Departamento de Investigación Operativa
Instituto de Computación - Facultad de Ingeniería
Universidad de la República (UdelaR)
Montevideo - Uruguay

Proyecto de Grado
Ingeniería en Computación

INTERFAZ DE ASISTENCIA A LA EXPERIMENTACIÓN CON ALGORITMIA PARA EL ORDENAMIENTO DE VEHÍCULOS Y TRIPULACIÓN EN EL TRANSPORTE URBANO COLECTIVO

INFORME FINAL

Año 2010

Estudiante Mario Di Genio

Tutor responsable Maria Urquhart
Usuario Responsable Antonio Mauttone
Co-Usuarios Responsables Giovanna Garula, Sebastián Alaggia y Rafael Alvarez

Palabras clave: ordenamiento de vehículos y tripulación en el transporte urbano colectivo, interacción persona-computadora, diseño de interfaces

Resumen

El Departamento de Investigación Operativa del Instituto de Computación (InCo) de la Facultad de Ingeniería desarrolló algoritmos para resolver el problema de **planificación y ordenamiento de vehículos y tripulación** para una compañía de transporte colectivo urbano y suburbano de pasajeros. Este problema consiste en asociar viajes a unidades de transporte de forma de minimizar los costos operativos formando servicios (secuencias de eventos o actividades), sujeto a ciertas restricciones (como por ejemplo: leyes laborales que se deben cumplir por parte del personal). La mayoría de las variantes de estos problemas pertenecen a la clase NP-difícil, por lo tanto la resolución de instancias de tamaño realista requiere la investigación de algoritmos aproximados. En este contexto se han desarrollado algoritmos basados en programación lineal, heurísticas y algoritmos genéticos. Estos desarrollos implementan la capa lógica de una herramienta de software de apoyo a la resolución de estos problemas. Como resultado, se obtiene un conjunto de libros o diarios de viajes o soluciones (planilla o conjunto de planillas con las actividades, eventos u horarios que cumplen los servicios o el personal para una estación del año). En el marco de este proyecto de grado de la carrera de Ingeniería en Computación se planteó el desarrollo de las interfaces, tanto de software como de usuario, que permiten al usuario interactuar con la capa lógica desarrollada, de manera que éste pueda obtener los resultados, visualizarlos apropiadamente y realizar la planificación de los libros, la cual incluye procedimientos de ordenamiento de la flota de ómnibus y la asignación de conductores.

Para cumplir con estos objetivos se desarrolló un sistema llamado **FingLab** que consiste en una interfaz gráfica que permite al usuario interactuar con los algoritmos, configurar su ejecución, realizar las ejecuciones de manera ordenada y clasificada, procesar los datos de entrada y salida de cada ejecución, representarlos gráficamente y realizar modificaciones manuales sobre las soluciones de ser necesario. Durante el proceso de desarrollo se realizaron estudios sobre el transporte urbano colectivo de pasajeros (y su planificación y ordenamiento), estado del arte de las interfaces y herramientas similares en el mercado.

Existen dos tipos de usuarios: los técnicos (que también los llamaremos usuarios Administradores o Programadores) y los no técnicos (que también los llamaremos usuarios Comunes o Planificadores). Los usuarios técnicos son quienes experimentan con varios algoritmos y módulos y les interesa cambiar su configuración. Su objetivo es experimentar y analizar las soluciones desde el punto de vista de la efectividad de los algoritmos, para poder seguir trabajando en su implementación. Estos usuarios son los responsables de este proyecto por parte de la Facultad de Ingeniería. Los usuarios no técnicos son aquellos a quienes les interesa planificar y ordenar su flota de vehículos y tripulación. No les interesa cómo estén implementados los algoritmos, solo les interesa ejecutarlos para obtener sus resultados y trabajar sobre las soluciones obtenidas. Estos usuarios corresponden a los encargados de la planificación en la compañía de transporte. FingLab es un sistema cuyo diseño es centrado en el usuario, por lo que las características de los mismos determinaron la forma en la que fue implementado.

El objetivo de FingLab es proveer de un sistema que además de cumplir con todos los requerimientos determinados por los usuarios, es flexible y permite incorporar módulos y adaptarse a los constantes cambios requeridos por los usuarios a futuro (en esta versión, los usuarios que desean experimentar con algoritmos). Dichos cambios se producen sobre todo en cuanto a los formatos de las soluciones producidas por los algoritmos, ya que el desarrollo de los algoritmos evoluciona constantemente y FingLab se puede adaptar para que pueda seguir trabajando con los nuevos formatos sin tener que realizar mayores modificaciones a su implementación. También posee funcionalidades que permiten exportar datos en diferentes formatos para el intercambio de datos con otras aplicaciones ya implementadas. De esta manera, FingLab permite cerrar todo el ciclo de trabajo requerido en la creación de los libros de servicios por la compañía de transporte. FingLab es también un Framework, ya que permite su extensión a diversas representaciones gráficas, lo que hace posible que se utilice como herramientas de trabajo para la experimentación con algoritmos que no tienen por qué ser de ordenamiento de vehículos y tripulación, reutilizando al máximo los componentes ya provistos por el sistema.

Índice

1 Introducción

En el presente informe se describe el proceso de desarrollo de una interfaz gráfica que permite manipular módulos que implementan los algoritmos para la resolución del problema de **planificación y ordenamiento de vehículos y tripulación** para una compañía de transporte colectivo urbano y suburbano de pasajeros. El sistema resultante llamado **FingLab** sirve de apoyo a la toma de decisiones en la planificación de las planillas de servicios de los ómnibus en el transporte suburbano, así como apoyo a la experimentación con los diversos algoritmos implementados y evaluación de los mismos.

En una compañía de transporte colectivo urbano y suburbano de pasajeros, se utilizan libros, o diarios de viajes (planilla o conjunto de planillas con las actividades, eventos u horarios que cumplen los servicios o el personal para una estación del año). La secuencia de viajes que realiza una unidad de transporte (ómnibus) se denomina servicio. Un servicio especifica la misión de un ómnibus durante todo un día útil (día de trabajo en que se realizan servicios y que puede ser un día hábil o no) y está compuesto por un conjunto de entre 0 a 3 turnos. Un turno es el trabajo planificado para ser realizado por un chofer en un día. Los diarios se clasifican en: si son para un día hábil (día de la semana que puede ser lunes, martes, miércoles, jueves o viernes siempre que no sea feriado) o día no hábil (día de la semana que puede ser sábado, domingo o cualquier día de la semana que sea feriado) dentro de una estación del año (primavera, verano, otoño o invierno). El tipo de día y la estación del año determinan el flujo de pasajeros y por ende la cantidad de viajes que realizan los servicios.

La planificación de los diarios es una tarea ardua que se desea realizar de la forma más eficiente posible, minimizando los costos operativos y, al mismo tiempo, cumpliendo con ciertas restricciones. No todas las actividades que se incluyen en una planilla generan ganancia, sino que existen para poder llevar a cabo los fines operacionales de la compañía. Existen varios tipos de actividades o eventos, cada una con sus características como se muestra en la Tabla 1-1.

Eventos	Definición
Viaje útil	Viaje que genera ganancia porque recoge pasajeros y cumple con los horarios y destinos estipulados.
Viaje directo sin pasajeros	También conocido como viaje expreso. No tienen fines económicos porque no recoge pasajeros, son viajes que le generan pérdidas a la empresa. Existen para poder trasladar un ómnibus del garage al lugar de salida del primer viaje útil, del lugar de destino del último viaje útil al garage y para llegar al lugar de salida de cualquier viaje útil que tenga asignado desde donde esté.
Disponible todo el día	El ómnibus no tiene ninguna actividad asignada en todo el día.
Disponible por un período	El ómnibus no tiene ninguna actividad asignada en un cierto período de tiempo.
Viaje del chofer como pasajero	Es un viaje donde un empleado o conductor viaja como pasajero para volver a su lugar de origen o para llegar al origen de un servicio que debe realizar.
Descansos	Tiempo en que un empleado o conductor no está realizando tareas de manejo. Igual cuenta como tiempo trabajado.
Cambio de turno	Momento en el servicio en el que se realiza el cambio de empleado o conductor.

Tabla 1-1: Tipos de eventos o actividades de un diario.

Por ejemplo, si un ómnibus debe realizar un viaje expreso para poder llegar al origen de su siguiente viaje útil, se genera un gasto que se desea minimizar, si es posible, con una mejor planificación. Entre las restricciones que se pueden tener, en nuestro caso, existen ciertas leyes laborales para los miembros de la tripulación que se deben obedecer, como por ejemplo: la cantidad de horas de los turnos, la cantidad de horas extra que se pueden realizar, la cantidad de descansos en el correr del día, distribuidos de una cierta manera estipulada. La planificación de los libros de servicios da origen a los problemas de ordenamiento que se pueden clasificar de la siguiente manera [21, 45]:

- **Vehicle Scheduling Problem (VSP)** o problema de ordenamiento de vehículos: Problema consistente en resolver la disposición de las unidades de transporte en el tiempo con objetivos

definidos (como la optimización en el uso de dichos recursos, la reducción de costos, etc.). Se refiere a la asignación de unidades de transporte a tareas en el tiempo [21, 45].

- **Crew Scheduling Problem (CSP)** o problema de ordenamiento de tripulación: Problema consistente en resolver la disposición de los recursos humanos en el tiempo, con objetivos definidos (como la optimización en el uso de dichos recursos, la reducción de costos, etc.). Se refiere a la asignación de recursos humanos a tareas. Está sujeto a restricciones que obedecen las reglas laborales correspondientes a los tiempos de descanso, su duración, su distribución durante la jornada, etc. [21, 45].

- **Integrated Scheduling Problem (ISP)** o problema de ordenamiento integrado: Es el problema de resolver del VSP y el CSP simultáneamente, en vez de por separado [21, 45].

A su vez estos tres tipos de problemas se pueden clasificar por otras características como puede ser la cantidad de depósitos o puntos desde donde salen y a donde vuelven las unidades de transporte [21, 45].

La mayoría de las variantes de estos problemas pertenecen a la clase NP-difícil, por lo tanto la resolución de instancias de tamaño realista requiere la investigación de algoritmos aproximados. En este contexto se han desarrollado algoritmos basados en programación lineal, heurísticas y algoritmos genéticos. La importancia de esta clase de problemas de decisión es que contiene muchos problemas de búsqueda y de optimización para los que se desea saber si existe una cierta solución o si existe una mejor solución que las conocidas [75].

El Departamento de Investigación Operativa del Instituto de Computación (InCo) de la Facultad de Ingeniería desarrolló algoritmos para resolver el problema de **planificación y ordenamiento de vehículos y tripulación** para una compañía de transporte colectivo urbano y suburbano de pasajeros. Este problema consiste en asociar viajes a unidades de transporte de forma de minimizar los costos operativos, sujeto a ciertas restricciones. Como resultado, se obtiene un conjunto de libros o diarios de viajes. Para ello, se han utilizado técnicas de programación lineal, heurísticas y algoritmos genéticos. Estos desarrollos implementan la capa lógica de una herramienta de software de apoyo a la resolución de estos problemas. En el marco de este proyecto de grado de la carrera de Ingeniería en Computación se planteó el desarrollo de las interfaces, tanto de software como de usuario, que permiten al usuario interactuar con la capa lógica desarrollada, de manera que éste pueda obtener los resultados, visualizarlos apropiadamente y realizar la planificación de los libros, la cual incluye procedimientos de ordenamiento de la flota de ómnibus y la asignación de conductores.

Para cumplir con estos objetivos se desarrolló un sistema llamado **FingLab** que consiste en una interfaz gráfica que permite al usuario interactuar con los algoritmos, configurar su ejecución, realizar las ejecuciones de manera ordenada y clasificada, procesar los datos de entrada y salida de cada ejecución, representarlos gráficamente y realizar modificaciones manuales sobre las soluciones de ser necesario. Durante el proceso de desarrollo se realizaron estudios sobre el transporte urbano colectivo de pasajeros (y su planificación y ordenamiento), estado del arte de las interfaces y herramientas similares en el mercado, que determinaron las decisiones de diseño de FingLab.

Existen dos tipos de usuarios: los técnicos (que también los llamaremos usuarios Administradores o Programadores) y los no técnicos (que también los llamaremos usuarios Comunes o Planificadores). Los usuarios técnicos son quienes experimentan con varios algoritmos y módulos, los desarrollan y les interesa cambiar su configuración. Su objetivo es experimentar y analizar las soluciones desde el punto de vista de la efectividad de los algoritmos (es decir, si las soluciones obtenidas son buenas o si se puede obtener mejores soluciones modificando la forma en que los algoritmos las construyen), para poder seguir trabajando en su desarrollo. Estos usuarios son los responsables de este proyecto por parte de la Facultad de Ingeniería. Los usuarios no técnicos son aquellos a quienes les interesa planificar y ordenar su flota de vehículos y tripulación. No les interesa cómo estén implementados los algoritmos, solo les interesa ejecutarlos para obtener sus resultados y trabajar sobre las soluciones obtenidas (es decir, realizar modificaciones manuales, ya que los algoritmos no devuelven soluciones óptimas). Estos usuarios corresponden a los encargados de la planificación en la compañía de transporte. FingLab es un sistema cuyo diseño es centrado en el usuario, por lo que las características de los mismos determinaron la forma en la que fue implementado.

FingLab posee las siguientes características:

- Provee las **interfaces** necesarias para el sistema de planificación de los libros de viajes de una compañía de transporte urbano y suburbano de pasajeros, el que incluye procedimientos de ordenamiento de la flota de ómnibus y la asignación de conductores, mantener y manipular los distintos libros planificados y las distintas soluciones obtenidas en el proceso de planificación.
- Sus interfaces forman parte de un **sistema de apoyo a la decisión** en la planificación de los libros de servicios de los ómnibus en el transporte suburbano. Por lo tanto, son amigables con el usuario y confiables, el usuario tiene siempre en claro qué es lo que hace y qué resultado debería esperar de las operaciones que está aplicando.
- Facilita la comunicación con el planificador y permite modificar las distintas soluciones obtenidas por los algoritmos de optimización e incorporar variantes propias.
- Contiene protocolos de comunicación entre:
    - El usuario y la algoritmia.
    - La algoritmia y FingLab.
    - Los distintos algoritmos desarrollados.
- Provee de interfaces internas de una herramienta de apoyo a la planificación, facilitando la interrelación de bases de datos estándares y las propias del sistema de ordenamiento de vehículos y tripulación.
- Provee de documentación de uso de la interfaz y protocolos.
- Es personalizable y permite su extensión para adaptarse a los cambios que el usuario pueda necesitar durante el ciclo de trabajo y también a futuro.
- Es lo suficientemente robusto como para desplegar los mensajes de error correctos frente a operaciones inesperadas del usuario y brindar ayuda en caso de que lo necesite.

El objetivo de FingLab es proveer de un sistema que además de cumplir con todos los requerimientos determinados por los usuarios, es flexible y permite incorporar módulos y adaptarse a los constantes cambios requeridos por los usuarios a futuro (en particular en esta versión vinculados a los usuarios que desean experimentar con algoritmos). Dichos cambios se producen sobre todo en cuanto a los formatos de las soluciones producidas por los algoritmos, ya que el desarrollo de los algoritmos evoluciona constantemente y FingLab se puede adaptar para que pueda seguir trabajando con los nuevos formatos sin tener que realizar mayores modificaciones a su implementación. También posee funcionalidades que permiten exportar datos en diferentes formatos para el intercambio de datos con otras aplicaciones ya implementadas.

De esta manera, FingLab permite cerrar todo el ciclo de trabajo requerido en la creación de los libros de servicios por la compañía de transporte. FingLab es también un Framework, ya que permite su extensión a diversas representaciones gráficas, lo que hace posible que se utilice como herramientas de trabajo para la experimentación con algoritmos que no tienen por qué ser de ordenamiento de vehículos y tripulación, reutilizando al máximo los componentes ya provistos por el sistema. FingLab se construyó en su totalidad y fue probado y aceptado por los usuarios de la Facultad de Ingeniería.

Este informe se focaliza principalmente en la descripción del producto desarrollado y su proceso de desarrollo está organizado de la siguiente manera:

- La sección 2 contiene el estudio del estado del arte en el área de interfaces.
- La sección 3 presenta un breve estudio del área de transporte y su planificación, haciendo énfasis en los problemas VSP y CSP.
- La sección 4 describe los algoritmos utilizados para el ordenamiento de vehículos y tripulación y cómo se realiza su ejecución.
- La sección 5 contiene la descripción de los requerimientos establecidos por los usuarios que fueron incluidos en FingLab, las restricciones y el ciclo de trabajo con FingLab.
- La sección 6 contiene el diseño de la interfaz gráfica de FingLab en base a los estudios realizados y los requerimientos recabados, las herramientas similares en el mercado y los prototipos implementados.

- La sección 7 contiene una breve descripción de la arquitectura de FingLab y sus subsistemas.
- La sección 8 describe el proceso de desarrollo de FingLab, cómo se llegó a su implementación final y el ciclo de trabajo completo mediante un ejemplo demostrativo de todas las funcionalidades implementadas
- La sección 9 presenta una demo con FingLab.
- La sección 10 presenta las conclusiones del trabajo realizado y plantea el trabajo futuro.
- La sección 11 presenta los agradecimientos a quienes contribuyeron en este proyecto.
- Al final del presente informe se encuentran los anexos a los que se hace referencia en el documento principal:
    - Anexo de estado del arte de las interfaces.
    - Anexo de planificación del transporte.
    - Anexo de documentación del proyecto.
    - La documentación técnica y de usuario de FingLab se entrega aparte junto con el CD que contiene la implementación del sistema.

2 Las interfaces de usuario

Como el objetivo es construir una interfaz gráfica de usuario, realizamos un estudio del área de interfaces y extrajimos aquellos conceptos que nos permitieron construir un sistema que no solo cumple con los requerimientos funcionales, sino que también el usuario se sienta cómodo usándolo y que por lo tanto, realmente lo use. Muchas veces un sistema cumple con lo que debe hacer, pero no significa que los usuarios lo encuentran fácil de usar o simplemente la forma en que la interfaz les provee información no les es útil para sus tareas. El construir un sistema usable, es decir, que el usuario pueda realmente interactuar con él y extraer la información que necesite, es imprescindible cuando se trata de interfaces, ya que éstas son las que acercan al usuario las herramientas que realizan el trabajo computacional y las ponen a su disposición. Por ello, es importante entender cómo llegamos a las interfaces de hoy en día, encontrar modelos que nos sirvieran de apoyo para comenzar el diseño del sistema y poder definir métricas que guiaran la construcción de la interfaz, para así terminar con un resultado que cumple con las expectativas del usuario [36].

El área que estudia el diseño de interfaces, así como la interacción de los usuarios con los sistemas computacionales es la **interacción persona-computadora** o interacción hombre-computadora (o **HCI** del inglés "Human-Computer Interaction") y del cual se desprenden los conceptos aplicados en la implementación de FingLab [32]. El área de HCI es muy amplia y abarca muchas disciplinas, por lo que nos enfocamos en aquellos aspectos relacionados directamente con nuestro proyecto. En el Anexo: Estado del arte de las interfaces se desarrolla más en detalle el estudio realizado.

Como parte de este estudio resaltamos la importancia de crear un **sistema enfocado en el usuario**. Debemos entender la motivación de cada usuario para usar el sistema y su perfil académico, ya que afecta su nivel de comprensión de los diferentes aspectos del sistema. El usuario debe tener su aporte en el diseño de la interfaz desde el comienzo, ya que así se puede verificar de forma temprana si el sistema está siguiendo las preferencias del usuario, que es quien eventualmente va a terminar usando el sistema [23, 28, 44, 49, 55]. Para ello, es imprescindible realizar **prototipos** que le permitan dar al usuario una idea de cómo va a lucir el sistema y así los desarrolladores pueden ver si las ideas que se manejan están siendo comprendidas realmente. Estos prototipos también sirven para detectar posibles problemas de diseño e incluso para definir los requerimientos [6, 9]. A lo largo del proceso de diseño se realizaron prototipos y se establecieron varias reuniones con los usuarios para entender de manera temprana sus perfiles y necesidades.

Un criterio importante es que el sistema presente **mensajes de error** adecuados, que permitan guiar al usuario en cuanto a las acciones a seguir, poder deshacer una acción cuando sea posible y darle una idea de si el error es algo inesperado o si es algo que el usuario realizó de manera incorrecta y cómo solucionarlo. Esto le da independencia al usuario y le da la confianza de utilizar el sistema y sentir que su trabajo está seguro (que no va a perder todo por un error) [49]. Como parte de este punto, FingLab brinda mensajes de éxito, advertencia o error en todas sus operaciones y tiene un mecanismo de logging (registro de acciones y errores) para así poder detectar cuál fue el problema causado y detectar el error si es un problema de implementación.

Una **documentación** que explique en forma detallada las acciones a seguir y a cada paso los errores que pueden ocurrir es también indispensable para dar soporte al usuario [49]. FingLab brinda ayuda y asistencia al usuario en varios niveles:

- Manuales de usuario: documentos dirigidos a ambos tipos de usuario. Ver Anexo: Documentación para más detalles.
- Manuales para desarrolladores: documentos dirigidos a aquellos usuarios que deseen continuar el desarrollo de FingLab. Ver Anexo: Documentación para más detalles.
- Ayuda en línea: dentro de FingLab se puede visualizar un resumen de los manuales de usuario.
- Ayuda contextual: etiquetas en los formularios que proveen de una explicación de lo que se debe ingresar en un determinado campo.

Estudiar cómo el usuario centra su **foco de atención** también es vital, especialmente cuando tratamos de un sistema que provee varios tipos de información. Debemos marcar o resaltar de algún modo el área de trabajo activa, es decir, la parte de la ventana de la aplicación en la cual el usuario está trabajando actualmente, para así mantener la atención visual dentro de un área específica.

Figura 2-1: Ejemplo de área de trabajo activa (marco azul) y no activa (sin marco azul) en FingLab.

También se plantea ocultar ciertas áreas de la pantalla cuando estas no sean necesarias para una parte del proceso de trabajo con el sistema para que el usuario pueda tener menos distracciones de su foco de atención y maximizar la información en la que se está concentrando en el momento [24]. En la interfaz gráfica de FingLab la zona activa de la pantalla se marca con un borde de color que la diferencia de otros sectores de la pantalla (como muestra la Figura 2-1) y todos los paneles se pueden contraer o expander a gusto del usuario. FingLab también permite el manejo de escalas para realizar acercamientos o alejamientos en el gráfico, lo cual permite centrar la atención del usuario según sus necesidades.

El **uso del color** debe ser analizado cuidadosamente. Los colores muy fuertes o demasiado contrastantes pueden perturbar la visualización de la información, así como desviar el foco de atención del usuario. Cuando se utilizan colores para la visualización se debe proveer de funcionalidades que permitan la personalización al gusto del usuario. De esta forma el usuario puede cambiar el "look and feel" (la apariencia) del sistema a su gusto [24]. En FingLab la utilización del color es importante sobre todo para: señalar los campos obligatorios en un formulario, marcar la zona activa de la interfaz, indicar una situación de éxito, advertencia o error, y diferenciar los tipos de datos que se están visualizando dentro de un gráfico (como muestra la Figura 2-2).

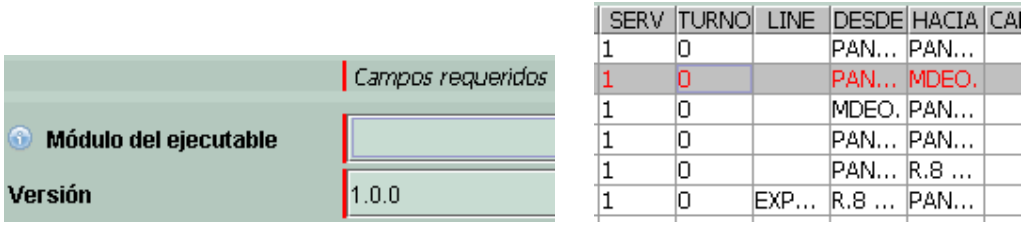


Figura 2-2: Ejemplos de uso de color para indicar campos requeridos en un formulario (izquierda) y señalar los datos seleccionados en una tabla (derecha) en FingLab.

También la iconografía fue cuidadosamente seleccionada para representar cada funcionalidad y que sea consistente en toda la interfaz (como muestra la Figura 2-3).

Figura 2-3: Ejemplos de iconografía en FingLab.

Existen **patrones de diseño** que han sido estudiados y analizados que nos permiten guiar el diseño de un sistema. Estos patrones especifican el problema que atacan y cómo resolverlo [23, 27]. Hay patrones que se utilizaron en el diseño de la arquitectura (para implementar funcionalidades complejas) y otros que se refieren al diseño de la interfaz de usuario (identificar elementos de la interfaz que podemos incluir y cómo construirlos para que el usuario los pueda utilizar sacándoles el mayor provecho). Los patrones de diseño de arquitectura utilizados se enumeran en la sección 7.2.1 Patrones de diseño. Los patrones de interfaz de usuario se aplicaron al implementar FingLab en: la organización de los ítems de los menús, el diseño de los formularios, la organización de la interfaz, el desarrollo del gráfico y el acceso a funcionalidades (acceso por teclado y mouse, botones, menús, hot keys –teclas de acceso rápido- y opciones que aparecen en pantalla al presionar el botón derecho del Mouse sobre una sección de la interfaz). Las aplicaciones de patrones en la interfaz de usuario se tomaron en cuenta en los prototipos construidos que se describen con más detalle en la sección 6.7 Prototipos.

Otro punto a tomar en cuenta es la **internacionalización**, es decir, una interfaz que sea competitiva en el mercado debe estar en diferentes idiomas. Por ello, todo texto que aparezca en la interfaz debe poder ser modificado a cualquier lenguaje de una forma automática, sin tener que revisar el código línea por línea [47]. FingLab permite la internacionalización de toda su interfaz, todos los mensajes y textos que se despliegan en pantalla se pueden traducir mediante un archivo de texto e incorporarlos sin necesidad de alterar la implementación (esto se explica más en detalle en el Anexo: Documentación).

Una parte fundamental del diseño de FingLab fue el estudio de **herramientas similares en el mercado** que nos pueden guiar en cuanto a los elementos que son deseables en este tipo de sistemas (especialmente los enfocados al área de transporte de pasajeros, en particular el ordenamiento de flota y su tripulación). En el diseño centrado en el usuario se debe analizar la competencia para poder encontrar esos puntos que hacen a un sistema fuerte en el mercado y también encontrar sectores del mismo que no están siendo abarcados (hay que pensar en los clientes, pero también en los clientes potenciales). Para ello se analizó qué tipo de visualizaciones se adaptan a las necesidades de cada usuario y si la forma en que se muestra la información le es útil para las tareas que debe realizar. Estas herramientas son analizadas con más detalle en el Anexo: Planificación del transporte. Nuevamente, la retroalimentación por parte del usuario fue fundamental y un prototipo fue creado específicamente para testear su reacción frente a la visualización. El tipo de gráfico y la estructura de las herramientas en el mercado determinaron en gran parte la visualización implementada en FingLab.

3 Planificación del transporte público

Como FingLab se enfoca al trabajo con algoritmos que resuelven problemas en el área del transporte público de pasajeros, realizamos un breve estudio del área para entender los retos que enfrenta y las soluciones propuestas. En el Anexo: Planificación del transporte se informa con más detalle el estudio realizado.

Se denomina "transporte" al traslado de personas o bienes de un lugar a otro. El transporte es una actividad fundamental de la Logística que consiste en colocar los productos de importancia en el momento preciso y en el destino deseado. Dentro de "transporte" se incluyen numerosos conceptos; los más importantes son infraestructuras, vehículos y operaciones. Los transportes pueden también distinguirse según la posesión y el uso de la flota. Por un lado, está el transporte público, en el que los vehículos son utilizables por cualquier persona previo pago de una cantidad de dinero. Por otro, está el transporte privado, aquél que es adquirido por personas particulares y cuyo uso queda restringido a sus dueños [75].

En los sistemas de transporte público intervienen dos actores [30]:

- *Usuarios*: Son todas aquellas personas que tienen necesidades de transporte y que deben utilizar tiempo y dinero para satisfacerlas. También llamados pasajeros.
- *Operadores*: Son aquellos que brindan a los usuarios el servicio de transporte, proveyendo para ello recursos económicos como son los vehículos, el combustible, la mano de obra y el mantenimiento. También llamados empresas de transporte.

Los objetivos de los usuarios y los operadores son contrapuestos, por lo que en contextos donde existe regulación, es responsabilidad de las autoridades determinar un nivel de compromiso adecuado. Desde el punto de vista de los operadores, se desea satisfacer la demanda cumpliendo con algunos objetivos y restricciones como:

- Minimizar los costos (de combustible, de mano de obra, de mantenimiento de depósitos, vehículos).
- Ubicar los depósitos de los vehículos (garajes).
- Ubicar las paradas de los vehículos.
- Reconocer los lugares y tiempos de tráfico.
- Reconocer los lugares y tiempos de demanda.
- Asociar viajes y turnos a vehículos (ordenamiento).
- Asignar turnos a mano de obra (listado de conductores).
- Obedecer la señalización.
- Obedecer las capacidades de las unidades.
- Cumplir con los tiempos de descanso de la mano de obra.
- Coordinar con otros sistemas de transporte (para que los clientes puedan realizar conexiones).
- Respetar zonas como hospitales, escuelas, etc. (pueden determinar si no se puede pasar por un lugar o si se debe pasar con una cierta frecuencia especialmente en determinadas horas).
- Respetar los tipos de pavimento (que pueden determinar por dónde ir).

Para los actores que cumplen el rol de planificación y regulación del transporte urbano y la urbanización en general, los problemas que se plantean son además:

- Controlar el tráfico con semáforos (permitir un correcto control del tráfico con la menor cantidad de semáforos).
- Establecer la señalización (para evitar cuellos de botella en ciertas ventanas de tiempo o en ciertos lugares).
- Seguridad vial (controlar la seguridad de las personas al realizar la señalización)

Desde el punto de vista de los usuarios, los objetivos pueden incluir:

- Recibir el servicio con costo adecuado a la distancia.
- Tiempo de recorrido mínimo.
- Evitar demoras (embotellamientos).
- Que el transporte pase con la frecuencia deseada para tener siempre disponibilidad.
- Que el transporte pase por los lugares deseados para tener siempre disponibilidad.
- Que el transporte pase en cierto intervalo de tiempo (ventanas de tiempo).

Todos estos elementos interactúan en la planificación del transporte público, que consta de las etapas que se muestra en el diagrama de la Figura 3-1 (imagen extraída de [45]).

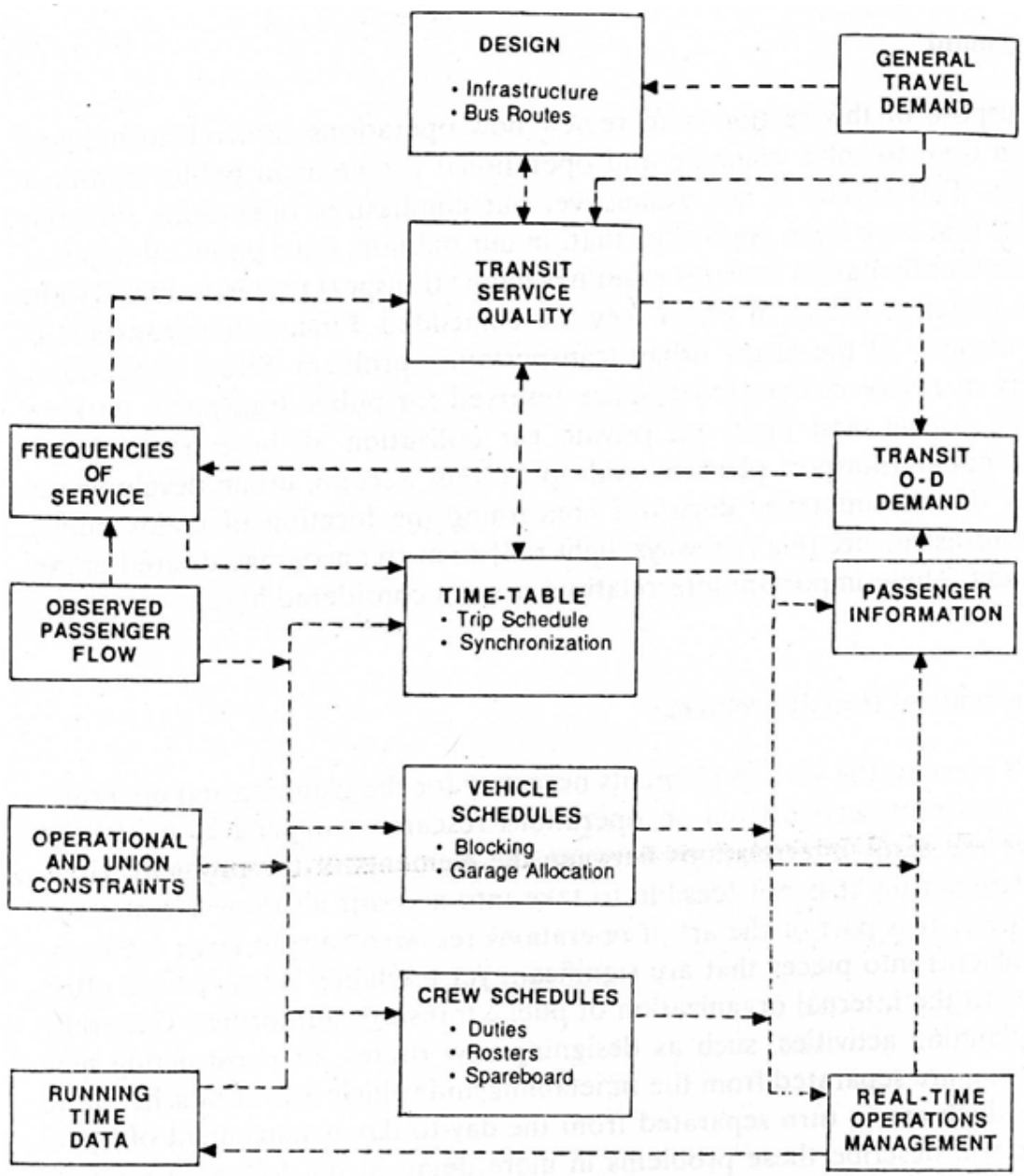


Figura 3-1: Etapas de la planificación del transporte de pasajeros.

En las distintas etapas del proceso de planificación del transporte público urbano se resuelven problemas de distintos tipo:

- *Estratégicos*: desarrollo a largo plazo del sistema como ser la construcción de infraestructuras (túneles, vías), determinación de los recorridos de buses, etc.
- *Tácticos*: incluye problemas como revisiones menores de rutas y asignaciones de frecuencias a rutas.
- *Operacionales*: incluye todas las tareas de ordenamiento de vehículos, tripulación y asignación de conductores a turnos y ómnibus (despacho).

En la Tabla 3-1 (tabla extraída de [13]) se resumen los objetivos y pasos de cada una de estas tareas (planificación, ordenamiento y despacho).

Fase	***Planificación***	***Ordenamiento***	***Despacho***
Alcance	Largo plazo	Período del diario de actividades	Día de la operación
Objetivo	Nivel de servicio	Reducción de costos	Hacerlo efectivamente
Pasos	Diseño de redes Planificación de líneas Realizar los diarios de actividades Planificación de precios	Ordenamiento de vehículos Ordenamiento de tareas Asignación de tareas	Asignación de personal Manejo de retrasos Manejo de errores Manejo de depósitos

Tabla 3-1: Objetivos y pasos del proceso de planificación del transporte de pasajeros.

Desarrollaremos los temas que tienen que ver con el ordenamiento de vehículos y tripulación dado que son estos problemas los que tienen relación directa con este proyecto.

Nota: las problemáticas del ordenamiento y asignación de turnos ocurren en otras áreas de nuestra sociedad, en las cuales muchas de las soluciones dadas en el área de transporte también son aplicables, como por ejemplo: cultivos forestales (en cada parcela de tierra se debe cultivar un vegetal diferente en cada estación del año para que se absorban ciertos minerales y dar tiempo a que se restaure la tierra para el cultivo nuevamente del mismo vegetal), turnos de personal (como en hospitales, estaciones y servicios que tienen varios turnos en el día), organización de salones de clase y grupos de alumnos, organización de tareas de un proyecto.

3.1 Modelos de planificación operacional

En todos los casos de problemas de ordenamiento se reconocen los siguientes elementos comunes:

- *Tiempo*: es una propiedad clave ya que se trata de establecer una organización de elementos en el tiempo. Lo que cambia de un caso a otro es la escala de dicho tiempo. En algunos casos, como viajes, aulas o turnos de enfermeras se trata de una escala de horas y minutos. En otros, como organización de exámenes y tareas de un proyecto, se trata de días, no importa la hora del día, pero sí importa en qué día o semana debe ocurrir un evento para que obedezca las restricciones del problema. Por último, en los más extremos, como es el caso de los cultivos forestales, se establece una escala semanal o hasta mensual, ya que son tareas que llevan muchas semanas a la vez. Existen tareas que abarcan todas las unidades del tiempo (como veremos en nuestro caso, hablamos de viajes que ocurren a una hora y minutos particulares, a lo largo de todo un turno, a lo largo de un día, que se repite varias veces a la semana durante meses – esto nos lleva a que si establecemos una línea de tiempo, debemos establecer diferentes vistas con diferentes unidades de tiempo, dependiendo de qué tan particulares son los datos que mostramos).
- *Recursos*: son los elementos que se van a asignar u ordenar en el tiempo. Pueden ser materiales: salones, medios de transporte, o pueden ser personas o eventos que ocurren.
- *Restricciones*:

- pueden ser restricciones explícitas: dos actividades que no pueden ser hechas o que no pueden ocurrir en el mismo tiempo (porque se superponen y tienen recursos en común) o en el mismo espacio (porque no alcanza el recurso espacio) o ambos.
- Pueden ser implícitas, como las leyes que se tienen que cumplir de personal.
- Pueden ser restricciones de antecedencia, dos cosas no pueden ocurrir simultáneamente porque una necesita que la otra ocurra primero.

- Requiere de mucho tiempo esfuerzo hacerlas, se trata de simplificar el proceso.
- Son actividades costosas y se trata de minimizar el costo (optimizar):
  - Costo de esfuerzo.
  - Costo de tiempo.
  - Costo de dinero.

En esta área, para el caso del transporte público se considera que se conoce la demanda de servicio en cada ruta, las frecuencias, y todas las restricciones operacionales y de personal. El problema consiste en asignar el conjunto de todos los viajes (recorrido efectuado por un ómnibus entre dos puntos geográficos de la ciudad) a un conjunto de vehículos y conductores con el costo operacional mínimo. El objetivo es definir:

- Diarios de actividades (planilla o conjunto de planillas con las actividades, eventos u horarios que cumplen los servicios o el personal).
- Diarios de vehículos.
- Diarios de personal.

Estos problemas interactúan entre sí como se muestra en el diagrama de la Figura 3-2 [21, 45, 51] (imagen extraída de [51]).

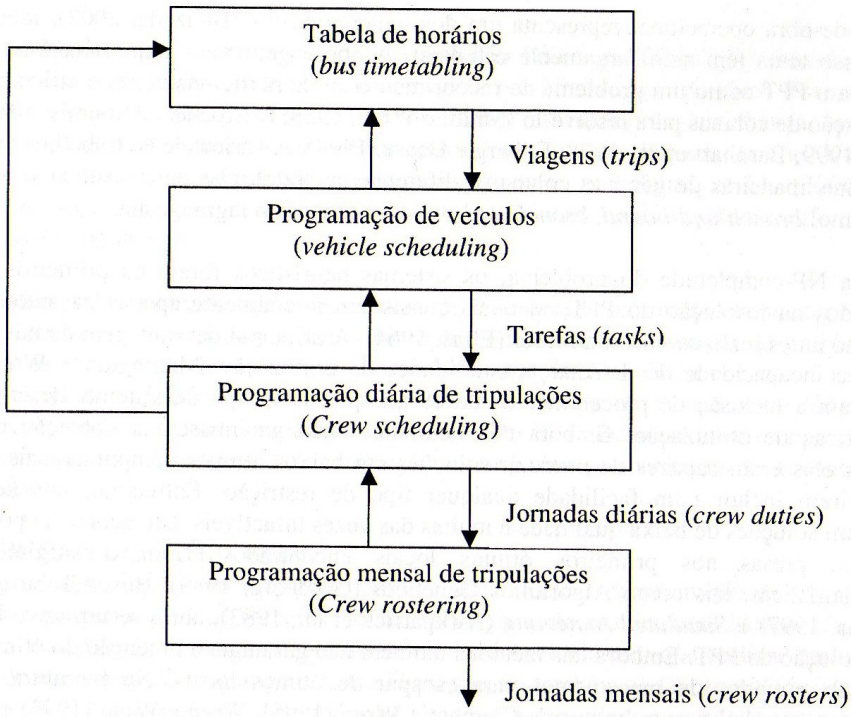


Figura 3-2: Interacción de los problemas en la planificación del transporte de pasajeros.

