manteniendo así el carácter de propuesta de metodologías de trabajo no una ejecución completa de estas. No será realizado un estudio en detalle del proyecto para el cumplimentado de toda la información intrínseca a los elementos que lo componen, que se asocia con cada una de las propiedades definidas, sino que se plantean las metodologías de trabajo para el profundo enriquecimiento de los modelos.

Figura 101- Set de Propiedades definida por la AOPJA [2 de 2] (Fuente: Agencia de Obra Pública de la Junta de Andalucía)

03_06_JAND_UD_PRESUP	texto	Código de la unidad presupuestaria del elemento
03_07_JAND_XXXXXXXX		Se deberá terminar de configurar y consensuar entre los agentes antes de la entrega del PEB por el adjudicatario
04_JAND_OBRA		
04_01_JAND_TAREA	texto	Código de la tarea del plan de obra a la que pertenece el elemento
04_02_JAND_CERTIFICACION	texto	Número de certificación en la que se incluye dicho elemento
04_03_JAND_ EJECUTADO	número	Porcentaje del elemento ejecutado en certificación
04_04_JAND_ENSAYOS	url*	Ruta para acceder a los documentos del plan de calidad de la obra
04_05_JAND_FICHA_TECNICA	url*	Ruta o nombre para acceder a la ficha técnica en cuestión o su referencia.
04_06_JAND_ASBUILT_PLANO	url*	Ruta o referencia a planos as built, ruta o nombre del documento
04_07_JAND_ASBUILT_DOC	url*	Ruta o referencia a documento as built, ruta o nombre del documento
04_0N_JAND_XXXXXXX		Deberá ser configurado y consensuado entre los agentes antes de la entrega del PEB por el adjudicatario
07_AOPJA_EXPLOT_Y_MANTEN		
07_01_CodigoActivo_COD_GMAO	Alfanuméricos de hasta 16 caracteres	Número de activo unico que se asigna en el GMAO (PRISMA3/4) al activo o lote de activos.
07_02_DenominaciónActivo_COD_GMAO	texto	Denominación de activo que se asigna en el GMAO (PRISMA3/4) al activo o lote de activos.
07_03_TipoActivo_TIP_GMAO	XX-XXX	Tipo de activo en el GMAO (PRISMA3/4) y que lo denomina como "Clase de equipo"
07_04_MantenedorActivo_MAT_GMAO	texto	Mantenedor que tiene asignado dicho activo en el GMAO (PRISMA3/4) y que lo denomina "Unidad de negocio".

STANDARIZED DATA TOOL

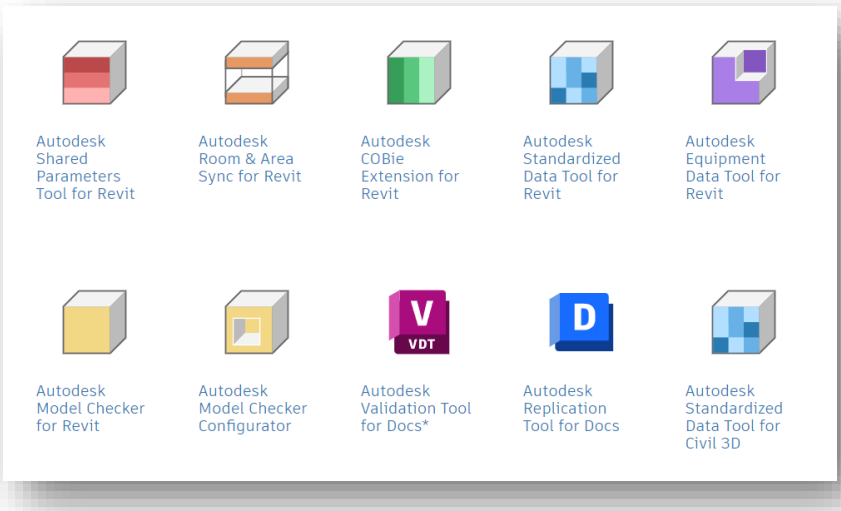
En el ámbito de la arquitectura, la ingeniería y la construcción (AEC), la interoperabilidad es un factor clave para la integración eficiente de flujos de trabajo. Autodesk, como líder en el desarrollo de software para la industria AEC, ha creado una serie de herramientas que buscan facilitar la interoperabilidad dentro de los procesos BIM. En primera instancia, estas herramientas, inicialmente conocidas como **Autodesk BIM Interoperability Tools**, se centraban en un conjunto de aplicaciones sencillas para Revit, orientadas a ayudar a arquitectos, ingenieros y contratistas en la gestión y optimización de sus flujos de trabajo BIM.

Figura 102 – Inicios de Autodesk Interoperability Tools (Fuente: Autodesk)

Con el tiempo, y en respuesta a las necesidades emergentes del sector, estas herramientas han evolucionado, ampliando su alcance y funcionalidad. En la actualidad, se conocen simplemente como **Autodesk Interoperability Tools**, reflejando esta evolución. Además de las aplicaciones iniciales para Revit, se han incorporado nuevas herramientas diseñadas para satisfacer las demandas de una gama más amplia de proyectos, incluyendo aquellos que requieren el uso de Civil 3D. En particular, la herramienta desarrollada para Civil 3D, que se enfoca en la gestión y enriquecimiento de datos, será clave en el presente trabajo.

El *HUB* de Autodesk actualmente incluye una variedad de aplicaciones que amplían las capacidades de interoperabilidad en múltiples plataformas. Las aplicaciones disponibles en este *HUB* se muestran en la siguiente figura:

Figura 103 – Autodesk Interoperability Tools Aplicaciones disponibles (Fuente: Autodesk)



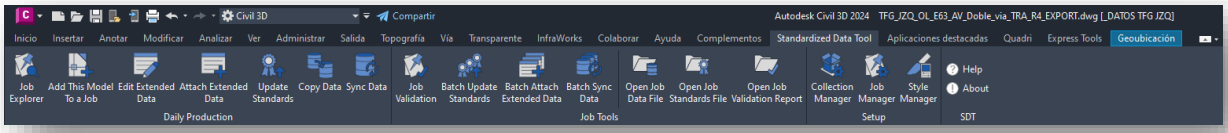
En el presente trabajo, se utilizará la herramienta específica de Civil 3D dentro del conjunto de Autodesk Interoperability Tools para el enriquecimiento de datos del modelo reestructurado, **Standarized Data Tool**. Este proceso es esencial para asegurar que también contenga la información detallada y estructurada que ha sido definida como requisito en el modelo para la gestión del proyecto y la operación futura del activo digital por parte del propietario. Asimismo, cabe destacar que en la fecha de la realización de este trabajo la extensión está únicamente disponible en inglés. Además, apenas se ha encontrado información sobre las funcionalidades

que oferta la herramienta y la forma correcta de usarlas, por lo que el presente estudio abordará su funcionamiento en profundidad definiendo así el uso que se le debe dar acorde a un *webinar* [39] en el que los desarrolladores muestran la herramienta.

Figura 104 – Esquema de flujo de Información de Standarized Data Tool (Fuente: Autodesk)

La herramienta se puede encontrar en el portal de gestión de productos de Autodesk, al igual que se ha mostrado con anterioridad para la herramienta de exportación IFC4.3 (Ver capítulo 5). Una vez está instalada la extensión se debe mostrar en el menú de trabajo una nueva ficha dedicada a la misma que toma su nombre. Como se puede apreciar en la figura, dentro de la ficha mencionada se dispondrá de una gran variedad de herramientas separadas en paneles para la edición de todos los datos contenidos en nuestro modelo.

Figura 105 - Ficha del Ribbon de la herramienta Standarized Data Tools en Civil3D (Fuente: Elaboración Propia)



Cada uno de los paneles está concebido para cumplir una función en específico, dotando al usuario de las herramientas suficientes para establecer un flujo de trabajo orientado a generar y gestionar una base de datos de la información que contiene nuestro modelo o colección de modelos.

Figura 106 – Panel de Producción de Diaria (Fuente: Elaboración Propia)

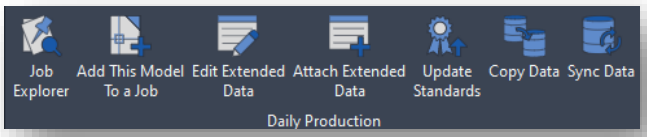
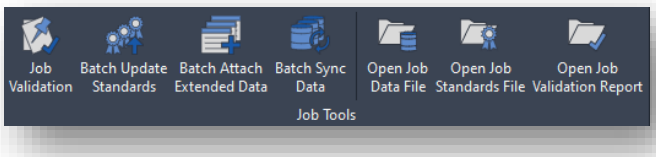
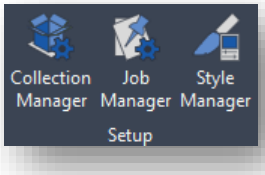


Figura 107 – Panel de Herramientas de Trabajo "Job" (Fuente: Elaboración Propia)



En primera instancia el panel con el que se ha de iniciar será el de configuración o lo que se conoce como el *setup* en terminología anglosajona. En dicho panel podemos apreciar que se introducen varios conceptos, *Collection*, *Job* y *Style Manager*.

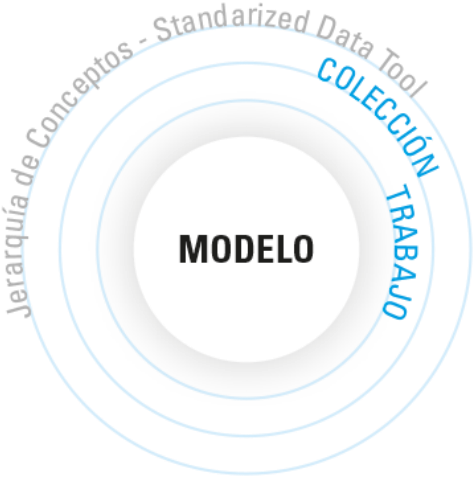
Figura 108 – Panel de Configuración (Fuente: Elaboración Propia)



Esta serie de conceptos son bajo los que la extensión jerarquiza la edición de la serie de propiedades de cada uno de nuestros modelos:

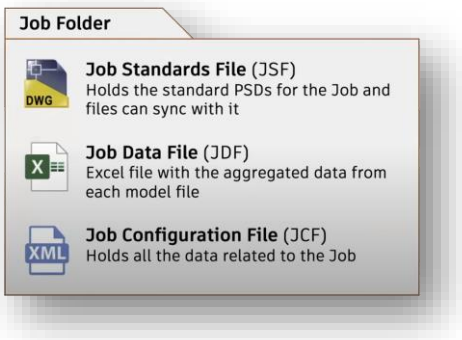
- Un ***job*** o **trabajo**, consiste en una serie de **modelos** (.dwg), de tal forma que cada uno de los modelos será parte de un trabajo. Este concepto sería el análogo a lo que se suele denominar proyecto.
- Una ***collection*** o **colección**, consiste en una serie de Trabajos, está concebida para grandes firmas que puedan necesitar organizar en mayor escala por estilo de proyecto o clientes. No obstante, este supertipo jerárquico de los trabajos es opcional y no necesario a la hora de trabajar con la herramienta.

Figura 109 – Jerarquía de Conceptos Standarized Data Tool (Fuente: Elaboración Propia)



Por lo tanto, la unidad básica con la que se manipula las propiedades que debemos de definir para los modelos que componen el caso de estudio será la del trabajo (Job). Cuando definimos un trabajo se genera una carpeta contenedora de la estructura que lo define:

Figura 110 – Estructura de un Trabajo (Fuente: Autodesk)

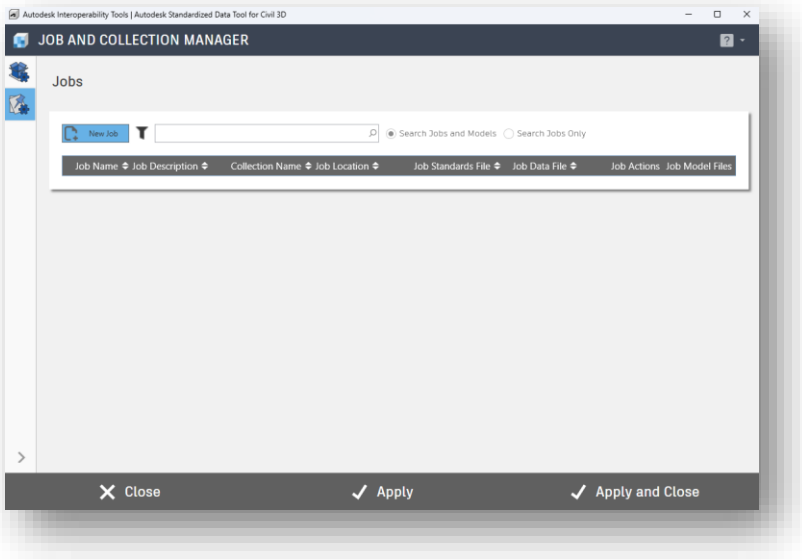


En esta carpeta se contiene:

- Un archivo estándar del trabajo de Civil3D (.dwg), que contiene todas las definiciones de los conjuntos de propiedades para el trabajo, de tal forma que el resto de los archivos puedan sincronizarse con él.
- Un archivo de datos del trabajo de Excel (.xlsx) que contiene los datos agregados de cada uno de nuestros modelos.
- Un archivo de configuración del trabajo en formato de lenguaje de marcado extensible (.xml), que contiene todos los datos relacionados con el trabajo y su funcionamiento en forma de texto.

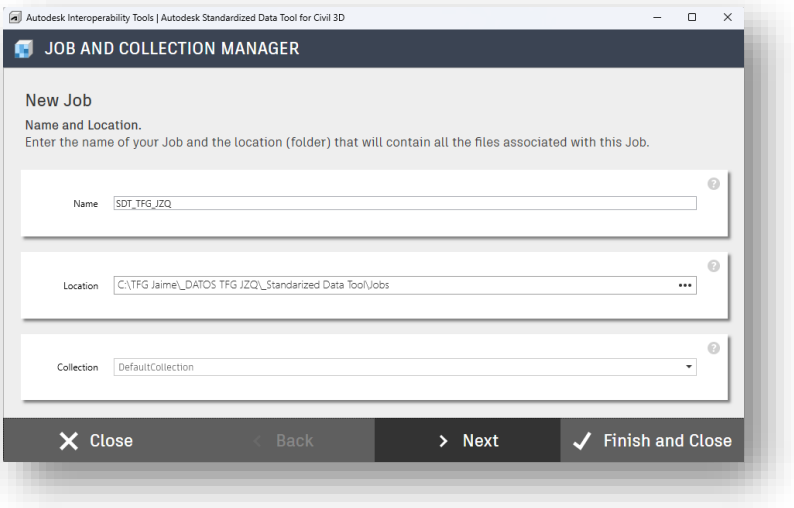
Vamos a comenzar entonces creando nuestro trabajo (*job*) para nuestro caso de estudio, por lo tanto, vamos al panel de configuración previamente mostrado (ver Figura 108 – Panel de Configuración (Fuente: Elaboración Propia)) y accionamos el botón de *Job Manager* (Administrador de Trabajos). Se abrirá una ventana como la que se muestra en la siguiente figura:

Figura 111 – Administrador de Trabajos (Fuente: Elaboración Propia)



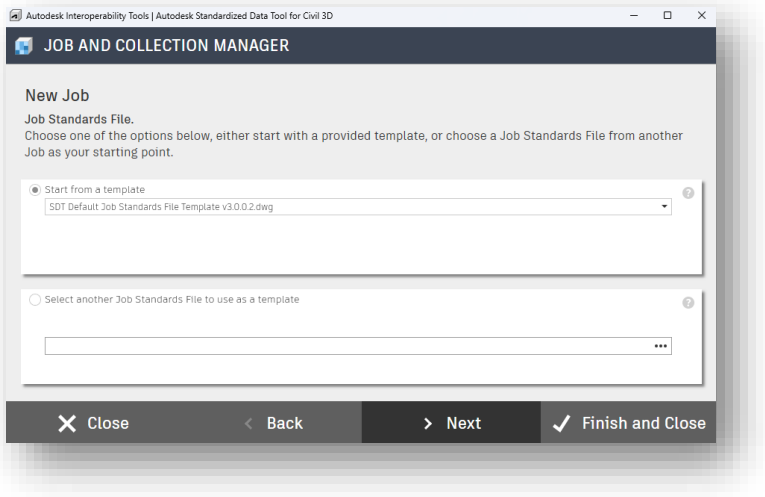
Dentro del Administrador de Trabajos lo primeo que se debe hacer es crear uno por lo que acto seguido de abrir esta ventana accionamos el botón que se encuentra resaltado en azul para generar un nuevo trabajo, *New Job*.

Figura 112 – Ventana de generar nuevo trabajo I (Fuente: Elaboración Propia)



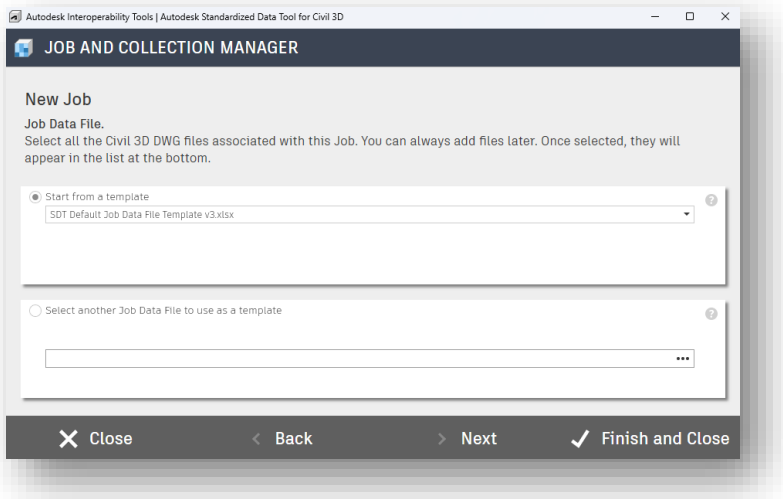
Esta ventana que nos muestra a continuación es la primera de una serie de pasos que debemos cumplimentar para generar así el trabajo del proyecto de forma satisfactoria. En este formulario inicial indicamos el nombre del trabajo, en este caso *SDT_TFG_JZQ* y como ubicación del archivo la carpeta de datos (*_DATOS*) del CDE con el que se trabaja. La colección, como se ha mencionado, al ser opcional se deja como colección por defecto, *DefaultCollection*. Una vez se ha terminado de cumplimentar se ha de seguir al siguiente formulario presionando el botón inferior de siguiente, *Next.*

Figura 113 - Ventana de generar nuevo trabajo II (Fuente: Elaboración Propia)



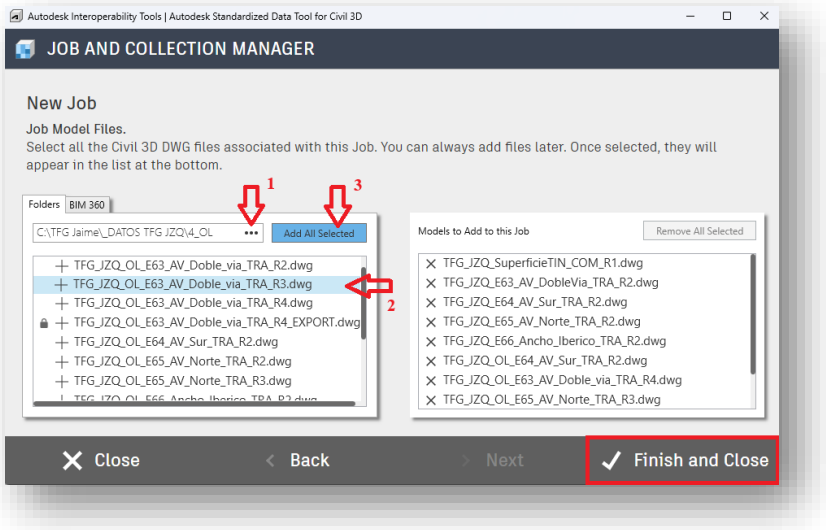
El siguiente formulario demanda que se seleccione una plantilla para la generación del archivo estándar del trabajo de Civil3D (.dwg) que ha sido comentado con anterioridad. En este caso, continuamos con una plantilla por defecto que nos proporciona la extensión. Se continua al siguiente formulario presionando el mismo botón de siguiente, *Next*.

Figura 114 - Ventana de generar nuevo trabajo III (Fuente: Elaboración Propia)



Esta vez, de forma análoga a la anterior, solicita una plantilla en este caso para la generación del archivo de datos del trabajo de Excel (.xlsx). En este caso se continuará con la plantilla proporcionada por la extensión, sin cambiar nada seguimos al siguiente formulario.

Figura 115 - Ventana de generar nuevo trabajo IV(Fuente: Elaboración Propia)



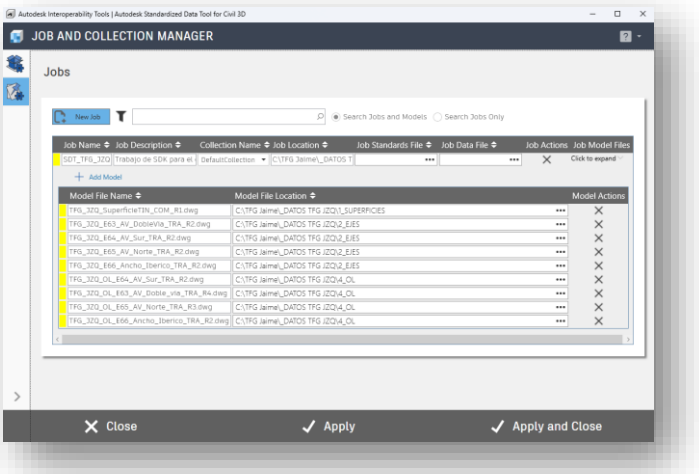
En este último formulario para generar el trabajo del caso de estudio se han de seguir los pasos mostrados en la figura:

1. Seleccionamos el símbolo de los tres puntos que hace alusión al explorador de archivos, para que así sea seleccionada la carpeta sobre la que hemos de consultar los ficheros de los modelos que se han de añadir al trabajo.
2. Una vez seleccionada la carpeta en la que se encuentran los modelos nos aparecerán en la lista todos los ficheros (.dwg) que se encuentran en ese directorio. Se deben seleccionar cada uno de los modelos que vamos a introducir.
3. Cuando ya estén todos los modelos que deseamos añadir seleccionados, se denota mediante un ligero color azul, presionamos el botón de añadir todos los ficheros seleccionados *"Add All Selected"*.

Este proceso puede repetirse para las distintas carpetas contenedoras de los modelos que debamos de añadir, en este caso se ha repetido para las carpetas de cada una de las disciplinas del proyecto (ver Figura 75 – Estructura de Datos Propuesta basada en _Shortcuts (Fuente: Elaboración Propia)).

Añadidos todos los modelos que se deseen, se presionará el botón de finalizar y cerrar *"Finish and Close"* resaltado en rojo en la figura. Así se habrá creado el trabajo de nuestro proyecto y se podrá comenzar a manipular los datos que contendrán los modelos.

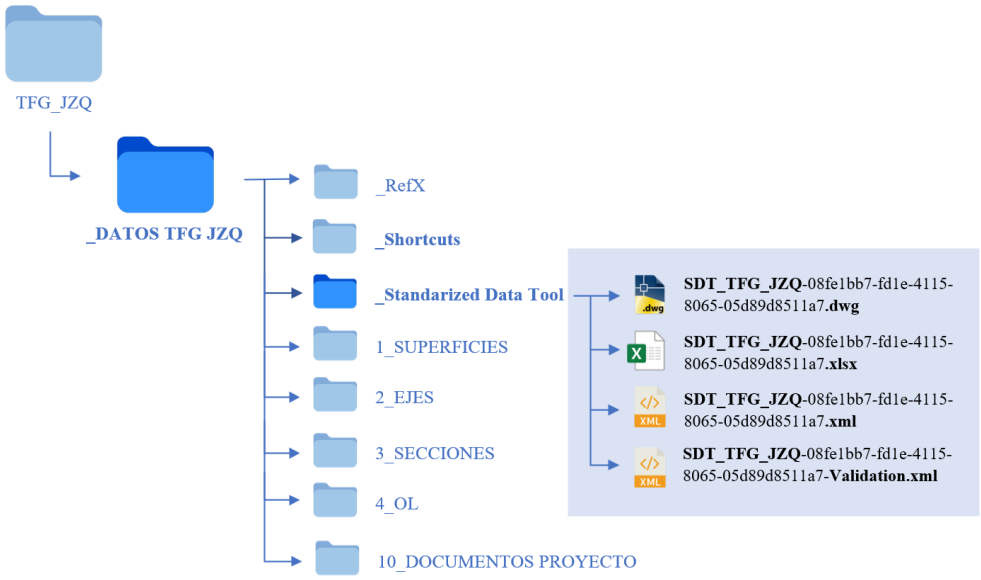
Figura 116 – Administrador de Trabajos de Standarized Data Tool (Fuente: Elaboración Propia)



Ya creado el trabajo, el administrador de trabajos nos mostrará toda la información que lo compone, se podrá consultar así los modelos que han sido introducidos, nombre, descripción, etc.

El trabajo generado, nos ha creado los archivos que lo componen en la carpeta del CDE que se ha seleccionado en el proceso, cabe destacar que esa carpeta fue creada antes de comenzar el proceso no es producto de la herramienta, así los archivos que componen nuestra tarea quedarán almacenados en la misma estructura de carpetas definida para que otros usuarios sigan trabajando sin pérdida de información.

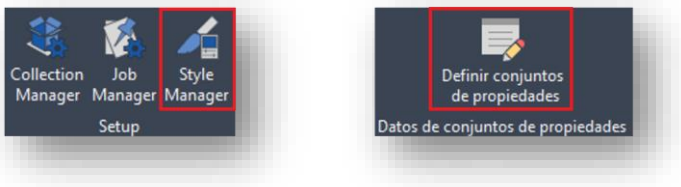
Figura 117 – Estructura de Archivos generados Standarized Data Tool (Fuente: Elaboración Propia)



Una vez generado nuestro trabajo, podemos continuar con la generación de los conjuntos de propiedades del proyecto. Para la generación y administración de los conjuntos de propiedades que buscamos generar en común para todos los modelos añadidos al trabajo se ha de configurar en el archivo (.dwg) que se ha generado en la carpeta **_Standarized Data Tool**.

Dentro del fichero descrito, se ha de ir de vuelta a la paleta de configuraciones (ver Figura 108 – Panel de Configuración (Fuente: Elaboración Propia)) para hacer uso de la funcionalidad de administrador de estilos *"Style Manager"*. Cabe remarcar que esta herramienta es la misma funcionalidad que ofrece la de "*Definir Conjuntos de propiedades"* que se encuentra en la ficha de *Administrar*, únicamente ha sido introducida en la ficha de la extensión SDT.

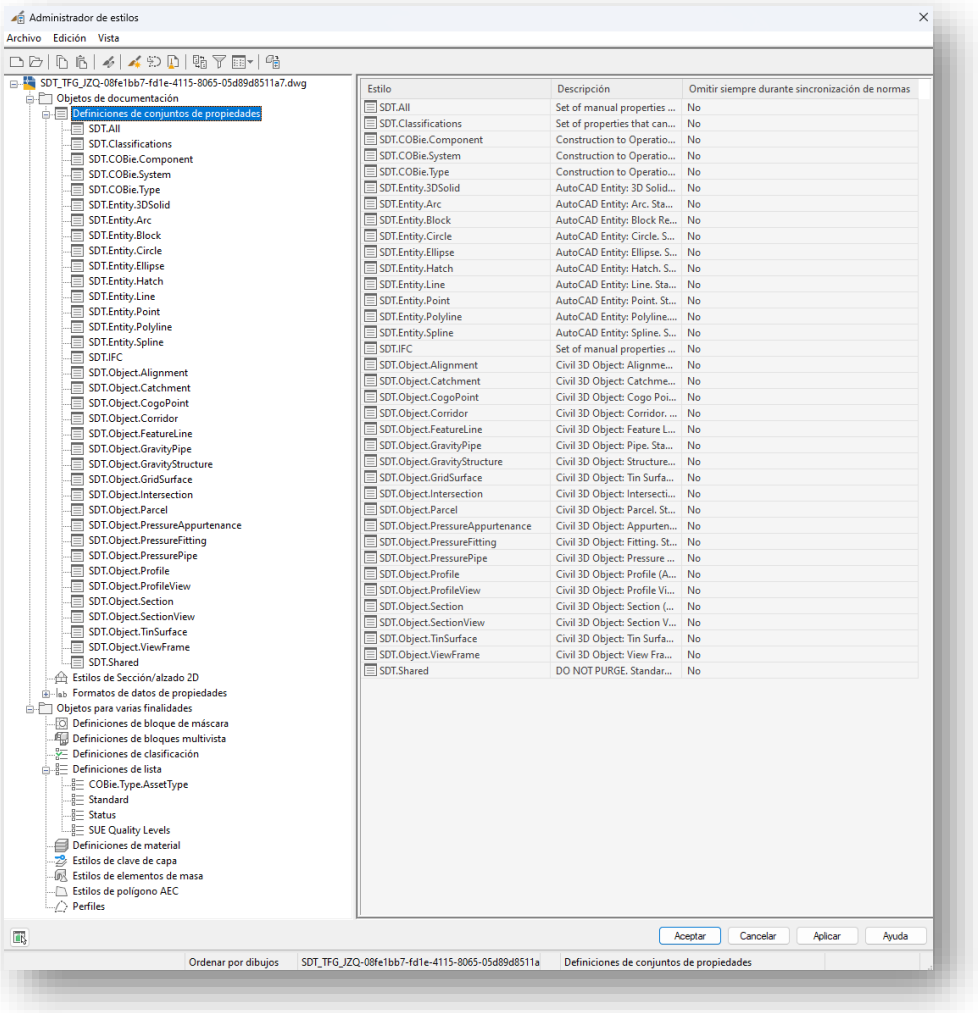
Figura 118 – Herramineta de Definición de Conjuntos de Propiedades (Fuente: Elaboración Propia)



Una vez accedemos al Style Manager de la SDT, podemos consultar cada uno de los conjuntos de propiedades que genera por defecto con el trabajo (job). Dentro del desplegable izquierdo del formulario, accedemos a *Objetos de documentación* para acceder a su vez a las *Definiciones de conjuntos de propiedades* que están en su interior. Así se podrán consultar todas y cada una de las propiedades que contienen los conjuntos que se han contemplado por defecto por los desarrolladores de la herramienta. Estos serán los conjuntos de propiedades que serán sincronizados para todos los modelos que han sido acoplados al trabajo previamente definido.

Asimismo, existe completa libertad para eliminar los conjuntos que no se desee, así como para la introducción de nuevos conjuntos. Además, si se revisan los nombre de cada uno de los conjuntos de propiedades se denota que existen casos específicos para una serie de objetos de Civil3D o entidades de AutoCAD, estas propiedades serán por tanto únicamente aplicables a dichos objetos, una opción interesante que será revisada más adelante.

Figura 119 – Administrador de Estilos para los conjuntos de Propiedades por defecto de SDT (Fuente: Elaboración Propia)



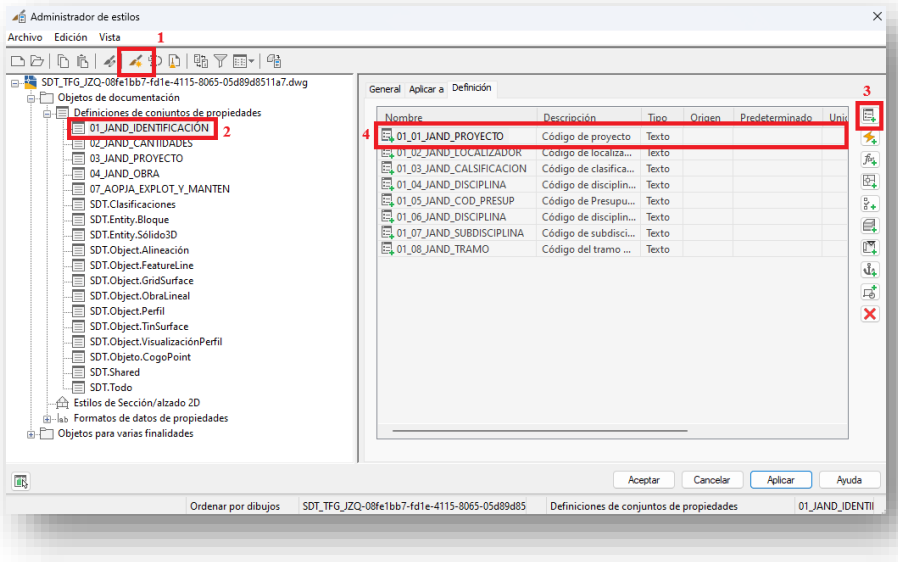
Uno de los problemas principales con estos conjuntos de propiedades predefinidos por la herramienta está en el idioma. Al ser la SDT una extensión del Autodesk Interoperability Tools desarrollada únicamente en inglés, todas y cada una de las propiedades están nombradas y descritas bajo términos anglosajones. Así pues, se seleccionarán los conjuntos más interesantes para el proyecto y serán transcritos al español. No obstante, se reitera que no es necesario el uso de la plantilla y se puede usar otro fichero (.dwg) para definirlo como el archivo estándar del trabajo.

Si se dispone de modelos de proyectos previos con alto enriquecimiento en conjuntos de propiedades que se deseen usar, puede proponerse ese mismo fichero del modelo (.dwg) y seleccionarse como plantilla en lugar de dejar marcada la opción por defecto que se ha seguido en el presente estudio.

Una vez eliminados los conjuntos de propiedades que no se van a usar en el estudio y transcritos al español los que se ha decidido usar, se han de crear los conjuntos definidos por la AOPJA que se han definido previamente. Para ello, se definen los siguientes pasos:

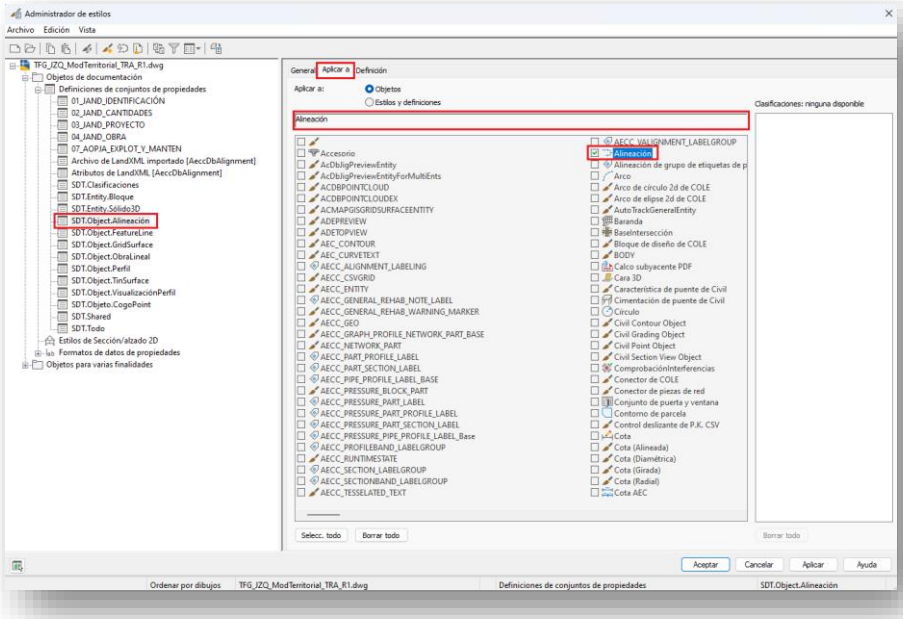
1. Generar un nuevo conjunto de propiedades.
2. Seleccionamos el nuevo conjunto en el explorador del *Administrador de Estilos*.
3. Una vez seleccionado nos aparece a la izquierda un menú vacío, generamos con el botón que proceda (distinto en función del estilo de propiedad) para la propiedad que se desea generar.
4. Generada la propiedad, aparecerá en la tabla de datos donde podremos editar sus metadatos como la descripción, tipo de dato, origen valor predeterminado por defecto, etc.

Figura 120 – Pasos para generar un Conjunto de Propiedades y las Propiedades que contiene (Fuente: Elaboración Propia)



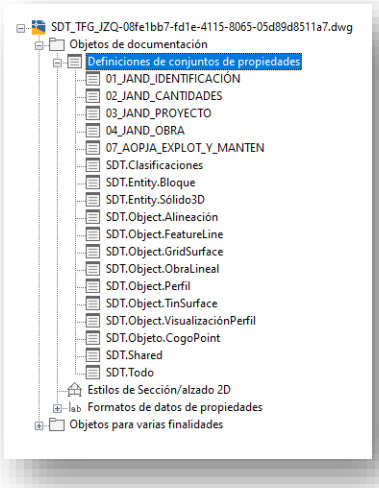
Tras la generación de todos los conjuntos de propiedades que proceden para el proyecto, en el explorador del *Administrador de Estilos* se muestran todos los conjuntos que almacena la plantilla de la SDT. A continuación, se han de generar las propiedades en cada uno de los modelos que han sido definidos en el trabajo de la SDT.

Figura 121 – Conjunto de Propiedades aplicado a Objeto específico de Civil3D (Fuente: Elaboración Propia)



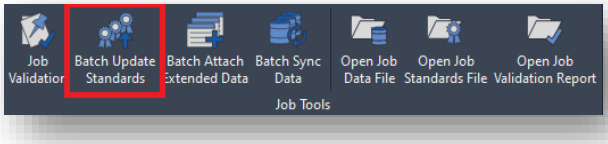
Con todos los conjuntos generados, es un buen momento para remarcar la naturaleza de aplicación de la que podemos dotar a cada uno. Como se aprecia en la figura, el conjunto de la SDT predefinido por defecto que se ha decidido mantener de *SDT.Objeto.Alineación* se aplica únicamente a los objetos de Civil3D a los que hace alusión su nombre. Para verificarlo, se debe acceder al conjunto dentro del explorador y revisar la pestaña que configura el espacio izquierdo del formulario, en este caso se debe seleccionar la pestaña *"Aplicar a"*, dentro del espacio de aplicación se puede seleccionar los objetos sobre los que se podrá aplicar el conjunto de propiedades, en la figura se ha resaltado de color rojo la selección pertinente a las *Alineaciones*. Asimismo, se dispone de la opción de seleccionar todos los objetos de la lista en la esquina inferior izquierda del formulario. A estos efectos, en las propiedades definidas por la AOPJA que se contemplan en el presente caso de estudio se ha tomado la decisión de no aplicar filtrado de objetos y generar los conjuntos habilitados para todos ellos.

Figura 122 – Conjuntos de Propiedades definidos en el archivo estándar del trabajo SDT (Fuente: Elaboración Propia)



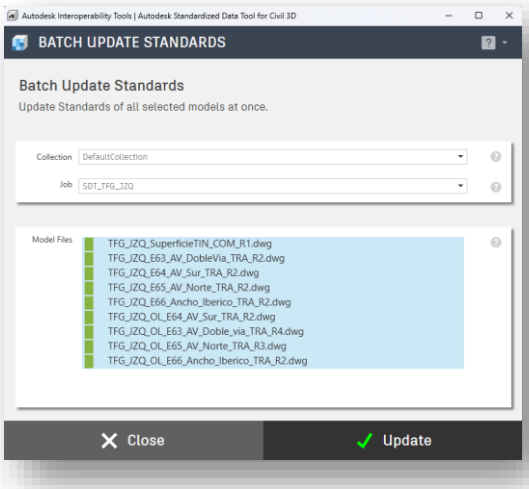
El siguiente paso entonces será lanzar todos esos conjuntos de propiedades desde nuestra plantilla de la herramienta *(SDT_TFG_JZQ[..].dwg)*, es decir el archivo estándar del trabajo, al resto de modelos. Para ello se usa la funcionalidad de actualización de lotes *"Batch Update Standards"* situada en el panel de herramientas de trabajo de la SDT.

Figura 123 – Actualización de Lotes en el ribbon de la SDT (Fuente: Elaboración Propia)



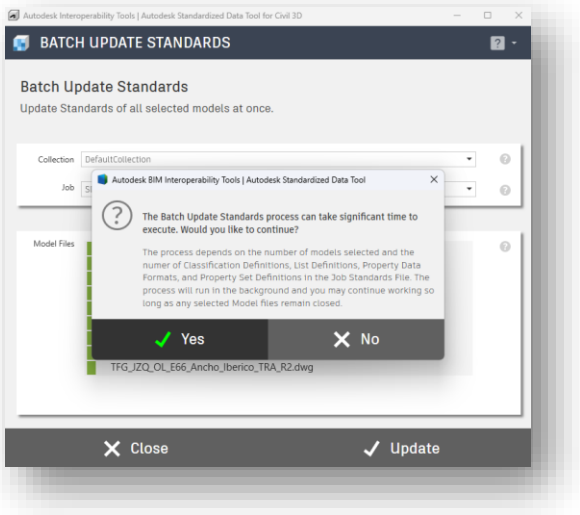
Al iniciar la funcionalidad muestra el siguiente formulario en el que permite filtrar por colección (como se ha comentado, en este caso no aplica) y trabajo. Por lo que se filtra por el trabajo que ha sido definido *SDT_TFG_JZQ* y aparece en el espacio inferior del formulario una lista de los modelos asociados al trabajo seleccionado. Al asegurar que todo está correcto, se ha de presionar el botón inferior de *"Update"*, es decir, actualizar.

Figura 124 – Formulario de Actualización de Lotes (Fuente: Elaboración Propia)



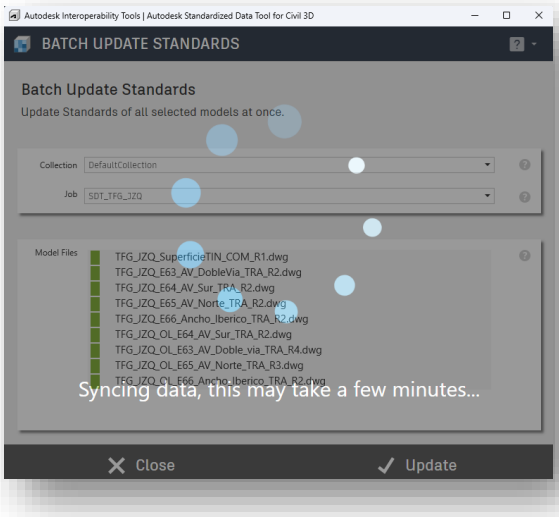
Cabe destacar que los modelos se encuentran resaltados en un ligero color azul porque están seleccionados. Para aplicar la actualización se deben seleccionar todos los modelos del trabajo. Se puede hacer una múltiple selección clicando cada uno de ellos mientras se mantiene la tecla de *ctrl.* presionada.

Figura 125 – Aviso previo a la Actualización de Lotes (Fuente: Elaboración Propia)



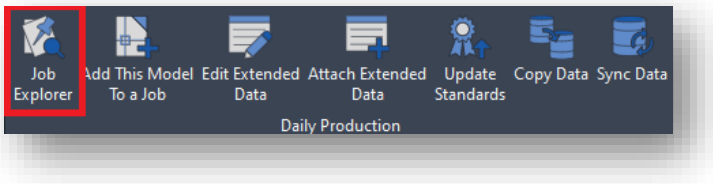
Al presionar el botón mencionado de actualizar *"Update"* se muestra un mensaje de advertencia en inglés que trata de avisar al usuario que este proceso puede tomar una cantidad considerable de tiempo, para asegurarse así de que el usuario desea continuar. Asimismo, remarca que el proceso dependerá del número de modelos que tenga el trabajo asignado y el número de conjuntos de propiedades y propiedades contenidas por cada uno. Además, aclara que el proceso será ejecutado en segundo plano. Si el usuario acepta continuar, ha de presionar el botón *"yes"* que muestra la figura.

Figura 126 – Pantalla de carga de la Actualización de Lotes (Fuente: Elaboración Propia)



Una vez terminado el proceso de actualización de todos los modelos, cada uno de los conjuntos de propiedades definidos se habrán cargado en los modelos y se podrá continuar con la asignación de los conjuntos a cada uno de los objetos de cada modelo. Para ello podemos usar la función de explorador de trabajo *"Job Explorer"*.

Figura 127 – Funcionalidad de explorador de trabajo en el panel de la SDT (Fuente: Elaboración Propia)

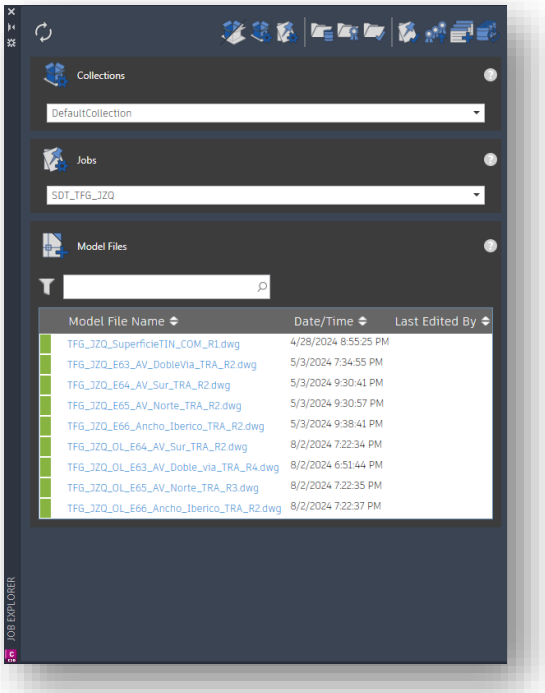


Esta funcionalidad permite acceder a cada uno de los modelos del trabajo que ha sido definido, siguiendo la dinámica de filtrado por colección y trabajo, muestra los modelos acordes a la selección. En este caso muestra los modelos que definimos al crear el trabajo que se corresponde con cada uno de los modelos individuales de cada disciplina. Si acciona uno de los modelos del formulario abrirá el fichero (.dwg) correspondiente.

Con la funcionalidad de actualización de lotes lo que se ha conseguido es tener toda la información necesaria en cada uno de los modelos, por lo que se irá accediendo a cada uno de ellos para definir los objetos que han de disponer de los conjuntos de propiedades que serán accesibles en todos los ficheros marcados, a partir del *"style manager"* de la SDT o la *Definición de Conjuntos de Propiedades* como se ha remarcado con anterioridad.

"Batch" hace alusión a *Lote* lo que implica que está ejecutando la funcionalidad para un lote de modelos dentro de un trabajo y es por ello que hemos de seleccionar los modelos sobre los que se desea aplicar la funcionalidad. Tras el uso de la extensión, se puede afirmar que todas las funcionalidades en su modalidad de *Lote* tienen problemas de funcionamiento. Al ser una herramienta en constante desarrollo, debe de estar acotado para casos concretos o situaciones sencillas. En todo caso siempre que no funcione la modalidad *"Batch"* se puede continuar usando su análogo individual, es decir, si no funciona correctamente la actualización de estándares por lotes que se ha comentado anteriormente *"Batch Update Standards"* se utilizará la opción individual de *"Update Standards"* modelo a modelo. Esa ha sido la metodología que se seguido en el estudio.

