

UNIVERSIDAD TÉCNICA FEDERICO SANTA MARÍA
DEPARTAMENTO DE INFORMÁTICA
VALPARAÍSO - CHILE



“APLICACIÓN MÓVIL DE MAPA UNIVERSITARIO, ENFOCADO EN ACCESIBILIDAD, FLEXIBILIDAD Y EFICIENCIA”

ESTEBAN ALONSO CARRILLO NARANJO

MEMORIA PARA OPTAR AL TÍTULO DE
INGENIERO CIVIL EN INFORMÁTICA

Profesor Guía: Carlos Castro
Profesor Correferente: Pablo Lezana

Agosto - 2024

DEDICATORIA

A José González Urra, mi pareja de vida y esposo. A mi madre, a mi padre, mis hermanas y
mis gatos.

AGRADECIMIENTOS

Agradezco la confianza, la paciencia, el apoyo y el invaluable amor que me entrega José. Se ha dedicado a cuidarme y protegerme, haciendo lo posible para estar conmigo y acompañarme a completar este desafío, juntos.

Quiero agradecer también el apoyo y el amor incondicional de mi familia, quienes se han preocupado en ayudarme cuando lo necesito y han dado mucho por mí y mi futuro.

Debo agradecer a mis profesores guía y correferente, quienes han entendido mis necesidades y han estado presentes. Han aportado constantemente a mi progreso en este proyecto.

Por último, agradecer el aporte desinteresado de Oscar Jalil, Director de Infraestructura y Jancob Olivares, Ingeniero de Gestión de Infraestructura. Además, del Centro Avanzado de Ingeniería Eléctrica y Electrónica (AC3E), FB0008, financiado por ANID.

RESUMEN

Resumen—La infraestructura de Campus Casa Central Valparaíso de la Universidad Técnica Federico Santa María es compleja y extensa, lo que dificulta el traslado en sus instalaciones. Como no existen soluciones prácticas y eficientes de navegación, en esta tesis se desarrolla una aplicación móvil diseñada para optimizar el desplazamiento y orientación dentro de la universidad. Se investiga y diseña la estructura y formato del mapa de la aplicación, considerando cómo se incluyen rutas, puntos de interés, edificios y sus niveles, resolviendo el problema de tridimensionalidad en aplicaciones de navegación. También se incorporan preferencias de usuario que interactúan con las propiedades de los caminos como: áreas verdes, ascensores, escaleras, entre otros. El proyecto se desarrolla con el framework *React Native* y se escribe en lenguaje de programación *TypeScript*. Se proyecta que este trabajo quede disponible para futuras actualizaciones, adaptaciones y mejoras, promoviendo su desarrollo y perfeccionamiento continuo.

Para acceder al código fuente, contactarme a través de mi casilla de correo electrónico: `esteban.carrillo@sansano.usm.cl`

Palabras Clave—Navegación dentro de Universidad, aplicación móvil, navegación accesible universal, optimización de rutas

ABSTRACT

Abstract—The infrastructure of the Campus Casa Central Valparaíso of Universidad Técnica Federico Santa María is extensive and complex, making it difficult to navigate its facilities. As there are no practical and efficient navigation solutions, this thesis develops a mobile application designed to optimize movement and orientation within university. The structure and format of the application's map are researched and designed, considering how routes, points of interest, buildings, and their levels are included, addressing the problem of three-dimensionality in navigation applications. User preferences are also incorporated, interacting with the properties of paths such as green areas, elevators, stairs, among others. The project is developed with the *React Native* framework and written in the programming language *TypeScript*. It is expected that this work will be available for future updates, adaptations, and improvements, promoting its continuous development and enhancement.

To access the source code, contact me through my email: `esteban.carrillo@sansano.usm.cl`

Keywords—Navigation inside University, mobile application, universally accessible navigation, route optimization

GLOSARIO

CC: Campus Casa Central Valparaíso.

GPS: *“Global Positioning System”*.

2D: Dos dimensiones.

3D: Tres dimensiones.

4D: Cuatro dimensiones.

SVG: *“Scalable Vector Graphics”*.

INS: *“Indoor Navigation System”*.

PLAN: *“Positioning, Localization, and Navigation”*.

JSON: *“JavaScript Object Notation”*.

GeoJSON: *“Geographic JavaScript Object Notation”*.

API: *“Application Programming Interface”*.

DA: Algoritmo de Dijkstra.

DFS: *“Depth-First Search”*.

BFS: *“Breadth-First Search”*.

BFA: Algoritmo de Bellman-Ford.

FWA: Algoritmo de Floyd-Warshall.

ÍNDICE DE CONTENIDOS

RESUMEN	IV
ABSTRACT	IV
GLOSARIO	V
ÍNDICE DE FIGURAS	VIII
ÍNDICE DE TABLAS	VIII
INTRODUCCIÓN	1
CAPÍTULO 1: DEFINICIÓN DEL PROBLEMA	2
1.1 PROBLEMA	2
1.1.1 TRASLADO INEFICIENTE	2
1.1.2 EXPERIENCIA DE VISITAS	3
1.1.3 ACCESIBILIDAD	3
1.1.4 ACCESO A SERVICIOS	3
1.2 OBJETIVOS	4
1.2.1 OBJETIVO GENERAL	4
1.2.2 OBJETIVOS ESPECÍFICOS	5
1.3 ALCANCE DEL PROYECTO	5
CAPÍTULO 2: MARCO CONCEPTUAL	6
2.1 ESTADO DEL ARTE	6
2.1.1 NAVEGACIÓN Y GEOLOCALIZACIÓN EN INTERIORES	7
2.2 ALGORITMOS Y EFICIENCIA	10
2.3 JSON y GEOJSON	12
CAPÍTULO 3: PROPUESTA DE SOLUCIÓN	14
3.1 DISEÑO	14
3.1.1 FORMATO DEL MAPA	14
3.1.2 AREAS	15
3.1.3 PISOS	16
3.1.4 ZONAS	17
3.1.5 PROPIEDADES	18
3.1.6 PREFERENCIAS	19
3.2 IMPLEMENTACIÓN	20
3.2.1 ORGANIZACIÓN	21
3.2.2 INFRAESTRUCTURA	24
3.2.3 PROYECCIÓN GEOGRÁFICA	26
CAPÍTULO 4: VALIDACIÓN DE LA SOLUCIÓN	27

4.1	ESTADÍSTICAS	28
4.2	PROYECCIONES	30
CAPÍTULO 5: CONCLUSIONES		32
ANEXOS		33
A	TURISMO LENTO	33
B	EJEMPLOS DE OBJETOS JSON	34
C	CAPTURAS DE PANTALLA DE DESMOS	37
D	VIDEOS DE EJEMPLO	39
REFERENCIAS BIBLIOGRÁFICAS		40

ÍNDICE DE FIGURAS

1	Búsqueda de punto de destino y generación de ruta en la aplicación <i>Campus Map</i>	7
2	Preferencias de ruta y resultado de trayecto en la aplicación <i>UniMaps</i>	8
3	Marco evaluativo para sistemas de posicionamiento en interiores	9
4	Ejemplo de un grafo en 2D, de nodos <i>a, b, c, d</i> y <i>e</i> conectados entre sí.	10
5	Ejemplo de un objeto JSON	12
6	Ejemplo de un objeto GeoJSON	13
7	Figuras de una red en 3D y en la segunda, corresponde a una versión plana de la primera	16
8	Grafo en 3D, donde la primera coordenada corresponde a un número arbitrario ("1" y "2")	17
9	Diagrama de procesos de la interacción de un usuario con la aplicación para generar un camino entre dos puntos	23
10	Ejemplo de área con más de un acceso, gracias a que es más sensato tener esas alternativas dado el espacio y estructura disponible	24
11	Captura de pantalla de la sección del archivo de diseño provisto por el Departamento de Infraestructura, con edificios demarcados como zonas distintivas	25
12	Captura de pantalla de la aplicación, mostrando una vista de usuario con las preferencias	27
13	Capturas de pantalla mostrando interacción táctil para realizar escalamiento y rotación del mapa	28
14	Capturas de pantalla dentro del edificio P. Se selecciona un área, luego otra área y se observa el camino generado	29
15	Capturas de pantalla dentro del edificio P, luego de seleccionar la opción para evitar bajar escaleras	30
16	Capturas de pantalla del camino desde el patio del edificio P hacia la sala P012, sin usar escaleras ni ascensores	31
17	Ejemplo en JSON de un objeto tipo "Area"	34
18	Ejemplo en JSON de un objeto tipo "Floor" que incluye objetos tipo Area	35
19	Ejemplo en JSON de un objeto tipo "Building" que incluye objetos tipo Floor	35
20	Ejemplo en JSON de las propiedades de una escalera, los nodos especificados en sus coordenadas y la conexión con el nodo de otro objeto	36
21	Captura de pantalla de <i>Desmos</i> con primer piso del edificio P de CC	37
22	Captura de pantalla de <i>Desmos</i> con áreas dibujadas sobre el plano	38
23	Captura de pantalla de <i>Desmos</i> con caminos dibujados sobre el plano	38

ÍNDICE DE TABLAS

1	Tipos de área, sin orden relevante	15
2	Propiedades de una arista.	19
3	Preferencias de usuario	20

4	Niveles de cada preferencia de usuario.	20
---	---	----

INTRODUCCIÓN

Las universidades, como centros de conocimiento y formación, han experimentado un notable crecimiento en términos de infraestructura y servicios. Los campus son cada vez más modernos y a menudo abarcan extensas áreas con múltiples edificios, instalaciones especializadas y zonas de recreación o estudio. Este crecimiento ha traído consigo desafíos significativos para la navegación y el acceso efectivo a los recursos, servicios e instalaciones disponibles.

Los estudiantes y funcionarios de la universidad, especialmente aquellos que son nuevos en el campus o con necesidades de accesibilidad universal, pueden enfrentar dificultades considerables para localizar aulas, oficinas, bibliotecas y otros puntos de interés. Esto también aplica a personas externas que sólo requieren estar un tiempo corto o realizan visitas esporádicamente, lo que implica que probablemente desconocen aún más el campus y sólo necesitan llegar al lugar de destino o, al contrario, desean explorar las instalaciones y aprovechar de observar y turistar.

Además, el avance en la tecnología móvil ha transformado la manera en que interactuamos con la información y los recursos. Las aplicaciones móviles se han convertido en una extensión integral de nuestras vidas diarias, proporcionando acceso instantáneo a una amplia gama de productos y servicios. En este sentido, una aplicación móvil que sirva como mapa universitario tiene el potencial de revolucionar la manera en que los miembros de la comunidad universitaria interactúan con su entorno, proporcionando una solución accesible y flexible para la navegación y la gestión de actividades.

Este proyecto se centra en el desarrollo de una aplicación móvil de un mapa del Campus Casa Central Valparaíso (CC) con un enfoque en accesibilidad, flexibilidad y eficiencia. La aplicación se diseña para permitir a los usuarios seleccionar puntos de inicio y destino dentro del campus universitario y visualizar una ruta optimizada entre estos puntos. Un desafío clave es obtener y manipular los datos estructurales de la universidad, para trabajar con ellos dentro de la aplicación. La flexibilidad también es una propiedad que se busca mantener en el contexto del código o del desarrollo de la aplicación en sí. Esto significa que cualquier persona que pueda continuar con la manutención y mejora (ya que funcionaría como un servicio), debería verse favorecida por el orden, utilidad y comodidad del proyecto en su estado actual, garantizado por herramientas desarrolladas y la estructuración del código.

CAPÍTULO 1

DEFINICIÓN DEL PROBLEMA

Es fácil notar que algunas construcciones de edificios o complejos estructurales no están diseñadas con el objetivo de que sean naturalmente simples de navegar. La planificación final suele mezclar aspectos de utilidad, arte, seguridad, eficiencia, solidez estructural y cualquier otro tópico esencial para una buena construcción.

El caso específico de la Universidad Técnica Federico Santa María, en CC ubicada en el cerro Placeres de la ciudad de Valparaíso, quienes llegan por primera vez o quienes aun no memorizan el espacio (estudiantes, profesores, funcionarios y visitas en general) pueden encontrarse con la dificultad de situarse en el lugar donde estén y también localizar su punto de destino. Esto puede ocurrir por una multitud de razones, como por ejemplo, puede ser debido a la gran cantidad de edificios y a la topografía de CC que se ubica principalmente en la ladera del cerro, o las diferentes formas de acceder a un mismo edificio desde diferentes pisos.

1.1. PROBLEMA

Existen variadas perspectivas respecto a cuales son las personas involucradas y el tipo de caso al que se aplica el problema de la navegación eficiente dentro de un emplazamiento. Por ejemplo, cuando se trata de una visita, puede aparecer el problema de proveer o guiar por un trayecto atractivo para que se aprecie el estilo estructural y elementos decorativos. De forma contraria, para un estudiante, se preferirían caminos cortos y accesibles (particularmente para personas discapacitadas), con fácil localización de servicios como lo son los baños, cafeterías, cajeros automáticos, etc.

Para simplificar estas situaciones en distintos componentes, a continuación se desglosarán en categorías.