En la planificación del servicio de transporte de una ciudad, el transporte público es dividido en un conjunto de líneas, cada cual normalmente identificada por un número, y que corresponde a un recorrido a ser realizado por un vehículo de un punto a otro de la ciudad. Para cada línea la respectiva frecuencia está basada en la demanda. Después de esto, una tabla de horarios es construida, dando como resultado viajes, a los cuales les son asociados un lugar de origen y otro de destino y un horario de inicio y otro de fin del viaje.

Conocidos los viajes a ser realizados por la empresa, el problema siguiente es la programación de vehículos. En este caso, el objetivo es hacer el dimensionamiento de la flota y la asignación de los vehículos a los viajes, de forma que los viajes programados para las diversas líneas de la empresa sean realizados. De la programación de los vehículos resulta un conjunto de viajes realizados por cada vehículo diariamente, desde la salida del garaje, hasta su retorno al mismo.

Para resolver el problema siguiente, de la programación diaria del personal, los viajes de cada vehículo, resultantes del problema de programación de vehículos, son agrupados en tareas. Una tarea es un conjunto de viajes a ser realizados por el mismo personal y en general no hay posibilidad de intercambiar personal en el transcurso de la tarea. A partir de ese conjunto de tareas de todos los vehículos, son generadas las jornadas diarias del personal, resultantes de la resolución del problema de programación diaria del personal. En la generación de las jornadas diarias del personal, una serie de reglas laborales y operacionales de la empresa deben ser cumplidas.

Una vez generadas las jornadas diarias de días útiles, sábados, domingos y feriados, estas deben ser ahora combinadas, de forma de hacer la escala a nivel mensual del personal, problema conocido como asignación de personal. En la elaboración de la escala mensual, se observa que una parte de las leyes laborales, de los acuerdos colectivos y de las reglas operacionales es contemplada en la confección de las jornadas diarias. Sin embargo, otras tantas deberán ser observadas, como por ejemplo:

- Que cada seis días de trabajo el personal tenga un día libre.
- El personal que hacen jornada doble tienen día libre obligatorio los sábados y domingos.

En una escala mensual se busca entre otros objetivos:

- Minimizar o eliminar los cambios de horario de trabajo del personal.
- Equilibrar las horas mensuales trabajadas por el personal.
- Reducir los cambios de líneas del personal a lo largo del mes.

En el diagrama de la Figura 3-3 (imagen extraída de [17]) se indican las entradas y las salidas de cada etapa de planificación.

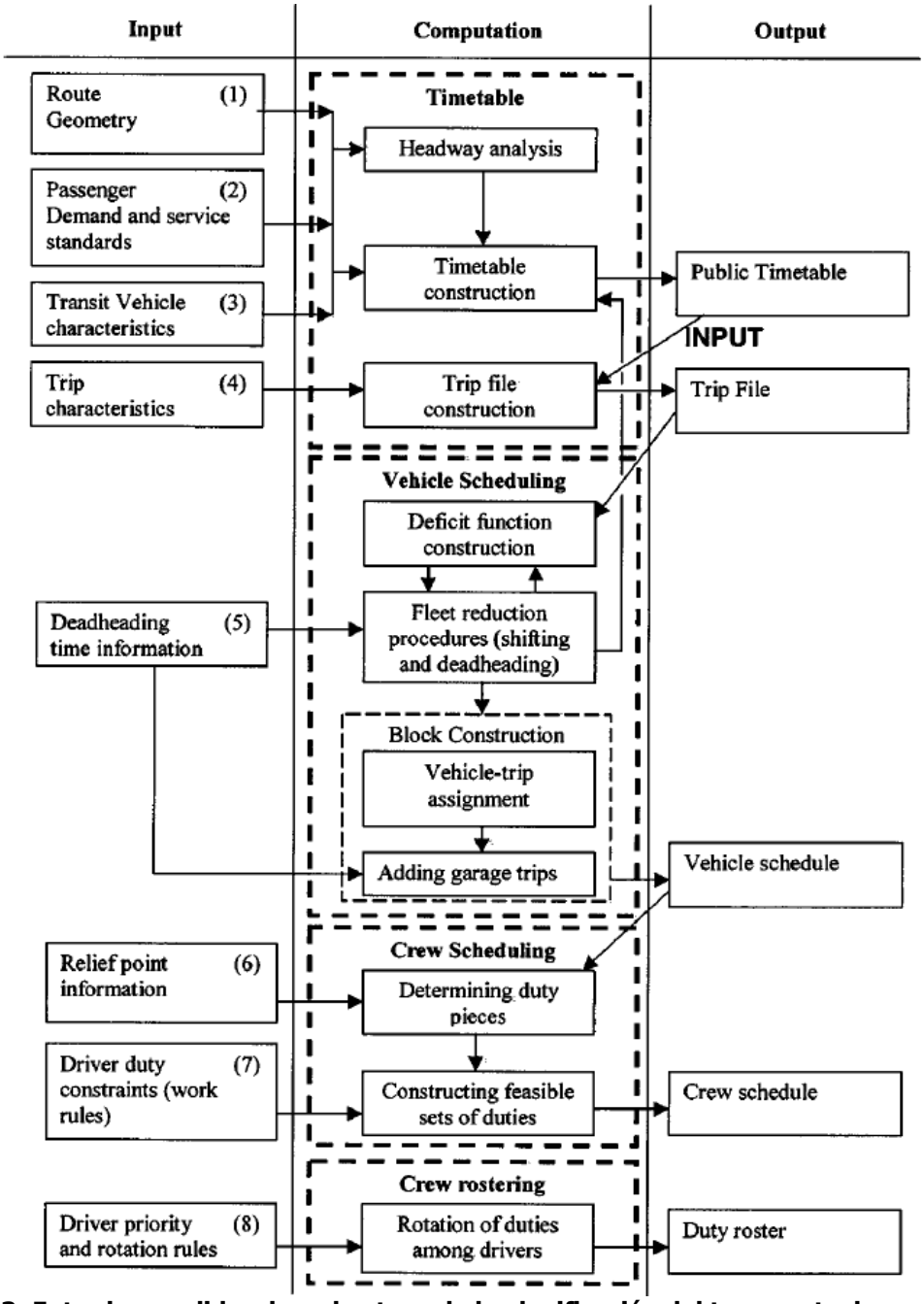


Figura 3-3: Entradas y salidas de cada etapa de la planificación del transporte de pasajeros.

3.2 Soluciones a los problemas del transporte público

La organización eficiente del transporte urbano es un aspecto fundamental tanto en países desarrollados como los en vías de desarrollo, ya que una proporción importante de los viajes en las ciudades medianas y grandes son efectuados utilizando el mismo [15, 38, 78]. Muchas veces no es necesario tener más recursos para cumplir con los objetivos, sino que una correcta ubicación o reorganización de los recursos existentes puede tener efectos considerables al cumplir con las demandas de ambas partes.

Problemas como la asignación de flota y personal, así como optimización de rutas y frecuencias, han recibido amplio tratamiento, contándose con modelos de optimización para los cuales se dispone de algoritmos eficientes de resolución [12, 19, 20, 26, 29, 33, 37, 38, 45, 46, 51, 54]. Con el paso del tiempo ha habido una gran expansión en el área, por un lado, hacia modelos que incorporen cada vez más características de la realidad, y, por otro, en la búsqueda de algoritmos que permitan resolver los problemas de manera eficiente [1, 2, 10, 14, 18].

Estos modelos y algoritmos deben su éxito, en cierta medida, a la evolución de los sistemas informáticos. El crecimiento en el poder de cómputo y la baja en sus costos, ha permitido disminuir los tiempos de ejecución de los algoritmos. Además, el desarrollo de los Sistemas de Información Geográfica resulta fundamental para lograr una adecuada interacción de los modelos y algoritmos con los encargados de realizar la planificación estratégica.

Dada su complejidad, estos problemas se resuelven utilizando algoritmos heurísticos. Cuando se trabaja con heurísticas, no existe una única manera de encontrar soluciones a los problemas de optimización y no es posible automatizar completamente la resolución de un problema mediante un modelo de optimización. Incluso es imprescindible brindar una mejora de las soluciones mediante modificaciones manuales, que requieren de la experiencia del planificador. Por ello, es necesario una planificación del transporte público urbano colectivo basada en herramientas de apoyo a la decisión que permitan analizar la realidad y comparar las distintas alternativas de acción posibles. Existen múltiples herramientas en el mercado que se dedican a esta área las cuales son analizadas en Anexo: Planificación del transporte.

4 Descripción del proceso de ejecución de los algoritmos de ordenamiento de vehículos y tripulación

El Departamento de Investigación Operativa del Instituto de Computación (InCo) de la Facultad de Ingeniería desarrolla algoritmos para resolver el problema de **planificación y ordenamiento de vehículos y tripulación** para una compañía de transporte colectivo urbano y suburbano de pasajeros. En particular, están desarrollando un algoritmo integrado que resuelva los dos problemas simultáneamente. Los algoritmos están implementados en lenguaje C y C++. Su ejecución se realiza desde consola y toman archivos en formato de texto plano como entrada y devuelven conjuntos de soluciones también en archivos en formato de texto plano. Llamaremos "soluciones" a los resultados obtenidos en las salidas de las ejecuciones de los algoritmos y consisten en conjuntos de uno o varios diarios de servicios.

Los algoritmos se ejecutan de forma iterativa, ya que primero se genera un conjunto de soluciones iniciales, y luego en las sucesivas ejecuciones, dichas soluciones se mejoran. Se desarrollaron dos tipos de algoritmos:

- *Algoritmo de soluciones iniciales*: genera las soluciones a partir de los datos de entrada que resuelven el problema de ordenamiento de vehículos. Los datos de entrada es una instancia de todos los viajes que la empresa debe cumplir en sus servicios.
- *Algoritmo integrado*: genera a partir de las salidas del algoritmo de soluciones iniciales las soluciones que resuelven el problema de ordenamiento de vehículos y personal simultáneamente. Son soluciones mejoradas de las obtenidas por el algoritmo de soluciones iniciales.

El proceso de ejecuciones de los algoritmos es el que se muestra en el diagrama de la Figura 4-1.

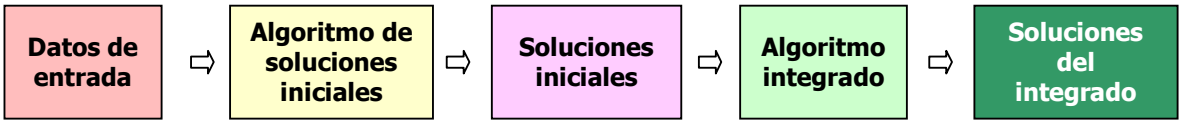


Figura 4-1: Proceso de ejecución de los algoritmos.

Antes de que un archivo sirva como entrada de una ejecución se le pueden hacer cambios manuales (por ejemplo, eliminar un diario del conjunto de soluciones). Es posible que cuando se obtenga una solución, existan viajes libres, es decir, que no estén asignados a ningún servicio del diario. **Por ello, FingLab permite no solo la visualización de los diarios, sino su modificación, ofreciendo funcionalidades para armar un servicio manualmente o incluso desarmar un servicio por completo**.

4.1 Ejecución de los algoritmos

La ejecución de los algoritmos se realiza por consola (o mediante un script que contiene la línea que realiza la llamada) con el siguiente formato:

```
<nombre ejecutable> <-x valor>
```

donde x es para que el ejecutable sepa cuál es el valor de parámetro que se le pasa a continuación y valor es el valor de dicho parámetro. Puede tener varios parámetros con el formato `<-x valor>`. Los algoritmos pueden estar implementados en cualquier lenguaje de programación y funcionan como unidades independientes, es decir, no dependen de ninguna instalación particular (ya sea de software o hardware), por lo que pueden ser ejecutados en cualquier computadora.

4.2 Entradas de los algoritmos

Los datos de entrada del algoritmo de soluciones iniciales contienen los viajes que deben ser asignados a servicios e información adicional necesaria para la creación de los servicios (distancias, tiempos, etc.). Estos datos se ingresan en un archivo de texto plano con el formato tabulado que se muestra en el ejemplo del diagrama de la Figura 4-2.

#Calidad	Comienzo de la tabla
	Encabezado de la tabla
CodCalidad Nombre	Nombre de columnas
3 Preferencial 2 Comun 0 NoSe	Datos
#Otra tabla	Fin de la tabla

Figura 4-2: Formato de los datos de entrada del algoritmo de soluciones iniciales.

4.3 Salidas de los algoritmos

Las soluciones consisten en un conjunto de diarios. Cada diario o libro tiene un conjunto de servicios y cada servicio puede tener hasta tres turnos. La estructura de los archivos de salida en texto plano es tabulada como se muestra en el ejemplo del diagrama de la Figura 4-3.

Solucion 1	Comienzo de la solución
Tiempo total = 77 segundos … Cant. servicios = 188	Encabezado de la solución
Servicio 1	Comienzo de tabla externa (servicio)
Hora comienzo = 3:10 … Capacidad restante = 2:10 Lista de turnos:	Encabezado de tabla externa (servicio)
-<0>-- Datos generales del turno:	Comienzo tabla interna (turno del servicio)
Cantidad de Kms = #	Encabezado tabla interna (turno del servicio)
Evento Duracion Origen	Nombre de columnas de tabla interna (turno del servicio)
Viaje 50 PANDO … Viaje 45 SUAREZ Viaje 50 PANDO	Datos de tabla interna (turno del servicio)
-<1>-- Datos generales del turno:	Fin de tabla interna (turno del servicio) / Comienzo de la siguiente tabla interna (turno del servicio)
Cantidad de Kms = #	Encabezado de tabla interna (turno del servicio)
Evento Duracion Origen	Nombre de columnas de tabla interna (turno del servicio)
Viaje 30 PANDO … Viaje 15 PANDO Viaje 20 SUAREZ	Datos de tabla interna (turno del servicio)
Servicio 2 …	Fin de tabla externa (servicio) / Comienzo de la siguiente tabla externa (servicio)

Figura 4-3: Formato de los datos de salida de los algoritmos.

5 Requerimientos del sistema

En esta sección se describen los requerimientos del sistema FingLab planteados por los usuarios. Todos estos requerimientos fueron refinados y en algunos casos ampliados a los largo del proyecto. En la documentación técnica y de usuario de FingLab se indica cuáles de estos requerimientos fueron implementados.

5.1 Usuarios

Los usuarios finales serán los encargados de la planificación en la compañía de transporte urbano, quienes poseen conocimientos de la realidad que les permiten entender cualquier situación que se les plantee y además tienen una gran capacidad de aprendizaje respecto a lo que se les plantee.

Otros usuarios serán los responsables del proyecto por parte de la Facultad de Ingeniería quienes poseen los conocimientos de la temática, así como los conocimientos técnicos para comprender la implementación del sistema a construir.

Los usuarios técnicos son quienes experimentan con varios algoritmos y módulos y les interesa cambiar su configuración. Estos usuarios se encargan de dar de alta en el sistema los algoritmos que los usuarios no técnicos podrán ejecutar para resolver los problemas de planificación y ordenamiento. Su objetivo es experimentar y analizar las soluciones desde el punto de vista de la efectividad de los algoritmos, para poder seguir trabajando en su implementación.

Los usuarios no técnicos son aquellos a quienes les interesa planificar y ordenar su flota de vehículos y tripulación. No les interesa cómo estén implementados los algoritmos, solo les interesa ejecutarlos para obtener sus resultados y trabajar sobre las soluciones obtenidas.

Existen dos **tipos de usuario**:

- ***Desarrollador***: Estos usuarios ingresan al sistema con perfil "Desarrollador" y tienen privilegios de tipo administrador. Es decir, tienen acceso a todas las funcionalidades ofrecidas por el sistema y determinan los algoritmos, módulos y otras configuraciones que se hacen accesibles a los usuarios planificadores que acceden al sistema en la misma computadora (y con el mismo usuario de la computadora). Estos usuarios tienen conocimientos de:
    - Programación
    - Diseño
    - Algoritmia
    - Metaheurísticas
    - Optimización
    - Combinatoria
    - Simulación

  Y su cometido es utilizar **FingLab** como una herramienta de apoyo a:
    - Desarrollo y experimentación con algoritmos
    - Estudio de soluciones producto de algoritmos
    - Experimentación con diferentes formas de cálculos de costos y factibilidad

- ***Planificador***: Estos usuarios ingresan al sistema con perfil "Planificador" y tienen privilegios de tipo standard. Es decir, tienen acceso a algunas de las funcionalidades ofrecidas por el sistema y utilizan los algoritmos, módulos y otras configuraciones que hechas accesibles por los usuarios desarrolladores que acceden al sistema en la misma computadora (y con el mismo usuario de la computadora). Estos usuarios tienen conocimientos de:
    - Transporte
    - Planificación de servicios

  Y su cometido es utilizar **FingLab** como una herramienta de apoyo a:

- Planificación de vehículos
- Planificación de flota

5.2 Requerimientos Funcionales

A continuación se describen los principales requerimientos funcionales del sistema.

5.2.1 Administración de Usuarios

Como es un sistema cuyo diseño es centrado en el usuario, debemos comprender las características del mismo. Existen dos tipos de usuarios: los técnicos (también los llamaremos usuarios Administradores o Programadores) y los no técnicos (que también los llamaremos usuarios Comunes o Planificadores). La diferencia entre ambos es sobre todo cuando se trata del mantenimiento de los algoritmos y los módulos de cálculo de factibilidad y costos que son parte del sistema.

El sistema debe ser monousuario, sin autenticación, pero al inicio debe preguntar si es un usuario técnico o no para poder habilitar o deshabilitar opciones para facilitar su manejo. Es responsabilidad del usuario que cuando ingrese al sistema no realice cambios que perjudiquen su utilización.

5.2.1.1 Iniciar sesión

El sistema permite iniciar una sesión de trabajo al seleccionar el tipo de usuario que es: técnico o no. Al iniciar la sesión, se habilitan las opciones correspondientes al tipo de dicho usuario.

Las soluciones, módulos, algoritmos u otras componentes creadas, modificadas o eliminadas en el sistema en una computadora solo estarán disponibles para todos los usuarios que inicien sesión en esa computadora y con el mismo usuario del sistema operativo (no existe una conexión de red donde se compartan datos en un directorio común, por lo que un mismo usuario podrá iniciar sesión en diferentes computadoras). La habilitación de opciones será como muestra la Tabla 5-1.

Notación:

☑	Si, se habilita esta funcionalidad a este usuario
✗	No, no se habilita esta funcionalidad a este usuario

Funcionalidad	Usuario planificador	Usuario desarrollador
Iniciar sesión	☑	☑
Cerrar sesión	☑	☑
Cambiar/Crear espacio de trabajo	☑	☑
Dar alta configuración de algoritmo	✗	☑
Consultar configuración de algoritmo	☑	☑
Editar configuración de algoritmo	✗	☑
Dar baja configuración de algoritmo	✗	☑
Dar alta módulo	✗	☑
Consultar módulo	☑	☑
Editar módulo	✗	☑
Dar baja módulo	✗	☑
Dar alta utilidad	✗	☑
Consultar utilidad	☑	☑
Dar alta idioma	✗	☑
Dar baja idioma	✗	☑
Soporte de versiones	☑	☑
Realizar ejecución de algoritmo	☑	☑
Realizar ejecución de utilidad	☑	☑

Exportar	☑	☑
Importar	☑	☑
Abrir diario	☑	☑
Cerrar diario	☑	☑
Guardar diario	☑	☑
Imprimir diario	☑	☑
Realizar cálculo de costos de solución	☑	☑
Realizar cálculo de factibilidad de solución	☑	☑
Realizar consultas sobre el diario	☑	☑
Crear/modificar servicio manualmente	☑	☑

Tabla 5-1: Habilitación de funcionalidades a usuarios.

Cuando se inicia una sesión de trabajo, el sistema se inicia en un directorio de trabajo (o espacio de trabajo) en el cual estarán disponibles algoritmos, módulos, soluciones obtenidas u otras componentes que el usuario guarde en él. El sistema permite cambiar de espacio de trabajo y así el usuario puede guardar diferentes algoritmos y soluciones en diferentes espacios de trabajo.

5.2.1.2 Cerrar sesión

El sistema permite cerrar una sesión de trabajo. En caso de existir datos no guardados se despliega un mensaje de advertencia que permita seleccionar si se desea guardar o no.

5.2.1.3 Crear/cambiar espacio de trabajo

El usuario puede seleccionar un directorio en el cual se guardarán todas su configuraciones de algoritmos y ejecuciones. Este directorio se llamará espacio de trabajo y un usuario puede tener varios y crear nuevos en cualquier momento. Cada espacio de trabajo es independiente uno de otro, lo que hay en un espacio no se ve en otro. Un espacio se crea en un determinado idioma y cuando se lo accede se despliega la interfaz en dicho idioma.

5.2.1.4 Dar alta idioma

El sistema permite dar de alta aquellos archivos que componen el idioma, y que proveen todos los mensajes que se muestren en pantalla (incluyendo etiquetas de menú, botones, etc.). Cuando el usuario crea un espacio de trabajo, lo crea en un determinado idioma, y cada vez que acceda a un espacio la interfaz se carga con el idioma correspondiente.

5.2.1.5 Dar baja idioma

El sistema permite dar de baja un idioma. Si un espacio de trabajo creado con un cierto idioma se accede y dicho idioma fue dado de baja, el espacio no podrá ser accedido.

5.2.2 Algoritmos

Para crear las soluciones que finalmente se representarán en la interfaz gráfica, se deben ejecutar algoritmos que creen dicha solución. Para ello es necesario que el sistema provea los siguientes requerimientos:

5.2.2.1 Dar alta/editar configuración de algoritmo

El sistema permite crear una configuración de algoritmo con su ejecutable y definir los tipos de parámetros que recibe y los retornos que puede tener. Estas configuraciones quedan guardadas en el sistema y dicho algoritmo luego se puede seleccionar para utilizarse al realizar una ejecución con los datos de entrada. La creación de dicha configuración solo será válida para la computadora en la que se creó la configuración (no existe una conexión de red donde se compartan datos en un directorio común). El sistema realiza una copia del ejecutable al espacio de trabajo actual para tenerlo siempre disponible dentro de él. El sistema también permite dar de alta el módulo que permite la lectura-escritura de soluciones con el formato correspondiente al algoritmo. De esta manera, cuando se debe leer una solución creada con un cierto algoritmo, para dicha lectura se utilizará el módulo asignado a ese algoritmo. Esto permite que, frente a cambios en los formatos, las soluciones puedan seguir siendo leídas y que el sistema pueda seguir funcionando. La configuración puede ser editada en cualquier momento.

5.2.2.2 Consultar configuración de algoritmo

El sistema permite consultar todos los datos de una configuración de algoritmo ingresando el nombre de dicha configuración.

5.2.2.3 Eliminar configuración de algoritmo

El sistema permite eliminar una configuración de algoritmo ingresando el nombre de dicha configuración, solamente si no existe una solución que utilice dicha configuración (sino, dicha solución no se podrá abrir en el futuro por las clases de lectura y escritura). La eliminación de dicha configuración (y sus clases de escritura y lectura) solo será válida para la computadora en la que se eliminó la configuración (no existe una conexión de red donde se compartan datos en un directorio común).

5.2.3 Ejecuciones y soluciones

Existen algoritmos que son heurísticas y crean un grupo de soluciones iniciales y existen algoritmos que son genéticos y que pueden generar mejores soluciones a partir de esas soluciones iniciales de las heurísticas o de las soluciones de otros algoritmos genéticos. Es decir que las salidas de los algoritmos se van pasando de ejecución a ejecución hasta que finalmente queda una solución compuesta por un solo diario (el diario de menor costo que la secuencia de algoritmos aplicados pudo crear).

No hay un límite de cuántas ejecuciones se pueden realizar hasta llegar al diario solución final, ni un orden en particular en el que se deben ejecutar los algoritmos (puedo aplicar la secuencia de algoritmos A-A-B, o A-B-A-A, etc. y llegar a distintas soluciones finales). Las soluciones creadas, modificadas o eliminadas en el sistema en una computadora solo estarán disponibles para todos los usuarios que inicien sesión en esa computadora (no existe una conexión de red donde se compartan datos en un directorio común). De esta necesidad se desprende que el sistema provea los siguientes requerimientos:

5.2.3.1 Realizar ejecución

El sistema permite realizar la ejecución de los algoritmos que se ingresaron al mismo. El sistema permite crear una solución a partir de un archivo de datos que se permite seleccionar. Dicho archivo de datos contiene los datos para un tipo de día (hábil o no hábil) dentro de una estación del año y puede ser la salida de otra ejecución anterior. Los datos a ingresar para la creación de una solución son:

- Nombre de la solución a crear o elegir una de las soluciones ya existentes para agregar una nueva ejecución.
- Año a la que corresponde.

- Estación del año a la que corresponde.
- Tipo de día a la que corresponde.
- Ejecución de la que deriva (si se seleccionó una de las soluciones ya existentes, entonces se puede elegir de cuál de sus ejecuciones deriva, sino se considera que es una ejecución desde cero).
- Algoritmo que se aplicará al archivo de entrada para crear la solución.
- Archivo con los datos de entrada que se puede seleccionar desde el sistema de archivos del usuario o sino se toma el archivo de salida de la ejecución previa (si se seleccionó alguna).
- Nombre del archivo de salida.
- Los valores de los parámetros que recibe.
- Llamada completa para poder editar otros parámetros manualmente.

El año, estación, tipo de día, nombre de la solución y ejecución de la que deriva determinarán cómo se guardará la solución en la jerarquía de versiones del sistema como se describe en la sección 5.2.3.2 Soporte de versiones de ejecuciones.

Las soluciones creadas en el sistema en una computadora solo estarán disponibles para los usuarios que inicien sesión en esa computadora (no existe una conexión de red donde se compartan datos en un directorio común).

5.2.3.2 Soporte de versiones de ejecuciones

El sistema debe permitir el soporte de versiones de las distintas soluciones creadas por las ejecuciones. La generación de soluciones se realiza según el año, la estación del año (primavera, verano, otoño o invierno), el tipo de día (hábil si es de lunes a viernes, o no hábil si es sábado, domingo o feriado).

Cada solución está compuesta por: Entrada + algoritmo + salida.

La entrada es parte de la solución y depende del algoritmo que se aplique, por lo tanto se debe guardar dentro de la solución cada vez que se genera una nueva solución.

Todos los archivos se guardarán dentro del sistema, más allá de las copias que existan en el sistema de archivos propio del usuario. Las soluciones se guardan dentro de la estructura con el nombre especificado al realizar la ejecución, pero se pueden exportar al sistema de archivos propio del usuario (ver requerimiento exportar).

5.2.3.3 Exportar

El sistema permite exportar toda una solución (entrada, salida y configuración de todas las ejecuciones) o una sola ejecución de una solución o solo un archivo determinado. Al exportar también se puede cambiar el formato de los datos para transformarlos en un formato que sea aceptado por un sistema externo. Los formatos que se manejarán serán txt (tal cual es descripto en la sección 4 Descripción del proceso de ejecución de los algoritmos de ordenamiento de vehículos y tripulación), dbf, xml y base de datos relacionales.

5.2.3.4 Importar

El sistema permite importar toda una solución (entrada, salida y configuración de todas las ejecuciones) o una sola ejecución de una solución o solo un archivo determinado. Dicha importación solo será válida para la computadora en la que se realizó la importación (no existe una conexión de red donde se compartan datos en un directorio común). Al importar también se puede cambiar el formato de los datos para transformarlos en el formato aceptado por FingLab. Los formatos que se manejarán serán txt (tal cual es descripto en la sección 4 Descripción del proceso de ejecución de los algoritmos de ordenamiento de vehículos y tripulación), dbf, xml y base de datos relacionales.

5.2.4 Diario

Cuando la solución contiene al menos un diario, se permite realizar las siguientes operaciones:

5.2.4.1 Abrir diario

El sistema permite cargar en memoria un diario de una determinada solución para poder realizar consultas sobre el mismo y dar la posibilidad de realizar modificaciones. Dicha solución se encuentra en el formato especificado para las salidas de los algoritmos (ver 4.3 Salidas de los algoritmos). Si el archivo contiene varios diarios, se puede ingresar el código del diario que se desea cargar. También se puede ingresar el turno dentro de dicho diario para alivianar la carga de datos y cargar solo un determinado turno, en vez de todos. Según los datos disponibles se habilitarán o deshabilitarán las opciones de las funcionalidades disponibles para el trabajo con dicho diario.

5.2.4.2 Cerrar diario

El sistema permite cerrar la solución con la que se está trabajando actualmente. Si se realizaron cambios y no se guardó la solución, entonces se despliega un alerta que permita seleccionar si se desea guardar o no (ver la sección 5.2.4.3 Guardar diario). Al cerrar el diario se destruyen todos los objetos creados para la utilización de la interfaz gráfica.

5.2.4.3 Guardar diario

El sistema permite guardar un diario sobre el cual se realizaron modificaciones. El diario siempre se guarda como una versión nueva dentro del mismo archivo o medio del cual se levantó el original (no se sobrescriben).

5.2.4.4 Imprimir diario

El sistema permite imprimir el diario que está abierto actualmente en el formato especificado por el usuario que sea más conveniente para su uso. Se debe tener en cuenta que todos los datos entren en el formato de la hoja. La impresión puede ser a una impresora directamente o a un archivo. Si es a un archivo se debe tratar de proveer la posibilidad de cambiar el formato de dicho archivo, es decir, crear plantillas de impresión.

Ventajas: La impresión a un archivo provee de una forma de intercambiar la información con quienes no estén tan familiarizados con el formato de la solución de los algoritmos, o que necesiten una forma de visualización diferente a la provista por la interfaz gráfica, personalizar la presentación, además de dar la facilidad de enviar dicha información en formato digital.

5.2.4.5 Información de la solución

El sistema permite visualizar para una solución, los siguientes datos:

- Fecha de creación de la solución.
- Autor que creó la solución.
- Configuración de algoritmo utilizado para crear la solución (y abrirla y guardarla).
- Módulo de cálculo de costos utilizado y valores de los parámetros.
- Módulo de cálculo de factibilidad utilizado y valores de los parámetros.
- Comentario con cualquier otra información extra que se quiera añadir sobre la solución.

5.2.5 Calcular el costo de la solución

El sistema permite desplegar el costo de la solución actualmente abierta completo y también el desglosado según las ponderaciones actuales en las opciones del módulo de costos utilizado. El sistema provee de una clase interfaz para calcular los costos de la solución abierta. El usuario puede implementar su propia función de cálculo de costos que debe implementar dicha clase interfaz. Además se provee de la utilidad de definir los parámetros para dicha función de costos (de la misma manera que se definen los parámetros para los algoritmos – ver la sección 5.2.2.1 Dar alta/editar configuración de algoritmo).

De esta manera, las funciones de cálculo de costos se pueden actualizar sin tocar la implementación del sistema, solamente agregando el módulo que implemente la fórmula de costos.

De este requerimiento se desprenden las siguientes funcionalidades:

5.2.5.1 Dar alta/editar módulo de cálculo de costos

El sistema permite agregar módulos de cálculo de costos. Dichos módulos se identificarán por un nombre, tendrán un archivo que implemente la clase interfaz para el cálculo de costos y se podrán definir parámetros para su utilización. El sistema realiza una copia del módulo para tenerlo siempre disponible dentro de él. Dicho módulo quedará disponible para ser seleccionado para aplicarlo a una solución y en el archivo de configuración de dicha solución se guardarán los valores de cualquier parámetro que necesite la función de cálculo de costo (ponderaciones). Los módulos creados en el sistema en una computadora solo estarán disponibles para los usuarios que inicien sesión en esa computadora (no existe una conexión de red donde se compartan datos en un directorio común). La configuración del módulo puede ser editada en cualquier momento.

Restricción: la implementación de la clase de cálculo de costos debe ser hecha en el lenguaje de programación del sistema.

5.2.5.2 Modificar módulo de cálculo de costos

El sistema permite modificar los parámetros de un módulo de cálculo de costos para una determinada solución. Los datos modificados en el sistema en una computadora solo estarán disponibles para los usuarios que inicien sesión en esa computadora (no existe una conexión de red donde se compartan datos en un directorio común).

5.2.5.3 Eliminar módulo de cálculo de costos

El sistema permite eliminar módulos de cálculo de costos. Dichos módulos se identificarán por un nombre. Si hay una solución que utilice dicho módulo, cuando se intenten calcular los costos se desplegará un mensaje de advertencia y se deberá seleccionar un módulo que esté disponible e ingresar los valores de los parámetros para ese módulo (los valores de los parámetros del módulo anterior se eliminarán). Los módulos eliminados en el sistema en una computadora solo se eliminarán para los usuarios que inicien sesión en esa computadora (no existe una conexión de red donde se compartan datos en un directorio común).

5.2.6 Calcular la factibilidad de la solución

El sistema permite evaluar la factibilidad de la solución (es decir, si cumple por ejemplo, con las reglas laborales estipuladas por ley para los empleados) actualmente abierta utilizando un módulo de cálculo de factibilidad disponible. El sistema provee de una clase interfaz para calcular la factibilidad de la solución abierta. El usuario puede implementar su propia función de cálculo de factibilidad que debe implementar dicha clase interfaz. Además se provee de la utilidad de definir los parámetros para dicha función de factibilidad (de la misma manera que se definen los parámetros para los algoritmos – ver la sección 5.2.2.1 Dar alta/editar configuración de algoritmo).

De esta manera, las funciones de cálculo de factibilidad se pueden actualizar sin tocar la implementación del sistema, solamente agregando el módulo que implemente la fórmula de costos.

De este requerimiento se desprenden las siguientes funcionalidades:

5.2.6.1 Dar alta/editar módulo de cálculo de factibilidad

El sistema permite agregar módulos de cálculo de factibilidad. Dichos módulos se identificarán por un nombre, tendrán un archivo que implemente la clase interfaz para el cálculo de factibilidad y se podrán definir parámetros para su utilización. El sistema realiza una copia del módulo para tenerlo siempre disponible dentro de él. Dicho módulo quedará disponible para ser seleccionado para aplicarlo a una solución y en el archivo de configuración de dicha solución se guardarán los valores de cualquier parámetro que necesite la función de cálculo de factibilidad (ponderaciones). Los módulos creados en el sistema en una computadora solo estarán disponibles para los usuarios que inicien sesión en esa computadora (no existe una conexión de red donde se compartan datos en un directorio común). La configuración del módulo puede ser editada en cualquier momento.

Restricción: la implementación de la clase de cálculo de factibilidad debe ser hecha en el lenguaje de programación del sistema.

5.2.6.2 Modificar módulo de cálculo de factibilidad

El sistema permite modificar los parámetros de un módulo de cálculo de factibilidad para una determinada solución. Los datos modificados en el sistema en una computadora solo estarán disponibles para los usuarios que inicien sesión en esa computadora (no existe una conexión de red donde se compartan datos en un directorio común).

5.2.6.3 Eliminar módulo de cálculo de factibilidad

El sistema permite eliminar módulos de cálculo de factibilidad. Dichos módulos se identificarán por un nombre. Si hay una solución que utilice dicho módulo, cuando se intente calcular la factibilidad se desplegará un mensaje de advertencia y se deberá seleccionar un módulo que esté disponible e ingresar los valores de los parámetros para ese módulo (los valores de los parámetros del módulo anterior se eliminarán). Los módulos eliminados en el sistema en una computadora solo se eliminarán para los usuarios que inicien sesión en esa computadora (no existe una conexión de red donde se compartan datos en un directorio común).

5.2.7 Consultas de datos de entrada

El sistema permite consultar los siguientes datos generales de uno (según su código de identificación) o todos los siguientes elementos en el diario actualmente abierto, por ejemplo:

- Servicio
- Línea
- Viaje
- Depósito
- Destino

5.2.8 Consultas sobre el diario

El sistema permite realizar consultas sobre el diario actualmente abierto. Las consultas serán las siguientes:

- **Diario de servicio**: al ingresar un código de identificación de un servicio devuelve el diario del mismo dentro de la solución. Este diario se puede visualizar como una tabla y en forma de gráfico. *Requerimientos de la capa lógica*: Pedir a la compañía de transporte urbano cómo lo ven actualmente (para comparar y sacar ideas) y proponer una mejor forma de visualización. Determinar qué datos se van a mostrar y cómo. Estos datos se deben guardar en el archivo de salida.
- **Diario de todos los servicios**: devuelve el diario de todos los servicios de la solución (también se puede consultar por sector). Este diario se puede visualizar como una tabla (son demasiados para verlos en forma de gráfico). *Requerimientos de la capa lógica*: Pedir a la compañía de transporte urbano cómo lo ven actualmente (para comparar y sacar ideas) y proponer una mejor forma de visualización. Determinar qué datos se van a mostrar y cómo. Estos datos se deben guardar en el archivo de salida.
- **Diario de conductor**: al ingresar un código de identificación de un conductor devuelve el diario del mismo dentro de la solución. Este diario se puede visualizar como una tabla y en forma de gráfico. *Requerimientos de la capa lógica*: Pedir a la compañía de transporte urbano cómo lo ven actualmente (para comparar y sacar ideas) y proponer una mejor forma de visualización. Determinar qué datos se van a mostrar y cómo. Estos datos se deben guardar en el archivo de salida.
- **Diario de todos los conductores**: devuelve el diario de todos los conductores de la solución (también se puede consultar por sector). Este diario se puede visualizar como una tabla (son demasiados para verlos en forma de gráfico). *Requerimientos de la capa lógica*: Pedir a la compañía de transporte urbano cómo lo ven actualmente (para comparar y sacar ideas) y proponer una mejor forma de visualización. Determinar qué datos se van a mostrar y cómo. Estos datos se deben guardar en el archivo de salida.
- **Lista de todos los viajes libres**: devuelve la lista de los viajes que no están asignados a ningún servicio dentro de la solución (también se puede consultar por sector). Esta lista se puede visualizar como una tabla en el caso de la consulta (son demasiados para verlos en forma de gráfico). *Requerimientos de la capa lógica*: Pedir a la compañía de transporte urbano cómo lo ven actualmente (para comparar y sacar ideas) y proponer una mejor forma de visualización. Determinar qué datos se van a mostrar y cómo. Estos datos se deben guardar en el archivo de salida.
- **Lista de todos los viajes expresos**: devuelve la lista de los viajes expresos que son parte de algún servicio dentro de la solución (también se puede consultar por sector). Esta lista se puede visualizar como una tabla en el caso de la consulta (son demasiados para verlos en forma de gráfico). *Requerimientos de la capa lógica*: Pedir a la compañía de transporte urbano cómo lo

ven actualmente (para comparar y sacar ideas) y proponer una mejor forma de visualización. Determinar qué datos se van a mostrar y cómo. Estos datos se deben guardar en el archivo de salida.