Figura 128 – Explorador de Trabajos de la SDT (Figura: Elaboración Propia)

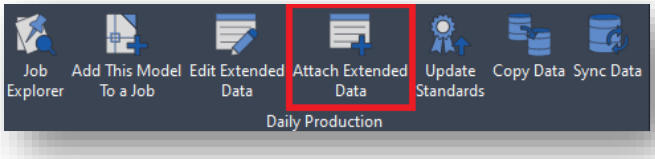


Entonces, una vez accedemos a cualquiera de los modelos que muestra el "*Job Explorer"* se deben de cargar los conjuntos de propiedades en los objetos. La Standarized Data Tool contempla eso en su funcionalidad de editar los datos extendidos, *"Edit Extended Data"*. Esta es la funcionalidad que se utiliza en el webinar consultado que se ha mencionado previamente, no obstante, tras el uso repetido de la extensión se deduce que la funcionalidad de adjuntar los datos extendidos, *"Attach Extended Data"* satisface en mejor medida las

necesidades del estudio. Ambas se distinguen en la dinámica de asignación, es decir, mientras en la opción de editar se ha de efectuar una previa selección en el modelo de los objetos sobre los que serán aplicados los conjuntos de propiedades, la opción de adjuntar permite seleccionar los objetos genéricos que se desean cumplimentar asignando automáticamente a cada uno de ellos que encuentren en el dibujo activo.

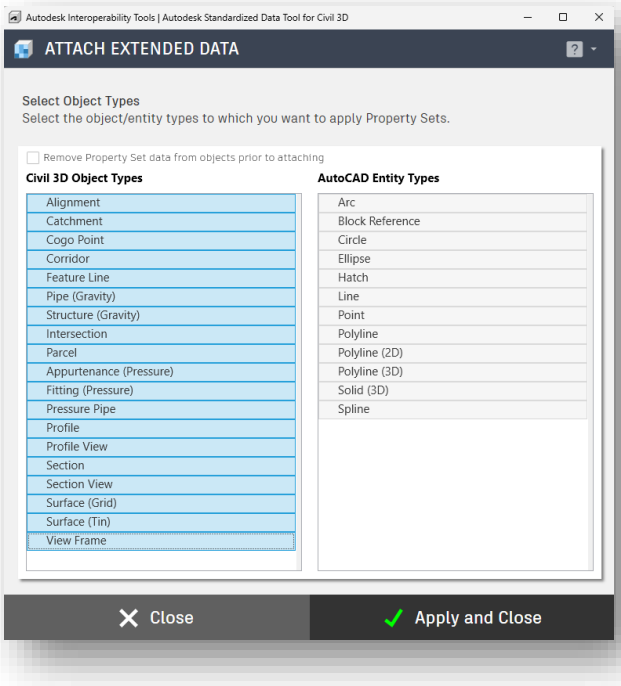
Queda por tanto justificado el uso óptimo que nos ofrece la función de adjuntar los datos extendidos.

Figura 129 – Funcionalidad de adjuntar datos extendidos en el panel de la SDT (Fuente: Elaboración Propia)



Por lo tanto, una vez sea accionada la funcionalidad de adjuntar datos extendidos, se abrirá un formulario que solicita al usuario que selecciones los tipos de objetos de Civil3D y/o los tipos de entidades de AutoCAD. El usuario debe seleccionar todos los objetos sobre los que desea añadir todos los conjuntos de propiedades que han sido definidos previamente en la plantilla. Siguiendo la dinámica de selección de la SDT (ver Figura 124 – Formulario de Actualización de Lotes (Fuente: Elaboración Propia)) seleccionamos en este caso todos los objetos y entidades que nos ofrece el formulario.

Figura 130 – Formulario de Adjuntar Datos Extendidos (Fuente: Elaboración Propia)



Cabe destacar la casilla sin marcar que se encuentra en la parte superior del panel derivado blanco del formulario, esa casilla permite al usuario eliminar los conjuntos de propiedades que puedan disponer ya los objetos sobre los que se va a aplicar los nuevos conjuntos. Así permite un control más sofisticado sobre los datos extendidos del modelo. En este caso se dejará sin marcar, conservando las propiedades que puedan traer los objetos por diseño.

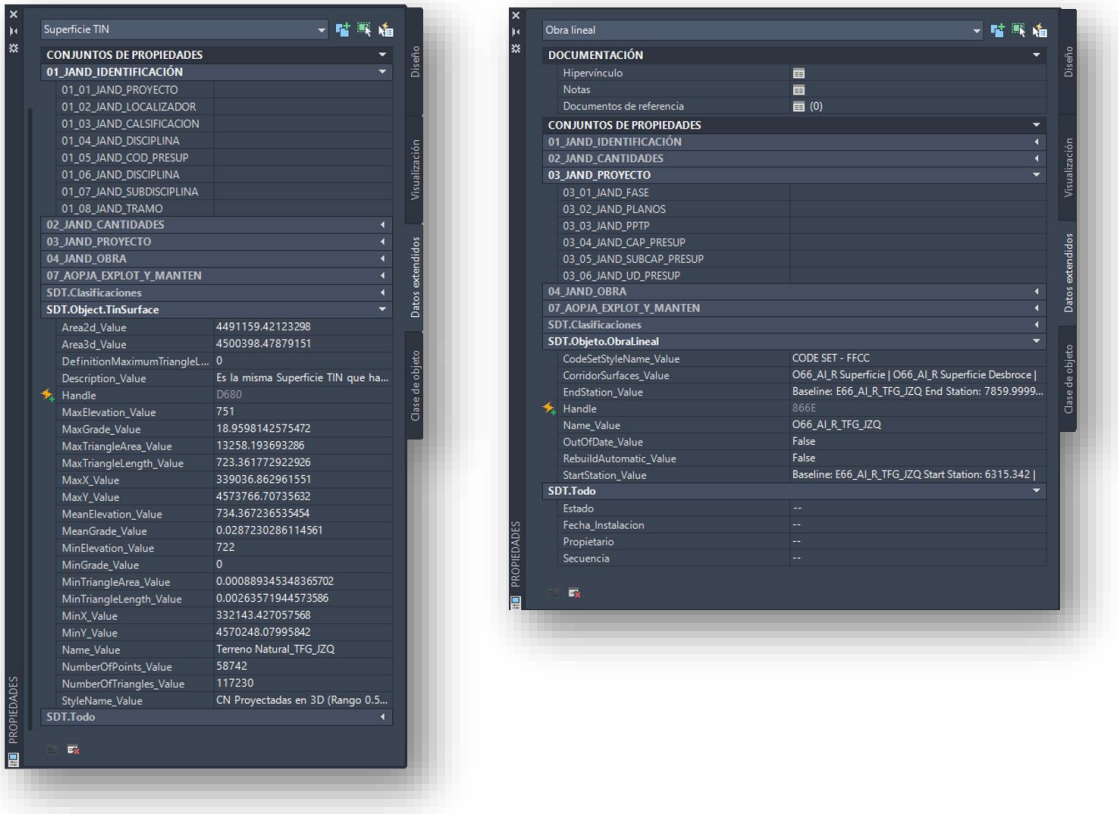
Al presionar el botón de aplicar y cerrar *"Apply and Close"*, se generarán los conjuntos en el modelo y cerrará la herramienta. Las propiedades se generarán vacías por defecto a no ser que se indique lo contrario en la configuración a la hora de generarlas dentro de su conjunto.

Cabe tener en cuenta que, a pesar de haber seleccionado todos los objetos del modelo, como resulta obvio, únicamente se aplicará los conjuntos sobre los objetos existentes en el modelo que estamos ejecutando el formulario. Además, como se ha resaltado con anterioridad, los conjuntos de propiedades pueden ser específicos para una serie de objetos, tal que no se generará en un objeto de *Superficie Tin* un conjunto de propiedades definido únicamente para alineaciones como el *SDT.Objeto.Alineación*.

Los conjuntos de propiedades definidos por la AOPJA serán aplicados a todos los objetos al ser estos de carácter genérico sin especificación de objeto de ningún tipo

Efectuando esta serie de acciones en cada uno de los modelos del trabajo acotado, se dispondrá de los conjuntos de propiedades pertinentes en cada uno de los objetos de tal forma que para culminar el enriquecimiento se deberán cumplimentar los datos en los conjuntos definidos. Es decir, hasta ahora se ha generado el contenedor de los datos a añadir sobre el modelo y se dispone en consecuencia de espacios dedicados al almacenado de cada uno de datos para los objetos que componen los modelos.

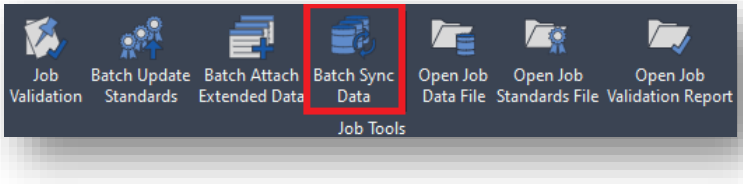
Figura 131 – Propiedades de Superficies y Obras Lineales cargadas en los Modelos (Fuente: Elaboración Propia)



Como se puede apreciar, existen varias propiedades de la SDT que se generan automáticamente con la información del modelo, es decir, a la hora de definir las propiedades se ha especificado unas fórmulas por código para que se autocomplete el valor de la propiedad por defecto. Estos valores se regeneran cada vez que se guarda el fichero, en consecuencia, una vez generado este tipo de propiedades en los objetos del modelo cada vez que se desee guardar se ejecutaran todas esas fórmulas y puede que consuma más tiempo del previsto.

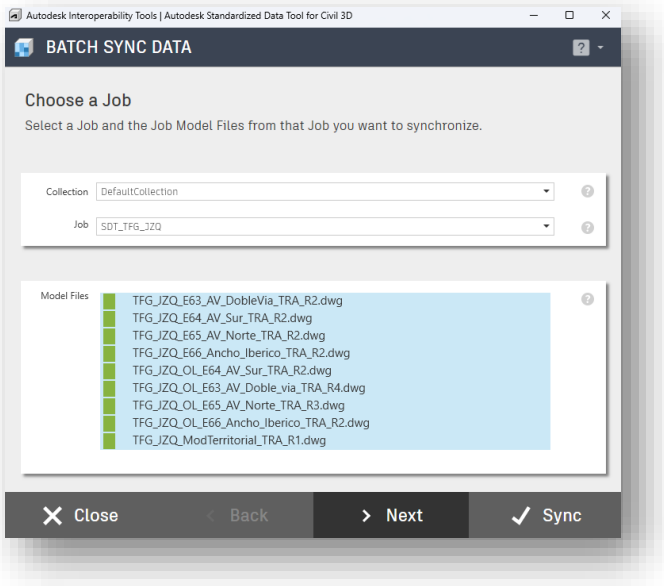
Asimismo, ahora se han de rellenar los valores de cada una de las propiedades, para ello la SDT ofrece funcionalidades de comunicación con una hoja Excel que será la que se generó por defecto al crear el trabajo, es decir lo que denominan archivo de datos del trabajo (.xlsx) que se mencionó con anterioridad. Para ello, se hace uso de la funcionalidad de sincronización de datos por lotes *"Batch Sync Data"*.

Figura 132 – Funcionalidad de sincronización de datos por lotes (Fuente: Elaboración Propia)



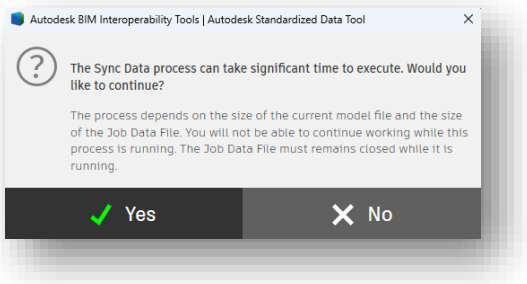
En esta ocasión al igual que en la actualización por lotes de estándares *"Batch Update Standards"* se sugiere continuar con la modalidad individual que será *"Sync Data"*. En el presente estudio los lotes daban fallos constantemente por problemas de desarrollo de la extensión por lo que se ha ejecutado individualmente modelo a modelo sin ningún problema.

Figura 133 – Formulario de sincronización de datos por lotes (Fuente: Elaboración Propia)



Como se aprecia en la figura una vez se acciona la funcionalidad de sincronización de datos por lotes lo primero que solicita al usuario es que defina el lote a sincronizar, es decir, seleccionar la cantidad de modelos dentro de los acotados en el trabajo que se desean añadir al lote. En este caso al continuar tras la selección lanza un error por lo que se continua con el mismo proceso, pero de forma individual para los datos de un único modelo.

Figura 134 – Aviso previo a la sincronización (Fuente: Elaboración Propia)



De vuelta, al accionar esta vez la funcionalidad de sincronización de datos *"Sync Data"* se muestra un aviso de la SDT. Este informa al usuario de que este proceso puede tomar una cantidad significante de tiempo, que dependerá directamente de la magnitud y cantidad de datos almacenados en el modelo con el que se está funcionando. Para continuar se presiona el botón inferior izquierdo *"Yes"*.

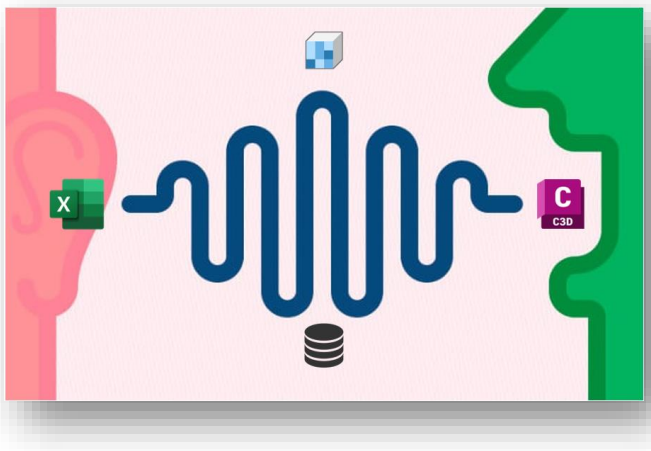
Aceptado el aviso, muestra el formulario inicial de sincronización de datos, este en primera instancia te permite

elegir entre dos opciones a partir de unos botones:

- Sincronizar los datos del archivo de datos (.xlsx) de Excel al modelo de Civil3D (.dwg)
- Sincronizar los datos del modelo de Civil3D (.dwg) al archivo de datos (.xlsx) de Excel

En términos de los elementos de comunicación, se ha de decidir quién será el **Emisor** y quien el **Receptor**, *Civil3D* o *Excel*, del **Mensaje** que son los datos contenidos en cada una de las propiedades que conforman los conjuntos generados, a partir de un **Canal** que será la Standarized Data Tool.

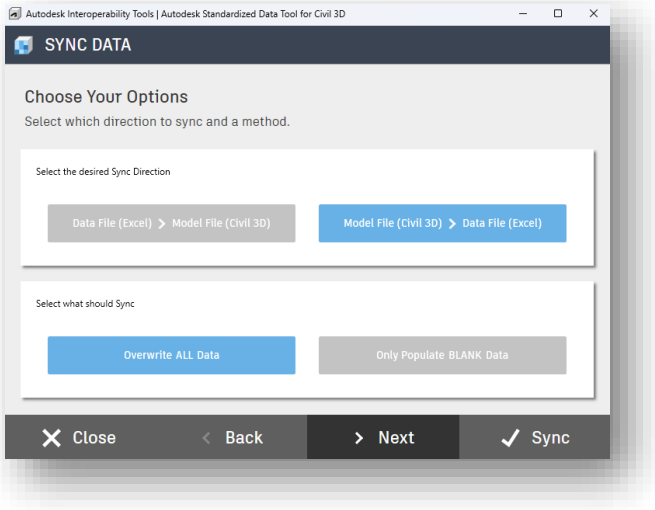
Figura 135 – Elementos de comunicación entre el modelo y el fichero de datos (Fuente: Elaboración Propia)



En este caso, se dispone de la gran mayoría de propiedades que pertenecen a los conjuntos vacías y se pretende completar la información desde el archivo de datos (.xlsx). Por lo que se selecciona la opción de la derecha que envía los datos del modelo al fichero *Excel*.

Además, en la parte inferior también se exige al usuario la elección entre dos opciones, en este caso se demanda que tome una decisión sobre el contenido que se ha de enviar al archivo de datos (.xlsx) si se ha de enviar todo sin restricción (izquierda) o si únicamente se desea enviar los datos vacíos (derecha). Se deduce que la última esta concebida para el flujo de trabajo de cumplimentado de la información.

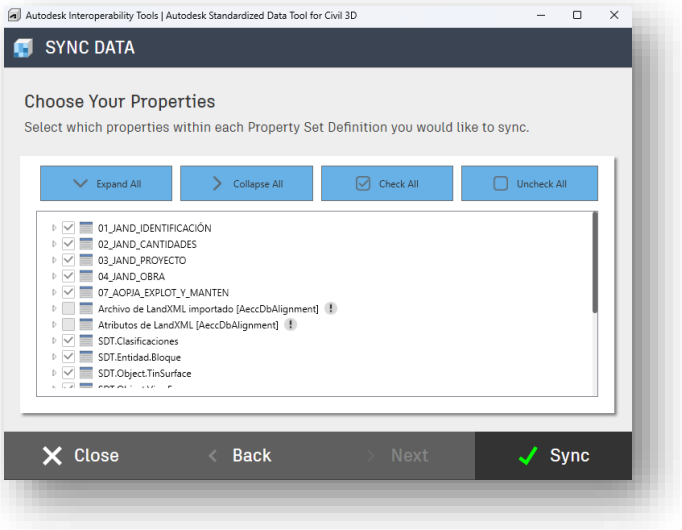
Figura 136 – Formulario de sincronización de datos I (Fuente: Elaboración Propia)



Se ha de presionar el botón de continuar *"Next"* para seguir configurando la sincronización de datos. Una vez se avanza al siguiente formulario aparece una lista con todos los conjuntos de propiedades presentes en el modelo, se pide al usuario que cumplimente las casillas de los conjuntos de propiedades que desea incluir en el proceso de sincronización. No obstante, destaca que varios conjuntos sin seleccionar presentan un icono de aviso

a su derecha. Esto es consecuencia de que en dichos conjuntos se encuentran propiedades configurada únicamente de lectura impidiendo al usuario modificarlas. Por lo tanto, no se permite la sincronización.

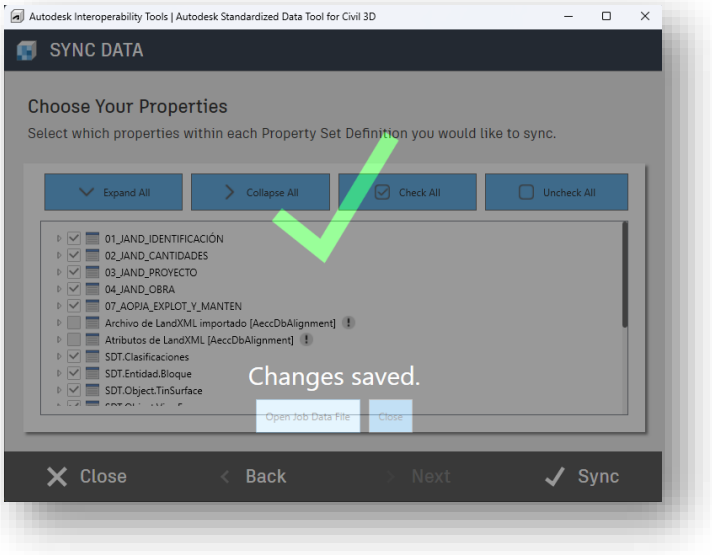
Figura 137 – Formulario de sincronización de datos II (Fuente: Elaboración Propia)



Una vez se han seleccionado todos los conjuntos de propiedades que se desean sincronizar se presiona el botón en la esquina inferior derecha de sincronizar *"Sync"*, seguramente este proceso consumirá una cantidad de tiempo, aunque depende de la cantidad de datos y el modelo.

Al terminar la sincronización mostrará por pantalla el símbolo de que todo está correcto y se ofrecen dos botones, abrir el archivo de datos de Excel (izquierda) o cerrar el formulario y continuar trabajando (derecha).

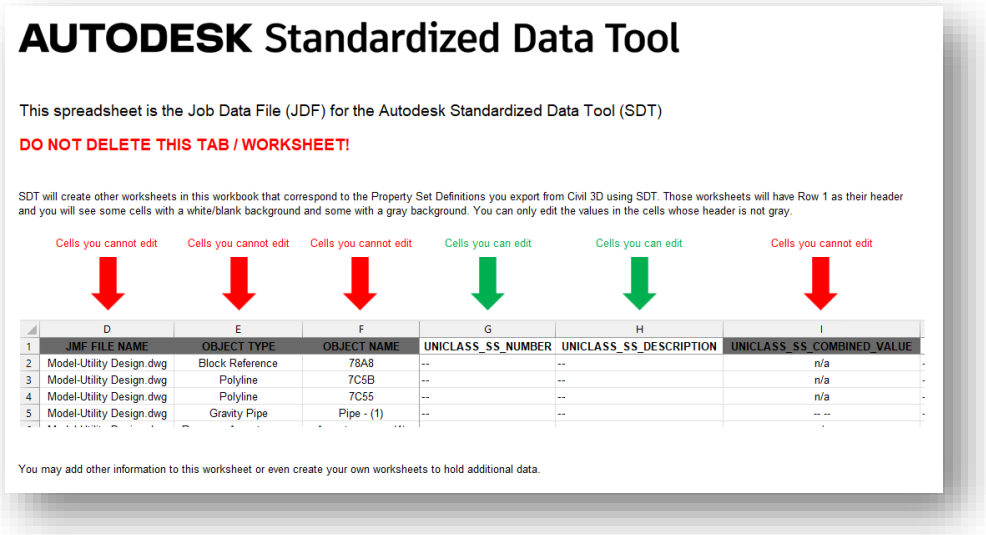
Figura 138 – Proceso de sincronización finalizado (Fuente: Elaboración Propia)



Una vez finalizado se continúa accediendo al fichero de datos del trabajo (.xlsx) que ha recibido toda la información en consecuencia de la sincronización efectuada.

Cabe recordar que este archivo de datos del trabajo (.xlsx) de Excel es siempre el mismo, y será con el que se va a trabajar. Recalcando que fue creado en la generación el trabajo en nuestro CDE (ver Figura 117 – Estructura de Archivos generados Standarized Data Tool (Fuente: Elaboración Propia)).

Figura 139 – Hoja inicial del libro de Excel sincronizado (Fuente: Elaboración Propia)



AUTODESK Standardized Data Tool

This spreadsheet is the Job Data File (JDF) for the Autodesk Standardized Data Tool (SDT)

DO NOT DELETE THIS TAB / WORKSHEET!

SDT will create other worksheets in this workbook that correspond to the Property Set Definitions you export from Civil 3D using SDT. Those worksheets will have Row 1 as their header and you will see some cells with a white/blank background and some with a gray background. You can only edit the values in the cells whose header is not gray.

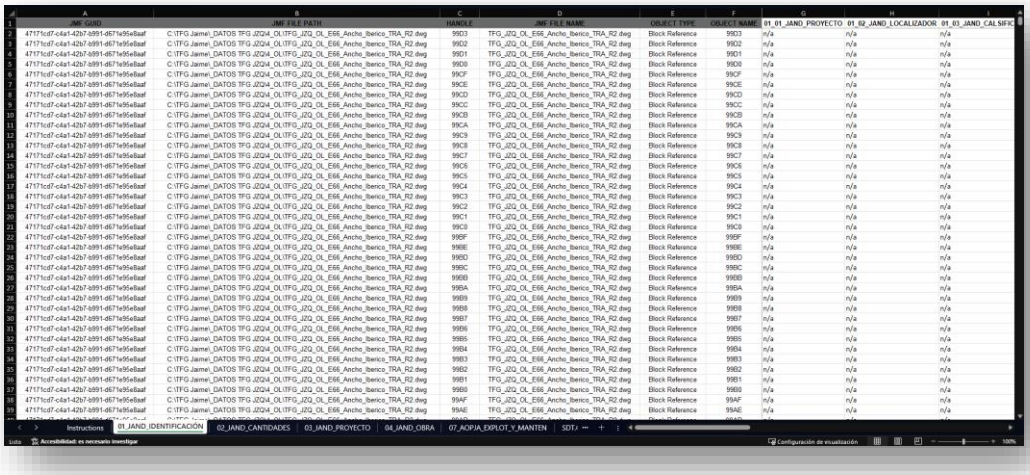
Cells you cannot edit | Cells you cannot edit | Cells you cannot edit | Cells you can edit | Cells you can edit | Cells you cannot edit

	D	E	F	G	H	I
1	JMF FILE NAME	OBJECT TYPE	OBJECT NAME	UNICLASS_SS_NUMBER	UNICLASS_SS_DESCRIPTION	UNICLASS_SS_COMBINED_VALUE
2	Model-Utility Design.dwg	Block Reference	78A8	--	--	n/a
3	Model-Utility Design.dwg	Polyline	7C5B	--	--	n/a
4	Model-Utility Design.dwg	Polyline	7C55	--	--	n/a
5	Model-Utility Design.dwg	Gravity Pipe	Pipe - (1)	--	--	-- --

You may add other information to this worksheet or even create your own worksheets to hold additional data.

Una vez se accede al fichero de datos (.xlsx) se puede apreciar en la figura lo que se dispone en la primera hoja del libro Excel. La SDT genera unas instrucciones dentro del libro de cálculo para que se tomen en cuenta las consideraciones propuestas para la de edición del fichero. La SDT creará hojas en el libro de Excel que corresponden a cada una de las definiciones de conjuntos de propiedades que han sido exportadas desde Civil3D utilizando la SDT. Esas hojas de trabajo dispondrán de la primera file como encabezado y se podrá comprobar como algunas celdas del encabezado se encuentran con fondo blanco y otras con fondo gris. Es importante tener en cuenta que únicamente pueden editarse los valores en las celdas cuya cabecera no sea gris.

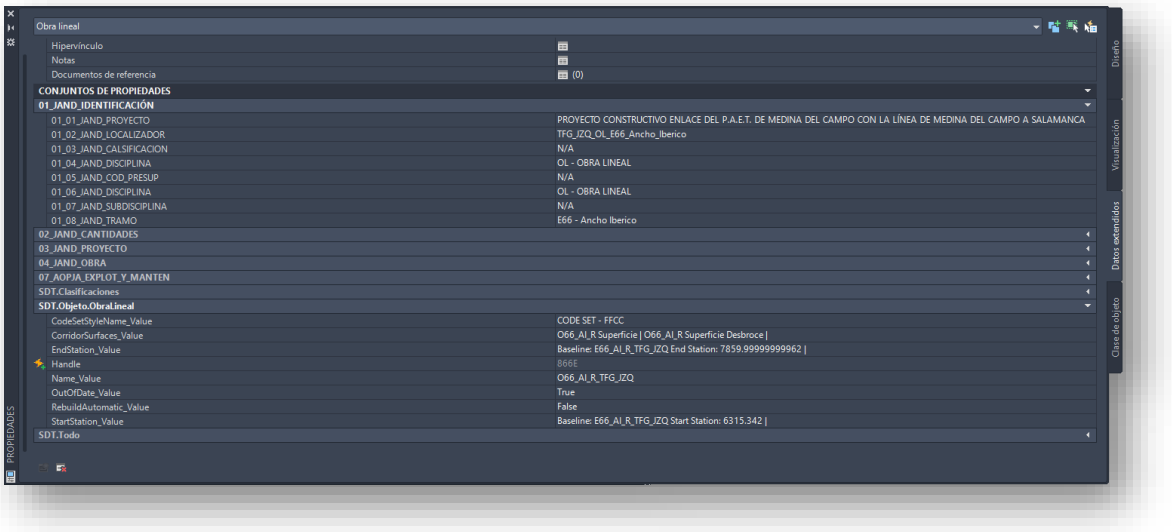
Figura 140 – Fichero Excel obtenido tras la sincronización (Fuente: Elaboración Propia)



Por lo tanto, se procede al cumplimentado, a modo de muestra, de la información del conjunto de propiedades *01_JAND_IDENTIFICACIÓN* que se conozca mediante el libro de cálculo (.xlsx).

Una vez ha sido cumplimentada la información se realiza el mismo proceso de sincronización de datos en camino inverso, es decir, esta vez el Emisor del mensaje (datos) será el archivo de datos del trabajo de Excel y el Receptor será el modelo de Civil3D. (ver Figura 135 – ***Elementos de comunicación entre el modelo y el fichero de datos (Fuente: Elaboración Propia)***)

Figura 141 – Propiedades actualizadas del Objeto de Obra Lineal del modelo (Fuente: Elaboración Propia)



Como puede apreciarse en la figura una vez ha sido sincronizado en el sentido inverso las propiedades se cumplimentan en los objetos del modelo. En este caso, como se ha concretado, únicamente se ha cumplimentado a modo expositivo el primer conjunto de propiedades *01_JAND_IDENTIFICACIÓN.* Siguiendo esta dinámica para cada uno de los modelos es posible enriquecer mediante la SDT todos los objetos de los modelos con las propiedades que se contemplan.

6.2.5 Exportación a formato OpenBIM

Una vez finalizado el proceso de enriquecimiento, limpieza y reestructuración de los modelos, se procede a la exportación de cada uno de ellos a un formato abierto, asegurando. El objetivo primordial es transformar los modelos con los que se ha estado trabajando, concebidos en la herramienta de software de diseño BIM para infraestructuras, Civil3D (formato .dwg), a un formato abierto y estandarizado IFC en su versión más actual, IFC4.3, que es el foco central de este trabajo de fin de grado.

Es relevante destacar que la extensión desarrollada por Autodesk para la exportación en IFC4.3, conocida como *"IFC Infrastructure"*, ya ha sido abordada en detalle en el capítulo 5, por lo cual no se profundizará en su funcionamiento y dinámica en esta sección. Sin embargo, se procederá con la exportación en serie de todos los modelos que se han tratado en el caso de estudio, siguiendo una metodología sistemática que permita un análisis riguroso del funcionamiento de la exportación. Los pasos de exportación fueron propuestos en su capítulo correspondiente (ver Figura 60 - ***Diagrama de Exportación IFC4.3 en Civil 3D (Fuente: Elaboración Propia)***), para este análisis se detallan a continuación:

1. **Exportación Genérica:**

   Se realiza una exportación inicial del modelo desde Civil3D sin configuraciones específicas, con el propósito de evaluar cómo la extensión interpreta y procesa los datos mediante sus capacidades de reconocimiento *REGEX* [40]. Este paso permite observar cómo se genera un modelo IFC con una asignación automatizada de clases y entidades, proporcionando una visión inicial de la eficacia y precisión de la herramienta de exportación.

2. **Exportación Personalizada:**

   Tras analizar los resultados de la exportación genérica, se examina cómo la extensión ha interpretado las asignaciones de cada elemento con su correspondiente entidad IFC. En base a estos resultados, se identifican las áreas que requieren ajustes y se procede a realizar cambios específicos en la configuración de la exportación, mapeado de las entidades y exportación de los conjuntos de propiedades. Esta fase personalizada busca optimizar la precisión del modelo abierto resultado de la exportación, asegurando que todas las entidades y clases se asignen correctamente conforme a los nuevos estándares de la última actualización de IFC para el dominio de las infraestructuras.

Una vez obtenidos los modelos abiertos resultantes de la exportación personalizada, se lleva a cabo un estudio detallado de cada uno para verificar su exactitud y conformidad con los requisitos esperados. Ese será el resultado con mayor exactitud que podrá ofrecer la herramienta, si se determina que el modelo no es correcto o completo, se debe continuar el proceso de refinado mediante alguna otra vía. En tales casos, se recurrirá a la implementación de una serie de códigos de programación específicos, desarrollados por el técnico BIM, que permitirán modificar el modelo en su formato abierto directamente y realizar las correcciones necesarias. Al modelado y edición de información con las herramientas de desarrollo que serán propuestas se le conoce como Modelado en nativo IFC.

Asimismo, este capítulo se limitará a presentar los códigos utilizados y los resultados obtenidos. Para un entendimiento más profundo de la programación aplicada, así como la metodología empleada en el desarrollo de estos códigos, se recomienda consultar el capítulo siguiente, donde se aborda exhaustivamente la lógica de programación y las propuestas técnicas planteadas.

Cabe destacar que, a la hora de estudiar el proceso de exportación, como caso de estudio únicamente se entrará en detalle de todas las iteraciones de un modelo de la disciplina de Obra Lineal, siendo este, el que más valor didáctico ofrece, debido a las implementaciones del dominio de ferrocarril *"IfcRailway"* en esta nueva actualización y su aplicación directa en el modelo de Obra Lineal. El resto de los modelos seguirán la misma dinámica y se presentarán únicamente los resultados finales, además de todo el control documental de las versiones de cada uno de los procesos iterativos.

ESTRUCTURA DE TRABAJO

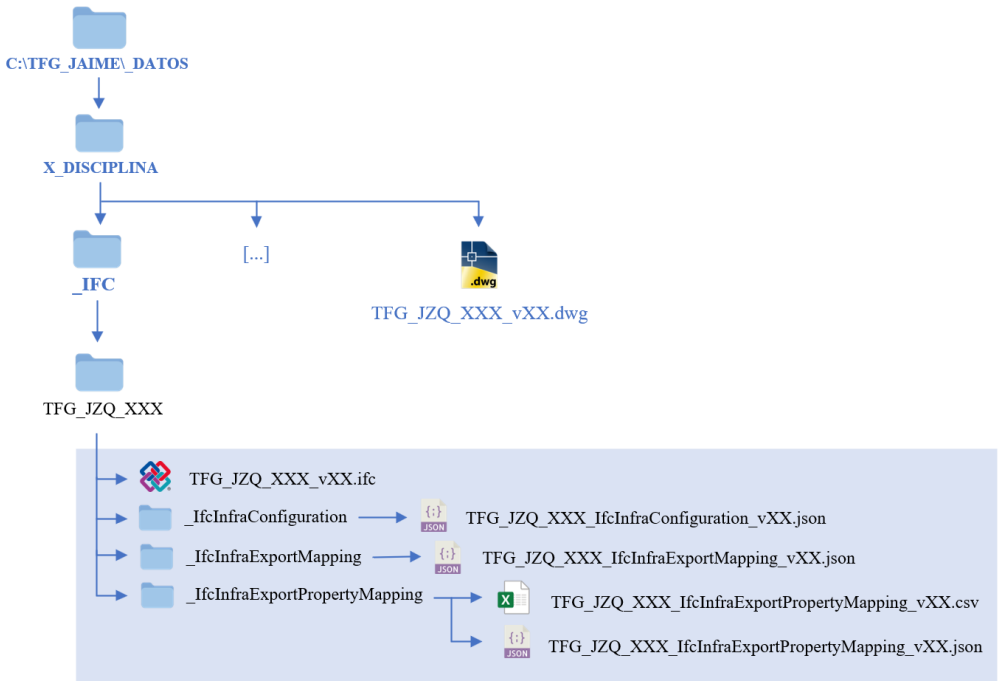
El proceso de exportación a formato OpenBIM no solo implica la conversión técnica de los datos, sino que también representa un paso crucial en la validación y garantía de calidad de los modelos BIM. Durante esta etapa, es fundamental realizar revisiones exhaustivas para asegurar que la información contenida en los modelos exportados mantenga su integridad y utilidad en un entorno colaborativo.

Es por ello por lo que la exportación debe ser vista como un proceso **iterativo**, donde la retroalimentación obtenida a partir de cada exportación personalizada se utiliza para mejorar continuamente la precisión de las exportaciones subsiguientes. Este enfoque no solo asegura la producción de modelos de calidad, sino que también facilita la identificación de problemas recurrentes que podrían requerir ajustes más profundos en las configuraciones de exportación o en la metodología de modelado utilizada en Civil3D.

Producto de esa iteración, surge la necesidad de generar diversidad de versiones de cada uno de los modelos. Para ello se debe definir una estructura de datos clara y ordenada que facilite el control documental y avance del proyecto. Se concibe así la carpeta *_IFC* dentro de cada una de las disciplinas pertinentes, dentro de esta carpeta se definen subcarpetas, una por cada modelo de Civil3D que se exporta de la disciplina, en las que se integrarán cada una de las versiones del modelo IFC generado, una por cada iteración. Además, dentro de ese nivel de carpetas se generará otra carpeta que almacenará cada una de las versiones de las plantillas JSON usadas que requiere el proceso de exportación personalizada. Ejerciendo así un control exhaustivo de los modelos generados para cada configuración específica de las plantillas pertinentes.

La versión de las plantillas que se usan para la generación de un modelo IFC a partir del proceso de exportación deben de tener la misma versión que el modelo resultado de la aplicación de estas. De lo contrario se pierde la asignación entre el modelo IFC resultado y sus plantillas JSON aplicadas, y en consecuencia el fin del control documental.

Figura 142 – Estructura de Datos propuesta para las plantillas de exportación IFC (Fuente: Elaboración Propia)



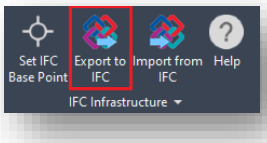
Este proceso es análogo al funcionamiento de las **Model View Definitions** (MVD) en el estándar IFC, donde cada exportación genera una "vista" o versión de la modelo adaptada a un conjunto de requisitos específicos. Al implementar esta metodología, se facilita no solo la personalización de las exportaciones, sino también el control de calidad y la validación continua de los modelos en base a los criterios definidos en las plantillas. Con un enfoque iterativo y estructurado, es posible optimizar la calidad de los modelos exportados, asegurar la coherencia de la información a lo largo del ciclo de vida del proyecto, y mantener un control exhaustivo de todas las versiones generadas.

Una vez los modelos estén exportados a formato abierto, las herramientas de software que se van a usar para su revisión, tanto de contenido gráfico como de datos e información, serán el visor gratuito **BIMvision** [41] y el software de modelado libre uso **Blender** [30] ,con su extensión desarrollada por la comunidad **BlenderBIM** para la apertura de modelos IFC.

EXPORTACIÓN GENÉRICA

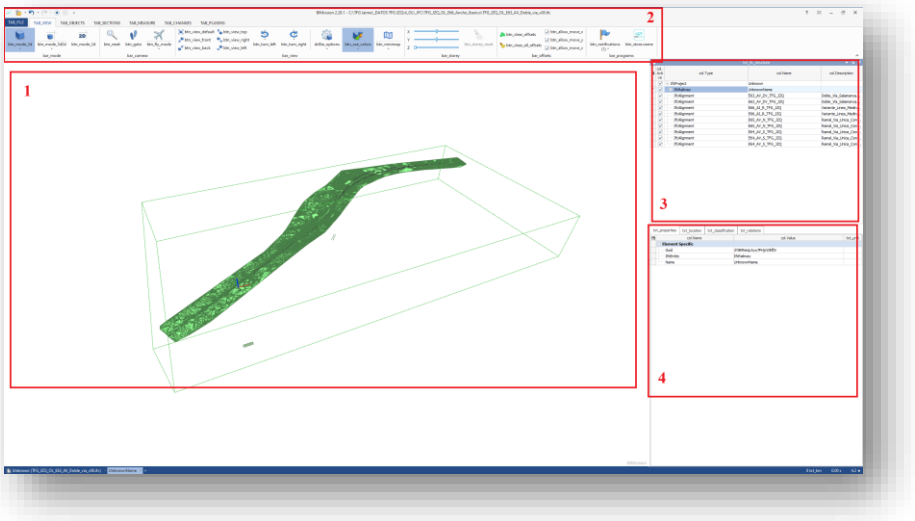
Para la exportación genérica no se necesita definir ni acotar nada en específico, es tan simple como ir a la funcionalidad de exportar a IFC que aparece en el panel de Complementos al instalar la extensión de IFC4.3 para Civil3D. Tal que, como se ha mencionado, el algoritmo desarrollado por autodesk para sus exportaciones hará todo el trabajo por el usuario.

Figura 143 – Funcionalidad de exportación IFC de la extensión IFC Infrastructure de Civil3D (Fuente: Elaboración Propia)



Como se mencionó anteriormente, una de las herramientas clave que se utilizará para la gestión de archivos en formato IFC es el visor gratuito BIMvision. Este visor de modelos abiertos permite acceder no solo a las representaciones gráficas, sino también a toda la información asociada a cada uno de los elementos del modelo. Además, incluye una serie de herramientas avanzadas que facilitan una exploración detallada del modelo, como planos de intersección gráfica, aislamiento de objetos, mediciones, entre otras funciones.

Figura 144 – Interfaz de Usuario del visor gratuito IFC BIMvision (Fuente: Elaboración Propia)



Dado que BIMvision se posiciona actualmente como uno de los mejores visores disponibles, especialmente en relación con la nueva actualización del estándar OpenBIM, es pertinente desglosar su interfaz de usuario para comprender mejor las funcionalidades que ofrece.

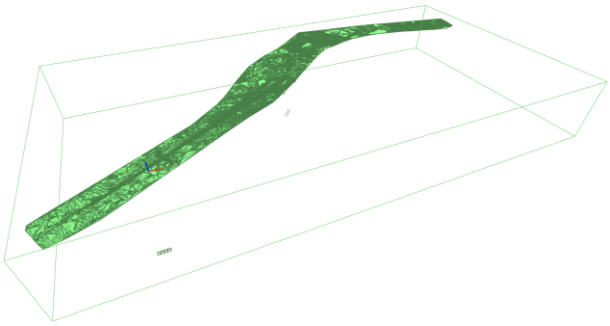
1. **Interfaz gráfica del modelo:** Esta sección permite explorar las representaciones tridimensionales de los elementos que componen el modelo. Es el componente central del software, y en adelante se mostrarán capturas de las representaciones obtenidas de los modelos. Cuando un elemento se resalta en color verde, indica que ha sido seleccionado, lo que facilita su identificación y análisis detallado.
2. **Barra de herramientas:** BIMvision ofrece una barra de herramientas repleta de funcionalidades que permiten un análisis exhaustivo del modelo. Se recomienda explorar los distintos paneles disponibles, cada uno de los cuales está dedicado a aspectos específicos del análisis, como la visualización, el

aislamiento de objetos, y la medición, entre otros.

3. **Árbol de elementos del modelo:** Esta característica es fundamental, ya que muestra la jerarquía y las relaciones entre los diferentes elementos del modelo, además de reflejar la estructura espacial del proyecto. Este árbol facilita la navegación y la comprensión de la organización del modelo, permitiendo a los usuarios localizar y analizar rápidamente los elementos de interés.
4. **Panel de información agregada:** Este panel muestra todos los datos asociados con los elementos seleccionados en el modelo, proporcionando detalles técnicos y especificaciones que son cruciales para el análisis y la toma de decisiones. Este panel es esencial para obtener una visión completa de las características y propiedades de cada componente del modelo.

Una vez que se conoce el visor, se procede con el estudio del primer modelo obtenido de la exportación genérica.

Figura 145 – Representación Gráfica del modelo resultado de la primera Exportación Genérica (Fuente: Elaboración Propia)



En primera instancia tras la revisión del modelo se deducen una serie de problemáticas:

- La herramienta exporta todos los objetos presentes del dibujo, a pesar de que algunos de ellos no estén originariamente definidos en este sino generados como acceso directo. Como lo son la superficie del terreno natural y las alineaciones de los distintos trazados.
- Además, exporta todas las polilíneas presentes en el modelo como una entidad de anotación *"IfcAnnotation"*. Esto son buenas noticias ya que demuestra las capacidades de intercambio de información, pero para este caso no es necesario la exportación de ninguna polilínea para el modelo.
- Las estructuras espaciales generadas son aceptables, aunque no óptimas. La herramienta a generado en primera instancia una estructura espacial basada en un proyecto *"IfcProject"* del que únicamente obtenemos un elemento de ferrocarril (IfcRailway).

  De la entidad de instalación de ferrocarril, como estructura espacial principal, se desglosa el terreno *"IfcGeographicElement"*, las polilíneas a modo de anotaciones del modelo *"IfcAnnotation"* y la entidad de Obra Lineal *"O63_AV_TFG_JZQ"* como parte de una instalación, es decir, el elemento parcial de una estructura espacial genérica *"IfcFacilityCommonPart"*. En su interior, se desglosan las distintas regiones de la Obra lineal, como la región *"RG – 6301"* entre otros. Estas regiones han sido definidas como partes de una instalación de ferrocarril *"IfcRailwayPart"* describiendo así una subestructura espacial que se desglosa del tramo que define este modelo.

  A su vez, cada una de estas regiones se desglosa en más partes (IfcRailwayPart), una por cada subensamblaje que compone el ensamblaje que define la región (denota la necesidad de estar mínimamente familiarizado con el concepto Obra Lineal como objeto de Civil3D y su definición para interpretar estos resultados).

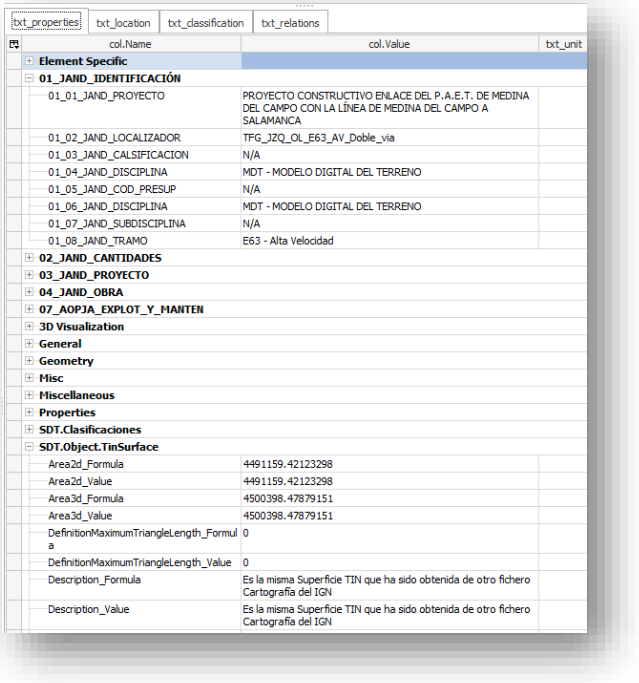
Cabe destacar que la problemática más crítica ha sido detectada en lo referente a la generación de los conjuntos de propiedades para cada uno de los elementos que conforman el modelo. Para las superficies, alineaciones, elementos de estructura espacial de la obra lineal, no ha habido ningún problema, en cambio para los elementos constructivos individuales que conforman la obra lineal no se han recibido propiedades. Esto ocurre en consecuencia de que estos elementos no son producto de un objeto acotado de Civil3D sobre el que se hayan asignado las propiedades con anterioridad, sino que es producto de la extrusión y generación tridimensional de los elementos que forman los subensamblajes que a su vez conforman cada uno de los ensamblajes que se

asignan a las regiones.

- Una vez se llega a la estructura espacial más básica que contiene los elementos constructivos *"IfcBuiltElement"*, se destaca que ninguno de los elementos esta caracterizado con una entidad concreta que haga referencia a su función, sino que son genéricos. Esto es una problemática importante y la principal causa por la que en esta nueva actualización del estándar se ha suprimido el famoso "*IfcBuildingElementProxy"*, ya que, en la medida de lo posible, no se deben dejar elementos caracterizados con entidades genéricas y mucho menos elementos que se relacionan con entidades que acaban de ser concebidas para su uso.

En el caso de que se deba de usar un elemento genérico constructivo, se denota la especial importancia de aplicar un tipo predefinido propio del usuario mediante *"USERDEFINED"*. Ese espacio, por tanto, estará disponible para acotar la entidad que el usuario encuentra que el esquema IFC actual no contempla.

Figura 146 – Conjunto de Propiedades de la superficie del terreno resultado de la exportación genérica (Fuente: Elaboración Propia)



Como conclusión de la revisión del modelo se propone tomar las siguientes medidas en el proceso de exportación:

- Acotar los elementos del modelo que se desean exportar, es decir, eliminar de la exportación del modelo el terreno, alineaciones y todas las polilíneas y bloques que se generan como *anotaciones*. La disposición de estos elementos no procede en el modelo que se está generando, sino que deben pertenecer al de sus correspondientes disciplinas.
- Planteamiento de una nueva estructura espacial acotada, eliminando contenedores espaciales redundantes y redefiniendo las entidades de los existentes. Además, es de especial importancia dotar del tipo predefinido específico que mejor se ajuste a cada uno de ellos.
- Acotar los conjuntos de propiedades que se generan, reduciendo la cantidad ya que el proceso genérico añade mucha información de forma automática. Únicamente serán exportados los conjuntos comentados en el apartado de enriquecimiento del modelo.