1.1.1. TRASLADO INEFICIENTE

Desde una perspectiva general, es posible que el camino elegido sea más largo o complejo en comparación a otras alternativas, lo que pueda facilitar que la persona se desoriente y se arriesgue sufrir un accidente (por ejemplo, si accede a un lugar con maquinaria especializada), transcurra por un lugar expuesto al clima u otros casos similares. El gasto de energía y el uso del tiempo son aspectos de gran importancia, sumados a los casos descritos anteriormente.

Si se tratara de un estudiante, es probable observar prioridades distintas al moverse de un

punto a otro comparado a las de una visita. Por ejemplo, se podría desear un camino extremadamente corto para llegar rápidamente al lugar de destino, sin importar cuán arduo sea, o por el contrario buscar el camino que requiera del menor esfuerzo posible, o bien maximizar el disfrute del paisaje dentro del campus.

Si bien se describirán otros aspectos que están relacionados de alguna forma u otra al traslado desfavorable, este es el punto principal de este escrito.

1.1.2. EXPERIENCIA DE VISITAS

El diseño estructural, decoraciones, naturaleza y otros, generan una gran impresión en las visitas que llegan a la universidad, principalmente aquellas que desconocen las instalaciones o que deseen conocer lugares emblemáticos del campus. La elección de un camino o trayecto que maximice el atractivo turístico del mismo, es útil para mejorar la experiencia de la visita.

En el Apéndice A se detalla el concepto de “turismo lento”, que también se enlaza con este problema.

1.1.3. ACCESIBILIDAD

La accesibilidad en el contexto de las instituciones de educación superior, es un problema que lleva un tiempo en la conciencia pública. Particularmente, ocurre en instituciones como la nuestra, al estar ubicada en la ladera de un cerro y por tener edificios con más de 80 años desde que fueron construidos.

Por otro lado es imprescindible proveer un acceso equitativo a la educación, sin importar las capacidades físicas o cognitivas que puedan tener los estudiantes. Hoy en día, lograr esto implica grandes desafíos y las complejidades varían dependiendo de las necesidades que cada persona pueda tener. Por ejemplo, la accesibilidad física requiere de diseños modernos y flexibles en edificaciones, pero hoy en día muchas no han sido diseñadas con una infraestructura adecuada para transitar dentro de ellas o para utilizar las instalaciones indiscriminadamente. En general hay una falta de ascensores, rampas, baños y/o salas aptas para discapacitados, señaléticas y herramientas para discapacidades visuales, auditivas, entre otras.

1.1.4. ACCESO A SERVICIOS

Si bien ya se ha mencionado el obstáculo con la disponibilidad de servicios como los son los ascensores o baños para discapacitados, es también de interés la forma en que se pueden localizar y conocer sus ubicaciones de forma oportuna. Este problema se intensifica gravemente cuando se tratan de herramientas de emergencia como las alarmas de fuego, extintores,

desfibriladores y otros.

1.2. OBJETIVOS

El objetivo principal es crear una aplicación móvil con un diseño moderno e intuitivo, donde el usuario pueda interactuar con un mapa de plantas sucesivas (una por piso) en dos dimensiones (2D) de la universidad, específicamente CC, y que luego de seleccionar un punto de partida y uno de destino (diremos “marcadores”), se indique un camino apropiado para moverse entre los puntos elegidos utilizando las preferencias que haya configurado el usuario. Estas preferencias pueden ser de accesibilidad, como por ejemplo que sólo se sugieran trayectos sin escaleras, o al contrario, circunstanciales como evitar áreas exteriores o preferir uso de ascensores.

Muchos proyectos informáticos tienen un considerable enfoque en obtener un gran nivel de eficiencia, por ejemplo, con el uso de memoria o la rapidez de procesamiento. Para este proyecto se busca solo un sistema funcional, sin ser necesariamente la herramienta más rápida y optimizada en sí misma, sino que tenga la base fundamental para ser flexible y extendible tanto al momento de que un usuario final lo utilice, como cuando sea necesario modificar o desarrollar sobre esta. Este tipo de compromisos surgen de la dificultad de lograr muchas optimizaciones que suelen definirse como requisitos, pero que al mismo tiempo sean lo suficientemente abstractos y accesibles para no requerir detalles de bajo nivel que disminuirían su flexibilidad a la hora de mantener el proyecto.

Establecer esta meta orientada a un “punto central” respecto a la flexibilidad es necesario para que este trabajo pueda mantenerse a largo plazo, distinto a lo que lo que suele suceder con muchos proyectos de memoria. Se necesita que permanezca como un servicio y eso implica que haya mantenimiento y mejoramiento regular del código. Se constituye este compromiso para favorecer a quienes mantengan el proyecto en el futuro.

A continuación se formalizan los objetivos que se esperan lograr.

1.2.1. OBJETIVO GENERAL

Diseñar un sistema interactivo con un mapa de CC, que permita: flexiblemente modificar y añadir contenido al código del proyecto (como desarrollador) y obtener de forma efectiva una ruta suficientemente buena u óptima dependiendo de las preferencias y arquitectura dispuesta (como usuario), enfocado a personas que desconocen trayectos óptimos y los establecimientos de la universidad, además de los futuros desarrolladores que mantengan el proyecto.

1.2.2. OBJETIVOS ESPECÍFICOS

1. Definir preferencias que se pueden seleccionar por el usuario para acotar o relajar por cuales lugares se puede transitar (por ejemplo, si se prefiere o no usar escaleras, usar ascensores o recorrer por áreas verdes).
2. Definir y asignar sobre cada espacio transitable los atributos que deben existir, tal que estos interactúen directamente con las preferencias del usuario.
3. Diseñar y especificar el formato con que se representaría un mapa.
4. Crear y guardar un mapa de CC que esté en el formato establecido, con el que se pueda interactuar.
5. Diseñar y crear una aplicación móvil que utilice el mapa guardado para visualizar las áreas de interés e interactuar de forma táctil.
6. Permitir la creación de marcadores (punto de partida y de destino) en áreas definidas por el mapa o en partes de la superficie.
7. Utilizar algún algoritmo que obtenga un óptimo o suficientemente buen camino que transite de un marcador a otro (independiente del lugar, piso o edificio en que se encuentren), seleccionados por el usuario.
8. Añadir al algoritmo el efecto de seleccionar las diferentes preferencias del usuario.

1.3. ALCANCE DEL PROYECTO

Dadas las restricciones del tiempo y la dificultad de definir la magnitud del resultado esperado, se hace un acercamiento de tipo “*vertical slice*”¹.

Hay muchas disciplinas incluidas en este trabajo y todas pueden llegar a un punto extremadamente complejo, dependiendo de los objetivos. Para mencionar algunos ejemplos: se podría realizar pruebas y estudios con usuarios para luego analizar los resultados o, dentro de la misma área, estudiar y enfocarse en la experiencia de usuario y las interfaces. También se podría trabajar sobre los algoritmos para elegir caminos entre dos puntos y que estos sean los más eficientes y rápidos en términos de optimización. Sin embargo, como uno de los pilares principales que se definieron fue que la aplicación sea un punto intermedio, se vuelve necesario que no todas las aristas de este proyecto sean las mejores alternativas en el mercado, sino que sean suficientes y funcionales.

¹Del inglés: “corte vertical”, representa el desarrollo transversal a lo largo de todas o la mayoría de las características esenciales de un proyecto para que sea funcional, normalmente desde la perspectiva de un usuario. Contrario a esto, existe un corte horizontal que se realizaría para solo una característica en profundidad.

CAPÍTULO 2

MARCO CONCEPTUAL

2.1. ESTADO DEL ARTE

Así como cualquier problema ingenieril, existen incontables estrategias para cumplir estos objetivos. En general, el uso de mapas físicos y señaléticas podría proporcionar una buena solución inicial; sin embargo, para el propósito de este escrito, se busca una perspectiva tecnológica que permita que la herramienta sea simple, cómoda y de bajo costo de producción o utilización.

Un análisis inicial de las aplicaciones móviles de navegación más populares de este año, según [Curry, 2024], comprenden algunas como “Google Maps”, “Uber” y “Waze”. Sin dar relevancia a cómo se definió la popularidad entre estas aplicaciones, se puede observar que existe una diferencia sustancial entre los objetivos que estas buscan resolver y lo que se plantea en este escrito. Se observa que, en general, una aplicación de navegación y traslado cumplen un rol a una gran escala (global) y manejan recursos altamente variables como lo son tarifas relacionadas a peajes, cantidad de vehículos, estado o calidad del camino, etc.

A diferencia de lo anterior, un grupo de aplicaciones que se emplean en trabajar a una escala menor, como lo sería una aplicación de navegación dentro de un campus universitario, abarcan requerimientos notablemente distintos. Por ejemplo, es importante considerar que el sistema de localización “*Global Positioning System*” (GPS)¹ podría dejar de funcionar si un usuario de halla dentro de un edificio (normalmente tienen una precisión cercana a los 5 metros, lo que puede significar estar a un lado u otro de un muro, cambiando radicalmente las condiciones del problema para evitar un estado erróneo como este), o también, crecería la necesidad de tener un mayor detalle de la información del camino y de cuales son los pasos a seguir para trasladarse de un punto a otro, ya que los usuarios no se movilizarían en vehículos grandes y en largas distancias. Por esta razón, se buscan y analizan aplicaciones móviles de navegación de universidades, para examinar la forma en que se intentan resolver los problemas mencionados en el capítulo anterior. Cabe mencionar que no hay necesariamente una alta disponibilidad de estas, ya que algunas solo utilizan un mapa que no es interactivo [University of Oregon, 2012], no permiten acceso a quienes tengan un correo institucional de la universidad asociada [Collegis Education, 2019] y otras manejan sólo el traslado entre edificios de la universidad y de las calles externas para entrar a los establecimientos sin considerar el interior de los establecimientos (ver Figura 1).

Una aplicación a la que se pudo acceder, es “*UniMaps*” para *Universität Bielefeld*². Cumple en gran medida con los objetivos propuestos; sin embargo, no presenta muchas opciones para las preferencias de usuario para la generación de rutas (ver Figura 2). Notamos que sólo

¹Sistema de navegación por satélite que permite determinar la ubicación geográfica precisa de dispositivos móviles y celulares.

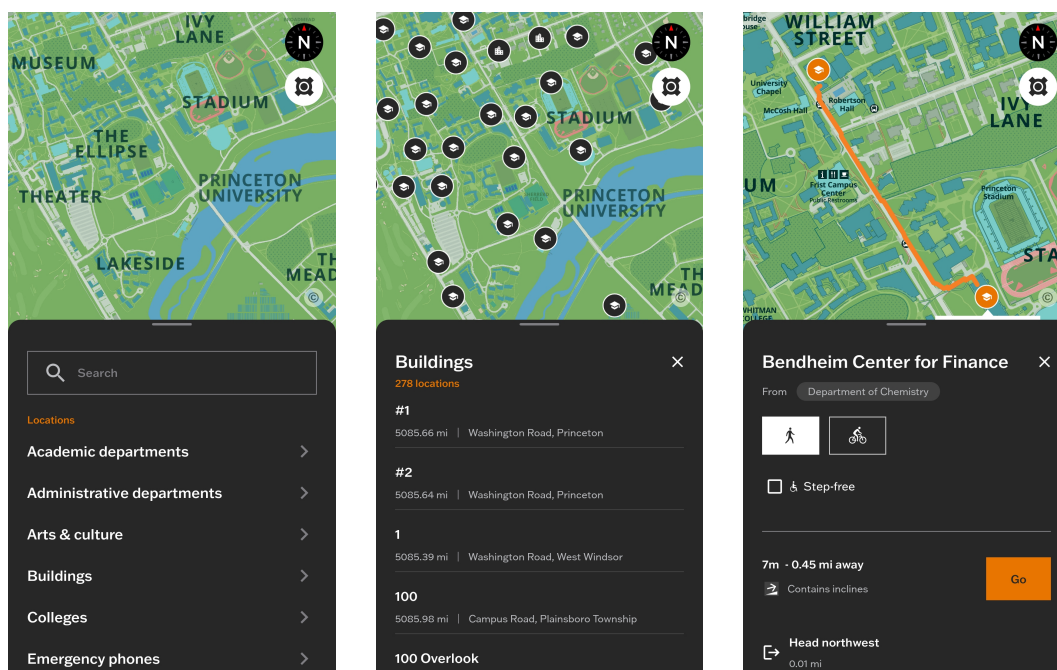


Figura 1: Capturas de pantalla (de izquierda a derecha): búsqueda por categorías de un punto de destino, luego la selección del punto de interés según la categoría “edificios” y, finalmente, ruta entre dos puntos. Fuente: [Princeton University, OIT AS, 2023].

se incluyen tres elementos que se pueden evitar durante un trayecto: escaleras, ascensores y multitudes.

2.1.1. NAVEGACIÓN Y GEOLOCALIZACIÓN EN INTERIORES

Si bien, para el caso de este proyecto, no es una necesidad imperante que se pueda localizar al usuario en tiempo real, es importante analizar las formas en que se ha podido resolver esto y qué desafíos traen esas metodologías.

En [Huang y Gartner, 2009] se realiza un estudio de varios sistemas de navegación en interiores (en inglés “*Indoor Navigation System*” o INS), donde se precisaron categorías que comprendería el proceso de inicio a fin para cualquiera de estos (ver Figura 3). Se destaca, para algunas de estas categorías, que se formó un estándar:

- “Posicionamiento en interiores”: Corresponde al uso de herramientas externas o heurísticas para encontrar la ubicación del usuario más precisa posible. En esta, se halla que muchos sistemas utilizaron señales de radio (como WiFi y Bluetooth).