- **Lista de todos los servicios con viajes expresos**: devuelve la lista de los servicios que tienen viajes expresos dentro de la solución (también se puede consultar por sector). Esta lista se puede visualizar como una tabla en el caso de la consulta (son demasiados para verlos en forma de gráfico). *Requerimientos de la capa lógica*: Pedir a la compañía de transporte urbano cómo lo ven actualmente (para comparar y sacar ideas) y proponer una mejor forma de visualización. Determinar qué datos se van a mostrar y cómo. Estos datos se deben guardar en el archivo de salida.
- **Lista de ómnibus en destino y hora**: al seleccionar un destino y una hora devuelve la lista de todos los ómnibus que se encuentran en el destino y hora especificados dentro de la solución (también se puede consultar por sector). Esta lista se puede visualizar como una tabla en el caso de la consulta (son demasiados para verlos en forma de gráfico). *Requerimientos de la capa lógica*: Pedir a la compañía de transporte urbano cómo lo ven actualmente (para comparar y sacar ideas) y proponer una mejor forma de visualización. Determinar qué datos se van a mostrar y cómo. Estos datos se deben guardar en el archivo de salida.
- **Lista de conductores en descanso en destino y hora**: al seleccionar un destino y una hora devuelve la lista de todos los conductores que se encuentran en descanso en el destino y hora especificados dentro de la solución (también se puede consultar por sector). Esta lista se puede visualizar como una tabla en el caso de la consulta (son demasiados para verlos en forma de gráfico). *Requerimientos de la capa lógica*: Pedir a la compañía de transporte urbano cómo lo ven actualmente (para comparar y sacar ideas) y proponer una mejor forma de visualización. Determinar qué datos se van a mostrar y cómo. Estos datos se deben guardar en el archivo de salida.
- **Lista de conductores en trabajo en hora y hacia destino**: al seleccionar un destino y una hora devuelve la lista de todos los conductores que se encuentran trabajando hacia el destino y en la hora especificados dentro de la solución (también se puede consultar por sector). Esta lista se puede visualizar como una tabla en el caso de la consulta (son demasiados para verlos en forma de gráfico). *Requerimientos de la capa lógica*: Pedir a la compañía de transporte urbano cómo lo ven actualmente (para comparar y sacar ideas) y proponer una mejor forma de visualización. Determinar qué datos se van a mostrar y cómo. Estos datos se deben guardar en el archivo de salida.

5.2.9 Crear servicio

El sistema permite crear un servicio desde cero dentro de un diario abierto. Para ello se debe disponer de las siguientes funcionalidades:

5.2.9.1 Consultas de elementos de entrada para crear un servicio

Para seleccionar los viajes con los que se armará el servicio, el sistema permite consultar:

- **Un servicio**: al ingresar un código de identificación de un servicio devuelve el diario del mismo dentro de la solución. Este diario se debe visualizar de forma que permita su manipulación y modificación (intercambiar parte de su secuencia con alguna parte de la secuencia del viaje que se está armando). Es igual a la consulta de diario de un servicio pero de forma que permita hacer cambios (ver los requerimientos de la capa lógica).
- **Un viaje**: al ingresar el código de identificación de un viaje lo devuelve. Este viaje se puede visualizar de forma que permita su manipulación y modificación (agregarlo al servicio que se está

armando). Es como la consulta de datos de elementos particulares de viaje (ver los requerimientos de la capa lógica).

- **Todos los viajes libres**: devuelve la lista de todos los viajes libres que se encuentran en la solución (también se puede consultar por sector). Esta lista se debe visualizar de forma que permita su manipulación y modificación (agregar uno de los viajes al servicio que se está armando). Es igual a la consulta de viajes libres pero de forma que permita hacer cambios (ver los requerimientos de la capa lógica).
- **Todos los servicios con viajes expresos**: devuelve la lista de todos los servicios con viajes expresos dentro de la solución (también se puede consultar por sector). Esta lista se debe visualizar de forma que permita su manipulación y modificación (intercambiar parte de su secuencia con alguna parte de la secuencia del viaje que se está armando). Es igual a la consulta de servicios con viajes expresos pero de forma que permita hacer cambios (ver los requerimientos de la capa lógica).

5.2.9.2 Consultas de elementos de entrada a partir de elementos de la solución parcial

A medida que se va armando un servicio, dado un viaje se va a necesitar saber qué otros viajes se pueden concatenar con él. Dos viajes son compatibles si es posible realizar uno después del otro utilizando un mismo vehículo. Por ejemplo, dos viajes que se superponen en el tiempo no son compatibles entre si. Dos viajes son compatibles si la hora de llegada del primero y la hora de salida del segundo pueden acomodarse a las reglas del problema (que no se superpongan, que dejen suficiente tiempo para el descanso del personal, o para poder llegar al lugar de origen del segundo, etc.). Con esto en consideración, se pueden realizar las siguientes consultas sobre los viajes que componen un servicio hasta el momento (para así continuar la construcción del nuevo servicio):

5.2.9.2.1 Viajes compatibles simples

Los viajes compatibles simples es cuando dado un viaje queremos saber qué otro viaje le podemos concatenar antes o después. Se pueden obtener los viajes compatibles de la siguiente manera:

- **Caso 1**: Dado un evento devolver los viajes compatibles libres o de otros servicios (según se elija) según la hora de fin para enganchar a continuación.
  - **Ejemplo**: Dado el amarillo que devuelva los otros viajes que lo pueden seguir, con los que lo puedo enganchar (los verdes) o intercambiar los que le siguen (sustituir el naranja) como muestra la Figura 5-1.

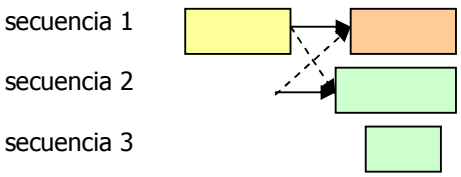


Figura 5-1: Viajes compatibles simples posteriores.

  - **Situaciones** en las que se da:
    - Cuando estoy construyendo un servicio a mano en secuencia.
    - Cuando estoy buscando sustituir una secuencia (naranja) que tiene muchos descansos o viajes expresos para mejorar el costo.

- **Caso 2**: Dado un evento devolver los viajes compatibles libres o de otros servicios (según se elija) según la hora de inicio para enganchar al principio.

- **Ejemplo**: Dado el amarillo que devuelva los otros viajes que lo pueden anteceder, con los que lo puedo enganchar (los verdes) o intercambiar los que le anteceden (sustituir al naranja) como muestra la Figura 5-2.

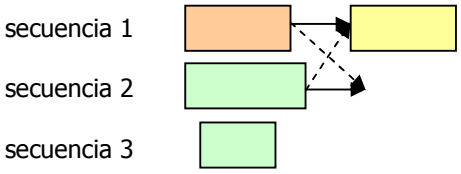


Figura 5-2: Viajes compatibles simples anteriores.

- **Situaciones** en las que se da:
  - Cuando estoy construyendo un servicio a mano en secuencia.
  - Cuando estoy buscando sustituir una secuencia (naranja) que tiene muchos descansos o viajes expresos para mejorar el costo.

- **Caso 3**: Dado un evento devolver los viajes compatibles libres (verdes) y/o de otros servicios (naranjas) según la hora de comienzo y fin para enganchar al principio y al final (Inverso al Caso 3).
  - **Ejemplo**: Dado el naranja que devuelva los otros viajes que lo pueden anteceder o seguir, libres o ya en secuencia como muestra la Figura 5-3.

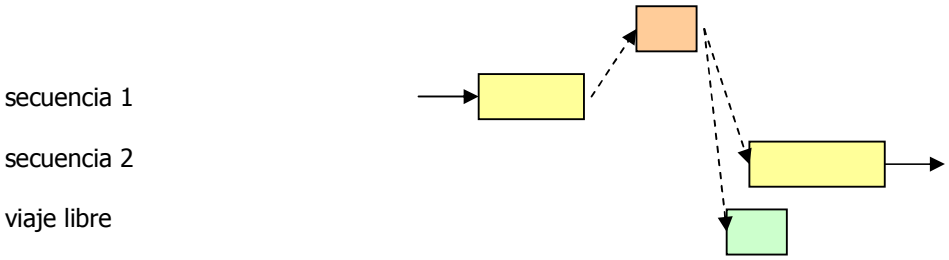


Figura 5-3: Viajes compatibles simples anteriores y posteriores.

- **Situaciones** en las que se da:
  - Cuando estoy construyendo un servicio a mano para enganchar secuencias y libres y tener más opciones.

5.2.9.2.2 Viajes compatibles complejos

Se pueden obtener los viajes compatibles complejos (secuencias de viajes que son compatibles) de la siguiente manera:

- **Caso 4**: Dado dos eventos devolver los viajes compatibles libres (verdes) y/o de otros servicios (naranjas) según la hora de comienzo y fin para sustituirlo (combinación de caso 1 y 2 – compatibilidad anterior y posterior).
    - **Ejemplo**: Dado los viajes en naranja que devuelva los otros viajes libres que los pueden sustituir, con los que lo puedo intercambiar, dando como resultado los verdes. Es decir, los viajes que pueden continuar luego del amarillo primero y pueden anteceder al amarillo último como muestra la Figura 5-4.

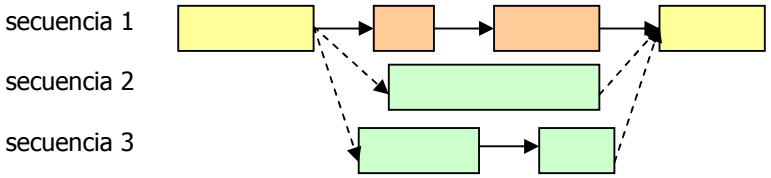


Figura 5-4: Viajes compatibles complejos.

    - **Situaciones** en las que se da:
        - Cuando estoy construyendo un servicio a mano para llenar un hueco.
        - Cuando estoy buscando sustituir una secuencia (naranja) que tiene muchos descansos o viajes expresos para mejorar el costo.

5.2.9.3 Agregar viaje al servicio

Dentro de los resultados de dichas consultas (cualesquiera de ellas) el sistema permite seleccionar un viaje y agregarlo al servicio que se está creando desde cero. Esta acción se puede deshacer mientras se mantenga el diario abierto (ver la sección 5.2.12 Deshacer acción).

5.2.9.4 Eliminar viaje del servicio

Dentro del servicio que se está creando el sistema permite seleccionar un viaje y eliminarlo del servicio. Dicho viaje queda libre para que otro servicio lo pueda incluir. Esta acción se puede deshacer mientras se mantenga el diario abierto (ver la sección 5.2.12 Deshacer acción).

5.2.9.5 Cálculos sobre el nuevo servicio

El sistema permite chequear:

- La factibilidad de dicho servicio (que cumpla con las regulaciones laborales, etc.).
- El costo desglosado de dicho servicio.
- El costo desglosado de la solución total incluyendo dicho servicio para evaluar la mejora.

Para ello, se selecciona el módulo (de costos o factibilidad) que se desea utilizar y se configuran los valores de los parámetros de dicho módulo para ser usados con esta solución (dichos valores son almacenados en el archivo de configuración de la solución). Se puede elegir no chequear ni la factibilidad, ni el costo.

5.2.9.6 Creación y adición del nuevo servicio al diario

Una vez que se creó el servicio se debe poder agregar el nuevo servicio al diario. El sistema debe permitir guardar dicho servicio agregándolo a la solución, actualizando el costo de la misma y actualizando los demás viajes que hayan quedado libres o servicios que se hayan modificado en el proceso de armado. Esta acción se puede deshacer mientras se mantenga el diario abierto (ver requerimiento deshacer).

5.2.10 Modificar servicio

El sistema permite abrir un servicio para modificarlo. Para la modificación el procedimiento es igual al que para crearlo: se pueden realizan las mismas consultas con las que se puede manipular el servicio, hacer cambios, evaluarlo y guardarlo.

5.2.11 Desglosar servicio

Cualquier instancia de los datos que se utilizan en la solución (viaje, línea, etc.) no puede ser eliminado, pero un servicio puede desglosarse, para que los viajes que lo componen queden como viajes libres. El sistema permite seleccionar un servicio para desglosarlo. Si alguno de los viajes antes pertenecía a otro servicio, dicho viaje queda libre, no se vuelve a asignar al servicio al último al que pertenecía antes.

5.2.12 Deshacer acción

Las acciones sobre el diario abierto se pueden deshacer mientras dicho diario se mantenga abierto (al cerrar el diario el historial de deshacer se elimina). Dependiendo de la complejidad de la última tarea realizada, se podrán deshacer como mínimo las últimas 10 acciones (algunas acciones no se podrán deshacer).

5.2.13 Log

El sistema debe mantener un log (o registro de acciones realizadas o errores cometidos) con todo lo que se hace en el sistema y qué usuario lo hizo. Dicho log será válido solo para las acciones que se realicen en el sistema en la computadora específica en la que se trabaje.

5.2.14 Ayuda

El sistema debe proveer ayuda en versión digital ya incorporada a la herramienta (porciones del manual de usuario para evacuar dudas rápidas).

5.3 Requerimientos No Funcionales

A continuación se describen los principales requerimientos no funcionales del sistema.

5.3.1 Portabilidad

El sistema debe ser portable, debe permitir su correcto funcionamiento en distintas plataformas y la buena comunicación con otros lenguajes.

5.3.2 Performance

La carga de datos puede ser costosa por lo que se espera que el sistema tenga tiempos de demora no excesivos.

5.3.3 Flexibilidad

El sistema debe permitir trabajar con distintos algoritmos, distintos formatos de salidas de los mismos, etc. Por lo que se espera una flexibilidad que permita su evolución al agregar más algoritmos y tipos de soluciones.

5.3.4 Robusto

Se espera un buen manejo de errores y que sea resistente a la carga de grandes cantidades de datos, así como el cálculo de costos y almacenaje en archivo. Utilizar hábilmente la habilitación y deshabilitación de opciones para evitar los errores posibles del usuario.

5.3.5 Documentación de usuario

Se espera una buena documentación de usuario tanto para los técnicos como no. Dicha documentación debe ser clara y permitir el trabajo a futuro sobre el sistema.

5.3.6 Mensajes de error

Se espera un buen manejo de los mensajes de error que permitan orientar al usuario en las acciones que debe tomar.

5.3.7 Comentarios

Se espera que el código esté bien comentado para guiar a alguien que pueda querer ampliar el sistema a futuro.

5.3.8 Visualización

El sistema debe permitir el soporte a las decisiones, por lo que la visualización de la información es de suma importancia. Especialmente al armar o comparar servicios se debe proveer de una forma de visualización que sea clara, intuitiva y que permita realizar los cambios necesarios de forma fácil.

5.3.9 Personalización

El sistema debe permitir personalizar lo más posible la forma en la que se ve la información. Por ejemplo seleccionar el color con el que se ven los eventos (en tablas o gráficos).

5.3.10 Calidad

No se requerirá cumplir con un nivel de calidad específico en ninguno de los productos (documentación e implementación). Lo principal es la conformidad de los usuarios. Sin embargo como es un tema que no se había considerado se apreciará cualquier estudio que se haga al respecto.

5.3.11 Herramientas

Se apreciará cualquier idea manejada en herramientas similares. También a la hora de elegir las herramientas de implementación se debe tener en cuenta los aspectos técnicos discutidos en este informe.

5.3.12 Opiniones de los usuarios

Se deberá tener en cuenta la opinión de los usuarios para ver cómo se puede trasladar lo que ellos usan hoy en día a un entorno digital.

5.3.13 Soporte a las decisiones

Existen las siguientes tareas a realizar para la toma de decisiones como muestra el diagrama de la Figura 5-5.

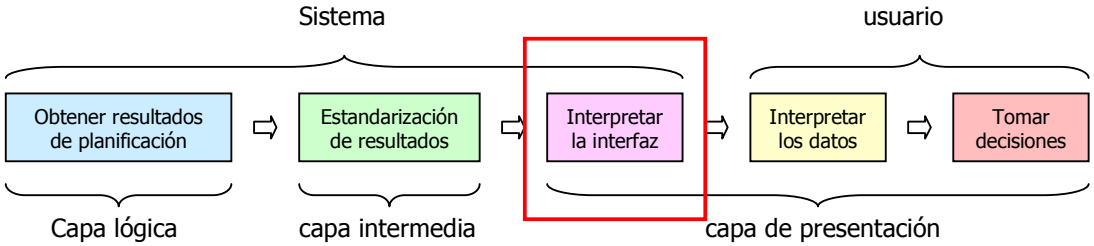


Figura 5-5: Proceso de toma de decisiones.

Divide & conquer

La percepción de la interfaz debe ser natural, para que la interpretación de la misma propiamente no sea un problema, sino que el problema se centre realmente en lo que se debe resolver, que es interpretar los datos y poder tomar las decisiones basado en ellos. Para ello la interfaz debe ser intuitiva, corresponderse con la visión mental que la persona tiene del problema y los datos deben visualizarse de forma clara y su entendimiento debe ser inmediato ("estoy viendo X", un viaje o evento). Lo que le queda al usuario es interpretar los datos ("qué significa X", qué puedo hacer con esta información, cómo puedo usarla para comparar, intercambiar, etc.).

5.4 Restricciones

A continuación se describen las restricciones planteadas para la realización del proyecto.

5.4.1 Lenguaje de programación / Herramientas de implementación

La selección de las herramientas de implementación, así como el lenguaje de programación, deberán tomar en cuenta los aspectos descritos para no tener que luego contar con restricciones en el sistema debido a una mala elección.

Las componentes como módulos de lectura, escritura, cálculo de costos y factibilidad deberán estar implementadas en el lenguaje de la interfaz gráfica para poder ser utilizadas.

5.4.2 Componentes reusados

Como algunas operaciones son implementadas por los algoritmos y se decidió traducirlas al lenguaje en el que se implementará la interfaz, existen ciertas limitantes en cuanto a las ideas o conceptos que se pueden manejar en la interfaz (según la problemática descrita actualmente). En el futuro podrían aparecer datos que no sean fácilmente representables en la interfaz lo que llevaría a un cambio en la interfaz propiamente.

5.4.3 Metodología para el diseño

Debido a que el diseño debe ser flexible, la metodología siempre será irrestricta. Es decir, se de be proveer de toda la flexibilidad posible en el diseño y ampliar las posibilidades al máximo.

5.4.4 Librerías

Se apreciará el uso de librería de tipo open source pero no es una limitante. En caso de utilizar o considerar utilizar herramientas pagas se debe documentar las facilidades que ofrecen y cómo se pueden utilizar en este proyecto.

5.5 Interfaces

A continuación se describen las interfaces a definir.

5.5.1 Interfaces de Usuario

La interfaz de usuario debe permitir todos los requerimientos funcionales especificados. Sus objetos deben permitir la suficiente flexibilidad como para trabajar con diferentes tipos de archivos de salida y representar su información. También debe proveer las facilidades de ampliarla con nuevas componentes.

5.5.2 Interfaces con Software

Se debe implementar una operación dentro de una interfaz que permita interactuar con la capa lógica según los requerimientos de capa lógica especificados para cada requerimiento funcional. Debe proveer la suficiente flexibilidad como para trabajar con distintos algoritmos, soluciones, componentes y formatos.

La Figura 5-6 muestra cómo las tareas de implementación van de lo más general a lo más particular.

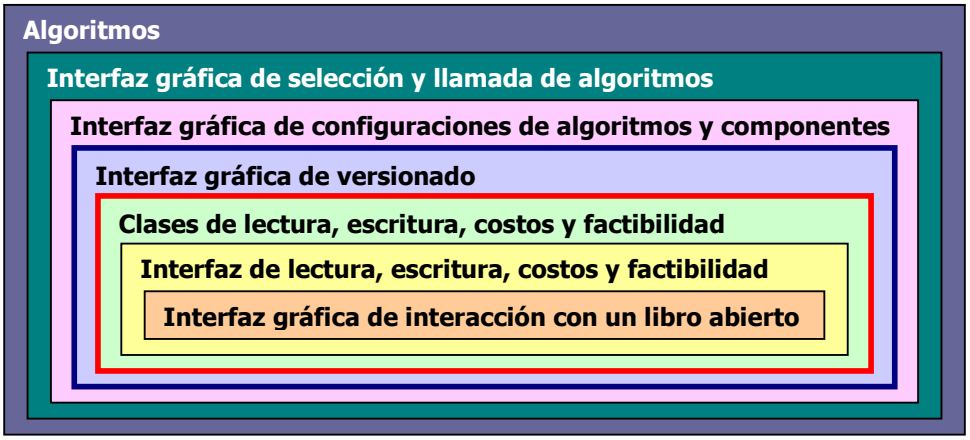


Figura 5-6: Interfaces a definir por el sistema.

5.6 Ciclo de trabajo

A continuación se describe cuál es el **ciclo de trabajo (máximo y mínimo) que los usuarios desean hacer** y cómo los resultados se pueden iterar entre FingLab y otros sistemas.

También se establece qué herramienta se utiliza en cada paso del ciclo de trabajo incluyendo el sistema FingLab (que es el sistema que vamos a construir). Se establece el flujo mínimo y máximo de trabajo posibles al experimentar con algoritmos e interactuar con una solución.

5.6.1 Ciclo de trabajo máximo

Paso en el ciclo de trabajo	Herramienta
1. Implementación del algoritmo y generar ejecutable como unidad funcional	IDE**
2. Implementación en java de módulos de lectura, costos, factibilidad *	Java IDE
3. Implementación en java de la visualización y funcionalidades particulares *	Java IDE
4. Alta del módulo de preferencias con la nueva visualización (si se hizo el punto 3) *	**FingLab**
5. Alta de los módulos de lectura, costos, factibilidad (si se hizo el punto 2) *	**FingLab**
6. Alta del algoritmo en el sistema	**FingLab**
7. Generación de datos de entrada de la primera ejecución	**FingLab /** Sistema externo
8. Ejecución del algoritmo	**FingLab**
9. Abrir, Consultar y modificar una solución obtenida	**FingLab**
10. Exportar la solución a formato de sistemas externos	**FingLab**
11. Abrir, Consultar y modificar una solución con sistemas externos	Sistema externo
12. Abrir la solución modificada por sistemas externos	**FingLab**
13. utilizar la solución como entrada para una nueva ejecución (punto 8) o seguir trabajando en la solución (punto 9)	**FingLab**

* opcional
** IDE del lenguaje en el que se implemente el algoritmo (ya se lenguaje C/C++ u otro)

5.6.2 Ciclo de trabajo mínimo

Paso en el ciclo de trabajo	Herramienta
1. Implementación del algoritmo y generar ejecutable como unidad funcional	IDE
2. Alta del algoritmo en el sistema	**FingLab**
3. Generación de datos de entrada de la primera ejecución	**FingLab /** Sistema externo
4. Ejecución del algoritmo	**FingLab**
5. Abrir, Consultar y modificar una solución obtenida	**FingLab**
6. Exportar la solución a formato de sistemas externos	**FingLab**
7. Abrir, Consultar y modificar una solución con sistemas externos	Sistema externo
8. Abrir la solución modificada por sistemas externos	**FingLab**
9. utilizar la solución como entrada para una nueva ejecución (punto 4) o seguir trabajando en la solución (punto 5)	**FingLab**

FingLab debe interactuar con otros sistemas, es decir, una solución creada en él debe poder pasarse a un sistema externo en el formato que dicho sistema acepte y viceversa. Por lo que es importante tener en cuenta cómo se realizará el mapeo de soluciones de un formato a otro y proveer de formas de hacerlo lo más automáticamente posible. Cómo mapear los datos es parte del requerimiento para exportar e importar soluciones y se explica en detalle en el Anexo: Documentación .

6 Diseño de la interfaz gráfica

En esta sección analizaremos los pasos seguidos para el diseño de la **interfaz gráfica** que construimos.

6.1 Herramientas en el mercado

Antes de comenzar a planificar nuestra interfaz realizamos un estudio de aquellas herramientas ya existentes en el mercado para tener una idea primaria del "look and feel" de sus interfaces, analizar sus fortalezas y debilidades. Existen múltiples herramientas en el mercado que se dedican al área de planificación del transporte las cuales son analizadas en el Anexo: Planificación del transporte. Lo que podemos extraer de ellas es:

- Son siempre usadas bajo licencia. No existen herramientas gratuitas, ni versiones demo para evaluarlas adecuadamente. Su precio es generalmente alto (dirigido a empresas, no usuarios particulares o investigadores).
- Abarcan varios problemas a la vez (ordenamiento de vehículos, ruteo, etc.). Por lo que proveen de una visualización apropiada para cada tipo.
- Son herramientas independientes, no fomentan la interacción con otros sistemas. Crean una relación de dependencia del usuario con el sistema, en vez de dar libertad al usuario a agregar sus propias visualizaciones.
- Los algoritmos en general ya vienen incorporados a la herramienta, no siempre se permite al usuario agregar sus propias implementaciones. Estas herramientas están dirigidas a los usuarios que les interesa los resultados, no la experimentación con algoritmos.
- Debido a la protección de código es difícil personalizarlas para casos particulares. Proveen de interfaces de software que permiten agregar alguna funcionalidad en algunos casos, pero no son muy flexibles.

6.2 Trabajo con procesos

Como estamos trabajando con ejecuciones que se van a lanzar desde nuestra aplicación, existen ciertos aspectos a tomar en cuenta al trabajar con los procesos de las mismas. Estas observaciones se desprenden como resultado del estudio de herramientas en el mercado de planificación de tareas que se encuentra en el en el Anexo: Planificación del transporte.

No se puede establecer con certeza cuánto tiempo una ejecución va a demorar, ya que por su naturaleza, los algoritmos dependen de los datos de entrada y los parámetros utilizados para su ejecución. Por ello, es importante dar al sistema la capacidad de poder **planificar las ejecuciones**, de forma que el usuario pueda establecer cuándo desea ejecutar un cierto algoritmo, con qué datos, dejar el computador encendido con el sistema abierto y que éste se encargue de realizar dicha ejecución. De esta manera, el usuario no tiene por qué estar presente al momento en que comienza la ejecución y poder dejar al sistema encargado de esta tarea. Otro aspecto relacionado con este punto es que el sistema pueda quedar residente, es decir, que no sea necesario tener la ventana del sistema maximizada o incluso minimizada en la barra de tareas del sistema operativo, sino que al minimizarlo quede residente en forma de ícono en la bandeja del sistema, y que se pueda maximizar nuevamente en cualquier momento si el usuario lo desea haciendo doble click en dicho ícono (como se muestra en la Figura 6-1).

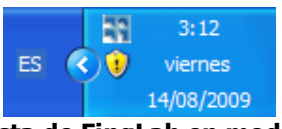


Figura 6-1: Vista de FingLab en modo residente.

El usuario debe ser capaz de poder saber el estado de cada proceso en todo momento y tener control del mismo (por ejemplo, ver los procesos planificados para su ejecución, si desea cancelarlo antes de que comience su ejecución, o pausarlo o matarlo durante su ejecución). Al darle la independencia al sistema de que lance las ejecuciones en una fecha determinada que el usuario le indique, sumado al hecho de no saber exactamente cuándo va a terminar una ejecución, se debe proveer de mecanismos de comunicación con el usuario cuando el proceso termine su ejecución. Para ello, existen tres formas básicas de avisar al usuario que una ejecución terminó:

1. *Alerta visual*: se despliega un cartel en pantalla que indica que un proceso terminó. Este tipo de alerta es ideal para aquellos usuarios que una vez que planificaron sus ejecuciones, puedan trabajar con otras aplicaciones y por medio de esta alerta percatarse de la culminación de un proceso. Incluso se puede personalizar según si el proceso finalizó de forma exitosa o no.
2. *Alerta sonora*: se reproduce un sonido que indica que un proceso terminó. Este tipo de alerta es ideal para aquellos usuarios que una vez que planificaron sus ejecuciones, puedan trabajar con otras aplicaciones o que incluso dejar el computador de lado sin poder ver la pantalla, pero pueden escuchar esta alerta y percatarse de la culminación de un proceso. También se puede personalizar según si el proceso finalizó de forma exitosa o no.
3. *Alerta de e-mail*: se envía un e-mail a una dirección predeterminada de que la ejecución culminó. Se puede incluso enviar los resultados de la ejecución (salida). Este tipo de alerta es ideal para aquellos usuarios que una vez que planificaron sus ejecuciones, puedan dejar el computador o incluso no estar físicamente cerca del mismo y por medio de esta alerta percatarse (de forma remota) de la culminación de un proceso.

Todos estos puntos se tomaron en cuenta y fueron implementados para proveer de un sistema más completo, que facilite y mejore la experiencia del usuario.

6.3 Soporte de versiones de ejecuciones

Para resolver el problema de soporte de versiones se ideó la siguiente jerarquía:

- Año
  - Estación del año
    - Tipo de día
      - Experimento
        - Ejecuciones
          - Ejecuciones derivadas

Cada ejecución se clasifica según el año, estación del año, tipo de día y experimento al que pertenece. Un experimento es una forma de agrupar las ejecuciones (por ejemplo, las que poseen los mismos datos de entrada pero procesadas por diferentes algoritmos).

Una ejecución está compuesta por: **Entrada + salida + configuración + ejecuciones derivadas**.

La entrada es parte de la solución y depende del algoritmo que se aplique, por lo tanto se debe guardar dentro de la solución cada vez que se haga una nueva solución.

La salida es generada por el algoritmo ejecutado.

El archivo de configuración es creado automáticamente por FingLab y guarda toda la información de la solución. El nombre del archivo de configuración contendrá el nombre del algoritmo que se utilizó.

La nomenclatura de los directorios para las ejecuciones será con numeración por niveles automática. Por ejemplo, la primera ejecución se llamará "Ejecucion.1", la primera ejecución derivada de ella se llamará "Ejecucion.1.1", la segunda ejecución derivada se llamará "Ejecucion.1.2" y así sucesivamente. Para no alterar

la generación de los nombres de las ejecuciones no se permite eliminar ninguna ejecución una vez planificada (sí se puede cancelar antes de que comience el proceso de ejecución).

Según los datos ingresados al planificar la ejecución, en la estructura de versiones se guardará con la una jerarquía como muestra la Figura 6-2.

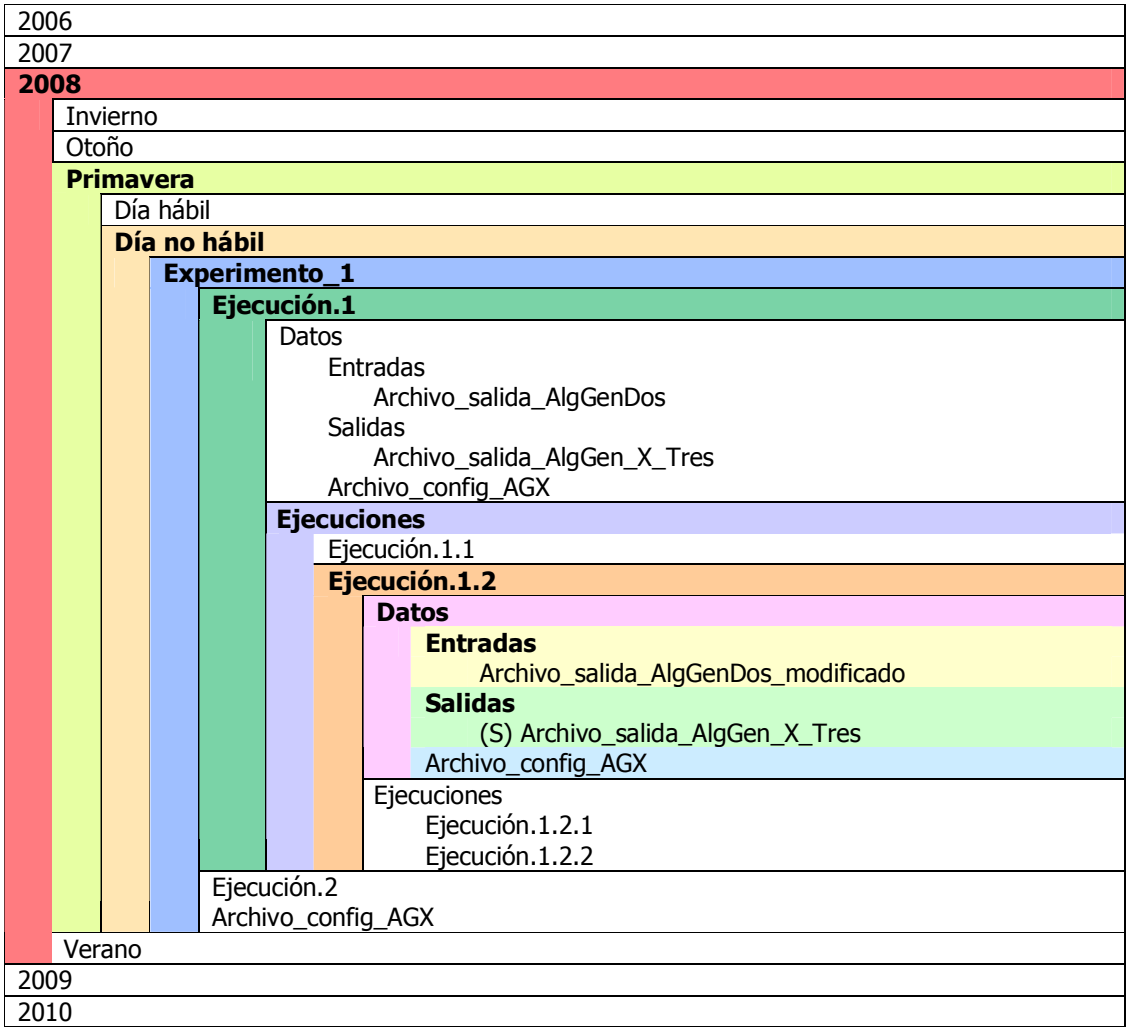


Figura 6-2: Ejemplo de jerarquía de versiones de ejecuciones en FingLab.

6.4 Estados de las ejecuciones

Las ejecuciones en FingLab pueden tener los siguientes estados:

- ***Planificada***: La ejecución está planificada para su lanzamiento en una determinada fecha y hora. El directorio de la ejecución fue creado y los archivos de entrada fueron copiados a dicho directorio para que al comenzar la ejecución todos los recursos necesarios estén disponibles.
- ***Cancelada***: La ejecución fue cancelada por el usuario antes de comenzar (cuando todavía estaba planificada). No se lanzará el proceso que realiza la ejecución y el directorio de la ejecución se mantendrá en el sistema para evitar problemas de numeración entre las ejecuciones.
- ***En ejecución***: El proceso de ejecución ya comenzó y está en desarrollo. Cuando culmine se mostrarán los mensajes correspondientes según el estado en el que haya finalizado la ejecución.

- ***Pausada***: El proceso está en ejecución y el usuario lo pone en pausa. El proceso de ejecución se puede reanudar posteriormente si el usuario lo desea.
- ***Interrumpida***: El proceso está en ejecución y el usuario la interrumpe (mata el proceso). El proceso de ejecución culmina y no se puede reanudar posteriormente.
- ***Finalizada correctamente***: El proceso de ejecución culminó con éxito y se despliegan los mensajes correspondientes.
- ***Finalizada con advertencias***: El proceso de ejecución culminó con advertencias (no significa que haya ocurrido un error) y se despliegan las advertencias correspondientes.
- ***Finalizada con errores***: El proceso de ejecución culminó con error o por alguna razón fue abortado. Se guarda un registro con todos los errores que hayan salido en pantalla durante su ejecución.

6.5 Estados del sistema

En base al ciclo de trabajo definido y los requerimientos, definimos los siguientes estados en los que el sistema puede estar:

1. ***Estado Inicial***: es el estado en el que se encuentra la aplicación al abrirla o cuando no hay seleccionada ninguna ejecución para consultar alguna solución. Si se selecciona una ejecución se pasa al estado "Ejecución seleccionada".
2. ***Estado Ejecución seleccionada***: es el estado en el que hay una ejecución seleccionada y se habilitan las funcionalidades de consulta sobre un diario de dicha ejecución. En este estado no hay ninguna solución cargada en memoria. Si se selecciona abrir un diario se pasa al estado "Diario abierto". Si se selecciona un elemento en el árbol de planificación que no es una ejecución se vuelve al estado inicial.
3. ***Estado Diario abierto***: es el estado en el que hay un diario abierto en memoria y se habilitan las funcionalidades para consultar datos solamente de dicho diario. Se carga el módulo de lectura-escritura de datos particular para dicho diario (si lo tiene, sino se usa el módulo de lectura-escritura por defecto). Si se selecciona cerrar el diario se vuelve al estado "Ejecución seleccionada". Si se selecciona armar un servicio manualmente en el diario actualmente abierto se pasa al estado "Armado manual de servicio".
4. ***Estado Armado manual de servicio***: es el estado en el cual se habilitan las funcionalidades para realizar modificaciones manuales dentro de la solución abierta y cargada en memoria. Si se selecciona cerrar o terminar el armado del servicio se vuelve al estado "Diario abierto".

En la Figura 6-3 se muestran los posibles estados en los que puede estar FingLab y las transiciones posibles de un estado a otro.

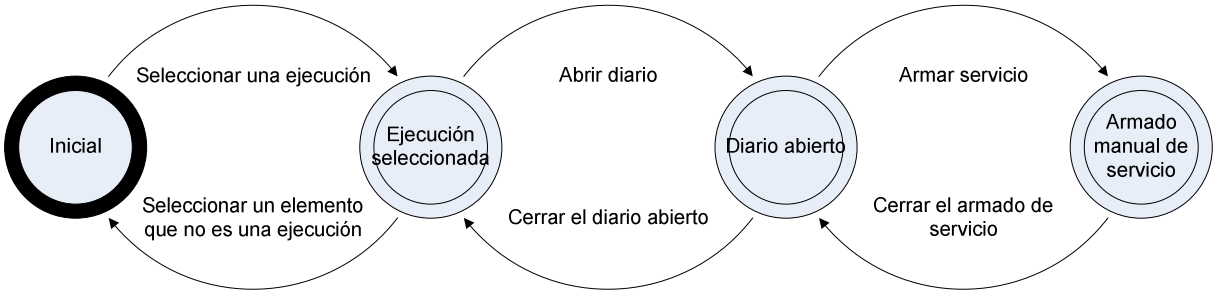


Figura 6-3: Estados y transiciones posibles de FingLab.

6.6 Visualización de datos

Realizamos un estudio de las visualizaciones gráficas posibles para los datos de forma que el usuario pueda manejarlos y analizarlos para la realización de las tareas de ordenamiento de la flota de vehículos y tripulación. Para realizar este estudio nos basamos en la siguiente información:

1. Requerimientos del problema.
2. Entrevistas con usuarios, tanto desarrolladores, como planificadores.
3. Estudio en el campo (análisis de herramientas en el mercado enfocadas al área de planificación de transporte).
4. Consultas con expertos en el área de HCI.

La meta de este estudio es analizar varias posibles representaciones visuales según las dimensiones del problema y encontrar aquella(s) que cumplan con los requerimientos establecidos, así como ofrecer al usuario el apoyo necesario para visualizar información, así como dar soporte a la toma de decisiones.

Una "dimensión del problema" es una posible métrica, variable o elemento que se desea representar visualmente o que se puede deducir por medio de la interpretación de la visualización de datos en forma gráfica. Según los requerimientos del problema planteado, las dimensiones que podemos identificar son:

- ***Tiempo***: al tratarse de ordenamiento o planificación la dimensión del tiempo es vital para saber qué está ocurriendo en cada momento con los recursos que se están manejando.
- ***Servicios***: los conjuntos de viajes que hemos construido o que estamos construyendo manualmente.
- ***Turnos***: la separación de los servicios según las normas de trabajo.
- ***Eventos***: viajes o períodos de tiempo en los que los recursos están cumpliendo una función determinada:
  - Descansos de la persona.
  - Tiempos de descanso.
  - Viajes comunes / Personas trabajando.
  - Viajes expresos.
  - Viajes de chofer como pasajero.
  - Disponibilidad por un período.
  - Disponibilidad por el día.
- ***Lugares***: los lugares donde se realizan los eventos.
- ***Ómnibus***: los medios en los cuales se realizan los servicios.
- ***Viajes compatibles***: poder comparar viajes para saber si se pueden concatenar para formar una secuencia.

En base a las dimensiones identificadas anteriormente existen otras dimensiones que se pueden deducir según lo que se represente visualmente:

- *Persona-tiempo*: lo que hace una persona en el correr del tiempo.
- *Persona/tiempos de descanso-tiempo*: lo que hace una persona o los tiempos de descanso a lo largo del tiempo.
- *Persona-lugar*: cómo una persona se mueve de un lugar a otro.
- *Ómnibus-tiempo*: cómo un ómnibus se mueve a lo largo del tiempo.
- *Ómnibus-lugar*: cómo se mueve un ómnibus de un lugar a otro.
- *Persona/ómnibus-tiempo*: cómo se mueve una persona o un ómnibus a lo largo del tiempo.