Figura 147 – Estructura espacial propuesta para los elementos de la Obra Lineal (Fuente: Elaboración Propia)

EXPORTACIÓN PERSONALIZADA

Se comienza por tanto con la personalización del proceso de exportación que ofrece la herramienta. Como ha sido mencionado la herramienta se configura y personaliza para cada caso específico mediante plantillas JSON (JavaScript Object Notation) que permiten la edición de variedad de campos.

Cada una de estas plantillas se generan de forma automática y difieren en función de los modelos (.dwg) para los que se crean, ya que el contenido de la plantilla depende directamente del contenido del dibujo de Civil3D.

Acorde con la estructura de trabajo definida, las plantillas son almacenadas de forma ordenada para discernir entre cada una de las casuísticas de exportación y sus resultados, no obstante, cabe destacar que para que el uso de la plantilla sea efectivo esta debe de situarse en el mismo directorio que el fichero .dwg que se desea exportar. Es decir, que si el modelo se encuentra en *"C:\TFG_JAIME_DATOS\4_OL\"*, en términos del CDE definido, las plantillas deben de situarse en esa misma dirección y con su nombre original. Como resultado, la forma de trabajar ha de ser:

1. Se generan las plantillas automatizadas desde la extensión de Civil3D.
2. Se guardan esas plantillas en su directorio definido y se editan.
3. Cuando queramos usar esa configuración de plantillas se COPIAN y se pegan en el mismo directorio del modelo con el nombre original del archivo, véase: *IfcInfraConfiguration, IfcInfraExportMapping, IfcInfraExportPropertyMapping*.

De esta forma no alteramos las plantillas ordenadas que permiten el control documental, y las genéricas que se encuentran en el directorio del modelo serán simplemente las de último uso y pueden ser eliminadas en cualquier momento.

Se recuerda que las plantillas deben de generarse a partir de la funcionalidad que se muestra en el desplegable del complemento instalado o mediante el comando de texto pertinente (ver figura XX).

Figura 148 – Generación de plantillas de exportación personalizada (Fuente: Elaboración Propia)

Tras la generación de los tres archivos pertinentes y su almacenado en el directorio que les corresponde se procede con su edición para una primera iteración de exportación personalizada.

En esta primera iteración se mostrarán todas las configuraciones propuestas para la exportación exitosa de la obra lineal a un modelo abierto definido acorde a las directrices señaladas. En primer lugar, la plantilla que será editada será la de mapeado de las entidades ifc, en esta plantilla es fundamental especificar en los argumentos pertinentes las entidades objetivo para cada uno de los elementos del modelo que civil3D ha detectado. Por lo tanto, acorde a las conclusiones previamente señaladas se establecen una serie de cambios correctivos para la mejora de la exportación.

Figura 149 – Fragmento de código JSON mapeado de entidades I (Fuente: Elaboración Propia)

```
"MapAlignmentStyleName": [
    {
      "Name": "_OCULTO",
      "IfcExportAs": "IfcAlignment",
      "Export": false
    },
```

En primer lugar, se especifica que todas las alineaciones que estén caracterizadas con el estilo *"_OCULTO"* no sean exportadas. Siguiendo esta dinámica, únicamente se debe actualizar el estilo de las alineaciones que no se deseen exportar al estilo mencionado.

Figura 150– Fragmento de código JSON mapeado de entidades II (Fuente: Elaboración Propia)

```
"MapCorridorStyleName": [
    {
      "Name": "Básico",
      "IfcExportAs": "IfcRailWay",
      "Export": true
    },
    {
      "Name": "Básico_ferrocarril",
      "IfcExportAs": "IfcRailWay",
      "Export": true
    },
    {
      "Name": "Estándar",
      "IfcExportAs": "IfcRailWay",
      "Export": true
    }
  ],
```

En el siguiente paso se ha especificado que cualquier Obra Lineal, de forma genérica, se exporte en una entidad ferroviaria *"IfcRailway"*. De esta forma, a pesar de que ya se exportaba correctamente se asegura la continuación de una exportación correcta.

Figura 151– Fragmento de código JSON mapeado de entidades IV (Fuente: Elaboración Propia)

```
"MapSurfaceStyleName": [
  {
    "Name": "_OCULTO",
    "IfcExportAs": "IfcGeographicElement.TERRAIN",
    "Export": false
  },
```

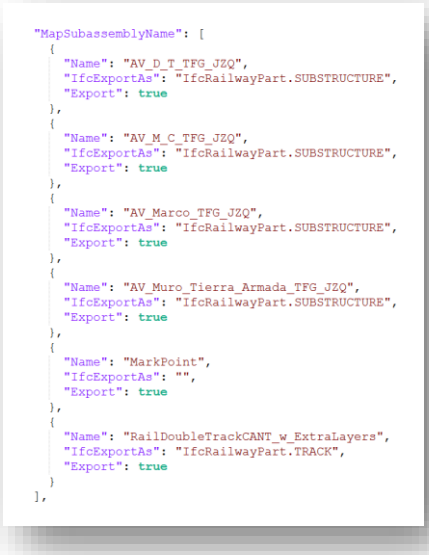
Siguiendo la misma dinámica que para los objetos de alineaciones, pero en este caso para las superficies se inhabilita la exportación para las superficies del modelo que estén en estilo "_OCULTO" de tal forma que se cambiará el estilo de la superficie del terreno natural.

Figura 152– Fragmento de código JSON mapeado de entidades III (Fuente: Elaboración Propia)

```
"MapBaselineRegionName": [
  {
    "Name": "RG\\ -\\ 6301",
    "IfcExportAs": "IfcFacilityPartCommon.SEGMENT",
    "Export": true
  },
  {
    "Name": "RG\\ -\\ 6302",
    "IfcExportAs": "IfcFacilityPartCommon.SEGMENT",
    "Export": true
  },
  {
    "Name": "RG\\ -\\ 6302\\ -\\ \\(1\\)",
    "IfcExportAs": "IfcFacilityPartCommon.SEGMENT",
    "Export": true
  },
  {
    "Name": "RG\\ -\\ 6302\\ -\\ \\(2\\)",
    "IfcExportAs": "IfcFacilityPartCommon.SEGMENT",
    "Export": true
  },
  {
    "Name": "RG\\ -\\ 6303",
    "IfcExportAs": "IfcFacilityPartCommon.SEGMENT",
    "Export": true
  },
  {
    "Name": "RG\\ -\\ 6304\\ ",
    "IfcExportAs": "IfcFacilityPartCommon.SEGMENT",
    "Export": true
  },
  {
    "Name": "RG\\ -\\ 6305L",
    "IfcExportAs": "IfcFacilityPartCommon.SEGMENT",
    "Export": true
  },
```


Conceptualmente referirse a las regiones de una obra lineal es lo mismo que referenciar a cada uno de los ensamblajes que se le asignan que, a su vez, están compuestos de subensamblajes. Estas regiones por lo tanto han sido definidas como un elemento de estructura espacial de carácter longitudinal que contendrá a cada uno de los subensamblajes, que serán definidos como una estructura espacial vertical. Siguiendo estas directrices se consigue estructurar el modelo espacialmente en tramos longitudinales, como han sido siempre estructuradas las obras lineales. Se considera por tanto un acierto el uso de la entidad *"IfcFacilityCommonPart"* que referencia a una parte común de estructura espaciales, en este caso acotada mediante el tipo predefinido de segmento *"SEGMENT"*, aludiendo a una porción lineal de una instalación, en este caso un *"IfcRailway"*.

Figura 153– Fragmento de código JSON mapeado de entidades V (Fuente: Elaboración Propia)



```
"MapSubassemblyName": [
  {
    "Name": "AV_D_T_TFG_JZQ",
    "IfcExportAs": "IfcRailwayPart.SUBSTRUCTURE",
    "Export": true
  },
  {
    "Name": "AV_M_C_TFG_JZQ",
    "IfcExportAs": "IfcRailwayPart.SUBSTRUCTURE",
    "Export": true
  },
  {
    "Name": "AV_Marco_TFG_JZQ",
    "IfcExportAs": "IfcRailwayPart.SUBSTRUCTURE",
    "Export": true
  },
  {
    "Name": "AV_Muro_Tierra_Armada_TFG_JZQ",
    "IfcExportAs": "IfcRailwayPart.SUBSTRUCTURE",
    "Export": true
  },
  {
    "Name": "MarkPoint",
    "IfcExportAs": "",
    "Export": true
  },
  {
    "Name": "RailDoubleTrackCANT_w_ExtraLayers",
    "IfcExportAs": "IfcRailwayPart.TRACK",
    "Export": true
  }
],
```

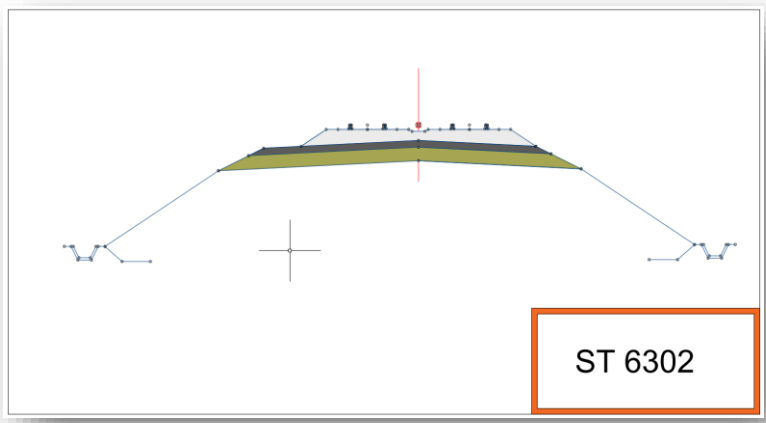

Cada uno de los subensamblajes es clasificado como una entidad de estructura espacial parcial de ferrocarril, es decir, como una parte de una instalación ferroviaria (IfcRailwayPart), además están caracterizados con tipos predefinidos que asignan al subensamblaje un carácter espacial, de tal forma que se ha clasificado:

- La estructura principal de la vía está definida por el subensamblaje propio de civil3D denominado *"RailwayDoubleTrackCANT_w_ExtraLayers"*, equivale al tipo predefinido de *"TRACK"* que se traduce como la estructura espacial que contiene todos los elementos que componen la vía.

- El resto de subensamblajes generan los desmontes, terraplenes, bermas, cunetas, muros, etc. de tal forma que se asigna a un tipo predefinido que engloba la subestructura de la obra lineal como *"SUBSTRUCTURE"*. Asimismo, lo ideal en este grupo es que se clasifique en una misma entidad espacial de subestructura y no por cada uno de los ensamblajes, por ello se les aplica a todos la misma entidad.

Una vez clasificadas las estructuras espaciales que engloban los elementos de los subensamblajes, la plantilla de mapeado permite acotar las entidades de cada uno de los elementos individuales de estos que serán los responsables de la generación del modelo tridimensional de la obra lineal y en consecuencia elementos individuales geométricos del modelo. Estos elementos, se estructuran en puntos geométricos, enlaces y formas.

Figura 154 – Ensamblaje 6302 del modelo en Civil3D (Fuente: Elaboración Propia)



Subensamblajes

Los **subensamblajes** son los componentes básicos que conforman un ensamblaje. Cada subensamblaje representa una parte específica del diseño, como un carril, una cuneta, una banqueta, un borde de calzada, etc. Dentro de cada subensamblaje, se difieren tres componentes clave:

- Los **puntos** en un subensamblaje son los vértices o puntos clave que definen la geometría del subensamblaje. Cada subensamblaje tiene un punto de inserción (donde se conecta con otros subensamblajes o con el eje del corredor) y otros puntos que definen su forma.

- Los **enlaces** o **vértices** son las líneas que conectan los puntos en un subensamblaje, definiendo así los contornos de la forma. Un enlace es una línea que conecta dos puntos, y estos enlaces definen las superficies y los bordes del subensamblaje.

- La **forma** en un subensamblaje se refiere a la geometría o al área que representa un componente del diseño en la sección transversal, está definida por los enlaces que forman un polígono cerrado generados en el subensamblaje. Por ejemplo, la forma de un carril de carretera sería el área rectangular o trapezoidal que ocupa el carril en el diseño.

Las plantillas de exportación permiten acotar las entidades de los tres componentes que forman el subensamblaje, mostrando los puntos, enlaces y formas de todos los subensamblajes presentes en el dibujo. En este caso, se deduce necesario la asignación de entidades IFC a las formas y enlaces, siendo estos elementos los encargados de mostrar la geometría final. Asimismo, cabe destacar que al ser una forma consecuencia de unos enlaces que forman una geometría cerrada, esos mismos enlaces no son de interés a la hora de la exportación porque generaría la superficie repetida de la forma.

Estos conceptos se esclarecerán a medida que revisemos los modelos resultados de estas reglas de personalización.

Figura 155 – Fragmento de código JSON mapeado de entidades VI (Fuente: Elaboración Propia)


```
"MapShapeCode": [
  {
    "Code": "Ballast",
    "IfcExportAs": "IfcCourse.BALLASTBED",
    "Export": true
  },
  {
    "Code": "Cuneta",
    "IfcExportAs": "IfcDistributionChamberElement.TRENCH",
    "Export": true
  },
  {
    "Code": "HLimpieza",
    "IfcExportAs": "IfcCourse.PROTECTION",
    "Export": true
  },
  {
    "Code": "Marco",
    "IfcExportAs": "IfcWall",
    "Export": true
  },
  {
    "Code": "Muro",
    "IfcExportAs": "IfcWall",
    "Export": true
  },
  {
    "Code": "Rail",
    "IfcExportAs": "IfcRail.RAIL",
    "Export": true
  },
  {
    "Code": "RellenoTrasdos",
    "IfcExportAs": "IfcEarthworksFill.BACKFILL",
    "Export": true
  },
  {
    "Code": "Subballast",
    "IfcExportAs": "IfcCourse.BALLASTBED",
    "Export": true
  },
  {
    "Code": "SubBase",
    "IfcExportAs": "IfcEarthworksFill.SUBGRADE",
    "Export": true
  }
],
```


Por lo tanto, se define cada una de las formas de los subensamblajes como sigue:

- La forma del balasto *"Ballast"* recibe el equivalente de entidad de capa *"IfcCourse"* con el tipo predefinido de cama de balasto *"BALLASTBED"*.

- La cuneta de la sección se ha aproximado mediante la entidad de elemento de sistemas de distribución *"IfcDistributionChamber"* dotado del tipo predefinido de trinchera o zanja *"TRENCH"*, siendo este el que más se asemeja al cumplimiento conceptual de una cuneta.

- El área que acota la forma del hormigón de limpieza se ha caracterizado como una entidad de capa *"IfcCourse"* que está acotada mediante el tipo predefinido de protección *"PROTECTION"*.

- La geometría que acota el marco consiste en un sistema estructural dotado de muros laterales que sostienen una losa superior. Al ser un elemento geométrico único no es posible clasificarlo como elemento de losa *"IfcSlab"* y muro *"IfcWall"* simultáneamente. Este podría ser un caso justificable como para usar un elemento constructivo genérico en el que se asigna un tipo predefinido personalizado

por el usuario *"USERDEFINED"*. No obstante, defendiendo la posición de que la definición de los elementos constructivos genéricos consiste en una mala práctica, será exportado con un muro que al estar contenido en la entidad espacial de subestructura ferroviaria se contextualiza el elemento de forma satisfactoria.

- La forma pertinente al muro lógicamente es equivalente a su entidad IFC comentada en para el caso anterior, *"IfcWall"*.
- Para los elementos de raíles se establece como entidad equivalente la de *"IfcRail"*, que entre todos los tipos predefinidos que el esquema ofrece procede especificar que es un raíl *"RAIL"*.
- El relleno generado para el trasdós del muro se deduce como una entidad de movimiento de tierras de rellenado, su equivalente en el esquema IFC será *"IfcEarthworksFill"* además se le dotará del tipo predefinido *"BACKFILL"* que engloba al relleno detrás de muros de contención u otras estructuras.
- Para la geometría de forma que genera la capa de sub-balasto se asigna de forma análoga una entidad de capa*"IfcCourse"* con un tipo predefinido de balasto, *"BALLASTBED"*.
- Por último, la subbase de la estructura que forma la obra lineal se corresponde con un relleno de suelo seleccionado que será definido como la entidad de relleno "IfcEarthworksFill" acotada mediante el tipo predefinido correspondiente a la primera capa base de una obra lineal *"SUBGRADE",* acorde a el elemento de movimiento de tierras que forma la estructura por debajo del pavimento y por encima del suelo natural que soportará las cargas de la estructura suprayacente.

Figura 156 - Fragmento de código JSON mapeado de entidades VII (Fuente: Elaboración Propia)


```
"MapLinkCode": [
  {
    "Code": "Ballast",
    "IfcExportAs": "",
    "Export": false
  },
  {
    "Code": "Ballast_Between_Ties",
    "IfcExportAs": "",
    "Export": false
  },
  {
    "Code": "Berma1",
    "IfcExportAs": "IfcEarthworksElement.BERM",
    "Export": true
  },
  {
    "Code": "Berma2",
    "IfcExportAs": "IfcEarthworksElement.BERM",
    "Export": true
  },
  {
    "Code": "Cuneta",
    "IfcExportAs": "",
    "Export": false
  },
```


Respecto a los elementos de enlace, que pertenecen a todos los subensamblajes que comprenden la Obra Lineal del modelo únicamente se ha establecido como parámetro de exportación:

- Las bermas de cada uno de los costados de la trazada, este elemento se caracteriza como un elemento de movimiento de tierra genérico para el que se ha tomado la decisión de aplicar un tipo predefinido por el usuario *"USERDEFINED"* concebido como el término anglosajón de berma *"BERM"*. De esta forma se explora la posibilidad de definir nuevos tipos predefinidos por el usuario en la plantilla
- Además, también se caracterizan los enlaces que generan las superficies de talud en desmonte como elemento de movimiento de tierras en desmonte *"IfcEarthworksCut"* acotado bajo el tipo predefinido

de excavación necesaria para construcción de instalaciones como una vía férrea en este caso *"CUT"*. Su antónimo, por el contrario, se identifica como la entidad de elemento de movimiento de tierras en relleno de material *"IfcEarthworksFill"* tomando el tipo predefinido de talud de terraplén *"SLOPEFILL"*.

Figura 157 - Fragmento de código JSON mapeado de entidades VIII (Fuente: Elaboración Propia)

```
  {
    "Code": "SubBase",
    "IfcExportAs": "",
    "Export": false
  },
  {
    "Code": "Talud-Desmonte",
    "IfcExportAs": "IfcEarthworksCut.CUT",
    "Export": true
  },
  {
    "Code": "Talud-Terraplen",
    "IfcExportAs": "IfcEarthworksFill.SLOPEFILL",
    "Export": true
  },
  {
    "Code": "Top",
    "IfcExportAs": "",
    "Export": false
  },
  {
    "Code": "Top_Rail",
    "IfcExportAs": "",
    "Export": false
  }
],
```


Una vez definidas todas las intervenciones en la plantilla JSON de mapeado de objetos de Civil3D a entidades IFC en *"IfcInfraExportMapping.JSON"*, se acota la plantilla *"IfcInfraConfiguration.JSON"* configuraciones más generales de la exportación.

Figura 158 – Fragmento de código JSON configuración de exportación I (Fuente: Elaboración Propia)

```
"GenerateLogFile": true,
"Export": {
  "DefaultIfcZip": false,
  "DefaultOutputFolder": "",
  "FacetDistanceToleranceMetric": 0.002,
  "FacetDistanceToleranceImperial": 0.006,
  "AbortIfOutOfDate": false,
  "AutomaticSaveDocument": false,
  "ExportPropertiesFromCivilEntityProperties": true,
  "ExportMaterials": true,
  "ExportAlignments": true,
  "ExportCorridors": true,
  "ExportCorridorFeatureLines": true,
  "ExportCorridorLinks": true,
  "ExportBridges": true,
  "ExportCogoPoints": true,
  "ExportFeatureLines": false,
  "ExportSurfaces": false,
  "ExportBlockReferences": true,
  "ExportSolids": true,
  "ExportBodies": true,
  "ExportPoints": true,
  "ExportPolylines": false,
  "ExportPolyFaceMesh": true,
```


Únicamente se muestran fragmentos que han sido modificados respecto a la plantilla generada por defecto por la extensión. En primer lugar, se destaca la opción general de exportación de objetos de Civil3D, en este caso se han desactivado las superficies y polilíneas. El resto de las opciones desactivadas son así por defecto.

Cabe destacar que a pesar de haber configurado anteriormente la inhabilitación de las alineaciones en estilo *"_OCULTO"* se ha dejado en esta configuración general activadas ya que si se desactivan no exporta correctamente la Obra Lineal, se deduce que el algoritmo de exportación necesita acceder a las alineaciones para generar los sólidos de la Obra Lineal y si se desactivan obtenemos un modelo sin Obra Lineal.

Figura 159 – Fragmento de código JSON configuración de exportación I (Fuente: Elaboración Propia)

```
"AuthorAttributes": {
  "Identification": "JZQ",
  "FamilyName": "Zaforteza Quintanilla",
  "GivenName": "Jaime",
  "OrganizationIdentification": "ETSI-US",
  "OrganizationName": "Escuela Técnica Superior de Ingenieria - Universidad de Sevilla",
  "OrganizationEmail": "",
  "Authorization": ""
},
```

Además, se ha configurado los datos de atribución al autor del modelo. Es fundamental firmar los modelos digitales generados, al igual que se exige la firma de la planimetría producida para un proyecto los modelos BIM tanto abiertos como cerrados han de tener una identificación de autor y organización.

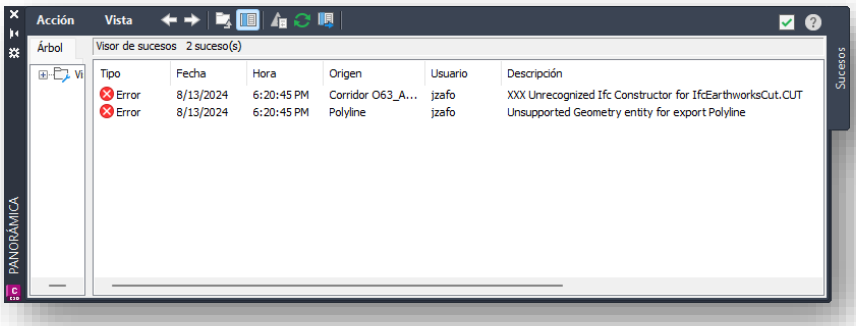
Figura 160– Fragmento de código JSON configuración de exportación I (Fuente: Elaboración Propia)

```
"PropertyTemplatePaths": [],
"PropertyManagementPaths": [ "C:/TFG Jaime/_DATOS TFG JZQ/4_OL/IfcInfraExportPropertyMapping.csv" ],
"PropertyListDelimiters": ";"
```

Por último, se concreta, a pesar de que no es necesario al ser el directorio donde deben situarse las plantillas por defecto, el directorio donde se encuentra la plantilla de gestión de propiedades. Esta plantilla consiste en un archivo csv que ha sido editado dejando únicamente activas los conjuntos de propiedades que se han generado en el presenta caso de estudio.

Tras estos últimos detalles se procede con la exportación, a pesar de que esta vez es personalizada, una vez están las plantillas en el mismo directorio del fichero (.dwg) que se está exportando no es necesario establecer ningún parámetro adicional desde Civil3D. Únicamente se ha de accionar la funcionalidad de exportación y seleccionar el directorio donde se desea almacenar el modelo abierto. En este caso se recalca que acorde a la estructura de trabajo establecida su directorio será: *"C:\TFG_JAIME_DATOS\4_OL_IFC\..."* en su carpeta de modelo pertinente, para así realizar un correcto control documental.

Figura 161 – Error de exportación entidad IFC no reconocida (Fuente: Elaboración Propia)

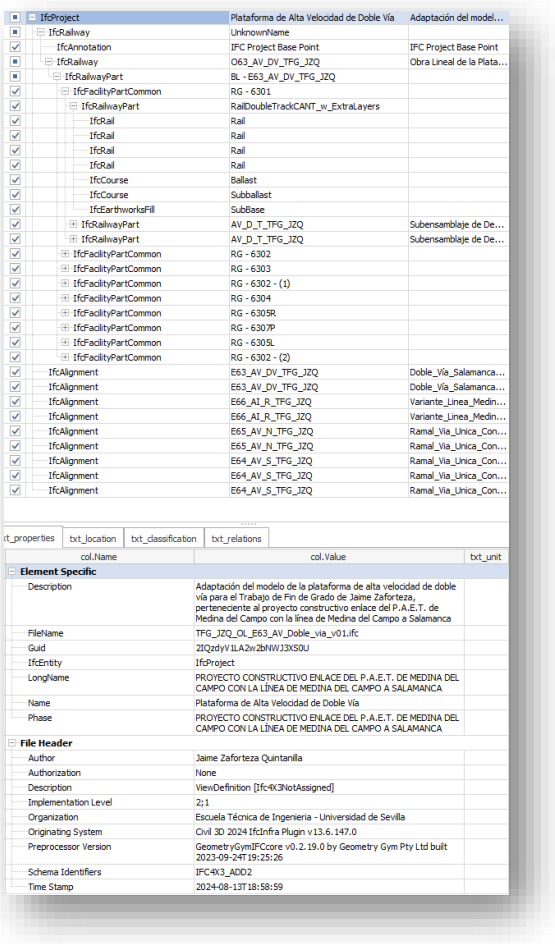


La primera sorpresa en el proceso de exportación resulta en la falta de identificación por parte del complemento de exportación de la entidad IFC *"IfcEarthworksCut"*, sorprendentemente no está registrada en la extensión. Esta entidad fue añadida en la nueva actualización del estándar IFC4.3 (ver Figura 27 - ***Novedades introducidas en la actualización IFC4.3 (Fuente: BuildingSmart International)***) y puede consultarse en la documentación oficial actual del estándar IFC [25].

Asimismo, se elimina de la plantilla de mapeado de entidades y se deja vacío el campo para el talud en desmonte, será por tanto exportado de forma genérica *"IfcBuiltElement"* pero será solucionado más adelante.

Una vez gestionado el primer error de la exportación se ha conseguido con éxito la obtención del modelo abierto. Tras su obtención, se procede a un análisis técnico riguroso de los resultados obtenidos mediante el visor gratuito BIMvision.

Figura 162 – Información del proyecto y desglose jerárquico del modelo IFC desde BIMvision (Fuente: Elaboración Propia)



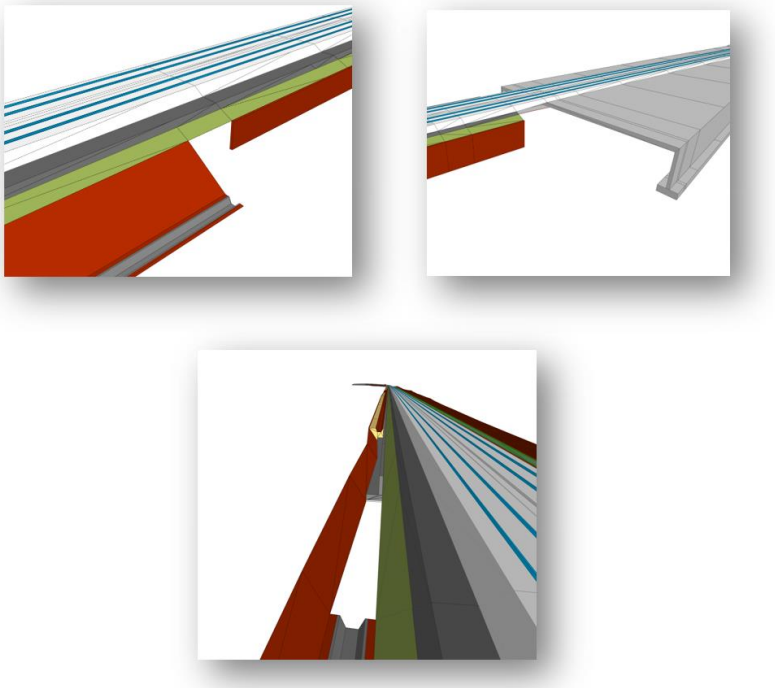
El modelo obtenido parece correcto en rasgos generales, la estructura jerárquica es correcta a pesar de que existe una estructura espacial *"IfcRailway"* identificada con nombre desconocido *"Unknown Name"* que se desconoce de donde proviene, pero deberá ser eliminada ya que no aporta información sobre el proyecto. No obstante, la entidad de misma naturaleza que le sigue identificada como *"O63_AD_DV_TFG_JZQ"* es la obra lineal del modelo y corresponde con el modelo de Civil3D. No obstante, en continuación con el desglose espacial que muestra el explorador del visor, se aprecia la línea base de la obra lineal *"BaseLine (BL)"* interpretada como un contenedor espacial "IfcRailayPart", este contenedor es redundante y carece de sentido espacial ya que contiene la instalación propuesta en su totalidad, es decir, no aporta valor de división espacial frente a su super-tipo jerárquico.

Asimismo, al continuar descendiendo por el desglose jerárquico se llega a las regiones de ensamblajes como*"RG-6001"* entre otros que correctamente mapeados con entidad *"IfcFacilityPartCommon.SEGMENT"* representa con éxito la división longitudinal de la obra lineal. Además, si se desglosa esta última mencionada, se aprecia cada uno de los subensamblajes que forman la región, caracterizados correctamente como *"IfcRailwayPart"* representando así la división espacial vertical de cada una de las regiones, y por último los

elementos constructivos que la componen. No obstante, cabe destacar que los subensamblajes de desmonte y terraplén que se pueden apreciar repetidos, se propone unir ambos en una única entidad espacial que represente la subestructura de la vía férrea.

Además, también se aprecia que las alineaciones están presentes en el modelo, así como algunas anotaciones *"IfcAnnotation"* y elementos constructivos genéricos *"IfcBuiltElement"* redundantes. Estos elementos deberán ser eliminados del modelo, será abordado en siguientes modificaciones ya desde el modelo nativo en IFC mediante el software Blender.

Figura 163 – Inconsistencias de generación geométrica del modelo abierto (Fuente: Elaboración Propia)



En lo respectivo a las representaciones geométricas se han encontrado varias incongruencias en los distintos tramos de la obra lineal. Existen varios tramos en los que los subensamblajes diseñados no ejecutan correctamente las formas y enlaces que tras su extrusión a la hora de generar el modelo de obra lineal pierden información y dejan vacíos gráficos.

Se propone la solución de las incongruencias gráficas del modelo IFC a partir de la activación de los objetos de líneas características de la obra lineal en plantilla de configuraciones de exportación. De tal forma que en el modelo se podrán apreciar los trazados que no se han materializado en objetos tridimensionales y podrá trazarse un modelado como solución desde Blender con su extensión que permite el trato de ficheros IFC BlenderBIM.

En cuanto a los fallos encontrados en las representaciones geométricas de los subensamblajes, cabe señalar que estos podrían deberse a dos posibles causas. Por un lado, es posible que los códigos de subensamblaje no hayan sido correctamente codificados por el autor para establecer los vínculos necesarios en el modelo. Por otro lado, existe la posibilidad de que el exportador simplemente haya omitido estos elementos durante el proceso de exportación a IFC. En este sentido, no se puede determinar con certeza si la raíz del problema radica en un fallo en la codificación de los subensamblajes o en el comportamiento del exportador, por lo que esta cuestión queda abierta a futuras investigaciones y revisiones técnicas.

Figura 164 - Activación de la exportación de líneas características en la configuración JSON (Fuente: Elaboración Propia)

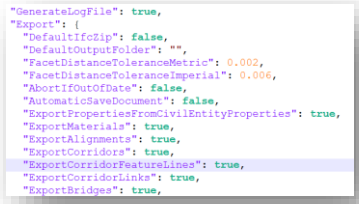
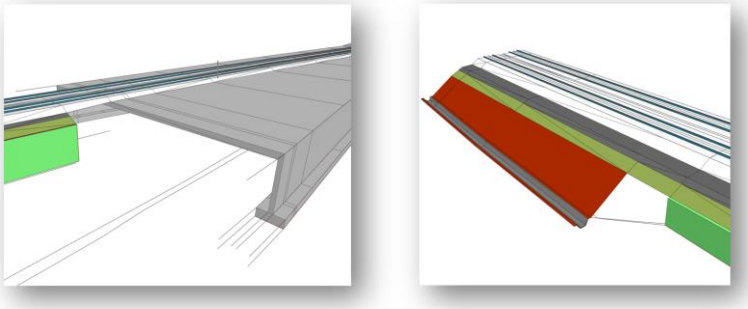


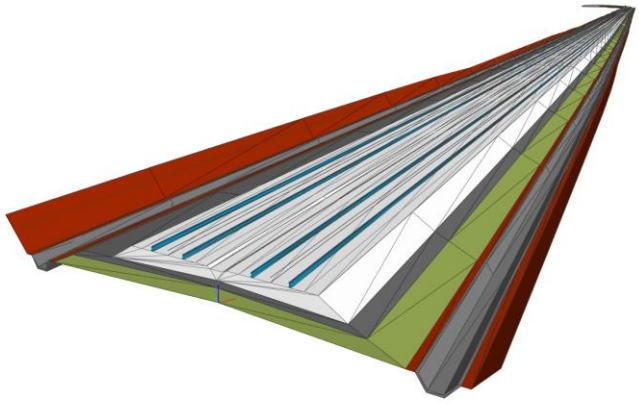
Figura 165– Modelo IFC con las líneas características de la Obra Lineal activadas (Fuente: Elaboración Propia)



Asimismo, la exportación en rasgos generales ha resultado exitosa, aunque autodesk no ofrezca suficientes herramientas para el refinado de las estructuras espaciales, filtrado de lo que únicamente desea mostrar el modelo, etc. Se ha conseguido un modelo con el que se permite seguir trabajando y es gracias a las herramientas presentadas a continuación de modelado en nativo IFC que se pueden asumir resultados mejorables.

Asimismo, al exportar el modelo con las líneas caracterísitcas, se observa que estas, que son las encargadas de la generación de la geometría a partir de los subensamblajes, aparecen en el modelo de referencia, lo que sugiere que la geometría base sí se ha generado correctamente. Este hecho aumenta las probabilidades de que el problema resida en el exportador desarrollado por Autodesk, que podría estar omitiendo la conversión completa de estos elementos a objetos tridimensionales en el formato IFC durante el proceso de exportación. Esto refuerza la hipótesis de que el exportador podría ser el responsable de las incongruencias gráficas observadas.

Figura 166 – Perspectiva de primera aproximación al modelo IFC (Fuente: Elaboración Propia)



EDICIÓN / MODELADO NATIVO IFC

Una vez se ha obtenido el modelo abierto IFC con el máximo refinado técnico que ofrece la herramienta de exportación *"IFC Infraestructure"* se continua con la búsqueda de una mayor edición y mejora de los detalles del modelo bajo software de modelado en IFC nativo.

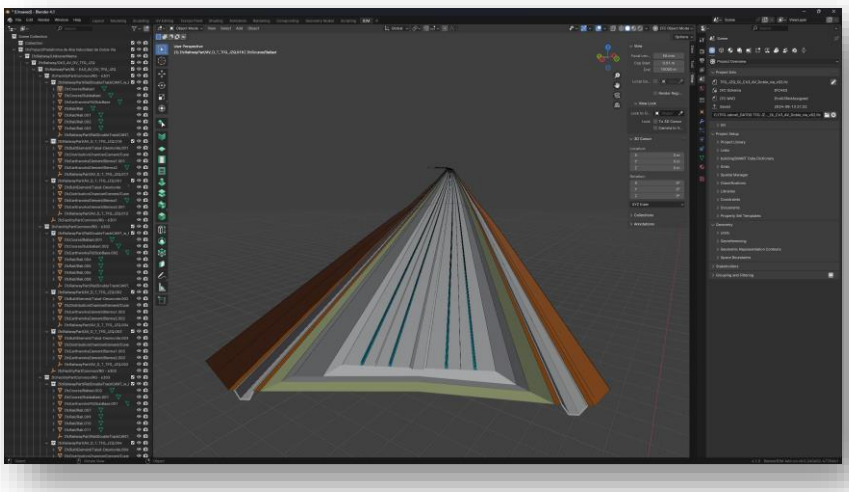
Cuando se hace referencia al modelado en IFC nativo, se alude a las herramientas que, al igual que civil3D y otros programas de diseño, permiten generar modelos BIM, en este caso generar modelos en IFC desde un inicio, permitiendo dejar de pensar en IFC como un formato de intercambio y comenzar a percibirlo como un activo. Modelar desde cero un modelo polivalente que acorde a normativa es común a todos. No obstante, las herramientas de modelado en formato abierto como nativo surgieron hace relativamente poco y no disponen del desarrollo de herramientas específicas con algoritmos de diseño como los que las grandes casas de software llevan años refinando.

Figura 167 – Logotipo de BlenderBIM (Fuente: Portal BlenderBIM)

En este contexto entra en escena Blender, en su mayor expresión en lo que se refiere al IFC cuando se combina con la extensión BlenderBIM. Consiste en una herramienta innovadora que ha sido desarrollada sobre la librería de pyhton *IfcOpenShell*. Se presenta como una extensión para Blender, un software de modelado 3D de código abierto, gratuito y extremadamente versátil. La principal característica distintiva de BlenderBIM es su capacidad para modelar directamente en el formato nativo IFC, lo que permite a los usuarios editar proyectos de forma directa en este formato utilizando una interfaz más intuitiva y familiar.

No será hasta el capítulo 7 que se entrará en detalle sobre la programación en Python, las librerías usadas, *IfcOpenShell* entre ellas y detalles de los desarrollos de códigos usados en el presente trabajo. Además, también se entrará en detalle sobre las posibilidades que ofrece Blender y su extensión BlenderBIM en lo referente a programación y cómo se ha abordado en este caso de estudio.

Figura 168 – Modelo exportado abierto desde Blender (Fuente: Elaboración Propia)

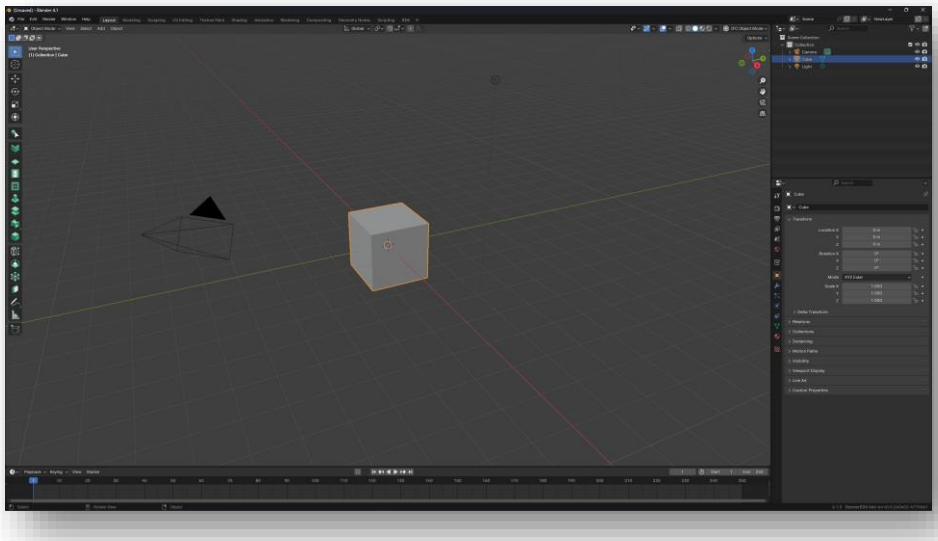


Algunas de las características y funcionalidades clave de BlenderBIM incluyen:

1. **Modelado en formato IFC**: BlenderBIM permite a los usuarios modelar y editar proyectos directamente en el formato IFC, lo que facilita la interoperabilidad con otros programas BIM y elimina la necesidad de convertir entre diferentes formatos de archivo.

2. **Soporte para scripts de Python**: BlenderBIM permite a los usuarios integrar scripts de Python dentro del software Blender, lo que amplía aún más las capacidades y funcionalidades disponibles para el modelado y la edición de proyectos.
3. **Software libre y gratuito**: BlenderBIM es un software de código abierto, lo que significa que es gratuito para descargar, utilizar y modificar. Esto lo hace accesible para una amplia gama de usuarios, incluidos estudiantes, profesionales y organizaciones.
4. **Interoperabilidad**: Al estar basado en la librería IfcOpenShell, BlenderBIM garantiza la interoperabilidad con otros programas y herramientas que admiten el estándar IFC, lo que facilita el intercambio de datos y la colaboración en proyectos BIM.

Figura 169 – Interfaz inicial de Blender (Fuente: Elaboración Propia)

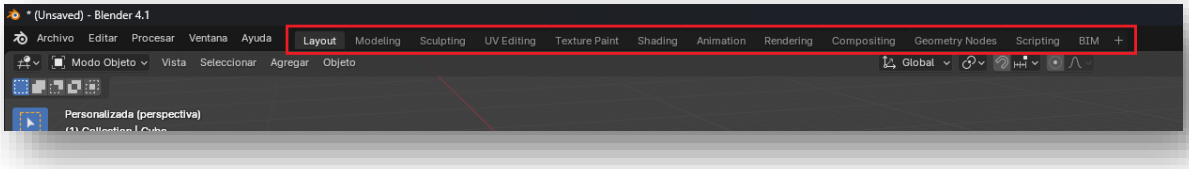


En conclusión, se decide que BlenderBIM es un proyecto de código abierto *(OpenSource)* que se alinea con los objetivos del IFC en la industria. Además, se considera la herramienta idónea para el presente caso de estudio cumpliendo las necesidades tanto de revisión como edición y modelado de los modelos abiertos.

Una vez se ha introducido BlenderBIM, se procede con un desglose básico, sin entrar en profundidad, de la interfaz de Blender y los cambios que introducen en esta la instalación del complemento BlenderBIM.

Como se ha apreciado en la figura anterior, al iniciar Blender, por defecto, siempre se inciará en un dibujo en blanco con un objeto tridimensional de un cubo en el espacio central de la ventana gráfica. Además, si se presta atención a la parte superior de la herramienta informática a la derecha de la barra de herramientas convencional se dispone de una serie de paneles que acotan los espacios de trabajo. Los espacios de trabajo es un concepto de Blender que define la interfaz de la herramienta dependiendo de con qué finalidad usamos el software.

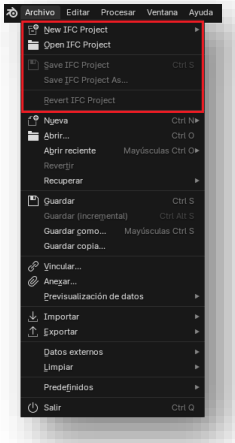
Figura 170 – Espacios de Trabajo de Blender (Fuente: Elaboración Propia)



Entre las que podemos encontrar, modelado básico y anotación *"Layout"*, modelado tridimensional *"Modeling"*, modelado específico con herramientas de esculpido de figuras *"Sculpting"*, edición y configuración de la luz *"UV editing"*, herramientas de creación y asignación de texturas *"Texture Paint"*, sombreado *"Shading"*, generación de animaciones *"Animation"*, configuración de los parámetros de renderizado del modelo *"Rendering"*, entorno de creación y edición de composición *"Composing"*, interfaz técnica de datos y gestión de la información de los nodos geométricos de los elementos *"Geometry Nodes"*, entorno de programación con Python para la ejecución y generación de códigos ***"Scripting"***, interfaz de modelado BIM bajo el estándar IFC introducido por la extensión comentada ***"BIM"***.

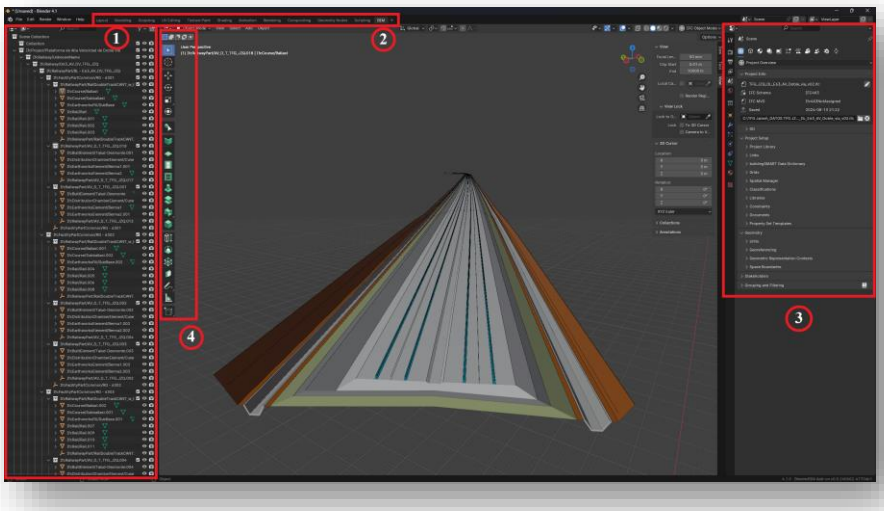
Donde Cabe destacar que serán los dos últimos resaltados entornos de trabajo los que se usarán en mayor cantidad para el caso de estudio. Por lo tanto, como se habrá deducido, ese último espacio de trabajo no es nativo por defecto de Blender, sino que se incluye con la instalación de la extensión de BlenderBIM. Además del espacio de trabajo se añaden infinidad de herramientas entre las que se destacan en primera instancia las de apertura de ficheros IFC para cargarlos en el modelo de Blender.

Figura 171 – Nuevas opciones de Archivo introducidas por BlenderBIM (Fuente: Elaboración Propia)



Una vez se abre el archivo IFC *"Open IFC Project"* almacenado tras la exportación, se abre el modelo en el espacio de trabajo de BIM y se distinguen las principales diferencias frente al resto de entornos acotados.

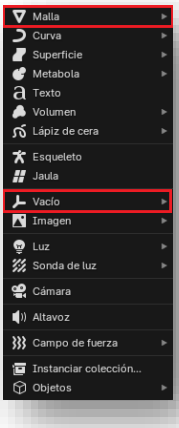
Figura 172 – Interfaz Gráfica del panel BIM de Blender (Fuente: Elaboración Propia)



Entre los distintos elementos de la interfaz gráfica del espacio de trabajo de BIM se distinguen principalmente:

1. **Árbol de elementos del modelo**, este desglose muestra la jerarquía de todos los elementos del modelo cargado, cada uno de los contenedores espaciales se agrupan en contenedores de Blender y se gestiona la entidad espacial del mismo contenedor como un elemento vacío que se traduce en una entidad intangible en el modelo digital, mientras que los objetos con representación gráfica se muestran como mallas *"mesh"* con el símbolo de un triángulo invertido.

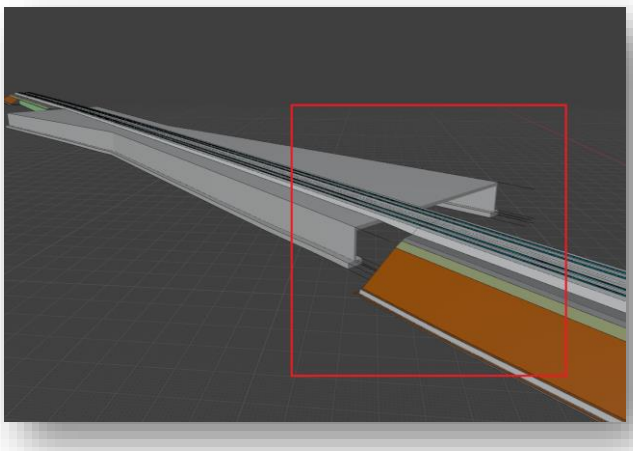
Figura 173 – Objetos de Blender (Fuente: Elaboración Propia)



2. **Espacios de trabajo**, como se ha visto con anterioridad Blender dispone de varios espacios de trabajo que disponen de herramientas preconfiguradas en la interfaz para facilitar el flujo de trabajo del usuario en función de la tarea que se dispone a ejecutar.
3. **Panel de herramientas**, desde el panel de herramientas se puede configurar la inmensa cantidad de parámetros de la escena, acceder a los objetos del modelo y sus propiedades y editarlas. En adelante se profundizará en el panel a medida que se necesite su uso.
4. **Acceso rápido a herramientas genéricas de Blender y añadidos de modelado BIM**, barra de herramientas a disposición rápida del usuario como estilo de cursor que se utiliza, selección de objetos, además de funciones de modelado introducidas por la extensión. No obstante, estas herramientas de modelado son algo básicas y enfocadas a edificación, por lo que no se usarán.