²Universidad de Bielefeld en Bielefeld, Alemania.

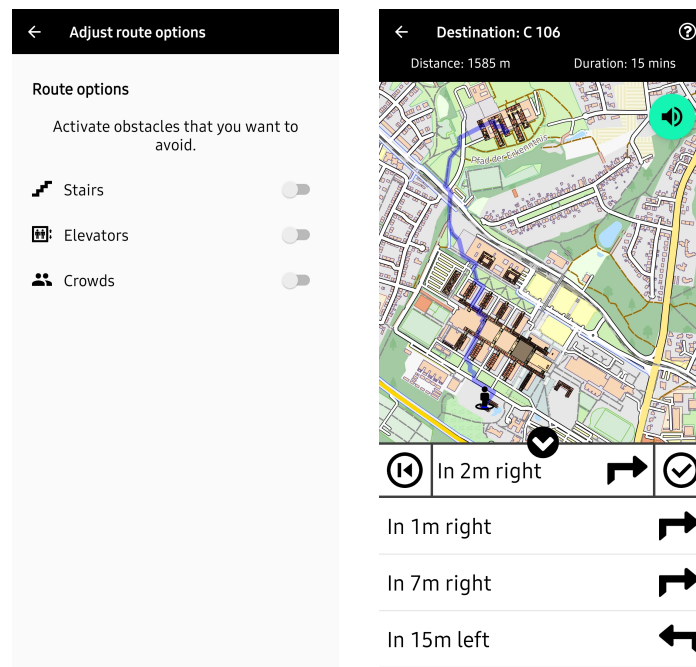


Figura 2: Capturas de pantalla (de izquierda a derecha): primero la selección de preferencias para cada ruta generada y luego el trayecto generado al seleccionar dos puntos. Fuente: [Zentrale Anlaufstelle Barrierefrei, 2019].

- “Adaptación contextual”: Diseñada para ese estudio, trata sobre analizar qué información o datos situacionales o circunstanciales son utilizadas (e.g. el clima, tiempo o preferencias de usuario), incluyendo cuales fueron las estrategias del manejo de estos recursos para optimizar la adaptabilidad. En esta categoría, fue común que los sistemas se enfocaran en justificar “cuáles” recursos y “dónde” se utilizaban para adaptar el proceso de navegación, pero no las razones que explican “por qué” o “cómo” (que se consideraron como dimensiones para el análisis).
- “Comunicación de la ruta”: Se evalúa la presentación, estética y formato en que se expresa la información relevante del trayecto e instrucciones al usuario, en donde se utilizaron mayormente una colección típica de formatos como mapas, signos o señales, textos y voz, pero se necesitó evaluar la eficiencia y efectividad de aquellos métodos.

Si bien aun no pasan 20 años desde la publicación, como hubo sustanciales progresos para todo lo relacionado a tecnología de posicionamiento, localización y navegación (PLAN, de sus siglas en inglés), se han adoptado nuevas tecnologías, como por ejemplo, en [Kunhoth *et al.*, 2020, Farahsari *et al.*, 2022, Pascacio *et al.*, 2021].

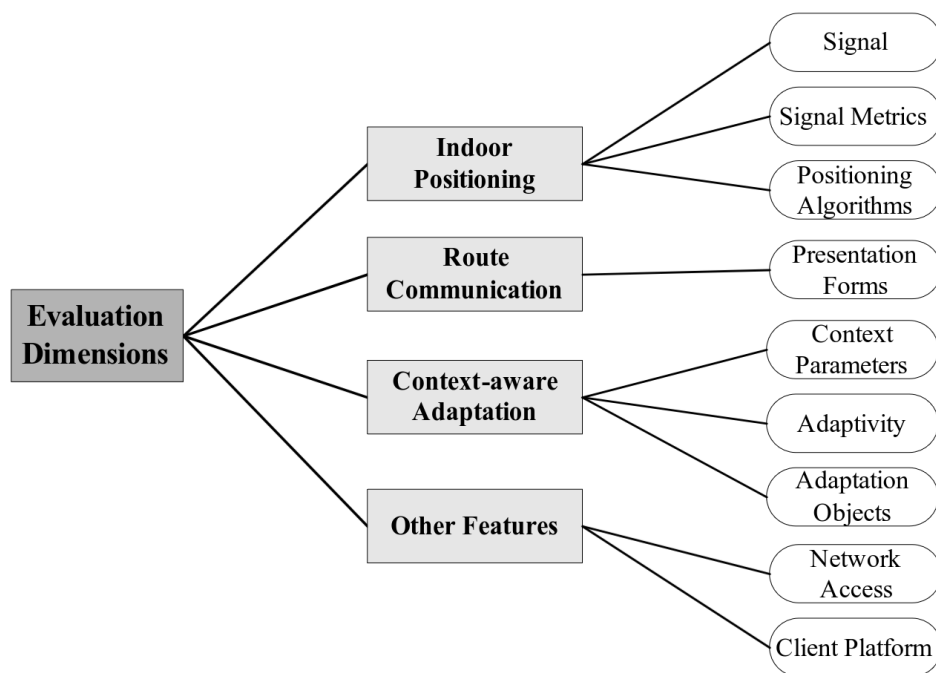


Figura 3: Marco evaluativo para sistemas de posicionamiento en interiores. Fuente: [Huang y Gartner, 2009].

2.2. ALGORITMOS Y EFICIENCIA

Para propósito de este escrito, se define el concepto de un “grafo” o “red”, como un conjunto de puntos o nodos que están conectados entre sí. Cada nodo corresponde a una coordenada en cualquier dimensión y, en este caso, en 2D o tres dimensiones (3D). La conexión entre un nodo y otro (diremos que son “vecinos”) se representa con una arista. Cada arista tendrá un peso asociado que representa la distancia entre ambos nodos; sin embargo, esto no es un requerimiento para ser un grafo. Un ejemplo de un grafo de nodos posicionados en 2D puede verse en la Figura 4.

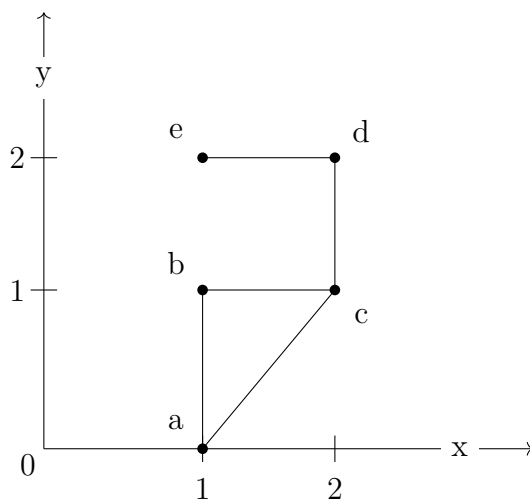


Figura 4: Ejemplo de un grafo en 2D, de nodos *a*, *b*, *c*, *d*, y *e* conectados entre sí. El nodo *c* tiene tres vecinos (*a*, *b* y *d*) y el nodo *e* sólo tiene a *d* como vecino. Fuente: Elaboración propia.

Cuando se trata de buscar el camino más corto entre dos puntos de un mapa, se puede resolver considerando que “el camino” en verdad corresponde a nodos y sus aristas, tal que el punto de inicio y fin son parte de un grafo. Existen algoritmos que permiten definir un nodo de partida y que encuentran la ruta más corta hasta otro nodo, visitando cada uno o sólo algunos.

Como una idea trivial, se puede comenzar desde el nodo inicial y buscar por cada vecino si es que eventualmente se puede llegar al nodo final, calculando el costo del viaje entre ellos. Esto resulta extremadamente costoso y potencialmente inservible si no se guardan qué nodos fueron visitados (sin una “memoria”), ya que podría ser una búsqueda interminable que queda atrapada en ciclos sin poder finalizar.

Para encontrar los caminos más eficientes, dependiendo del tipo de costo que se use (para efectos de esta sección, diremos “cortos” independientemente), se analizan diversos algoritmos:

- Algoritmo de Dijkstra (DA): Permite determinar el camino más corto, siempre y cuando

no se usen pesos negativos. Su uso es ideal para grafos de tamaño mediano o pequeño, ya que usa una cola de prioridad para explorar las aristas de menor peso, pero visita todas las conexiones y eso entrega el camino más eficiente de todas las alternativas [Dijkstra, 1959].

- Algoritmo A^* (“A asterisco” o “A estrella”): Se puede considerar una extensión de DA y normalmente encuentra el mejor camino de un nodo a otro, sin la necesidad de visitar todos los nodos del grafo. Suele desempeñarse mejor que DA dado que su uso en grafos de tamaño grande tiene buenos resultados y en un tiempo acotado. Se necesita definir una heurística que determine cual nodo siguiente se debe visitar y cuales descartar, por lo que la calidad del algoritmo yace en esta heurística [Hart *et al.*, 1968, Dechter y Pearl, 1985].
- “*Depth-First Search*” (DFS): Realiza una búsqueda en forma “profunda”, avanzando por cada primer vecino que no ha sido visitado hasta necesitar devolverse (realizando “*backtracking*”). Suele ser eficiente en grafos que son poco densos (es decir, cada nodo tiene pocos vecinos), pero tiene costo un asociado al número de veces que debe retornar o devolverse [Tarjan, 1972].
- “*Breadth-First Search*” (BFS): Al contrario que DFS, la búsqueda es “a lo largo”, visitando cada vecino disponible para un mismo nodo antes de avanzar. Normalmente es apropiado usarlo para grafos sin pesos o para encontrar nodos conectados; sin embargo, tiene un gran costo de exploración de caminos si la meta se encuentra a una cantidad de nodos de distancia [Zhou y Hansen, 2006].
- Algoritmo de Bellman-Ford (BFA): Similar a DA, puede obtener el menor costo de un nodo hacia el resto, pero este puede trabajar con pesos negativos y con caminos cíclicos [Dela Cruz *et al.*, 2016].
- Algoritmo de Floyd-Warshall (FWA): Con una complejidad temporal alta (a cambio de la complejidad en el uso de espacio o memoria), puede encontrar los pares de caminos más cortos entre todos los nodos [Dela Cruz *et al.*, 2016].

Es claro que existen muchos otros algoritmos, en particular porque se pueden mezclar varios en un mismo proceso (como en [Dinitz e Itzhak, 2017] que define un híbrido de BFA y DA). La elección del algoritmo dependerá de varios factores y uno de ellos es la complejidad en sí misma de implementarlo, ya que se trabajarán nodos en algún formato y se necesitará encontrar una manera apropiada de visitarlos. Por otro lado, es importante que encuentre un buen resultado en un tiempo acotado, ya que no es necesario obtener el mejor trayecto de un punto a otro, pero sí debe ser suficientemente bueno dadas las preferencias de usuario y las condiciones del camino. Finalmente, se determina que la mejor opción para este proyecto, es DA, ya que es fácil de implementar, es bastante documentado y sirve como un buen algoritmo inicial para determinar que la aplicación funciona, independiente de su eficiencia.

2.3. JSON y GEOJSON

Existe un formato de texto para el intercambio de datos que es ligero, legible e intuitivo llamado *“JavaScript Object Notation”* (JSON). Cumple el rol de ser fácil de analizar tanto por personas como por máquinas por su sintaxis simple y define varios tipos de elementos que cumplen diferentes metas. Algunos de estos son: números (*“number”*), texto (*“string”*), booleanos (*“boolean”*) y el tipo especial *“null”*. Los otros dos que existen, son los arreglos (*“array”*), que pueden contener una lista de cualquiera de los tipos disponibles de JSON y luego los objetos (*“object”*), que corresponden al par *“llave-valor”*, usando un texto único para la llave que la relaciona a su valor, el cual puede ser cualquier tipo JSON. Un ejemplo de un objeto JSON se dispone en la Figura 5.

```

1  {
2    "name": "Esteban",
3    "age": 25,
4    "car": null,
5    "roles": [
6      { "id": "student", "city": "Valparaíso" },
7      { "id": "employee", "city": "Santiago" }
8    ]
9  }

```

Figura 5: Ejemplo de un objeto JSON. Fuente: Elaboración propia.

Dada la necesidad de buscar herramientas y métodos que faciliten, flexibilicen y estandaricen el procesamiento de datos relacionados a características geográficas y espaciales, es importante mencionar el formato *“Geographic JavaScript Object Notation”* (GeoJSON).

Este se basa en JSON, para guardar diferentes características *“geoespaciales”* y se encarga de categorizarlas de forma sencilla y legible, pero útil para procesar datos al igual que JSON. Existen varios tipos de objetos geoespaciales o geometrías, pero los principales son los siguientes:

- Punto (*“Point”*): Es un punto en 2D específico, con coordenadas de latitud y longitud.
- Línea (*“LineString”*): Varios puntos en secuencia, conectados por segmentos de línea.
- Polígono (*“Polygon”*): Área delimitada por una serie de puntos conectados que forman un borde cerrado.

Además, se establecen dos colecciones: *“GeometryCollection”* que agrupa cualquier mezcla de las geometrías anteriores y *“FeatureCollection”* que contiene la colección anterior y además incluye propiedades (un objeto JSON) que se asignan al conjunto entero.