- *Persona/ómnibus-lugar*: cómo se mueve una persona o un ómnibus de un lugar a otro.
- *Lugar/ómnibus-tiempo*: si un ómnibus está en un lugar en un momento determinado (esto define si un viaje es compatible).

Lugar y tiempo son dos medidas que no tienen sentido por sí solas, es su resultado lo que define una persona, un viaje, un ómnibus, un tiempo de descanso, etc.

Los tiempos de descanso y los viajes compatibles comparten características como que un ómnibus no puede estar en un lugar si no tiene tiempo de llegar y una persona no puede estar en un lugar si no está allí, o no puede llegar allí a tiempo o si no descansó lo suficiente.

Un ómnibus, una persona y un servicio son lo mismo, para la realidad de nuestro problema, todos los viajes de un servicio son realizados por un mismo ómnibus y la(s) misma(s) persona(s), con lo cual podemos reducir dimensiones.

Luego del estudio que se detalla en el Anexo: Documentación , llegamos a la conclusión que un Diagrama de Gantt parece ser la forma de visualización más apropiada, que encontramos por el momento. El paso del tiempo en el eje x es más natural e intuitivo. Al mostrar las personas en el eje y se puede ver la duración de los eventos y se tiene la sensación de movimiento de espacio en relación al tiempo sin necesariamente graficar los lugares. Es fácil de comparar los tiempos de descanso compatibles para intercambiar personal. Esta vista también funciona si en vez de conductores consideramos servicios u ómnibus, por lo que constituye una forma de visualización muy viable para nuestro caso.

Algunas notas de importancia respecto a este tipo de gráfico son:

- Debido a la gran cantidad de servicios a graficar, no todos se van a poder visualizar a la vez, por lo que se debe proveer de formas para filtrar los resultados que se pueden ver simultáneamente.
- La visualización debe proveer de buen scroll (barras de desplazamiento) tanto vertical como horizontal para poder ver por completo todos los servicios planificados.
- La visualización también debe tener buenas herramientas de zoom in y zoom out para ver detalles cuando nos acercamos (zoom in) y omitirlos cuando nos alejamos (zoom out).
- Al imprimir el gráfico debe entrar en una hoja A4, por lo que se deben hacer pruebas de configuración de la impresión.
- El diagrama de Gantt tiene como limitación que si bien se puede ver los servicios, no permite ver el uso de los recursos. Es decir, si bien se pueden graficar los servicios por ómnibus, no se puede visualizar qué chofer está asignado a cada uno.
- Es importante que el gráfico tenga funcionalidades de acercamiento (zoom in) para ver detalles y alejamiento (zoom out) para tener una visión global de los resultados.

La visualización más efectiva entonces para el problema de ordenamiento, encontrada hasta el momento, es un **diagrama de Gantt**. Permite visualizar o intuir todas las dimensiones del problema de la forma más clara y, aún con ciertas limitaciones, representa todos los elementos que ayudan a seguir la línea de pensamiento del usuario. Por lo tanto, esta visualización será la que se implementó en el proyecto.

6.7 Prototipos

Crear **prototipos** en las fases tempranas del proyecto es de gran importancia, ya que al principio de un proyecto es cuando se toman ciertas decisiones (de diseño, de herramientas a utilizar, etc.) y el prototipado ayuda a examinar si dichas decisiones son viables. Especialmente cuando se trata de crear una interfaz gráfica se debe crear un prototipo que el usuario pueda inmediatamente examinar para indicarle al programador si va en la dirección correcta o si hay algo que definitivamente no representa lo que el cliente desea [6].

Por medio de estos prototipos podemos determinar qué no se está contemplando, qué es de agrado al usuario y definitivamente qué es lo que se debería cambiar antes de comenzar el proceso de desarrollo. Los prototipos presentados en este informe se utilizaron como una primera forma de atacar el problema y como un medio para la recolección de requerimientos.

Para este proyecto se desarrollaron 4 prototipos (que se detallan en el Anexo: Documentación):

- *Prototipo en Visual Basic .NET*: permite dar una visión general de la interfaz tal como fue diseñada en papel.
- *Prototipo Java 1*: permite ver cómo se pueden ejecutar programas implementados en Lenguaje C, desde código en Lenguaje Java, planificar, pausar o matar procesos y captar sus salidas en pantalla.
- *Prototipo Java 2*: permite ver cómo se pueden cargar clases implementadas en Java en tiempo de ejecución (llamada en inglés "reflection"). De esta manera, si nos definimos una interfaz que deben cumplir los módulos de lectura de soluciones, cálculo de costos y factibilidad, un usuario programador puede implementar sus propios módulos aparte, darlos de alta en el sistema y sustituirlos en tiempo de ejecución. Esto se explica con más detalle en la sección 6.7.1 Intercambio de módulos.
- *Prototipo HTML*: permite ver algunas posibles interacciones con el gráfico de Gantt (sobre todo para armar servicios manualmente). Si bien se consideró la posibilidad de realizar una interfaz completamente Web (en lenguaje HTML), esto se descartó debido a los problemas de compatibilidad entre navegadores Web y las diferentes versiones de los mismos (también llamado "crossbrowsing"), lo cual dificulta el mantenimiento de la aplicación y que la visualización se vea entorpecida por elementos que se despliegan diferente en cada tipo de navegador. Esto se explica con más detalle en la sección 6.7.2 Manipulación.

6.7.1 Intercambio de módulos

Del prototipo en Java Nº 2, mediante la utilización de la propiedad "reflection" vemos que podemos cargar módulo en tiempo de ejecución. Esto nos es de mucha utilidad para los cambios que ocurren cuando un usuario experimenta con algoritmos: cambios en los formatos de lectura-escritura, cambios en las fórmulas de cálculo de costos, cambios en las leyes laborales que afectan la factibilidad de un servicio, etc.

De esta manera, si nos definimos una interfaz que deben cumplir los módulos de lectura-escritura de soluciones, cálculo de costos y factibilidad, un usuario programador puede implementar sus propios módulos aparte, darlos de alta en el sistema y sustituirlos en tiempo de ejecución. Esto nos permite cambiar la forma de guardar datos y hacer cálculos, sin tener que tocar el código base de FingLab. Esta capacidad es muy útil, ya que la forma de realizar cálculos puede variar y cada vez que esto ocurra, no deseamos que el código base esté continuamente siendo modificado, lo cual puede causar la introducción de errores en la aplicación. Así también, cada vez que cambia el formato de las salidas de los algoritmos, no se tiene que modificar código base, sino que se hace un módulo que lea ese nuevo formato, se da de alta en el sistema y se lo asigna a la hora de leer resultados en tiempo de ejecución.

6.7.1.1 Intercambio de módulo lectura-escritura

Para los usuarios que experimentan con algoritmos, los formatos de salidas pueden cambiar constantemente, por lo que FingLab considera una forma de lectura flexible y configurable que permite seguir trabajando con estos formatos aún cuando ocurran ciertos cambios en los mismos. FingLab también permite la incorporación de módulos que implementen la lectura y escritura en un cierto formato cumpliendo con una interfaz de software definida. Esto se explica con más detalle en el Anexo: Documentación .

El sistema permite abrir un archivo de salida con la solución de los algoritmos para poder trabajar con ella, visualizar información, realizarle modificaciones y guardar esas modificaciones (las modificaciones posibles se encuentran en las secciones 5.2.9 Crear servicio y 5.2.10 Modificar servicio). Esto implica la carga de los datos de una solución (levantar el archivo de salida de la solución).

Para ello, se debe poder levantar los datos del archivo de salida y crear los elementos con los que trabaja la interfaz y que permiten realizar las modificaciones y la visualización de datos. Como los archivos de salida pueden ser de un tipo determinado (txt, dbf, xml, etc.) y tener un formato que depende del algoritmo aplicado, para levantar dicho archivo, el sistema utilizará el módulo de lectura-escritura provisto por la configuración del algoritmo que creó dicha solución. El sistema provee de una clase interfaz para crear todos los objetos necesarios para utilizar la interfaz gráfica y el módulo debe implementar dicha clase interfaz. El sistema provee una clase de lectura-escritura por defecto que lee todos los datos de un archivo de texto plano según la estructura definida en la sección 4 Descripción del proceso de ejecución de los algoritmos de ordenamiento de vehículos y tripulación.

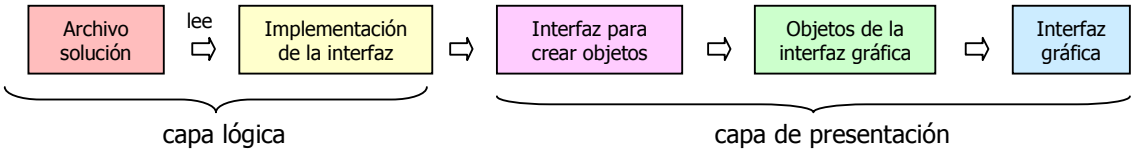


Figura 6-4: Proceso de lectura de una solución.

Si la interfaz tiene ciertos objetos (fijos, que no se pueden cambiar) y la salida al cargarse tiene que adaptarse a esos objetos, se puede perder información valiosa (ver la sección 5.2.7). Se permite tener un solo diario abierto a la vez, ya que los datos son muchos y su carga compromete la performance del sistema. (Puede quedar como trabajo futuro levantar varios a la vez).

Ventajas: de esta manera se tiene una plataforma extensible a cualquier tipo de archivo de salida y cualquier formato (como es el usuario el que se encarga de leer los datos, los datos pueden estar en un archivo de texto plano, o xml, o incluso puede ser una base de datos). Evita que los archivos tengan que hacerse siempre de un determinado formato y evita que la interfaz se deba adaptar según los datos. Quien implemente los algoritmos lo único que tiene que hacer es implementar la interfaz que crea los objetos con los que trabaja la interfaz gráfica y él mismo decide cómo usarlos. Si hay más datos en el archivo de salida que los que los objetos de la interfaz gráfica toleran, entonces cada objeto de la interfaz gráfica provee de un campo que permite mostrar todos esos datos (por ejemplo una cadena de caracteres), para que no se pierda información, pero dicha información no es vital para el uso de la interfaz gráfica. Si hay más datos en un objeto de la interfaz gráfica que en el archivo de salida, ciertos objetos no se podrán crear, o serán incompletos y la interfaz gráfica limitará las opciones de lo que se puede hacer o visualizar. La interacción de la interfaz gráfica con los algoritmos es mínima, es solo la invocación para que los ejecute al crear un diario, luego no necesita de ninguna otra comunicación para leer, modificar o guardar la solución, lo cual disminuye las dependencias de compatibilidad entre ambos.

Restricción: la implementación de la clase de lectura debe ser hecha en el lenguaje de programación del sistema.

El sistema permite volver a generar el archivo de salida en el formato estipulado, con todas las modificaciones que se hayan realizado mientras se tuvo el diario abierto. Esto implica una persistencia de los datos de una solución (guardar el archivo de salida de la solución). Las soluciones creadas, modificadas o

eliminadas en el sistema en una computadora solo estarán disponibles para los usuarios que inicien sesión en esa computadora (no existe una conexión de red donde se compartan datos en un directorio común). Si el archivo contiene varios diarios, se puede ingresar el código del diario que se desea guardar y la implementación se encargará de hacerlo si se desea que funcione de esa manera.

Para ello, se debe poder guardar los datos en el archivo de salida según fueron modificados los elementos con los que trabaja la interfaz. Como los archivos de salida pueden ser de un tipo determinado (txt, dbf, xml, etc.) y tener un formato que depende del algoritmo aplicado, para guardar dicho archivo, el sistema utilizará el módulo de lectura-escritura provisto por la configuración del algoritmo que creó dicha solución. El sistema provee de una clase interfaz para devolver todos los objetos de la interfaz gráfica y el módulo de lectura-escritura debe implementar dicha clase interfaz. El sistema provee una clase de lectura-escritura por defecto que guarda todos los datos en formato de tipo texto plano según la estructura definida en la sección 4 Descripción del proceso de ejecución de los algoritmos de ordenamiento de vehículos y tripulación.

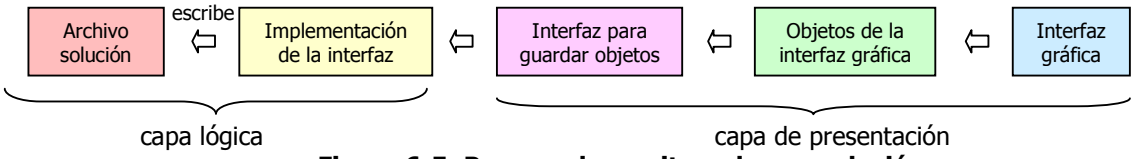


Figura 6-5: Proceso de escritura de una solución.

Ventajas: de esta manera se tiene una plataforma extensible a cualquier tipo de archivo de salida y cualquier formato (como es el usuario el que se encarga de leer los datos, los datos pueden estar en un archivo de texto plano, o xml, o incluso puede ser una base de datos). Evita que los archivos tengan que hacerse siempre de un determinado formato y evita que la interfaz se deba adaptar según los datos. Quien implemente los algoritmos lo único que tiene que hacer es implementar la interfaz que guarda los objetos con los que trabaja la interfaz gráfica y él mismo decide cómo guardarlos. La interacción de la interfaz gráfica con los algoritmos es mínima, es solo la invocación para que los ejecute al crear un diario, luego no necesita de ninguna otra comunicación para leer, modificar o guardar la solución, lo cual disminuye las dependencias de compatibilidad entre ambos.

Restricción: la implementación de la clase de escritura debe ser hecha en el lenguaje de programación del sistema.

6.7.1.2 Intercambio de módulos de cálculo de costos y factibilidad

Los algoritmos tienen en cuenta dos factores a la hora de evaluar las soluciones:

1. **Costo**: el costo de la solución y la forma de calcularlo puede cambiar con el tiempo, por lo que si se desea calcularlo se debe poder configurar el valor de los coeficientes utilizados y hasta cambiar la función de cálculo completa por una diferente.
2. **Factibilidad**: se refiere a si una secuencia de viajes es viable, ya que para un servicio, se deben respetar ciertas leyes laborales que restringen el que cualquier secuencia (aún con un costo más óptimo) sea viable. Las leyes laborales pueden ser por ejemplo: el conductor debe tener un cierto número de descansos a lo largo del servicio, de una determinada duración o distribuidos de una determinada manera. Estas leyes también pueden cambiar con el tiempo.

FingLab toma en consideración estas dos formas de evaluación de soluciones, ya que al realizar cambios manualmente, como por ejemplo, crear servicios, se debe evaluar si el nuevo servicio es factible, su costo, y cómo su inserción afecta el costo total de un diario. También provee de las interfaces necesarias para realizar los cambios de coeficientes, así como de fórmulas que puedan llegar a necesitarse mediante el intercambio de módulos que las implementen. Esto se explica con más detalle en el Anexo: Documentación .

6.7.2 Manipulación en la interfaz gráfica

Mediante el prototipo HTML pusimos a prueba diferentes estrategias de manipulación de elementos, siguiendo patrones de diseño que estudiamos en el estudio del estado del arte de las interfaces. La meta es proveer de una forma de manipulación fácil, sobre todo asistiendo y ayudando a la creación de los nuevos servicios. Aquí se presentan algunas notas de lo que es deseable a la hora de manipular los datos, sobre todo para la creación de los nuevos servicios (producto del prototipo Web realizado que se muestra en la Figura 6-6).

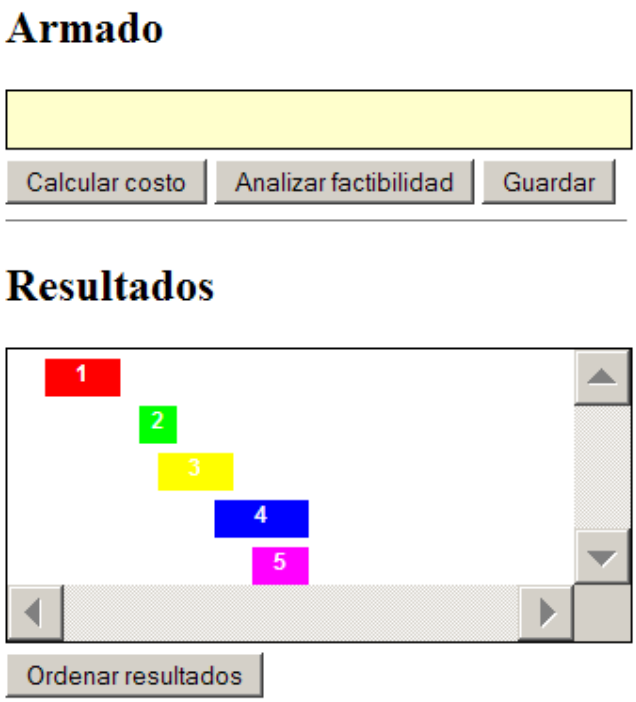


Figura 6-6: Prototipo con la interfaz de armado de un nuevo servicio.

- Todo se mostrará de forma que se permita apreciar la continuidad en el tiempo de los eventos. Debe haber una clara representación de la ubicación en el tiempo de los eventos.
- Proveer de dos zonas:
  - Una en la que se va armando los viajes.
  - Otra abajo (con scroll) en la que se muestran los resultados que se van pidiendo (otros servicios, viajes libres, viajes compatibles, etc.).
- La zona de armado debe ser siempre visible en la pantalla.
- La zona de resultados contiene los viajes que se obtienen producto de consultas al diario. Estos viajes se utilizarán para ser incorporados al nuevo servicio. Esta zona debe tener Scholl (barras de desplazamiento) porque pueden ser muchos.
- Se debe mostrar los viajes resultado de una consulta ordenados por compatibilidad. Los viajes se ordenan por compatibilidad según 1º la calidad del viaje, 2º lo cerca que está en tiempo y 3º lo cerca que está en distancia.

Para más información sobre las observaciones desprendidas de este prototipo, vea el anexo de Prototipos.

6.8 Lenguaje Java

Si bien otros lenguajes fueron considerados, como por ejemplo Visual Basic .NET (que fue utilizado para construir uno de los prototipos del proyecto), por experiencias previas y por la necesidad de portabilidad elegimos Java. Java ofrece muchas ventajas en comparación a otros lenguajes [5, 7] y a continuación se enumeran las razones por las que elegimos este lenguaje:

- Es portable, podemos ejecutarlo en otros sistemas operativos que no sean necesariamente en el que se desarrolló la aplicación.
- Se pueden crear aplicaciones que se pueden ejecutar dentro de un explorador Web.
- Podemos ejecutar desde java cualquier programa en otro lenguaje (como se detalla en la sección Prototipos).
- Permite la sustitución de módulos en tiempo de ejecución (como se detalla en la sección Prototipos).

6.9 Diseño planificado

Luego de tomar en cuenta los requerimientos definidos, el estudio de las interfaces y el análisis de las visualizaciones posibles, llegamos a definir que la ventana de **FingLab** debe tener 8 regiones básicas como muestra la Figura 6-7.

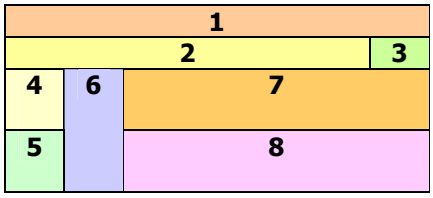


Figura 6-7: Diseño planificado de FingLab.

1. *Barra de menú*: contiene los ítems de todas las funcionalidades disponibles al usuario con que se ingresó. Todas las opciones son accesibles por medio de hot keys.
2. *Barra de botones*: contiene los ítems más importantes de las funcionalidades disponibles al usuario con que se ingresó.
3. *Banner de consejos útiles*: despliega consejos útiles al usuario durante la sesión de trabajo. Al colocar el Mouse sobre el banner se despliega una explicación más extensa del consejo.
4. *Panel de Planificaciones / Configuración*: si el usuario con que se ingresó es planificador Standard se despliega solo el panel de planificaciones, pero si es desarrollador administrador se despliegan dos pestañas:
    a. *Planificaciones*: contiene todas las ejecuciones de algoritmos categorizadas y versionadas según se lanzaron en el espacio de trabajo.
    b. *Configuración*: contiene todos los algoritmos, módulos de costos y factibilidad que se dieron de alta en el espacio de trabajo.
5. *Panel de Ejecuciones*: contiene botones con las ejecuciones cuya ejecución empezó o terminó durante la sesión de trabajo y las planificadas que aún no se lanzaron. Al presionar el botón de una ejecución se despliega el formulario con todos los datos de la ejecución correspondiente.
6. *Panel de Consultas*: contiene el árbol con todas las consultas realizadas al diario de viajes.
7. *Panel de Visualización en formato Tabla*: contiene la visualización en formato de tabla de la consulta seleccionada en el panel de consultas.
8. *Panel de Visualización en formato Gráfico*: contiene la visualización en formato gráfico de la consulta seleccionada en el panel de consultas.

Cuando se hace click en el panel de planificaciones / configuración, o en el panel de ejecuciones, o en el panel de consultas, o en el panel de visualización en formato tabla, o en el panel de visualización en formato gráfico, dicho panel se marca con un borde azul para indicar al usuario la región activa actual del sistema (así, mantenemos el foco de atención del usuario como vimos en el estudio del estado del arte de las interfaces). También se proveerán de opciones de botón derecho del Mouse, así como acceso a funcionalidades por medio de teclado (hot keys).

7 Descripción de la arquitectura

7.1 Modelo de casos de uso relevantes a la arquitectura

Los casos relevantes a la arquitectura y que definen las decisiones de diseño realizadas se muestran en la Figura 7-1 (diagramas creados con Microsoft Visio). Estos casos de uso aparecen vinculados al usuario de tipo desarrollador, pero para el usuario de tipo planificador es un subconjunto de los mismos, según la accesibilidad a las funcionalidades definida en los requerimientos (ver Tabla 5-1).

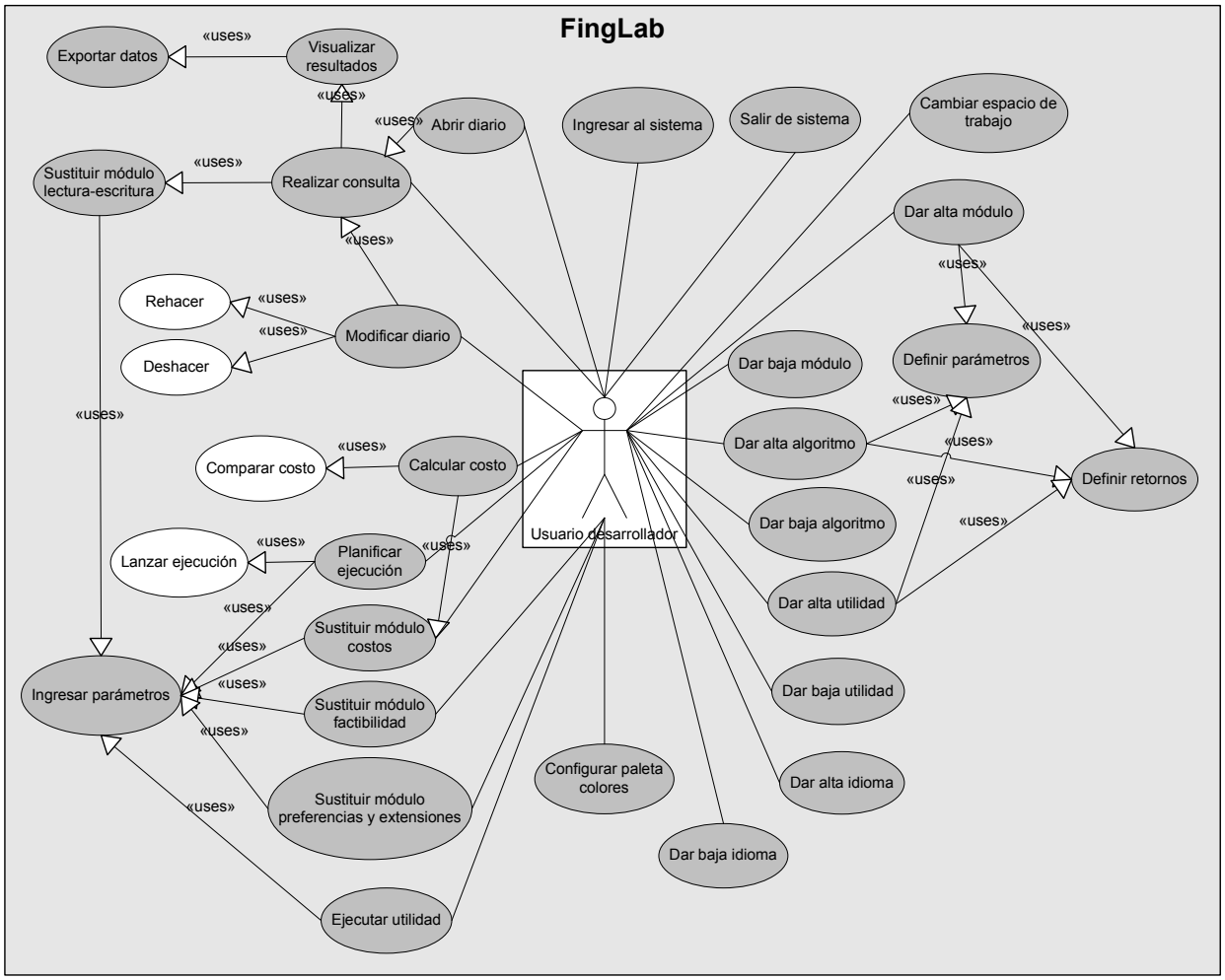


Figura 7-1: Casos de uso relevantes a la arquitectura.

A continuación brevemente se detallan las razones por las cuales estos casos de uso son relevantes al diseño de la arquitectura. Una explicación más detallada de cada caso de uso se encuentra en el Anexo: Documentación .

Ingresar al sistema: Al ingresar al sistema, se realiza la verificación del tipo de usuario que utilizará el sistema (y por ende, las funcionalidades que se pondrán a su disposición) y el espacio de trabajo a cargar (en el idioma correspondiente en que esté dicho espacio y con las opciones por defecto definidas para el mismo).

Salir del sistema: Al salir del sistema, se realiza la descarga de cualquier objeto de visualización creado y, si hay datos que no están guardados, se da la opción al usuario de guardarlos antes de salir.

Cambiar espacio de trabajo: El usuario puede cambiar de espacio de trabajo o crear uno nuevo en el cual puede mantener los algoritmos que desee y las ejecuciones de los mismos. Cada espacio de trabajo es

independiente uno de otro, se crea en un idioma determinado y se puede configurar de manera particular, lo cual permite definir las funcionalidades que solo estén disponibles para dicho espacio de trabajo.

Dar alta de módulo: Al dar de alta un módulo (ya sea, de lectura-escritura, de costos, de factibilidad o de preferencias y extensiones) se definen los tipos de parámetros que recibe dicho módulo al crear una instancia del mismo. Esto permitirá crear las instancias a la hora de sustituir los módulos dinámicamente.

Dar baja de módulo: Al dar de baja un módulo que fue dado de alta por el usuario, el mismo ya no quedará disponible para su utilización, por lo que cualquier configuración que utilice dicho módulo deberá utilizar los módulos por defecto provistos por FingLab.

Dar alta de utilidad: Al dar de alta una utilidad, se permite al usuario incorporar y reutilizar programas ejecutables desarrollados en otros lenguajes a FingLab. De esta manera, no es necesario re-implementar dicho programa en el mismo lenguaje de programación de FingLab y su ejecución permite producir resultados que pueden ser utilizados en las ejecuciones o al armar un servicio manualmente (como por ejemplo, un programa que cree todos los posibles viajes expresos entre todos los destinos de una solución).

Dar baja de utilidad: Al dar de baja una utilidad, la misma ya no estará disponible para su utilización, por lo que no se puede dar la opción de invocarla nuevamente dentro de FingLab.

Dar alta de algoritmo: Al dar de alta un algoritmo, se definen los tipos de parámetros que recibe al invocar su ejecución y los tipos de retorno que puede tener para así generar las advertencias adecuadas al usuario cuando su ejecución termine. Esto permitirá realizar las ejecuciones que crean las soluciones a los problemas de ordenamiento. Se le puede asignar un módulo de lectura-escritura, que permita leer y escribir las soluciones en el formato con que son creadas por el algoritmo (en caso de no ser el formato por defecto), lo cual da flexibilidad a la hora de realizar consultas sobre un diario.

Dar baja de algoritmo: Al dar de baja un algoritmo, el mismo ya no estará a disposición de los usuarios para realizar ejecuciones con el mismo. Cualquier solución generada con el mismo se mantendrá en FingLab y podrán seguir siendo consultadas o utilizadas como entradas para una ejecución posterior (con algún otro algoritmo).

Dar alta de idioma: Al dar de alta un idioma, éste queda a disposición de los usuarios para crear un espacio de trabajo nuevo en el mismo. Un idioma permite redefinir todos los mensajes y etiquetas de la interfaz, y además, definir nuevas etiquetas en el menú, que podrán dar acceso a funcionalidades que el usuario implemente en un módulo de preferencias desarrollado por él. La habilitación o no de las opciones del menú también es manejada por medio de la definición del idioma.

Dar baja de idioma: Al dar de baja un idioma no se podrá acceder a ningún espacio de trabajo que esté definido en dicho idioma.

Configurar paleta de colores: La configuración de la paleta de colores consiste en la definición de todos los colores, escalas y tamaños de fuente manejados en la interfaz (especialmente en el gráfico). Incluye la definición de horas pico dentro del gráfico y el zoom in/out en el mismo.

Sustituir módulo de costos: Al sustituir un módulo de costos, se deben ingresar los valores de los parámetros (según la definición de los tipos de parámetros definida al dar de alta dicho módulo). La instancia de dicho módulo se crea y, a partir de este momento, todos los cálculos de costos se realizarán con las fórmulas definidas en el mismo. Si el usuario lo desea puede volver a seguir utilizando módulo de costos por defecto y la instancia del módulo anterior se destruye.

Sustituir módulo de factibilidad: Al sustituir un módulo de factibilidad, se deben ingresar los valores de los parámetros (según la definición de los tipos de parámetros definida al dar de alta dicho módulo). La instancia de dicho módulo se crea y, a partir de este momento, todos los chequeos de factibilidad se realizarán con las fórmulas definidas en el mismo. Si el usuario lo desea puede volver a seguir utilizando módulo de factibilidad por defecto y la instancia del módulo anterior se destruye.

Sustituir módulo de preferencias y extensiones: Al sustituir un módulo de preferencias y extensiones, se deben ingresar los valores de los parámetros (según la definición de los tipos de parámetros definida al dar de alta dicho módulo). La instancia de dicho módulo se crea y, a partir de este momento, se utilizarán las funcionalidades definidas en el mismo. Si el usuario lo desea puede volver a seguir utilizando módulo de preferencias y extensiones por defecto y la instancia del módulo anterior se destruye.

Ejecutar utilidad: Al ejecutar una utilidad, se debe seleccionar la ejecución en la cual se desean generar resultados con dicha utilidad e ingresar los valores de los parámetros (según la definición de los tipos de parámetros definida al dar de alta dicha utilidad). Los resultados generados con dicha utilidad se podrán utilizar, por ejemplo, para el armado manual de servicios.

Calcular costos: Al calcular los costos, se utiliza el módulo de costos del cual hay una instancia creada y se podrán comparar los costos de hasta dos diarios simultáneamente.

Planificar ejecución: Al planificar una ejecución, se deben ingresar los valores de los parámetros del algoritmo (según la definición de los tipos de parámetros definida al dar de alta dicho algoritmo). Se genera una instancia que iniciará la ejecución del algoritmo con los datos ingresados y dará aviso al usuario cuando la ejecución culmine. El lanzamiento de la ejecución se puede cancelar antes de que comience y pausar o abortar durante su ejecución. Todos los resultados se guardarán en el directorio creado para dicha ejecución. Cualquier error durante la ejecución se registra para que el usuario pueda analizarlo posteriormente.

Abrir diario: Al abrir un diario, se carga el módulo de lectura-escritura definido para el algoritmo con el que se creó dicho diario. Una vez creada la instancia del módulo correspondiente se crea una estructura que mantiene el diario cargado en memoria para su consulta.

Modificar diario: Al modificar un diario (que está abierto), se pueden realizar alteraciones a su estructura: crear nuevos servicios manualmente y desglosar servicios que pertenezcan al diario. Esta funcionalidad incluye las propiedades de deshacer o rehacer cambios cuando se está trabajando en el armado y se guardará como un diario nuevo en la ejecución a la que corresponde.

Realizar consulta: Al realizar una consulta, los resultados se despliegan los datos en las diferentes áreas de la interfaz (en formato tabular y gráfico) para su análisis. También se da la posibilidad al usuario de formatear y exportar dichos datos para que puedan ser utilizados por otra aplicación.

7.2 Modelo de diseño

Para cumplir con los requerimientos descriptos en la sección 5 Requerimientos del sistema agrupamos las funcionalidades en paquetes como muestra la Figura 7-2 (diagramas creados con NetBeans).

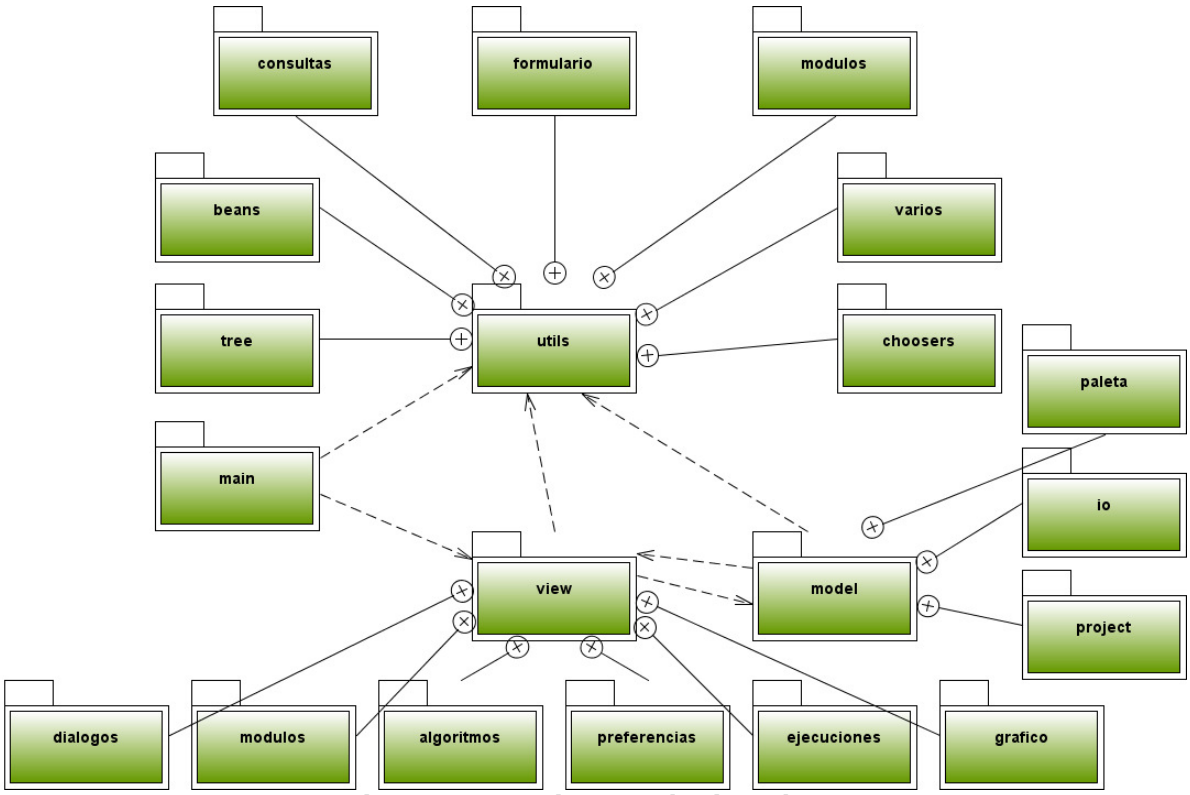


Figura 7-2: Arquitectura de FingLab.

Notación:

A ⊕— B B está contenido en A.

 Paquete de clases.

A -→ B Relación de dependencia. El elemento A depende de B.

A continuación se describe de forma general los patrones de diseño utilizados en el desarrollo de FingLab y las funcionalidades de cada paquete o subsistema. Una descripción mas detallada, junto con los casos de uso y diagramas de clases, se encuentra en el Anexo: Documentación .

7.2.1 Patrones de diseño

Los patrones de diseño son soluciones especificadas a problemas comunes de diseño de software [27]. Para la implementación de FingLab se utilizaron tres patrones de diseño: Model-View-Controller, Observer y Command. A continuación brevemente se explica en qué consiste cada patrón y la razón de su utilización. Para más detalles sobre los patrones vea el Anexo: Documentación .

7.2.1.1 Patrón de diseño: Model-View-Controller

Este tipo de arquitectura, que también es considerada un patrón, permite tener de forma independiente la lógica de la aplicación ("controller" o "controlador"), de la forma en que se guardan los datos ("model" o modelo) y de la forma en que los datos se presentan en la interfaz ("view" o "visualización"). Esto facilita el desarrollo independiente, el testeo y el mantenimiento de cada parte [27]. En FingLab utilizamos este patrón para poder almacenar los datos como se desee (en archivo de texto, base de datos, etc.), leer o escribir los datos en el formato correspondiente (la parte controlador) y presentar los datos en la interfaz con la visualización provista.

7.2.1.2 Patrón de diseño: Observer

Este patrón consiste en un objeto, llamado "subject" (o "sujeto"), el cual mantiene una lista de sus dependientes, llamados "observers" (u "observadores"), a los cuales notifica automáticamente de cualquier cambio de estado, usualmente mediante una llamada a uno de sus métodos. De esta manera, cuando el subject cambia, todos los objetos que dependen de él son notificados para tomar las acciones correspondientes para reflejar dicho cambio [27]. En FingLab utilizamos este patrón para que cuando se obtiene un resultado de una consulta, las diferentes vistas de los resultados (la vista en formato de tabla y en formato gráfico), se actualicen automáticamente para mostrar dicho resultado.

7.2.1.3 Patrón de diseño: Command

Este patrón consiste en un objeto que representa o encapsula toda la información necesaria para llamar a un método más tarde en el tiempo. Esta información incluye el nombre del método, el objeto que tiene dicho método y los valores para los parámetros de dicho método [27]. De esta manera, al implementar este patrón en FingLab, podemos guardar las operaciones que se van realizando durante la modificación manual de servicios, e implementar la funcionalidad de deshacer o rehacer.

7.2.2 Subsistema Main

Este subsistema contiene la clase que inicia la ejecución de FingLab. Verifica si está instalado, y si no lo está lo instala. También crea el menú. Contiene el caso ingresar al sistema y cambiar de espacio de trabajo.

7.2.3 Subsistema View

Este subsistema crea la ventana de FingLab con todos sus paneles y contiene la clase que escucha todos los eventos que se producen en ella.

7.2.3.1 Subsistema Dialogos

Este subsistema contiene las clases de los formularios para abrir un diario, ingresar los datos de cada parámetros al ejecutar un algoritmos y editar las opciones de gráfico (colores, escalas, etc.). Contiene los casos de uso: abrir diario y configurar la paleta de colores.

7.2.3.2 Subsistema Módulos

Este subsistema contiene las clases interfaz que deben implementar los módulos de costos (por ejemplo, para cambiar la fórmula de cálculo de costos), factibilidad (por ejemplo, para cambiar la implementación de las leyes laborales) y extensión (por ejemplo, para cambiar la visualización), así como las clases por defecto que las implementan. Contiene los casos de uso: sustituir módulos de costos, factibilidad y preferencias y extensiones. Este subsistema implementa la parte controller (controlador) del patrón de diseño Model-View-Controller que se utilizó para dar independencia entre el formato de los datos y la forma en que se despliegan en pantalla. este patrón de diseño se explica en el Anexo: Estado del arte de las interfacesy para que el usuario pueda crear e incorporar sus propios módulos se detalla en el Anexo: Documentación .