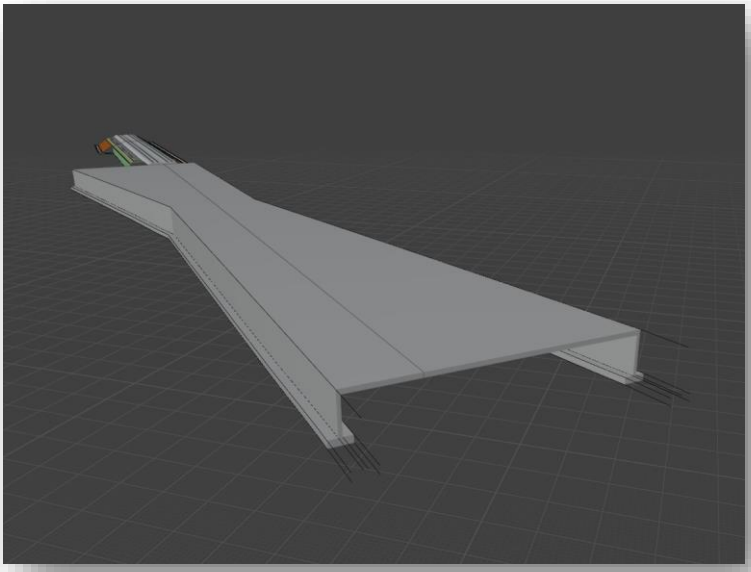
Una vez se ha comprendido la interfaz del software se procede con el ajuste de las incongruencias gráficas que se mostraron mediante el visor BIMvision.

Figura 174 – Incongruencia del modelo a solucionar (Figura: Elaboración Propia)



El marco que forma la subestructura bajo la vía férrea se compone de dos objetos espaciales, una por cada lateral de la vía. Se va a plantear la extrusión de la geometría existente explicando cada uno de los pasos seguidos para efectuar un cambio en la representación gráfica de un elemento IFC mediante el software de modelado Blender.

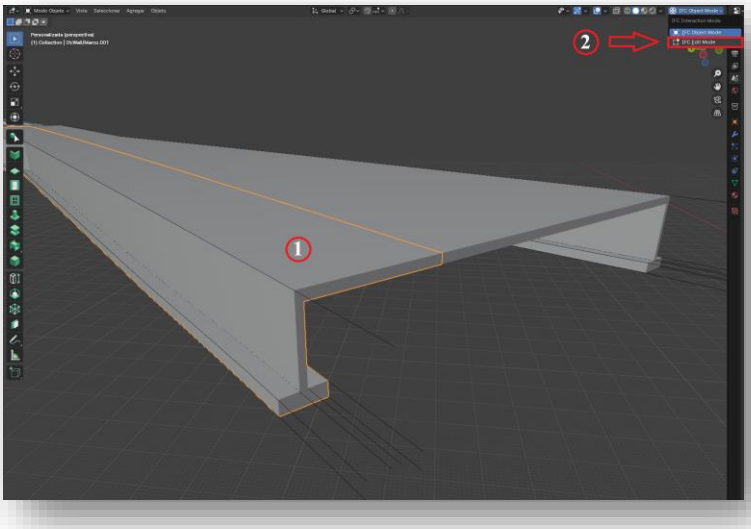
Figura 175 – Marco de subestructura de un tramo de la obra lineal (Figura: Elaboración Propia)



Para un modelado más cómodo se ocultan el resto de los elementos que no interesan, se ha ocultado por tanto los objetos de la vía y el tramo siguiente de la obra lineal al completo. Para ocultar un objeto en Blender, simplemente se necesita seleccionarlo en el espacio de modelo y presionar la tecla ***H***.

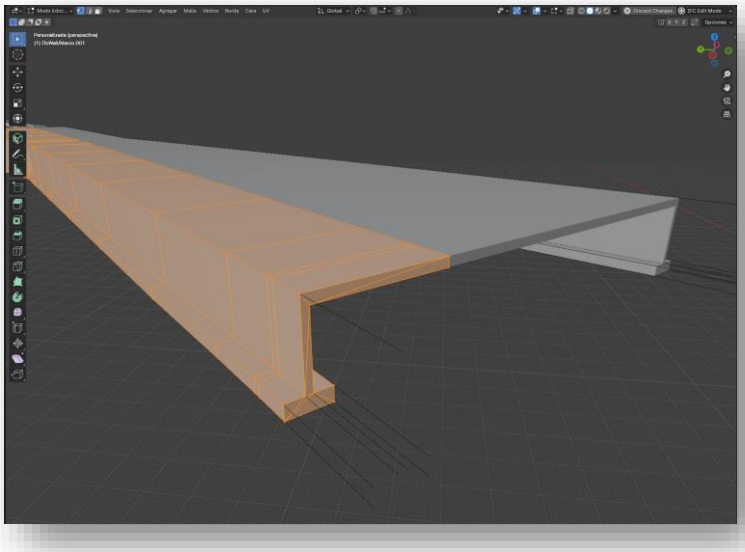
Como se puede apreciar en el modelo, la decisión de exportar las líneas características de la obra lineal no ha sido trivial, tras el análisis de los fallos de generación geométrica en pequeños tramos de la obra lineal se dedujo que se necesitarían las entidades de referencia que la definen. Por lo tanto, gracias a las líneas características se podrá reconstruir la parte del marco que falta, de una forma similar al dibujo digital asistido *CAD* al que estamos acostumbrados.

Figura 176 – Cambiar a la Edición de Objetos IFC de Blender I (Fuente: Elaboración Propia)



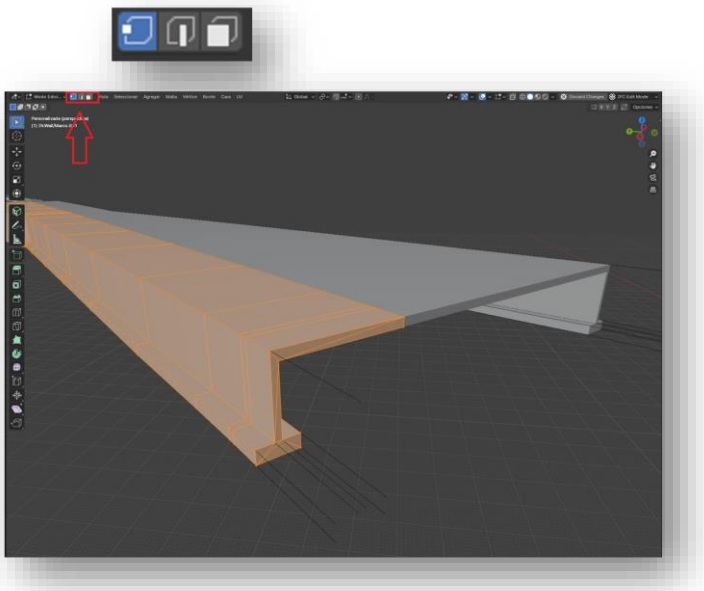
Para editar un objeto se debe seleccionar el objeto que se desea editar y una vez esté seleccionado en el espacio (dispondrá de un contorno anaranjado) se procede a cambiar al modo de edición de IFC en la esquina superior derecha del espacio de trabajo como se muestra en la figura.

Figura 177 – Cambiar a la Edición de Objetos IFC de Blender II (Fuente: Elaboración Propia)



Una vez se ha cambiado al modo de edición se deben de apreciar el modelo los vértices, bordes y caras del mallado del elemento. Estos tres elementos mencionados son los pilares de la geometría que se representa por lo que serán mencionados constantemente. Se puede seguir una modificación por vértices, por bordes o por caras, como se muestra en la siguiente figura se deberá usar la herramienta que se desee siendo cada uno de los cubos respectivamente la selección de una entidad.

Figura 178 – Herramientas de selección para el modo de Edición (Fuente: Elaboración Propia)

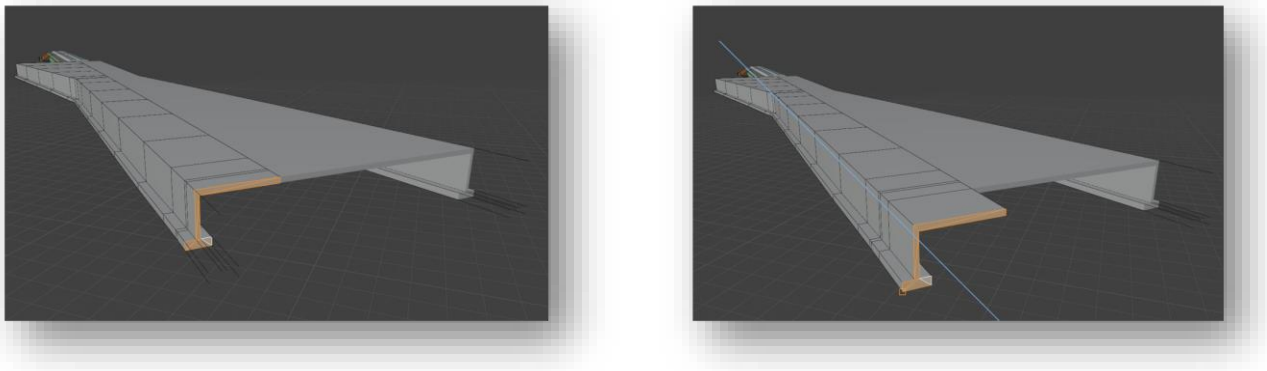


Además, es importante activar la opción de *"Adherir"* con símbolo de imán que se encuentra en la misma barra que la de selección mostrada en la figura anterior, pero aproximadamente en el centro de la pantalla. Esta herramienta nos ayudará a realizar los dibujos de forma similar al CAD, como si estuviese imantado al elemento que se seleccione, en este caso se ha seleccionado vértices.

Figura 179 – Activación de herramienta de adherir en Blender (Fuente: Elaboración Propia)

Una vez configurado el entorno para realizar las ediciones del modelado, en primera instancia como la estructura del lateral izquierdo sigue una trayectoria perpendicular a su sección simplemente bastará con efectuar una extrusión de la misma seleccionando todas las caras que la componen. Se selecciona por tanto la herramienta de selección por caras.

Figura 180 – Extrusión de la sección del marco desde Blender (Fuente: Elaboración propia)



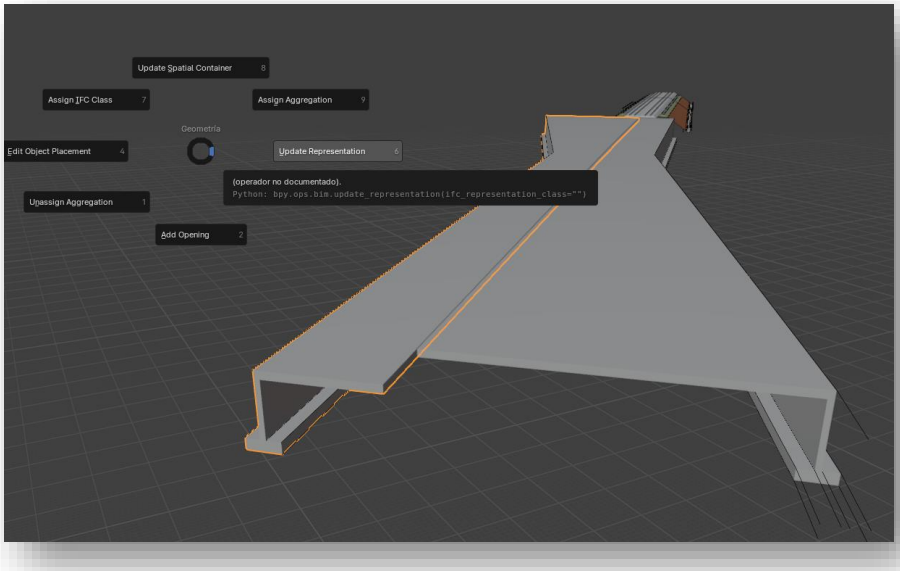
Una vez están seleccionadas todas las caras de la sección se presiona la tecla **E** y comienza la extrusión de la sección, como se ha activado la opción de adherir se busca el vértice final de la línea característica y se extruye la figura hasta ese vértice que se selecciona con facilidad gracias a la adhesión.

La edición de modelado para esta figura ya estaría, pero falta un último paso y es actualizar la representación gráfica del IFC. Este paso es crucial ya que si no se actualiza la representación gráfica se modificará la geometría del modelo, pero no se conseguirá registrarla en el IFC que es el objetivo final.

Acorde a la documentación de la extensión, si se sale del modo de edición (de la misma forma que anteriormente) de nuevo al modo de objeto, y presionando las teclas **SHIFT+E** se despliega un modelo dinámico alrededor del

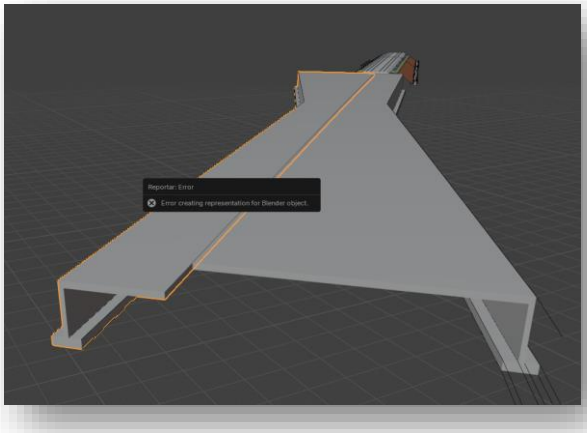
cursor con opciones específicas de BlenderBIM, deberemos tener la figura modificada seleccionada y presionar el botón de actualizar representación *"update representation"*.

Figura 181 – Actualización de la representación en BlenderBIM (Fuente: Elaboración Propia)



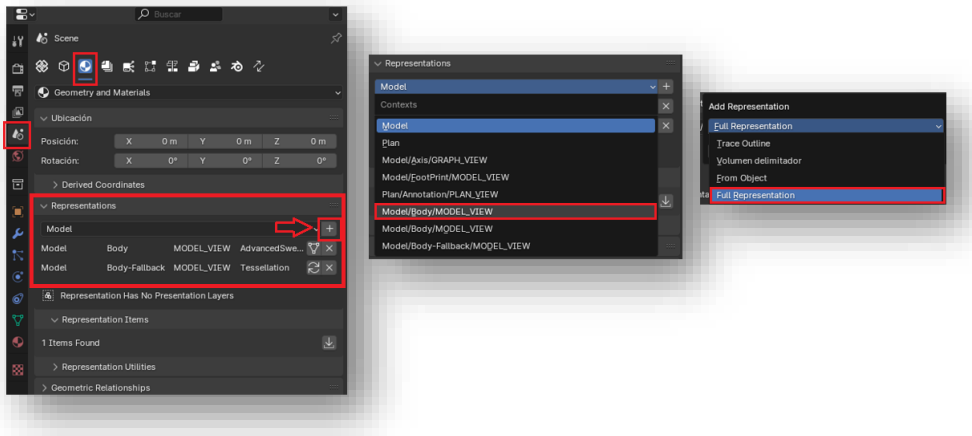
Esa es la forma correcta de actualizar la geometría acorde el IFC, pero en este caso debido al constante desarrollo de la herramienta arroja un error. De tal forma que se deberá de proceder siguiendo otra dinámica de registro. No obstante, se ha presentado porque lo más probable es que sea corregido en futuras actualizaciones.

Figura 182 – Error de actualización de la representación en BlenderBIM (Fuente: Elaboración Propia)



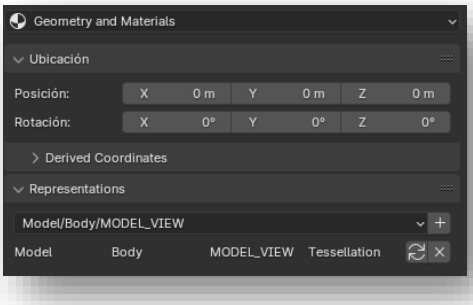
La solución la encontraremos en la barra de herramientas descrita en la descripción de la interfaz de Blender (ver figura XX), dentro del panel de escena *"scene"* en la opción de geometría y materiales *"geometry and materials"* podemos encontrar las representaciones que dispone el elemento que seleccionamos (se ha de tener seleccionado el elemento modificado) en el deplegable *"Representations"*. En primer lugar, se despliega el contexto con el que vamos a añadir una representación y seleccionamos la *"MODEL_VIEW"*, posteriormente se ha de presionar el botón de añadir que se muestra con un símbolo positivo *"+"* y se seleccione la representación completa de *"Full Representation.*

Figura 183 – Actualización manual de la representación del elemento (Fuente: Elaboración Propia)



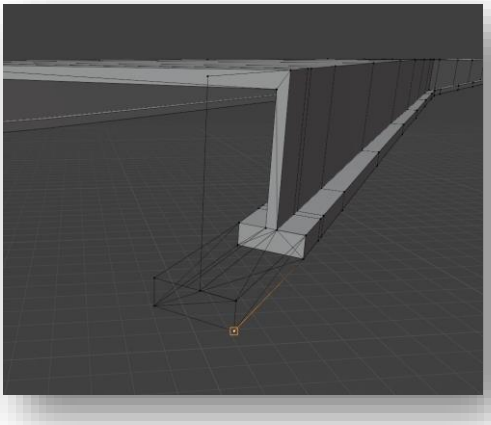
Por último, una vez añadida la nueva representación, se eliminan el resto de las representaciones, quedando así únicamente la representación recién actualizada con las modificaciones introducidas.

Figura 184 – Actualización de la representación manual terminada (Fuente: Elaboración Propia)



Una vez terminado con el elemento de marco izquierdo se comienza con el remodelado del marco derecho, que al no ser extruido perpendicularmente a su sección se ha decidido generar su mallado a partir de la extrusión de vértices generando así una serie de bordes, que triangulando y completando los huecos con mallas se generará un modelado exitoso, siguiendo con las líneas características que marca el trazado.

Figura 185 – Modelado del marco de subestructura mediante extrusión de vértices en Blender (Fuente: Elaboración Propia)



Para la extrusión de vértices se debe tener marcada la herramienta de selección en vértices (ver figura XX) y al seleccionar un vértice presionar la tecla **E** una vez se presiona, nos permite generar una extrusión que genera por tanto un borde o *"línea"* que gracias a disponer la herramienta de adherir nos permitirá ir modelando en forma de triangulación. Además, a medida que los triángulos se vayan generando se ha de ir seleccionando los grupos de vértices del triángulo y presionando la tecla **F** se genera la malla.

Figura 186 – Proceso de generación del nuevo mallado de la extensión del marco I (Fuente: Elaboración Propia)

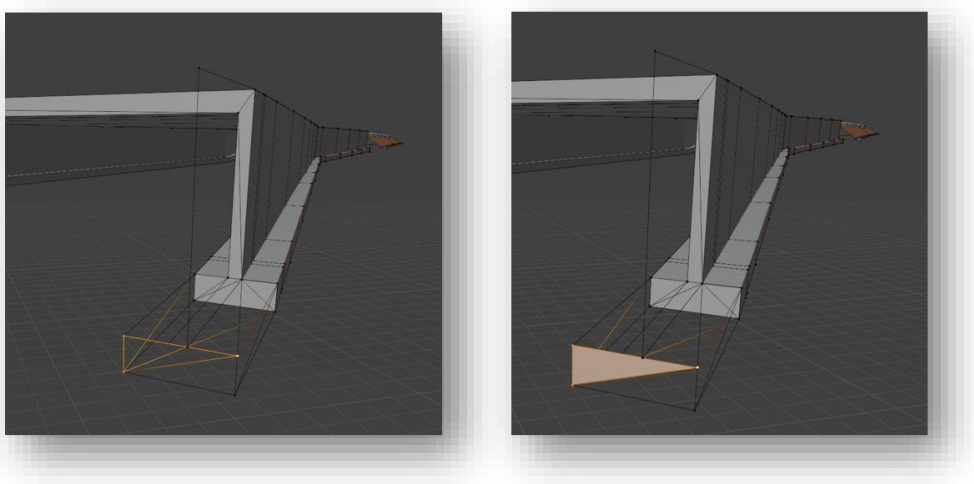
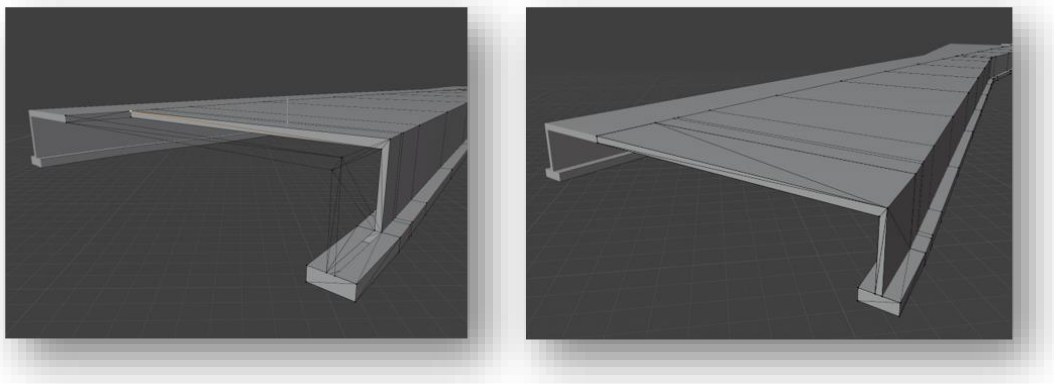


Figura 187 - Proceso de generación del nuevo mallado de la extensión del marco II (Fuente: Elaboración Propia)

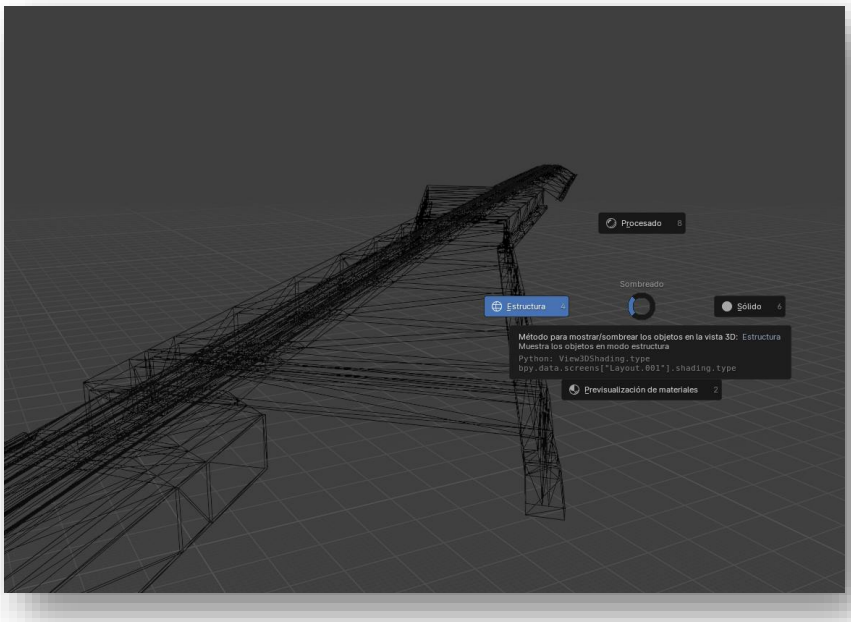


Terminado el mallado del elemento completo debe recordarse la actualización de la representación del elemento, siguiendo los mismos pasos que en el caso anterior (recordar salir del modo de edición), para que así los cambios geométricos queden registrados en el modelo IFC a posteriori.

Remodelado el marco ya se puede continuar con el resto de los elementos que no se documentarán con tanto detalle, ya que siguen la misma dinámica de modelado. Además, cabe destacar que se puede cambiar el estilo de visualización del modelo a un estilo alámbrico estructural que muestra los vértices y bordes de todas las mallas que conforman los elementos del modelo. Si se presiona la tecla **Z** se despliega un menú circular, en ese menú seleccionamos la opción de la izquierda de ***Estructura*** y se podrá apreciar la geometría generada.

Este aspecto se destaca en relación con el aligeramiento de modelos. En muchos casos la triangulación no se genera de forma óptima en el modelo y se sobredimensionan los triángulos con muchos más vértices y bordes. En este estilo de visión del modelo se pueden detectar anomalías que ralentizan el modelo y corregirlas con las técnicas mostradas, optimizando así el funcionamiento del IFC.

Figura 188 – Visión alámbrica de Estructura del modelo en Blender (Fuente: Elaboración Propia)



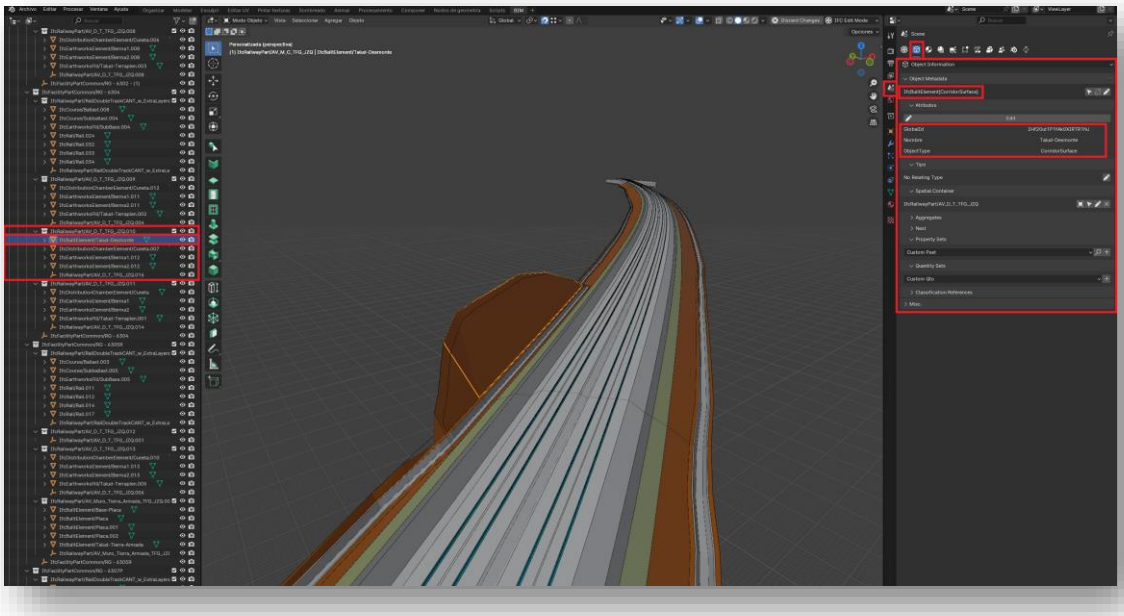
Siguiendo esta dinámica con las herramientas de modificación del modelado IFC desde Blender se pueden solucionar muchas incongruencias obtenidas de forma geométrica, añadir elementos nuevos desde cero, efectuar modificaciones y optimizar la geometría del modelo como se ha comentado previamente, entre otros.

Por lo que en el presente caso de estudio se ha revisado minuciosamente cada una de las incongruencias geométricas y aportado una solución, no obstante, por simplificación no se ha detallado paso a paso cada una de las modificaciones, compárese la versión obtenida de la exportación con la última generada por Blender para apreciar todos los cambios.

Entonces, al estar los ligeros errores geométricos corregidos ya no necesitamos las líneas características en el modelo, esto conduce a la cuestión de cómo limpiar un IFC mediante BlenderBIM. Ya que, además, también conviene eliminar las alineaciones *"IfcAlignment"*, anotaciones *"IfcAnnotation"* y todos los elementos genéricos residuales *"IfcBuiltElement"* que se hayan podido generar. Además, también se realizarán modificaciones, en el caso de los taludes en desmonte, en el proceso de exportación no se pudo asignar su entidad equivalente en IFC *"IfcEarthworksCut.CUT"* por lo que se aprovechará para asignarle su clase correspondiente mediante BlenderBIM.

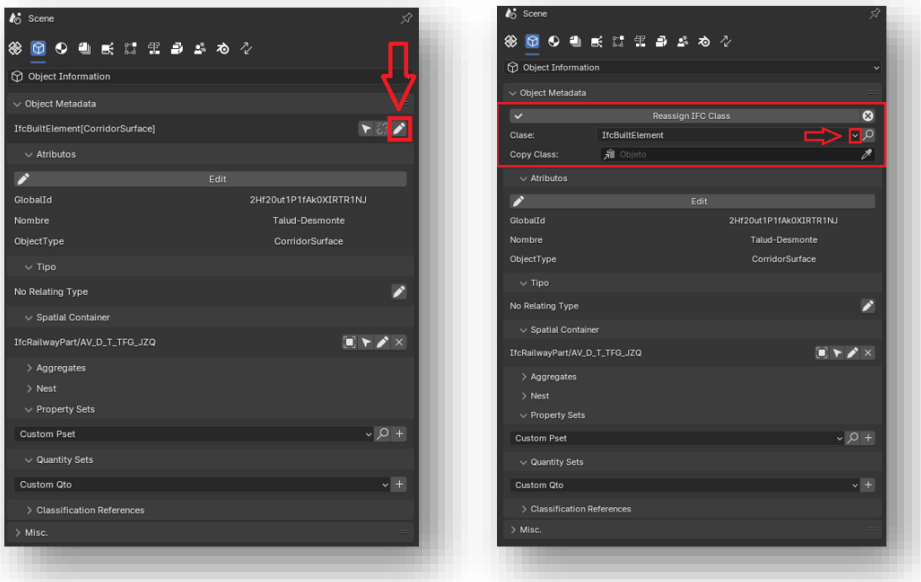
En primer lugar, se explicará paso a paso como modificar los atributos IFC de un elemento desde la interfaz de Blender, para posteriormente mostrar cómo se han aplicado algoritmos de Python que automatizan la tarea de eliminación, reasignación y modificación de cada uno de los elementos.

Figura 189 – Modificación de la entidad IFC asignada a un elemento en BlenderBIM I (Fuente: Elaboración Propia)



En primer lugar, se debe seleccionar el elemento que se desea modificar, puede ser seleccionado mediante selección espacial en la visión del modelo o en el árbol jerárquico de elementos que se dispone en la izquierda de la interfaz. Como puede comprobarse, se ha seleccionado un desmonte que en este momento está identificado como un elemento genérico constructivo *"IfcBuiltElement"*. En la parte derecha de la interfaz se dispone del panel de herramientas donde se deberá entrar en el panel específico de la escena *"scene"* en la opción de información del objeto *"Object Information"*.

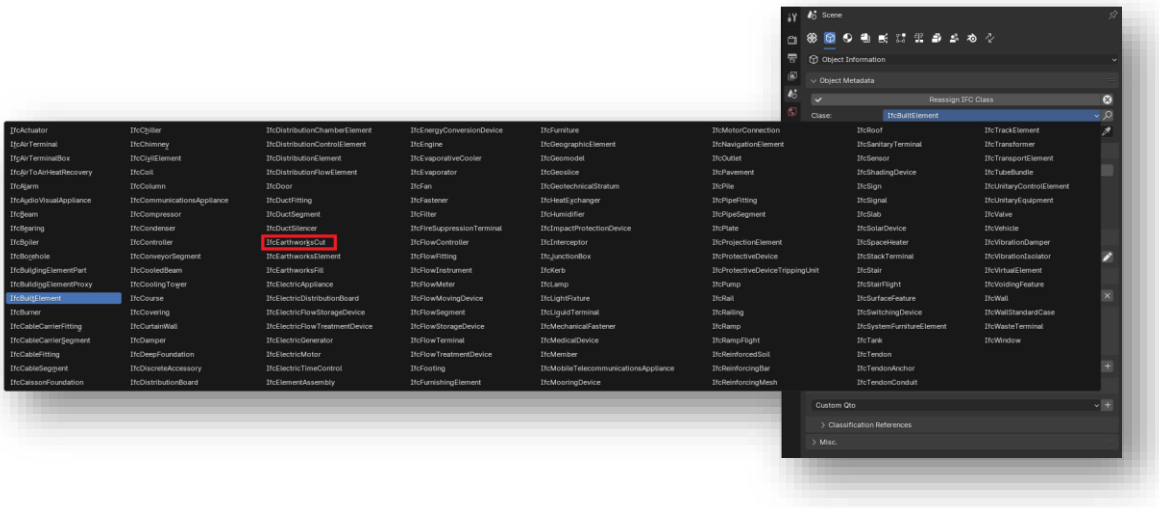
Figura 190 – Modificación de la entidad IFC asignada a un elemento en BlenderBIM II (Fuente: Elaboración Propia)



Dentro del panel de información del objeto, se puede encontrar toda la información correspondiente al elemento en el modelo IFC. En lo que corresponde a la cuestión que se trata, se ha de centrar la atención en el primer elemento de información dentro del desglose de los metadatos del elemento. Indica la entidad IFC que está

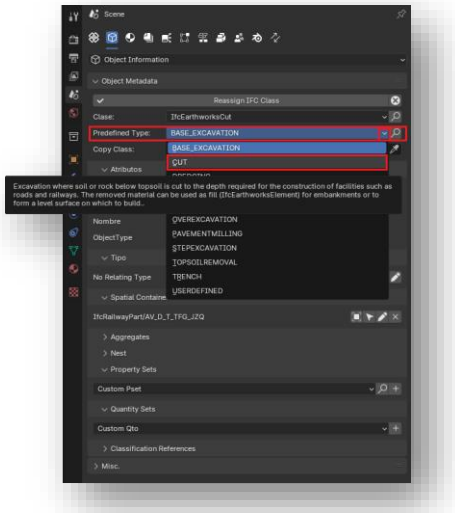
asignada al elemento y a su derecha dispone de varios botones que serán de especial interés. En este caso, se accionará el último botón con símbolo de lápiz que alude a la edición de la información. Una vez se acciona el botón, como se aprecia en la figura se despliega un submenú con los atributos que se pueden modificar, en este caso únicamente aparece la entidad IFC que tiene asignada el elemento y a su derecha esta vez se muestra tanto un icono de desplegable como una lupa. Para este ejemplo, se accionará el desplegable que muestra todas las entidades IFC que dispone el esquema actualizado IFC4.3, no obstante, la lupa acciona una búsqueda que a partir de las palabras que usemos se podrá encontrar la entidad IFC que se desea.

Figura 191 – Modificación de la entidad IFC asignada a un elemento en BlenderBIM III (Fuente: Elaboración Propia)



En el interior del desplegable se pueden encontrar todas y cada una de la clase de las que dispone el esquema y entre ellas se muestra la entidad de movimiento de tierras de desmonte *"IfcEarthworksCut"*, a diferencia de Autodesk, parece que BlenderBIM sí tiene el esquema actualizado. Se selecciona la entidad que se desea reasignar y se cerrará el desplegable. Una vez se ha seleccionado una entidad a la que se le pueden asignar tipos predefinidos, BlenderBIM mostrará bajo la entidad IFC un nuevo campo de información para ellos *"Predefined Type"*. Su funcionamiento es análogo al de la clase de entidad IFC.

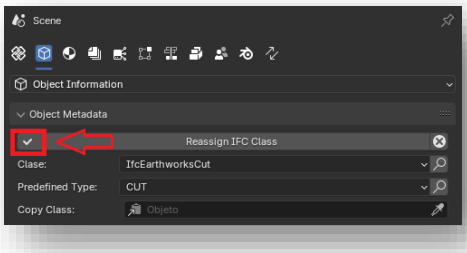
Figura 192 – Modificación de la entidad IFC asignada a un elemento en BlenderBIM IV (Fuente: Elaboración Propia)



Se selecciona por tanto el tipo predefinido que se le deseaba asignar desde un inicio en las plantillas de exportación *"CUT"*. Cabe destacar, que a pesar de que está en inglés, cuando se dispone el cursor sobre una entidad IFC o un tipo predefinido de la misma se muestra un mensaje emergente bajo el cursor que incluye la descripción del elemento que corresponde con la documentación oficial del último esquema IFC de buildingSmart [25].

Como se aprecia en la siguiente figura, antes de finalizar la edición el último campo después de los tipos predefinidos consiste en la copia de clases *"Copy Class"* a partir de ese campo podemos seleccionar otros elementos del modelo y copiar su clase y tipo predefinido que tienen asignados, esto es útil cuando se está cambiando muchos elementos de misma naturaleza que ya están presentes en el modelo.

Figura 193 – Modificación de la entidad IFC asignada a un elemento en BlenderBIM V (Fuente: Elaboración Propia)



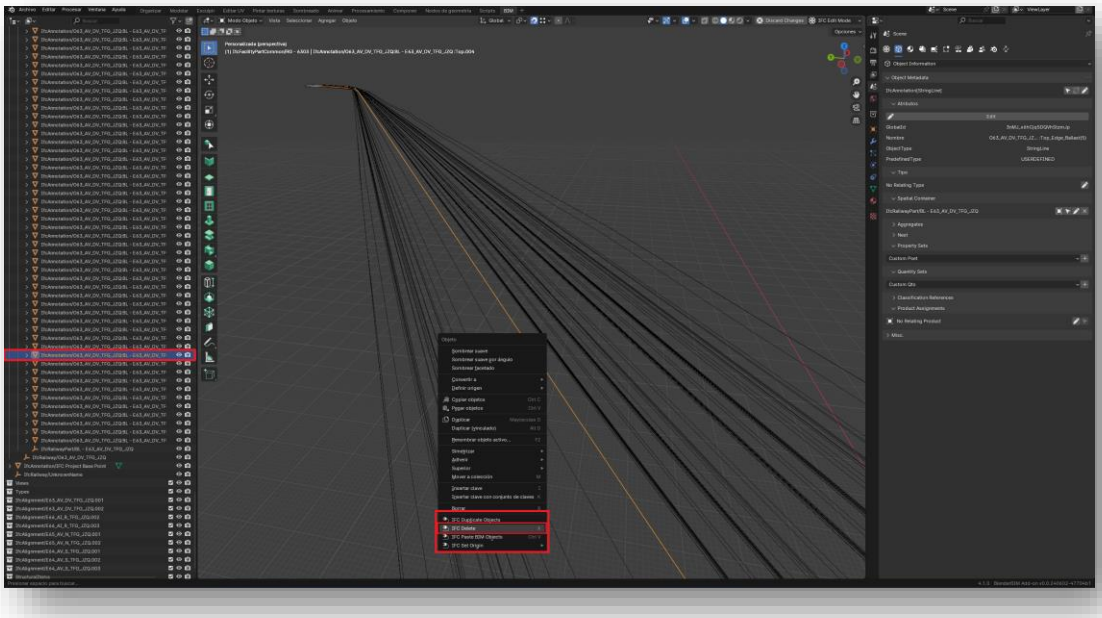
Una vez se ha finalizado la edición del elemento se ha de presionar el botón en la esquina superior izquierda con símbolo de *"check"* que se destaca en rojo en la figura. Así se registran los cambios y la nueva entidad IFC ha sido asignada.

Modificados los elementos que fueron erróneamente asignados en una primera iteración desde Civil3D ahora se mostrará como se pueden eliminar correctamente los que no han de permanecer en el modelo. De igual forma que para la reasignación se mostrará para conocimiento del usuario medio como puede ejecutarse esta tarea mediante la interfaz del usuario, pero posteriormente se ejecutarán toda la limpieza de elementos sobrantes en un código de Python automatizado.

Se destaca la eliminación de las entidades IFC porque no deben eliminarse como un elemento genérico de modelado de Blender, sino que debe de accionarse especialmente con las funcionalidades de BlenderBIM, esto se debe a que al eliminar un elemento no solo se ha de eliminar en el modelo de Blender sino también del código que contiene el esquema del fichero (.ifc) que se está tratando. Por ello se ha de usar la eliminación de elemento IFC y nunca la genérica de Blender ya que no se registrará en el fichero IFC.

En la siguiente figura, se aprecia como se han ocultado los elementos constructivos (seleccionar objetos a ocultar y presionar tecla **H**) de un tramo de la obra lineal para poder apreciar la cantidad de líneas características que hay en el dibujo, para mostrar cómo se deben de eliminar. Una vez se tiene seleccionada la entidad a eliminar en este caso una de las muchas líneas características "*IfcAnnotation"* y se presiona clic derecho. En el menú que despliega, se puede apreciar como en la parte inferior se encuentran todas y cada una de las opciones que ofrece BlenderBIM en lo referente a IFC, bastará con efectuar la de eliminación *"IFC delete"* o presionar la tecla **X**.

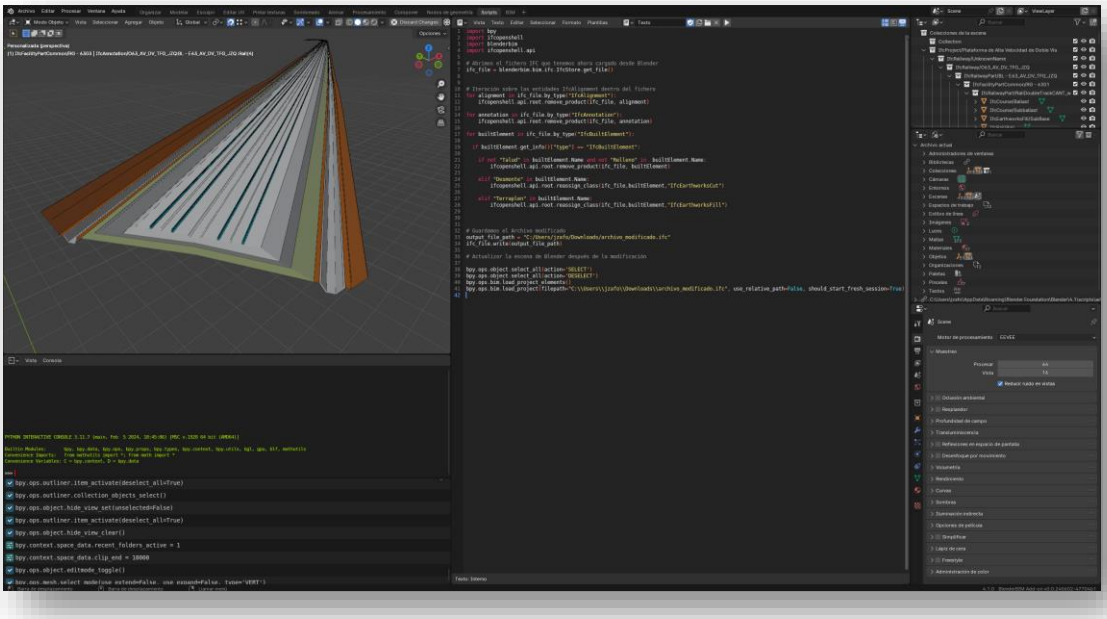
Figura 194 - Eliminación de una entidad IFC en BlenderBIM (Fuente: Elaboración Propia)



Para volver a visualizar directamente todos los elementos que han sido ocultados se presionan las teclas ***ALT+H***.

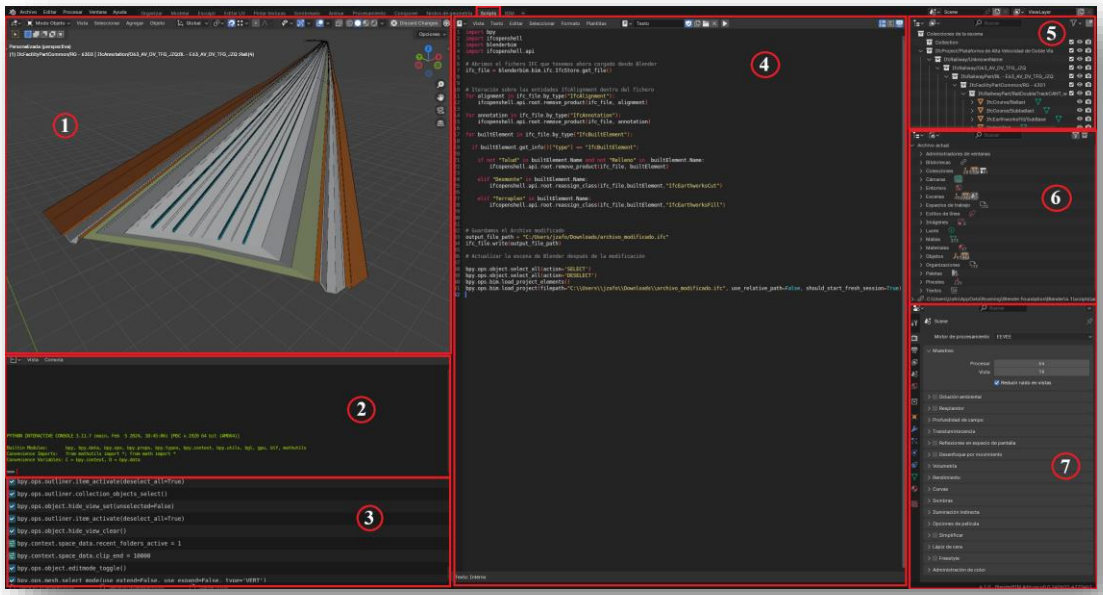
Ahora se mostrará el entorno de trabajo para la ejecución y generación de algoritmos mediante el lenguaje de programación Python (para más detalle de la programación aplicada y códigos desarrollados (ver capítulo 7) de tal forma que se mostrarán también los pasos seguidos para ejecutar los códigos que satisfacen las necesidades mostradas a lo largo del capítulo de forma automatizada para optimizar el modelo.

Figura 195 – Espacio de trabajo de entorno de programación "Script" en Blender (Fuente: Elaboración Propia)



Para aplicar los códigos que han sido redactados para la edición automatizada del modelo se ha de cambiar Blender al espacio de trabajo de entorno de programación *"Script"* en este espacio de trabajo se podrá ejecutar códigos, escribirlos desde cero y revisar la consola de Blender acorde a las instrucciones que se envían.

Figura 196 – Desglose del espacio de trabajo "Script" en Blender (Fuente: Elaboración Propia)



Para un mejor entendimiento del entorno, se ha desglosado cada uno de los paneles de Blender que lo conforman:

1. **Interfaz gráfica del modelo**, la vista 3D del modelo se desplaza a un segundo plano, colocándose en la esquina superior izquierda. Aunque este entorno está orientado principalmente a la programación, es fundamental mantener la vista 3D accesible. Esto permite a los desarrolladores verificar visualmente los efectos de sus scripts en el modelo en tiempo real, garantizando que las automatizaciones y modificaciones realizadas a través del código se reflejen correctamente en la interfaz gráfica.

2. **Consola de Blender,** es un componente crucial en el espacio de trabajo de Scripts. Aquí, los desarrolladores pueden escribir comandos en Python y ver su ejecución en tiempo real. Es una herramienta indispensable para la depuración, ya que permite la ejecución inmediata de pequeños fragmentos de código y proporciona retroalimentación directa sobre su funcionamiento.

3. **Historial de Comandos Ejecutados,** este panel muestra una lista de todos los comandos que Blender ha ejecutado recientemente. Es útil para los desarrolladores que desean revisar o reutilizar comandos anteriores sin tener que reescribirlos. Además, proporciona una visión clara del flujo de trabajo y permite identificar rápidamente cualquier error o comando problemático.