En el contexto de navegación dentro de una universidad, GeoJSON resulta particularmente útil para manejar elementos de infraestructura (edificios, áreas, accesos y otros puntos de interés) y su representación visual, definir rutas de transporte o caminos transitables en general y facilita la integración a sistemas informáticos dada su flexibilidad de procesamiento. Se observa un ejemplo de un *FeatureCollection* para un área en la Figura 6.

```

1  {
2    "type": "Feature",
3    "geometry": {
4      "type": "Polygon",
5      "coordinates": [[
6        [0, 0], [2, 1],
7        [4, 5], [2, 5],
8        [1, 4], [0, 0]
9      ]]
10   },
11   "properties": {
12     "name": "Juegos",
13     "description": "Área de juegos",
14     "materiales": 5,
15   }
16 }

```

Figura 6: Ejemplo de un objeto GeoJSON. Fuente: Elaboración propia.

CAPÍTULO 3

PROPUESTA DE SOLUCIÓN

3.1. DISEÑO

El proyecto nace originalmente por la necesidad de tener más libertad de elección a la hora de viajar de un punto a otro usando una aplicación de navegación. Es factible que, para un usuario, la rapidez o economía del trayecto (como en el caso de los peajes, por ejemplo) sea un factor despreciable. En cambio, se podrían considerar un sinnúmero de propiedades distintas del trayecto, como el estado del camino, las condiciones de la persona, el clima, entre otros. En otras palabras, se busca ampliar el concepto de un “camino ideal”.

Primero, se plantea la forma en que se puede definir los diferentes lugares de un mapa, y lo que signifique un camino de un punto a otro.

3.1.1. FORMATO DEL MAPA

En el contexto de la universidad y CC en particular, se entiende que un usuario tendrá como objetivo moverse desde un lugar a otro. Esto significaría que cualquiera de los puntos que se puedan seleccionar como inicio o fin pueden ser: una sala de clases, un patio, un kiosko, entre otros, o cualquier lugar transitable.

Para formular el diseño del mapa, se definen nodos que representan una localización en particular, los cuales se conectan a otros nodos para generar un grafo. La conexión de un nodo a otro significa que se puede transitar desde el primero al segundo y la distancia debe corresponder a una métrica real, es decir, la posición correspondería a la latitud y longitud. Cabe mencionar que cada nodo debe tener una o más conexiones, ya que un nodo sin vecinos, sería un punto aislado donde no habría forma de llegar o salir de él.

Es importante diferenciar entre aquellos datos que representan los caminos posibles de recorrer y las áreas que corresponden a puntos de interés, como los mencionados anteriormente. Diferenciar entre áreas y caminos, sirve para la interacción del usuario, ya que permite seleccionar un área como marcador conociendo el lugar por nombre, sin la necesidad de reconocer el punto exacto en el mapa.

3.1.2. AREAS

Un área se define como un polígono convexo¹ que encierra un lugar de interés. Cada área puede cumplir uno de los siguientes roles dispuestos en la Tabla 1

Tabla 1: Tipos de área, sin orden relevante

Fuente: Elaboración Propia.

Área	Descripción
Computador	Salas de computación o “laboratorios”
Dibujo	Salas de dibujo
Clases	Salas de clase o cátedras
Ocio	Salas de juegos u ocio
Estar	Espacios abiertos de estar
Estudio	Salas de estudio
Profesor	Salas para profesores
Multiuso	Salas multiuso o multipropósito
Auditorio	Salas auditorio
Escaleras	Área de escaleras (que suben o bajan)
Baño	Baños (hombre, mujer o unisex)
Camarines	Camarines, vestidores y duchas
Ascensor	Áreas de ascensores
Estacionamiento	Estacionamientos (autos, bicicletas, etc.)
Cruce peatonal	Cruces de peatones
Portería	Porterías del campus (accesos)
Patio	Sectores al aire libre de recreación
Comedor	Sectores de comedor con mesas
Kiosco	Negocios (venta de comida o artículos)
Restringido	Áreas restringidas (no accesibles)
Administración	Áreas de administración, coordinación o mesones de atención
Banco	Bancos o cajeros automáticos
Oficina	Oficinas o salas de reuniones
Médico	Servicios de atención médica (no urgentes)
Emergencia	Extintores, desfibriladores y servicios de urgencia

Un área existe para cumplir dos propósitos: primero es permitir una ayuda visual para identificar su existencia y diferenciar su propósito de otras (usando colores, íconos o nombres) y luego, proveer la capacidad de interactuar directamente con el área misma, asistiendo a la creación de los marcadores. En este proyecto, la variedad de tipos de área es mayoritariamente visual (exceptuando algunos, como por ejemplo las escaleras, baños y ascensores, que tienen comportamientos únicos); sin embargo, facilita la edición y creación de nuevos tipos de área y comportamientos propios.

¹Tipo de polígono donde sus ángulos interiores son todos menos de 180°, es decir, no tiene “indentaciones”.

La información de un área se maneja con un objeto de tipo JSON que diremos que tiene la estructura “*Area*” (en inglés). Es decir, un área debe cumplir con el formato *Area*, el cual necesita: un identificador único, un polígono que defina su forma, una lista de puntos de acceso al área (pueden ser cero o más) y otros elementos que establezcan cómo se debe visualizar e interactuar (el color, nombre del área o ícono, si es seleccionable, etc.)

A modo de ejemplo y en forma simplificada, se observa en la Figura 17 del Apéndice B un objeto *Area* específico usando JSON¹.

3.1.3. PISOS

En este punto, es importante notar que existe un problema de tridimensionalidad. En un edificio, la posición de un nodo (diremos con coordenadas x e y) en 2D para un piso cualquiera, será exactamente el mismo que un nodo con las mismas coordenadas pero en un piso de arriba o abajo. Si sólo se consideran estas coordenadas 2D como tal, se acompleja la interfaz visual para diferenciar en qué piso se encontraría el usuario o el nodo actualmente y, la interacción para seleccionar una ubicación, se vuelve prácticamente ineficiente. Esto ocurre a pesar de ser posible manejar un grafo de nodos en 3D “aplastados” en 2D, similar a un hilo posicionado en un plano (ver Figura 7). Es decir, a pesar de tener nodos que existen en un mismo lugar, las conexiones individuales se mantienen (en la figura anterior, se entendería esto como la imposibilidad de pasar del nodo b al d directamente con un sólo “paso”).

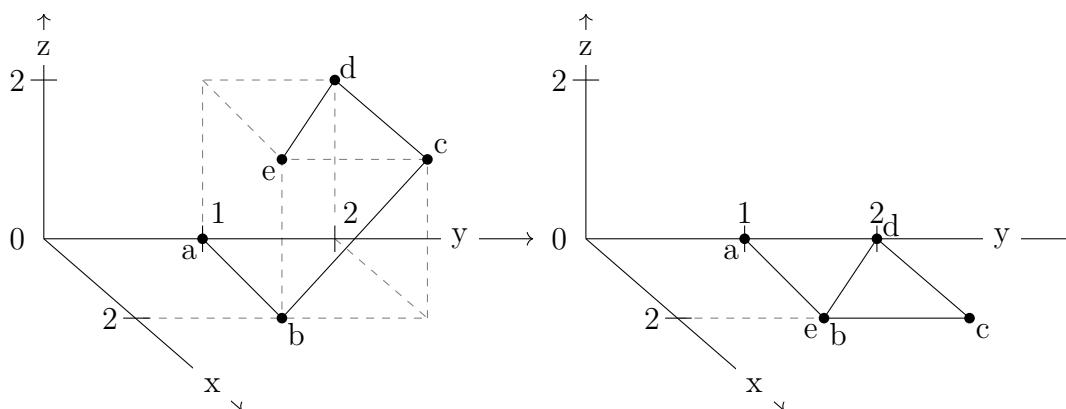


Figura 7: La figura izquierda contiene una red en 3D que conecta secuencialmente los nodos a , b , c , d , y e . En la derecha, se ve una versión plana del anterior (notar que los nodos b y d comparten posición, pero no las conexiones a nodos vecinos). Fuente: Elaboración propia.

Una solución trivial, es incluir una 3ra coordenada z ; sin embargo, se dificulta sustancialmente la forma de conseguir data de gran precisión, que puede empeorar los resultados si se considerase algún valor incorrecto (ya que un cálculo de distancias tomará en cuenta

¹Este y los siguientes ejemplos escritos en JSON, en verdad se calificarían como inválidos por contener comentarios usando notación “//”; sin embargo, se describirá el formato como JSON independiente del caso para facilitar la explicación, asumiendo que los comentarios no son parte de la data.

pequeñas variaciones). Esto, además, perjudica la visualización y el procesamiento de las rutas, ya que pueden existir desniveles significativos para un mismo piso, los cuales podrían erróneamente como nodos en pisos diferentes. Por lo tanto, se busca un compromiso que facilite la gestión visual y conceptual para un usuario y el tratamiento de los datos para un desarrollador.

Se resuelve esto acotando el trabajo que debe suplir esta 3ra coordenada, ya que no se requiere un valor exacto y sólo se necesita distinguir un piso de otro en forma “lógica”. En otras palabras, cada nodo tendrá además de su posición en 2D (x, y) , un identificador arbitrario f (por “*floor*” en inglés) que indica a qué piso corresponde el nodo independiente de la altura real (ver Figura 8), obteniéndose (f, x, y) .

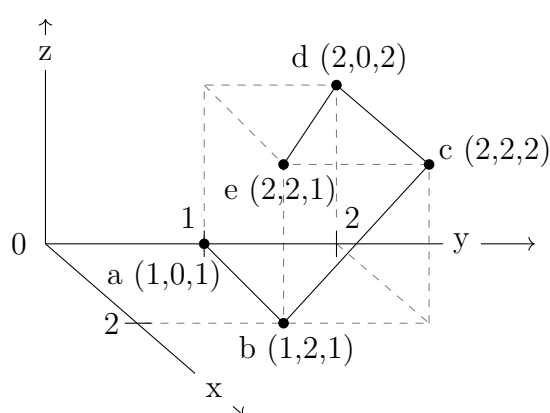


Figura 8: El grafo de nodos a, b, c, d y e , conectados entre sí, contiene nodos donde la primera coordenada corresponde a un número arbitrario (“1” y “2”). Fuente: Elaboración propia.

El costo de cambiar de un piso a otro dependerá del método usado (por ejemplo, al usar escaleras o ascensores) y no de la distancia vertical.

Para trabajar sobre la data de un piso se considera la siguiente estructura en JSON, definiendo el tipo “*Floor*” (ver Figura 18 en el Apéndice B). Para este, se requiere un polígono que marque el borde del piso completo (es decir, el espacio que incluye todas las áreas y lugares transitables) y aquellas áreas que existen en el piso actual del tipo *Area*.

3.1.4. ZONAS

Siguiendo la idea anterior se puede observar el caso en que dos coordenadas que existen una sobre otra (en diferentes pisos), serían incongruentes si estas no pertenecieran al mismo edificio. Un caso específico en el que esto ocurre, es en el edificio P de CC. En el piso -1, existe un túnel que cruza debajo de la calle, donde el “piso superior” no correspondería a un espacio dentro del edificio, ya que está en el exterior y fuera del campus. Cuando se trabaje con los nodos que corresponden a este sector, se necesitará una forma de diferenciar entre los pisos de diferentes lugares o edificios.

Se soluciona definiendo una zona “global” que corresponde a la vista general del mapa, en otras palabras, el campus completo y luego una zona para cada edificio (un identificador z , proveniente de inglés: “zone”). De esta forma, se establece que un nodo ahora contiene 4 componentes o cuatro dimensiones (4D): un identificador z para la zona, un identificador f para el piso y las dos coordenadas x e y . Dicho de otro modo, corresponde a (z, f, x, y) .

Es importante aclarar que para la zona global se utilizará el piso “1”, ya que puede ser arbitrario para cada una. El valor debe ser coherente con la verticalidad “lógica” y no necesariamente con un cambio de altura real. Esto significa que las entradas o accesos de los edificios que se encuentran en un piso cualquiera como un piso “2”, pero que se conectan a la zona global, estarían entonces definidos con una conexión al piso “1” de esta, independiente de que estén físicamente a la misma altura. Esto se observa con más detalle en las figuras descritas en el Apartado 3.2.3.

Un edificio que requiere una visualización interna de sus pisos, se formula con un objeto JSON de tipo “*Building*”. Si bien la zona global y cada edificio son todos “zonas”, existe una jerarquía clara donde cada edificio existe dentro de la vista global.

Un *Building* consiste en un identificador único del edificio y cada piso que se relaciona con su dato *Floor* correspondiente. Al igual que los tipos anteriores (*Area* y *Floor*), existen más atributos que se omiten, pero se simplifican porque son principalmente para procesar estos datos gráficamente.

En la Figura 19 del Apéndice B se observa un ejemplo para un edificio.

3.1.5. PROPIEDADES

Como fue descrito anteriormente, las rutas o caminos consisten en un grafo de nodos conectados entre sí. Por lo tanto cada arista contiene atributos o propiedades que deberán enlazarse con las preferencias de usuario que se detallan en la sección siguiente.

Principalmente contienen la información de la zona y el piso donde se encuentran las aristas, pero si corresponden a un elemento especial (por ejemplo: escaleras, ascensores, entradas o salidas, etc.), estos tendrían otras propiedades como se describen en la Tabla 2.