7.2.3.3 Subsistema Algoritmos

Este subsistema contiene todas las clases de los formularios para dar de alta, baja y modificación a las configuraciones de algoritmos, formulario de definición de retornos de los algoritmos, la clase interfaz que deben implementar los módulos de lectura-escritura de las soluciones producidas por los algoritmos, así como el módulo de lectura-escritura por defecto del sistema. Contiene los casos de uso: alta y baja de algoritmo.

7.2.3.4 Subsistema Preferencias

Este subsistema contiene las clases de los formularios para la edición de las alertas de sonido, opciones de bases de datos relacionales, reporte de errores y manejo de idiomas (para la internacionalización).

7.2.3.5 Subsistema Ejecuciones

Este subsistema contiene todas las clases para realizar las ejecuciones de los algoritmos, configurarlas, seleccionar opciones avanzadas de configuración de ejecución y el panel de las ejecuciones pendientes en el sistema. Contiene los casos de uso planificar y lanzar ejecución.

7.2.3.6 Subsistema Grafico

Este subsistema contiene todas las clases para la creación del gráfico de Gantt, así como la clase interfaz que un gráfico debe implementar para poder ser incorporado a FingLab. Implementa la parte "view" (visualización) del patrón de diseño Model-View-Controller.

7.2.4 Subsistema Model

Este subsistema tiene como función mantener el estado actual del sistema: qué diario hay abierto, cuál es el espacio de trabajo actual, etc.

7.2.4.1 Subsistema Paleta

Este subsistema tiene todas las clases para mantener cargadas las opciones de gráfico actuales: colores, escalas, etc.

7.2.4.2 Subsistema IO

Este subsistema contiene todas las clases para leer y escribir los archivos de configuración internos que maneja FingLab que son del tipo xml.

7.2.4.3 Subsistema Project

Este subsistema contiene las clases para mantener el estado actual del sistema: qué diario hay abierto, cuál es el espacio de trabajo actual, etc. Incluye también las clases encargadas de la funcionalidad hacer/deshacer.

7.2.5 Subsistema Utils

Este subsistema contiene las utilidades utilizadas por todo el sistema. Estas utilidades se agruparon para que puedan servir de librería y ser reusadas.

7.2.5.1 Subsistema Beans

Este subsistema contiene las clases que mantienen la estructura de un diario (BeanDiario) en memoria al abrirlo, con sus servicios (BeanServicio) y turnos (BeanTurno) como muestra el diagrama de la Figura 7-3. Implementa la parte "model" (modelo de datos) del patrón de diseño Model-View-Controller.

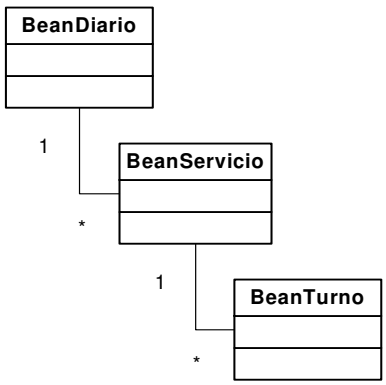


Figura 7-3: Clases para la estructura de un diario.

7.2.5.2 Subsistema Tree

Este subsistema contiene las clases para desplegar el árbol tanto de planificación, como de resultados de consultas.

7.2.5.3 Subsistema Consultas

Este subsistema contiene las clases de los formularios para realizar consultas y exportación de datos en diferentes formatos.

7.2.5.4 Subsistema Formulario

Este subsistema contiene las clases para construir formularios con el formato standard de FingLab sin requerir librerías externas.

7.2.5.5 Subsistema Modulos

Este subsistema contiene todas las clases de los formularios para dar de alta, baja y modificación a módulos y el formulario de definición de parámetros de los módulos (que también se usa para la definición de los parámetros de una configuración de algoritmo). Contiene los casos de uso: alta y baja de módulo y definir de parámetros.

7.2.5.6 Subsistema Choosers

Este subsistema contiene las clases para seleccionar un archivo del disco duro (para dar de alta los ejecutables de los algoritmos), seleccionar una fecha en el calendario (para seleccionar una fecha de ejecución de un algoritmo) y seleccionar una fuente (para la paleta de colores).

7.2.5.7 Subsistema Varios

Este subsistema contiene utilidades variadas que son usadas en todo el sistema. Abarcan desde enumerar el contenido de un directorio, a utilidades para leer desde xml, txt, dbf y bases de datos relacionales.

8 Proceso de desarrollo

En esta sección se detallan los pasos del proceso de desarrollo de la interfaz diseñada.

8.1 El punto de partida: LGantt

En el análisis de herramientas que implementan diagramas de Gantt (que se detalla en el Anexo: Documentación) encontramos a Lgantt. Este sistema permite la administración de proyectos y recursos, y posee una visualización gráfica de diagrama de Gantt [50]. La Figura 8-1 (imagen extraída de http://ghannid.homeip.net/wiki/LGanttPrd_en.wiki) muestra su interfaz gráfica.

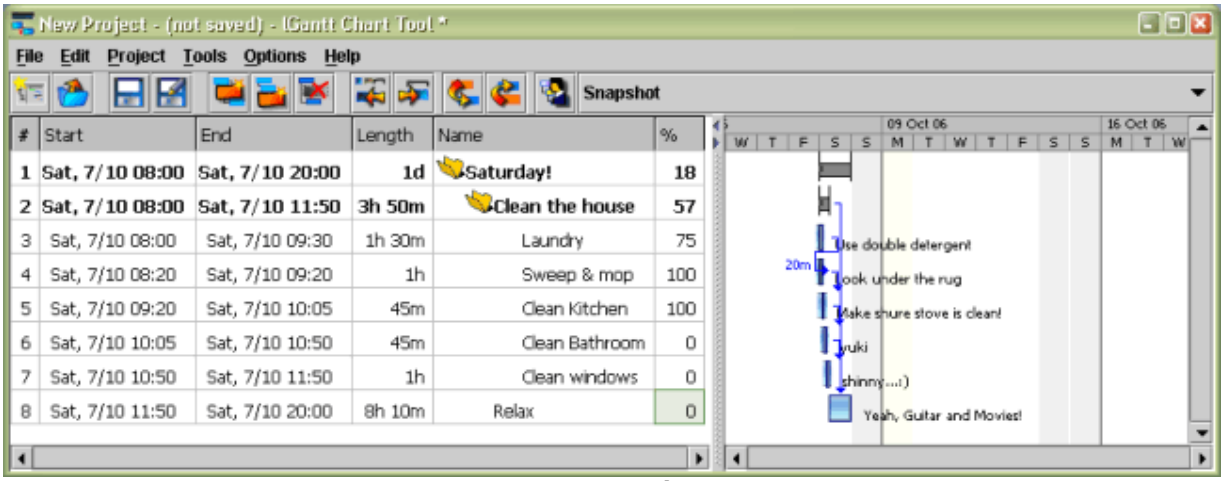


Figura 8-1: Interfaz gráfica de LGantt.

Está implementado en Java, es una herramienta libre y el código fuente está disponible, lo cual nos permite usarlo de base para construir FingLab. Con este punto de partida seguimos los siguientes pasos:

- Eliminamos del código original todos aquellos elementos que no utilizaremos (manejo de recursos, funcionalidades relativas a la duración de los eventos, etc.).
- Realizamos modificaciones al código restante para que los componentes que reutilizamos cumplan con nuestros requerimientos:
  - El diagrama de Gantt lo alteramos para que soporte una escala de minutos (lo mínimo que toleraba eran horas).
  - El diagrama de Gantt le pusimos una escala máxima de 24 horas ya que los resultados de la planificación que se grafican son siempre dentro de un día.
  - Agregamos aquellos paneles que faltan: el panel de planificaciones, el panel de ejecuciones pendientes, el panel de resultados y el panel de tabla (utilizamos el mismo tipo de panel que tenía originalmente, ya que permiten que se oculten/muestren según desee el usuario). Agregamos la funcionalidad de marcar el área activa en la pantalla cuando se hace click en algún panel en particular.
- Realizamos la implementación de las funcionalidades requeridas para nuestro sistema.

FingLab es una aplicación de escritorio, pero si se desea, se puede convertir en un applet y ponerlo a disposición para ser accedido en línea, gracias a que el lenguaje Java lo permite.

8.2 Desarrollo de FingLab

Siguiendo los resultados de nuestro estudio llegamos a la implementación de FingLab, como se muestra en la Figura 8-2.

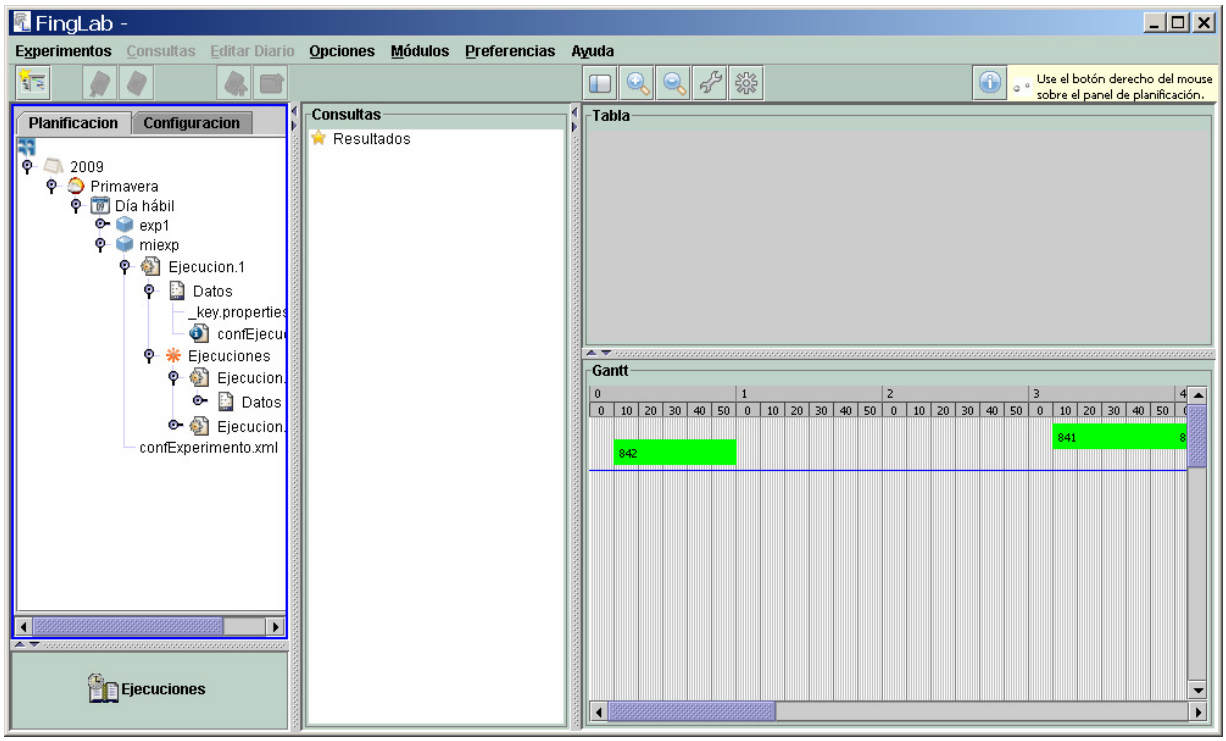


Figura 8-2: Interfaz gráfica de FingLab.

Como se puede apreciar, todos los elementos que se contemplaron en el diseño aparecen en el producto final, luego de un largo proceso de desarrollo. Permite dar de alta los ejecutables de los algoritmos, ejecutarlos, organizar sus ejecuciones y visualizar los resultados. FingLab tiene un módulo de cálculo de lectura de archivos con las soluciones, cálculo de costos y cálculos de factibilidad, pero gracias a las propiedades del lenguaje Java, podemos sustituir dichos módulos en tiempo de ejecución para evaluar los resultados de diferentes formas. Los detalles de cómo trabajar con FingLab para realizar las tareas que se especificaron en el ciclo de trabajo se encuentran en el Anexo: Documentación .

A continuación se especifican las diferentes iteraciones de dicho proceso. Si bien FingLab cumple con los requerimientos, durante su desarrollo se profundizó en la reutilización de componentes y en implementar un sistema que pueda ser fácilmente adaptable a otros tipos de algoritmo y experimentación, por lo que se explica cómo se llegó a desarrollar el producto final [48, 53].

Durante el proceso de desarrollo se tomaron en cuenta las siguientes normas de implementación:

- Generalización y parametrización de funcionalidades
  - Hacer formularios genéricos y parametrizables:
    - Fácil construcción de formularios con funciones que implementen los seteos gráficos.
    - Pasar por parámetro los mensajes en el idioma correspondiente.
    - Pasar por parámetro los directorios con los que se trabaja.
  - Agrupar campos de un formulario en un vector:
    - Ejecutar funciones de chequeo, limpieza para habilitar botones, etc.
  - Reconocer en qué lugares del código conviene hacer una llamada a un módulo intercambiable para agregar funcionalidad.

- Reutilización de código
  - Parametrizar cualquier línea de código:
    - Convertirla en una utilidad no importa si es corta.
  - Hacer funciones generales y parametrizables que se puedan aplicar a la mayor cantidad de casos posibles para ofrecer robustez frente a cambios de formato por ejemplo.

- Revisión de código
  - Reducir la cantidad de líneas de código:
    - Con estas pautas el código de una clase puede reducirse en un promedio de 30% a 40%.
    - Más fácil de verificar e identificar de dónde vienen los errores.
    - Más fácil visualización de lo que se está haciendo sin distracciones del código gráfico o código de utilidades.
    - Mayor mantenibilidad.

Para ello, el ciclo de ingeniería de software (en todas sus fases: análisis, diseño, implementación y verificación) se repitió tres veces, cada vez agregando un cambio de visión que llevó al producto a un nuevo nivel de desarrollo. Cada iteración tenía un meta, una vez llegada a esa meta se analizaba el producto resultado de la iteración y se planteaba una nueva meta a alcanzar (Figura 8-3). Si bien esto cambió el proceso de desarrollo planificado en un principio, el resultado es un producto que cumple con los requerimientos establecidos pero también mucho más amplio de lo esperado.

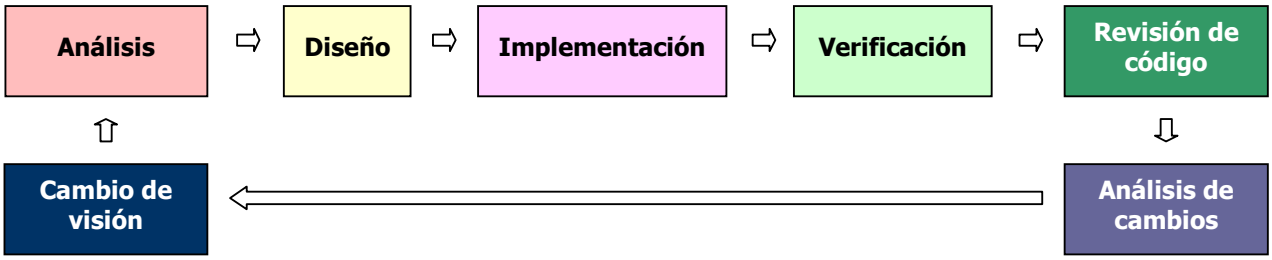


Figura 8-3: Proceso de desarrollo de FingLab.

8.3 Iteraciones del proceso

Durante todo el proceso se realizaron tres iteraciones:

1. Meta: Construir una herramienta de experimentación con algoritmos de planificación de vehículos
   a. Hacer formularios para obtener funcionalidades primarias (algoritmos, módulos y ejecuciones).
   b. Hacer formularios para obtener funcionalidades secundarias (opciones de bases de datos, email y sonidos).
      i. El programa de mandar mails y ejecutar sonidos es muy particular (no funciona con todos los protocolos de email y todos los formatos de archivos de sonido) -> Aparece la necesidad del módulo de preferencias y extensiones (tener estas funcionalidades en un módulo aparte que pueda ser sustituido por el usuario si lo desea).
      ii. Aumenta el número de formularios -> Aparece la necesidad de tener herramientas que faciliten la implementación de los formularios.
   c. Posibles cambios en los formatos de las salidas.
      i. Aparece la necesidad de ofrecer posibilidades de configuración.

2. Meta: Construir una herramienta de experimentación con algoritmos de planificación de vehículos y proveer de configurabilidad por el usuario y facilitar la programación por parte del usuario.
   a. Módulo de preferencias.
   b. Elaborar archivos de configuración.

c. Hacer una biblioteca de funcionalidades generales.
d. Hacer clases con funcionalidades para crear formularios.
    i. Aparece la necesidad de refinar formularios para reducir código y facilitar verificación utilizando las nuevas bibliotecas.
        1. Decidir qué puede ser reemplazable.
        2. Decidir qué puede ser configurable.
        3. Decidir qué puede ser fijo.
        4. Decidir qué puede ser utilidad.

3. Meta: Trabajar en los módulos reemplazables y construir una herramienta de experimentación que permita trabajar con cualquier algoritmo (con una visualización particular que es el diagrama de Gantt que se puede sustituir por cualquier otra según el tipo de algoritmo)
    a. Hacer la visualización reemplazable.
    b. Hacer los módulos por defecto de costos, factibilidad y lectura reemplazables por medio del módulo de preferencias.
    c. Hacer bibliotecas más específicas agrupando funcionalidades:
        i. Funcionalidades para trabajar con bases de datos.
        ii. Funcionalidades para cargar clases dinámicamente (trabajando con la propiedad reflection de Java).
        iii. Funcionalidades para trabajar con archivos y directorios.
        iv. Funcionalidades generales utilizadas en todo el sistema.
        v. Funcionalidades para lectura y escritura de datos en formatos:
            1. Bases de datos relacionales.
            2. Archivos DBF.
            3. Archivos XML.
            4. Archivos TXT.
        vi. Funcionalidades para trabajar con archivos multimedia (archivos de sonido).
        vii. Funcionalidades para trabajar con usuarios.

8.4 El producto final: FingLab como framework

Durante el proceso de desarrollo se manejaron una lluvia de conceptos como muestra la Figura 8-4.

(imagen generada con [8])

Figura 8-4: Conceptos manejados en el proceso de desarrollo de FingLab.

Lo que hicimos en cada iteración fue tomar todos esos conceptos, analizarlos, abstraerlos y generar una nueva visión de la herramienta para llevarla a desarrollo como muestra la Figura 8-5.

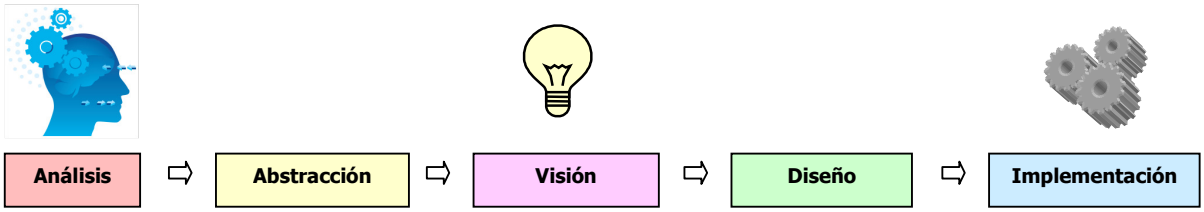


Figura 8-5: Proceso de abstracción y rediseño de FingLab.

Esta abstracción de conceptos forma un puzzle de piezas que componen el sistema final como muestra la Figura 8-6, en el cual algunas piezas se pueden sustituir para generar un nuevo sistema que cumpla con nuevos requerimientos.

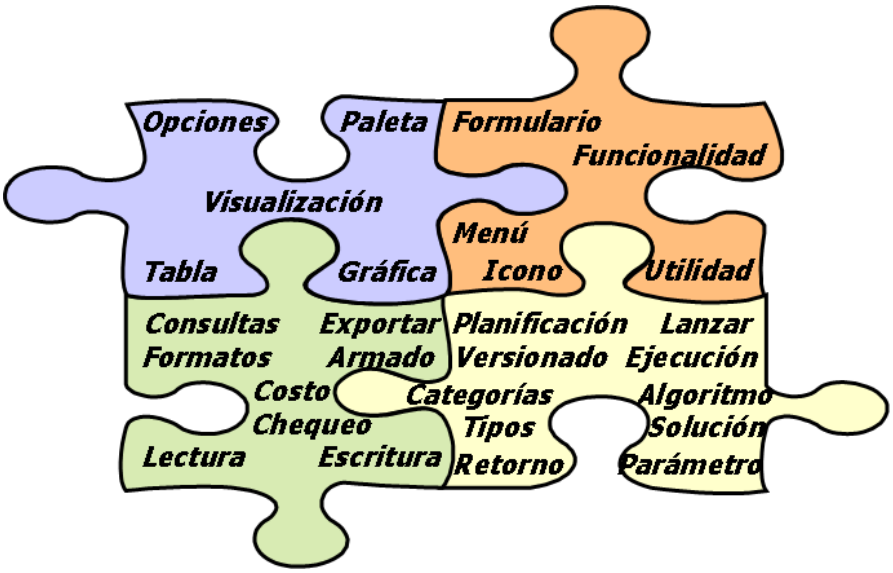


Figura 8-6: Puzzle de conceptos.

La definición de **FingLab** depende del usuario a quien esté destinado:

- Para un usuario final Standard o planificador es un sistema que le permite interactuar con soluciones de algoritmos y visualizar los datos así como realizar modificaciones.
- Para un usuario final Administrador o Desarrollador es un sistema que le permite experimentar con diversos algoritmos y analizar sus soluciones.

Pero para que ambos usuarios puedan trabajar con **FingLab**, es el usuario desarrollador quien pone a disposición de los usuarios finales (tanto desarrollador-Administrador como planificador-Standard) las funcionalidades ofrecidas por **FingLab** para trabajar con los tipos de algoritmos con los que se desea trabajar. Es por eso que **FingLab** es más que un sistema de apoyo a la experimentación o planificación, sino que también es un Framework que permite la creación de estos sistemas. Por eso podemos referirnos a **FingLab** como:

FingLab v. 1.0

***F**ramework* para ***in**terfaces* ***g**ráficas* de ***L**aboratorios* de ***a**lgoritmia* y ***b**eta-testing*

Podemos así crear con FingLab un laboratorio propio de experimentación de acuerdo al tipo de algoritmos que estemos probando, con su propia visualización, lectura de datos y consultas.

Luego de las sucesivas fases de desarrollo tenemos una herramienta que permite:

- Cambiar el menú y agregar funcionalidades nuevas.
- Manejar usuarios.
- Crear nuevos formularios.
- Cambiar la estructura de directorios de un espacio de trabajo.
- Agregar un idioma por defecto.
- Programar un módulo de lectura / escritura (IO).
    - Cambiar los keys (la forma de leer un archivo con datos).
- Programar un módulo de cálculo de costos.
- Programar un módulo de factibilidad.
- Programar un módulo de extensiones y preferencias.
    - Cambiar los módulos por defecto.
- Cambiar la visualización.
- Cambiar las categorías de clasificación de las ejecuciones.
- Cambiar los íconos del árbol de planificación.
- Cambiar la planificación de las ejecuciones.
- Cambiar la ejecución de una alerta de sonido (poder ejecutar archivos que sean wma, mp3, etc.).
- Cambiar el protocolo de envío de mails (diferente al que está por defecto).
- Cambiar las plantillas de impresión.
- Cambiar la forma de impresión (imprimir en formatos que no sean pdf).
- Cambiar los estados del sistema (reprogramar la máquina de estados) y las reacciones del sistema ante cambios de estado (transiciones de estado).
- Incorporar utilidades que no necesariamente pueden estar programadas en Java.

8.5 Flexibilidad y Reuso de los componentes de la interfaz gráfica

Como resultado del proceso de desarrollo FingLab tiene componentes de tipo fijo (que no se pueden cambiar a menos que que altere el código del sistema), personalizables (que se pueden cambiar dando de alto nuevos componentes) y reusables (librerías con funcionalidades que pueden ser reutilizadas al implementar módulos para añadir a FingLab). A continuación se detallan algunos de estos componentes:

- **Fijo**
    - *Estructura de instalación*: no se puede modificar ya que es lo mínimo fijo que se requiere para el funcionamiento de la aplicación.

- **Personalizable**
    - *Paneles gráficos*: se puede redefinir en el módulo de preferencias y extensiones los paneles a agregar en la interfaz, para así cambiar las formas de visualización (por ejemplo: en vez de un gantt, agregar otra forma de visualización).
    - *Funcionalidades del menú*: se pueden implementar en el módulo de preferencias y extensiones con los comandos definidos en el archivo de menú.
    - *Menú y barra de botones*: se pueden redefinir los ítems del menú y barra de botones e incluso agregar nuevos, con habilitación/deshabilitación o visibilidad según el estado del sistema. También se pueden agregar áreas de relleno flexibles cuando hay botones que aparecen o desaparecen en la barra de botones.
    - *Textos, mensajes e iconografía*: se puede redefinir los textos y mensajes de toda la interfaz en el archivo de messages y menú, así como los íconos e incluso definir nuevos para la utilización en nuevas funcionalidades.

- *Estructura de espacio de trabajo*: la estructura del espacio de trabajo se puede modificar (en parte) en el archivo de messages para poder contener nuevos directorios en los que guardar componentes (módulos).
  - *Categorización de las planificaciones*: las planificaciones se pueden categorizar de manera flexible, definiendo las categorías en el archivo de messages dependiendo si cada categoría tiene ítems predeterminados o no.
  - *Tipos de eventos*: los tipos de eventos se pueden definir para cada algoritmo en el módulo de lectura con sus colores por defecto.
  - *Lectura y escritura en diferentes medios y formatos*: el formato o medio en el cual se guardan los datos puede cambiar, pero el usuario siempre podrá implementar módulos de lectura y escritura para dichos datos, que se asignan según el algoritmo con el que se trabaja con cada conjunto de datos.
  - *Cálculos*: la forma de realizar cálculos puede ser reimplementada por el usuario y asignada en tiempo de ejecución.
- *Preferencias generales*
  - *Alertas de sonido*: las alertas se dan de alta en el directorio de instalación para toda la aplicación y pueden ser configuradas por el usuario.
  - *Notificaciones de email*: las notificaciones se pueden configurar a diferentes niveles:
    - Directorio de instalación: para brindar soporte a la aplicación en caso de errores inesperados.
    - Directorio de espacio de trabajo: para apoyar el trabajo cooperativo y compartir soluciones.
    - Directorio de experimento: para apoyar el trabajo cooperativo y compartir soluciones.
    - Directorio de ejecuciones: para apoyar el trabajo cooperativo, compartir soluciones y enviar las soluciones al final de una ejecución.
  - *Opciones de BD*: las opciones de BD se pueden configurar a diferentes niveles:
    - Directorio de instalación: para tener una configuración por defecto para todos los espacios de trabajo.
    - Directorio de espacio de trabajo: para tener una configuración por defecto para todos los experimentos.
    - Directorio de experimento: para tener una configuración por defecto para todas las ejecuciones.
    - Directorio de ejecuciones: para tener una configuración de la cual leer o escribir datos en el medio BD.
  - *Drivers*: los drivers se dan de alta en el directorio de instalación para toda la aplicación y estar disponibles para las opciones de BD.
  - *Idiomas*: los idiomas se dan de alta en el directorio de instalación para toda la aplicación y pueden ser configurados por el usuario para modificar los menús, barra de botones, textos, mensajes e iconografía.

- **Reusable**
  - *Mecanismo de construcción del menú drop-down y barra de botones con estados del sistema*: al cambiar el archivo de menú se puede seguir utilizando el mecanismo que utiliza dicho archivo para la construcción del menú y la barra de botones en la aplicación, así como aplicando los cambios de estado a los mismos.

- *Mecanismo de planificación y lanzamiento de ejecuciones*: al cambiar la categorización de ejecuciones se puede seguir utilizando el mecanismo que planifica y lanza las ejecuciones, así como definir el nombre de la nueva ejecución.
- *Opciones de visualización (colores y fuentes). Observer de visualización*: se puede reutilizar el formulario de definición de colores, fuentes y escalas para cualquier gráfico. Cada vez que se realice una modificación el gráfico se actualiza aplicando la nueva selección.
- *Mecanismo de ABMC de algoritmos*: se puede seguir utilizando el mecanismo de ABMC de algoritmos para cualquier tipo de algoritmo (con sus parámetros y retornos) y definir en qué directorio se desea guardar.
- *Mecanismo de consultas*: se puede seguir utilizando el mecanismo de consultas basado en los módulos de lectura o escritura definidos para un algoritmo.
- *Mecanismo de armado*: se puede seguir utilizando el mecanismo de armado basado en los módulos de lectura o escritura definidos para un algoritmo.
- *Mecanismo de ABMC de módulos*: se puede seguir utilizando el mecanismo de ABMC de módulos para cualquier tipo de módulo y definir en qué directorio se desea guardar.
- *Visualización*: toda la interfaz gráfica se puede reutilizar modificando algunos de sus componentes:
  - *Paneles*: se puede intercambiar en el módulo de preferencias qué paneles se desea ver o conservar los que ya hay por defecto.
  - *Formularios*: se puede reutilizar los formularios que ya hay o definir nuevos formularios basados en un formulario genérico que hay a disposición del usuario.
  - *Árbol de versionado*: se puede reutilizar e árbol de versionado para la planificación, configuración o cualquier otro nuevo aspecto que se desea agregar a la aplicación.
  - *Árbol de consultas*: se puede reutilizar el árbol de consultas para mantener las consultas realizadas que se van a graficar en la tabla y el gantt.
  - *Árbol de armado*: se puede reutilizar el árbol de armado para armar cualquier nuevo elemento que se desea agregar a los datos de un algoritmo y graficarlo en la tabla y el gantt.
  - *Tabla*: se puede reutilizar la tabla con cualquier cantidad de columnas y filas.
  - *Gantt*: se puede reutilizar para algoritmos que requieran este tipo de visualización o intercambiarlo con algún otro tipo de visualización.

9 Demo con FingLab

Al iniciar la ejecución de FingLab por primera vez se debe crear el espacio de trabajo en el cual se guardarán los algoritmos y ejecuciones con los cuales el usuario trabajará. Seleccione el idioma en el que desea crear el espacio de trabajo y presione el botón "Aceptar" como muestra la Figura 9-1.

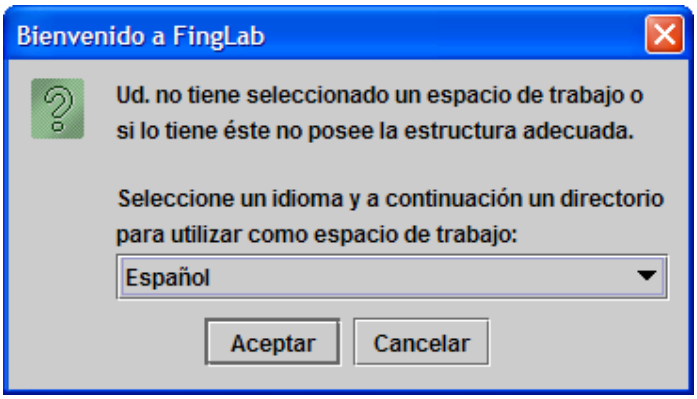


Figura 9-1: Selección de idioma.

Al presionar el botón "Aceptar" se despliega un selector de archivos que permite explorar el disco duro para seleccionar el directorio donde se creará el espacio de trabajo. El directorio debe ser vacío (o contener la estructura de directorios especificada para un espacio de trabajo) y debe ser creado antes de presionar el botón "Aceptar". Al presionar el botón "Abrir" en el selector de archivos, Si el directorio seleccionado es vacío y se pudo crear la estructura de directorios necesaria para el espacio de trabajo (o si la estructura ya existía y es correcta) se desplegará el mensaje que permite seleccionar con qué usuario desea entrar al espacio de trabajo como muestra la Figura 9-2.

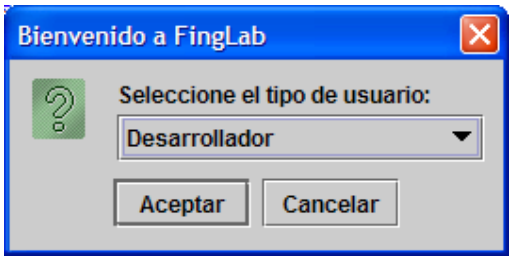


Figura 9-2: Selección de tipo de usuario.

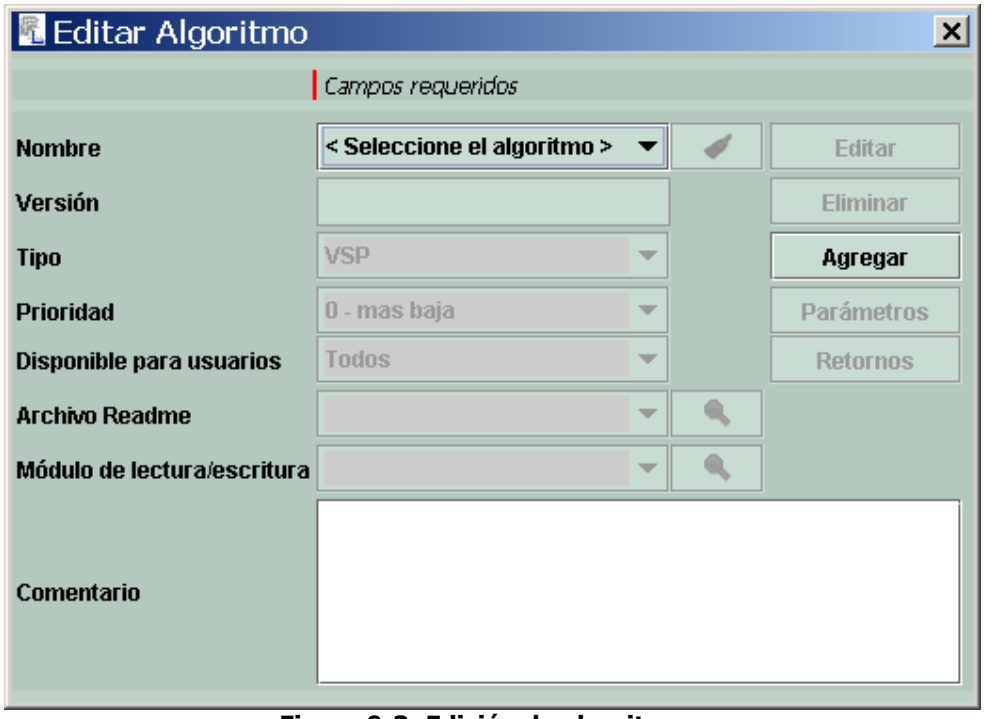


Figura 9-3: Edición de algoritmos.

En el formulario de edición de algoritmos tenemos todos los algoritmos que están disponibles en el sistema, como muestra la Figura 9-3. Al presionar el botón "Agregar", podemos incorporar un nuevo algoritmo al sistema.

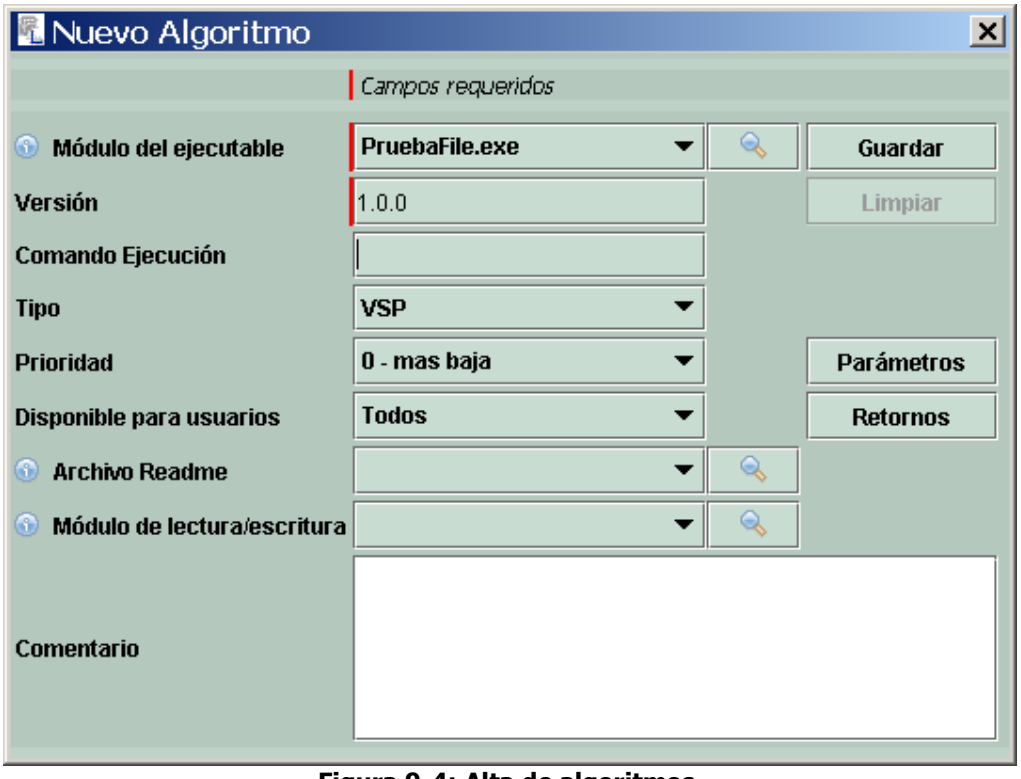


Figura 9-4: Alta de algoritmos.

En el formulario de alta que se muestra en la Figura 9-4, ingresamos todos los datos del algoritmo, incluyendo su ejecutable, los parámetros que recibe (con sus respectivos tipos) como muestra la Figura 9-5 y los retornos

que podemos obtener de su ejecución (para desplegar mensajes adecuados al finalizar la ejecución) como muestra la Figura 9-6.

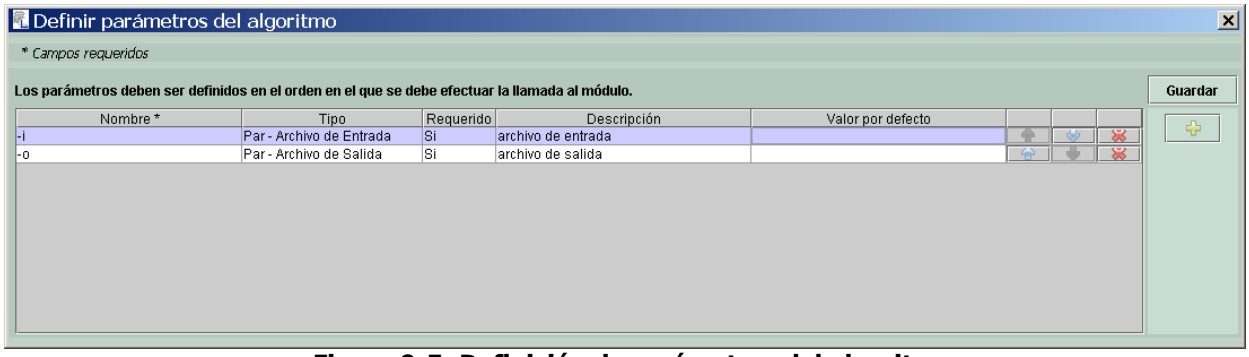


Figura 9-5: Definición de parámetros del algoritmo.