4. **Editor de texto para generar Scripts**, donde los desarrolladores pueden escribir, editar y guardar sus scripts en Python. Este panel ofrece herramientas como resaltado de sintaxis y numeración de líneas, lo que facilita la escritura y depuración de código. Es el corazón del entorno de programación en Blender, ya que aquí se crean los scripts que automatizan tareas, generan herramientas personalizadas y extienden las funcionalidades de Blender.

5. **Árbol de elementos del modelo (Outliner)**, es un panel que muestra una vista jerárquica de todos los objetos y elementos que componen la escena actual. En el espacio de trabajo de Scripts, es particularmente útil para navegar rápidamente entre diferentes elementos del modelo mientras se escribe

código, permitiendo a los desarrolladores identificar y seleccionar objetos específicos que serán manipulados a través de scripts.

6. **Explorador de archivos**, permite a los desarrolladores acceder a los archivos de su sistema directamente desde Blender. Es útil para abrir scripts existentes, guardar nuevos archivos de código, y organizar los proyectos de manera eficiente. El Explorador de Archivos facilita la gestión de los recursos y scripts necesarios para el desarrollo en Blender.

7. **Panel de Herramientas**, ofrece acceso rápido a diversas funciones y herramientas.

Acorde a las necesidades descritas anteriormente el script desarrollado serán varias líneas de código que automaticen:

- Eliminación de todas las entidades *"IfcAnnotation"* del modelo.
- Eliminación de todas las entidades *"IfcAlignment"* del modelo.
- Reasignación de la entidad *"IfcEarthworksCut"* a los desmontes.
- Reasignación de la entidad *"IfcEarthworksFill"* a algunos rellenos que no se han asignado.
- Asignación de la entidad *"IfcPlate"* a las placas que contienen la tierra armada que no se nos ha asignado ya que no aparecían en la plantilla.

Figura 197 – Script generado para modificaciones de las entidades del primer modelo (Fuente: Elaboración Propia)

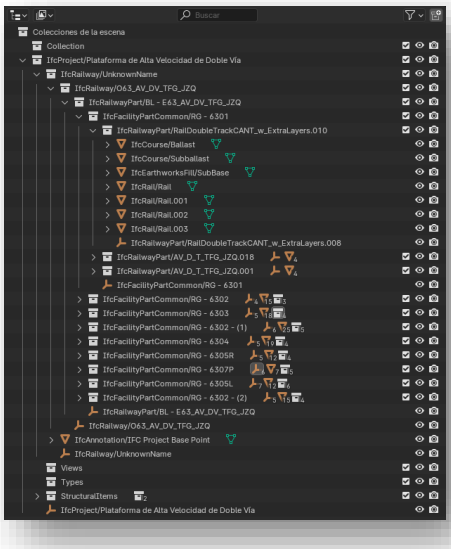
```
Vista  Texto  Editar  Seleccionar  Formato  Plantillas   Texto

import bpy
import ifcopenshell
import blenderbim
import ifcopenshell.api

# Abrimos el fichero IFC que tenemos ahora cargado desde Blender
ifc_file = blenderbim.bim.ifc.IfcStore.get_file()


# Iteración sobre las entidades IfcAlignment dentro del fichero
for alignment in ifc_file.by_type("IfcAlignment"):
    ifcopenshell.api.root.remove_product(ifc_file, alignment)

for annotation in ifc_file.by_type("IfcAnnotation"):
    ifcopenshell.api.root.remove_product(ifc_file, annotation)

for builtElement in ifc_file.by_type("IfcBuiltElement"):

  if builtElement.get_info()["type"] == "IfcBuiltElement":

    if not builtElement.Name.lower().__contains__("talud") and not builtElement.Name.lower().__contains__("relleno") and not builtElement.Name.low

        ifcopenshell.api.root.remove_product(ifc_file, builtElement)

    elif "desmonte" in builtElement.Name.lower():
        ifcopenshell.api.root.reassign_class(ifc_file,builtElement,"IfcEarthworksCut", "CUT")

    elif "terraplen" in builtElement.Name.lower():
        ifcopenshell.api.root.reassign_class(ifc_file,builtElement,"IfcEarthworksFill", "SLOPEFILL")

    elif "placa" in builtElement.Name.lower():
        ifcopenshell.api.root.reassign_class(ifc_file,builtElement,"IfcPlate")



# Guardamos el Archivo modificado
output_file_path = "C:/TFG_JAIME/_DATOS/4_OL/_IFC/TFG_JZQ_OL_E63_AV_Doble_via_v07"
ifc_file.write(output_file_path)

# Actualizar la escena de Blender después de la modificación

bpy.ops.object.select_all(action='SELECT')
bpy.ops.object.select_all(action='DESELECT')
bpy.ops.bim.load_project_elements()
bpy.ops.bim.load_project(output_file_path, use_relative_path=False, should_start_fresh_session=True)

```

Una vez se ha terminado de escribir el script, se presiona el símbolo de ejecutar *"play"* y se ejecuta el código redactado en la consola de Blender. El código está escrito para modificar el IFC de forma nativa, guardar una nueva versión en el directorio del CDE comentado y forzar a Blender a abrir ese nuevo modelo almacenado.

Este código podría haber sido escrito fuera de Blender mediante la librería de Python *IfcOpenshell* como se verá en el capítulo 7.

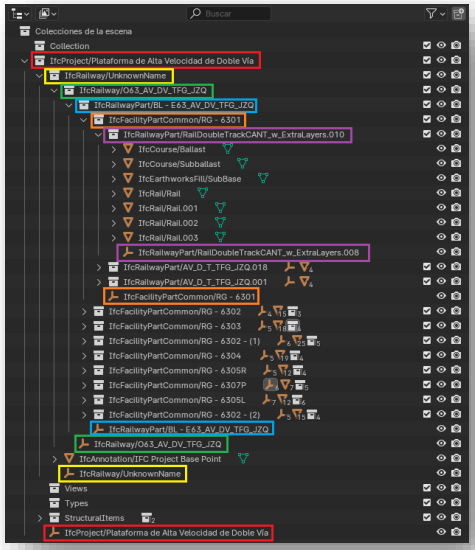
Ya se han arreglado todos los fallos de geometría del modelo y reasignado todas las entidades de los elementos que no habían sido correctamente asignados. Recapitulando, únicamente faltaría reajustar correctamente las estructuras espaciales del modelo y generar los conjuntos de propiedades de forma nativa en el IFC gracias a BlenderBIM. Como se recordará, no se han podido exportar los conjuntos de propiedades a cada uno de los elementos de la obra lineal por lo que se proponen soluciones a partir de BlenderBIM y desarrollo de código.

Figura 198 - Árbol de elementos del modelo (Outliner) de BlenderBIM (Fuente: Elaboración Propia)



Como se trató de explicar con anterioridad, en el árbol de elementos distinguimos principalmente entre los elementos constructivos del modelo que disponen de representación gráfica, mallado y los elementos no tangibles que representan las estructuras espaciales. Cada uno de los contenedores espaciales está representado por una colección de Blender (como una carpeta en una estructura de archivos), y su entidad IFC de estructura espacial suele estar la última dentro del esquema del contenedor. En la siguiente figura se muestra por una relación de colores, el objeto de colección y su entidad IFC de estructura espacial equivalente.

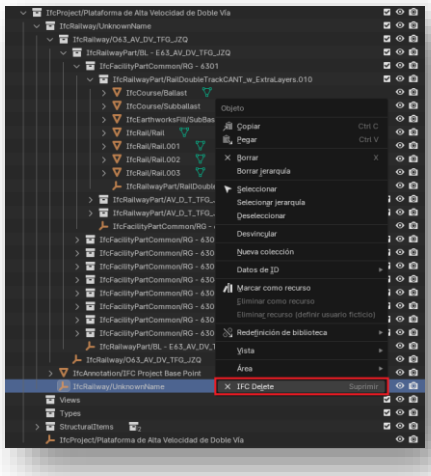
Figura 199 – Relación entre colecciones de Blender y sus entidades espaciales IFC (Fuente: Elaboración Propia)



En la figura se puede apreciar claramente como cada entidad espacial se encuentra al final de cada una de las colecciones, es la entidad espacial la que almacena toda la información sobre la estructura y no la colección. La colección únicamente es un sistema de organización de Blender que toma BlenderBIM para jerarquizar las entidades presentes en el modelo.

Se va a eliminar las estructuras espaciales redundantes de *"IfcRailway"* identificada como *nombre desconocido "Unknwon Name"* resaltada en amarillo en la figura anterior y la "IfcRailwayPart" de la línea base de la obra lineal *"BL - E63_AV_DV_TFG_JZQ"* resaltada en azul.

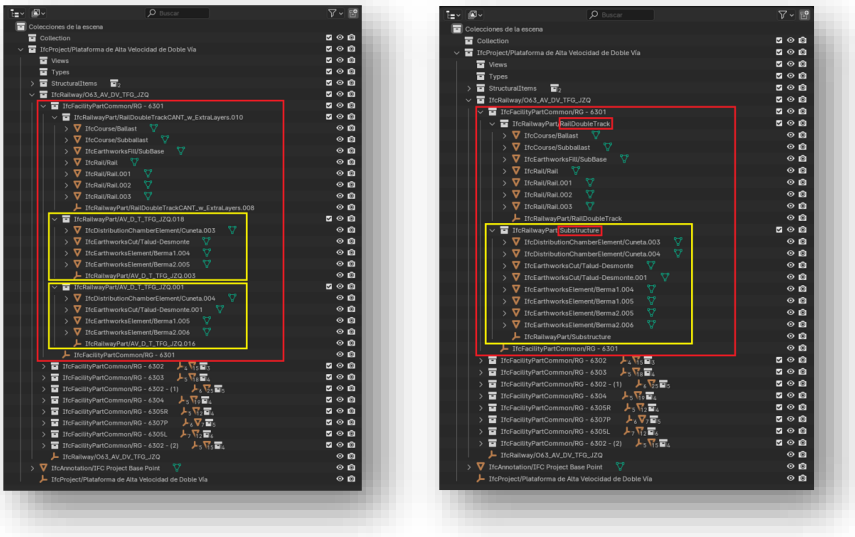
Figura 200 – Eliminación de estructura espacial redundante en BlenderBIM (Fuente: Elaboración Propia)



Además, se va a modificar la estructura espacial dentro de cada uno de los tramos de la obra líneas. Ahora, los tramos están subdivididos en cada uno de los subensamblajes que lo componen, pero en ocasiones se han generado varias veces el mismo subensamblaje y tampoco tiene sentido estructurar espacialmente de forma separada el desmonte de un lateral y el desmonte del otro lateral. Por ello, se reestructurarán todos los tramos

juntando todos los elementos en dos estructuras espaciales, una de la vía y sus componentes de la obra lineal y otra de subestructura de la obra lineal.

Figura 201 – Cambios en la estructura espacial del modelo en BlenderBIM (Fuente: Elaboración Propia)



Para ejecutar los cambios en la estructura espacial es realmente simple, únicamente se ha de tener especial atención en el árbol de elementos que se muestra en la figura. Para mover los elementos de una estructura espacial a otra, se han de seleccionar y arrastrar hasta la otra estructura y una vez se ha quedado la estructura que se desea eliminar vacía, se efectúa el control *"IFC delete"* como se ha aplicado con anterioridad. Hay que recordar que para eliminarlo correctamente ha de ser a la entidad espacial IFC y no a la colección de Blender. Además, se han concebido de nuevo los nombres de las estructuras espaciales verticales como carril doble vía *"RailDoubleTrack"* para la estructura de la vía de la obra lineal y subestructura *"Substructure"* para los elementos que conforman la base de la vía.

Figura 202 – Estructura espacial de subestructura para el tramo RG-6301(Fuente: Elaboración Propia)

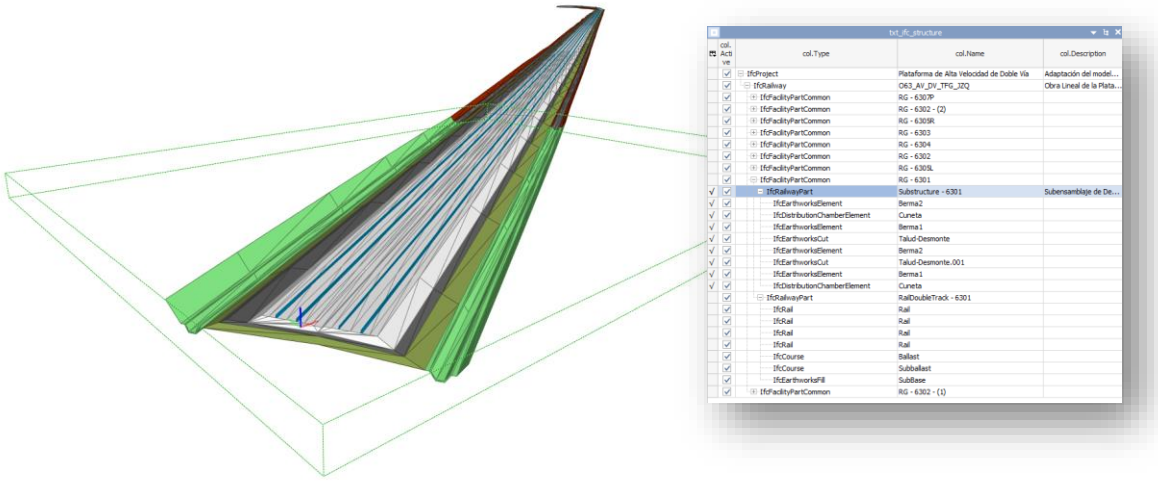
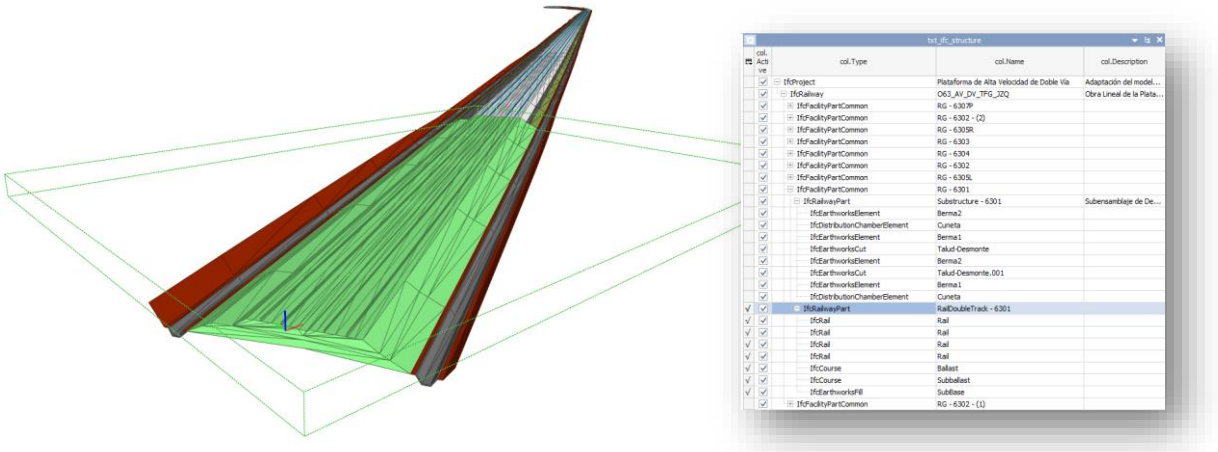


Figura 203 – Estructura espacial de vía para el tramo RG-6301(Fuente: Elaboración Propia)



7 PROGRAMACIÓN PARA IFC

El presente capítulo se centra en la metodología empleada para el desarrollo, análisis y validación de modelos BIM en formato IFC4.3, utilizando herramientas de código abierto, específicamente Python y la librería *IfcOpenShell*. En un contexto donde la interoperabilidad y la eficiencia son fundamentales para la evolución del sector de la construcción, a lo largo de la presente investigación se ha comprobado que la adopción de estándares abiertos como IFC4.3 ha demostrado ser una estrategia clave para mejorar la coordinación y la calidad en proyectos de infraestructura. Este capítulo no solo expone las técnicas y herramientas utilizadas para su más refinada edición, sino que también justifica su elección en base a su relevancia en el ámbito profesional y su capacidad para resolver los desafíos actuales en la gestión de modelos BIM.

La implementación de esta metodología responde a la creciente necesidad en la industria de la construcción de contar con procesos que permitan modificar, analizar y validar modelos abiertos BIM de manera eficiente y precisa debido a su aumento de presencia en el sector. El lenguaje de programación de libre uso Python ha emergido, así como una herramienta poderosa en este campo, debido a su flexibilidad, accesibilidad y capacidad para integrarse con una amplia variedad de aplicaciones utilizadas en la construcción digital. La respuesta de esta presencia de Python en el ámbito de los modelos abiertos es consecuencia de en *IfcOpenShell*, una librería diseñada para manipular archivos IFC, se ha consolidado como una solución indispensable para profesionales que buscan garantizar la interoperabilidad de sus modelos BIM con otros sistemas y plataformas además de ser el motor del desarrollo de complementos IFC en software de modelado libre como Blender.

Para abordar la metodología empleada, este capítulo se estructura en primer lugar como una introducción a los conceptos de programación que se emplean como lenguajes, librerías, software de libre uso, complementos desarrollados sobre otro software, etc. Posteriormente, una vez abordados los conceptos básicos de la programación, se profundizará en las herramientas y metodologías planteadas para el presente trabajo.

Cabe destacar que lejos de tomar un carácter innovador este capítulo pretende acercar a la comunidad de los ingenieros civiles de forma didáctica a la programación aplicada sobre modelos BIM abiertos en formato IFC, no pretende pues inventar nuevas tecnologías sino mostrar las herramientas existentes que están a la disposición de quien se embarque en este campo de aplicación a los modelos BIM.

7.1 Introducción a la Programación

En el ámbito de la ingeniería civil y, específicamente, en el modelado BIM, la programación ha emergido como una herramienta indispensable para el análisis, manipulación y validación de modelos digitales. La creciente complejidad de los proyectos de infraestructura, junto con la necesidad de interoperabilidad entre diversas plataformas de software, ha impulsado a los ingenieros a adoptar la programación como una competencia clave para optimizar flujos de trabajo y garantizar la precisión en los modelos.

La programación en este contexto no solo permite automatizar tareas repetitivas, sino que también facilita la personalización de procesos, la integración de datos de múltiples fuentes y la creación de herramientas específicas que pueden manejar los desafíos únicos que presenta el modelado de infraestructuras civiles. Este capítulo proporciona una introducción a los conceptos básicos de programación relevantes para el trabajo con modelos BIM, con un enfoque en las herramientas y tecnologías más utilizadas en el sector, y cómo estas se aplican en la práctica para mejorar la eficiencia y la calidad de los proyectos de construcción.

PYTHON

Python ha logrado posicionarse como uno de los lenguajes de programación más populares y versátiles en la industria de la ingeniería y la construcción digital. Su sintaxis clara y legible facilita el aprendizaje y uso, permitiendo que tanto principiantes como expertos puedan desarrollar aplicaciones y scripts con relativa facilidad. En el contexto del BIM, Python se destaca por su capacidad para integrar diferentes herramientas y manejar grandes volúmenes de datos de manera eficiente.

Una de las razones por las que Python es ampliamente utilizado en la ingeniería civil es su extensa colección de

bibliotecas especializadas, que permiten desde el análisis numérico hasta la manipulación directa de modelos BIM en formato IFC. Estas bibliotecas no solo amplían las capacidades del lenguaje, sino que también permiten a los desarrolladores crear soluciones altamente personalizadas que se ajusten a las necesidades específicas de cada proyecto.

Figura 204 – Lenguaje de programación de libre uso Python (Fuente: Wikipedia)

Características clave de Python:

- **Sintaxis intuitiva y fácil de aprender**: La sintaxis de Python es simple y directa, lo que facilita su aprendizaje y uso. Esta característica es particularmente valiosa en entornos donde la programación no es la actividad principal, pero se requiere como herramienta complementaria.
- **Amplia comunidad y recursos de aprendizaje**: Python cuenta con una de las comunidades más grandes y activas en el mundo de la programación. Esta comunidad proporciona una gran cantidad de recursos, incluyendo documentación, tutoriales, y foros de soporte que son esenciales para resolver problemas y aprender nuevas técnicas.
- **Gran cantidad de bibliotecas especializadas para ingeniería y BIM**: Python ofrece una variedad de bibliotecas que están específicamente diseñadas para tareas de ingeniería, como *NumPy* y *Pandas* para el análisis de datos, y *Matplotlib* para la visualización. En el contexto de BIM, bibliotecas como *IfcOpenShell* permiten a los usuarios trabajar directamente con archivos IFC.
- **Capacidad de integración con otras herramientas y software de modelado**: Python se integra fácilmente con otros sistemas y software utilizados en la construcción digital, como Blender, AutoCAD, y Revit. Esta capacidad de integración es crucial para asegurar la interoperabilidad y eficiencia en el manejo de modelos BIM.

Python no solo es una herramienta para automatizar tareas, sino que también proporciona una base sólida para el desarrollo de aplicaciones más complejas que pueden abordar los desafíos específicos del modelado y análisis de infraestructuras civiles.

Cabe destacar que **Python** es un lenguaje de fácil comprensión para usuarios principiantes a la programación. No obstante, si el usuario conoce lenguajes como **Matlab**, que es el caso de los ingenieros civiles formados en la ETSI, la curva de aprendizaje se maximiza. Teniendo en cuenta los beneficios que ofrece este lenguaje de libre uso, se recomienda altamente a los ingenieros civiles con conocimiento de **Matlab** que tomen **Python** si están interesados en la materia que se trata.

BIBLIOTECAS Y FRAMEWORKS

Las bibliotecas y frameworks son componentes esenciales en la programación moderna, ya que proporcionan colecciones de código escrito de forma previa por otros desarrolladores que extienden las funcionalidades de un lenguaje de programación. En el contexto del BIM y el estándar IFC, estas herramientas permiten a los desarrolladores y profesionales manipular modelos con precisión, validar su conformidad con los estándares, y generar informes o visualizaciones que faciliten la toma de decisiones.

Una de las bibliotecas más destacadas en este ámbito es *IfcOpenShell*, que se ha convertido en una herramienta indispensable para trabajar con archivos IFC. Esta biblioteca de código abierto es ampliamente utilizada en la comunidad de BIM por su capacidad para leer, escribir y manipular modelos BIM en formato IFC.

IfcOpenShell es una biblioteca diseñada específicamente para facilitar la interacción con archivos IFC, el estándar abierto para la interoperabilidad en la construcción digital. Esta herramienta permite a los usuarios realizar una amplia variedad de operaciones, desde la extracción de información geométrica y de atributos hasta la modificación de elementos del modelo y la validación de la estructura del archivo.

- **Extracción de información geométrica y de atributos**: IfcOpenShell permite acceder de manera eficiente a la información contenida en un archivo IFC, lo que incluye tanto datos geométricos como atributos específicos de los elementos del modelo. Esto es esencial para analizar y validar modelos de manera precisa.
- **Modificación de elementos del modelo**: Con IfcOpenShell, los usuarios pueden modificar elementos del modelo directamente, ajustando parámetros y propiedades según las necesidades del proyecto. Esto es especialmente útil para alinear los modelos con los requisitos del estándar IFC4.3.
- **Validación de la estructura y contenido del archivo IFC**: La biblioteca también incluye herramientas para verificar que la estructura del archivo IFC sea conforme con las especificaciones del estándar. Esto asegura que los modelos puedan ser utilizados de manera efectiva en un entorno colaborativo.

Figura 205 – Biblioteca IfcOpenShell (Fuente: ifcopenshell.org)

Además de IfcOpenShell, existen otras bibliotecas y frameworks que son particularmente útiles en el análisis y manipulación de datos en el contexto del BIM:

Figura 206 – Biblioteca NumPy (Fuente: numpy.org)

NumPy: Una biblioteca fundamental para el análisis y manipulación de datos numéricos. Proporciona estructuras de datos eficientes, como arreglos y matrices multidimensionales, que son esenciales para manejar grandes volúmenes de datos en aplicaciones científicas y de ingeniería. NumPy también incluye funciones matemáticas avanzadas que permiten realizar operaciones complejas de manera rápida y eficiente.

Figura 207 – Biblioteca Pandas (Fuente: pandas.pydata.org)

Pandas: Consiste en la biblioteca más poderosa para la manipulación y análisis de datos tabulares. Ofrece estructuras de datos como *DataFrames*, que facilitan la organización, filtrado, y análisis de datos en formato de tabla. Pandas es particularmente útil para trabajar con datos heterogéneos y realizar operaciones como la limpieza de datos, la agregación de estadísticas, y la manipulación de series temporales.

Figura 208 – Biblioteca Matplotlib (Fuente: matplotlib.org)

Matplotlib: Esta biblioteca es ampliamente utilizada para la visualización de datos. Permite a los usuarios crear gráficos y visualizaciones detalladas que pueden ayudar a interpretar los resultados del análisis de modelos BIM.

Estas bibliotecas no solo complementan las capacidades de Python, sino que también potencian su uso en el análisis y gestión de modelos BIM, permitiendo a los profesionales abordar tareas complejas de manera más

eficiente y con mayor precisión.

ENTORNOS DE DESARROLLO Y HERRAMIENTAS

El entorno de desarrollo es un componente crucial en la programación, ya que influye directamente en la eficiencia y la calidad del código que se produce. En el contexto de la programación aplicada al presente estudio, los entornos de desarrollo disponibles y las herramientas asociadas han sido seleccionados cuidadosamente para optimizar el flujo de trabajo con los modelos y las librerías que se pretenden usar garantizando así una correcta integración.

Tradicionalmente, los desarrolladores han utilizado entornos de desarrollo integrado (IDE), como *Visual Studio Code* o *PyCharm*, para escribir, depurar y ejecutar código. Estos IDEs ofrecen potentes características, como la depuración paso a paso, la integración con sistemas de control de versiones y el soporte para múltiples lenguajes de programación. En un entorno de desarrollo convencional, el flujo de trabajo generalmente consiste en escribir el código completo de un programa, ejecutarlo en su totalidad y luego revisar los resultados. Sin embargo, este enfoque puede presentar limitaciones cuando se trata de tareas que requieren iteración rápida y experimentación, como el análisis de datos o el trabajo con modelos BIM.

Figura 209 - Entornos de desarrollo integrado: PyCharm, Visual Studio Code

En un IDE tradicional, cada vez que se desea probar una modificación, es necesario ejecutar todo el código, lo cual puede ser ineficiente, especialmente cuando se trabaja con grandes volúmenes de datos o modelos complejos. Además, los IDEs no están diseñados para integrar documentación y análisis en un solo flujo de trabajo, lo que puede dificultar la trazabilidad y la comprensión de los procesos utilizados, sino para escribir, compilar y depurar código entre otros.

Figura 210 – Compañía responsable del desarrollo de los JupyterNotebooks (Fuente: jupyter.org)

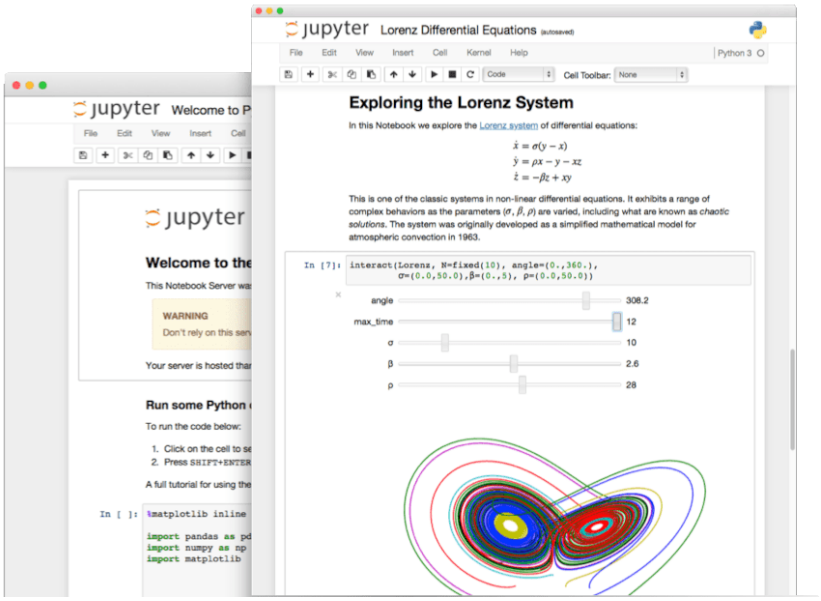
Es en este contexto donde *Jupyter Notebooks* ofrece una solución innovadora que ha ganado popularidad entre los profesionales de *Data Analytics* y, en este caso, para el análisis de modelos IFC mediante Python. A diferencia de los IDEs convencionales, *Jupyter Notebooks* permite a los usuarios escribir y ejecutar código en pequeños bloques o "celdas". Esta capacidad para probar y ajustar el código por partes es especialmente útil cuando se trabaja con modelos BIM complejos, ya que permite experimentar con diferentes enfoques, ajustar parámetros sobre la marcha y obtener retroalimentación inmediata sin la necesidad de ejecutar todo el código de una sola vez.

Jupyter Notebooks se ha consolidado como una herramienta interactiva que permite a los desarrolladores combinar código ejecutable, visualizaciones y texto narrativo en un solo documento. Esta integración es particularmente útil en el análisis de modelos BIM, donde la documentación y el análisis deben ir de la mano para asegurar la precisión y claridad en cada etapa del proyecto.

Una de las mayores ventajas de Jupyter Notebooks es su capacidad para facilitar el **prototipado rápido y la exploración de datos**. Los usuarios pueden escribir y ejecutar código en bloques independientes, lo que les

permite desarrollar soluciones de manera iterativa. Este enfoque no solo agiliza el proceso de desarrollo, sino que también permite experimentar con diferentes enfoques para resolver problemas específicos, como ajustar parámetros en modelos IFC. Esta capacidad de iterar rápidamente es crucial en proyectos BIM, donde la flexibilidad y la capacidad de ajuste continuo son esenciales.

Figura 211 – Ejemplos de Cuadernos de código JupyterNotebook (.ipynb) (Fuente: jupyter.org)



Además, Jupyter Notebooks sobresale en la **documentación de procesos de análisis**. Cada notebook puede incluir texto explicativo y visualizaciones junto al código, lo que hace que el proceso de análisis sea más transparente y comprensible. Esto es especialmente importante en proyectos colaborativos de ingeniería civil, donde varios miembros del equipo necesitan seguir y entender no solo los resultados, sino también los métodos utilizados para llegar a ellos. La documentación integrada asegura que cualquier cambio en el modelo o en el código sea claramente registrado y accesible para todos los involucrados.

Finalmente, Jupyter Notebooks es una herramienta poderosa para la **creación de informes interactivos**. A diferencia de los informes estáticos tradicionales, los notebooks permiten a los usuarios interactuar con los datos subyacentes, explorar diferentes escenarios y visualizar los resultados de manera dinámica. Esta característica es particularmente valiosa ya que las decisiones a menudo dependen de la capacidad de visualizar y manipular datos en tiempo real. Al permitir esta interactividad, Jupyter Notebooks convierte los informes en herramientas dinámicas para la toma de decisiones, en lugar de simples documentos de referencia.

Google Colab: Una Potente Extensión de Jupyter Notebooks en la Nube

Google Colab, o *Google Collaboratory*, es una plataforma basada en la nube que extiende las capacidades de Jupyter Notebooks, proporcionando un entorno en línea donde los usuarios pueden ejecutar, compartir y colaborar en notebooks sin necesidad de instalar software localmente. Esta herramienta no solo conserva todas las ventajas de Jupyter Notebooks, como la capacidad de ejecutar código en bloques, documentar procesos y crear informes interactivos, sino que también introduce características adicionales que la convierten en una opción ideal para proyectos de análisis de modelos BIM, especialmente en un contexto de investigación.

Figura 212 – Google Colab (Fuente: Google)

Una de las mayores ventajas de Google Colab es su **acceso gratuito a recursos de computación avanzados**, incluyendo unidades de procesamiento gráfico (GPUs). Este acceso es particularmente relevante cuando se

trabaja con modelos BIM complejos o grandes volúmenes de datos, que requieren un poder de cómputo significativo para ejecutar simulaciones, realizar análisis y generar visualizaciones en 3D. En un entorno de investigación, donde los recursos de hardware pueden ser limitados, Colab ofrece una solución escalable y accesible para llevar a cabo tareas computacionalmente intensivas sin necesidad de invertir en infraestructura costosa.

Además, Google Colab facilita la **colaboración en tiempo real**, lo que es esencial en proyectos de ingeniería civil y BIM que suelen involucrar a múltiples profesionales trabajando en equipo. Varios usuarios pueden acceder y trabajar en el mismo notebook simultáneamente, revisando, editando y ejecutando código de forma colaborativa. Esta capacidad de colaboración en tiempo real no solo mejora la eficiencia del equipo, sino que también asegura que todos los miembros estén alineados y puedan contribuir de manera equitativa al desarrollo del proyecto. En investigaciones donde es necesario coordinar esfuerzos entre diferentes disciplinas o ubicaciones geográficas, Google Colab se convierte en una herramienta invaluable para mantener la coherencia y la calidad del trabajo.

La **integración con Google Drive** es otra característica destacada de Google Colab, que simplifica el almacenamiento, organización y acceso a los notebooks y datos asociados al proyecto. Esta integración permite a los usuarios guardar automáticamente su trabajo en la nube, asegurando que todos los archivos estén disponibles y respaldados de manera segura. Además, la posibilidad de compartir archivos directamente desde Google Drive facilita la distribución de notebooks entre los miembros del equipo o con otros colaboradores externos, sin la necesidad de manejar complicados procesos de exportación e importación de datos.

En relación con lo anteriormente explicado sobre Jupyter Notebooks, Google Colab amplía estas capacidades al proporcionar un entorno accesible y colaborativo que es ideal para nuestra investigación. Al combinar la interactividad y flexibilidad de Jupyter Notebooks con la escalabilidad y accesibilidad de la computación en la nube, Google Colab se posiciona como una herramienta excepcionalmente poderosa para el análisis de modelos IFC mediante Python en un contexto de investigación.

Visual Studio Code: Una Herramienta Versátil para la Programación y la Gestión de Jupyter Notebooks

VSCode es un editor de código fuente que ha ganado una inmensa popularidad entre los desarrolladores de software debido a su potencia, flexibilidad y personalización. Este entorno de desarrollo integrado (IDE) no solo es altamente adaptable a diferentes lenguajes de programación, sino que también ofrece una amplia gama de extensiones que permiten a los usuarios personalizar su experiencia de desarrollo de acuerdo con las necesidades específicas de sus proyectos. Para el presente análisis de ficheros IFC mediante la programación en Python, VSCode se presenta como una opción sólida, especialmente cuando se trata de gestionar y ejecutar Jupyter Notebooks en un entorno local.

VSCode soporta múltiples lenguajes de programación, lo que permite a los desarrolladores trabajar con una variedad de tecnologías dentro de un solo entorno. Esto es especialmente útil en proyectos complejos que pueden involucrar diferentes lenguajes y herramientas. La capacidad de VSCode para manejar Python, el lenguaje preferido para la manipulación de modelos IFC en el contexto de BIM, junto con otros lenguajes como JavaScript y C++, lo convierte en un entorno ideal para desarrolladores que buscan integrar diversas tecnologías en su flujo de trabajo.

Figura 213 – Uso de Jupyter Notebooks dentro del IDE VSCode (Fuente: Elaboración Propia)

Una de las características más destacadas de VSCode es su **amplia gama de extensiones**, que mejoran la productividad y facilitan la personalización del entorno. Para los desarrolladores que trabajan con Jupyter Notebooks, la extensión "Python" de Microsoft para VSCode ofrece un soporte completo para la creación y ejecución de notebooks directamente dentro del IDE. Esto permite a los usuarios beneficiarse de todas las ventajas de Jupyter Notebooks, como la ejecución de código en bloques, la integración de visualizaciones y la

documentación en línea, mientras aprovechan las potentes herramientas de desarrollo de VSCode. Esto significa que los desarrolladores pueden combinar la interactividad de los notebooks con las capacidades avanzadas de edición, depuración y control de versiones que ofrece VSCode.

El soporte de VSCode para **capacidad de depuración integrada** es otro aspecto clave que lo distingue de otros editores o entornos de desarrollo. La depuración es esencial para el desarrollo de software de alta calidad, y VSCode proporciona herramientas que permiten a los desarrolladores identificar y resolver errores en el código de manera eficiente. En el contexto del análisis de modelos BIM y la programación en Python, donde la precisión es crítica, la capacidad de depurar código de manera efectiva dentro del mismo entorno en el que se desarrollan los notebooks es una ventaja significativa.

Además, la **integración con sistemas de control de versiones como Git** es una de las fortalezas de VSCode que resulta particularmente útil en proyectos colaborativos y de investigación. Git permite a los desarrolladores gestionar el control de versiones directamente desde el editor, facilitando el seguimiento de cambios, la colaboración en equipo y la gestión de diferentes ramas del proyecto. Esta integración es crucial para proyectos de ingeniería civil que requieren un control riguroso de versiones y un historial de cambios claro, especialmente cuando se trabaja en equipos distribuidos.

Sin embargo, a pesar de sus numerosas ventajas, VSCode también tiene algunas limitaciones en comparación con plataformas como Google Colab. Mientras que Google Colab ofrece acceso a recursos de computación en la nube, lo que es ideal para tareas intensivas en recursos como la ejecución de modelos grandes o complejos, VSCode depende del hardware local del usuario. Esto significa que los proyectos que requieren un alto rendimiento computacional pueden verse limitados por la capacidad del hardware del usuario. Además, aunque VSCode facilita la ejecución de notebooks Jupyter en un entorno local, no proporciona la misma facilidad de colaboración en tiempo real que Google Colab, donde varios usuarios pueden trabajar simultáneamente en el mismo notebook.

COMPLEMENTOS Y EXTENSIONES (ADD-ONS)

Los complementos y extensiones son herramientas adicionales que se integran con programas existentes para ampliar su funcionalidad. En el contexto del modelado BIM y la programación, estas extensiones permiten a los usuarios realizar tareas que no serían posibles con las funcionalidades básicas del software.

BlenderBIM es un complemento desarrollado mediante la librería *IfcOpenshell* para Blender, un software de modelado 3D de código abierto ampliamente relacionado con Python. Este complemento ha sido desarrollado para facilitar el trabajo con modelos BIM en formato IFC, permitiendo una mayor flexibilidad y control sobre los datos y geometría de los ficheros que se tratan.

- **Importar y exportar archivos IFC**: *BlenderBIM* permite la importación y exportación de archivos IFC, lo que facilita la integración de modelos BIM con otros sistemas y plataformas.
- **Crear y editar modelos BIM directamente en Blender**: Este complemento proporciona herramientas avanzadas para la edición de modelos BIM, permitiendo a los usuarios realizar modificaciones detalladas y específicas que se alineen con los estándares de la industria.
- **Realizar análisis y validaciones de modelos IFC**: *BlenderBIM* incluye funcionalidades para analizar y validar modelos, asegurando que cumplen con los requisitos del estándar IFC4.3 y que son adecuados para su uso en proyectos de infraestructura.

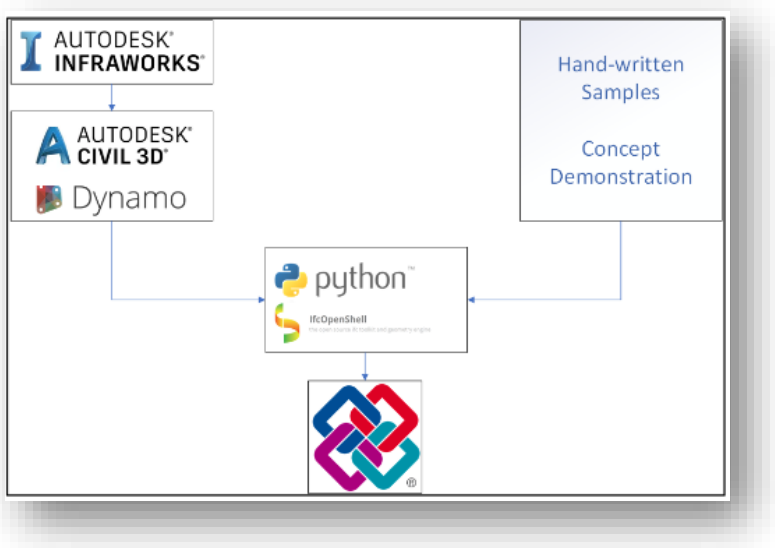
La combinación de estas herramientas y tecnologías proporciona un ecosistema robusto para el desarrollo, análisis y validación de modelos BIM en formato IFC. Al dominar estas herramientas, los profesionales de la ingeniería civil pueden ser capaces de mejorar significativamente la precisión, eficiencia y calidad de sus proyectos, asegurando que estos cumplen con los estándares más altos de la industria.

A lo largo de este capítulo, se exploran las metodologías propuestas a partir de la programación y cómo estas herramientas se aplican en la práctica (ver capítulo 6) para abordar los desafíos específicos del modelado de infraestructuras civiles, proporcionando ejemplos concretos y recomendaciones basadas en las mejores prácticas. Este enfoque permitirá a los profesionales no solo comprender los conceptos básicos, sino también aplicar estas técnicas en sus propios proyectos, mejorando así la calidad y eficacia de sus entregables.

7.2 IfcOpenShell y BlenderBIM

En el ámbito del modelado BIM y, más específicamente, en la implementación del estándar IFC4.3, herramientas como *IfcOpenShell* y BlenderBIM han desempeñado un papel crucial en la evolución de los flujos de trabajo dentro de la ingeniería civil. A lo largo de este trabajo, ya se ha introducido y discutido el valor de estas herramientas en términos de su capacidad para manejar, manipular y validar modelos IFC. En este subcapítulo, se abordará cómo estas herramientas se integran de manera práctica y efectiva dentro del contexto de un proyecto real, como es el desarrollo de infraestructuras viales, y en particular, en la colaboración innovadora entre Autodesk y la Universidad Técnica de Múnich (TUM).

Figura 214 – Flujo de trabajo colaborativo Autodesk y TUM para aspectos del desarrollo de IFC4.3 (Fuente: Building Smart)



La colaboración entre Autodesk y TUM se enmarca en un esfuerzo por avanzar en la adopción de estándares abiertos como IFC4.3, con un enfoque particular en mejorar la precisión y la interoperabilidad en proyectos de infraestructuras viales. Esta colaboración permitió el desarrollo de un prototipo de aplicación que utiliza IfcOpenShell para alinear los modelos BIM con los requisitos específicos del estándar IFC4.3, particularmente en el desarrollo de *"IfcRoad"* para infraestructuras de carreteras. Esta integración de *IfcOpenShell* en el flujo de trabajo muestra cómo esta librería, potenciada por Python, resultó fundamental para manejar y validar los datos complejos necesarios para garantizar la interoperabilidad y precisión de los modelos desarrollados en plataformas como Autodesk Infraworks y Civil 3D.

La importancia de este prototipo radica en su capacidad para integrar diversas herramientas y tecnologías, asegurando que los modelos no solo cumplan con los altos estándares de calidad, sino que también sean plenamente interoperables entre diferentes plataformas y sistemas utilizados en la construcción.

El diagrama de flujo de trabajo, presentado anteriormente, ilustra cómo ***IfcOpenShell***, en combinación con ***Python***, facilita la manipulación de datos IFC para garantizar que los modelos se adhieran a los estrictos estándares definidos en IFC4.3. Esto muestra el uso de la librería para **entornos profesionales** de desarrollo del estándar IFC entre los distintos participantes y justifica la necesidad del conocimiento de esta para todo aquel que desee profundizar en el uso y adaptación del estándar a las necesidades de cada casuística.

En este subcapítulo, se explora cómo *IfcOpenShell* se emplea para la manipulación y validación de datos en modelos IFC. Se analizará cómo, mediante el uso de Python, se pueden automatizar procesos de revisión, de los datos intrínsecos al fichero IFC, complejos que facilitan el trabajo con grandes volúmenes de datos y modelos detallados, asegurando que estos sean consistentes con los estándares y fáciles de gestionar en un entorno colaborativo. *IfcOpenShell*, al integrarse con entornos de desarrollo como *Jupyter Notebooks*,

permite crear scripts que pueden iterar sobre un modelo BIM, validando y ajustando los elementos según sea necesario. Este enfoque no solo mejora la eficiencia, sino que también asegura que los modelos se mantengan alineados con las especificaciones técnicas a lo largo de todo el ciclo de vida del proyecto.

Asimismo, se examinará el papel de *BlenderBIM* como una herramienta clave para la edición y validación de modelos IFC dentro de Blender. Aunque Blender es ampliamente conocido como un software de modelado 3D, su extensión BlenderBIM permite trabajar directamente con modelos IFC, ofreciendo una flexibilidad que otras plataformas propietarias no siempre proporcionan. En este subcapítulo, se profundizará en cómo *BlenderBIM* se utilizó en la colaboración entre Autodesk y TUM para realizar modificaciones precisas en modelos BIM, permitiendo que los modelos se adapten rápidamente a las necesidades específicas del proyecto.

Además, se explicará cómo la integración de Python en *BlenderBIM* facilita la automatización de tareas rutinarias, lo que no solo reduce el tiempo y esfuerzo necesarios para realizar ajustes, sino que también mejora la precisión al minimizar la intervención manual. Se presentarán ejemplos concretos de scripts y flujos de trabajo automatizados para ilustrar cómo estas herramientas pueden aplicarse en un entorno real, haciendo énfasis en su utilidad en proyectos de infraestructuras viales.

Finalmente, este subcapítulo concluirá con una reflexión sobre la importancia de la adopción de estándares abiertos como IFC4.3 y cómo *IfcOpenShell* y *BlenderBIM*, al integrarse en el flujo de trabajo BIM, contribuyen a una mayor interoperabilidad y precisión en la ingeniería civil. La colaboración entre Autodesk y TUM servirá como un caso de estudio ejemplar que demuestra cómo estas herramientas pueden elevar los estándares de calidad y eficiencia en el diseño y ejecución de proyectos de infraestructura, asegurando que los modelos BIM sean robustos, precisos y fácilmente adaptables a diferentes necesidades y plataformas tecnológicas.

7.2.1 IfcOpenShell: Herramienta clave para la manipulación de archivos IFC

IfcOpenShell es una herramienta de código abierto diseñada específicamente para interactuar con archivos IFC. Desde su concepción, *IfcOpenShell* ha sido desarrollada con el objetivo de proporcionar una solución flexible y accesible para los profesionales del BIM, permitiendo la manipulación precisa de modelos digitales en un entorno que fomenta la interoperabilidad. Esta herramienta se ha convertido en un componente esencial en el ecosistema BIM debido a su capacidad para integrarse sin problemas con Python, un lenguaje de programación ampliamente utilizado en la automatización y análisis de datos en diversas industrias.

La historia de *IfcOpenShell* está marcada por su evolución continua en respuesta a las necesidades de la industria de la construcción, que ha demandado herramientas capaces de manejar archivos IFC de manera eficiente y sin depender de soluciones propietarias. Como una herramienta de código abierto, IfcOpenShell ha ganado popularidad debido a su accesibilidad y la posibilidad de personalización por parte de los usuarios, lo que permite adaptarla a las necesidades específicas de cada proyecto. Su desarrollo ha sido impulsado por una comunidad activa de usuarios y desarrolladores que han contribuido a mejorar sus funcionalidades y expandir su alcance en el mercado profesional.

El propósito de *IfcOpenShell* dentro del ecosistema BIM es claro: proporcionar una herramienta robusta y flexible para leer, escribir, modificar y validar archivos IFC. Esta capacidad es fundamental para asegurar que los modelos BIM puedan ser utilizados en un entorno colaborativo donde diferentes herramientas y plataformas deben interactuar de manera eficiente. Al ser compatible con *Python*, *IfcOpenShell* permite a los usuarios aprovechar la potencia de este lenguaje para crear scripts personalizados que automatizan tareas complejas, optimizando el flujo de trabajo y reduciendo la posibilidad de errores manuales.