La propiedad llamada “conn” es importante, ya que identifica que el último nodo listado en “coordinates” es el que forma una arista con el que se definió en “conn”. Esto es para informar al sistema que ocurre un cambio de zona o de piso y que el camino continuaría desde ese nodo; sin embargo, el contenido de esta propiedad sólo sirve para referenciar y no es en sí un nodo. Además, se describe con la misma información que se detalló anteriormente (z, f, x, y) . En el Apéndice B hay un ejemplo de esto, con nodos de una escalera en la Figura 20.

Tabla 2: Propiedades de una arista.

Fuente: Elaboración Propia.

Propiedad	Descripción
"zone"	Identificador de la zona
"floor"	Identificador del piso
"coordinates"	Lista de posiciones de nodos (x, y)
"conn"	Identificador de la zona, piso y tipo de conexión hacia el segundo nodo de la arista (pueden ser: "acceso", para salir o entrar a una zona nueva y "escalera" o "ascensor" para cambiar de piso)
"covered"	Si está bajo techo o si está cubierto
"accessible"	Si es un camino accesible universal ¹
"stairs"	Si la arista es una escalera
"elevator"	Si la arista es un ascensor
"spacious"	Si la arista se encuentra en un espacio amplio (un ascensor no lo cumpliría y la definición es arbitraria)
"green"	Si la arista está aledaña o sobre un área verde o de vegetación
"area"	Si la arista tiene un nodo que existe dentro de un área, este se considera como "acceso" a él y tendría el identificador correspondiente (si no, se usa un valor nulo)

3.1.6. PREFERENCIAS

Un diseño inicial considera que el camino ideal de un punto a otro se da por el costo asociado a la distancia al recorrer por cada nodo. Sin embargo, un aspecto esencial del proyecto es considerar diferentes preferencias del usuario al momento de generar un camino entre los dos puntos seleccionados. Esto significa que, dependiendo de las preferencias, el costo final dependerá de qué elementos se hallen y cómo se afectado por el valor de la preferencia.

Cada preferencia de usuario debe relacionarse con algún atributo delineado anteriormente. Es decir, para definir una preferencia de usuario, se necesita trabajar sobre algún o algunos atributos en cada arista para afectar el peso que se haya calculado por la distancia física.

Las preferencias establecidas se observan en la Tabla 3. Cada uno de estos puede estar en uno de cinco estados que proveen un mayor control en la selección del camino ideal para un usuario (ver Tabla 4).

La utilidad de contar con más de un nivel (distinto a tener sólo "evitar" en algún elemento del camino, como se vió anteriormente en la Figura 2), es para mejorar la libertad y control de elección del usuario. Se vuelve menos restrictivo y no garantiza que el camino entre los marcadores contenga o no la preferencia seleccionada, pero sí la favorece.

De esta forma, por ejemplo, si un usuario desea no subir escaleras y favorecer áreas verdes,

Tabla 3: Preferencias de usuario

Fuente: Elaboración Propia.

Propiedad	Descripción
Vía accesible universal	No se cruzan elementos que no son accesibles en forma universal
Vía techada o cubierta	Existe un techo o algo que cubre
Vía con amplio espacio	El espacio no es pequeño o apretado, como un ascensor
Vía con áreas verdes	Hay vegetación o es un área verde
Vía subiendo escaleras	El trayecto sube por escaleras
Vía bajando escaleras	El trayecto baja por escaleras
Vía usando ascensores	El trayecto utiliza ascensores

Tabla 4: Niveles de cada preferencia de usuario.

Fuente: Elaboración Propia.

Nivel	Descripción
Prioritario	Provee el menor costo asociado a la preferencia, tal que se priorice su valor
Más	Rebaja el costo con menor intensidad, permitiendo que otras alternativas puedan ser útiles
Desactivado (prioridad normal)	No cambia el costo obtenido por la distancia (este es el nivel seleccionado por defecto)
Menos	Aumenta el costo en la misma razón que el nivel “Más”
Nada	Establece que el costo es de valor infinito, por lo que se vuelve imposible transitar por él

entonces puede seleccionar que la preferencia “Vía subiendo escaleras” esté en un estado “Nada” (asignándole un peso infinito, tal que nunca se pueda generar un camino por escaleras que suban) y “Vía con áreas verdes” esté en “Más” para disminuir el peso de caminos que transiten por esos lugares.

3.2. IMPLEMENTACIÓN

Ya que se trata de una aplicación móvil, se busca una plataforma de desarrollo que sea sólida, probada, familiar y cómoda. Se determina que la mejor opción es utilizar “*React Native*”. Este es un *framework*¹ que facilita la creación de aplicaciones móviles, usando como base el

¹Conjunto de herramientas y principios que facilitan el desarrollo de aplicaciones al proporcionar una estructura predefinida y componentes reutilizables.

lenguaje de programación *JavaScript*¹ y los elementos y paradigmas de *React*².

Gracias a que *React Native* tiene una enorme cantidad de librerías (funciones y componentes reutilizables), se puede aprovechar de encontrar herramientas que ya fueron creadas que facilitan el desarrollo de la aplicación.

La principal librería que fue necesaria incluir, corresponde a “*maplibre-react-native*” en [MapLibre, 2022], que provee la base esencial para visualizar y añadir elementos a un mapa geográfico interactivo. Existe además la librería “*geojson-path-finder*” en [Per Liedman, 2016], la cual sirve para encontrar el camino más corto entre dos puntos de una red 2D, usando el formato GeoJSON y con el DA discutido previamente (el hecho de que ya exista su implementación favorece la elección del algoritmo). Sin embargo, esta no fue utilizada tal y como se creó, sino que se fueron reemplazando partes del código para que se adaptaran al diseño de los nodos en 4D.

3.2.1. ORGANIZACIÓN

Para detallar la organización del código que permite crear la aplicación móvil de este proyecto, se describirá jerárquicamente la estructura de los archivos. Existe un archivo llamado “*App.tsx*”, que cumple el rol de ser el punto de entrada cuando la aplicación inicia (define configuraciones y estados iniciales, elementos de navegación, estilos, etc.). Este exporta un componente³ raíz “*App*”, que es el contenedor principal e inicial de todo el resto de la aplicación. Junto a este archivo, hay una carpeta “*src*” que contiene las funciones, vistas, herramientas y componentes principales. A continuación se detalla cómo se estructuró el contenido de esta carpeta:

- *vistas/*: carpeta de componentes que se encargan de agrupar lógicamente aquellos elementos que se visibilizarían en una vista o “página”.
 - *exterior.tsx*: archivo que muestra el mapa con los objetos de la “vista global” o “zona global” que se mencionaba en secciones pasadas.
 - *interior.tsx*: para un edificio, expone un sólo piso a la vez, pero contiene botones que permiten al usuario poder cambiar entre ellos. Esta vista recibe como parámetros cual edificio debe mostrar.
 - *opciones.tsx*: archivo que grafica la lista de preferencias de usuario. Se accede a esta vista desde cualquiera de las otras dos, usando un botón.

¹Lenguaje de programación principalmente para el desarrollo web. Permite crear páginas web interactivas y dinámicas, y es esencial para el desarrollo de aplicaciones web modernas.

²Herramienta basada en *JavaScript* que facilita la construcción de interfaces de usuario interactivas y eficientes, la creación de componentes reutilizables, la gestión del estado y la reactividad de la aplicación web o móvil.

³Unidad modular y reutilizable, que define una parte de la interfaz de usuario. Similar a una función o clase que acepta argumentos o propiedades (llamados “*props*”) y retorna elementos para renderizar.

- `útil/`: carpeta con funciones independientes o autocontenidas que son reutilizadas para el cálculo o manejo de data en varios lugares distintos del código.
- `tipos/`: contiene definiciones de objetos, los cuales cumplen un rol de facilitar la programación y son parte integral del desarrollo con el lenguaje de programación *TypeScript*. En esta carpeta no hay funciones o componentes, sólo se exportan definiciones de tipos.
- `data/`: guarda información sobre el mapa, ya sean las áreas o los caminos que existen por cada zona (en objetos JSON).
 - `global.ts`: archivo con las áreas, límites y caminos de la zona global.
 - `p/`: carpeta con las áreas, límites y caminos de cada piso del edificio P, separado por archivo.
- `componentes/`: carpeta que mantiene componentes de *React Native* independientes o autocontenidos, que son reutilizables en diferentes vistas.
 - `botones/`: componentes que contienen y actúan como botones.
 - `pisos.tsx`: tiene botones para cambiar de piso, cuando uno se encuentra en la vista de algún edificio.
 - `configuraciones.tsx`: archivo que muestra un botón con un ícono de engranaje que cambia la vista actual por la de las preferencias en “`opciones.tsx`”.
 - `preferencia.tsx`: botón interactivo de múltiple elección que muestra los 5 niveles que puede tener una preferencia.
 - `visual/`: carpeta de elementos visuales, particularmente que se generan sobre el mapa.
 - `area.tsx`: muestra una figura (interactiva) de un área, dependiendo de los datos y propiedades definidos en “`data`”.
 - `edificio.tsx`: genera un edificio dentro de la zona global, que usa la información de “`data`”.
 - `ícono.tsx`: archivo encargado de dibujar íconos en las áreas o cualquier otro componente.
 - `perímetro.tsx`: marca el límite visual del mapa en las vistas.
 - `camino/`: contiene lógica específica para el algoritmo que procesa el material de “`data`” y, también, con los componentes para la interacción y visualización del camino.

Conociendo la estructura organizacional del código, se puede continuar con el diseño del proceso principal del proyecto: seleccionar dos puntos y que se muestre un camino entre ellos. Se dispone un diagrama de procesos en la Figura 9, que es simplificado y consiste en

la selección de estos marcadores en la aplicación. No se incluyó la capacidad de establecer preferencias de usuario para manipular la creación del camino entre los marcadores, porque es una operación paralela y sólo se encarga de asignar un valor y no de calcular pesos o manipular data.

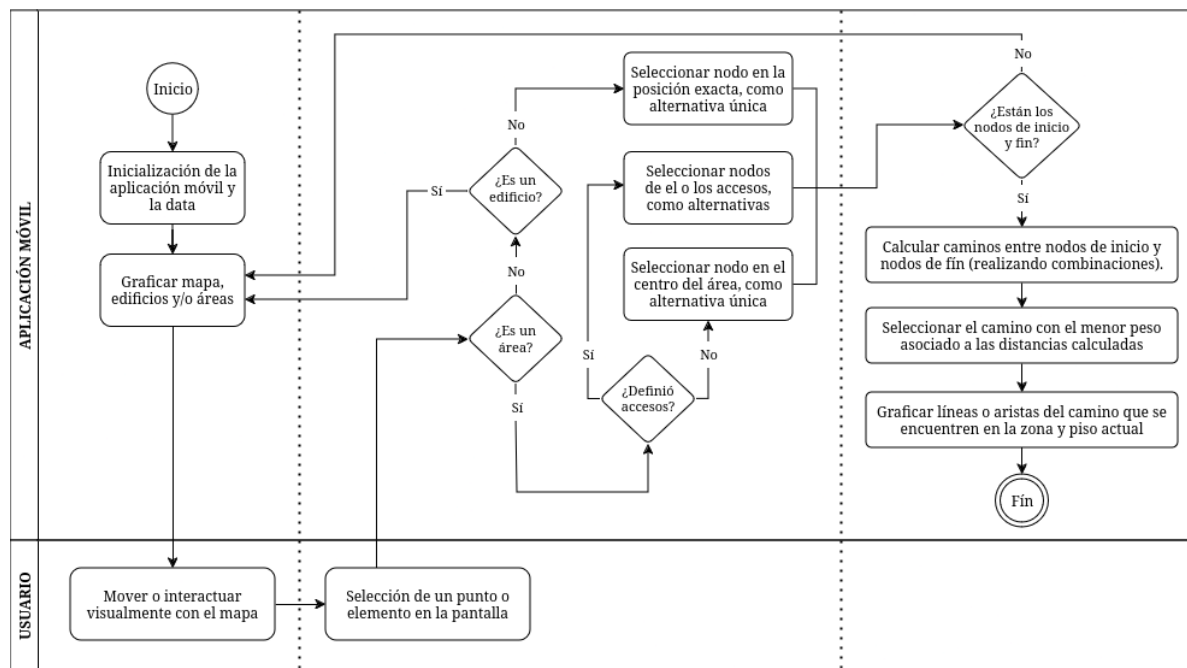


Figura 9: Diagrama de procesos que representa la interacción de un usuario con la aplicación para marcar dos puntos (uno de inicio y otro de fin) y luego, visualizar un camino entre ambos. Fuente: Elaboración propia.

En la figura se puede recalcar que al seleccionar alguno de los marcadores, ambos pueden tener más de un nodo alternativo a la vez si es que se trata de un área; sin embargo, esta información es interna y el usuario lo desconoce (sólo se informa que el área es uno de los puntos).

Esto puede ocurrir cuando se selecciona un área, porque esta tendría físicamente más de una entrada, puerta o acceso, por lo que es necesario calcular un camino para todas las opciones disponibles y también porque en aquellas áreas abiertas o de virtualmente infinitas formas de entrar, se asignan manualmente accesos que tengan sentido con las rutas disponibles, tal que algún camino que se forme hacia esta, deberá ser lo más corta posible (un ejemplo de esto se puede ver en la Figura 10).