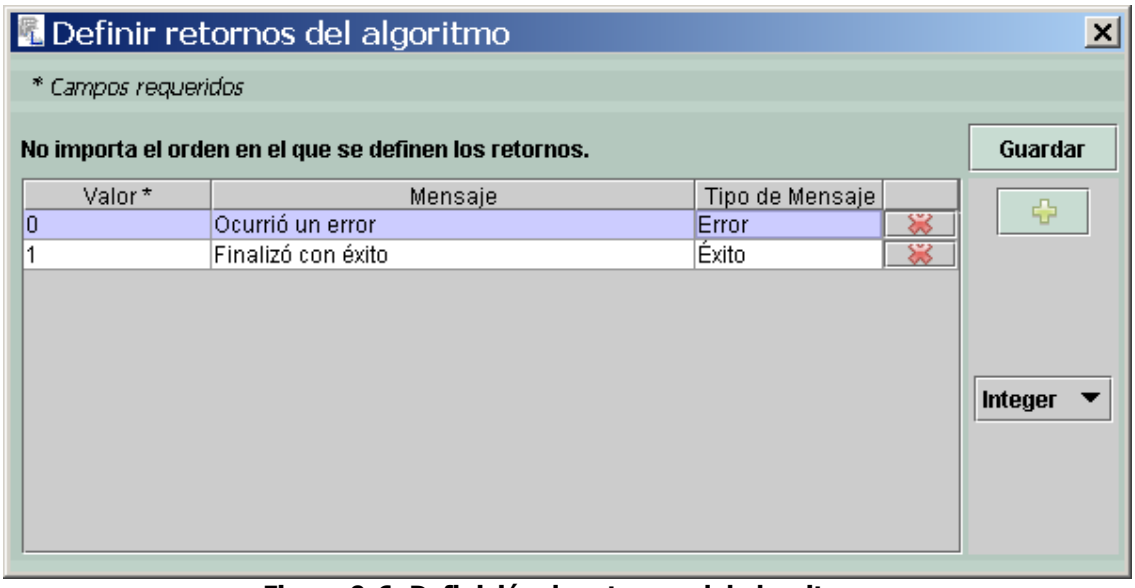


Figura 9-6: Definición de retornos del algoritmo.

Figura 9-7: Alta de algoritmo exitosa.

Al presionar guardar en el formulario de alta de algoritmo, si no hubo errores se despliega un mensaje de éxito, como muestra la Figura 9-7.

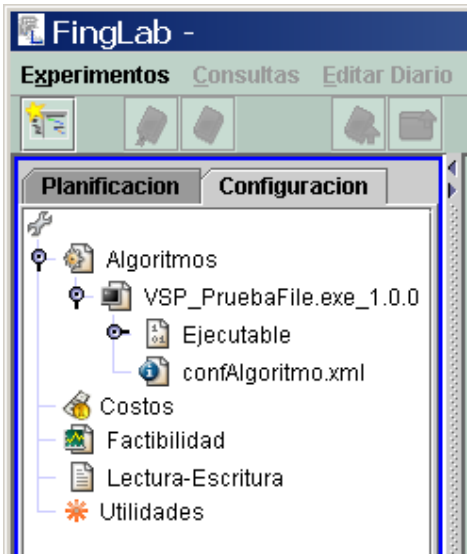


Figura 9-8: Algoritmo incorporado a FingLab.

Inmediatamente, se puede observar en el panel de configuración que el algoritmo fue dado de alta con su ejecutable, como muestra la Figura 9-8.

Para lanzar una ejecución, se debe completar el formulario de la Figura 9-9. En él se puede seleccionar cómo se va a clasificar el algoritmo (por año, estación del año y tipo de día), si la ejecución pertenece a un experimento ya existente o a uno nuevo y si esta ejecución es derivada de otra ya realizada.

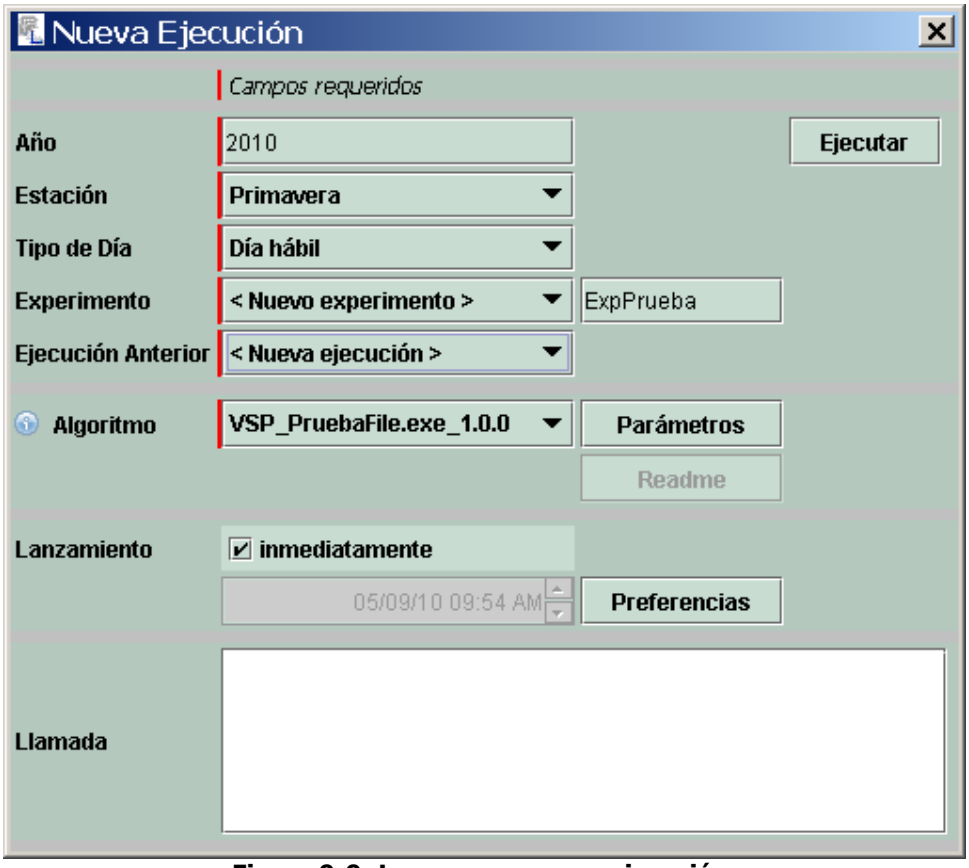


Figura 9-9: Lanzar una nueva ejecución.

Para ejecutar el algoritmo se deben ingresar valores a los parámetros que éste recibe. Para ello, al presionar el botón "Parámetros", se despliega un formulario con campos para completar según los tipos de parámetros definidos al dar de alta el algoritmo como muestra la Figura 9-10.

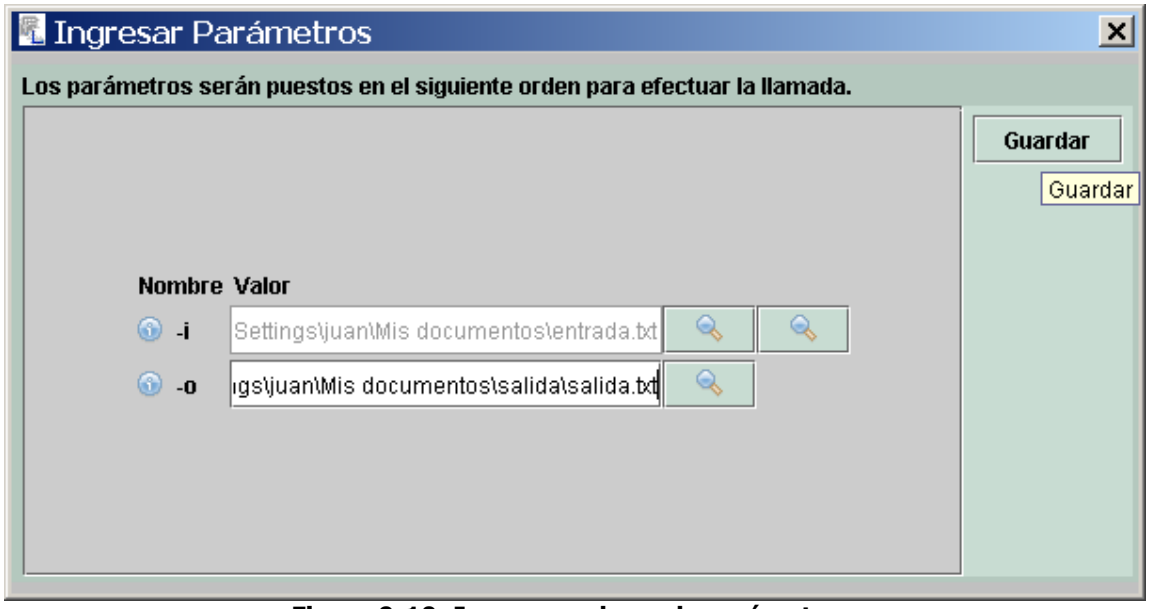


Figura 9-10: Ingresar valores de parámetros.

Una vez que se presiona el botón ejecutar, la ejecución se agenda como muestra la Figura 9-11.

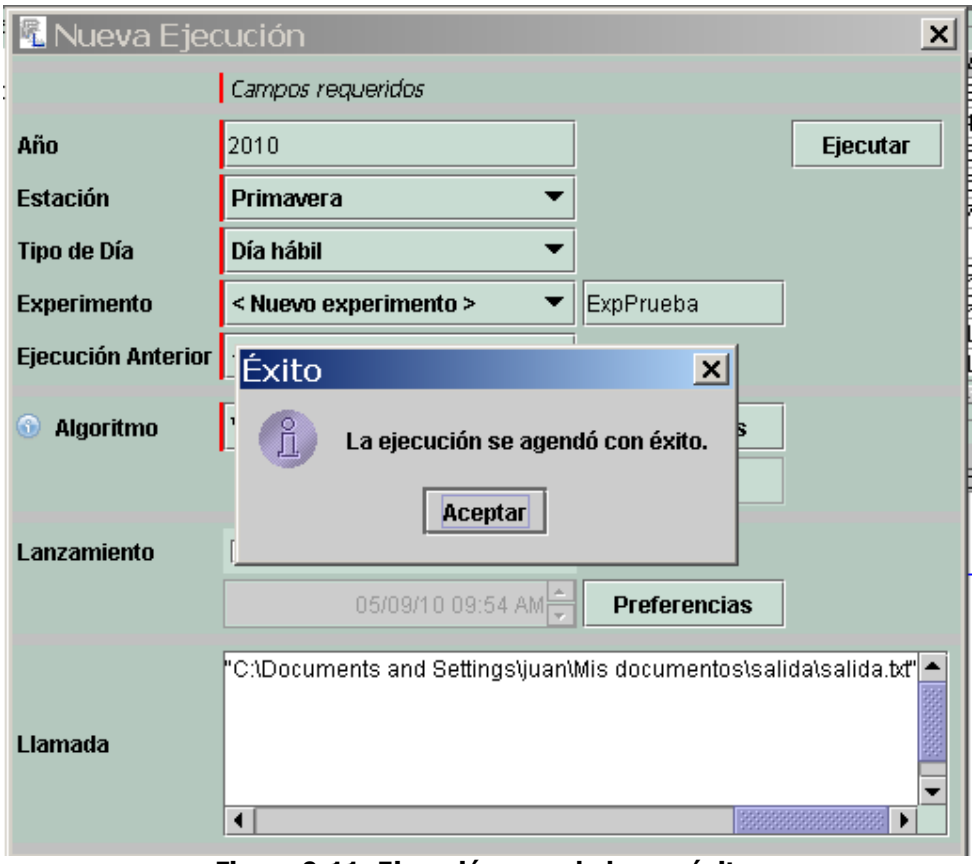


Figura 9-11: Ejecución agendada con éxito.

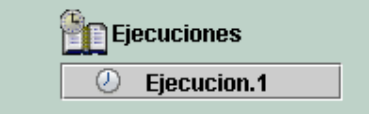


Figura 9-12: Ejecución agendada.

La ejecución también aparece en forma de botón en el panel de ejecuciones pendientes, como muestra la Figura 9-12. Al presionar el botón correspondiente se puede volver a visualizar el formulario de la Figura 9-9 y realizar cambios a la ejecución antes de que sea lanzada o suspenderla si ya fue lanzada.

Figura 9-13: Ejecución finalizada con sus resultados.

Una vez finalizada la ejecución, los resultados de la misma aparecen en el árbol de planificación, como muestra la Figura 9-13.

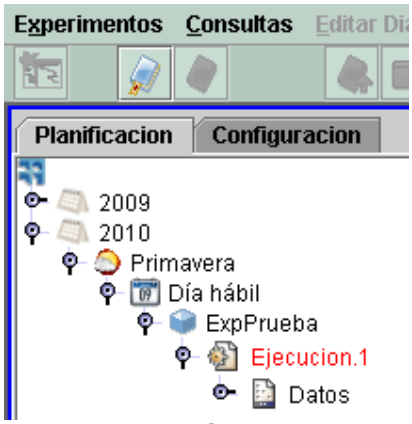


Figura 9-14: Selección de una ejecución.

Para trabajar con una solución, seleccionamos la ejecución a la cual corresponde en el árbol de planificación como muestra la Figura 9-14. Al seleccionarla, se habilitan las opciones de dar de abrir un diario.

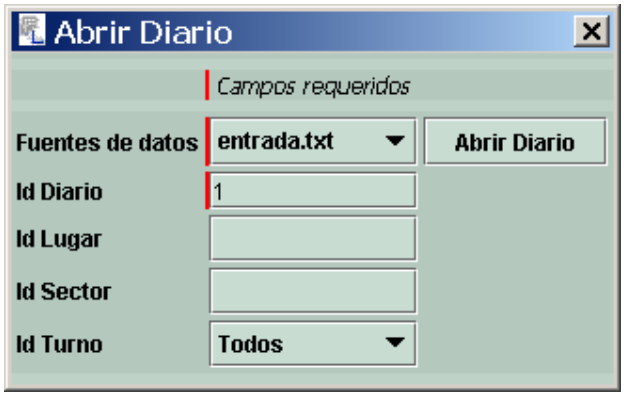


Figura 9-15: Abrir diario.

Al presionar la opción de abrir diario, se despliega el formulario para ingresar los datos del diario a abrir: archivo o fuente de datos en la que se encuentra, identificador del diario y otros datos en caso que se desee cargar solo parte del diario, como muestra la Figura 9-15.

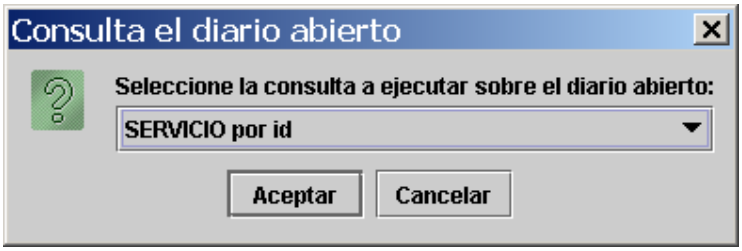


Figura 9-16: Realizar consulta.

Al abrir un diario, éste se carga en memoria y se pueden realizar consultas sobre el mismo, como muestra la Figura 9-16.

Los resultados de dicha consulta se muestran en el árbol de consultas, en formato de tabla y en representación gráfica de Gantt, como muestra la Figura 9-17.

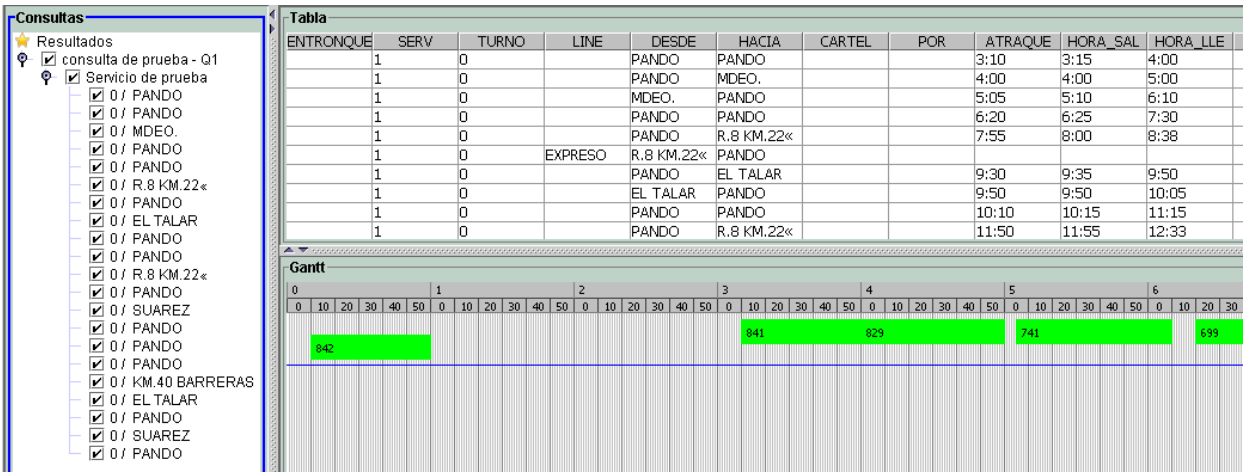


Figura 9-17: Resultados de la consulta.

Figura 9-18: Armar nuevo servicio.

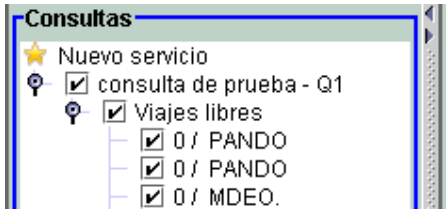


Figura 9-19: Resultados de la consulta para armar nuevo servicio.

Al tener un diario abierto podemos también realizar modificaciones sobre el mismo. Al seleccionar la opción de armado de servicio, en el árbol de consulta se despliega un nodo con el nuevo servicio (vacío) y se habilitan opciones que permiten agregar viajes al mismo, como muestra la Figura 9-18.

Al realizar consultas, obtenemos viajes que podemos incorporar al nuevo servicio, como muestra la Figura 9-19. Durante el armado se realizan chequeos de factibilidad, para asegurarse que los viajes incorporados a la secuencia sean compatibles y puedan formar un servicio que se pueda realizar. Una vez terminado el armado, el diario se puede guardar como un nuevo diario con el nuevo servicio incorporado.

FingLab permite la comparación de costos entre dos soluciones. Primero se calcula el costo de una solución y se guarda temporalmente. Luego se calcula el costo de otra solución y se guarda también temporalmente. El formulario de costos permite comparar uno a uno los valores calculados para cada solución y que el usuario pueda visualizar qué datos subieron, bajaron o quedaron en el mismo valor, como muestra la Figura 9-20.

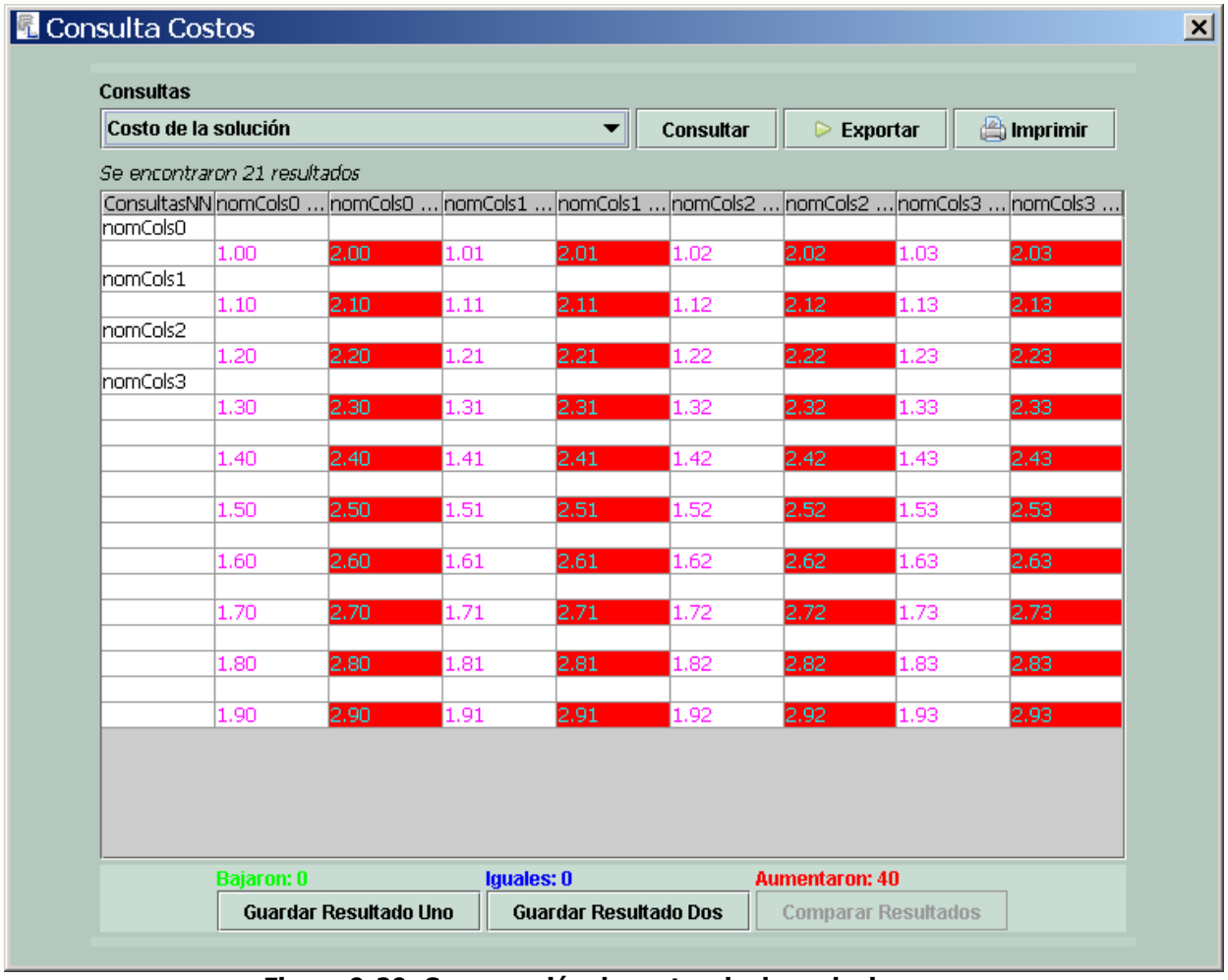
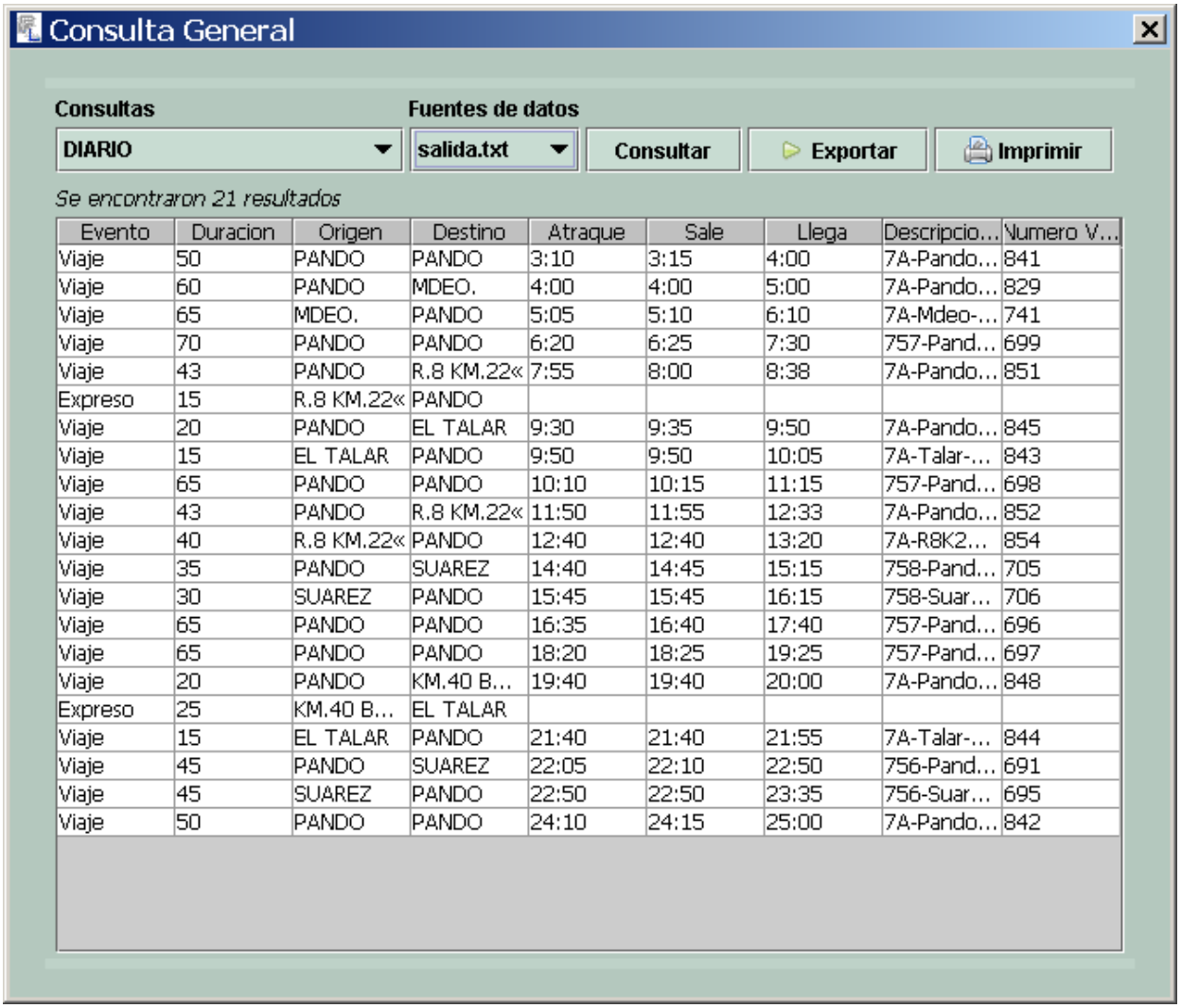


Figura 9-20: Comparación de costos de dos soluciones.

Un diario también puede ser exportado para su utilización con otros sistemas. En la Figura 9-21 se muestra un diario simple, en formato tabular.



Evento	Duracion	Origen	Destino	Atraque	Sale	Llega	Descripcio...	Numero V...
Viaje	50	PANDO	PANDO	3:10	3:15	4:00	7A-Pando...	841
Viaje	60	PANDO	MDEO.	4:00	4:00	5:00	7A-Pando...	829
Viaje	65	MDEO.	PANDO	5:05	5:10	6:10	7A-Mdeo-...	741
Viaje	70	PANDO	PANDO	6:20	6:25	7:30	757-Pand...	699
Viaje	43	PANDO	R.8 KM.22«	7:55	8:00	8:38	7A-Pando...	851
Expreso	15	R.8 KM.22«	PANDO					
Viaje	20	PANDO	EL TALAR	9:30	9:35	9:50	7A-Pando...	845
Viaje	15	EL TALAR	PANDO	9:50	9:50	10:05	7A-Talar-...	843
Viaje	65	PANDO	PANDO	10:10	10:15	11:15	757-Pand...	698
Viaje	43	PANDO	R.8 KM.22«	11:50	11:55	12:33	7A-Pando...	852
Viaje	40	R.8 KM.22«	PANDO	12:40	12:40	13:20	7A-R8K2...	854
Viaje	35	PANDO	SUAREZ	14:40	14:45	15:15	758-Pand...	705
Viaje	30	SUAREZ	PANDO	15:45	15:45	16:15	758-Suar...	706
Viaje	65	PANDO	PANDO	16:35	16:40	17:40	757-Pand...	696
Viaje	65	PANDO	PANDO	18:20	18:25	19:25	757-Pand...	697
Viaje	20	PANDO	KM.40 B...	19:40	19:40	20:00	7A-Pando...	848
Expreso	25	KM.40 B...	EL TALAR					
Viaje	15	EL TALAR	PANDO	21:40	21:40	21:55	7A-Talar-...	844
Viaje	45	PANDO	SUAREZ	22:05	22:10	22:50	756-Pand...	691
Viaje	45	SUAREZ	PANDO	22:50	22:50	23:35	756-Suar...	695
Viaje	50	PANDO	PANDO	24:10	24:15	25:00	7A-Pando...	842

Figura 9-21: Consulta de un diario.

Al presionar la opción Exportar, se despliega otro formulario, en el cual se puede especificar el mapeo a una estructura diferente a la que está guardado el diario. En la Figura 9-22 se muestra como las columnas que se leen desde un archivo de texto plano, se pueden mapear a columnas en un archivo dbf que puede ser leido con otro sistema. De esta manera, FingLab permite la interoperabilidad con otros sistemas, aún cuando los formatos que manejan cada uno son diferentes.

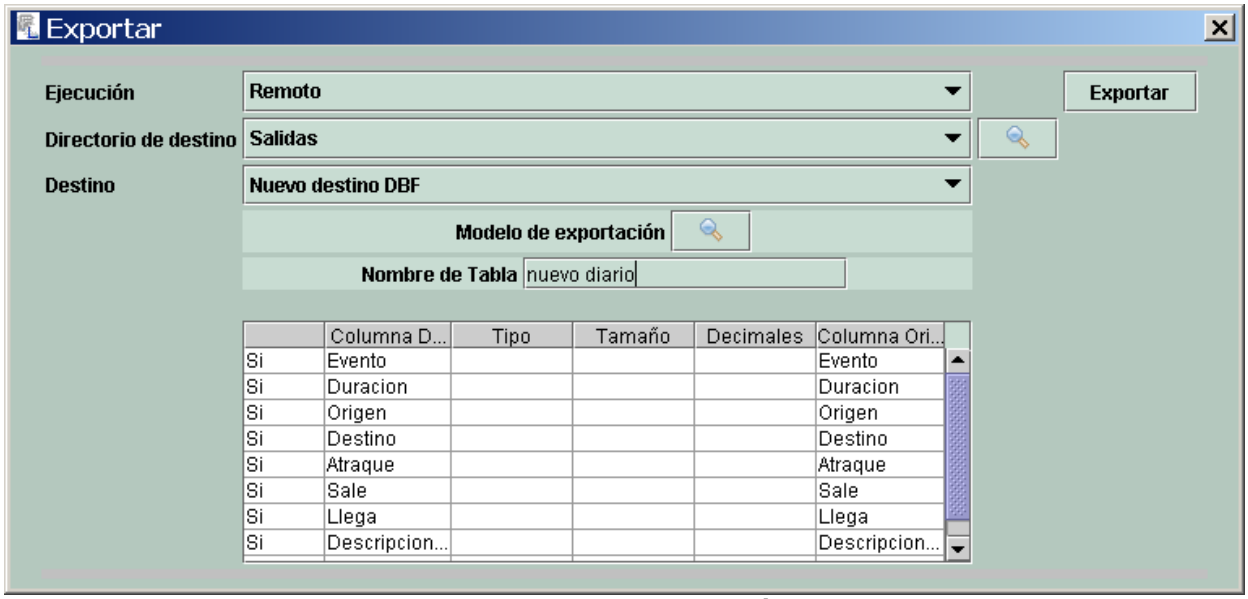


Figura 9-22: Mapeo y exportación de datos.

Nota: se hicieron entrega a los tutores de distribuciones del subsistema de FingLab que permite el mapeo y exportación de soluciones a otros formatos para que pudieran utilizarlo mientras FingLab estaba todavía en desarrollo.

10 Conclusiones y trabajo futuro

10.1 Conclusiones

FingLab permite la experimentación con algoritmos de ordenamiento de vehículos y tripulación y el análisis de sus resultados con una interfaz amigable, personalizable y extendible. Es su capacidad de ser extendido agregando módulos que pueden ser implementados aparte, lo que le permite tener flexibilidad en cuanto al formato de las soluciones (que puede variar constantemente si se está experimentando) y al cálculo de costos y factibilidad de las soluciones (que pueden variar periódicamente). También la capacidad de mapear resultados a otros formatos fomenta su interacción con otros sistemas, sin tener que utilizar un componente intermedio que transforme las soluciones de un formato a otro.

El diseño de FingLab está basado no solo en los usuarios actuales, que incluye investigadores (del área de investigación con algoritmos) y planificadores (del área de transporte), sino también en los potenciales (usuarios que hablan cualquier idioma, investigan con cualquier otro tipo de algoritmos, que tienen diferentes formatos de almacenamiento de datos, etc.). Esto lo hace un sistema que puede ofrecer competencia en el mercado. El poder intercambiar visualizaciones lo hace una herramienta de propósito múltiple para el área de investigación, ya que de esta manera se puede trabajar con diferentes tipos de algoritmos, que requieran visualizaciones que no necesariamente sean de Diagrama de Gantt.

FingLab se adapta a las necesidades de distintos tipos de usuario y acompaña su ciclo de trabajo, ya sea, experimentando con algoritmos o planificando los diarios. Presenta la flexibilidad, capacidad de personalización e interacción con otros sistemas, que otras herramientas en el mercado no ofrecen actualmente. El hecho que tenga dos perfiles de usuarios, hace que fomente el trabajo cooperativo y la retroalimentación entre desarrolladores (de algoritmos) y clientes finales (planificadores).

10.2 Trabajo Futuro

Como trabajo a futuro queda planteado seguir experimentando con FingLab, agregando diferentes módulos de costos y factibilidad para verificar si las interfaces de software definidas para esos módulos están completas o si es necesario agregar más funciones a las mismas. También queda abierta la posibilidad de experimentar con otros tipos de algoritmos que requieran diferentes tipos de visualizaciones para sus resultados y analizar qué componentes se deberían agregar al código base de FingLab para soportarlos. En la documentación técnica y de usuario de FingLab se especifican aquellas funcionalidades que no se pudieron implementar o que tienen algún bug que debe resolverse en el futuro.

La versión actual de FingLab presenta un módulo de factibilidad que realiza chequeos básicos para el armado de servicios como que los viajes no se superpongan y se que pueda realizar toda la secuencia sin problemas. A esto hay que sumarle las reglas laborales que también influyen en cuanto a si una secuencia de viajes es factible o no. Dichas reglas laborales han cambiado, por lo que se plantea implementar a futuro las reglas que estén vigentes.

Para la versión actual de FingLab se realizó un estudio de todas las interacciones con el gráfico que son deseables. Debido a problemas de implementación no se logró incorporar todas las interacciones a la versión actual, por lo que se plantea la posibilidad de agregarlas a futuro. La información detallada sobre todo lo que se desea lograr con la visualización gráfica se encuentra en el anexo de Prototipos.

El trabajo futuro con diferentes algoritmos es lo que permitirá el sucesivo desarrollo de FingLab como una herramienta de apoyo tanto a la investigación, como a la planificación.

11 Agradecimientos

Deseamos agradecer la colaboración de la empresa de transporte COPSA, que sirvió como ejemplo de usuario planificador y cuya retroalimentación fue utilizada en la implementación y mejora del sistema FingLab. También nos facilitaron un sistema que actualmente utilizan para armar los diarios de servicios manualmente y que nos sirvió de ejemplo como sistema externo con el cual FingLab puede interactuar, importando y exportando diarios de un sistema a otro.

Anexos

A Anexo: Estado del arte de las interfaces

A.1 Introducción

En este anexo se realiza un estudio del **estado del arte de las interfaces** y la **interacción persona-computadora** o interacción hombre-computadora (o **HCI** del inglés "Human-Computer Interaction") y del cual se desprenden los conceptos a aplicar en la implementación del proyecto. El área de HCI es muy amplia y abarca muchas disciplinas, por lo que este estudio se enfoca en aquellos aspectos relacionados directamente con nuestro proyecto.

A.2 Definiciones y propósitos

No existe una definición concreta para el conjunto de conceptos incluidos en HCI. En la Tabla A-1 se muestran algunas aproximaciones a la definición de HCI, así como sus objetivos, los cuales también contribuyen a definirla (definiciones y conceptos extraídos de [32, 40]).

¿Qué es?	¿Para qué sirve?
"el estudio de las computadoras y los mayores fenómenos que las rodean" según los autores Newell, Perlis, y Simon (1967)	• Estudiar el hardware, el software y su impacto en la interacción. • Estudiar las personas y sus capacidades. • Analizar la tarea del sistema versus las necesidades del usuario. • Implementar un diseño centrado en el usuario. • Analizar el impacto organizacional.
"el estudio sistemático de los procesos algorítmicos que describen y transforman la información: su teoría, análisis, diseño, eficiencia, implementación y aplicación" según los autores Denning, et al (1988)	Para implementar sistemas que sean usables. La norma ISO 9241 define la usabilidad como: "el rango en el cual un producto puede ser usado por unos usuarios específicos para alcanzar ciertas metas especificadas con efectividad, eficiencia y satisfacción en un contexto de uso especificado."
La interacción hombre ordenador es la disciplina relacionada con el diseño, evaluación e implementación de sistemas informáticos interactivos para la utilización humana, estudiando todos los fenómenos que lo rodean.	Desde el punto de vista de la ingeniería de software, la GUI es la cara visible de todo sistema. Es la parte con la que el usuario entra en contacto y por lo tanto el principal medio de comunicación con el exterior. Como medio de comunicación, su objetivo es desarrollar un mensaje que pueda ser transmitido y recibido a tiempo y de manera exacta. El diseño de los sistemas informáticos debe apoyar a los usuarios para que puedan realizar su tarea en forma segura y eficiente.
"La interacción hombre ordenador se relaciona con el diseño de sistemas informáticos para que las personas puedan llevar a cabo sus actividades productivamente y con seguridad." según el autor Preece (1994)	Es importante la comunicación entre los usuarios y los diseñadores. Los primeros son quienes definen la misión del sistema, determinando claramente los objetivos. Los diseñadores deben proponer alternativas de diseño para consideración de los usuarios, teniendo en cuenta tanto las metas específicas de la tarea y los factores humanos. También deben entender los intereses de los usuarios, establecer lazos de empatía con ellos y guiarse por la realimentación recibida para buscar nuevas alternativas. Los diseños acertados tendrán una comprensión cuidadosa y exhaustiva de la comunidad de usuarios y de las tareas que se deben realizar, ya que un diseño que es apropiado para una comunidad de usuarios puede no serlo para otra.
HCI estudia las relaciones que se establecen entre las personas y las computadoras, por lo tanto será	Con el fin de tener un concepto más aproximado sobre el campo de la HCI contemplamos en que está

necesaria la participación de expertos en áreas como psicología, diseño de sistemas para usuarios, ergonometría e ingeniería entre otras. Al momento de diseñar un sistema de HCI es necesario analizar las partes que involucran la interacción y sus características.	especializado: • Unión de las tareas de los humanos con las máquinas. • Capacidades humanas para utilizar las máquinas (incluyendo la capacidad de entender las interfaces) • Algoritmos y programas de la interfaz en sí. • Conceptos de ingeniería que se plantean a la hora de diseñar y construir interfaces. • El proceso de especificación, diseño, e implementación de la interfaz. • Sacrificios del diseño.
Disciplina que estudia el intercambio de información entre las personas y los computadores. Ésta se encarga del diseño, evaluación e implementación de los aparatos tecnológicos interactivos, estudiando el mayor número de casos que les pueda llegar a afectar. El objetivo es que el intercambio sea más eficiente: minimizar errores, incrementar la satisfacción, disminuir la frustración y, en definitiva, hacer más productivas las tareas que rodean a las personas y los computadores.	Como la HCI estudia a un humano y a una computadora en interacción, deriva del conocimiento de soporte de ambos lados, el humano y la máquina. Del lado de la máquina, técnicas en gráficos de computadora, sistemas operativos, lenguajes de programación y ambientes de desarrollo son relevantes. Del lado del humano, teoría de la comunicación, disciplinas de diseño industrial y gráfico, lingüística, ciencias sociales, psicología cognitiva y conducta humana, métodos de diseño e ingeniería son relevantes.

Tabla A-1: Definiciones y objetivos de HCI.

A.3 Breve reseña histórica

El área de HCI ha estado en constante evolución desde su nacimiento en los años 60. Existen muchos estudios detallados sobre las diferentes herramientas creadas, así como bibliografía, incluyendo líneas de tiempo con cada hecho importante, así como fotos de los diferentes sistemas lanzadas al mercado [3, 42, 56].

Los gráficos por computadora nacieron de la utilización del tubo de rayos catódicos (pantalla) y de las primeras utilizaciones de pen drives. Desde entonces se han realizado numerosas investigaciones que han cambiado los objetivos de las interfaces y de las computadoras en sí, desde solamente hacer cálculos a facilitar su uso para usuarios no expertos. Este cambio de objetivos ha llevado a enfocarse en el usuario y cómo acercar las funcionalidades del computador a él mediante las interfaces de una manera natural e intuitiva.

La Figura A-1 muestra un gráfico de línea de tiempo, muy resumido, en el cual representamos los hechos más significativos que ocurrieron en esta disciplina desde 1945 hasta la fecha (información extraída de [32, 40] y compilada en formato de gráfico).