FUNCIONALIDADES PRINCIPALES

IfcOpenShell ofrece un conjunto de funcionalidades clave que la convierten en una herramienta indispensable para la manipulación de archivos IFC en proyectos BIM. Estas funcionalidades incluyen:

- **Lectura y escritura de archivos IFC**: *IfcOpenShell* permite abrir, explorar y modificar archivos IFC, proporcionando acceso a todas las entidades y propiedades contenidas en el modelo. Esto es crucial

para los profesionales que necesitan realizar análisis detallados o ajustes en los modelos sin perder la integridad de los datos originales.

- **Extracción de geometrías y atributos**: Una de las capacidades más poderosas de IfcOpenShell es la extracción de datos geométricos y atributos asociados a los elementos del modelo. Esto permite a los usuarios realizar análisis específicos sobre componentes individuales del modelo, lo que es esencial en proyectos donde se requiere un alto nivel de detalle y precisión.
- **Modificación de entidades y propiedades**: IfcOpenShell no solo permite la visualización de modelos, sino también la modificación de entidades y propiedades dentro del archivo IFC. Esta funcionalidad es fundamental para adaptar los modelos a cambios en los requisitos del proyecto o para corregir errores detectados durante el proceso de validación.
- **Validación de la estructura de modelos según el estándar IFC4.3**: La validación de modelos es una tarea crítica en cualquier proyecto BIM. IfcOpenShell facilita esta tarea al permitir la verificación de que la estructura del modelo cumple con las especificaciones del estándar IFC4.3. Esto asegura que los modelos sean interoperables y puedan ser utilizados sin problemas en diferentes plataformas y por diferentes equipos de trabajo.

INTEGRACIÓN CON PYTHON

Uno de los aspectos más valiosos de IfcOpenShell es su integración con Python, que permite a los usuarios automatizar tareas repetitivas y complejas en el análisis de modelos BIM. Python, conocido por su simplicidad y poder, es ideal para la creación de scripts que pueden gestionar grandes volúmenes de datos y ejecutar procesos de análisis detallados de manera eficiente.

- **Automatización de tareas repetitivas**: Mediante la combinación de IfcOpenShell con Python, se pueden desarrollar scripts que automatizan tareas comunes en el análisis de modelos BIM, como la extracción de datos, la validación de la estructura del modelo y la generación de informes. Esta automatización no solo ahorra tiempo, sino que también reduce el riesgo de errores manuales, aumentando la precisión y consistencia de los resultados.
- **Ejemplos prácticos de automatización**: La flexibilidad de IfcOpenShell permite la creación de scripts personalizados para satisfacer las necesidades específicas de cada proyecto. Por ejemplo, se pueden desarrollar scripts que extraigan automáticamente todas las entidades de un tipo particular (como columnas o vigas) y validen sus propiedades según las especificaciones del proyecto. Otro ejemplo podría ser un script que recorra todas las relaciones espaciales del modelo para asegurar que estén correctamente definidas según el estándar IFC4.3.
- **Uso en Jupyter Notebooks**: IfcOpenShell también puede ser utilizado en combinación con Jupyter Notebooks, lo que facilita la creación de flujos de trabajo interactivos donde se puede ejecutar código Python en bloques, visualizar resultados, y documentar el proceso de análisis. Este enfoque es especialmente útil en entornos educativos o de investigación, donde se necesita una alta transparencia y la capacidad de experimentar con diferentes métodos de análisis. Los cuadernos Jupyter que se desarrollen pueden servir como material didáctico, proporcionando ejemplos concretos de cómo utilizar IfcOpenShell para analizar y manipular modelos IFC de manera efectiva.

7.2.2 Flujo de trabajo basado en Jupyter Notebooks para el análisis de modelos IFC

Como se ha abordado con anterioridad, el uso de *Jupyter Notebooks* en combinación con *IfcOpenShell* ofrece un enfoque potente y flexible para la manipulación y análisis de archivos IFC en proyectos de modelado BIM, por lo que se ha explorado su uso y aplicación para el presente trabajo. Será en esta subsección dende se expondrán las razones por las que *Jupyter Notebooks* es la herramienta ideal para trabajar con *IfcOpenShell* mediante algunos ejemplos prácticos que ilustran cómo se puede utilizar este entorno para extraer, modificar y validar datos en archivos IFC.

El entorno interactivo en el que el código puede ser ejecutado en bloques o celdas que nos permiten los *Jupyter Notebooks*, es lo particularmente ventajoso cuando se trabaja con archivos IFC, que suelen ser complejos y contener una gran cantidad de datos. Esta capacidad de segmentar el análisis en bloques permite a los usuarios

probar, ajustar y visualizar las respuestas por partes del código de manera incremental, lo que facilita la detección y corrección de los datos y errores en etapas tempranas del proceso.

Por ejemplo, al analizar un modelo BIM, podría ser necesario **extraer y visualizar ciertos datos antes de proceder a la modificación del modelo**. En lugar de ejecutar todo el script de una sola vez, *Jupyter Notebooks* permite ejecutar una celda que únicamente extrae los datos, permitiendo revisar y validar esa información antes de pasar al siguiente paso. Esta capacidad de análisis por bloques no solo optimiza el tiempo y los recursos, sino que también aumenta la precisión al permitir un enfoque iterativo y cuidadoso en cada etapa del análisis.

Otra ventaja crucial de *Jupyter Notebooks* es su capacidad para **integrar código, texto y visualizaciones** en un solo documento interactivo. Esta característica es especialmente útil en el análisis de archivos IFC, donde es importante no solo ejecutar el código, sino también documentar el proceso y los resultados obtenidos.

Al integrar explicaciones detalladas y comentarios junto con el código ejecutable, los *Jupyter Notebooks* facilitan la comprensión del flujo de trabajo por parte de otros miembros del equipo o futuros usuarios que necesiten replicar el análisis. Además, las imágenes generadas a partir de los datos extraídos se pueden incluir directamente en el notebook, lo que permite revisar los resultados en contexto y tomar decisiones informadas sin necesidad de herramientas adicionales.

La capacidad de crear informes interactivos también significa que **los *Jupyter Notebooks* pueden servir como documentación viva de un proyecto BIM**, donde el código y los resultados están siempre accesibles y actualizables. Esto es particularmente valioso en entornos colaborativos, donde la transparencia y la reproducibilidad son esenciales.

A continuación, se presenta los flujos de trabajo más básicos de uso de la herramienta *IfcOpenShell* para posteriormente, una vez entendidos los procesos básicos, mostrar las rutinas diseñades en los cuadernos de programación de *Jupyter Notebooks*. Estos ejemplos ilustrarán cómo se puede estructurar un análisis típico en un notebook, aprovechando las capacidades interactivas de *Jupyter*.

EXTRACCIÓN DE DATOS

El primer paso en el análisis de un archivo IFC suele ser la extracción de datos relevantes. Con *IfcOpenShell*, se pueden listar y revisar las entidades contenidas en el archivo de manera rápida y eficiente. El siguiente código ejemplifica cómo se puede realizar esta tarea en un *Jupyter Notebook*:

Código 01 – Fragmento de Extracción de Datos con IfcOpenShell (Lenguaje: Python)

```python
import ifcopenshell
# Cargar el archivo IFC
ifc_file = ifcopenshell.open('ruta/del/archivo.ifc')

# Listar todas las entidades del modelo
entities = ifc_file.by_type('IfcWall')
for entity in entities:
    print(f"GlobalId: {entity.GlobalId}, Name: {entity.Name}")
```

Este consiste en uno de los ejemplos más básicos de la librería, el primer paso es cargar en el programa *Ifcopenshell*. Cabe recordar que se debe concebir *Ifcopenshell* como una caja de herramientas (Biblioteca/Framework) que contiene todo lo necesario para abrir y explorar archivos IFC. Si no se importa esta herramienta, el programa no sabría cómo manejar estos archivos.

A continuación, se procede a abrir un archivo IFC especificando su ruta. Esta acción carga en el programa toda la información contenida en el archivo, que incluye detalles sobre los elementos que componen el modelo del edificio, como paredes, puertas, ventanas, entre otros.

Una vez que el archivo está abierto, se filtran las entidades para encontrar aquellas que son de tipo *"IfcWall"*. Después de identificar los muros, se itera sobre cada una de estas entidades para extraer y mostrar dos atributos clave: el *GlobalId* y el *Name*. Se recuerda que el *GlobalId* es un identificador único que distingue cada entidad IFC dentro del modelo, mientras que el *Name* es el nombre asignado a esa entidad, proporcionando una identificación más descriptiva.

MODIFICACIÓN DE DATOS

Código 02 – Fragmento de Modificación de Datos con IfcOpenShell (Lenguaje: Python)

```
# Modificar el nombre de todas las paredes
for entity in entities:
    entity.Name = "Nuevo Muro"
# Guardar los cambios en un nuevo archivo IFC
ifc_file.write('ruta/nuevo_archivo.ifc')
```

Primero, se utiliza un bucle *"for"* para recorrer todas las entidades que representan paredes en el modelo. Estas entidades fueron previamente identificadas en un paso anterior mediante la función *by_type('IfcWall')*.

Dentro de este bucle, para cada muro (o entidad que se haya filtrado), se cambia el valor del atributo *Name*. Este atributo contiene el nombre del muro, y en este caso, se modifica para que todos los muros *"IfcWall"* del modelo reciban el nombre "Nuevo Muro".

Esto se realiza de la siguiente manera:

- **Recorrido de las entidades**: Se itera con un bucle *"for"* sobre cada una de las entidades que representan paredes.
- **Modificación del nombre**: Para cada entidad, se asigna el valor "Pared Modificada" al atributo *Name*. Esto significa que, después de ejecutar el código, todas las paredes en el modelo tendrán este nuevo nombre.

Este tipo de modificación es útil cuando se necesita estandarizar nombres, aplicar una convención de nombres específica, o simplemente cuando se quiere realizar un cambio masivo en las propiedades de un conjunto de entidades, ya que al igual que se ha editado el nombre de la entidad en este caso, se pueden modificar el resto de los atributos tanto de información como geométricos, aunque este último requiere de conocimientos algo más elevados de programación.

Después de realizar las modificaciones, es importante **guardar los cambios**. Sin embargo, en lugar de sobrescribir el archivo original (lo cual podría ser arriesgado si se desea mantener una copia sin cambios), se guarda en un nuevo archivo IFC. Este paso se lleva a cabo utilizando la función *write()* proporcionada por *IfcOpenShell*. Se especifica una nueva ruta y nombre para el archivo modificado, lo que crea un archivo IFC que contiene todas las modificaciones realizadas.

VERIFICACIÓN DE LA INFORMACIÓN

Además de visualizar y modificar los datos contenidos en el IFC es posible establecer comprobaciones logísticas de las distintas entidades que componen el modelo. Es por tanto procedente dedicar un ejemplo básico de comprobación para mostrar al usuario las capacidades de la biblioteca. En este ejemplo, tomando las entidades de muros *"IfcWall"* que han sido hipotéticamente filtradas se comprueba si están acotadas espacialmente dentro de alguna entidad espacial IFC como las muchas que han sido mostradas a lo largo de este trabajo.

Este fragmento de código en Python se puede utilizar para verificar si todos los *"IfcWall"* en un modelo IFC están correctamente asociados con una estructura espacial dentro del proyecto IFC. En un modelo de edificación, las estructuras espaciales suelen representar los niveles *"IfcBuildingStore"*, o áreas específicas dentro del edificio. Como se ha tratado de ilustrar en el trabajo, es importante que cada elemento, esté asociado con una de estas estructuras espaciales para asegurar la coherencia y la organización del modelo.

El código comienza iterando sobre todas las entidades que representan los muros *"IfcWall"* en el modelo IFC. Esto se hace utilizando la función *by_type('IfcWall')*, que recupera todas las paredes presentes en el modelo. Para cada muro, el código procede a verificar si está contenida en alguna estructura espacial. Esto se realiza mediante los siguientes pasos:

- Obtener las relaciones inversas: Se utiliza la función *get_inverse(wall)* para obtener todas las relaciones en las que la pared participa. Las relaciones inversas son conexiones entre diferentes elementos del modelo, y en este caso se busca específicamente si la pared está contenida dentro de una estructura espacial.

- Inicialización de la variable bandera de verificación: Se define una variable *in_spatial_structure* y se establece en *False* inicialmente. Esta variable se usará para determinar si se encuentra una relación que indique que la pared está contenida en una estructura espacial.

- Búsqueda de la relación *IfcRelContainedInSpatialStructure*: A continuación, se itera sobre las relaciones obtenidas en el paso anterior. Si se encuentra una relación del tipo *IfcRelContainedInSpatialStructure*, significa que la pared está contenida dentro de una estructura espacial. En ese caso, la variable bandera *in_spatial_structure* se establece en *True*, y se guarda la referencia a la estructura espacial relacionada para su uso posterior.

Después de verificar cada pared, el código imprime un mensaje basado en si la pared está o no asociada con una estructura espacial:

- Si la pared está contenida en una estructura espacial: Se imprime un mensaje que indica que la pared está correctamente contenida en una estructura espacial, mostrando tanto el GlobalId de la pared como el nombre y el GlobalId de la estructura espacial.

- Si la pared no está contenida en ninguna estructura espacial: Se imprime una advertencia que indica que la pared no está asociada con ninguna estructura espacial. Esto es importante porque podría indicar un problema en el modelo que debe ser corregido.

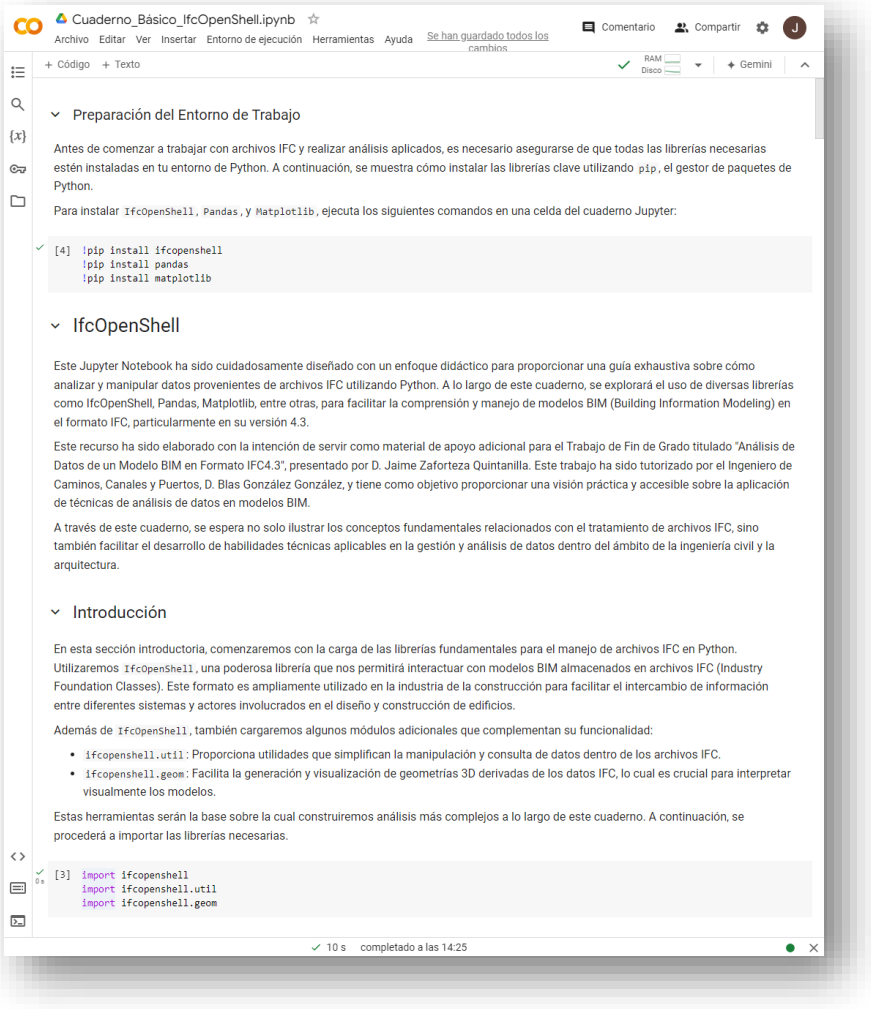
Código 03 – Fragmento de Verificación de Información con IfcOpenShell (Lenguaje: Python)

```
# Verificar que todas las paredes están contenidas en una estructura espacial
for wall in ifc_file.by_type('IfcWall'):
# Obtener las relaciones inversas (contenidos en estructuras espaciales)
    related_structures = ifc_file.get_inverse(wall)
    in_spatial_structure = False
# Buscar estructuras espaciales asociadas
    for relation in related_structures:
        if relation.is_a('IfcRelContainedInSpatialStructure'):
            in_spatial_structure = True
            spatial_structure = relation.RelatingStructure
            break
# Imprimir la estructura espacial o advertencia si no hay asociación
    if in_spatial_structure:
        print(f'El muro con GlobalId {wall.GlobalId} está contenido en la estructura espacial: {spatial_structure.Name} (GlobalId: {spatial_structure.GlobalId})')
    else:
        print(f'Advertencia: El muro con GlobalId {wall.GlobalId} no está contenido en ninguna estructura espacial.')
```

CUADERNOS JUPYTER DIDÁCTICOS

Para facilitar la aplicación de las técnicas discutidas en este capítulo a proyectos reales, se han desarrollado una serie de cuadernos Jupyter que se incluyen en los anexos de este trabajo. Estos cuadernos no solo presentan ejemplos prácticos, sino que también ofrecen explicaciones detalladas y ejercicios que permiten al lector profundizar en las funcionalidades de *IfcOpenShell*. Cada cuaderno está diseñado para ser autosuficiente, guiando al usuario por las funcionalidades básicas de análisis y automatización de procesos de los datos contenidos en modelos IFC. A partir de estos conocimientos el usuario será capaz de elaborar scripts desarrollados en complejidad para construir rutinas de automatización de datos e inclusive añadir técnicas de análisis de datos competitivas.

Figura 215 – Cuaderno Jupyter Notebook de aprendizaje IfcOpenShell desde Google Colab (Fuente: Elaboración Propia)



Estos cuadernos Jupyter definitivos están destinados a proporcionar a los profesionales del BIM y a los estudiantes de ingeniería civil una comprensión inicial práctica y aplicable del uso de *IfcOpenShell* y *Python* en la manipulación y análisis de modelos IFC. Siguiendo las instrucciones y ejemplos proporcionados, el lector podrá adquirir las habilidades necesarias para automatizar tareas comunes en el manejo de archivos IFC, mejorando así la eficiencia y la precisión en sus proyectos.

El cuaderno Jupyter completo, que detalla todas las funcionalidades de IfcOpenShell y ofrece ejemplos prácticos para la automatización de tareas en modelos IFC mediante Python, se encuentra disponible en el **Anejo 1** de este Trabajo de Fin de Grado. Este cuaderno incluye explicaciones detalladas, código comentado, y ejercicios adicionales para que el lector pueda profundizar en el uso de estas herramientas en sus propios proyectos. Se recomienda revisar el anejo para acceder a todos los recursos necesarios y aprovechar al máximo las técnicas discutidas en este capítulo.

7.2.3 BlenderBIM: Herramienta *OpenSource* para la Edición de Modelos BIM

BlenderBIM es una poderosa extensión del software de modelado 3D de código abierto **Blender**, diseñada para trabajar con archivos IFC (Industry Foundation Classes). Esta herramienta ofrece una alternativa flexible y accesible a las soluciones propietarias tradicionales en el entorno BIM, como Revit, permitiendo a los usuarios importar, editar y exportar modelos BIM en formato IFC. BlenderBIM ha ganado popularidad en la comunidad de profesionales del BIM debido a su enfoque en estándares abiertos y su capacidad para integrarse en flujos de trabajo colaborativos.

En primer lugar, **Blender** es un software de modelado 3D ampliamente utilizado en diversas industrias, que ha sido reconocido por su versatilidad, capacidad de personalización, y, lo que es más importante en el contexto que trata del openBIM, su código abierto. **BlenderBIM**, como extensión de Blender, añade funcionalidades específicas para la gestión y edición de modelos IFC, permitiendo a los usuarios manipular directamente los datos del modelo BIM sin necesidad de herramientas propietarias costosas.

BlenderBIM permite a los profesionales del sector importar modelos IFC a Blender, realizar modificaciones detalladas en la geometría, las propiedades y las relaciones espaciales del modelo, y luego exportar los cambios de nuevo a un archivo IFC. Esto es especialmente útil en proyectos donde la interoperabilidad y la adherencia a estándares abiertos son esenciales para el éxito del proyecto. Además, al ser una herramienta de código abierto, BlenderBIM ofrece una flexibilidad significativa, permitiendo a los usuarios adaptar la herramienta a sus necesidades específicas mediante scripts en Python y complementos personalizados.

FUNCIONALIDADES CLAVE

Como se puede apreciar en el capítulo 6 dedicado al caso de estudio, *BlenderBIM* ofrece una variedad de funcionalidades de edición y modelado del fichero IFC. Mediante estas funcionalidades el usuario es capaz de personalizar y generar nuevos flujos de trabajo en un entorno BIM abierto de forma nativa. Entre las distintas funcionalidades apreciadas, a lo largo del caso de estudio, se destacan:

1. Importación y exportación de archivos IFC

Una de las funcionalidades más importantes de BlenderBIM es su capacidad para importar y exportar archivos IFC de manera eficiente y precisa. Esto permite a los usuarios traer modelos BIM desde otras plataformas, como Revit, ArchiCAD o inclusive Civil3D, para los modelos de diseño BIM de infraestructuras, al entorno de Blender para su edición y análisis. La importación de archivos IFC en BlenderBIM es intuitiva y permite mantener la integridad de los datos del modelo, asegurando que toda la información, como las propiedades de los elementos y las relaciones espaciales, se preserven durante el proceso.

La capacidad de exportar archivos IFC desde BlenderBIM es igualmente crucial, ya que permite a los usuarios realizar modificaciones en Blender y luego integrar estos cambios de vuelta en el flujo de trabajo BIM general.

2. Modificación de geometrías, relaciones espaciales y propiedades del modelo BIM

BlenderBIM ofrece herramientas avanzadas para la modificación de la geometría de los modelos BIM. Los usuarios pueden editar los elementos geométricos directamente dentro de Blender, utilizando las poderosas herramientas de modelado 3D de Blender. Esto incluye la capacidad de ajustar formas, dimensiones, y relaciones entre diferentes elementos del modelo.

Además de la edición geométrica, BlenderBIM permite la modificación de relaciones espaciales y propiedades de los elementos. Por ejemplo, se pueden reasignar elementos a diferentes espacios o niveles dentro del modelo, o ajustar las propiedades de los materiales para reflejar cambios en los requisitos del proyecto. La capacidad de modificar estos aspectos del modelo BIM directamente en Blender hace que BlenderBIM sea

una herramienta muy versátil para proyectos donde la precisión y el control sobre cada detalle del modelo son cruciales.

COMPARATIVA DE BlenderBIM FRENTE A LAS ALTERNATIVAS DE PAGO

Cuando se compara **BlenderBIM** con otras soluciones propietarias como Revit, surgen varias ventajas y desventajas que son importantes de considerar:

Tabla 15 – Comparativa entre BlenderBIM y software BIM de pago como Revit (Fuente: Elaboración Propia)

	BlenderBIM	**Revit**
Flexibilidad y Personalización	Ofrece una gran flexibilidad debido a su naturaleza de código abierto. Los usuarios pueden personalizar la herramienta mediante scripts en Python y complementos, adaptándola a las necesidades específicas del proyecto	Si bien **Revit** es una herramienta muy robusta, está limitada por su arquitectura propietaria, lo que dificulta la personalización a nivel profundo sin depender de complementos aprobados por Autodesk.
Interoperabilidad Estándares Abiertos	Está completamente alineado con los principios de estándares abiertos, como IFC, lo que facilita la interoperabilidad en entornos BIM donde se utilizan múltiples herramientas y plataformas.	Aunque **Revit** soporta la exportación e importación de archivos IFC, su enfoque principal está en su propio formato de archivo propietario (.rvt), lo que a veces puede crear problemas de compatibilidad o pérdida de datos al trabajar en un entorno multi-herramienta.
Costo	Al ser una herramienta de código abierto, **BlenderBIM** es gratuito, lo que lo hace accesible para una amplia gama de usuarios, incluyendo estudiantes, pequeñas empresas, y proyectos con presupuesto limitado.	**Revit** es una herramienta costosa, lo que puede ser una barrera de entrada para algunos usuarios, especialmente aquellos que buscan una solución BIM flexible sin un gran compromiso financiero.
Comunidad y Soporte	La comunidad de **Blender** es muy activa, con numerosos foros, tutoriales y recursos disponibles. Sin embargo, debido a que es una herramienta de código abierto, el soporte técnico oficial puede ser limitado comparado con soluciones propietarias.	**Revit** ofrece un soporte técnico robusto y está respaldado por Autodesk, una de las empresas líderes en software de diseño. Esto garantiza que los usuarios tengan acceso a actualizaciones regulares y soporte profesional.

En definitiva, **BlenderBIM** se presenta como una alternativa poderosa y flexible para la edición de modelos BIM, especialmente en entornos donde la interoperabilidad y el uso de estándares abiertos son prioritarios como el que comienza a enmarcar el Plan BIM España (ver capítulo 3). Su comparación con herramientas propietarias como Revit muestra que, si bien cada una tiene sus fortalezas, BlenderBIM destaca por su flexibilidad, accesibilidad, y compromiso con los principios de código abierto. Para muchos profesionales del BIM, BlenderBIM representa una herramienta crucial que puede complementar o incluso reemplazar soluciones más tradicionales, dependiendo de los requisitos específicos del proyecto.

7.2.4 Automatización con Python en BlenderBIM

BlenderBIM, al ser una extensión de Blender, no solo permite la manipulación visual de modelos BIM, sino que también ofrece una integración robusta con Python, lo que habilita la automatización de tareas dentro del entorno de trabajo (ver Figura 197 – ***Script generado para modificaciones de las entidades del primer modelo (Fuente: Elaboración Propia)***). Esta capacidad es especialmente valiosa en proyectos de gran escala, como infraestructuras viales, donde la eficiencia y precisión en la gestión de modelos BIM son cruciales. En este subcapítulo, se expondrá cómo Python se puede utilizar en BlenderBIM para automatizar tareas comunes repetitivas, y generar así la

optimización de los flujos de trabajo garantizando la conformidad de los modelos con estándares como IFC4.3.

Compatibilidad de BlenderBIM con scripts en Python

Blender, como plataforma de modelado 3D, ha sido históricamente conocida por su capacidad de ser completamente personalizable a través de scripts en Python. BlenderBIM extiende esta capacidad a la edición y manipulación de modelos BIM, permitiendo a los usuarios escribir scripts que pueden automatizar desde tareas sencillas hasta procesos complejos en la gestión de archivos IFC.

La compatibilidad de BlenderBIM con Python se basa en la flexibilidad que ofrece el entorno de Blender para interactuar con sus datos internos y estructuras. Los scripts de Python pueden acceder directamente a las entidades de un archivo IFC importado, permitiendo realizar operaciones en masa, verificar la consistencia del modelo, o incluso generar informes automáticos sobre el estado del modelo.

Automatización de tareas comunes

Una de las principales ventajas de utilizar Python en BlenderBIM es la posibilidad de automatizar tareas que, de otro modo, consumirían mucho tiempo si se realizaran manualmente. Entre las tareas que se pueden automatizar se incluyen:

- **Creación y Modificación de Modelos**: Mediante scripts, se pueden generar elementos del modelo, asignarles propiedades y ubicarlos dentro de la estructura espacial del modelo BIM. Por ejemplo, se pueden crear varias instancias de un mismo elemento con diferentes propiedades geométricas o de material, lo cual es útil en proyectos que requieren múltiples variantes de un mismo componente constructivo.
- **Validación de Modelos**: Los scripts en Python pueden recorrer todas las entidades del modelo para verificar que cumplan con ciertas condiciones, como estar asociadas a una estructura espacial o tener materiales asignados. Esta validación automatizada es clave para asegurar que los modelos cumplen con los requisitos de calidad antes de ser utilizados en fases posteriores del proyecto.
- **Exportación Automatizada**: Después de modificar el modelo en BlenderBIM, los scripts pueden automatizar la exportación del archivo IFC, asegurando que todos los cambios se guarden correctamente y que el archivo resultante esté listo para su uso en otras herramientas BIM.

APLICACIÓN PRÁCTICA EN INFRAESTRUCTURAS: Análisis del Script Python en BlenderBIM

El script mostrado en el caso de estudio (ver Figura 197 – ***Script generado para modificaciones de las entidades del primer modelo (Fuente: Elaboración Propia)***) es un ejemplo práctico de cómo utilizar Python dentro de BlenderBIM para automatizar tareas comunes en la manipulación y modificación de modelos BIM, específicamente en el contexto de infraestructuras viales. A continuación, se desglosa y comenta cada sección del script para explicar su funcionamiento y cómo se aplica en un flujo de trabajo BIM típico.

Código 04 – Importación de librerías necesarias desde el Script de Blender (Lenguaje: Python)

```
import bpy
import ifcopenshell
import blenderbim
import ifcopenshell.api
```

bpy: Es la librería que permite interactuar con Blender a través de Python, facilitando la creación y manipulación de objetos 3D en la escena.

ifcopenshell: Es la librería principal para manejar archivos IFC dentro de Python, permitiendo abrir, leer, y modificar la información contenida en estos archivos.

blenderbim: Específicamente, se utiliza la API de BlenderBIM para extender las funcionalidades de Blender al trabajar con archivos IFC.

ifcopenshell.api: Se emplea para acceder a funciones específicas de IfcOpenShell que facilitan operaciones comunes, como la modificación y eliminación de entidades dentro del archivo IFC.

Código 05 – Cargar el modelo IFC en el Script de Blender (Lenguaje: Python)

```
# Abrimos el fichero IFC que tenemos ahora cargado desde Blender
ifc_file = blenderbim.bim.ifc.IfcStore.get_file()
```

Aquí se carga el archivo IFC que ya ha sido importado en Blender, es decir, el que se encuentra abierto en el modelo donde se está escribiendo el *script*. La función *IfcStore.get_file()* obtiene la referencia al archivo IFC actualmente en uso dentro de BlenderBIM, permitiendo interactuar directamente con él.

Código 06 – Eliminación de Entidades IfcAlignment y IfcAnnotation en el Script de Blender (Lenguaje: Python)

```
# Iteración sobre las entidades IfcAlignment dentro del fichero
for alignment in ifc_file.by_type('IfcAlignment'):
    ifcopenshell.api.root.remove_product(ifc_file, alignment)

for annotation in ifc_file.by_type('IfcAnnotation'):
    ifcopenshell.api.root.remove_product(ifc_file, annotation)
```

El script recorre todas las entidades del tipo *"IfcAlignment"* y *"IfcAnnotation"* en el archivo IFC y las elimina utilizando la función *remove_product*. Esto ha sido útil en el caso de estudio para limpiar el modelo de las anotaciones que generaba la herramienta de exportación de Civil3D automáticamente y borrar las alineaciones por defecto necesarias para exportar la obra líneas y que ya no son necesarias.

Código 07 – Filtro y Modificación de Entidades IfcBuiltElement en el Script de Blender (Lenguaje: Python)

```
for builtElement in ifc_file.by_type("IfcBuiltElement"):
    if builtElement.get_info()["type"] == "IfcBuiltElement":
        if not builtElement.Name.lower().__contains__("talud") and not
            builtElement.Name.lower().__contains__("relleno") and not
            builtElement.Name.lower().__contains__("desmonte"):

                ifcopenshell.api.root.remove_product(ifc_file,
                                                     builtElement)

        elif "desmonte" in builtElement.Name.lower():

                ifcopenshell.api.root.reassign_class(ifc_file,builtEl
                ement,"IfcEarthworksCut", "CUT")

        elif "terraplen" in builtElement.Name.lower():

                ifcopenshell.api.root.reassign_class(ifc_file,builtEl
                ement,"IfcEarthworksFill", "SLOPEFILL")

        elif "placa" in builtElement.Name.lower():

                ifcopenshell.api.root.reassign_class(ifc_file,builtEl
                ement,"IfcPlate")
```

El script recorre todas las entidades de tipo *"IfcBuiltElement"*. Primero, elimina aquellas que no contienen en su nombre las palabras clave *"talud"*, *"relleno"* o *"desmonte"*, lo que indica que solo se mantienen los elementos relacionados con movimientos de tierra que tras el análisis son los que se desean conservar, el resto eran basura que se generó redundante en el modelo, producto del proceso de exportación.

Para las entidades cuyo nombre contiene las palabras clave, el script reasigna su clase IFC utilizando *reassign_class*:

- *"Desmonte"* se reasigna como "*IfcEarthworksCut"*, indicando un corte de tierra.
- *"Terraplen"*, se reasigna como *"IfcEarthworksFill"*, que representa un relleno.
- *"Placa"* se reasigna como *"IfcPlate"*, lo que sugiere que estos elementos son placas estructurales.

Código 08 – Exportación del Archivo Modificado en el Script de Blender (Lenguaje: Python)

```
# Guardamos el Archivo modificado
output_file_path= "C:/TFG_JAIME_DATOS/4_OL/…
                  …IFC_TFG_JZQ_01_E63_AV_Doble_via_V07"
ifc_file.write(output_file_path)
```

Después de realizar las modificaciones, el script guarda el archivo IFC en una nueva ruta. Esto asegura que las modificaciones se preserven sin sobrescribir el archivo original, lo que es una buena práctica en la gestión de versiones de archivos BIM.

Código 09 – Actualización de la Escena del modelo Blender en el Script de Blender (Lenguaje: Python)

```
# Actualizar la escena de Blender después de la modificación
bpy.ops.object.select_all(action='SELECT')
bpy.ops.object.delete(use_global=False)
bpy.ops.bim.project_elements()
bpy.ops.bim.load_project(output_file_path, use_relative_path=False,…
                                        …should_start_fresh_session=True)
```

Finalmente, en esta última instancia del *script* este selecciona y elimina todos los objetos en la escena de Blender *bpy.ops.object.select_all* y *bpy.ops.object.delete* para luego cargar el modelo actualizado desde el archivo IFC modificado. Esto asegura que la vista en Blender refleje las modificaciones realizadas en el archivo IFC.

8 CONCLUSIONES Y FUTURAS LÍNEAS DE INVESTIGACIÓN

Las conclusiones de este Trabajo de Fin de Grado destacan la relevancia de las herramientas digitales avanzadas, específicamente aquellas basadas en estándares abiertos como IFC, en la gestión eficiente de infraestructuras civiles. A lo largo del desarrollo del trabajo, se ha demostrado cómo el estándar IFC está en proceso de adaptación en el sector de la ingeniería Civil e irá tomando mucha más presencia en un futuro, además de cómo el uso de herramientas como *IfcOpenShell* y *BlenderBIM* pueden integrarse efectivamente en los flujos de trabajo BIM, optimizando la manipulación, modificación y validación de modelos complejos.

Estas tecnologías no solo facilitan la interoperabilidad y la colaboración entre diferentes plataformas y equipos, sino que también permiten un enfoque más preciso y controlado en la ejecución de proyectos de infraestructuras viales que permitirán conducir al sector de la construcción pública a una optimización de gran escala.

8.1 Conclusiones

El presente trabajo de investigación ha permitido extraer una serie de conclusiones fundamentales en relación con la adopción y aplicación del estándar IFC4.3 en el contexto de la ingeniería civil, particularmente en la descripción y gestión de infraestructuras ferroviarias. Asimismo, se han analizado las implicaciones del Plan BIM España y el uso de software BIM y lenguajes de programación en la mejora de la interoperabilidad y la calidad de los modelos generados.

1. El estándar IFC 4x3 cumple con los requisitos mínimos para describir una obra de infraestructuras civiles como es la de una línea ferroviaria.

El análisis realizado ha confirmado que el estándar IFC4.3 es capaz de satisfacer los requisitos esenciales para la representación de infraestructuras civiles, como las líneas ferroviarias. Las novedades introducidas en el esquema, especialmente el conjunto de estructuras espaciales, han demostrado ser acertadas, facilitando una mejor gestión y organización de los modelos BIM. Estas estructuras espaciales permiten una segmentación más clara y precisa de los elementos del modelo, lo que a su vez mejora la capacidad de gestión y manipulación de los datos. Este avance es particularmente significativo en proyectos de gran envergadura y complejidad, donde la precisión y claridad en la organización de la información son cruciales.

2. En España, la demanda de uno de los estándares Open BIM ha aumentado considerablemente en consecuencia del Plan BIM España.

El Plan BIM España, que promueve la adopción progresiva de la metodología BIM en la contratación pública, tendrá un impacto notable en la demanda de estándares Open BIM, como el IFC. Aunque aún no está claro si esta demanda resultará en la generación de modelos IFC de alta calidad informativa, es innegable que la cantidad de modelos a exportar crecerá sustancialmente. Esto plantea un reto significativo para los profesionales y empresas involucradas, que deberán asegurar que los modelos generados cumplan con los requisitos de calidad necesarios para ser útiles en un entorno colaborativo y de largo plazo.

3. El software de diseño BIM revisados, como Civil 3D, son plenamente capaces de efectuar diseños elaborados de infraestructuras, pero necesitan no solo mejorar sino elaborar procesos de exportación OpenBIM para ingenieros civiles que no requieran de alto conocimiento informático.

El software BIM actual, como Civil 3D, han demostrado ser herramientas poderosas para el diseño de infraestructuras complejas. Sin embargo, la investigación ha revelado que estos programas aún presentan limitaciones significativas en el proceso de exportación a formatos Open BIM, como el IFC. A menudo se pierde información crítica durante el proceso de exportación, lo que reduce la utilidad de los modelos generados para su posterior uso en otros entornos de software. Esta situación subraya la necesidad de mejorar las capacidades de exportación de estos programas para asegurar que la información se mantenga completa y precisa en el formato IFC, facilitando así la interoperabilidad y el uso efectivo de los modelos en diferentes fases del proyecto.

4. Las herramientas de modelado en IFC nativo se encuentran en continuo desarrollo y, aun así, muestran ser los únicos capaces de elaborar modelos en IFC 4.3, dado que la mayoría de las potentes herramientas de diseño BIM actuales no lo hacen.

Aunque los modelos IFC nativos están aún en una fase temprana de desarrollo, la investigación ha mostrado que ya son herramientas altamente útiles en el ámbito de la ingeniería civil. Estos modelos permiten una representación precisa de las infraestructuras y facilitan la interoperabilidad entre diferentes plataformas y actores del proyecto. A medida que se desarrollen más capacidades y se generen scripts personalizados para el modelado geométrico de infraestructuras, es probable que estos modelos adquieran una presencia cada vez mayor en la industria. La evolución de los modelos IFC nativos permitirá a los profesionales adaptarse mejor a los requerimientos específicos de sus proyectos, optimizando tanto el proceso de diseño como el de ejecución.

5. El uso de lenguajes de programación como Python permite a los ingenieros civiles, conocer, entender e interpretar los datos de los modelos BIM para efectuar análisis realmente interesantes en un futuro.

Finalmente, el uso de lenguajes de programación como Python ha demostrado ser una herramienta valiosa para el análisis y la interpretación de los datos contenidos en los modelos BIM. Python no solo facilita la manipulación de estos datos, sino que también permite realizar análisis más profundos y personalizados, mejorando la capacidad de los profesionales para interpretar y utilizar la información de los modelos en la toma de decisiones. En el futuro, la integración de Python en los flujos de trabajo BIM puede mejorar significativamente la capacidad de los participantes del proyecto para comprender y utilizar los modelos de manera más efectiva, potenciando así la calidad y precisión de los proyectos de ingeniería civil.

Estas conclusiones subrayan la importancia de continuar desarrollando y adoptando estándares abiertos y herramientas avanzadas de software y programación en el sector de la ingeniería civil, con el fin de mejorar la interoperabilidad, la calidad de los modelos BIM y la eficacia general de los proyectos de infraestructura.

8.2 Futuras Líneas de Investigación

El presente trabajo de fin de grado ha explorado varios aspectos clave del estándar OpenBIM, IFC en particular del esquema IFC4.3, aplicados a infraestructuras civiles. Sin embargo, existen áreas que, debido a la amplitud del tema, se han abordado de manera superficial o se han omitido por cuestiones de alcance. Estas áreas ofrecen un amplio campo para futuras investigaciones que podrían profundizar y ampliar los conocimientos en este campo. A continuación, se proponen varias líneas de investigación que podrían ser de gran valor para la comunidad académica y profesional.

1. Profundización en las Relaciones y Propiedades de las Entidades IFC

Uno de los pilares fundamentales del esquema IFC son las relaciones y las propiedades de las entidades, aspectos que han sido tratados de manera limitada en este trabajo. Las relaciones entre entidades y las propiedades que las describen son cruciales para la correcta estructuración y funcionalidad de los modelos BIM. En futuras investigaciones, sería de gran interés profundizar en cómo se estructuran estas entidades de relaciones y cómo interactúan con las entidades de propiedades y objetos. Este enfoque permitiría entender mejor cómo se establecen las conexiones entre los distintos objetos que componen un modelo, lo cual es esencial para garantizar la interoperabilidad y la coherencia de la información en proyectos complejos. Un análisis detallado de estas estructuras podría facilitar la optimización de los modelos IFC y mejorar la precisión y utilidad de los datos para los distintos actores involucrados en el proceso de construcción.

2. Investigación en el Nuevo Dominio de Puertos y Canales

El presente trabajo se ha centrado en obras lineales, principalmente en infraestructuras como carreteras y ferrocarriles además del ámbito común a las infraestructuras. No obstante, el ámbito de las infraestructuras marítimas introducido también en IFC4.3, como puertos y canales, ha sido omitido para simplificar el alcance de esta investigación. Se recomienda que futuras investigaciones exploren en profundidad el dominio de puertos y canales dentro del estándar IFC. Este enfoque permitiría desarrollar modelos BIM

más completos y precisos para estas infraestructuras, mejorando la planificación, gestión y mantenimiento de proyectos en este ámbito como los puertos. Además, estudiar cómo se puede adaptar el esquema IFC para representar de manera efectiva las características específicas de las infraestructuras marítimas sería un avance significativo en la aplicación de BIM a proyectos de gran escala en este sector.

3. Estrategias de modelado nativo en BlenderBIM para Infraestructuras Civiles.

El modelado nativo en IFC utilizando herramientas de OpenBIM, como BlenderBIM, representa una prometedora área de investigación con una clara proyección futura. Como se discutió en las conclusiones, estas herramientas OpenBIM están destinadas a adquirir una relevancia creciente en la industria AEC (Arquitectura, Ingeniería y Construcción). Se recomienda encarecidamente investigar cómo se puede utilizar BlenderBIM para el modelado nativo de infraestructuras, explorando no solo las capacidades actuales de la herramienta, sino también cómo se pueden desarrollar y optimizar scripts para mejorar el modelado geométrico y la integración de datos complejos en los modelos IFC.

Además, investigar cómo estas herramientas pueden incorporar dimensiones adicionales, como la 4D (planificación) y 5D (costos), dentro del esquema IFC, que es totalmente posible y el esquema dispone de las entidades dispuestas para ello, podría transformar significativamente la forma en que se gestionan y ejecutan los proyectos de infraestructura. Este enfoque permitiría no solo la visualización y simulación temporal de los proyectos, sino también una mejor integración de los aspectos financieros y de recursos, ofreciendo un modelo más integral y útil para todas las fases del ciclo de vida de una infraestructura que contiene toda la información como una base de datos.

4. Análisis de la Información de los Modelos BIM mediante Técnicas de Programación y Análisis de Datos

Un área clave que merece una investigación más profunda es el análisis exhaustivo de la información contenida en los modelos BIM utilizando técnicas de programación y análisis de datos. El uso de lenguajes de programación como Python y las técnicas de análisis presentadas en esta investigación han demostrado ser herramientas poderosas para explorar y entender la riqueza de datos que los modelos BIM pueden ofrecer. El objeto inicial de este trabajo era abordar esta cuestión en profundidad, no obstante, la imposibilidad de encontrar modelos abiertos de infraestructuras correctamente ejecutados y enriquecidos de información que sea posible analizar ocasionó un cambio en el objeto del trabajo, teniendo que establecer primero las bases metodológicas de todo lo desarrollado en el presente trabajo. .

En futuras investigaciones, sería valioso profundizar en cómo estos lenguajes permiten a los usuarios realizar estudios detallados de los modelos, extrayendo y analizando información que podría pasar desapercibida con métodos tradicionales. Esto no solo mejoraría la capacidad de interpretación de los modelos, sino que también facilitaría la toma de decisiones informadas durante las distintas fases de un proyecto. Las implicaciones de este enfoque son significativas, ya que proporcionan un mayor control sobre la calidad de los modelos y permiten una mejor gestión de los recursos y tiempos en proyectos de infraestructura.

9 REFERENCIAS

[1] Asociación Española de Normalización, "UNE-EN ISO 16739-1 2024," 2024.

[2] M. Bekboliev, "Infrastructure Domain Roadmap," 2018.

[3] buildingSmart International, "The buildingSMART Process." Accessed: Aug. 05, 2024. [Online]. Available: https://www.buildingsmart.org/about/bsi-process/#:~:text=The%20buildingSMART%20standards%20development%20process,user%20needs%20and%20standard%20concept.

[4] R. Bergs, P. Jackson, C. Castaing, and J. Plume, "buildingSMART InfraRoom Forethought Whitepaper InfraRoom Roadmap Working Group."

[5] Comisión Interministerial BIM, "PLAN BIM en la contratación pública," 2023.

[6] M. Baldwin, *BIM Manager*. Anaya, 2019.

[7] Comisión Interministerial BIM, "Observatorio de la Comisión Interministerial BIM." Accessed: Mar. 17, 2024. [Online]. Available: https://cibim.mitma.es/observatorio-cibim

[8] Comisión Interministerial BIM, "Análisis de la Inclusión de Requisitos BIM en la Licitación Pública Española, Informe 24 - Cuarto Trimestre 2023," 2023.

[9] Comisión Interministerial BIM, "PLAN BIM en la contratación pública," 2023.

[10] B. International, "BuildingSmart Standard, Industry Foundation Classes (IFC)." Accessed: Mar. 01, 2024. [Online]. Available: https://technical.buildingsmart.org/standards/ifc/

[11] A. Española, "Norma Española en la gestión de inmuebles mediante IFC (Industry Foundation Classes). Parte 1 : Esquema de datos (ISO 16739-1 : 2018) (Ratificada por la Asociación Española de Normalización en abril de 2020 .)," 2020.

[12] B. International, "BuildingSmart Standar, IFC Formats." Accessed: Mar. 02, 2024. [Online]. Available: https://technical.buildingsmart.org/standards/ifc/ifc-formats/

[13] B. International, "IFC 4.3 ADD2 Official Documentation." Accessed: Mar. 01, 2024. [Online]. Available: https://standards.buildingsmart.org/IFC/RELEASE/IFC4_3/

[14] M. Pszczolka, "Spatial Breakdown Structure in IFC 4.3." Accessed: Mar. 03, 2024. [Online]. Available: https://bimcorner.com/spatial-breakdown-structure-in-ifc-4-3/

[15] Wikipedia, "Estructura de descomposición del trabajo (EDT)." Accessed: Mar. 03, 2024. [Online]. Available: https://es.wikipedia.org/wiki/Estructura_de_descomposición_del_trabajo

[16] H. Moon *et al.*, "BuildingSmart bSI IFC Road Conceptual Model Report UML Model Report for Road Elements," 2020.