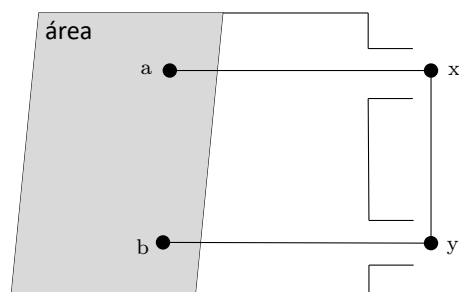


Figura 10: Ejemplo de un caso donde se necesitan dos accesos (a y b) al área en gris, ya que hay dos rutas que empiezan en los nodos x e y . Lo más sensato es tener dos alternativas rectas dado el espacio y estructura disponible. Fuente: Elaboración propia.

3.2.2. INFRAESTRUCTURA

Para comenzar a obtener los datos geográficos y prepararlos para la aplicación, se tiene la necesidad de obtener la información estructural de CC. Gracias a la Dirección de Infraestructura de la universidad, particularmente de Oscar Jalil (Director de Infraestructura) y Jancob Olivares (Ingeniero de Gestión de Infraestructura), quienes colaboraron de manera desinteresada para proveer con estos planos requeridos sin inconvenientes.

Los planos de CC incluyen todas las áreas, edificios, vegetación y muchas otras cosas que pueden ser de interés. Originalmente se consideraba la opción de trabajar directamente sobre estos archivos de diseño, ya que existía la opción de encontrar cómo manipular los datos para que la aplicación pudiera graficarlos; sin embargo, la dificultad del proyecto aumentaría considerablemente, dado que estos son extremadamente complejos y con elementos que se necesitaría descartar o cambiar.

Los planos incluyen una vista global, equivalente a la que se trabaja en este proyecto, en función de que incluye los edificios de CC como una zona que indica el nombre de cada edificio correspondiente sin mostrar los interiores (ver Figura 11). Los otros archivos dispuestos, contienen todos los pisos de cada edificio del campus, por lo que se hace un enfoque en los que corresponden al edificio “P” para este trabajo.

El material de diseño estructural de cada piso, se ocupa como base para determinar la posición relativa de cada elemento. Por ejemplo, se puede observar las salas de clase, los baños, paredes, puertas, escaleras, etc.

En modo preliminar, se busca convertir la información de los planos a objetos JSON como los descritos en secciones pasadas (*Area*, *Building*, etc.). Para esto, se encuentra una aplicación web en línea llamada “*Desmos Graphing Calculator*” [Desmos Studio, PBC, 2011] (diremos simplemente “*Desmos*”), que se encarga de graficar componentes matemáticos (puntos, curvas, funciones y otras cosas). La intención es aprovechar que esta herramienta es flexible y avanzada, por lo que se pueden añadir imágenes y construir elementos interactivos que faciliten el proceso de añadir áreas (ver Figuras 21 y 22 en el Apéndice C). Es importante destacar

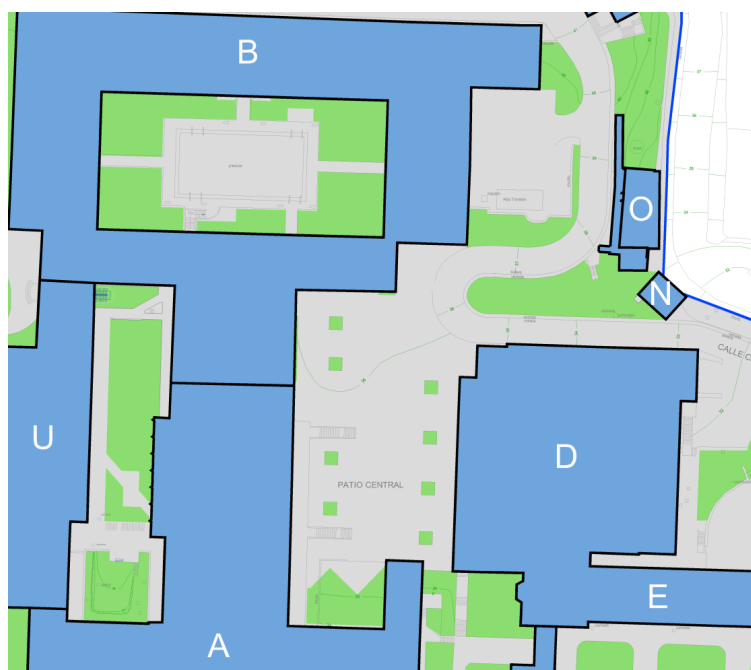


Figura 11: Captura de pantalla de la sección de un archivo de diseño que muestra parte de CC visto desde arriba. Se detallan los edificios correspondientes demarcados como zonas ("A", "B", "D", etc.). Fuente: Archivo de diseño facilitado por la Dirección de Infraestructura de la Universidad.

que cada nodo de estas áreas, es un valor arbitrario en el sistema de coordenadas cartesiano. De momento esto no se puede traducir a información geográfica.

Los colores y formatos de los nombres de cada área en esta aplicación web, sirven el propósito de transformar cada línea en un objeto *Area*. Esto se puede aprovechar, gracias a la flexibilidad y capacidad de manipular la información con funciones propiamente dispuestas por *Desmos*, las cuales son parte de su interfaz de programación de aplicaciones o "*Application Programming Interface*" (API)¹, en inglés.

Se pueden ejecutar instrucciones en el navegador que se encarguen de tomar todo y convertirlo como se desee. Luego, el resultado de la conversión serían los objetos que se guardan en el código dentro de la carpeta "data".

Resta evaluar los caminos que se crearon dentro de cada piso. En la Figura 23 del Apéndice C, se observan líneas segmentadas, otras punteadas y con diversos colores que cumplen la misma función que con las áreas (especificar atributos a la hora de convertir la data en objetos JSON). Estos valores también se guardan en el mismo lugar.

¹Grupo de reglas y herramientas que facilita la integración, comunicación, interoperabilidad y expansión de funcionalidades entre sistemas informáticos. Actúa como un intermediario, permitiendo acceso a funciones o datos de otros sistemas, de forma estandarizada y eficiente, sin necesidad de conocer el código interno.

3.2.3. PROYECCIÓN GEOGRÁFICA

Para realizar la conversión del plano cartesiano en 2D a uno geográfico, se observa que se necesita hacer un proceso inverso a la proyección de Mercator¹.

La siguiente expresión representa la función inversa mencionada, tal que una coordenada (x, y) se transforme a longitud y latitud con (λ, ϕ) , respectivamente. Sin embargo, durante su uso, se añadió un factor de escalamiento y de traslación, ya que la posición y tamaño usado del plano en *Desmos* fue arbitrario y se necesita encontrar dónde posicionarlo en el mapa.

$$\text{mercator}^{-1}(x, y) = \left(x, \left(\arctan(e^y) - \frac{\pi}{4} \right) \cdot 2 \right) \quad (1)$$

¹Proyección cilíndrica donde las líneas de longitud se disponen verticales y equidistantes, todas perpendiculares a las líneas de latitud. Esto causa que las distancias se expandan cada vez más, según cuan lejos estén del ecuador.

CAPÍTULO 4

VALIDACIÓN DE LA SOLUCIÓN

Para validar la solución, se completa un mapa del edificio P de CC. La decisión de realizarlo para este edificio, es porque tiene más de un piso y estos contienen variadas áreas de interés como servicios, salas, laboratorios, escaleras y ascensores, permitiendo validar las opciones incorporadas en la aplicación.

Se logra completar los objetivos propuestos. Primero, se definieron cuales son las preferencias de usuario disponibles y esto se puede observar en la Figura 12, que muestra la vista de configuraciones y todas comienzan con el valor por defecto que corresponde a estar desactivada.



Figura 12: Captura de pantalla de la aplicación, mostrando una vista de usuario con las preferencias descritas en el Apartado 3.1.6. Fuente: Elaboración propia.

Luego, se definieron cuales son los atributos o propiedades que pueden existir para cada segmento del camino, los cuales fueron correctamente aplicados cuando se crearon y procesaron los datos del camino en el Apartado 3.2.2.

La creación y la interactividad con el mapa, dentro de la aplicación, se observa en las capturas de pantalla de la Figura 13. Este utiliza el formato correcto, muestra los elementos y permite interactuar con él. Se pueden seleccionar puntos de inicio y fin (marcadores), tal que se genere un camino eficiente entre ellos.

En la figura Figura 14 se ve el piso 1 del edificio P, seguidamente de haber seleccionado el área en verde que corresponde al patio. A continuación, se selecciona la sala P012 del piso -1, generando un camino marcado en azul claro que también se puede observar desde el piso anterior (el inicio del trayecto).

Después de cambiar algunas preferencias del trayecto, se puede notar su evolución como en las Figuras 15 y 16. En la primera, se establece que no se deben bajar escaleras, por lo que se utiliza el ascensor. En la segunda figura, se incluye la preferencia de no bajar ascensores y ahora la ruta pasa por la entrada del edificio en el piso 1 y, luego de cruzar la calle en el exterior, se reingresa al campus nuevamente a través del túnel que se observa en la última imagen del piso -1.

En el Apéndice D se incluyen grabaciones de las interacciones descritas en las figuras anteriores.

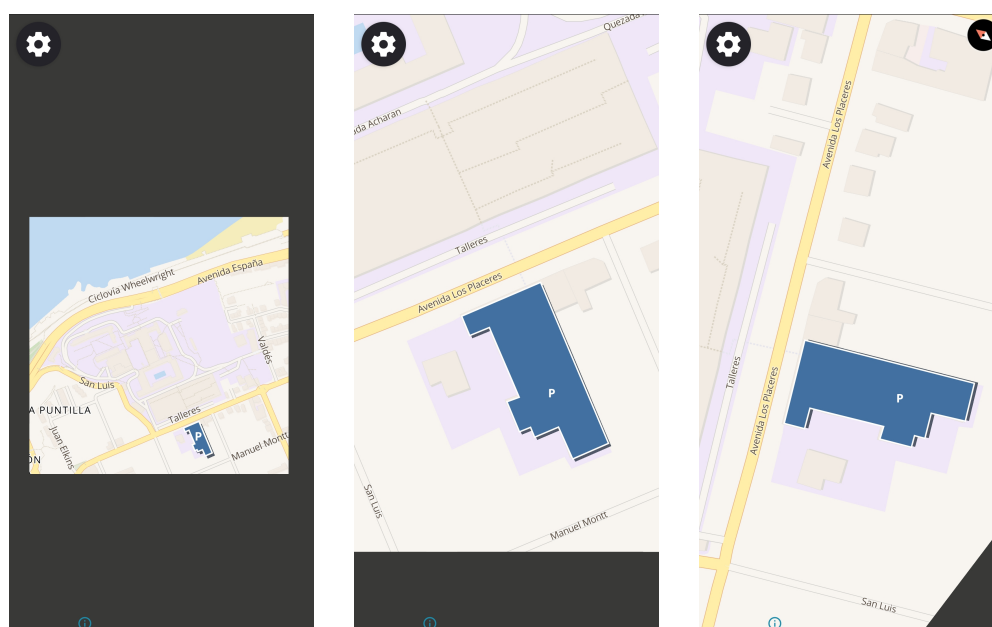


Figura 13: Capturas de pantalla (de izquierda a derecha) mostrando interacción táctil para realizar escalamiento y rotación del mapa. Fuente: Elaboración propia.

Finalmente, se puede añadir que para cambiar un detalle respecto a las áreas o al camino, sólo se necesita editar los objetos JSON guardados, ya que serán procesados automáticamente por la aplicación y no se necesita trabajar sobre la funcionalidad dentro del código.

4.1. ESTADÍSTICAS

La aplicación se puede construir para los sistemas operativos móviles más conocidos: “Android” y “iOS”. Para efectos de desarrollo, se probó en *Android*, que corresponde al sistema

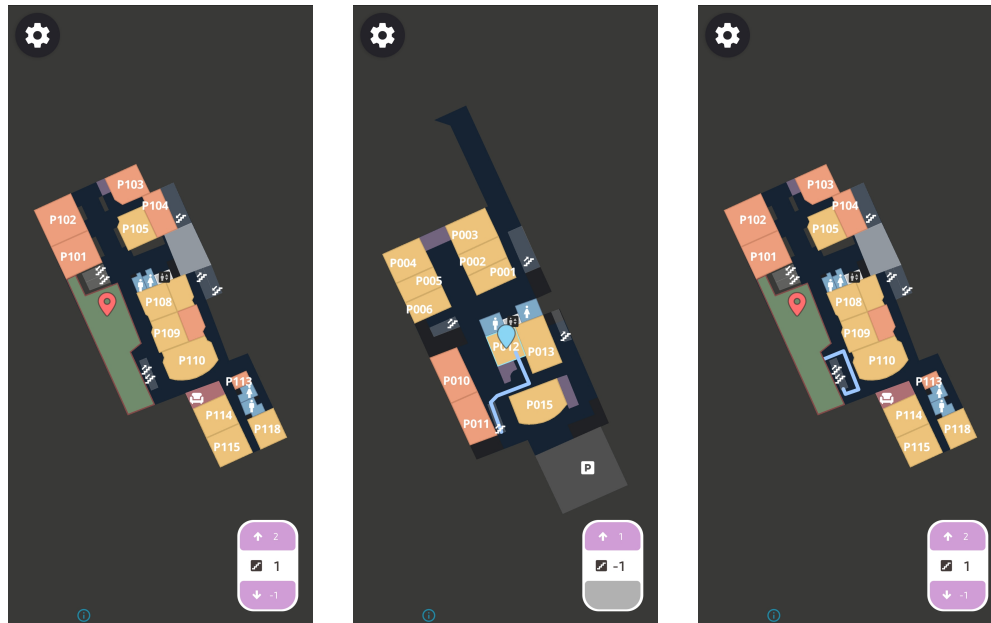


Figura 14: Capturas de pantalla (de izquierda a derecha) dentro del edificio P. Se selecciona un área en la primera imagen, luego otra área se observa el camino generado en las dos últimas. Fuente: Elaboración propia.

que se tiene disponible.