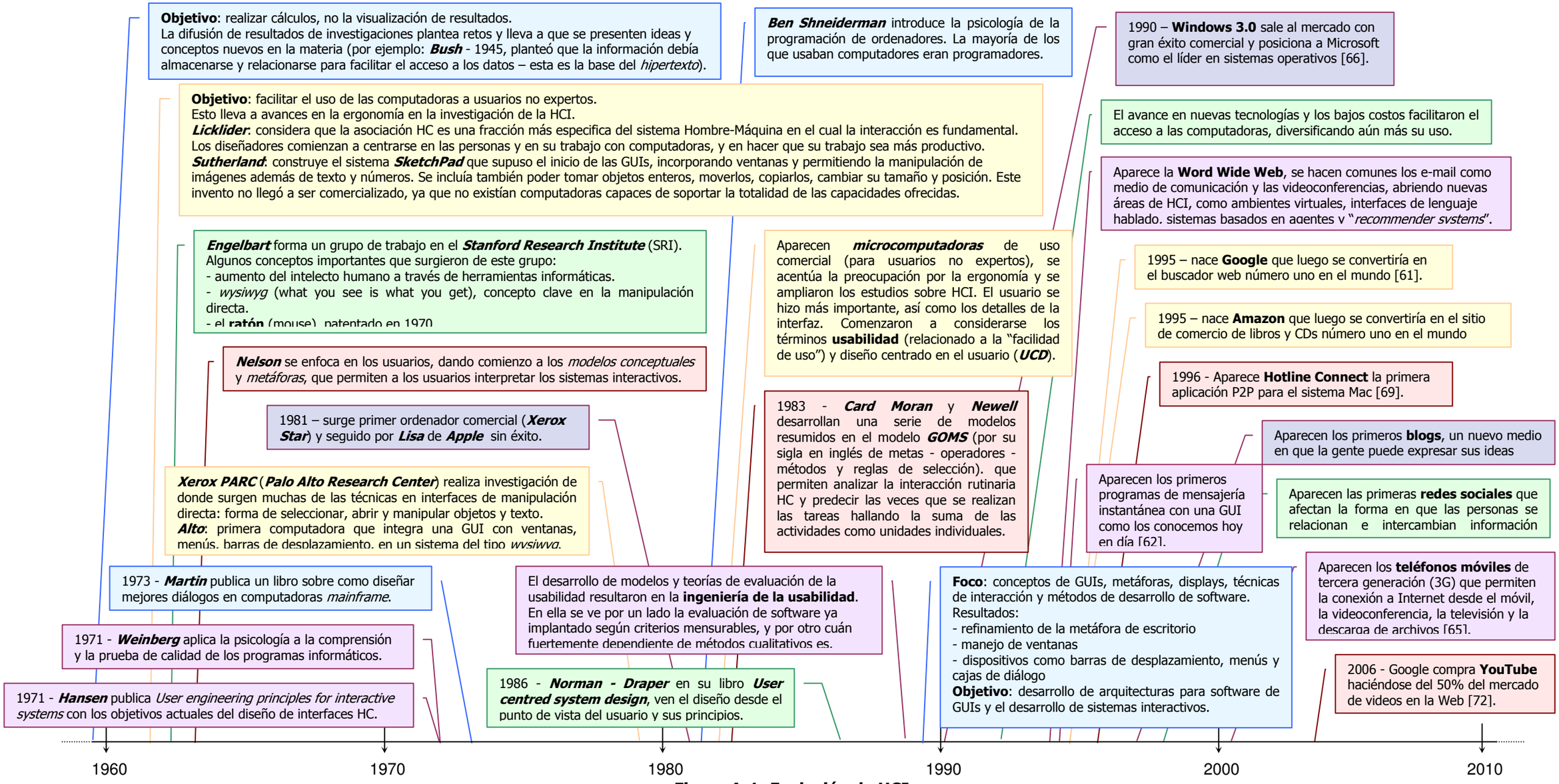


Figura A-1: Evolución de HCI.

A.3.1 Historia reciente

La bibliografía disponible sobre la historia de HCI en general cita hechos que datan de a lo sumo hace unos 10 o 15 años. Pero en estos años, con el nacimiento de la World Wide Web, hemos visto muchos avances en el área que, aunque no son reconocidos aún como parte de la historia (por ser muy recientes), han afectado las interfaces y la forma en que las personas interactúan con las computadoras. A continuación se enumeran algunos hechos que merecen ser recalcados como parte de la historia reciente en HCI, por la forma en que han cambiado los hábitos de las personas y cómo se han convertido en parte de la interacción cotidiana con los sistemas computacionales, al punto que casi ni notamos el impacto que han tenido.

- Con la aparición de la Word Wide Web, todos los conceptos de usabilidad aplicados a sistemas se trasladan y adaptan a las páginas Web. También se deben crear nuevos paradigmas para estas interfaces debido a su particular dinamismo (la actualización de la información) comparado con los sistemas de software habituales.
- El email, los servicios de mensajería instantánea y las videoconferencias se convierten en medios de comunicación imprescindibles tanto en el trabajo como en la vida personal. En el área laboral, estos medios fomentan el outsourcing o subcontratación, especialmente a distancia, donde se puede tener una respuesta inmediata y hacer seguimiento de los avances de los proyectos, así como permiten a trabajadores interactuar como si estuvieran presentes en el mismo lugar, aún estando en países diferentes. En la vida personal, las personas pueden mantener contacto con sus seres queridos de una forma más inmediata, manteniéndose al tanto unos de otros, dando una sensación de cercanía y comunidad que antes solo podía ser realizada por teléfono [62].
- En 1995 nace Google que luego se convertiría en el motor de búsqueda número uno en el mundo, gracias a su velocidad de respuesta y calidad de resultados. Hoy en día, la empresa abarca un gran número de aplicaciones que incluyen correo electrónico, mensajería instantánea, compartir documentos online y sistemas de información geográfica entre otros. La influencia de Google en los hábitos de los usuarios al consultar Internet es indiscutible e incluso el término "google" fue agregado al diccionario de la lengua inglesa para referirse a "buscar información en Internet utilizando un motor de búsqueda, especialmente Google.com" [61].
- En 1995 aparece Amazon, con la misión de crear una librería online. La estrategia de ofrecer libros a bajo costo y de envío rápido lo convirtió en el sitio de comercio de libros número uno en el mundo y actualmente abarca CDs, DVDs, software y videojuegos entre otros productos. La forma de buscar productos pudiendo seleccionar categorías y acotar los resultados hace que los usuarios se inclinen más a realizar sus compras online que en librerías reales. Los hábitos de consumo y compra de las personas fueron modificados por sitios como Amazon y también ha influenciado la forma en que se investigan los mismos para ser utilizados con propósitos de marketing. Es así que aparecen los agentes de recomendación, software que permite realizar sugerencias a los usuarios según el producto que están visualizando o los productos que el usuario ha comprado previa o recientemente. Las personas no se encuentran solamente utilizando las interfaces para buscar información, sino también para realizar tareas en su vida personal (como buscar un buen libro, una buena película acorde a sus gustos). La comodidad de hacerlo desde sus casas y la variedad en los catálogos disponibles hace que la interacción con estos sistemas sea preferible antes que ir a una librería o videoclub [59].
- En 1995 Ebay se presenta como una red de subastas donde los usuarios pueden poner a la venta productos nuevos o usados a precios fijos o por la modalidad de subasta. De la misma forma que Amazon, Ebay influencia la forma de consumo de los usuarios, especialmente para acceder a productos por precios módicos o que directamente no están dentro del área física en que el usuario se encuentra.
- Las compañías de tarjetas de crédito, por ejemplo, deben poner a su disposición formas de pago online gracias al éxito de las compras online. Debido a esta necesidad de realizar estos pagos entre usuarios aparecen compañías como PayPal [70].
- Los éxitos comerciales de Amazon e Ebay hacen que las empresas en general comiencen a tener sus propios sitios Web. La forma de presentar sus catálogos ofreciendo sus servicios y productos y acotarlos según determinados criterios, así como las publicidades y recomendaciones que se

insertan en las páginas, se convierte en una forma standard de interacción para los usuarios que realizan sus compras o contrataciones online.

- En 1996 aparece Hotline Connect la primera aplicación P2P para el sistema Mac [69]. Pero no es sino hasta el año 2000, cuando la conexión a Internet sobre todo en hogares se hace común, más económica y de mayor velocidad, que el servicio de distribución de archivos de música (en formato MP3) Napster alcanzó una alta popularidad. Dicha popularidad fue en parte a una demanda entablada por empresas discográficas que pusieron en discusión el tema de los derechos de autor para materiales en formato electrónico. Este tipo de servicio también cambió los hábitos de los usuarios a la hora de realizar sus compras (prefieren descargar una canción en vez de comprar todo un CD) y también cambió la forma de marketing de las empresas al ofrecer sus productos como música y películas en formato electrónico. Si bien los servidores Napster fueron cerrados por orden judicial, la difusión de este caso hizo que usuarios no solo migraran, sino que también adoptaran el uso de otros sistemas de intercambio de archivos existentes. El bajar archivos con estos sistemas se ha hecho más común y se ha transferido al negocio del cine a medida que el ancho de banda de las conexiones aumentara y permitiera la transferencia de archivos más grandes [68].
- En 1996 se crea el puerto USB con el propósito de eliminar la necesidad de adquirir tarjetas separadas para conectar periféricos y mejorar la capacidad de reconocimiento automático (plug-and-play). Poco a poco este tipo de puerto fue desplazando a otros y para el caso de los pendrives o dispositivos de almacenamiento portables es un standard. Hoy en día son los puertos más reconocidos por todos los usuarios por su facilidad de conexión [76].
- Aparecen abreviaciones y siglas como resultado del uso de correo electrónico, mensajería instantánea y mensajes de texto. Esto ha generado la creación de un nuevo lenguaje extendido y cambios en los hábitos (Las personas comienzan a preferir enviar un mensaje de texto en vez de llamar directamente por teléfono). También se utilizan emoticones o imágenes que representan una determinada emoción que se entrelaza con el lenguaje escrito.
- A fines de la década de los 90, surgieron las bitácoras o diarios online (que luego se conocerían por el nombre de "blog"), sitios web donde los usuarios podían escribir sobre su vida, sus opiniones y sus creencias para compartirlas con todo el mundo. A medida que las herramientas de software para mantener y publicar artículos de este estilo avanzaron, los blogs se hicieron más comunes y ya en 2001 existían varios que contaban con popularidad. Hoy en día es un servicio muy popular y así como los sitios web, se han convertido en herramientas que las empresas utilizan para mantener contacto con los usuarios. En ciertas partes del mundo donde la libertad de expresión es más limitada, se han convertido en una forma fácil de difundir ideas y crear comunidades virtuales de pensamiento. Algunas de las entradas de estos blogs pueden incluirse fácilmente en sitios Web, lo cual ha cambiado la forma en que se actualiza el contenido de las páginas [64]. En el año 2009, Twitter saltó a la fama con su modalidad de microblogging, donde los artículos en vez de tener un largo ilimitado, tiene un largo acotado de poco más de 100 caracteres. Al igual que los mensajes de texto, el microblogging adopta abreviaciones para expresar palabras o incluso ideas completas y conceptos en pocos caracteres [71].
- En 2005 se crea YouTube, un sitio en el que los usuarios pueden subir y compartir videos. Google ya había intentado poner en marcha un proyecto similar con GoogleVideo, pero dado el éxito de YouTube, Google lo compra en 2006 haciéndose del 50% del mercado de videos en la Web [72]. De esta manera los sitios ya no tienen por qué tener sus videos en el mismo servidor que sus páginas, sino que los suben a YouTube y luego en sus página incluyen el código que permite introducir el video con su correspondiente barra de control. Esta interfaz para videos se ha hecho un standard para toda Internet.
- En 2006 las redes sociales, sitios destinados a mantener en contacto unos usuarios con otros (según su situación geográfica, académica o laboral) y que ya habían tenido sus inicios a finales de la década de los 90, llegan a su punto cumbre cuando MySpace [67] y Facebook [60] se convierten en fenómenos contando con millones de usuarios alrededor del mundo. La cantidad de personas que tienen su perfil en dichas redes es la fortaleza de estos sitios y este fue el inicio de una nueva era que cambió las interfaces, ya que actualmente todos los sitios Web, también tienen cuentas en todas las redes sociales para abrirse paso y acceder a nuevos usuarios. Hoy en día las páginas Web cuentan con una barra que permite agregar el sitio como un amigo al perfil de cualquier red social,

así como despliegan en la misma algunas fotos de los amigos que tienen en cada red (como muestra la Figura A-2 –imágenes extraídas de www.facebook.com y www.boxofficemojo.com -).

Figura A-2: Widgets para redes sociales y sitios de bookmarking.

Esto permite más el contacto directo y personal entre las empresas y sus clientes y ayudan a conocer más la opinión de los usuarios acerca de cualquier tema y al instante. El impacto es tal que se estima que muchos trabajadores pasan en promedio entre 45 y 90 minutos diarios en su horario de trabajo, navegando por las redes sociales según CNN y revisarlos se ha convertido en un hábito como revisar el correo electrónico.

- Los aparatos por los cuales las personas utilizan software no solamente incluye computadores, sino también teléfonos que permiten navegar por Internet y enviar e-mails. De esta manera los sitios Web deben tener una versión que pueda ser navegada desde un computador, y otra para cuando se accede desde un teléfono móvil. Hay una gran variedad de productos disponibles en el mercado de donde el usuario puede elegir (con teclado incluido en el aparato o con teclado en pantalla tipo touch-screen o táctil). La forma de interactuar puede ser elegida por el usuario y se crea software especialmente para abastecer este mercado [65].

A.3.2 El futuro próximo

Hay ciertas tendencias que parecen estar marcando la pauta en cuanto a lo que se refiere a interfaces:

- Gracias al éxito en los cines de películas en 3D, la venta de videojuegos en 3D y el desarrollo de la tecnología visual, las grandes empresas están implementando sistemas que soporten este tipo de

visualización. Para ello se requiere desarrollo tanto de hardware como de software, pero se han presentado numerosos productos que ya cuentan con la capacidad de soportar 3D [58].

- Los libros electrónicos (o eBooks) no son algo nuevo, los primeros ya se habían presentado en el 2001, pero recién ahora las empresas parecen estar dedicadas en masa al desarrollo de productos que permitan visualizar simplemente archivos en formato pdf que se pueden cargar desde un pendrive. En general consisten solamente de una pantalla liviana con un puerto usb al que se pueden cargar los archivos. Algunos incluso permiten por medio de tecnología táctil: subrayar, tachar y resaltar. Se presentan como una alternativa ecológica, liviana y portable frente a los libros de papel. Sin embargo, al igual que los sistemas P2P estos productos están causando controversia por el daño que le causaría a los derechos de autor de los libros si las copias electrónicas de los mismos se distribuyen gratuitamente [74].
- Luego del éxito del Iphone de la compañía Apple [65], el cual está basado completamente en la tecnología táctil (o en inglés touch-screen), esta tecnología es la que parece imponerse en los computadores futuros. Además de computadores, se cuenta la presentación del Ipad de Apple en enero de 2010 [73], un producto intermedio entre un computador y una agenda personal que permite realizar tareas básicas como acceder a Internet, visualizar documentos y tomar notas. Existe una gran tendencia por parte de las principales compañías de electrónica de proveer de hardware y software que permita acceder a todas las funcionalidades tocando el punto de la pantalla donde se desea hacer click e ingresar texto con un teclado virtual también en pantalla. De esta manera se tiende a sustituir el teclado y el Mouse (periféricos físicos) por componentes virtuales, estrechando aún más la conexión entre el usuario y la tarea a realizar.

A.4 Componentes de la HCI

Existen tres componentes de la HCI, cada uno con sus características, fortalezas y debilidades, y formas de percibir y transmitir información [4] como muestra la Tabla A-2.

Personas		Interacción	Computadoras	
Capacidades	- Creatividad. - Iniciativa. - Manejo de excepciones. - Habilidad para aprender de la experiencia. - Manejo adecuado de problemas mal definidos. - Buenas habilidades motoras. - Juicio. - Sentido de ética y responsabilidad. - Flexibilidad y adaptabilidad.	- Diseño del sistema centrado en el usuario. - Usabilidad. - Análisis. - Diseño. - Desarrollo. - Evaluación. - Principios. - Metodologías. - Patrones. - Prototipos. - Metáforas. - Manipulación directa.	- Precisión y Repetitividad. - Rapidez y exactitud en los cálculos. - Incansabilidad. - Objetividad. - Paciencia. - Robustez física. - Enorme potencia gráfica y sonora.	*Capacidades*
Forma de interactuar	Capacidades físicas. Capacidades cognitivas y percepción.		Dispositivos de interacción.	*Forma de interactuar*
Otros factores	Factores ergonómicos. Personalidad. Edad. Discapacidades. Cultura.		Dispositivos especiales. Software especial.	*Otros factores*

Tabla A-2: Componentes de HCI.

A.4.1 Las personas

Hay que tener en cuenta que el ser humano tiene una capacidad limitada de procesar información, lo cual es muy importante considerar al hacer el diseño (por ello, se debe tener cuidado en cuanto a la cantidad de información que se despliega simultáneamente) [41]. Nos podemos comunicar a través de cuatro **canales de entrada/salida**: visión, audición, tacto y movimiento, por lo cual es necesario considerar como funcionan los sistemas sensoriales humanos. Por mucho tiempo, las interfaces presentaban la información sólo en formato visual y en algunos casos también en formato sonoro, por lo que el sentido de la vista ha sido el más estudiado en HCI seguido por el sentido auditivo. Actualmente, las interfaces presentan información en otros formatos físicos y los usuarios reciben esta información a través de los otros sentidos. Existen interfaces virtuales en donde se explora el canal auditivo y el olfativo, y la realidad virtual exige la exploración de todos los sentidos por igual.

En el libro de presentación de GOMS, de Card, Moran y Newell los canales de entrada que se consideraban eran solamente el ojo y el oído [57]. GOMS (del inglés "Goals, Operators, Methods, and Selection rules" o en español "metas, operadores, métodos y reglas de selección") son métodos cuantitativos, que nos ayudan a mediar y a argumentar con números sobre las bondades o debilidades de una interfaz. permiten conocer, cuánto tiempo le llevará a un trabajador experimentado, a desarrollar una operación particular cuando este utiliza una determinada interfaz (ingresar datos por teclado, hacer click en un botón, seleccionar en un menú, etc.). Así como GOMS existen otros métodos para calcular la velocidad de respuesta o la dificultad de una persona en realizar una tarea. La ley de Fitts es un método utilizado para calcular el índice de dificultad de un movimiento en base a la distancia que debe cubrir el movimiento y la tolerancia de la región dentro de la cual

el movimiento debe terminar. La ley de Hicks es otro método que permite calcular el tiempo de realizar un movimiento cuando se debe elegir entre varias alternativas.

La información recibida se almacena en la memoria sensorial, la memoria a corto plazo y la memoria a largo plazo. Una vez que recibimos la información, ésta es procesada a través del razonamiento y de habilidades adquiridas, como por ejemplo el hecho de poder resolver problemas o el detectar errores. Todo este proceso afectará el estado emocional del usuario, dado que influye directamente sobre las capacidades de una persona. Además, un hecho que no se puede pasar por alto es que todos los usuarios tendrán habilidades comunes, pero habrá otras que variarán según la persona.

La Psicología Cognitiva estudia los **procesos perceptivos** (aquellos que son disparados por los sentidos) como: la atención, el lenguaje, el aprendizaje, la solución de problemas y la toma de decisiones, etc. Estudia el modo en que las personas obtienen información y la procesan (las formas, los colores, la perspectiva) para en un ambiente lleno de estímulos ser capaz de discriminar y centrar la atención en uno solo de ellos. Explica por qué algunos estímulos se recuerdan más que otros, los límites en nuestra percepción, la capacidad de la memoria y los tipos de memoria (sensorial, de corto y largo plazo) entre otros [16, 49].

Algunos de estos estudios han examinado los siguientes puntos:

- El color y la forma de un objeto y cómo afectan a la vista al procesarlo. El color nos ayuda a determinar como se disponen los objetos en el espacio (distancia), como se mueven y cuales son sus formas. Se han estudiado las combinaciones de colores que facilitan el reconocimiento de objetos (algunas combinaciones de colores pueden provocar ilusiones ópticas o hacer que no se puedan diferenciar los objetos unos de otros). El sistema visual responde de manera diferente a distintos colores y se debe considerar que hay personas que no diferencian los colores (daltónicos), por lo que estos factores deben ser estudiados cuidadosamente. Algunos investigadores consideran en sus estudios el tiempo de respuesta humana al variar los estímulos visuales, o tiempo de adaptarse a mayor o menor brillo [24, 49].
- La capacidad humana para identificar un objeto en un contexto o determinar la velocidad o dirección del movimiento de un punto. Las variaciones del rango espectral y la sensibilidad de las personas [49].
- El impacto del fulgor visual y de los factores que afectan el rendimiento del motor perceptivo, como: sueño, fatiga, carga mental, conocimiento de resultados, etc.[77]
- El texto es una parte importante de la interfaz, aunque predomine la información gráfica, se ha estudiado su tamaño, tipo de letra, mayúsculas, color, sitio, porque todos estos factores influyen en la comprensión de la información [77].
- Las representaciones gráficas deben ser construidas de acuerdo a las necesidades del usuario, que éste las pueda interpretar y que den apoyo al proceso de análisis y soporte de decisiones. Muchas veces es necesario tener varias representaciones gráficas de la misma información dentro un solo sistema [11, 31, 44].
- Es importante proveer de formas de filtrado de información, o mostrar/ocultar datos por demanda, cuando es más de la que el usuario puede analizar.

Se deben considerar en gran medida las capacidades cognitivas y de percepción ya que: son básicas para explicar cómo las personas interactúan con los objetos/máquinas, son vitales para la visualización de información y son indispensables para satisfacer al usuario al manejar un sistema, facilitar el aprendizaje y aumentar el rendimiento.

La ergonomía trata la forma en la que las personas utilizan los objetos, tratando de mejorar la comodidad y eficiencia, haciendo más fácil su uso. En el ámbito informático, la ergonomía trata principalmente conceptos de hardware, como los monitores y teclados, pero aplica también a temas como la legibilidad en pantalla. En el aspecto ergonómico se toman en cuenta datos básicos sobre las dimensiones humanas promedio, llegando a la conclusión de que no existe la imagen de un usuario promedio que permita adecuar todos los dispositivos a una medida fija. Cuando el diseño no puede ser acomodado a una gran parte de la población, se intenta solucionar brindando controles de ajuste (por ejemplo la altura de la silla). También hay personas con discapacidades que deben tener hardware y software especial para poder interactuar con el computador (por

ejemplo: micrófono para poder dar comandos de voz, software que reconozca comandos –o movimientos de ojos para personas en estado parapléjico- y ejecute las acciones indicadas, o software de lectura de pantalla para personas invidentes).

Existen otros factores como los ambientales, que influencian el bienestar de la persona que está trabajando y que también deben tenerse en cuenta en el momento de establecer las posiciones de trabajo: niveles de iluminación, balance de luminosidad y parpadeo, equipos reflectivos, ruidos y vibración, temperatura del aire, movimiento, humedad y equipos de temperatura. El diseño del lugar de trabajo es importante para asegurar la satisfacción del usuario, alto rendimiento, y bajo nivel de error.

Las diferencias en la personalidad pueden provocar alteraciones en la propia comunicación [47]. De esta forma, personas tímidas tendrán un comportamiento cauto y prudente ante una computadora, a diferencia de personas extrovertidas y nerviosas que serán más osadas en su uso. También podemos encontrar diferencias motivadas por el entorno socio–cultural donde se encuentra el usuario, que puede afectar al lenguaje utilizado, expresiones y terminología, modo de trabajar, etc.[49]

A.4.2 El computador

El sistema utilizado puede afectar de diferentes formas al usuario. Los dispositivos de entrada permiten introducir texto, como sería el caso del teclado del computador, el teclado de un teléfono, el habla o bien un escrito a mano; dibujos; selecciones por pantalla, con el ratón por ejemplo. Como dispositivos de salida contaríamos con diversos tipos de pantallas, mayoritariamente aquellas que son de mapas de bits, pantallas de gran tamaño de uso en lugares públicos, etc. A largo plazo se podría contar también con papel digital. Los sistemas de realidad virtual y de visualización con 3D juegan un rol muy importante en el mundo de la interactividad persona-computador. También serán importantes los dispositivos en contacto con el mundo físico, por ejemplo controles físicos, como sensores de temperatura, movimiento, etc. Por otra parte tenemos diferentes tipos de impresoras con sus propias características, fuentes y caracteres. Y también escáneres y aparatos de reconocimiento óptico. Con respecto a la memoria, cuentan con la RAM a corto plazo y discos magnéticos y ópticos a largo plazo. Hay que tener en cuenta que tienen una capacidad limitada con relación directa con el formato del documento o del vídeo. Los métodos de acceso a la memoria pueden suponer una ayuda, sin embargo, en ocasiones, también una traba para el usuario. El último rasgo característico es el procesamiento. El computador tendrá un límite de velocidad en el procesamiento, por otra parte afectará a la velocidad de procesamiento al hecho de utilizar una red de trabajo u otra [49].

Existen básicamente tres grupos de dispositivos: de entrada, de proceso y de salida de información. Los utilizados durante la interacción entre el usuario y la computadora son los de entrada y salida como muestra la Tabla A-3. Estos son relevantes al estudiar los factores ergonómicos y de diseño e influyen en la velocidad de ejecución, precisión, errores y tiempo de aprendizaje.

Entrada		**Salida**	
Dispositivo	**Función**	**Dispositivo**	**Función**
Teclado.	Introducir texto	Monitor.	Desplegar información visual.
- Ratón. - Lápiz óptico. - Pantalla táctil.	- Manipular objetos en pantalla. - Permitir que el usuario centre su atención en la pantalla sin memorizar comandos y evitando errores al digitar.	- Parlantes. - Audífonos.	- Manipular información sonora. - El usuario no necesita fijar la vista en un lugar particular (porque escucha los mensajes del sistema).
Micrófono.	Introducir información sonora.		
Reconocimiento del habla, movimientos de ojos y cabeza.	Ayuda a discapacitados a introducir comandos.		

Tabla A-3: Dispositivos de Entrada y Salida.

Los dispositivos de salida visuales se convirtieron en la mayor fuente de realimentación desde la computadora hacia los usuarios. Se simplifica la interacción con los usuarios, si los dispositivos de salida visuales poseen alta calidad en los aspectos físicos como brillo, contraste entre caracteres y fondo,

resolución, parpadeo, combinación de colores, tamaño de caracteres, etc. Es recomendable que estas características puedan ser configuradas por el usuario: dándole control sobre contraste y brillo, tamaño y tipo de fuente, etc. [49]. También existen formas de simular dispositivos de entrada, como por ejemplo el teclado virtual (como el que muestra la Figura A-3 –imagen extraída del programa "My heritage" http://www.myheritage.es/ -) para introducir texto mediante clicks del Mouse en las teclas de la pantalla. Esto es de mucha utilidad para proveer de un teclado que tenga caracteres especiales de algunos idiomas (como por ejemplo la letra ñ del español) si no se tiene configurado el teclado físico.

Figura A-3: Ejemplo de teclado virtual.

A.4.3 La interacción

La interacción es "el conjunto de procesos, diálogos, y acciones a través de las cuales el usuario emplea e interactúa con una computadora" (Baecker & Buxton, 1987, p. 40). Una forma de aproximarnos al concepto de interacción es con la noción de "look and feel" (que en español significa "cómo se ve y se siente"). El "look" de una interfaz se refiere a la apariencia visual de la misma, mientras que el "feel" se refiere a su aspecto interactivo [52].

Un sistema no solo debe verse bien, sino que el usuario debe sentirse cómodo usándolo, sino la interacción no se desarrolla para el máximo beneficio del usuario o simplemente no ocurre (el usuario deja de usarlo). Es importante que haya una buena comunicación entre usuario y computador, por este motivo la interfaz tiene que estar diseñada pensando en las necesidades del usuario y, al mismo tiempo, manteniendo las funcionalidades que debe proveer. Es de vital importancia este buen entendimiento entre ambas partes dado que sino la interacción no será posible.

La Figura A-4 (imagen extraída de [32]) muestra el ciclo de intercambio de información durante la interacción entre una persona y una computadora y el rol de todos los factores y características mencionadas anteriormente.

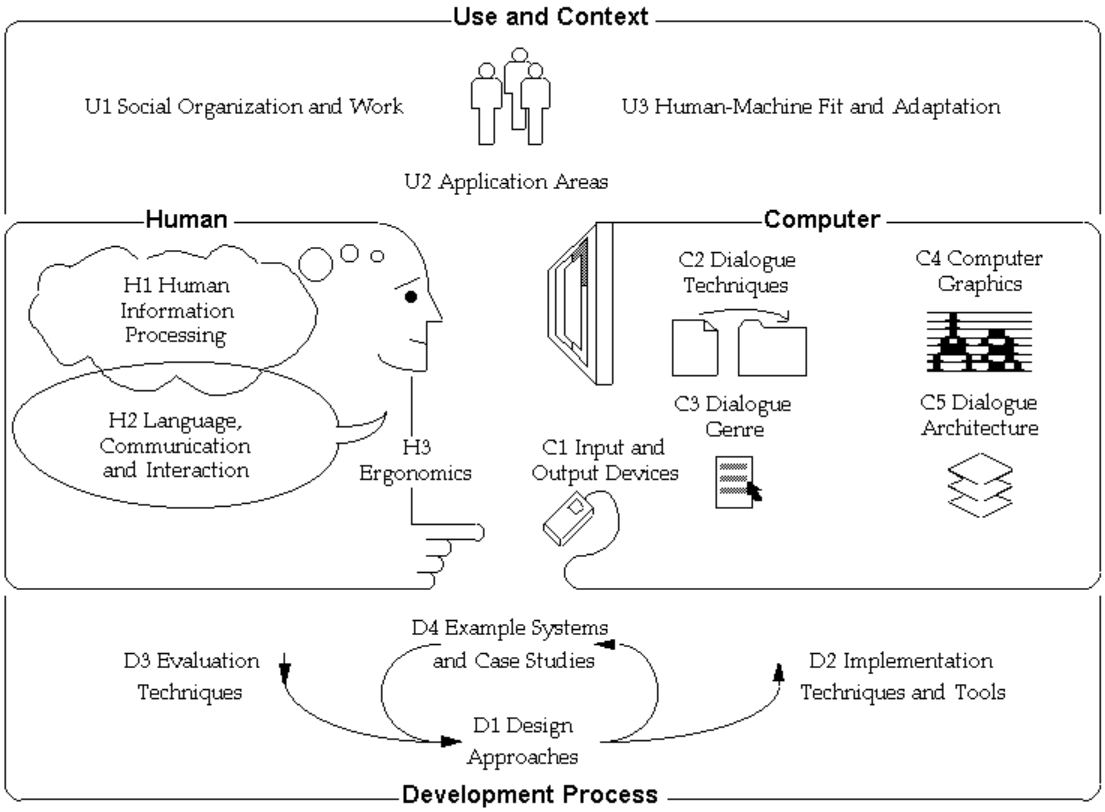


Figura A-4: Proceso de interacción persona-computadora.

Los sistemas efectivos generan una sensación positiva, de éxito, competencia y claridad en el usuario, por lo que el diseñador debe entender completamente a la comunidad de usuarios para la cual realiza el diseño, así como todas las tareas que ésta realiza [36]. De esta manera, los usuarios podrán predecir como reaccionará el sistema en respuesta a cada una de sus acciones. Además, logrará que la interfaz casi desaparezca, permitiendo a los usuarios concentrarse en su trabajo real, sin que la interpretación de la interfaz sea una tarea extra. Es casi imposible generalizar un conjunto de usuarios, especialmente cuando se está implementando un sistema cuyo objetivo es una gran comunidad (como una ciudad, o incluso en el caso de una aplicación Web, el mundo). Sin embargo, existen algunas pautas generales a seguir que pueden ayudar a los diseñadores a lograr un sistema aceptado por los usuarios [49]:

- *Funcionalidad apropiada*: Si el sistema no hace lo que se supone debe hacer, el producto no será aceptado por el usuario, sin importar lo bien que esté hecho el diseño. Por ello, en la recolección de requerimientos se debe ser muy específico e indagar sobre las tareas frecuentes, a las ocasionales y también a las de condiciones de emergencia (categorizar las funcionalidades). Las funcionalidades deben ser distribuidas de forma que no agobien al usuario (que el usuario no deba tener que hacer 30 pasos para ver un resultado) y compliquen aún más el proceso de implementación y verificación.
- *Confiabilidad, disponibilidad, seguridad e integridad de los datos*: Los usuarios siempre desconfían en los sistemas (hay un sentimiento innato de que el sistema va a perder su trabajo). Esto sumado a las malas experiencias de los mismos, puede contribuir al desuso del sistema. El sistema debe tener disponibilidad (que el sistema esté siempre listo para cuando el usuario lo necesite) y que cuando ocurran errores se sepa por qué ocurrieron y que se pueda recuperar de los mismos. Los usuarios deben de sentir que tienen control sobre la aplicación, y no al revés, el sistema da soporte a sus habilidades. También se debe considerar: privacidad, seguridad e integridad de datos y protección contra destrucciones accidentales o maliciosas.
- *Estandarización, consistencia y portabilidad*: Si el sistema tiene una interfaz que tiene características comunes con otras aplicaciones similares (estandarización), el usuario aprende su utilización más rápido y la cantidad de errores que comete en su utilización es menor. La *consistencia* refiere a una secuencia de acciones, términos, unidades, disposiciones, color y tipografía comunes en una

aplicación. Esta consistencia también es extendida para incluir compatibilidad entre aplicaciones y con sistemas no basados en computadoras (registros escritos en papel). La *portabilidad* refiere al potencial de convertir datos y compartir interfaces de usuario a través de distintos ambientes de software y de hardware.

- *Presupuesto y Planificación*: una planificación cuidadosa permite desarrollar el producto en los tiempos establecidos y mantenerse dentro del presupuesto asignado.

Existen factores humanos mensurables y centrales para la evaluación de una interfaz que guían a los diseñadores, no en cuanto a las funcionalidades en sí, sino a su utilización [49]:

- *Tiempo de aprendizaje*: Cuánto le toma a un usuario promedio del grupo aprender el manejo de las funcionalidades más importantes del sistema.
- *Plazo de retención*: Cuánto un usuario puede recordar sobre cómo funciona el sistema luego de una hora, o un día, etc. Esto sirve para medir la curva de aprendizaje, ya que el usuario debe retener cada vez más, cuanto más contacto tenga con el sistema.
- *Velocidad de desempeño*: Cuánto le toma a un usuario promedio del grupo realizar las tareas más importantes con el sistema luego de haber aprendido su uso.
- *Índice de errores de usuario*: Cuáles son los errores más frecuentes al realizar las tareas con el sistema, cuál es la capacidad de recuperación del sistema frente a los mismos y qué mensajes provee el sistema para guiar al usuario a que no los cometa de nuevo o cómo puede proseguir.
- *Satisfacción subjetiva*: Realizar entrevistas con cuestionarios evaluando el nivel de satisfacción de los usuarios con cada aspecto del sistema y recibir sugerencias y comentarios.

Una vez establecidas las metas del sistema, se debe clasificarlas para identificar cuáles son de vital importancia y cuáles tienen menos peso, para así saber cómo enfocar el trabajo de desarrollo. La retroalimentación del usuario y su rol en el desarrollo desde muy temprano debe ser tomado en cuenta. Esta forma de

A.4.4 Diseño del sistema centrado en el usuario

Extraído de [23, 28, 44, 55].

Se entiende por "diseño centrado en el usuario" (del inglés "user-centered design" o su sigla UCD) que los diseñadores ven el sistema desde el punto de vista del usuario: comprenden el contexto de uso, su entorno de trabajo, las tareas que éste realiza y lo involucran activamente en el proceso de desarrollo. En contraste, la idea de "diseño centrado en el uso" plantea que el diseñador sólo necesita enfocarse en conocer las tareas del usuario y por lo tanto el usuario no es involucrado en el proceso de diseño.

Existen diferencias entre un sistema diseñado convencionalmente y uno centrado en el usuario, como se muestra en la Tabla A-4.

Diseño centrado en el uso	Diseño centrado en el usuario
❑ Enfocado a la tecnología. ❑ Enfocado a los componentes. ❑ Cooperación multidisciplinaria limitada. ❑ Enfocado a la arquitectura interna. ❑ No hay especialización en la experiencia de usuario. ❑ Enfocado un poco en la competencia. ❑ El desarrollo es previo a la validación del usuario. ❑ La calidad se ve en los defectos del producto. ❑ Enfocado poco en medidas del usuario. ❑ Enfocado a los clientes actuales.	❑ Enfocado a las necesidades del usuario. ❑ Enfocado a dar soluciones. ❑ Se trabaja en un equipo multidisciplinario. ❑ Enfocado al diseño externo. ❑ Se especializa en la experiencia de usuario. ❑ Enfocado a la competencia. ❑ Se desarrollan los diseños validados por el usuario. ❑ La calidad se ve desde el punto de vista del usuario. ❑ Enfocado principalmente en métricas definidas por los usuarios. ❑ Enfocado a los clientes actuales pero también en los futuros.

Tabla A-4: Características del diseño centrado en el uso y diseño centrado en el usuario.

A.4.4.1 Principios del diseño centrado en el usuario

Estos son los principios en los que se basa un diseño centrado en el usuario [55]:

- *Establecer metas de negocio*: determinar el mercado, los usuarios y la competencia es central.
- *Entender a los usuarios*: los usuarios son la fuerza detrás de todo diseño. Se debe entender su contexto, sus tareas, su ambiente, su preparación académica, etc.
- *Diseñar la experiencia total de usuario*: todo lo que el usuario ve y toca es diseñado en conjunto por un equipo multidisciplinario. El prototipado es una parte fundamental de este punto.
- *Evaluar diseños*: la retroalimentación del usuario es recolectada frecuentemente y es la que marca la dirección del diseño y el desarrollo.
- *Estudiar la competencia*: el diseño competitivo requiere enfocarse en la competencia y sus clientes. Analizar las herramientas similares que hay en el mercado es parte de este punto, para reconocer lo que se puede hacer y lo que hace falta en el mercado.
- *Administración de usuarios*: la retroalimentación de usuarios es integral a los planes del producto, las prioridades y las decisiones a tomar.

A.5 Usabilidad

Extraído de [22, 25, 39, 43].

La norma ISO 9241(como fue mencionado en la introducción) define la usabilidad como: "el rango en el cual un producto puede ser usado por unos usuarios específicos para alcanzar ciertas metas especificadas con efectividad, eficiencia y satisfacción en un contexto de uso especificado." [3]

La usabilidad de un sistema[18], vista como un medio para alcanzar un objetivo, se divide en dos componentes, una que hace referencia a su funcionalidad o utilidad funcional y otra basada en la forma en que los usuarios pueden usar esta funcionalidad. Esta última es la que trataremos en esta sección.

La efectividad es la precisión y la integridad con que los usuarios pueden alcanzar los objetivos que fueron especificados. Esta noción también está relacionada a la facilidad de aprendizaje, la tasa de errores del sistema y la facilidad con que los usuarios recuerdan el sistema.

La eficiencia es entendida como los recursos que se utilizan para lograr la precisión e integridad con que los usuarios alcanzan los objetivos especificados.

Por satisfacción se entenderá la ausencia de incomodidad y la actitud positiva del usuario al utilizar el producto. Este factor es subjetivo.

A.6 Proceso de desarrollo de una aplicación interactiva.