[17] B. International, "Documentación Oficial IFC4X3_ADD2 - IfcSpatialStructureElement." Accessed: Mar. 03, 2023. [Online]. Available: https://standards.buildingsmart.org/IFC/RELEASE/IFC4_3/HTML/lexical/IfcSpatialStructureElement.htm

[18] BIM Corner, "¿Qué puede aportar IFC4 a la Industria?" Accessed: Apr. 26, 2024. [Online]. Available: https://bimcorner.com/what-can-ifc4-bring-to-the-industry/

[19] A. El-Amraoui-Farssi, V. Gomez-Jauregui, C. Manchado, and C. Otero, "IFC for Infrastructures: New Open Standards for Intelligent Data BT - Advances in Design Engineering II," F. Cavas Martínez, G. Peris-Fajarnes, P. Morer Camo, I. Lengua Lengua, and B. Defez García, Eds., Cham: Springer International Publishing, 2022, pp. 38–45.

[20] B. International, "Anejo F - IFC4.3 ADD2 Documentación Oficial." Accessed: Apr. 25, 2024. [Online]. Available: https://standards.buildingsmart.org/IFC/RELEASE/IFC4_3/HTML/annex-f.html

[21] A. Colegio de Ingenieros de Caminos, Canales y Puertos, “La Ingeniería Civil en España.” Accessed: Mar. 19, 2024. [Online]. Available: https://caminosandalucia.es/wp-content/uploads/2018/08/20180829_LaRegulacionIngenieriaCivilenEspana.pdf

[22] Colegio de Ingenieros de Obras Públicas, “Portal Web del Colegio de Ingenieros de Obras Públicas.” Accessed: Mar. 19, 2024. [Online]. Available: https://ingenieros-civiles.es/

[23] C. U. Ab, M. Weise, D. Peel, V. Ruby-lewis, and K. Moss, “Building Smart bSI UML Model Report - Part 1 - Harmonised UML Report – Introduction and Background,” pp. 1–159, 2020.

[24] H. Moon *et al.*, “BuildingSmart bSI IFC Road Conceptual Model Report UML Model Report for Road Elements, Annex I – Example instance diagrams,” 2020.

[25] B. International, “IFC 4.3 ADD2 Official Documentation.” Accessed: Mar. 01, 2024. [Online]. Available: https://standards.buildingsmart.org/IFC/RELEASE/IFC4_3/

[26] B. International, “IFC 4.2 Official Documentation.” Accessed: Mar. 05, 2024. [Online]. Available: https://standards.buildingsmart.org/IFC/DEV/IFC4_2/FINAL/HTML/

[27] P. Benning *et al.*, “BuildingSmart bSI Bridge Fast Track Project - Conceptual Model,” pp. 1–71, 2018.

[28] B. International, “Participantes en la certificación IFC.” Accessed: Apr. 26, 2024. [Online]. Available: https://technical.buildingsmart.org/services/certification/ifc-certification-participants/

[29] “Portal Web del estándar LandXML.”

[30] BlenderBIM, “BlenderBIM Extension page.”

[31] Euskal Trenbide Sarea, “Tutorial Blender BIM - Manual BIM de ETS,” 2024.

[32] Autodesk, “Portal de descargas Autodesk.” Accessed: Mar. 14, 2024. [Online]. Available: https://manage.autodesk.com/products

[33] Autodesk, “Portal de Ayuda Autodesk - Civil 3D.” Accessed: Mar. 14, 2024. [Online]. Available: https://help.autodesk.com/view/CIV3D/2024/ESP/

[34] Asociación Española de Normalización y Certificación, “ISO 19650-1:2018 - Organización y digitalización de la información en obras de edificación e ingeniería civil que utilizan BIM - Parte 1: Conceptos y principios,” *UNE, Normalización Española*, 2019.

[35] D. P. Viera, B. González González Cotutor, and F. G. Romero, “Trabajo Fin de Máster MODELADO BIM DE UNA LÍNEA FERROVIARIA DE ALTA VELOCIDAD CON DYNAMO PARA DIRECCIÓN DE OBRA.”

[36] L. Ramos, D. Tutores, B. González González, F. García, R. Copropietarios, and D. Trabajo, “BIM GEOTECNICO. LA CONSTRUCCION DIGITAL DEL TERRENO CON UNIDADES GEOTECNICAS.”

[37] J. M. Rivera Zafra, E. J. Galeote Gallardo, and A. Sanchez Lopez, “PPTP de la OBRA CIVIL Y SUPERESTRUCTURA DE LA METROPOLITANO DE GRANADA. TRAMO ARMILLA-CHURRIANA DE LA VEGA,” 2022. [Online]. Available: https://ws050.juntadeandalucia.es/verificarFirma

[38] Agencia de Obra Pública de la Junta de Andalucía, “Portal Web AOPJA - Implantación de la metodología BIM.”

[39] Autodesk Interoperability Tools, “Overview of the Standardized Data Tool for Civil 3D - Webinar.”

[40] Wikipedia, “Expresión regular (REGEX) - Wikipedia.”

[41] BIMvision, “Portal Web del visualizador BIMvision.”

ANEJOS

ANEJO 1º
CUADERNO JUPYTER IFCOPENSHELL

Preparación del Entorno de Trabajo

Antes de comenzar a trabajar con archivos IFC y realizar análisis aplicados, es necesario asegurarse de que todas las librerías necesarias estén instaladas en tu entorno de Python. A continuación, se muestra cómo instalar las librerías clave utilizando `pip`, el gestor de paquetes de Python.

Para instalar `IfcOpenShell`, `Pandas`, y `Matplotlib`, ejecuta los siguientes comandos en una celda del cuaderno Jupyter:

```
!pip install ifcopenshell
!pip install pandas
!pip install matplotlib
```

```
Collecting ifcopenshell
  Downloading ifcopenshell-0.8.0-py310-none-
manylinux_2_31_x86_64.whl.metadata (11 kB)
Requirement already satisfied: shapely in
/usr/local/lib/python3.10/dist-packages (from ifcopenshell) (2.0.6)
Requirement already satisfied: numpy in
/usr/local/lib/python3.10/dist-packages (from ifcopenshell) (1.26.4)
Collecting isodate (from ifcopenshell)
  Downloading isodate-0.6.1-py2.py3-none-any.whl.metadata (9.6 kB)
Requirement already satisfied: python-dateutil in
/usr/local/lib/python3.10/dist-packages (from ifcopenshell) (2.8.2)
Collecting lark (from ifcopenshell)
  Downloading lark-1.2.2-py3-none-any.whl.metadata (1.8 kB)
Requirement already satisfied: typing-extensions in
/usr/local/lib/python3.10/dist-packages (from ifcopenshell) (4.12.2)
Requirement already satisfied: six in
/usr/local/lib/python3.10/dist-packages (from isodate->ifcopenshell)
(1.16.0)
Downloading ifcopenshell-0.8.0-py310-none-manylinux_2_31_x86_64.whl
(40.8 MB)
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 40.8/40.8 MB 16.9 MB/s eta
0:00:00
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 41.7/41.7 kB 2.8 MB/s eta
0:00:00
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 111.0/111.0 kB 8.4 MB/s eta
0:00:00
ent already satisfied: pandas in /usr/local/lib/python3.10/dist-
packages (2.1.4)
Requirement already satisfied: numpy<2,>=1.22.4 in
/usr/local/lib/python3.10/dist-packages (from pandas) (1.26.4)
Requirement already satisfied: python-dateutil>=2.8.2 in
/usr/local/lib/python3.10/dist-packages (from pandas) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in
/usr/local/lib/python3.10/dist-packages (from pandas) (2024.1)
Requirement already satisfied: tzdata>=2022.1 in
/usr/local/lib/python3.10/dist-packages (from pandas) (2024.1)
Requirement already satisfied: six>=1.5 in
/usr/local/lib/python3.10/dist-packages (from python-
dateutil>=2.8.2->pandas) (1.16.0)
```

```
Requirement already satisfied: matplotlib in
/usr/local/lib/python3.10/dist-packages (3.7.1)
Requirement already satisfied: contourpy>=1.0.1 in
/usr/local/lib/python3.10/dist-packages (from matplotlib) (1.3.0)
Requirement already satisfied: cycler>=0.10 in
/usr/local/lib/python3.10/dist-packages (from matplotlib) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in
/usr/local/lib/python3.10/dist-packages (from matplotlib) (4.53.1)
Requirement already satisfied: kiwisolver>=1.0.1 in
/usr/local/lib/python3.10/dist-packages (from matplotlib) (1.4.5)
Requirement already satisfied: numpy>=1.20 in
/usr/local/lib/python3.10/dist-packages (from matplotlib) (1.26.4)
Requirement already satisfied: packaging>=20.0 in
/usr/local/lib/python3.10/dist-packages (from matplotlib) (24.1)
Requirement already satisfied: pillow>=6.2.0 in
/usr/local/lib/python3.10/dist-packages (from matplotlib) (9.4.0)
Requirement already satisfied: pyparsing>=2.3.1 in
/usr/local/lib/python3.10/dist-packages (from matplotlib) (3.1.4)
Requirement already satisfied: python-dateutil>=2.7 in
/usr/local/lib/python3.10/dist-packages (from matplotlib) (2.8.2)
Requirement already satisfied: six>=1.5 in
/usr/local/lib/python3.10/dist-packages (from python-dateutil>=2.7-
>matplotlib) (1.16.0)
```

IfcOpenShell

Este Jupyter Notebook ha sido cuidadosamente diseñado con un enfoque didáctico para proporcionar una guía exhaustiva sobre cómo analizar y manipular datos provenientes de archivos IFC utilizando Python. A lo largo de este cuaderno, se explorará el uso de diversas librerías como IfcOpenShell, Pandas, Matplotlib, entre otras, para facilitar la comprensión y manejo de modelos BIM (Building Information Modeling) en el formato IFC, particularmente en su versión 4.3.

Este recurso ha sido elaborado con la intención de servir como material de apoyo adicional para el Trabajo de Fin de Grado titulado "Análisis de Datos de un Modelo BIM en Formato IFC4.3", presentado por D. Jaime Zaforteza Quintanilla. Este trabajo ha sido tutorizado por el Ingeniero de Caminos, Canales y Puertos, D. Blas González González, y tiene como objetivo proporcionar una visión práctica y accesible sobre la aplicación de técnicas de análisis de datos en modelos BIM.

A través de este cuaderno, se espera no solo ilustrar los conceptos fundamentales relacionados con el tratamiento de archivos IFC, sino también facilitar el desarrollo de habilidades técnicas aplicables en la gestión y análisis de datos dentro del ámbito de la ingeniería civil y la arquitectura.

Introducción

En esta sección introductoria, comenzaremos con la carga de las librerías fundamentales para el manejo de archivos IFC en Python. Utilizaremos `IfcOpenShell`, una poderosa librería que nos permitirá interactuar con modelos BIM almacenados en archivos IFC (Industry Foundation Classes). Este formato es ampliamente utilizado en la industria de la

construcción para facilitar el intercambio de información entre diferentes sistemas y actores involucrados en el diseño y construcción de edificios.

Además de `IfcOpenShell`, también cargaremos algunos módulos adicionales que complementan su funcionalidad:

- `ifcopenshell.util`: Proporciona utilidades que simplifican la manipulación y consulta de datos dentro de los archivos IFC.
- `ifcopenshell.geom`: Facilita la generación y visualización de geometrías 3D derivadas de los datos IFC, lo cual es crucial para interpretar visualmente los modelos.

Estas herramientas serán la base sobre la cual construiremos análisis más complejos a lo largo de este cuaderno. A continuación, se procederá a importar las librerías necesarias.

```
import ifcopenshell
import ifcopenshell.util
import ifcopenshell.geom
import ifcopenshell.api
```

Carga del Modelo IFC

El primer paso para trabajar con datos en un archivo IFC es cargar el modelo en una variable de Python que nos permita acceder y manipular su contenido. Para ello, utilizaremos el módulo `IfcOpenShell`, el cual proporciona la función `.open()` para abrir y leer archivos IFC.

A continuación, se muestra cómo importar el módulo `IfcOpenShell` y cargar un archivo IFC en una variable llamada `modelo`. Este archivo IFC se encuentra en un directorio específico de tu sistema de archivos, que hemos definido previamente en la variable `directorio`.

```
directorio = "/content/TFG_JZQ_OL_E63_AV_Doble_via_v09.ifc"

# Cargamos el archivo IFC en una variable
modelo = ifcopenshell.open(directorio)
```

Verificación del Esquema del Archivo IFC

Una vez que se ha cargado el modelo IFC en la variable `modelo`, es importante determinar a qué esquema IFC corresponde el archivo. Los esquemas IFC definen las reglas y la estructura de cómo se organiza la información dentro del archivo, y es crucial conocer el esquema para comprender cómo interactuar con los datos.

Los esquemas IFC más comunes incluyen versiones como IFC2x3, IFC4, e IFC4.3, cada uno con diferentes niveles de complejidad y capacidad para representar elementos y relaciones dentro de un modelo BIM.

Para verificar a qué esquema corresponde el archivo IFC que hemos cargado, utilizaremos el siguiente enfoque:

```
esquema = modelo.schema
print(f"El archivo IFC corresponde al esquema: {esquema}")
```

```
El archivo IFC corresponde al esquema: IFC4X3
```

Filtrado del Modelo IFC para Obtener Distintos Elementos

Una vez que se ha cargado el modelo IFC y se ha verificado el esquema al que pertenece, es posible que desees explorar o manipular elementos específicos dentro del archivo. El módulo `IfcOpenShell` proporciona varias funciones útiles para filtrar y acceder a los distintos elementos del modelo IFC. A continuación, se presentarán tres métodos comunes para filtrar y obtener elementos dentro del modelo.

1. Filtrado por ID

Cada elemento dentro de un archivo IFC tiene un identificador único llamado **ID**. Este identificador es un número entero que permite acceder directamente a un elemento específico dentro del modelo.

modelo.by_id(1): Esta función devuelve el elemento dentro del modelo que tiene el ID 1. Cada elemento del modelo tiene un ID único, y esta función permite acceder a él de manera directa.

2. Filtrado por GUID

El **GUID** (Global Unique Identifier) es un identificador único global que se utiliza para identificar elementos de manera universal y consistente, incluso a través de diferentes sistemas y aplicaciones.

modelo.by_guid('0KMpiAlnb52RgQuM1CwVfd'): Esta función busca y devuelve el elemento que tiene el GUID especificado. Los GUID son particularmente útiles cuando se trabaja con modelos que han sido importados o exportados entre diferentes plataformas.

3. Filtrado por Tipo de Elemento

El modelo IFC está compuesto por una gran variedad de tipos de elementos, como paredes, puertas, ventanas, etc. Filtrar por tipo de elemento permite acceder a todos los elementos de un tipo específico dentro del modelo.

modelo.by_type("IfcCourse"): Esta función devuelve una lista de todos los elementos que son del tipo `IfcCourse` (es decir, todas las capas) en el modelo. Al añadir `[0]`, se accede al primer elemento de esta lista, mostrando el primer `IfcCourse` encontrado en el archivo.

Importancia de los Métodos de Filtrado

Estos métodos de filtrado son esenciales para navegar y manipular grandes modelos BIM, permitiendo a los usuarios seleccionar, inspeccionar y modificar elementos específicos de manera eficiente. El uso de filtros por ID, GUID y tipo de elemento proporciona flexibilidad y precisión en el análisis y la gestión de los datos contenidos en un archivo IFC.

Al dominar estos métodos, se puede obtener un mayor control sobre los datos del modelo y facilitar el desarrollo de análisis y operaciones más complejas.

```
print(modelo.by_id(1))
```

```
print(modelo.by_guid('3tEwE8N1LBdA2j8CbNjPY1'))

print(modelo.by_type("IfcCourse")[0])
```

```
#1=IfcOrganization($,'Autodesk',$,$,$)
#34031=IfcCourse('3tEwE8N1LBdA2j8CbNjPY1',$,'Ballast',
$,'CorridorShape',#34030,#34029,$,.BALLASTBED.)
#34031=IfcCourse('3tEwE8N1LBdA2j8CbNjPY1',$,'Ballast',
$,'CorridorShape',#34030,#34029,$,.BALLASTBED.)
```

Una de las opciones más interesantes para trabajar con modelos IFC es la capacidad de seleccionar elementos específicos mediante el filtrado basado en el tipo de instancia de IFC que se desea analizar. Esto se puede lograr utilizando la función **.by_type()**.

La función **.by_type()** permite buscar y recuperar todos los elementos de un modelo que pertenecen a un tipo específico, como `IfcWall` para paredes, `IfcDoor` para puertas, entre otros. Este método es extremadamente útil cuando se necesita trabajar con una categoría particular de elementos dentro del modelo, facilitando su análisis y manipulación.

Además, existe otra función complementaria llamada **is_a()** que puede ser utilizada para verificar el tipo de entidad IFC de un elemento. Esta función genera un estado booleano (True o False) dependiendo de si el elemento pertenece a la clase especificada o a una de sus clases derivadas. En otras palabras, **is_a()** devuelve `True` si el tipo de entidad IFC del elemento coincide con la clase concreta indicada o si pertenece a una jerarquía de clases relacionadas con dicha entidad.

Estas herramientas combinadas permiten una interacción flexible y precisa con los modelos IFC, permitiendo filtrar, identificar y trabajar con los elementos del modelo de manera más eficaz, según el tipo de datos que se desea analizar o modificar.

```
capa = modelo.by_type("IfcCourse")[0]

print(capa.is_a("IfcCourse")) # clase que corresponde

print(capa.is_a("IfcEarthworksElement"))

print(capa.is_a("IfcElement")) # parentesco de un elemento
constructivo, la capa desciende de la entidad de un elemento
```

```
True
False
True
```

Podemos acceder a los atributos de una instancia IFC de forma individual, lo que nos permite obtener información específica sobre un elemento particular del modelo. Sin embargo, este proceso puede resultar tedioso y poco eficiente, especialmente cuando se trabaja con un gran número de atributos o instancias.

Para simplificar este proceso, se puede utilizar la función **.get_info()**. Esta función es particularmente útil porque devuelve un diccionario que contiene toda la información asociada con una instancia específica del modelo IFC.

El diccionario que genera **.get_info()** incluye todas las propiedades y atributos del elemento en cuestión, organizados en pares clave-valor. Las claves corresponden a los nombres de los atributos, mientras que los valores contienen los datos asociados a esos atributos.

Esta función permite obtener una visión completa de todos los atributos de una instancia de manera rápida y organizada, facilitando así el análisis de datos en el modelo. Es una herramienta poderosa cuando se necesita inspeccionar o extraer información detallada sobre un elemento sin tener que acceder a cada atributo individualmente.

```
capa.get_info()

{'id': 34031,
 'type': 'IfcCourse',
 'GlobalId': '3tEwE8N1LBdA2j8CbNjPY1',
 'OwnerHistory': None,
 'Name': 'Ballast',
 'Description': None,
 'ObjectType': 'CorridorShape',
 'ObjectPlacement': #34030=IfcLocalPlacement(#81251,#29),
 'Representation': #34029=IfcProductDefinitionShape($,$,
(#33948,#34028)),
 'Tag': None,
 'PredefinedType': 'BALLASTBED'}
```

Utilizando las claves del diccionario que se obtiene con la función **.get_info()**, podemos acceder a los valores de cada propiedad de una instancia específica en el modelo IFC. Esto nos permite extraer y manipular de manera eficiente la información relevante de cualquier elemento.

Para acceder a un valor específico, simplemente se utiliza el nombre de la propiedad como clave dentro del diccionario. Por ejemplo, si se desea obtener el valor de una propiedad como `GlobalId`, `Name`, o cualquier otra, se puede hacer directamente utilizando su título como clave.

Importante: Excepción para la Propiedad *type*

Es fundamental destacar que la propiedad *type* de una instancia no se obtiene únicamente a través del diccionario generado por **.get_info()**. En su lugar, para determinar el tipo de una instancia IFC, se puede utilizar la función **.is_a()**.

La función **.is_a()** devuelve el tipo de la instancia y puede verificar si una instancia pertenece a una clase específica o a una jerarquía de clases. Esto es útil para identificar la categoría exacta del elemento dentro del modelo IFC, ya que **.is_a()** no solo confirma el tipo, sino que también permite comprobar si el elemento pertenece a una subclase relacionada.

Ejemplo:

- Para obtener el valor del `GlobalId`: `instancia.get_info()['GlobalId']`
- Para obtener el tipo de la instancia: `instancia.is_a()`

Este método combinado de utilizar **.get_info()** para las propiedades generales y **.is_a()** para el tipo proporciona una forma completa y eficiente de trabajar con la información de instancias en un modelo IFC.

```
tipo_instancia = capa.is_a()

Nombre_capa = capa.Name
```

```
Descripcion_capa = capa.Description

print(f"La instancia es de clase:{tipo_instancia}, con el siguiente
Nombre: {Nombre_capa}, donde nos detalle la siguiente descripción:
{Descripcion_capa}")
```

```
La instancia es de clase:IfcCourse, con el siguiente Nombre:
Ballast, donde nos detalle la siguiente descripción: None
```

Para acceder a las **Propiedades** de una instancia en el modelo IFC, así como a sus conjuntos de propiedades (*property_set*), y también a las **Cantidades** (*Quantities*) y sus conjuntos de cantidades (*Qty_set*), es necesario utilizar un módulo de utilidades que proporciona `IfcOpenShell`. Este módulo, conocido como `ifcopenshell.util`, facilita el acceso y manipulación de estos datos específicos.

El módulo `ifcopenshell.util`, que fue importado al inicio del cuaderno junto con los demás módulos de la librería, incluye funciones que permiten extraer y organizar la información contenida en los *property_sets* y *quantity_sets* de una instancia.

Acceso a Propiedades y Cantidades

Usando las funciones de `ifcopenshell.util`, se puede obtener un diccionario que contiene subdiccionarios, donde cada subdiccionario representa un conjunto de propiedades (*property_set*) o un conjunto de cantidades (*quantity_set*). Dentro de estos subdiccionarios, las claves representan los nombres de las propiedades o cantidades, y los valores contienen la información correspondiente a cada una.

Funcionamiento:

1. **Propiedades (Psets):** Se pueden obtener utilizando funciones específicas de `ifcopenshell.util`, que recogen todos los *property_sets* asociados a una instancia y los organizan en un diccionario. Cada *property_set* contiene varias propiedades que describen diferentes aspectos del elemento.

2. **Cantidades (Qsets):** De manera similar, se puede acceder a las cantidades relacionadas con una instancia, organizadas dentro de *quantity_sets*. Estas cantidades suelen incluir información sobre dimensiones, áreas, volúmenes, entre otros atributos cuantitativos importantes.

La función `ifcopenshell.util.element.get_psets(muro)` se utiliza para extraer todos los conjuntos de propiedades (*property sets*) asociados a una instancia específica en un modelo IFC. En este caso, se asume que la variable `muro` representa una instancia de `IfcWall` o cualquier otro elemento dentro del modelo.

Funcionamiento de `get_psets`

Al llamar a `ifcopenshell.util.element.get_psets(muro)`, se obtiene un diccionario donde cada clave corresponde al nombre de un *property set* (Pset) y cada valor es otro diccionario que contiene las propiedades individuales dentro de ese conjunto. Estas propiedades describen diversos atributos del elemento, como su material, dimensiones, o características de rendimiento.

Ejemplo de Uso:

Supongamos que tienes una pared (`muro`) en tu modelo IFC y deseas obtener un resumen de todas las propiedades asociadas a esa pared. Al ejecutar `get_psets(muro)`, recibirás un diccionario con la estructura de todos los *property sets* y sus respectivas propiedades.

Estructura del Resultado:

- **Clave:** Nombre del *property set* (por ejemplo, "Pset_WallCommon").
- **Valor:** Otro diccionario donde las claves son los nombres de las propiedades (por ejemplo, "FireRating", "LoadBearing") y los valores son los datos correspondientes a esas propiedades.

Importancia de `get_psets`:

Esta función es extremadamente útil para inspeccionar y analizar de manera exhaustiva las propiedades de los elementos en un modelo BIM. Te permite acceder de forma estructurada a toda la información relevante sobre un elemento, lo cual es crucial para tareas de gestión de datos, evaluación de cumplimiento normativo, y optimización del diseño.

Al utilizar `get_psets`, se facilita el trabajo con modelos IFC, permitiendo extraer y trabajar con datos precisos y detallados sobre cada elemento del modelo, lo cual es esencial para mantener la calidad y coherencia de los proyectos en el ámbito de la arquitectura, ingeniería y construcción.

```
ifcopenshell.util.element.get_psets(capa)
```

Algunos atributos en un modelo IFC son conocidos como "atributos inversos". Estos atributos se generan cuando otro elemento dentro del modelo establece una referencia hacia el elemento en cuestión. En otras palabras, los atributos inversos indican relaciones en las que el elemento está implicado debido a la acción de otros elementos del modelo.

¿Qué Son los Atributos Inversos?

Los atributos inversos se utilizan para definir cómo otros elementos del modelo interactúan con el elemento actual. Estas interacciones pueden ser de varios tipos, como:

- **Creación de vacíos:** Por ejemplo, si una ventana o una puerta crea un vacío en una pared.
- **Unión de elementos:** Como cuando se unen dos paredes.
- **Definición de cantidades:** Al definir los valores de las cantidades de materiales o dimensiones en relación con el elemento.

Ejemplo de Uso: `capa.IsDefinedBy`

Al tratar un atributo inverso, se puede acceder a él de la misma manera que se accedería a un atributo normal. Por ejemplo, `capa.IsDefinedBy` es un atributo inverso que devolverá todas las entidades de relación que están vinculadas a ese muro. Esto incluye cualquier entidad que defina o modifique las características de la capa, como propiedades, cantidades, o relaciones espaciales.

Funcionamiento:

- **Atributo Inverso:** `capa.IsDefinedBy`

- Este atributo devolverá una lista de entidades que establecen relaciones con el muro, como relaciones de propiedades (`IfcRelDefinesByProperties`), relaciones de cantidad (`IfcRelDefinesByQuantities`), entre otras.

Importancia de los Atributos Inversos:

Los atributos inversos son cruciales para entender el contexto completo de un elemento dentro del modelo IFC. Permiten rastrear cómo otros elementos del modelo influyen en o dependen del elemento actual, lo que es esencial para un análisis detallado y preciso del modelo BIM.

Tratarlos como atributos normales facilita su acceso y manipulación, permitiendo a los usuarios explorar las relaciones y dependencias entre elementos de manera efectiva. Esto es particularmente útil en la gestión de grandes proyectos, donde la interrelación entre los componentes es compleja y de vital importancia para la integridad del modelo.

```
capa.IsDefinedBy  # La capa no nos arroja ningun resultado ya que por el momento como se ha perdido la información no dispone de propiedades asignadas

()
```

Si queremos ver todas las instancias que están referenciando a otra en un modelo IFC, podemos utilizar la función `ifc_file.get_inverse(ENTIDAD_IFC)`. Esta función devuelve una lista de todas las entidades que tienen una referencia a la entidad especificada, permitiendo identificar las relaciones y dependencias dentro del modelo.

```
modelo.get_inverse(capa)

{#34035=IfcRelAssociatesClassification('2rIBVrSRPCLf0KNrUaPaKe',$,$,$,
(#34031,#38222,#46124,#56033,#62420,#66472,#68659,#69694,#71033),#34034),
 #33635=IfcRelContainedInSpatialStructure('0Pm0aC9b2tmcbx2Y4FAc9W',$,$,' Container for Elements',
(#33634,#33733,#33828,#33923,#34031,#34101,#34166),#33532)}
```

Diferencia entre `.IsDefinedBy` y `get_inverse()`

Es importante distinguir entre dos métodos clave que permiten explorar las relaciones de un elemento dentro de un modelo IFC:

1. **.IsDefinedBy**:
   - **Uso:** Se aplica directamente sobre un objeto de instancia (por ejemplo, un muro, una puerta, etc.).
   - **Función:** Devuelve todas las entidades de relación que definen o establecen una conexión con esa instancia. Esto incluye propiedades, cantidades, y cualquier otra relación que influya directamente en el elemento.
   - **Ejemplo:** `capa.IsDefinedBy` muestra las entidades que definen propiedades o cantidades para el muro.
2. **get_inverse()**:
   - **Uso:** Se aplica sobre un objeto del modelo completo (`ifc_file`), no directamente sobre una instancia específica.

- **Función:** Devuelve todas las instancias dentro del modelo que están referenciando a la instancia seleccionada. Es decir, muestra todos los elementos que dependen de o interactúan con la instancia.
- **Ejemplo:** `ifc_file.get_inverse(capa)` devuelve todas las entidades en el modelo que hacen referencia al muro, como ventanas que crean vacíos en él, o elementos estructurales que están conectados.

Relación Opuesta: Elementos a los que Nuestro Muro se Refiere

Ahora, para entender la relación opuesta, es decir, para ver todos los elementos a los que nuestro muro se refiere, debemos explorar las relaciones que nuestro muro establece con otros elementos en el modelo. Esto se puede lograr analizando atributos específicos del muro que contengan referencias a otras instancias.

Por ejemplo, un muro podría referirse a:

- **Materiales**: La capa podría referirse a los materiales de construcción utilizados.
- **Espacios**: Podría estar relacionado con espacios específicos en la infraestructura.
- **Conexiones**: Podría estar conectado con otros elementos estructurales de la vía ferroviaria.

Explorar estas relaciones nos permite comprender cómo la capa interactúa y depende de otros componentes dentro del modelo BIM, proporcionando una visión completa de su integración en el diseño.

```
modelo.traverse(capa) # Así obtendriamos todas las instancias a las que accede o se refiere nuestra capa

[#34031=IfcCourse('3tEwE8N1LBdA2j8CbNjPY1',$,'Ballast',
$,'CorridorShape',#34030,#34029,$,.BALLASTBED.),
 #34030=IfcLocalPlacement(#81251,#29),
 #81251=IfcLocalPlacement(#81482,#81250),
 #81482=IfcLocalPlacement(#81477,#81481),
 #81477=IfcLocalPlacement($,#81476),
 #81476=IfcAxis2Placement3D(#81473,#81474,#81475),
 #81473=IfcCartesianPoint((0.,0.,0.)),
 #81474=IfcDirection((0.,0.,1.)),
 #81475=IfcDirection((1.,0.,0.)),
 #81481=IfcAxis2Placement3D(#81478,#81479,#81480),
 #81478=IfcCartesianPoint((0.,0.,0.)),
 #81479=IfcDirection((0.,0.,1.)),
 #81480=IfcDirection((1.,0.,0.)),
 #81250=IfcAxis2Placement3D(#81247,#81248,#81249),
 #81247=IfcCartesianPoint((0.,0.,0.)),
 #81248=IfcDirection((0.,0.,1.)),
 #81249=IfcDirection((1.,0.,0.)),
 #29=IfcAxis2Placement3D(#28,$,$),
 #28=IfcCartesianPoint((0.,0.,0.)),
 #34029=IfcProductDefinitionShape($,$,(#33948,#34028)),
 #33948=IfcShapeRepresentation(#32183,'Body','AdvancedSweptSolid',
(#33947)),

#32183=IfcGeometricRepresentationSubContext('Body','Model',*,*,*,*,#
22,$,.MODEL_VIEW.,$),
```

```
#22=IfcGeometricRepresentationContext('3D','Model',3,0.0001,#24,#25)
,
 #24=IfcAxis2Placement3D(#23,$,$),
 #23=IfcCartesianPoint((0.,0.,0.)),
 #25=IfcDirection((0.,1.)),
 #33947=IfcSectionedSolidHorizontal(#33530,
(#33931,#33934,#33937,#33940,#33943,#33946),
(#33518,#33520,#33522,#33524,#33526,#33528)),
 #33530=IfcIndexedPolyCurve(#33529,$,$),
 #33529=IfcCartesianPointList3D(((8.65576,5.00777,745.44032),
(17.30763,10.02228,745.40964),(34.6001,20.07067,745.34829),
(51.8783,30.14359,745.28693),(69.1431,40.23947,745.22558),
(86.39538,50.35672,745.16422)),$),
 #33931=IfcArbitraryClosedProfileDef(.AREA.,$,#33930),
 #33930=IfcIndexedPolyCurve(#33929,
(IfcLineIndex((1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,1))),$),
 #33929=IfcCartesianPointList2D(((-5.1767,-0.93284),(-4.068,-
0.1937),(-3.55,-0.1937),(-2.25,-0.1937),(-0.95,-0.1937),(-0.432,-
0.1937),(-0.282,-0.2778),(0.282,-0.2778),(0.432,-0.1937),(0.95,-
0.1937),(2.25,-0.1937),(3.55,-0.1937),(4.068,-0.1937),(5.1767,-
0.93284),(0.,-0.674)),
('Bottom_Edge_Ballast','Top_Edge_Ballast','','','','Top_Edge_Ballast
','Top_Edge_Ballast','Top_Edge_Ballast','Top_Edge_Ballast','','','',
'Top_Edge_Ballast','Bottom_Edge_Ballast','Top_Subballast')),
 IfcLineIndex((1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,1)),
 #33934=IfcArbitraryClosedProfileDef(.AREA.,$,#33933),
 #33933=IfcIndexedPolyCurve(#33932,
(IfcLineIndex((1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,1))),$),
 #33932=IfcCartesianPointList2D(((-5.1767,-0.93284),(-4.068,-
0.1937),(-3.55,-0.1937),(-2.25,-0.1937),(-0.95,-0.1937),(-0.432,-
0.1937),(-0.282,-0.2778),(0.282,-0.2778),(0.432,-0.1937),(0.95,-
0.1937),(2.25,-0.1937),(3.55,-0.1937),(4.068,-0.1937),(5.1767,-
0.93284),(0.,-0.674)),
('Bottom_Edge_Ballast','Top_Edge_Ballast','','','','Top_Edge_Ballast
','Top_Edge_Ballast','Top_Edge_Ballast','Top_Edge_Ballast','','','',
'Top_Edge_Ballast','Bottom_Edge_Ballast','Top_Subballast')),
 IfcLineIndex((1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,1)),
 #33937=IfcArbitraryClosedProfileDef(.AREA.,$,#33936),
 #33936=IfcIndexedPolyCurve(#33935,
(IfcLineIndex((1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,1))),$),
 #33935=IfcCartesianPointList2D(((-5.1767,-0.93284),(-4.068,-
0.1937),(-3.55,-0.1937),(-2.25,-0.1937),(-0.95,-0.1937),(-0.432,-
0.1937),(-0.282,-0.2778),(0.282,-0.2778),(0.432,-0.1937),(0.95,-
0.1937),(2.25,-0.1937),(3.55,-0.1937),(4.068,-0.1937),(5.1767,-
0.93284),(0.,-0.674)),
('Bottom_Edge_Ballast','Top_Edge_Ballast','','','','Top_Edge_Ballast
','Top_Edge_Ballast','Top_Edge_Ballast','Top_Edge_Ballast','','','',
'Top_Edge_Ballast','Bottom_Edge_Ballast','Top_Subballast')),
 IfcLineIndex((1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,1)),
 #33940=IfcArbitraryClosedProfileDef(.AREA.,$,#33939),
 #33939=IfcIndexedPolyCurve(#33938,
(IfcLineIndex((1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,1))),$),
```

```
 #33938=IfcCartesianPointList2D(((-5.1767,-0.93284),(-4.068,-
0.1937),(-3.55,-0.1937),(-2.25,-0.1937),(-0.95,-0.1937),(-0.432,-
0.1937),(-0.282,-0.2778),(0.282,-0.2778),(0.432,-0.1937),(0.95,-
0.1937),(2.25,-0.1937),(3.55,-0.1937),(4.068,-0.1937),(5.1767,-
0.93284),(0.,-0.674)),
('Bottom_Edge_Ballast','Top_Edge_Ballast','','','','Top_Edge_Ballast
','Top_Edge_Ballast','Top_Edge_Ballast','Top_Edge_Ballast','','','',
'Top_Edge_Ballast','Bottom_Edge_Ballast','Top_Subballast')),
 IfcLineIndex((1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,1)),
 #33943=IfcArbitraryClosedProfileDef(.AREA.,$,#33942),
 #33942=IfcIndexedPolyCurve(#33941,
(IfcLineIndex((1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,1))),$),
 #33941=IfcCartesianPointList2D(((-5.1767,-0.93284),(-4.068,-
0.1937),(-3.55,-0.1937),(-2.25,-0.1937),(-0.95,-0.1937),(-0.432,-
0.1937),(-0.282,-0.2778),(0.282,-0.2778),(0.432,-0.1937),(0.95,-
0.1937),(2.25,-0.1937),(3.55,-0.1937),(4.068,-0.1937),(5.1767,-
0.93284),(0.,-0.674)),
('Bottom_Edge_Ballast','Top_Edge_Ballast','','','','Top_Edge_Ballast
','Top_Edge_Ballast','Top_Edge_Ballast','Top_Edge_Ballast','','','',
'Top_Edge_Ballast','Bottom_Edge_Ballast','Top_Subballast')),
 IfcLineIndex((1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,1)),
 #33946=IfcArbitraryClosedProfileDef(.AREA.,$,#33945),
 #33945=IfcIndexedPolyCurve(#33944,
(IfcLineIndex((1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,1))),$),
 #33944=IfcCartesianPointList2D(((-5.1767,-0.93284),(-4.068,-
0.1937),(-3.55,-0.1937),(-2.25,-0.1937),(-0.95,-0.1937),(-0.432,-
0.1937),(-0.282,-0.2778),(0.282,-0.2778),(0.432,-0.1937),(0.95,-
0.1937),(2.25,-0.1937),(3.55,-0.1937),(4.068,-0.1937),(5.1767,-
0.93284),(0.,-0.674)),
('Bottom_Edge_Ballast','Top_Edge_Ballast','','','','Top_Edge_Ballast
','Top_Edge_Ballast','Top_Edge_Ballast','Top_Edge_Ballast','','','',
'Top_Edge_Ballast','Bottom_Edge_Ballast','Top_Subballast')),
 IfcLineIndex((1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,1)),
 #33518=IfcAxis2PlacementLinear(#33517,#122,$),

#33517=IfcPointByDistanceExpression(IfcLengthMeasure(10.000000078749
9),$,$,$,#407),
 IfcLengthMeasure(10.0000000078749),

#407=IfcGradientCurve((#345,#352,#358,#365,#371,#378,#384,#391,#397,
#404),.F.,#114,$),

#345=IfcCurveSegment(.CONTSAMEGRADIENTSAMECURVATURE.,#344,IfcLengthM
easure(0.),IfcLengthMeasure(644.829510388072),#75),
 #344=IfcAxis2Placement2D(#342,#343),
 #342=IfcCartesianPoint((0.,745.471)),
 #343=IfcDirection((0.99999529,-0.00306776)),
 IfcLengthMeasure(0.),
 IfcLengthMeasure(644.829510388072),
 #75=IfcLine(#72,#74),
 #72=IfcCartesianPoint((0.,0.)),
 #74=IfcVector(#73,1.),
 #73=IfcDirection((1.,0.)),
```

```
#352=IfcCurveSegment(.CONTSAMEGRADIENTSAMECURVATURE.,#350,IfcParamet
erValue(0.),IfcParameterValue(172.666007820471),#351),
 #350=IfcAxis2Placement2D(#348,#349),
 #348=IfcCartesianPoint((644.82648,743.49282)),
 #349=IfcDirection((0.99999529,-0.00306776)),
 IfcParameterValue(0.),
 IfcParameterValue(172.666007820471),
 #351=IfcPolynomialCurve(#64,(0.,1.),(743.492815122969,-
0.00306777862016657,-2.49999999999E-06),$),
 #64=IfcAxis2Placement2D(#63,$),
 #63=IfcCartesianPoint((0.,0.)),

#358=IfcCurveSegment(.CONTSAMEGRADIENTSAMECURVATURE.,#357,IfcLengthM
easure(0.),IfcLengthMeasure(1093.9908876596),#75),
 #357=IfcAxis2Placement2D(#355,#356),
 #355=IfcCartesianPoint((817.49248,742.88858)),
 #356=IfcDirection((0.99999227,-0.00393108)),
 IfcLengthMeasure(0.),
 IfcLengthMeasure(1093.9908876596),

#365=IfcCurveSegment(.CONTSAMEGRADIENTSAMECURVATURE.,#363,IfcParamet
erValue(0.),IfcParameterValue(164.356509140379),#364),
 #363=IfcAxis2Placement2D(#361,#362),
 #361=IfcCartesianPoint((1911.47492,738.58802)),
 #362=IfcDirection((0.99999227,-0.00393108)),
 IfcParameterValue(0.),
 IfcParameterValue(164.356509140379),
 #364=IfcPolynomialCurve(#64,(0.,1.),(738.588016337968,-
0.00393110865926832,-3.333333333334E-05),$),

#371=IfcCurveSegment(.CONTSAMEGRADIENTSAMECURVATURE.,#370,IfcLengthM
easure(0.),IfcLengthMeasure(78.1163772517592),#75),
 #370=IfcAxis2Placement2D(#368,#369),
 #368=IfcCartesianPoint((2075.83143,737.04148)),
 #369=IfcDirection((0.99988919,-0.01488656)),
 IfcLengthMeasure(0.),
 IfcLengthMeasure(78.1163772517592),

#378=IfcCurveSegment(.CONTSAMEGRADIENTSAMECURVATURE.,#376,IfcParamet
erValue(0.),IfcParameterValue(305.490630428967),#377),
 #376=IfcAxis2Placement2D(#374,#375),
 #374=IfcCartesianPoint((2153.93915,735.87859)),
 #375=IfcDirection((0.99988919,-0.01488656)),
 IfcParameterValue(0.),
 IfcParameterValue(305.490630428967),
 #377=IfcPolynomialCurve(#64,(0.,1.),(735.878593541212,-
0.0148882092686261,3.333333333333E-05),$),

#384=IfcCurveSegment(.CONTSAMEGRADIENTSAMECURVATURE.,#383,IfcLengthM
easure(0.),IfcLengthMeasure(664.226225091042),#75),
 #383=IfcAxis2Placement2D(#381,#382),
 #381=IfcCartesianPoint((2459.42978,734.4412)),
 #382=IfcDirection((0.999985,0.00547775)),
 IfcLengthMeasure(0.),
```

```
 IfcLengthMeasure(664.226225091042),

#391=IfcCurveSegment(.CONTSAMEGRADIENTSAMECURVATURE.,#389,IfcParamet
erValue(0.),IfcParameterValue(351.507056642586),#390),
 #389=IfcAxis2Placement2D(#387,#388),
 #387=IfcCartesianPoint((3123.64604,738.07967)),
 #388=IfcDirection((0.999985,0.00547775)),
 IfcParameterValue(0.),
 IfcParameterValue(351.507056642586),
 #390=IfcPolynomialCurve(#64,(0.,1.),
(738.079668202272,0.00547783275997181,-3.331519574151E-05),$),

#397=IfcCurveSegment(.CONTSAMEGRADIENTSAMECURVATURE.,#396,IfcLengthM
easure(0.),IfcLengthMeasure(124.865769022731),#75),
 #396=IfcAxis2Placement2D(#394,#395),
 #394=IfcCartesianPoint((3475.1531,735.88883)),
 #395=IfcDirection((0.99983906,-0.01794033)),
 IfcLengthMeasure(0.),
 IfcLengthMeasure(124.865769022731),

#404=IfcCurveSegment(.CONTSAMEGRADIENTSAMECURVATURE.,#402,IfcParamet
erValue(0.),IfcParameterValue(221.233292076737),#403),
 #402=IfcAxis2Placement2D(#400,#401),
 #400=IfcCartesianPoint((3599.99877,733.6487)),
 #401=IfcDirection((0.99984182,-0.01778574)),
 IfcParameterValue(0.),
 IfcParameterValue(221.233292076737),
 #403=IfcPolynomialCurve(#64,(0.,1.),(733.648699025624,-
0.0177885571784107,3.334358707774E-05),$),
 #114=IfcCompositeCurve((#66,#76,#82,#88,#94,#99,#105,#111),.F.),

#66=IfcCurveSegment(.CONTSAMEGRADIENTSAMECURVATURE.,#61,IfcLengthMea
sure(-354.055147669013),IfcLengthMeasure(354.055147669013),#65),
 #61=IfcAxis2Placement2D(#36,#62),
 #36=IfcCartesianPoint((0.,0.)),
 #62=IfcDirection((0.86577444,0.50043443)),
 IfcLengthMeasure(-354.055147669013),
 IfcLengthMeasure(354.055147669013),
 #65=IfcClothoid(#64,-2103.72901752829),

#76=IfcCurveSegment(.CONTSAMEGRADIENTSAMECURVATURE.,#70,IfcLengthMea
sure(0.),IfcLengthMeasure(1954.59947035847),#75),
 #70=IfcAxis2Placement2D(#40,#71),
 #40=IfcCartesianPoint((304.84268,180.06596)),
 #71=IfcDirection((0.85860058,0.51264515)),
 IfcLengthMeasure(0.),
 IfcLengthMeasure(1954.59947035847),

#82=IfcCurveSegment(.CONTSAMEGRADIENTSAMECURVATURE.,#79,IfcLengthMea
sure(0.),IfcLengthMeasure(275.000108006667),#81),
 #79=IfcAxis2Placement2D(#43,#80),
 #43=IfcCartesianPoint((1983.06292,1182.0819)),
 #80=IfcDirection((0.85860058,0.51264515)),
 IfcLengthMeasure(0.),
```

```
 IfcLengthMeasure(275.000108006667),
 #81=IfcClothoid(#64,812.404000006898),

#88=IfcCurveSegment(.CONTSAMEGRADIENTSAMECURVATURE.,#85,IfcLengthMea
sure(0.),IfcLengthMeasure(341.988938920502),#87),
 #85=IfcAxis2Placement2D(#46,#86),
 #46=IfcCartesianPoint((2216.40903,1327.5212)),
 #86=IfcDirection((0.82783761,0.56096781)),
 IfcLengthMeasure(0.),
 IfcLengthMeasure(341.988938920502),
 #87=IfcCircle(#64,2400.),

#94=IfcCurveSegment(.CONTSAMEGRADIENTSAMECURVATURE.,#91,IfcLengthMea
sure(-275.000108006667),IfcLengthMeasure(275.000108006667),#93),
 #91=IfcAxis2Placement2D(#49,#92),
 #49=IfcCartesianPoint((2484.91783,1538.85433)),
 #92=IfcDirection((0.73978214,0.67284648)),
 IfcLengthMeasure(-275.000108006667),
 IfcLengthMeasure(275.000108006667),
 #93=IfcClothoid(#64,-812.404000006898),

#99=IfcCurveSegment(.CONTSAMEGRADIENTSAMECURVATURE.,#97,IfcLengthMea
sure(0.),IfcLengthMeasure(399.300612222264),#75),
 #97=IfcAxis2Placement2D(#52,#98),
 #52=IfcCartesianPoint((2681.11539,1731.49263)),
 #98=IfcDirection((0.70004094,0.71410271)),
 IfcLengthMeasure(0.),
 IfcLengthMeasure(399.300612222264),

#105=IfcCurveSegment(.CONTSAMEGRADIENTSAMECURVATURE.,#102,IfcLengthM
easure(0.),IfcLengthMeasure(-221.287675554857),#104),
 #102=IfcAxis2Placement2D(#55,#103),
 #55=IfcCartesianPoint((2960.64216,2016.63428)),
 #103=IfcDirection((0.70004094,0.71410271)),
 IfcLengthMeasure(0.),
 IfcLengthMeasure(-221.287675554857),
 #104=IfcCircle(#64,7320.08077),

#111=IfcCurveSegment(.CONTSAMEGRADIENTSAMECURVATURE.,#108,IfcLengthM
easure(0.),IfcLengthMeasure(0.),#110),
 #108=IfcAxis2Placement2D(#58,#109),
 #58=IfcCartesianPoint((3117.91734,2172.29104)),
 #109=IfcDirection((0.72130529,0.69261727)),
 IfcLengthMeasure(0.),
 IfcLengthMeasure(0.),
 #110=IfcCircle(#64,7320.08077),
 #122=IfcDirection((0.,0.,1.)),
 #33520=IfcAxis2PlacementLinear(#33519,#122,$),
 #33519=IfcPointByDistanceExpression(IfcLengthMeasure(20.),$,$,
$,#407),
 IfcLengthMeasure(20.),
 #33522=IfcAxis2PlacementLinear(#33521,#122,$),
 #33521=IfcPointByDistanceExpression(IfcLengthMeasure(40.),$,$,
$,#407),
```

```
 IfcLengthMeasure(40.),
 #33524=IfcAxis2PlacementLinear(#33523,#122,$),
 #33523=IfcPointByDistanceExpression(IfcLengthMeasure(60.),$,$,
$,#407),
 IfcLengthMeasure(60.),
 #33526=IfcAxis2PlacementLinear(#33525,#122,$),
 #33525=IfcPointByDistanceExpression(IfcLengthMeasure(80.),$,$,
$,#407),
 IfcLengthMeasure(80.),
 #33528=IfcAxis2PlacementLinear(#33527,#122,$),

#33527=IfcPointByDistanceExpression(IfcLengthMeasure(99.999999963137),$,$,$,#407),
 IfcLengthMeasure(99.9999999963137),
 #34028=IfcShapeRepresentation(#33561,'Body-
Fallback','Tessellation',(#33950)),
 #33561=IfcGeometricRepresentationSubContext('Body-
Fallback','Model',*,*,*,*,#22,$,.MODEL_VIEW.,$),
 #33950=IfcPolygonalFaceSet(#33949,$,
(#33951,#33952,#33953,#33954,#33955,#33956,#33957,#33958,#33959,#33960,#33961,#33962,#33963,#33964,#33965,#33966,#33967,#33968,#33969,#33970,#33971,#33972,#33973,#33974,#33975,#33976,#33977,#33978,#33979,#33980,#33981,#33982,#33983,#33984,#33985,#33986,#33987,#33988,#33989,#33990,#33991,#33992,#33993,#33994,#33995,#33996,#33997,#33998,#33999,#34000,#34001,#34002,#34003,#34004,#34005,#34006,#34007,#34008,#34009,#34010,#34011,#34012,#34013,#34014,#34015,#34016,#34017,#34018,#34019,#34020,#34021,#34022,#34023,#34024,#34025,#34026,#34027),$),
 #33949=IfcCartesianPointList3D(((11.24989,0.52796,744.50749),
(10.69431,1.48741,745.24662),(10.43473,1.93568,745.24662),
(9.78327,3.06067,745.24662),(9.13182,4.18566,745.24662),
(8.87224,4.63393,745.24662),(8.79708,4.76374,745.16252),
(8.51445,5.25181,745.16252),(8.43928,5.38162,745.24662),
(8.1797,5.82988,745.24662),(7.52825,6.95488,745.24662),
(6.87679,8.07987,745.24662),(6.61722,8.52814,745.24662),
(6.06163,9.48759,744.50749),(8.65576,5.00777,744.76632),
(19.90519,5.54445,744.47681),(19.34887,6.50348,745.21594),
(19.08894,6.95154,745.21594),(18.43663,8.07604,745.21594),
(17.78432,9.20053,745.21594),(17.52439,9.6486,745.21594),
(17.44913,9.77835,745.13184),(17.16612,10.2662,745.13184),
(17.09086,10.39595,745.21594),(16.83093,10.84402,745.21594),
(16.17862,11.96852,745.21594),(15.52631,13.09301,745.21594),
(15.26638,13.54108,745.21594),(14.71006,14.5001,744.47681),
(17.30763,10.02228,744.73564),(37.20422,15.59665,744.41545),
(36.64649,16.55486,745.15459),(36.38592,17.00255,745.15459),
(35.73196,18.12608,745.15459),(35.078,19.24962,745.15459),
(34.81742,19.69731,745.15459),(34.74196,19.82695,745.07049),
(34.45824,20.31439,745.07049),(34.38279,20.44403,745.15459),
(34.12221,20.89172,745.15459),(33.46825,22.01525,745.15459),
(32.81429,23.13879,745.15459),(32.55371,23.58648,745.15459),
(31.99598,24.54468,744.41545),(34.6001,20.07067,744.67429),
(54.48856,25.67316,744.3541),(53.92952,26.6306,745.09323),
(53.66833,27.07793,745.09323),(53.01282,28.20057,745.09323),
(52.35732,29.3232,745.09323),(52.09613,29.77053,745.09323),
```

```
(52.02049,29.90007,745.00913),(51.73611,30.38712,745.00913),
(51.66047,30.51666,745.09323),(51.39928,30.96398,745.09323),
(50.74377,32.08662,745.09323),(50.08827,33.20926,745.09323),
(49.82708,33.65659,745.09323),(49.26803,34.61403,744.3541),
(51.8783,30.14359,744.61293),(71.7591,35.77239,744.29274),
(71.19883,36.72911,745.03188),(70.93706,37.17611,745.03188),
(70.28011,38.2979,745.03188),(69.62317,39.4197,745.03188),
(69.3614,39.86669,745.03188),(69.2856,39.99613,744.94778),
(69.00059,40.48281,744.94778),(68.92479,40.61225,745.03188),
(68.66302,41.05924,745.03188),(68.00608,42.18104,745.03188),
(67.34914,43.30283,745.03188),(67.08737,43.74983,745.03188),
(66.52709,44.70655,744.29274),(69.1431,40.23947,744.55158),
(89.01671,45.89277,744.23139),(88.45529,46.84882,744.97052),
(88.19299,47.2955,744.97052),(87.53471,48.41651,744.97052),
(86.87643,49.53752,744.97052),(86.61413,49.9842,744.97052),
(86.53817,50.11355,744.88642),(86.25258,50.59989,744.88642),
(86.17663,50.72924,744.97052),(85.91433,51.17592,744.97052),
(85.25604,52.29693,744.97052),(84.59776,53.41794,744.97052),
(84.33546,53.86462,744.97052),(83.77405,54.82067,744.23139),
(86.39538,50.35672,744.49022)),$),

#33951=IfcIndexedPolygonalFace((1,2,3,4,5,6,7,8,9,10,11,12,13,14,15)
),
 #33952=IfcIndexedPolygonalFace((1,16,17,2)),
 #33953=IfcIndexedPolygonalFace((2,17,18,3)),
 #33954=IfcIndexedPolygonalFace((3,18,19,4)),
 #33955=IfcIndexedPolygonalFace((4,19,20,5)),
 #33956=IfcIndexedPolygonalFace((5,20,21,6)),
 #33957=IfcIndexedPolygonalFace((6,21,22,7)),
 #33958=IfcIndexedPolygonalFace((7,22,23,8)),
 #33959=IfcIndexedPolygonalFace((8,23,24,9)),
 #33960=IfcIndexedPolygonalFace((9,24,25,10)),
 #33961=IfcIndexedPolygonalFace((10,25,26,11)),
 #33962=IfcIndexedPolygonalFace((11,26,27,12)),
 #33963=IfcIndexedPolygonalFace((12,27,28,13)),
 #33964=IfcIndexedPolygonalFace((13,28,29,14)),
 #33965=IfcIndexedPolygonalFace((14,29,30,15)),
 #33966=IfcIndexedPolygonalFace((15,30,16,1)),
 #33967=IfcIndexedPolygonalFace((16,31,32,17)),
 #33968=IfcIndexedPolygonalFace((17,32,33,18)),
 #33969=IfcIndexedPolygonalFace((18,33,34,19)),
 #33970=IfcIndexedPolygonalFace((19,34,35,20)),
 #33971=IfcIndexedPolygonalFace((20,35,36,21)),
 #33972=IfcIndexedPolygonalFace((21,36,37,22)),
 #33973=IfcIndexedPolygonalFace((22,37,38,23)),
 #33974=IfcIndexedPolygonalFace((23,38,39,24)),
 #33975=IfcIndexedPolygonalFace((24,39,40,25)),
 #33976=IfcIndexedPolygonalFace((25,40,41,26)),
 #33977=IfcIndexedPolygonalFace((26,41,42,27)),
 #33978=IfcIndexedPolygonalFace((27,42,43,28)),
 #33979=IfcIndexedPolygonalFace((28,43,44,29)),
 #33980=IfcIndexedPolygonalFace((29,44,45,30)),
 #33981=IfcIndexedPolygonalFace((30,45,31,16)),
```

```
#33982=IfcIndexedPolygonalFace((31,46,47,32)),
#33983=IfcIndexedPolygonalFace((32,47,48,33)),
#33984=IfcIndexedPolygonalFace((33,48,49,34)),
#33985=IfcIndexedPolygonalFace((34,49,50,35)),
#33986=IfcIndexedPolygonalFace((35,50,51,36)),
#33987=IfcIndexedPolygonalFace((36,51,52,37)),
#33988=IfcIndexedPolygonalFace((37,52,53,38)),
#33989=IfcIndexedPolygonalFace((38,53,54,39)),
#33990=IfcIndexedPolygonalFace((39,54,55,40)),
#33991=IfcIndexedPolygonalFace((40,55,56,41)),
#33992=IfcIndexedPolygonalFace((41,56,57,42)),
#33993=IfcIndexedPolygonalFace((42,57,58,43)),
#33994=IfcIndexedPolygonalFace((43,58,59,44)),
#33995=IfcIndexedPolygonalFace((44,59,60,45)),
#33996=IfcIndexedPolygonalFace((45,60,46,31)),
#33997=IfcIndexedPolygonalFace((46,61,62,47)),
#33998=IfcIndexedPolygonalFace((47,62,63,48)),
#33999=IfcIndexedPolygonalFace((48,63,64,49)),
#34000=IfcIndexedPolygonalFace((49,64,65,50)),
#34001=IfcIndexedPolygonalFace((50,65,66,51)),
#34002=IfcIndexedPolygonalFace((51,66,67,52)),
#34003=IfcIndexedPolygonalFace((52,67,68,53)),
#34004=IfcIndexedPolygonalFace((53,68,69,54)),
#34005=IfcIndexedPolygonalFace((54,69,70,55)),
#34006=IfcIndexedPolygonalFace((55,70,71,56)),
#34007=IfcIndexedPolygonalFace((56,71,72,57)),
#34008=IfcIndexedPolygonalFace((57,72,73,58)),
#34009=IfcIndexedPolygonalFace((58,73,74,59)),
#34010=IfcIndexedPolygonalFace((59,74,75,60)),
#34011=IfcIndexedPolygonalFace((60,75,61,46)),
#34012=IfcIndexedPolygonalFace((61,76,77,62)),
#34013=IfcIndexedPolygonalFace((62,77,78,63)),
#34014=IfcIndexedPolygonalFace((63,78,79,64)),
#34015=IfcIndexedPolygonalFace((64,79,80,65)),
#34016=IfcIndexedPolygonalFace((65,80,81,66)),
#34017=IfcIndexedPolygonalFace((66,81,82,67)),
#34018=IfcIndexedPolygonalFace((67,82,83,68)),
#34019=IfcIndexedPolygonalFace((68,83,84,69)),
#34020=IfcIndexedPolygonalFace((69,84,85,70)),
#34021=IfcIndexedPolygonalFace((70,85,86,71)),
#34022=IfcIndexedPolygonalFace((71,86,87,72)),
#34023=IfcIndexedPolygonalFace((72,87,88,73)),
#34024=IfcIndexedPolygonalFace((73,88,89,74)),
#34025=IfcIndexedPolygonalFace((74,89,90,75)),
#34026=IfcIndexedPolygonalFace((75,90,76,61)),

#34027=IfcIndexedPolygonalFace((76,90,89,88,87,86,85,84,83,82,81,80,
79,78,77))]

# Podemos referirnos únicamente a las que se refiere bajo 1 solo
nivel jerárquico:
modelo.traverse(capa, max_levels=1)

# Al no imprimir el valor con la función print(), unicamente
```

```
obtendremos el valor de la
# última línea de código

[#34031=IfcCourse('3tEwE8N1LBdA2j8CbNjPY1',$,'Ballast',
$,'CorridorShape',#34030,#34029,$,.BALLASTBED.),
 #34030=IfcLocalPlacement(#81251,#29),
 #34029=IfcProductDefinitionShape($,$,(#33948,#34028))]
```