La aplicación pesa aproximadamente 400MB en modo desarrollo, que es un modo que facilita realizar cambios para observarlos instantáneamente mientras se desarrolla, y pesa 325MB cuando se construye en su versión final.

La selección de dos puntos hace que se tarde aproximadamente 500ms en generar un trayecto. En general, hay una variación dependiendo de la ubicación de estos, ya que distancias grandes o sectores con muchas alternativas afectan el procesamiento. El uso de preferencias no afecta a lo anterior por sí solo, ya que no añade un cálculo nuevo, sino que siempre se evalúa si las rutas tienen algún peso nuevo.

Se estima que ampliar el mapa a todo el campus, podría afectar considerablemente el tiempo de procesamiento de los caminos. Esto incluye también el uso de memoria en tiempo de ejecución al mantener mucha información que no se está utilizando en todo momento. Claramente empeoraría si se incluyen, además, otros emplazamientos de la universidad. En su estado actual, la aplicación puede extenderse hasta cierto límite antes de ser activamente difícil de utilizar, probablemente siendo la mitad o quizás el campus completo. Desde ahí, se necesitará reformular el método para guardar y acceder a la información.

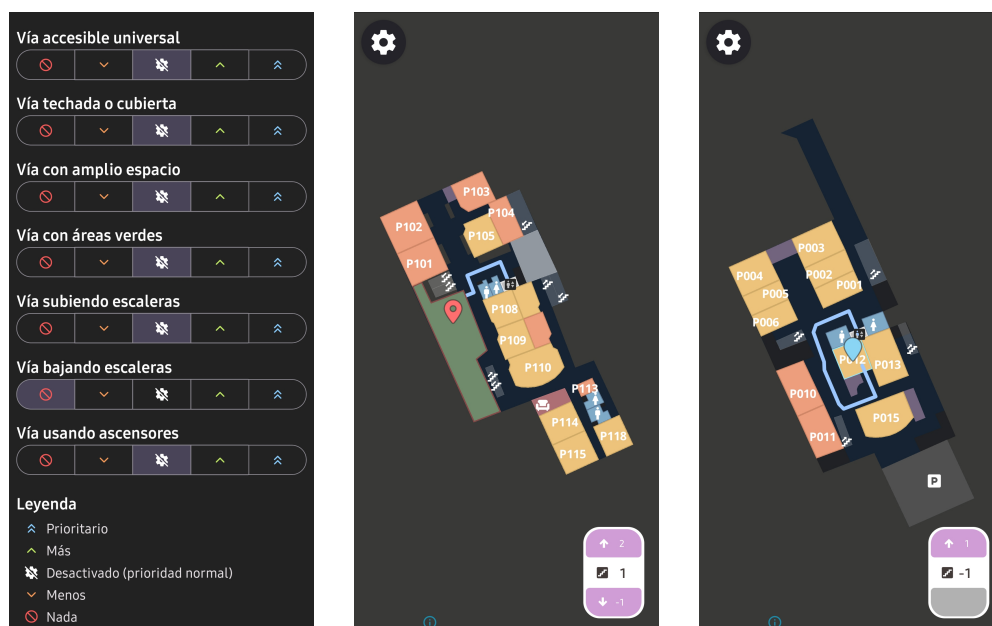


Figura 15: Capturas de pantalla (de izquierda a derecha) dentro del edificio P, luego de seleccionar la opción para evitar bajar escaleras. Fuente: Elaboración propia.

4.2. PROYECCIONES

Este proyecto tiene infinito potencial, ya que sólo se trabajó en la superficie de lo que puede llegar a ser esta aplicación. Sería ideal que se incluyeran todas las áreas, edificios y caminos posibles dentro CC. Luego, se pueden incluir el resto de los emplazamientos.

Los datos del mapa se encuentran cargados y procesados al momento de iniciar el sistema, pero no requiere mucho trabajo moverlos y que funcione la aplicación con un pre-procesamiento que manipule los datos y los deje listos para ser utilizados, disminuyendo así también el espacio que utiliza la aplicación. Realizar esto no afectaría la facilidad o flexibilidad de edición, puesto que se puede automatizar este procesamiento previo y el efecto es el mismo.

Una herramienta secundaria que se encargue de crear los datos necesarios para el mapa sería bastante útil. Para el proyecto se trabajó con *Desmos*, pero este generaba trabas que afectaban al desarrollo respecto a tiempo y comodidad (obviamente no fue creada con el objetivo de ser útil en este proyecto) y aun así, fue eficiente su uso comparado a otras alternativas.

Respecto a accesibilidad y experiencia de usuario, hay bastante material para complementar. No hay un elemento que indique en qué parte de la ruta se está actualmente o dónde se encuentra el usuario en un principio. Existe la opción de trabajar sobre la infraestructura para proveer de códigos QR (por ejemplo) u otros elementos informáticos que faciliten la ubicación del usuario dentro de la aplicación. Se puede notar que tampoco hay rutas alternativas

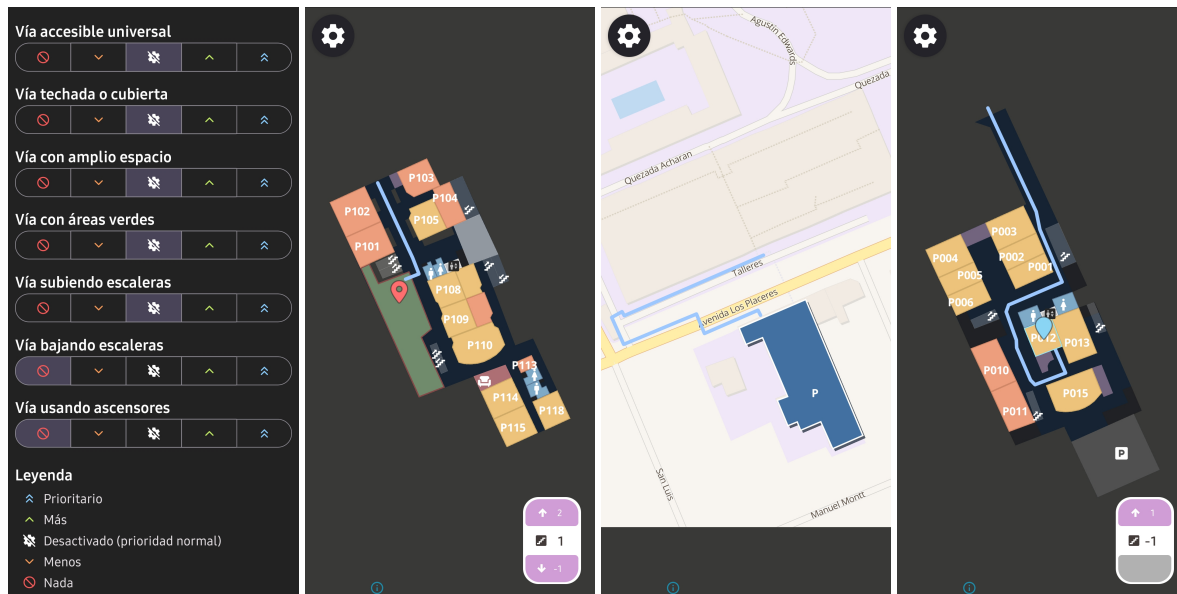


Figura 16: Capturas de pantalla (de izquierda a derecha) del camino desde el patio del edificio P hacia la sala P012, sin usar escaleras ni ascensores. Fuente: Elaboración propia.

que se generen para el usuario. Una adición invaluable sería completar los mecanismos y la interfaz de usuario que la pondrían a un estándar apropiado para una aplicación de navegación de hoy en día (por ejemplo, incluir una barra de búsqueda, permitir que la localización actual del usuario funcione como punto de inicio, cambios estéticos con modo oscuro o claro, etc.).

Similar a lo anterior, hay extensas maneras de mejorar la calidad de la implementación. Hay procesos realizan más trabajo de lo necesario, pudiendo mejorar el tiempo de generación de caminos o el uso de memoria.

Es importante también, realizar una evaluación correcta del impacto y de la utilidad de las preferencias y los distintos niveles tanto en estudiantes, funcionarios, visitas, etc. Esto significaría comprobar que realmente se está ayudando y cumpliendo con la necesidad de encontrar caminos que se adecúen a las preferencias y de navegar eficientemente dentro de la universidad.

Dentro del mismo contexto, un sistema de retroalimentación o de opiniones, ya sea integrado o no a la aplicación, cumpliría un rol importante para mantener este proyecto como un servicio. Un ejemplo de esto, se vería cuando eventualmente hayan cambios estructurales, salas deshabilitadas, sectores cerrados u otras cosas que podrían afectar a la representación estática que se tiene del campus actualmente. Por otra parte, el funcionamiento de la aplicación podría presentar errores que se hayan escapado durante el desarrollo y un ciclo de actualizaciones periódicas beneficiaría el ecosistema del servicio.

CAPÍTULO 5

CONCLUSIONES

El desarrollo de la aplicación móvil enfocada en accesibilidad, flexibilidad y eficiencia dentro del campus universitario ha demostrado ser una herramienta innovadora y de gran utilidad. Permite a los usuarios seleccionar preferencias específicas y variadas para planificar sus rutas, considerando aspectos como evitar escaleras o transitar por áreas verdes. Este enfoque personalizado no solo mejora la usabilidad, experiencia y movilización de dentro del campus, sino que también puede adaptarse a las necesidades cambiantes de los usuarios y a futuras expansiones del proyecto. Lamentablemente, dentro de las limitaciones en el alcance de este proyecto, sólo se pudo abordar un sector pequeño del campus.

Se logra resolver un problema de tridimensionalidad en navegación de interiores, particularmente por la dificultad de mantener y trabajar con este tipo de datos. La solución de simplificar la coordenada que represente verticalidad y luego incluir otra coordenada que indique cambios de zona ha sido correcta. Gracias a esto, se puede representar visualmente y procesar cada elemento en forma lógica e intuitiva, ya que se formuló teniendo en cuenta estas limitaciones de diseño. Aun así, hay una necesidad de estandarizar el formato de la información, que ahora existe en cuatro dimensiones. El método para enlazar elementos que están en diferentes zonas o pisos, se vería beneficiado por un componente que formalice y procure que se mantenga un orden y estructura apropiado.

Además, la flexibilidad del sistema, que permite modificar y añadir contenido sin necesidad de reestructurar el código, asegura que este servicio se mantenga relevante y útil a largo plazo. Esto es esencial para que el proyecto no sólo cumpla con los objetivos definidos, sino que también pueda evolucionar con el tiempo, integrando nuevas funcionalidades y adaptándose a cambios estructurales en el campus. Sin embargo, cabe mencionar que el formato elegido para el procesamiento de los datos del mapa universitario, puede resultar ineficiente a una mayor escala (por ejemplo, añadiendo más lugares u otras sedes). Esto significa que se necesita replantear el formato, tal que se mantenga la legibilidad y flexibilidad de edición, pero que sea resiliente al aumento sustancial de información. El preprocesamiento de los datos y creación de herramientas intermedias que faciliten el desarrollo se mantiene alineado con los objetivos propuestos, por lo que es importante tenerlo en cuenta.

Finalmente, este proyecto tiene el potencial de convertirse en una pieza clave para mejorar la accesibilidad y traslado en el entorno universitario, sirviendo como modelo para ser replicado en otras instituciones.

ANEXOS

A. TURISMO LENTO

En [Moira *et al.*, 2017, Oh *et al.*, 2014], se observa que práctica del turismo en general no tiene un concepto muy acotado, particularmente con la idea de “turismo en masa”, el cual ha sido extremadamente prevalente en los últimos tiempos y no suele ser definido claramente. Sin embargo, se remarcan varios conceptos que consideran como “propiedades” del turismo en masa, como por ejemplo: el consumo y daño del medio ambiente, es dirigido y organizado para el público general por parte de empresas u operadores particulares, falta de planificación e infraestructura a gran escala, excesivo poder y costo económico y territorial, etc. En general, la perspectiva social, cultural, política y, en parte económica, es negativa.

Naturalmente, el conflicto que tiene la nueva madurez cultural, que es consciente del impacto en todos los ámbitos que tiene el turismo en masa, favorece el apareamiento de nuevas alternativas que suplan la necesidad del turismo en las personas. Una de estas alternativas, es el “turismo lento”.

El turismo lento no nace por cuenta propia, sino que tiene origen en otros dos movimientos: “Movimiento Lento” (valoriza la calidad, antes que la cantidad, para disfrutar de experiencias, actividades y relaciones teniendo en cuenta la sostenibilidad) y “Ciudad Lenta” (prioriza el desarrollo sustentable, preservación y promoción cultural y de comida local, mejoras de diseño urbanos y organización política local, etc.).