Añadir Property Sets al Modelo IFC

Un paso clave que debemos considerar en nuestro análisis es la adición de los *Property Sets* (conjuntos de propiedades) al modelo IFC. Como se explica en la memoria del trabajo, durante la exportación de los datos, es posible que algunos conjuntos de propiedades se hayan perdido. Para solventar este problema, se pueden generar estos conjuntos de propiedades de forma nativa dentro del modelo IFC utilizando código.

En este ejemplo, se crean varios *Property Sets* vacíos y se asignan a las entidades constructivas. Aunque no llenaremos estos conjuntos de propiedades con información en este momento, estableceremos la estructura básica de los *Property Sets* para que estén disponibles dentro del modelo.

Explicación de Código:

1. **Selección del Elemento**: Se selecciona una entidad constructiva del archivo IFC (por ejemplo, un muro o cualquier otro elemento `IfcElement`) al cual se le añadirán los conjuntos de propiedades.

2. **Creación de Property Sets**:

   - Se crean varios conjuntos de propiedades como `01_JAND_IDENTIFICACION`, `02_JAND_CANTIDADES`, `03_JAND_PROYECTO`, y `04_JAND_OBRA`.
   - Cada conjunto de propiedades incluye una lista de atributos definidos en el diccionario, aunque sus valores se inicializan vacíos (`""`).

3. **Asignación de Property Sets**: Una vez creados los *Property Sets*, se asocian al elemento seleccionado utilizando funciones de `ifcopenshell.api`, que permiten tanto agregar un conjunto de propiedades como editarlo.

Este enfoque permite la creación y asignación de conjuntos de propiedades personalizados a las entidades del modelo BIM, lo que garantiza que el modelo mantenga la estructura adecuada para futuros análisis o usos en diferentes plataformas.

```
# Selecciona un elemento al que quieras añadir el Property Set, en
este caso lo haremos para una capa
elemento = modelo.by_type("IfcCourse")[0]  # Este es solo un
ejemplo, debes ajustar para seleccionar los elementos necesarios.

# Property Set 01_JAND_IDENTIFICACION
pset_01 = ifcopenshell.api.run("pset.add_pset", modelo,
product=elemento, name="01_JAND_IDENTIFICACION")
properties_01 = {
    "01_01_JAND_PROYECTO": "",
    "01_02_JAND_LOCALIZADOR": "",
    "01_03_JAND_CLASIFICACION": "",
```

```
    "01_04_JAND_DISCIPLINA": "",
    "01_05_JAND_SUBDISCIPLINA": "",
    "01_06_JAND_CDD_PRESUP": "",
    "01_07_JAND_XXXXXXX": ""
}
ifcopenshell.api.run("pset.edit_pset", modelo, pset=pset_01,
properties=properties_01)

# Property Set 02_JAND_CANTIDADES
pset_02 = ifcopenshell.api.run("pset.add_pset", modelo,
product=elemento, name="02_JAND_CANTIDADES")
properties_02 = {
    "02_01_JAND_UNIDAD": "",
    "02_02_JAND_LONGITUD": "",
    "02_03_JAND_ESPESOR": "",
    "02_04_JAND_AREA": "",
    "02_05_JAND_VOLUMEN": "",
    "02_06_JAND_XXXXXXX": ""
}
ifcopenshell.api.run("pset.edit_pset", modelo, pset=pset_02,
properties=properties_02)

# Property Set 03_JAND_PROYECTO
pset_03 = ifcopenshell.api.run("pset.add_pset", modelo,
product=elemento, name="03_JAND_PROYECTO")
properties_03 = {
    "03_01_JAND_FASE": "",
    "03_02_JAND_PLANOS": "",
    "03_03_JAND_PPTP": "",
    "03_04_JAND_CAP_PRESUP": "",
    "03_05_JAND_SUBCAP_PRESUP": "",
    "03_06_JAND_UD_PRESUP": ""
}
ifcopenshell.api.run("pset.edit_pset", modelo, pset=pset_03,
properties=properties_03)

# Property Set 04_JAND_OBRA
pset_04 = ifcopenshell.api.run("pset.add_pset", modelo,
product=elemento, name="04_JAND_OBRA")
properties_04 = {
    "04_01_JAND_XXXXXXX": ""
}
ifcopenshell.api.run("pset.edit_pset", modelo, pset=pset_04,
properties=properties_04)

ifcopenshell.util.element.get_psets(capa)

{'01_JAND_IDENTIFICACION': {'01_01_JAND_PROYECTO': '',
  '01_02_JAND_LOCALIZADOR': '',
  '01_03_JAND_CLASIFICACION': '',
  '01_04_JAND_DISCIPLINA': '',
  '01_05_JAND_SUBDISCIPLINA': '',
  '01_06_JAND_CDD_PRESUP': '',
  '01_07_JAND_XXXXXXX': '',
```

```
  'id': 81733},
 '02_JAND_CANTIDADES': {'02_01_JAND_UNIDAD': '',
  '02_02_JAND_LONGITUD': '',
  '02_03_JAND_ESPESOR': '',
  '02_04_JAND_AREA': '',
  '02_05_JAND_VOLUMEN': '',
  '02_06_JAND_XXXXXXX': '',
  'id': 81742},
 '03_JAND_PROYECTO': {'03_01_JAND_FASE': '',
  '03_02_JAND_PLANOS': '',
  '03_03_JAND_PPTP': '',
  '03_04_JAND_CAP_PRESUP': '',
  '03_05_JAND_SUBCAP_PRESUP': '',
  '03_06_JAND_UD_PRESUP': '',
  'id': 81750},
 '04_JAND_OBRA': {'04_01_JAND_XXXXXXX': '', 'id': 81758}}
```

```
capa.IsDefinedBy
```

```
(#81734=IfcRelDefinesByProperties('2gPloV6lj9$8$XMuLKKbOQ',$,$,$,
(#34031),#81733),
 #81743=IfcRelDefinesByProperties('2UKyqnptH04fJomzY2$mSl',$,$,$,
(#34031),#81742),
 #81751=IfcRelDefinesByProperties('1n3wrNcZDBixz27OdblO6c',$,$,$,
(#34031),#81750),
 #81759=IfcRelDefinesByProperties('0NZkWssJn0Mx6u89bfaBKj',$,$,$,
(#34031),#81758))
```

Modificación de Datos en un Modelo IFC

Además de inspeccionar y analizar un modelo IFC, también es posible modificar los datos asociados a los elementos dentro del modelo de manera directa y sencilla. Esto se logra asignando nuevos valores a las propiedades de las instancias.

Proceso de Modificación

Para modificar un dato, se sigue un proceso simple:

1. **Seleccionar la Instancia**: Primero, se identifica la instancia (por ejemplo, un muro, una puerta, etc.) que se desea modificar.
2. **Asignar el Nuevo Valor**: Se define una variable o se asigna directamente el nuevo valor al atributo deseado de la instancia.

Ejemplo de Modificación:

Supongamos que queremos cambiar el nombre de un muro en nuestro modelo IFC. El proceso sería:

1. Seleccionamos el muro de interés.
2. Asignamos un nuevo valor al atributo `Name` de esa instancia.

```
# Por ejemplo cambiamos el nombre de nuestro muro:
```

```
capa.Name = "Nuevo Nombre de Capa"

# tal que:

capa.get_info() # podemos ver que se ha actualizado los datos que
contiene la instancia.

{'id': 34031,
 'type': 'IfcCourse',
 'GlobalId': '3tEwE8N1LBdA2j8CbNjPY1',
 'OwnerHistory': None,
 'Name': 'Nuevo Nombre de Capa',
 'Description': None,
 'ObjectType': 'CorridorShape',
 'ObjectPlacement': #34030=IfcLocalPlacement(#81251,#29),
 'Representation': #34029=IfcProductDefinitionShape($,$,
(#33948,#34028)),
 'Tag': None,
 'PredefinedType': 'BALLASTBED'}
```

Guardar Cambios en un Nuevo Archivo IFC

Después de realizar modificaciones en los datos dentro de un modelo IFC, es posible que desees guardar estos cambios. Para mantener una copia del modelo original intacta, es recomendable guardar el modelo modificado como un nuevo archivo IFC.

Proceso para Guardar Cambios:

1. **Modificar el Modelo**: Realiza los cambios necesarios en las propiedades de las instancias del modelo.

2. **Guardar el Modelo Modificado**: Una vez que todos los cambios han sido realizados, el modelo puede ser guardado utilizando la función de guardado proporcionada por `IfcOpenShell`.

Ejemplo de Guardado:

Supongamos que se han realizado varios cambios en un modelo, como modificar nombres de paredes o ajustar propiedades. Para guardar estos cambios en un nuevo archivo, se utiliza una función de guardado especificando la ruta y el nombre del nuevo archivo.

Detalles del Proceso:

- La función de guardado permite guardar el modelo modificado en la ubicación especificada, creando un nuevo archivo IFC que contiene todos los cambios realizados. Es necesario especificar la ruta completa y el nombre del nuevo archivo.

Beneficios de Guardar un Nuevo Archivo:

Guardar el modelo modificado como un nuevo archivo permite mantener una versión del modelo con los cambios recientes, mientras se conserva el archivo original sin modificaciones. Esto es útil para mantener un historial de cambios o para distribuir diferentes versiones del modelo a distintos colaboradores en un proyecto.

De esta manera, los cambios realizados se preservan y pueden ser revisados o utilizados en futuras etapas del proyecto, garantizando la continuidad y consistencia en la gestión del modelo BIM.

```
nuevo_directorio = "/content/nuevo.ifc"

modelo.write(nuevo_directorio)
```

Ejemplos de Código

En esta sección, se presentarán una serie de ejemplos prácticos que ilustran cómo obtener datos e información específica de un archivo IFC utilizando Python y `IfcOpenShell`. Estos ejemplos están diseñados para ayudarte a comprender mejor cómo interactuar con los elementos del modelo y extraer la información necesaria para tu análisis o proyecto.

A continuación se muestran ejemplos de tareas básicas comunes. En todos los ejemplos, se supone que tiene un modelo IFC cargado en la variable de modelo de la siguiente manera:

```
import pandas as pd
import matplotlib.pyplot as plt
```

Análisis Aplicado con IfcOpenShell y Herramientas Adicionales

Una vez que se han comprendido las funcionalidades básicas de `IfcOpenShell` para interactuar con archivos IFC, es posible llevar el análisis a un nivel más avanzado mediante la integración de herramientas adicionales como `Pandas` y `Matplotlib`. Estas bibliotecas permiten organizar, manipular y visualizar los datos extraídos del modelo IFC de manera más estructurada y visualmente atractiva.

1. **Pandas para Manipulación de Datos**

`Pandas` es una poderosa biblioteca en Python que facilita el manejo y análisis de datos. Al fusionar `IfcOpenShell` con `Pandas`, se pueden transformar los datos extraídos de los archivos IFC en estructuras de datos tabulares como `DataFrames`. Esto permite:

- **Organización de Datos:** Los datos de los elementos IFC pueden organizarse en tablas, donde cada fila representa una instancia (como una pared o una ventana) y cada columna un atributo o propiedad.
- **Análisis Estadístico:** `Pandas` permite realizar cálculos, filtrados, y agregaciones sobre los datos, facilitando el análisis estadístico y la generación de informes.
- **Exportación de Datos:** Los resultados del análisis pueden exportarse fácilmente a formatos como CSV, Excel, o JSON para compartir o documentar.

2. **Matplotlib para Visualización de Datos**

`Matplotlib` es una biblioteca de visualización en Python que permite crear gráficos y visualizaciones de datos. Al integrar `Matplotlib` con `IfcOpenShell` y `Pandas`, se puede:

- **Visualizar Distribuciones:** Crear gráficos de barras, histogramas, y gráficos de dispersión para visualizar la distribución de diferentes atributos, como la altura de las paredes o el tamaño de las ventanas.
- **Análisis Comparativo:** Comparar diferentes elementos del modelo mediante gráficos que muestren las variaciones en sus propiedades o cantidades.

- **Visualización Espacial:** Aunque `Matplotlib` no está diseñado específicamente para visualizaciones 3D, puede usarse para representar ciertos aspectos geométricos de los datos de manera simplificada.

Ejemplo de Integración

Un flujo de trabajo típico podría incluir:

- **Extracción de Datos:** Utilizar `IfcOpenShell` para extraer información relevante del modelo IFC, como las dimensiones de los elementos, sus propiedades, y relaciones.
- **Organización en DataFrame:** Crear un `DataFrame` de `Pandas` con los datos extraídos, facilitando su manipulación y análisis.
- **Visualización:** Utilizar `Matplotlib` para crear gráficos que resuman y presenten los hallazgos de manera clara y concisa.

Extracción de Todas las Entidades del Archivo IFC

En este ejemplo, se muestra cómo extraer todas las entidades de tipo `IfcElement` desde un archivo IFC utilizando `IfcOpenShell`, organizar esa información en un diccionario, y finalmente convertirla en un `DataFrame` de `Pandas` para un análisis estructurado y más avanzado.

Explicación del Proceso:

1. **Extracción de Entidades IFC**:
   - Se utiliza la función `modelo.by_type("IfcElement")` para obtener todas las entidades del modelo que pertenecen al tipo `IfcElement`. Este tipo agrupa una amplia variedad de elementos constructivos como paredes, puertas, ventanas, etc.
2. **Iteración sobre las Entidades**:
   - Se itera sobre cada una de las entidades extraídas del archivo. Para cada entidad, se obtiene:
     - **Tipo de Entidad**: Se utiliza `entity.is_a()` para determinar a qué tipo de clase pertenece la entidad.
     - **ID de Entidad**: En este caso, se extrae el atributo `Name` para obtener una identificación más clara del elemento. Es importante que `Name` esté definido en el archivo.
3. **Almacenamiento en un Diccionario**:
   - Para cada entidad, se crea un diccionario con la información clave (tipo de entidad e ID). Luego, este diccionario se agrega a una lista que contendrá los detalles de todas las entidades.
4. **Creación de un DataFrame de Pandas**:
   - Una vez que se ha recopilado toda la información, se convierte la lista de diccionarios en un `DataFrame` de `Pandas`. Este formato tabular facilita la manipulación, filtrado y análisis de los datos.

Este enfoque permite estructurar y analizar los datos de las entidades dentro del archivo IFC de manera más eficiente, utilizando las potentes funcionalidades de `Pandas` para organizar y explorar los atributos de los elementos del modelo.

```
# Extraer todas las entidades del archivo IFC
entities = modelo.by_type("IfcElement")
entities_list = []
```

```
# Iterar sobre cada entidad y guardar sus valores dentro de un
diccionario
for entity in entities:
    entity_type = entity.is_a()
    entity_id = entity.Name
    entities = {"Tipo de entidad": entity_type}

    entities_list.append(entities)

# Guardar el diccionario en un DataFrame de Pandas
df_entities = pd.DataFrame(entities_list)
```

Contar la Frecuencia de Tipos de Entidades en el DataFrame

Una vez que los datos de las entidades han sido organizados en un `DataFrame` de `Pandas`, es posible realizar análisis adicionales, como contar cuántas veces aparece cada tipo de entidad dentro del modelo IFC.

Proceso:

1. **Conteo de Frecuencias**:
   - Utilizando la función `value_counts()`, se puede obtener el número de veces que cada tipo de entidad aparece en el `DataFrame`. Esto genera un resumen que muestra la frecuencia de cada tipo de entidad en el modelo.
2. **Aplicación en el DataFrame**:
   - El conteo se aplicará sobre la columna que contiene el tipo de entidad (en este caso, la columna "Tipo de entidad"). El resultado es una lista con el tipo de entidad y el número de veces que esa entidad aparece en el archivo IFC.

Este método es útil para obtener una visión general de la distribución de elementos dentro del modelo. Por ejemplo, permite saber cuántas paredes (`IfcWall`), ventanas (`IfcWindow`), o puertas (`IfcDoor`) hay en el modelo, proporcionando información relevante para el análisis y planificación.

```
conteo = df_entities.value_counts()

conteo
```

```
Tipo de entidad
IfcRail                          36
IfcEarthworksElement             34
IfcEarthworksFill                19
IfcCourse                        18
IfcDistributionChamberElement    16
IfcEarthworksCut                 10
IfcWall                           3
IfcPlate                          2
Name: count, dtype: int64
```

Visualización de la Distribución de Entidades

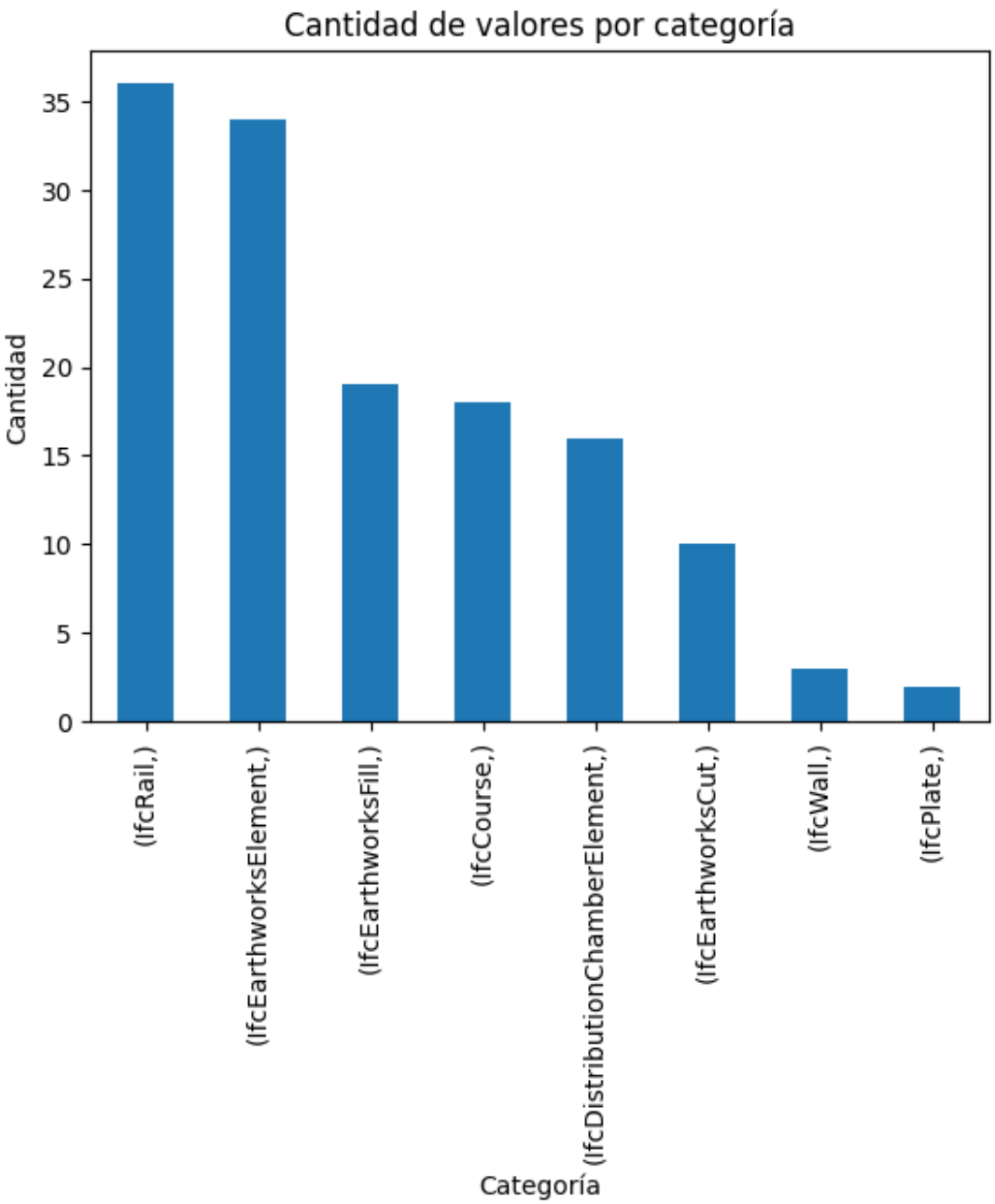
Una vez realizado el conteo de los tipos de entidades, se puede visualizar la distribución utilizando un gráfico de barras. Para ello, se usa el método `plot(kind='bar')` de `Pandas`, que genera un gráfico de barras mostrando la cantidad de cada tipo de entidad en el modelo.

Explicación:

- **`plot(kind='bar')`**: Genera un gráfico de barras con las categorías (tipos de entidad) en el eje X y la cantidad en el eje Y.
- **Etiquetas y Título**: Se añaden etiquetas a los ejes y un título descriptivo al gráfico.
- **`plt.show()`**: Muestra el gráfico generado.

Esta visualización es útil para identificar rápidamente la distribución y cantidad de diferentes entidades dentro del modelo IFC.

```
conteo.plot(kind='bar')
plt.xlabel('Categoría')
plt.ylabel('Cantidad')
plt.title('Cantidad de valores por categoría')
plt.show()
```



OBETENER TODOS LOS TIPOS DE CAPA:

```
for tipo_capa in modelo.by_type("IfcCourse"):
    print("El elemento de Capa, IfcCourse es:", tipo_capa)
    print("El nombre de ese tipo de capa es:", tipo_capa.Name)
```

```
El elemento de Capa, IfcCourse es:
#34031=IfcCourse('3tEwE8N1LBdA2j8CbNjPY1',$,'Nuevo Nombre de Capa',
$,'CorridorShape',#34030,#34029,$,.BALLASTBED.)
El nombre de ese tipo de capa es: Nuevo Nombre de Capa
El elemento de Capa, IfcCourse es:
#34101=IfcCourse('1783vLb7HEhB3SqCHPDVbG',$,'Subballast',
$,'CorridorShape',#34100,#34099,$,.BALLASTBED.)
El nombre de ese tipo de capa es: Subballast
El elemento de Capa, IfcCourse es:
```

```
#38222=IfcCourse('1HaoYtHYb5fBhXdFDyPwHS',$,'Ballast',
$,'CorridorShape',#38221,#38220,$,.BALLASTBED.)
El nombre de ese tipo de capa es: Ballast
El elemento de Capa, IfcCourse es:
#38667=IfcCourse('1SsmYIqwvAUBuYZyrPMlyU',$,'Subballast',
$,'CorridorShape',#38666,#38665,$,.BALLASTBED.)
El nombre de ese tipo de capa es: Subballast
El elemento de Capa, IfcCourse es:
#46124=IfcCourse('2okXC2b7n6CPX6M09P1xy_',$,'Ballast',
$,'CorridorShape',#46123,#46122,$,.BALLASTBED.)
El nombre de ese tipo de capa es: Ballast
El elemento de Capa, IfcCourse es:
#46719=IfcCourse('0EPn0njMn10OQBSMzZGngM',$,'Subballast',
$,'CorridorShape',#46718,#46717,$,.BALLASTBED.)
El nombre de ese tipo de capa es: Subballast
El elemento de Capa, IfcCourse es:
#49085=IfcCourse('3z7SktHl9BiB1rhEY7yVxY',$,'HLimpieza',
$,'CorridorShape',#81527,#49083,$,.PROTECTION.)
El nombre de ese tipo de capa es: HLimpieza
El elemento de Capa, IfcCourse es:
#56033=IfcCourse('1hXTJBrY5F1wU2ot5GHqYR',$,'Ballast',
$,'CorridorShape',#56032,#56031,$,.BALLASTBED.)
El nombre de ese tipo de capa es: Ballast
El elemento de Capa, IfcCourse es:
#56441=IfcCourse('2n09Eh2GL1thIpR5srV0_h',$,'Subballast',
$,'CorridorShape',#56440,#56439,$,.BALLASTBED.)
El nombre de ese tipo de capa es: Subballast
El elemento de Capa, IfcCourse es:
#62420=IfcCourse('0PF083QVTCnffrUQF02y11',$,'Ballast',
$,'CorridorShape',#62419,#62418,$,.BALLASTBED.)
El nombre de ese tipo de capa es: Ballast
El elemento de Capa, IfcCourse es:
#62906=IfcCourse('1KOt6I8ir5D9VzGmvedjJs',$,'Subballast',
$,'CorridorShape',#62905,#62904,$,.BALLASTBED.)
El nombre de ese tipo de capa es: Subballast
El elemento de Capa, IfcCourse es:
#66472=IfcCourse('20NTDcTJP2PeLOHtl3RdEB',$,'Ballast',
$,'CorridorShape',#66471,#66470,$,.BALLASTBED.)
El nombre de ese tipo de capa es: Ballast
El elemento de Capa, IfcCourse es:
#66581=IfcCourse('2my6Uw1Bj5QwXMjwyHfhgh',$,'Subballast',
$,'CorridorShape',#66580,#66579,$,.BALLASTBED.)
El nombre de ese tipo de capa es: Subballast
El elemento de Capa, IfcCourse es:
#68659=IfcCourse('2UFr4FOcnEYB1RFDBVZaEH',$,'Ballast',
$,'CorridorShape',#68658,#68657,$,.BALLASTBED.)
El nombre de ese tipo de capa es: Ballast
El elemento de Capa, IfcCourse es:
#69694=IfcCourse('1R2sqBTSv4oRktzabRDgN5',$,'Ballast',
$,'CorridorShape',#69693,#69692,$,.BALLASTBED.)
El nombre de ese tipo de capa es: Ballast
El elemento de Capa, IfcCourse es:
#69765=IfcCourse('1k6b3i3ZX4ZwPv7A51MfJ7',$,'Subballast',
```

```
$,'CorridorShape',#81532,#69763,$,.BALLASTBED.)
El nombre de ese tipo de capa es: Subballast
El elemento de Capa, IfcCourse es:
#71033=IfcCourse('3sJz_bFnH4RAQEH$w$8uGH',$,'Ballast',
$,'CorridorShape',#71032,#71031,$,.BALLASTBED.)
El nombre de ese tipo de capa es: Ballast
El elemento de Capa, IfcCourse es:
#71138=IfcCourse('25Xauq0efAwvs4z8lzNdRX',$,'Subballast',
$,'CorridorShape',#71137,#71136,$,.BALLASTBED.)
El nombre de ese tipo de capa es: Subballast
```

Sacar todas las instancias de una capa (ifcCourse), por ejemplo, de un determinado tipo (ifcCourseType)

```
for course_type in modelo.by_type("IfcCourseType"):
    print("El tipo de capa es:", course_type.Name)
    capas = ifcopenshell.util.element.get_types(course_type)
    print(f"Tenemos una cantidad de: {len(capas)} de ese tipo de
capa")
    for capa in capas:
        print("El nombre asignado a esa capa es:", capa.Name) # Las
capas no disponen de subtipos
```

Identificación del Contenedor de Elementos en el Modelo IFC

En los modelos BIM, muchos elementos están organizados dentro de **contenedores espaciales**, como niveles de construcción (*BuildingStorey*), espacios (*Spaces*), o zonas específicas. Por ejemplo, los muros suelen estar situados en un nivel o piso, y los equipos pueden estar colocados dentro de espacios definidos.

Extracción del Contenedor Espacial:

- **`get_container(capa)`**: Esta función del módulo `ifcopenshell.util.element` permite obtener el contenedor espacial en el que se encuentra una instancia determinada, en este caso, la variable `capa` que representa un elemento como un muro.
- **Ejemplo Real**: Al ejecutar el código, se obtendrá el contenedor espacial en el que está situado el muro (o capa en este caso). En el ejemplo dado, se especifica que el muro está en el nivel "Erdgeschoss", que en alemán significa planta baja o "Nivel 01".

Uso Práctico:

Este tipo de análisis es útil para entender la distribución de los elementos dentro del edificio y asegurarse de que cada elemento está correctamente ubicado dentro de su correspondiente nivel, espacio, o zona.

```
# Los muros suelen estar situados en un (BuildingStorey) Nivel o
Piso, los equipos pueden estar situados en espacios, etc.
container = ifcopenshell.util.element.get_container(capa)

# El muro esta en el nivel 01, Erdgeschoss es Planta Baja en Alemán.
print(f"La capa está contenida en: {container.Name}")
```

```
La capa está contenida en: RailDoubleTrack - 6301

container.get_info()

{'id': 33532,
 'type': 'IfcRailwayPart',
 'GlobalId': '1LhWVYqin4ARx4lIB9QlAU',
 'OwnerHistory': None,
 'Name': 'RailDoubleTrack - 6301',
 'Description': None,
 'ObjectType': 'TRACKSTRUCTURE',
 'ObjectPlacement': #81251=IfcLocalPlacement(#81482,#81250),
 'Representation': None,
 'LongName': None,
 'CompositionType': None,
 'UsageType': 'LATERAL',
 'PredefinedType': 'USERDEFINED'}
```

Descomposición de Contenedores Espaciales y sus Elementos

En este ejemplo, se descompone una parte espacial del modelo, como un `IfcRailwayPart`, y se listan todos los elementos que están contenidos dentro de esa parte del modelo. La función `get_decomposition()` se utiliza para obtener los elementos desglosados que están directamente relacionados con un contenedor espacial.

Explicación:

1. **Recorrido de `IfcRailwayPart`**:
   - Se itera sobre todas las instancias del tipo `IfcRailwayPart`, que representan secciones específicas de un sistema ferroviario dentro del modelo.
2. **Obtención de Elementos Descompuestos**:
   - Para cada parte espacial, se utiliza la función `get_decomposition()` para obtener los elementos que están contenidos en ese `IfcRailwayPart`.
3. **Visualización de los Resultados**:
   - Se imprime la cantidad de elementos contenidos dentro de cada `IfcRailwayPart` y se listan los nombres de esos elementos.

Aplicación:

Este enfoque permite analizar cómo los elementos están organizados dentro de diferentes partes espaciales del modelo, proporcionando una visión detallada de la estructura jerárquica y la relación espacial de los elementos en el modelo BIM.

```python
for spatialPart in modelo.by_type("IfcRailwayPart"):
    elements =
ifcopenshell.util.element.get_decomposition(spatialPart)
    print(f"Hay una cantidad de {len(elements)} elementos
localizados en el contenedor: {spatialPart.Name}, son:")
    for element in elements:
        print(element.Name)
```

```
Hay una cantidad de 7 elementos localizados en el contenedor:
RailDoubleTrack - 6301, son:
Rail
Rail
Rail
Rail
Nuevo Nombre de Capa
Subballast
SubBase
Hay una cantidad de 8 elementos localizados en el contenedor:
Substructure - 6301, son:
Berma2
Cuneta
Berma1
Talud-Desmonte
Berma2
Talud-Desmonte.001
Berma1
Cuneta
Hay una cantidad de 7 elementos localizados en el contenedor:
RailDoubleTrack - 6302, son:
Rail
Rail
Rail
Rail
Ballast
Subballast
SubBase
Hay una cantidad de 8 elementos localizados en el contenedor:
Substructure - 6302, son:
Cuneta
Talud-Desmonte.007
Berma1
Berma2
Talud-Desmonte.006
Berma2
Berma1
Cuneta
Hay una cantidad de 7 elementos localizados en el contenedor:
RailDoubleTrack- 6303, son:
Rail
Rail
Rail
Rail
Ballast
Subballast
SubBase
Hay una cantidad de 11 elementos localizados en el contenedor:
Substructure - 6303, son:
RellenoTrasdos
HLimpieza
Muro
Berma1
Cunetas
```

```
Cuneta
Berma2
RellenoTrados
Talud-Desmonte.008
Talud-Desmonte.009
RellenoTrasdos
Hay una cantidad de 7 elementos localizados en el contenedor:
RailDoubleTrack -  6302 (1), son:
Rail
Rail
Rail
Rail
Ballast
Subballast
SubBase
Hay una cantidad de 18 elementos localizados en el contenedor:
Substructure - 6302 (1), son:
Berma1
Berma1
Talud-Terraplen
Talud-Desmonte.002
Berma1
Talud-Terraplen
Berma2
Berma2
Cuneta
Talud-Desmonte.005
Berma1
Cuneta
Talud-Desmonte.003
Berma2
Berma1
Berma2
Cuneta
Berma2
Hay una cantidad de 7 elementos localizados en el contenedor:
RailDoubleTrack - 6304, son:
Rail
Rail
Rail
Rail
Ballast
Subballast
SubBase
Hay una cantidad de 12 elementos localizados en el contenedor:
Substructure - 6304, son:
Berma1
Berma2
Berma1
Talud-Terraplen
Talud-Desmonte.004
Cuneta
Cuneta
Berma2
```

```
Berma2
Talud-Terraplen
Berma1
Cuneta
Hay una cantidad de 7 elementos localizados en el contenedor:
RailDoubleTrack - 6305R, son:
Rail
Rail
Rail
Rail
Ballast
Subballast
SubBase
Hay una cantidad de 5 elementos localizados en el contenedor:
Substructure - 6305R, son:
Talud-Terraplen
Placa
Berma2
Cuneta
Berma1
Hay una cantidad de 5 elementos localizados en el contenedor:
RailDoubleTrack - 6307P, son:
Rail
Rail
Rail
Rail
Ballast
Hay una cantidad de 2 elementos localizados en el contenedor:
Substructure - 6307P, son:
Marco
Marco
Hay una cantidad de 5 elementos localizados en el contenedor:
RailDoubleTrackCANT_w_ExtraLayers, son:
Rail
Rail
Rail
Rail
Ballast
Hay una cantidad de 7 elementos localizados en el contenedor:
Substructure - 6305L, son:
Subballast
Cuneta
Placa.001
SubBase
Talud-Terraplen
Berma1
Berma2
Hay una cantidad de 7 elementos localizados en el contenedor:
RailDoubleTrack - 6302 (2), son:
Rail
Rail
Rail
Rail
Ballast
```

```
Subballast
SubBase
Hay una cantidad de 8 elementos localizados en el contenedor:
Substructure - 6302 (2), son:
Berma1
Cuneta
Talud-Terraplen
Berma2
Berma1
Cuneta
Talud-Terraplen
Berma2
```

Obtención de Referencias de Clasificación en el Modelo IFC

En los modelos BIM, los elementos a menudo están asociados con sistemas de clasificación estandarizados, como Uniclass 2015 o OmniClass. Estas clasificaciones proporcionan una estructura organizada para definir y categorizar los diferentes elementos dentro del modelo, como capas, materiales, o componentes.

Explicación del Código:

1. **Obtención de Referencias de Clasificación**:
   - La función `get_references(capa)` se utiliza para obtener todas las referencias de clasificación asociadas a un elemento, en este caso, la variable `capa` que podría representar una capa o muro.
2. **Iteración sobre las Referencias**:
   - Se itera sobre cada referencia de clasificación obtenida. Estas referencias son códigos alfanuméricos que identifican un tipo específico de clasificación dentro del sistema. Un ejemplo de código sería `Pr_30_59_99_02`.
3. **Obtención del Sistema de Clasificación**:
   - Utilizando la función `get_classification(referencia)`, se obtiene el sistema de clasificación al que pertenece la referencia. Este sistema podría ser, por ejemplo, Uniclass 2015 o cualquier otro estándar utilizado en la industria de la construcción.
4. **Impresión de Resultados**:
   - Para cada referencia de clasificación, se imprime tanto el código de referencia como el nombre del sistema de clasificación del que forma parte.

```
import ifcopenshell.util.classification

referencias = ifcopenshell.util.classification.get_references(capa)
for referencia in referencias:
    # Un código de referencia sería Pr_30_59_99_02
    print("La capa tiene una referencia de clasificación de:",
referencia[1])
    # Un sistema puede ser Uniclass 2015
    sistema =
ifcopenshell.util.classification.get_classification(referencia)
    print("Esa referencia es parte de un sistema:", sistema.Name)
```

```
La capa tiene una referencia de clasificación de: Ballast
Esa referencia es parte de un sistema: Civil 3D
```