Se definen algunas características más específicas y no exhaustivas en [Caffyn, 2012], que indican mas bien ejemplos prácticos como: minimizar las distancias de viajes (sobre autos, aviones, etc.), maximizar el tiempo dispuesto para el viaje, explorar y descubrir el lugar de destino en profundidad, aprender talentos o actividades nuevas, recibir experiencias auténticas y de calidad, y más.

B. EJEMPLOS DE OBJETOS JSON

```

1  {
2    // identificador único
3    "id": "kiosco-patio-central",
4    // puntos [x, y] del polígono
5    "polygon": [
6      [ -71.595567, -33.035355 ],
7      [ -71.595542, -33.035387 ],
8      ...,
9    ],
10   // punto de acceso al kiosco
11   "entrances": [
12     [ -71.595570, -33.035382 ]
13   ],
14   ...
15 }

```

Figura 17: Ejemplo en JSON de un objeto tipo “Area”. Fuente: Elaboración propia.

```

1  {
2    // límites del piso
3    "boundary": [
4      [ -71.594520, -33.036563 ],
5      [ -71.594132, -33.037144 ],
6      [ -71.593973, -33.037086 ],
7      ...
8    ],
9    // areas dentro piso actual
10   "areas": [
11     { "id": "p001", ... },
12     { "id": "p002", ... },
13     ...
14   ],
15 }

```

Figura 18: Ejemplo en JSON de un objeto tipo “*Floor*” que incluye objetos tipo *Area*. Fuente: Elaboración propia.

```

1  {
2    // identificador único
3    "id": "p",
4    // límite y áreas de cada piso
5    "-1": {
6      "boundary": [ ... ],
7      "areas": [ ... ],
8    },
9    "1": {
10     "boundary": [ ... ],
11     "areas": [ ... ],
12   },
13   ...
14 }

```

Figura 19: Ejemplo en JSON de un objeto tipo “*Building*” que incluye objetos tipo *Floor*. Fuente: Elaboración propia.

```

1  {
2    "zone": "p",
3    "floor": "1",
4    "coordinates": [
5      [-71.594359, -33.036923],
6      [-71.594343, -33.036956],
7    ],
8    "accessible": false,
9    "stairs": true,
10   "conn": {
11     "zone": "p",
12     "floor": "2",
13     "coordinate": [ -71.594326, -33.036990 ]
14   }
15 }

```

Figura 20: Ejemplo en JSON de las propiedades de una escalera y los nodos que contiene en la llave “coordinates”. La última coordenada en esta (la línea 6), es la que se conecta con el nodo que se representa en “conn” y que existe en otro objeto. Fuente: Elaboración propia.

C. CAPTURAS DE PANTALLA DE DESMOS

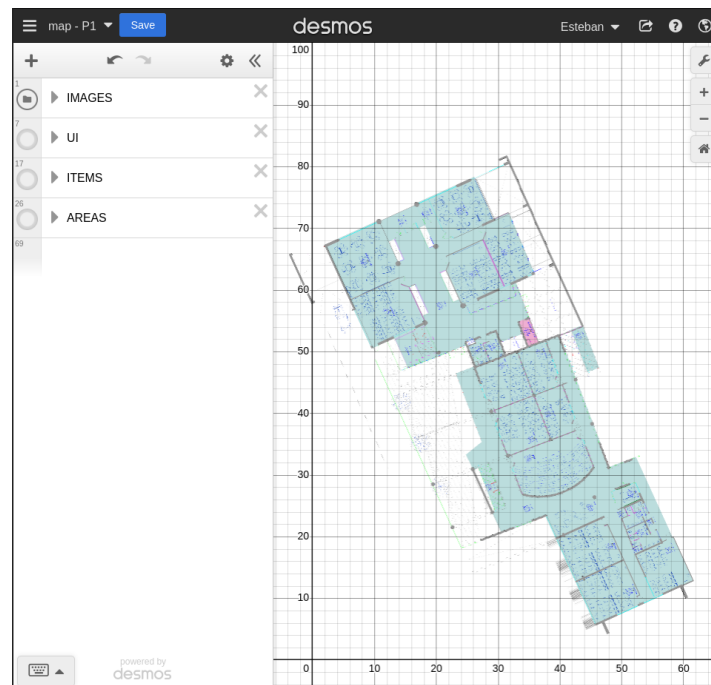


Figura 21: Captura de pantalla de *Desmos* con una imagen de un plano provisto por la Dirección de Infraestructura (primer piso del edificio P de CC). Fuente: Elaboración propia.

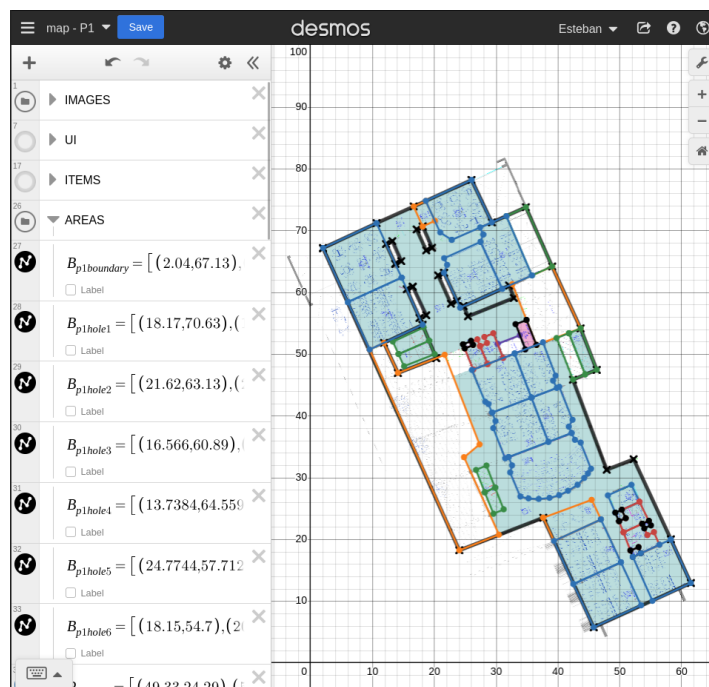


Figura 22: Captura de pantalla de *Desmos* con áreas poligonales (conjunto de puntos) creadas sobre la imagen del plano provisto por la Dirección de Infraestructura (primer piso del edificio P de CC). Fuente: Elaboración propia.

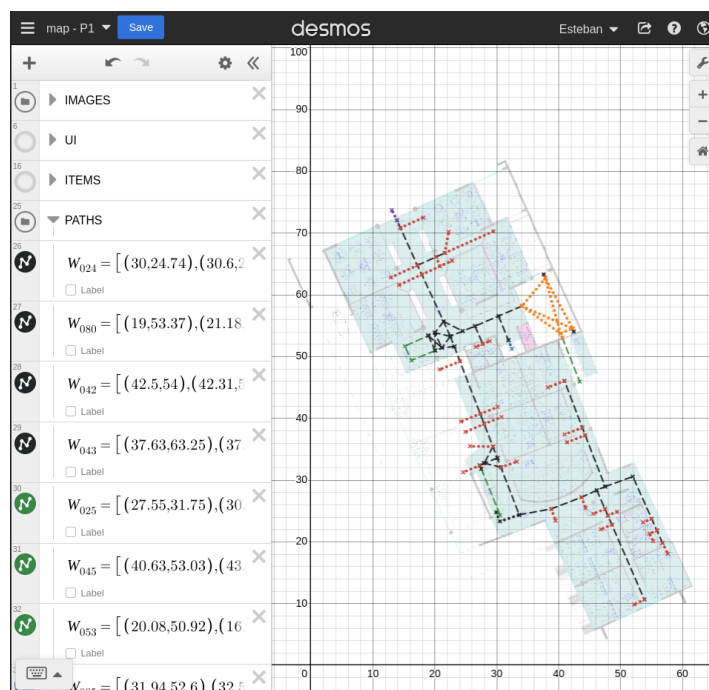


Figura 23: Captura de pantalla de *Desmos* con caminos (conexiones de puntos) creados sobre la imagen del plano provisto por la Dirección de Infraestructura (primer piso del edificio P de CC). Fuente: Elaboración propia.

D. VIDEOS DE EJEMPLO

- Video que muestra la interacción con la aplicación desarrollada. Se seleccionan dos puntos y el sistema genera un camino apropiado, que además se vea afectado por las preferencias de usuario (bajar escaleras y usar ascensor).

Enlace: <https://youtu.be/fpsuxc9QVWo>

- Video que incluye la interacción con la vista "global" del mapa y el caso en que las preferencias de usuario son muy restrictivas (y no se encuentra un camino).

Enlace: https://youtu.be/-15_NUnQpf0

REFERENCIAS BIBLIOGRÁFICAS

- [Caffyn, 2012] Caffyn, A. (2012). Advocating and implementing slow tourism. *Tourism Recreation Research*, 37(1):77–80.
- [Collegis Education, 2019] Collegis Education (2019). Rasmussen university. <https://play.google.com/store/apps/details?id=com.readyeducation.rasmussen>, Último acceso 2024-04-17.
- [Curry, 2024] Curry, D. (2024). Most popular apps (2024). <https://www.businessofapps.com/data/most-popular-apps/>, Último acceso 2024-04-17.
- [Dechter y Pearl, 1985] Dechter, R. y Pearl, J. (1985). Generalized best-first search strategies and the optimality of a*. *Journal of the ACM*, 32(3):505–536.
- [Dela Cruz et al., 2016] Dela Cruz, J. C., Magwili, G. V., Mundo, J. P., Gregorio, G. P., Lamo-ca, M. L., y Villasenor, J. A. (2016). Items-mapping and route optimization in a grocery store using dijkstra's, bellman-ford and floyd-warshall algorithms. *2016 IEEE Region 10 Conference (TENCON)*.
- [Desmos Studio, PBC, 2011] Desmos Studio, PBC (2011). Desmos graphic calculator. <https://www.desmos.com/calculator>, Último acceso 2024-08-24.
- [Dijkstra, 1959] Dijkstra, E. W. (1959). A note on two problems in connexion with graphs. *Numerische Mathematik*, 1(1):269–271.
- [Dinitz e Itzhak, 2017] Dinitz, Y. e Itzhak, R. (2017). Hybrid bellman–ford–dijkstra algorithm. *Journal of Discrete Algorithms*, 42:35–44.
- [Farahsari et al., 2022] Farahsari, P. S., Farahzadi, A., Rezazadeh, J., y Bagheri, A. (2022). A survey on indoor positioning systems for iot-based applications. *IEEE Internet of Things Journal*, 9(10):7680–7699.
- [Hart et al., 1968] Hart, P. E., Nilsson, N. J., y Raphael, B. (1968). A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on Systems Science and Cybernetics*, 4(2):100–107.
- [Huang y Gartner, 2009] Huang, H. y Gartner, G. (2009). A survey of mobile indoor navigation systems. *Lecture Notes in Geoinformation and Cartography*, p. 305–319.
- [Kunhoth et al., 2020] Kunhoth, J., Karkar, A., Al-Maadeed, S., y Al-Ali, A. (2020). Indoor positioning and wayfinding systems: A survey. *Human-centric Computing and Information Sciences*, 10(18).
- [MapLibre, 2022] MapLibre (2022). Maplibre gl sdk for react native. <https://github.com/maplibre/maplibre-react-native>, Último acceso 2024-08-22.

- [Moira *et al.*, 2017] Moira, P., Mylonopoulos, D., y Kondoudaki, E. (2017). The application of slow movement to tourism: Is slow tourism a new paradigm? *Journal of Tourism and Leisure Studies*, 2(2).
- [Oh *et al.*, 2014] Oh, H., Assaf, A. G., y Baloglu, S. (2014). Motivations and goals of slow tourism. *Journal of Travel Research*, 55(2):205–219.
- [Pascacio *et al.*, 2021] Pascacio, P., Casteleyn, S., Torres-Sospedra, J., Lohan, E. S., y Nurmi, J. (2021). Collaborative indoor positioning systems: A systematic review. *Sensors*, 21(3):1002.
- [Per Liedman , 2016] Per Liedman (2016). Geojson path finder. <https://github.com/perliedman/geojson-path-finder>, Último acceso 2024-08-22.
- [Princeton University, OIT AS, 2023] Princeton University, OIT AS (2023). Campus map. <https://play.google.com/store/apps/details?id=edu.princeton.wayfinding>, Último acceso 2024-04-17.
- [Tarjan, 1972] Tarjan, R. (1972). Depth-first search and linear graph algorithms. *SIAM Journal on Computing*, 1(2):146–160.
- [University of Oregon, 2012] University of Oregon (2012). Uoregon. <https://play.google.com/store/apps/details?id=edu.uoregon>, Último acceso 2024-04-17.
- [Zentrale Anlaufstelle Barrierefrei, 2019] Zentrale Anlaufstelle Barrierefrei (2019). Unimaps. https://play.google.com/store/apps/details?id=de.uni_maps, Último acceso 2024-01-24.
- [Zhou y Hansen, 2006] Zhou, R. y Hansen, E. A. (2006). Breadth-first heuristic search. *Artificial Intelligence*, 170(4–5):385–408.