Extraído de [24, 40, 43, 44, 49]

El proceso de desarrollo de software consiste básicamente en cuatro pasos: análisis (donde se determinan los requerimientos del sistema, es decir, qué se desea que el sistema haga), diseño (donde se determina cómo va a ser el sistema), implementación (donde se construye el sistema) y verificación (donde se chequea que el sistema hace lo que se especificó en la etapa de análisis). Este proceso no es lineal, sino que se realizan ciclos de diseño, desarrollo y evaluación. Estos ciclos son producto de la retroalimentación de los usuarios durante el análisis de requerimientos, el prototipado (construcción de prototipos durante la etapa de análisis y diseño) y la evaluación de la implementación (las versiones del sistema que se van construyendo). Cada vez que el usuario se pone en contacto con lo que se está haciendo, se desprende una lista de opiniones o comentarios (que pueden ser positivos o negativos), que impulsan cambios en el sistema y producen iteraciones en el proceso de diseño.

Para realizar este proceso existen numerosos principios, metodologías y patrones que permiten a los desarrolladores entender y considerar las necesidades de los usuarios. En las siguientes subsecciones haremos un breve repaso por algunos de ellos.

A.6.1 Principios de diseño

Extraído de [24, 40, 43, 44, 49]

Los principios de diseño son objetivos generales que guían al diseñador a establecer las metas del diseño, pero no estableciendo métodos que especifiquen cómo cumplir con las mismas. Simplemente ayudan a organizar el trabajo de diseño y mantener el rumbo del proyecto en sus primeras etapas. A continuación se resumen algunos principios establecidos por diferentes autores.

Autor (año)	Principios
Hansen (1971)	• *Conocer al usuario*. Por lo tanto hacer diseños adaptados a ellos.

	• *Minimizar la memorización*. Proveer selección de ítems en lugar de tener que digitar datos. • *Optimizar las operaciones*. Las operaciones más comunes deben encontrarse fácilmente basados en el uso que se hace del sistema. • *Pensar en errores y proveer mensajes claros*. Evitar que se cometan los errores más comunes, hacer que las acciones sean reversibles y garantizar la integridad del sistema ante fallos
Rubinstein y Hersh (1984)	• *Separar diseño de implementación* • *Respetar las reglas de conversación humana* • *Interrumpir con cuidado* • *Enseñar con ejemplos, no con formalismos* • *Intentar aprender los objetivos como parte del sistema* • *Escribir la guía del usuario primero* • *Ejemplos, ejemplos, ejemplos* • *Sin sorpresas* • No culpar al usuario
Heckel	• *Conocer tu tema* • *Conocer tu audiencia* • *Comunicarse visualmente* • *Hablar el lenguaje del usuario* • *Comunicarse con metáforas* • *Darle control al usuario* • *Evitar frustrar al usuario* • *No permitir que el usuario se distraiga con mecanismos* • *Orientar al usuario en el mundo* • *Servir a usuarios principiantes y usuarios con experiencia* • *Considerar la primera impresión* • *Construir un modelo en la mente del usuario* • *Hacer que el diseño sea simple* • *Pero no tan simple* • Es necesario tener visión

Shneiderman (1992)	• *Reconocer la diversidad:* Antes de comenzar un diseño se debe realizar la caracterización de los usuarios y de las situaciones, de forma tan precisa y completa como sea posible.
	
	Conocer al usuario es el primer principio de Hansen (1971). Bajo el aspecto de una idea muy simple se esconde un objetivo muy difícil de lograr. Si bien desde el área de Factores Humanos se pueden obtener ideas, no existe método o teoría que permita comprender a todos los usuarios y sus tareas.
	
	El entender a los usuarios y sus tareas puede ser realizado en varios niveles.
	
	Frecuentemente es de ayuda crear un perfil de características de los usuarios. Esto incluye: características físicas (edad, por ejemplo), conocimiento y experiencia (educación) en computación y con las tareas, habilidad para lectura y tipeo, características psicológicas como actitud y motivación.
	
	Trabajos y tareas pueden ser clasificados en términos de su estructura e importancia, el nivel de entrenamiento y soporte dado, y la naturaleza del sistema utilizado, por ejemplo, regular o casual, obligatorio o discrecional. Esta clasificación puede ayudar a sugerir iniciativas de diseño apropiadas
	
	• *Prevenir los errores antes de que ocurran*
	• *Ocho reglas de oro para el diseño del diálogo*
	  • *Esforzarse por la consistencia*
	  • *Permitir atajos para usuarios frecuentes*

	• *Ofrecer realimentación (feedback) informativa* • *Diseñar diálogos para mostrar trabajos pendientes* • *Ofrecer una gestión de errores sencilla* • *Permitir que las acciones se reviertan fácilmente* • *Soportar que el usuario tenga el control* • Reducir la carga cognitiva de la memoria a corto plazo
Preece (1994)	• *Estudiar la población de usuarios* • *Reducir la carga cognitiva* • *Aplicar técnicas de ingeniería para resolver la problemática del error humano* • Mantener consistencia y claridad

A.6.2 Metodologías de diseño

Extraído de [24, 40, 43, 44, 49]

Como se menciona en la sección anterior, los principios, si bien ayudan a guiar el proceso, no establecen métodos para realizar el diseño. Pero existen metodologías que, al aplicarlas de forma secuencial, producen un buen diseño.

Autor (año)	**Principios**
Gould (1991)	• *Atención temprana en los usuarios (fase de arranque).* Comprender al usuario y el trabajo que realiza. Métodos: • Hablar con los usuarios. • Visitar el lugar de trabajo del cliente. • Observar al usuario trabajar. • Diseño participativo. • Análisis de tareas. • Objetivos de comportamiento testeables. • *Testeo temprano y continuo por usuarios (fase de diseño inicial)*. Aproximación empírica es la única factible. Métodos: • Escenarios impresos. • Manuales de usuario en etapas tempranas. • Simulaciones. • Prototipos en etapas tempranas. • Demostraciones tempranas. • Pensar en voz alta. • Prototipos formales. • Concursos (trate de destruirlo). • Estudios de campo. • Estudios de seguimiento. • *Diseño iterativo (fase de desarrollo iterativa)*. Mejora iterativa del sistema. Métodos: • Herramientas de software. • Organizaciones de trabajo en desarrollo de sistemas. • *Diseño integrado (fase de instalación del sistema)*. Cubrir todos los aspectos de usabilidad evolucionándolos en paralelo y bajo un mismo control (interfaz, ayuda, plan de entrenamiento a usuarios, documentación). Es una metodología bastante informal y muchas decisiones (como cuánto iterar y qué técnicas utilizar) quedan a cargo de los diseñadores e desarrolladores.
Rubinstein y Hersh (1984)	• *Recolección de información:* Plantear objetivos, requerimientos de mercado, requerimientos técnicos, estado del arte, estándares. Realizar el análisis de

	tareas de los usuarios en su entorno. • *Diseño:* Especificar los "mitos" y "modelos conceptuales" a través de los cuales el usuario ve y comprende el sistema. También se debe definir el "modelo de uso" que describe cómo se debe usar el sistema, la documentación y los métodos para manejo y recuperación de errores. • *Implementación:* Diseño y construcción, primero de un prototipo y luego del sistema. • *Evaluación:* Testeo y evaluación de modo formal e informal del sistema. • *Accionamiento (Deployment):* Entrega del sistema, y evaluación de las reacciones del usuario y del mercado.
Shneiderman (1992)	presenta un proceso de 8 etapas para "el ciclo de vida para el desarrollo de sistemas interactivos" (lifecycle for interactive-systems development). Ofrece estas ocho etapas como un marco (framework) que puede ser usado como punto de partida por la gerencia de un proyecto (project management) y enfatiza que este proceso deberá ser adaptado a las necesidades específicas de cada proyecto. Nuevamente se proponen, testeos en cada etapa e iteraciones entre las etapas. Además se hace especial hincapié en la necesidad de involucrar y ayudar a la comunidad de usuarios y de obtener un planeamiento cuidadoso para la confiabilidad, disponibilidad, seguridad, integridad, estandarización, portabilidad e integración. • *Recolección de información:* A través de entrevistas, cuestionarios y análisis de tareas. Preparar cronogramas. Estimar costos de desarrollo, entrenamiento, uso y mantenimiento. Diseñar estrategias de testeo. • *Definir requerimientos y semántica:* Especificar restricciones de seguridad, privacidad e integridad. Crear documentos con guías a seguir. Definir objetivos, tareas, acciones y secuencias. Definir un glosario con el vocabulario utilizado en el ambiente. • *Diseñar la sintaxis y las facilidades de soporte:* Especificar tiempos de respuesta del sistema, diseñar realimentación (feedback) informativa de cada acción. Desarrollar mensajes y manejo de errores. Escribir manuales de usuario y tutoriales. Realizar tests de usabilidad a través de maquetas y prototipos. • *Especificar los periféricos*: Considerar color, tamaño de pantalla y resolución. Establecer requerimientos para líneas de comunicación. Considerar el entorno de trabajo, ruidos, luz etc. Realizar pilotos y testear. • *Desarrollar software:* Producir diseños top-down modulares. Enfatizar en que el software se pueda modificar y mantener de manera sencilla. Asegurar confiabilidad y seguridad. Proveer documentación adecuada. Hacer posible el monitoreo de usuarios y de rendimiento. Realizar tests realistas de uso. • *Integrar el sistema y diseminarse entre los usuarios:* Asegurarse que los usuarios se involucren en todas las etapas. Conducir testeos de aceptación Dar entrenamiento adecuado y permitir consultas de los usuarios. • *Entrenar y educar a la comunidad de usuarios:* A través de consultas telefónicas, personalmente u on-line. Conducir evaluaciones objetivas y subjetivas sobre el sistema en curso y conocer las mejoras propuestas. Responder a las sugerencias sobre mejoras en lo que resulte de las evaluaciones. • *Preparar un plan evolutivo:* Diseñar para que el sistema sea fácil de reparar y mejorar. Usar cuestionarios y entrevistas para obtener realimentación

	desde los usuarios. Programar revisiones e informes regulares.
Nielsen (1993)	se centró en la necesidad de "ingeniería de usabilidad", en el desarrollo de familias de productos enteras, a medida, que evolucionan por extensos períodos de tiempo. Por tal motivo, desarrolló un "modelo para el ciclo de vida de la ingeniería de usabilidad" como un conjunto de once etapas con diseño iterativo: • *Conocer al usuario.* • Características individuales. • Tareas comunes y deseadas. • Análisis funcional. • La evolución del usuario y de su trabajo. • *Análisis competitivo.* • *Definir objetivos de usabilidad. Análisis de impacto financiero.* • *Diseño paralelo.* • *Diseño participativo.* • *Diseño coordinado de toda la interfaz.* • *Aplicar guías y heurísticas.* • *Hacer prototipos.* • *Test empírico.* • *Diseño iterativo. Capturar las bases racionales del diseño.* • *Recoger información del terreno de uso.*

A.6.3 Patrones de diseño

Extraído de [24, 40, 27]

La importancia del uso de patrones en el diseño de interfaces se basa en que aplicando técnicas de patrones se contribuye a un mejor diseño.

Los patrones de diseño son una manera formal de documentar una solución probada a un problema recurrente dentro de un contexto específico. Aportan un lenguaje común de referencia para las personas implicadas, salvan ambigüedades o malas interpretaciones, mejoran la comunicación y comprensión entre los componentes del equipo, y al estar expresados en términos del dominio del problema permiten validar los requerimientos con el usuario final.

Para que los patrones puedan ser utilizados en diferentes contextos, es importante que sean genéricos.

Es conveniente no comenzar un diseño de cero, sino basarse en diseños de interfaces ya realizados y con resultados exitosos, de esta forma un diseñador podría estar diseñando una interfaz sin saber que está aplicando patrones.

Ver Anexo Patrones y reglas de diseño de interfaces gráficas de usuario.

A.7 Prototipos

Extraído de [24, 40, 43]

Es importante la visualización del sistema en etapas tempranas del proceso de diseño. Para esta visualización, en general, se construyen prototipos y metáforas que permiten la simulación del funcionamiento del sistema. Los prototipos proporcionan una visión más certera del ambiente.

Teniendo en cuenta la forma en que se ha planteado el modelo centrado en el usuario, no se puede comenzar una implementación de un sistema a partir de un diseño inicial. Es necesario hacer evaluaciones de la usabilidad en etapas tempranas y para esto son indispensables los prototipos. En general se construyen

rápidamente y a bajos costos, lo que favorece la posibilidad de cambiarlos muchas veces. Los prototipos son documentos, diseños o sistemas que simulan o implementan partes del sistema final. En un diseño centrado en el usuario, éste participará activamente en el desarrollo de un producto, por lo tanto el prototipo es una herramienta muy útil ya que el usuario podrá evaluar el producto desde las primeras fases del desarrollo (Modelo del ciclo de vida basado en prototipos).

A.8 Metáforas

Extraído de [24, 40, 43, 44, 49]

"Consiste en la sustitución del término propio por otro que tiene con él una relación de analogía".

Las metáforas ayudan a comprender el dominio final que se desarrollará, basándose en un dominio inicial ya conocido por el usuario.

Tomas Erickson (1990) discutió el uso de la metáfora en el lenguaje cotidiano, luego utilizó las metáforas en interfaces. Al describir el proceso de generación de metáforas para interfaces, enfatizó en la necesidad de entender las dos partes que la metáfora conecta: el usuario y el sistema. También reiteró el consejo introducido por Gould, de la necesidad de un diseño iterativo y enfocado en los usuarios.

En una presentación no publicada, Verplank (1991) menciona maneras en que el bosquejo ("sketching" en inglés) y la metáfora contribuyen al diseño. Sostiene que existen tres metáforas que han dominado el diseño de interfaces persona-computadora:

1. "la computadora como persona" refiriéndose a los lenguajes de comandos y diálogos.
2. "la computadora como herramienta" centrada en la representación total y el control directo y continuo.
3. "la computadora como medio", metáfora que enfatiza la noción de alcanzar personas distantes y objetos e información. Cada una de estas metáforas trae consigo muchas asociaciones y restricciones en el pensamiento de interfaces de usuario.

A.9 Manipulación directa

Extraído de [24, 40, 43, 44, 49]

A.9.1 Introducción

La interfaz gráfica de manipulación directa es aquella en la cual a través de acciones físicas los usuarios manipulan los objetos gráficos. Estos objetos son elaborados de acuerdo a entidades y/o experiencias del mundo real. Además, las acciones que se pueden ejecutar sobre ellos son realizadas comúnmente utilizando el ratón. Una buena interfaz de manipulación directa da el sentido de control de la aplicación al usuario.

Históricamente, muchas interfaces han sido construidas bajo la metáfora de conversación, en ellas la comunicación se realiza por medio de lenguaje de comandos. Pero la figura de la interfaz, como intermediaria de un mundo oculto impide que el usuario, logre crear un compromiso con los objetos de interés. Es decir, el usuario está en contacto directo con estructuras del lenguaje, estructuras que se refieren a objetos de interés, pero que no son los objetos mismos. En este tipo de interfaces el sistema describe los resultados de las acciones.

En cambio la metáfora del modelo del mundo 1 soporta el concepto de directness ("directitud"). Es decir, en lugar de describir la acción de interés, el usuario realiza esas acciones, y el sistema presenta directamente las acciones llevadas a cabo sobre los objetos.

Las propiedades claves son que los objetos, cualquiera sea su forma, tienen comportamientos, pueden ser referenciados por otros objetos y la referencia a un objeto provoca que este tenga un comportamiento. El objetivo es permitir al usuario actuar como si la representación fuera la cosa misma.

Algunos autores han intentado describir los principios que componen la manipulación directa. Ted Nelson (1980), percibió gran aceptación en los usuarios cuando la interfaz es construida por lo que él llama "principle of virtuality", una representación de la realidad que puede ser manipulada. Rutkowski (1982) derivó un concepto similar en su "principle of transparency". "El usuario es capaz de aplicar su pensamiento directamente en la tarea, la herramienta en sí misma parece desaparecer".

La manipulación directa, también ha sido vista desde la perspectiva de resolución de problemas y aprender investigando. Las representaciones físicas, espaciales o visuales parecen ser más fáciles de retener y manipular que las representaciones basadas en números o texto.

Es un estilo de interfaz muy fácil de aprender y muy poderosa, muchas veces es presentada como el mejor estilo de interfaz, pero sin embargo tiene algunas limitaciones.

La exactitud dentro de la manipulación es librada muchas veces a la habilidad que tenga el usuario con el uso del ratón (por ejemplo: ajustar un botón movible en un valor especificado).

Requiere de mucho conocimiento del mundo real (por ejemplo: es necesario que una persona sepa utilizar una calculadora común para que realmente le resulte fácil el uso de una que esté representada en pantalla).

Otro de los problemas con la manipulación directa es que no todas las tareas pueden ser descritas por objetos concretos y no todas las acciones se pueden hacer directamente. Por ejemplo, uno de los primeros problemas que se presentaron fue como representar el concepto de buffer, que se solucionó utilizando el concepto de cortar y pegar.

Comúnmente manipulación directa es utilizada en combinación con otros estilos de interfaces, como la ejecución de comandos por medio de menús o por el teclado, permitiendo una mejor usabilidad de la misma.

What You See Is What You Get ("Lo que ves es lo que hay" se traduciría del inglés, también conocido por sus siglas WYSIWYG) y manipulación directa son conceptos diferentes y no se deben confundir. Mientras que la representación de una imagen gráfica puede ser modificada por la manipulación directa de la misma, en una wysiwyg puede ser modificada por otro estilo de interacción (por ejemplo: comandos). El uso de estos dos conceptos en conjunto produce interfaces muy poderosas, fáciles de aprender y rápidas de usar.

A.9.2 Principios de la manipulación directa

Extraído de [24, 40, 49]

Shneiderman propuso tres principios al describir interfaces de Manipulación Directa:

- Representación continua de los objetos y acciones de interés.
- Acciones físicas o presionar botones etiquetados en vez de sintaxis compleja.
- Operaciones rápidas, incrementales y reversibles cuyo impacto sobre el objeto de interés es visible inmediatamente.

A.9.3 Cualidades de la manipulación directa

Extraído de [24, 40, 49]

Shneiderman planteó que utilizando los principios que fueron expuestos anteriormente, sería posible diseñar un sistema con las cualidades que se detallan a continuación, sin embargo, en el libro de Hutchis, James y Norman se puede encontrar una crítica a estas cualidades. Los beneficios planteados por Shneiderman se detallan en letra cursiva y las revisiones en letra regular.

"Los novatos pueden aprender rápidamente funcionalidades básicas, usualmente a través de la demostración hecha por un usuario más experto".

En realidad la velocidad de aprendizaje se da fundamentalmente cuando el novato conoce sobre el dominio de la tarea. El usuario siente que trabaja directamente en dicho dominio y conoce los elementos que la interfaz requiere para su uso. En cuanto a la demostración, esta es una forma probada y efectiva de aprendizaje, y esto no está relacionado con la manipulación directa. Sin embargo, sí es una ventaja de la manipulación directa el hecho de que las acciones y sus resultados son visibles inmediatamente para el usuario.

"Expertos pueden trabajar de forma extremadamente rápida."

Esto no necesariamente es así, ya que los usuarios expertos pueden acceder más rápido a una funcionalidad digitando un comando, que en un ambiente gráfico.

"Usuarios "Intermitentes" que sean expertos pueden retener conceptos operativos."

El experto en usar un sistema refleja también ser experto en el tema en cuestión. En estos casos, el tema está bien establecido en la memoria de este usuario. Además, si en la interfaz existe un buen mapeo semántico de las acciones, el olvido será lento y será fácil volver a recordar aquello ya olvidado. Esto funciona para Manipulación Directa y para otros métodos, mientras haya sistemas bien diseñados.

"Mensajes de error se necesitan raramente."

Dado que los resultados son inmediatamente visibles, entonces no se necesitarían mensajes, pero es posible que ocurran errores conceptuales que son operaciones legales y que son difíciles de detectar. Por lo tanto, hay ocasiones en que un mensaje es deseable, pero el sistema no lo presenta dando por obvio que el usuario detectará que determinada operación no fue realizada. Esto es lo que Norman & Lewis definen como estrategia "do nothing", la cual no es siempre deseable. Además, es natural que los usuarios comentan errores durante la ejecución de tareas, es deseable que los sistemas provean la posibilidad de recuperarse de los mismos. Es importante programar para prevenir los errores. Para esto también en conveniente, que el usuario tenga suficiente realimentación del estado del sistema, facilitando la tarea de no cometer errores.

"Los usuarios pueden ver inmediatamente si sus acciones están siguiendo sus objetivos, y si no, pueden cambiar la dirección de su actividad."

Si el sistema no es reversible, no hay diferencia con otros sistemas, dado que es difícil cambiar la dirección. Esto significa que haya "deshacer".

"Usuarios han reducido su ansiedad a causa de que el sistema es comprensible y porque las acciones son fácilmente reversibles"

Esto es difícil de probar. Se requerirían 2 sistemas con iguales capacidades y distancia semántica. El concepto de distancia semántica será expuesto más adelante.

A.9.4 Objetivo

Extraído de [24, 40, 49]

En la manipulación directa, en lugar de tener un medio computacional abstracto, todo su manejo es realizado gráficamente, en una forma que se corresponde naturalmente con la manera en que el usuario piensa. No hay operaciones ocultas, ni sintaxis, ni nombres de comandos que aprender. Esto se corresponde también con el concepto de wysiwyg. El sistema requiere que el usuario tenga habilidad en el dominio de las tareas, pero muy pocos conocimientos computacionales.

A.9.5 Una explicación cognitiva de la Manipulación directa

Extraído de [24, 40, 49]

Directness es una impresión o sentimiento respecto a una interfaz. Parte de ese sentimiento es producido cuando el usuario se adapta a la utilización de una interfaz, por tal motivo, el diseñador no puede estar seguro que controla completamente un proceso, ni que puede tomarse todo el crédito por el sentimiento de directness que pueda experimentar el usuario. La sensación de directness es siempre relativa.

El concepto de directness es explicado en un esquema de interacción hombre – máquina basado en la teoría de la acción de Norman. Sobre todo en las 7 etapas de la actividad que realiza una persona cuando ejecuta una tarea y en los dos "abismos" que separan los objetivos e intenciones del usuario de los conceptos y operaciones representados por un sistema. Estos conceptos fueron incorporados por Norman al estudiar la Ingeniería Cognitiva.

A.9.6 "Teoría de la acción" de Norman

Extraído de [24, 40, 49]

Esta teoría busca ayudar a los diseñadores a comprender la opinión de los usuarios, la percepción, la cognición, y las funciones del motor perceptivo.

Norman sugiere las "siete etapas de la actividad del usuario" y las interdependencias asociadas que son características en la interacción entre el hombre y la máquina. Estas etapas se dan cuando un usuario realiza una tarea. En detalle, Norman define un "Abismo de Ejecución " y un "Abismo de Evaluación" que corresponden a las dificultades que se presentan al interactuar un usuario con una máquina. También especifica cuándo se ha tendido un puente para intentar disminuir la distancia en ambos abismos. Abismos, puentes y distancias serán estudiados con más detalles las secciones siguientes.

Abismos de Ejecución y Evaluación

Los objetivos del usuario y el estado del sistema difieren mucho en forma y contenido, los primeros son expresados en términos psicológicos y el segundo presenta su estado actual en términos físicos.

El *abismo de Ejecución* se refiere a la transformación requerida a los objetivos del usuario en acciones de entrada al sistema.

El *abismo de Evaluación* significa el problema de mostrar las reacciones del sistema en una representación que pueda ser percibida, interpretada y evaluada por el usuario con respecto a sus objetivos.

Estos abismos pueden salvase, construyendo puentes en ambas direcciones, como muestra la Figura A-5.

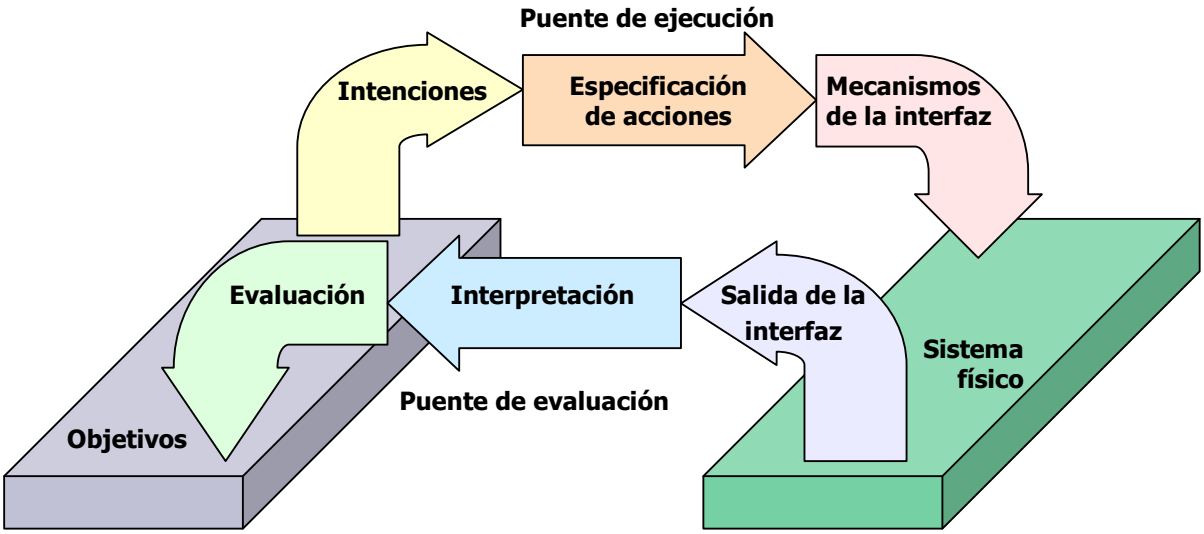


Figura A-5: Puentes de ejecución y evaluación

El diseñador trazará puentes desde el sistema al usuario, construyendo entradas y salidas de la interfaz que se correspondan de la mejor manera posible con las necesidades psicológicas del usuario.

El usuario trazará puentes creando planes, secuencias de acción, e interpretaciones del sistema que sirven para desplazarse de la descripción normal de objetivos e intenciones a algo cercano al sistema físico.

- *Puente de ejecución*
  - *Intenciones*: es lo que el usuario desea hacer. Esta intensión podría estar influenciada por la interfaz.
  - *Especificaciones de acciones*: consiste en establecer correspondencias entre: los objetivos de las intenciones y lo que se puede hacer con las variables del sistema, los mecanismos físicos y las variables del sistema, el estado del sistema y los objetivos e intenciones.
  - *Ejecución de las intenciones*: está determinada por los periféricos de entrada del sistema y consiste en la acción física sobre los mismos.
- *Puente de evaluación*
  - *Salida por los dispositivos (de salida) de la Interfaz*: son realizados por el sistema, diseñados por el diseñador y el usuario utiliza el proceso de percepción de dichos dispositivos.
  - *Interpretación*: el usuario interpreta lo percibido en la interfaz.
  - *Evaluación*: Comparación entre la interpretación del sistema y los objetivos e intenciones originales.

A.9.6.1.1 Dos aspectos de Directness*: Distancia y Compromiso*

Extraído de [24, 40, 49]

Según Hutchins se consideran dos aspectos distintos del sentimiento de directness; uno referido a la distancia entre los pensamientos del usuario y los requerimientos físicos del sistema que se está usando. Este aspecto es llamado Distancia, enfatizando que no es una propiedad exclusiva de la interfaz, sino que involucra una relación entre la tarea que el usuario tiene en mente y la forma en que puede realizarse a través de la interfaz. El otro aspecto concierne al sentimiento de Compromiso que un usuario experimenta cuando está manipulando directamente los objetos de su interés.

Distancia: Si la distancia es corta, los pensamientos son trasladados directamente en las acciones requeridas por el sistema y la salida del sistema es interpretada sin dificultad en términos de los objetivos de interés del usuario. Aquí, el tema crítico es minimizar el esfuerzo requerido para trazar los puentes de Ejecución y Evaluación. Esto puede lograrse haciendo que los comandos y mecanismos del sistema se correspondan con los pensamientos y objetivos del usuario, y además que los dispositivos de salida presenten un buen modelo conceptual del sistema que pueda ser percibido, interpretado y evaluado por los usuarios. El objetivo en ambos casos es minimizar el esfuerzo cognitivo.

Compromiso: Es la impresión o sensación de que se están manipulando directamente los objetos de interés. El usuario se siente completamente comprometido con el control de los objetos, no con el programa o la computadora, sino con los objetos semánticos de sus objetivos e intenciones.

A.9.6.1.2 Dos formas de distancia: Semántica y Articulatoria

Extraído de [24, 40, 49]

Retomando el tema de la distancia, se explora la forma en que una interfaz puede ser directa o indirecta con respecto a una tarea particular.

Cuando se interactúa con un dispositivo, se está usando un lenguaje de interfaz. Es decir, existe un lenguaje para describir las acciones que pueden ser ejecutadas, a esto se le llama lenguaje de entrada. Después que esas acciones son ejecutadas, el dispositivo dejará disponible una indicación de que es lo que ha sucedido, esto es descripto en el leguaje de salida de la interfaz.

Cada expresión en el lenguaje de la interfaz tiene un significado y una forma. La distancia semántica refleja la relación entre las intenciones del usuario y el significado de la expresión en el lenguaje de la interfaz (de entrada y salida). La distancia articulatoria refleja la relación entre la forma física de una expresión en el lenguaje de la interfaz y su significado, (entrada y salida)

A.9.6.1.3 Distancia Semántica

Extraído de [24, 40, 49]

Distancia semántica es la relación entre el significado de una expresión en el lenguaje de la interfaz y lo que el usuario quiere decir. La idea es que la tarea sea descrita al nivel más cercano a las intenciones del usuario para minimizar la distancia en el Abismo de Ejecución.

Directness semántica requiere hacer una correspondencia entre el nivel de descripción del lenguaje de interfaz y nivel en que la persona piensa en cómo hacer la tarea. Cuánto más sea lo que el usuario tiene que proveer, mayor será la distancia.

Con respecto al Abismo de Evaluación, refiere a la cantidad de procesamiento que necesita realizar un usuario para determinar que la tarea ha sido realizada. Si los términos de la salida no son aquellos utilizados por la intención del usuario, el usuario tendrá que traducir este salida en términos compatibles con la intención, de manera de poder hacer la evaluación. Si la salida reduce la carga de trabajo mental, la distancia semántica entre la intención y el lenguaje de salida será más corta. Lo ideal es que la salida represente directamente a los conceptos semánticos (por ejemplo sistemas wysiwyg).

En general, hay sólo dos formas básicas de **reducir la distancia semántica**, una desde el lado del sistema y otra desde el lado del usuario:

- El diseñador puede trazar un puente entre los abismos construyendo un lenguaje más especializado y de mayor orden que se dirija hacia el usuario, haciendo la semántica de los lenguajes de entrada y salida corresponda con los del usuario.

- El usuario puede trazar un puente adquiriendo competencia y construyendo nuevas estructuras mentales para sobrepasar los abismos. En particular, esto requiere que el usuario automatice la secuencia de respuesta y aprender a pensar en el mismo lenguaje que aquel requerido por el sistema.

A.9.6.1.4 Distancia Articulatoria

Extraído de [24, 40, 49]

Relación entre el significado de una expresión y su representación.

El lenguaje natural tiene estructura fonética (lenguaje hablado) y tipográfica (lenguaje escrito), de manera similar los ítems que forman una interfaz tienen una estructura física. Directness articulatoria es la relación entre el significado de una expresión y su forma física.

La distancia articulatoria:

En Ejecución: refiere a la concordancia de la relación entre el significado de la acción requerida y la forma en la que se realiza. Sucede cuando la forma física de las acciones no corresponde directamente con su significado.
Entrada: secuencia de teclas del teclado, movimiento del ratón, una palabra hablada.

En Evaluación: se refiere a la distancia entre la forma de la salida o respuesta del sistema y su significado. Sucede cuando la forma física de la salida no corresponde directamente con su significado.
Salida: cadena de caracteres en la pantalla, cambio en un ícono, un gráfico, un sonido, etc.

En el lenguaje hablado esta relación se llama onomatopeya, pero se usa poco. En los lenguajes de la interfaz debería ser más fácil de explotar la similitud articulatoria.

Métodos de evaluación

Extraído de [24, 40, 49]

La evaluación de la usabilidad implica analizar el entorno y los usuarios que van a utilizar el producto, probar un prototipo, diseño o producto con una selección de usuarios, analizar el diseño con expertos, etc., en definitiva, conseguir su integración en el ciclo permitiendo la realización de un diseño centrado en el usuario.

"La evaluación comprende un conjunto de metodologías y técnicas que estudian la usabilidad de un sistema interactivo en diferentes etapas del ciclo de vida.".

Los evaluadores inspeccionan y revisan un conjunto de aspectos relacionados con la usabilidad de la interfaz. Hay tres tipos distintos de inspecciones: evaluación heurística, recorridos cognitivos e inspección de estándares.

A.9.6.1.4.1 Evaluación heurística con guías de usabilidad

Extraído de [24, 40, 43, 44, 49]

Guías: Colección de tests aplicados a la interfaz para determinar si esta es satisfactoria.

Las guías deben ser formuladas de forma que se puedan probar. De esta forma, diseñadores o verificadores podrán examinar si una interfaz cumple o no con las guías. Existen guías vagas, contradictorias y formuladas en un nivel inapropiado de especificidad.

Finalmente, es muy difícil tratar con un conjunto tan grande de guías. Nielsen realizó una lista de 10 heurísticas de diseño, basado en 249 problemas de usabilidad encontrados en 11 proyectos. Estas guías son de "sentido común" conocidas por los especialistas en usabilidad, pero de todas formas facilita el trabajo tenerlas juntas mientras se examina una interfaz.

10 Heurísticas de Jacob Nielsen

Extraído de [24, 40, 43]

Visibilidad del estado del sistema: El usuario debe saber exactamente que es lo que el sistema está haciendo, el sistema debe proveer la realimentación apropiada dentro de un tiempo razonable.

Similitud entre el sistema y el mundo real: El sistema debe hablar el lenguaje del usuario. Debe seguir convenciones del mundo real, hacer que la información aparezca en orden lógico y natural.

Control del usuario y libertad: Los usuarios a menudo eligen funciones del sistema por error y necesitarán una "salida de emergencia" para dejar el estado no buscado de forma claramente determinada. Se debe soportar "deshacer" y "rehacer".

Consistencia y estándares: Los usuarios no deben preguntarse si diferentes acciones, palabras o situaciones significan la misma cosa. Seguir las convenciones de plataforma.

Prevención de Errores: Mejor que mensajes de error es un cuidadoso diseño que prevenga la ocurrencia de problemas.

Reconocimiento mejor que recuerdo: Hacer visible los objetos, las acciones y las opciones. El usuario no tendría que recordar información de una parte del diálogo a otra. Instrucciones para el uso del sistema deberían ser visibles o fácilmente recuperable cuando se lo requiera.

Flexibilidad y eficiencia de uso: Aceleradores –no vistos por los usuarios novatos- pueden acelerar la interacción del usuario experto de forma que el sistema sirva tanto al novato como al experto.

Estética y diseño minimalista: Los diálogos no deben contener información irrelevante o raramente necesaria. Cada unidad extra de información compite con la información relevante y disminuye su visibilidad relativa.

Ayudar a los usuarios a reconocer, diagnosticar y recuperarse de los errores: Los mensajes de error deberían ser expresados en lenguaje llano, indicar de forma precisa el problema y sugerir una solución de forma constructiva.

Ayuda y documentación: Aun pensando que es mejor si el sistema puede ser usado sin documentación, puede ser necesario proveer ayuda (help) y documentación. Cualquier información de este tipo debería ser fácil de buscar, debería estar enfocada en las tareas del usuario, listar pasos concretos, y no ser demasiado larga.

Estas 10 heurísticas fueron retomadas y adaptadas específicamente para la Web por Keith Instone en 1997.

A.9.6.1.4.2 Recorrido cognitivo (o ensayos de conocimiento)

Extraído de [24, 40, 43, 44, 49]

En los ensayos de conocimiento del diseño de una interfaz, un conjunto de tareas representativas es seleccionado y llevado a cabo. Así sea basado en un artículo o en un prototipo, el diseño debe estar totalmente especificado, permitiendo al sistema responder a cualquier acción descripta. Lo que se pretende es determinar la velocidad de aprendizaje. Planteadas las tareas, el analista explicará la interacción que el usuario puede realizar con la interfaz y que acciones están disponibles. Se evalúa si el usuario desarrolló las tareas necesarias para llegar al objetivo satisfactoriamente. Si el diseño de la interfaz es bueno, las intenciones del usuario provocarán que se seleccione la acción apropiada. La interfaz, además, debe presentar una realimentación indicando que se están realizando progresos para completar la tarea.

Durante esta evaluación los desarrolladores completan formularios que requieren información sobre los objetivos de los usuarios, tareas y subtareas, conocimiento, el estado visible de la interfaz, y las relaciones y cambios sobre estos puntos. Las preguntas están basadas en una teoría cognitiva de las relaciones entre objetivos, acciones y el estado visible de la interfaz.

Las cuatro preguntas más importantes son:

¿El usuario intentará lograr el efecto correcto?
¿El usuario va a notar que la acción correcta está disponible?
¿El usuario podrá asociar la acción correcta con el efecto que se intenta lograr mediante ella?
¿Si la acción correcta es realizada, el usuario podrá ver que el progreso se va haciendo a través de la solución de la tarea?

Esta prueba aplica principalmente a la etapa de diseño, sin embargo también puede ser aplicada a otras etapas.

A.9.6.1.4.3 Evaluación de estándares

Extraído de [24, 40, 43, 44, 49]

La evaluación de estándares es realizada por usuarios expertos en estándares. Los estándares pueden ser de facto o de iure. Los de facto son los que nacen a partir de un producto que tiene una amplia aceptación en el mercado. Los estándares iure son los que están apoyados por alguna institución dedicada a producir estándares. Ejemplos de comités que producen estándares iure son ISO- IEEE- ANSI. El experto realiza la inspección de la interfaz analizando cada una de las características definidas en el estándar.

A.9.6.1.5 Indagación

Extraído de [24, 40, 43, 44, 49]

- *Observación de campo*: Durante las observaciones de campo se realizan evaluaciones en el lugar de trabajo donde los usuarios realizan la tarea que se está analizando. El objetivo es observarlos, ver la forma en que realizan las tareas, y tratar de entender el modelo mental que tienen de las mismas. Se pueden realizar preguntas durante el proceso.
- *Grupo de discusión dirigido*: Se realiza un foro de discusión sobre el sistema entre varios usuarios, con la participación de un moderador que sea especializado en temas de factores humanos. El moderador propone los temas a tratar, captura las ideas de los usuarios y observa su evolución.
- *Entrevistas*: Se realizan entrevistas a los usuarios para obtener información de una manera directa. Sirven para evaluar las preferencias, impresiones y actitudes de los usuarios. En general se parte de un conjunto de preguntas para lograr que la entrevista sea efectiva y la persona que las realiza puede ir modificando las preguntas de acuerdo a como se desarrolle logrando un mayor beneficio.

- *Cuestionarios*: Es menos personal que la entrevista, pero llega a un grupo mayor de personas.
- *Logueo de errores (registro de errroes)*: Se realizan modificaciones en el programa para loguear de las acciones que realiza el usuario. Permite relevar las tareas desarrolladas por el usuario y obtener datos automáticamente.

A.9.6.1.6 Test

Extraído de [24, 40, 43, 44, 49]

- *Pensando en voz alta*: En este método se pide al usuario que exprese en voz alta todo lo que siente piensa y opina, mientras está haciendo uso de la aplicación. El usuario cuenta con la interfaz (que puede ser un prototipo) y un conjunto de tareas para desarrollar. Sirve para conocer los modelos mentales de los usuarios y la terminología que emplean.
- *Interacción constructiva o aprendizaje por codescubrimiento*: Es similar al método "pensando en voz alta", pero es ejecutado por dos usuarios simultáneamente. La ventaja de éste método es que para un usuario es más razonable expresar sus pensamientos cuando interactúa con alguien y no cuando interactúa con una máquina.
- *Medida de prestaciones*: Se pretende mejorara la usabilidad de la aplicación. Se trabaja con usuarios y tareas reales, se registra la forma en la que trabajan y se intenta detectar y solucionar problemas que puedan presentarse.
- *Método del conductor (coaching method)*: Este testeo sirve para conocer la información que es necesario entregar a usuarios inexpertos para poder usar la interfaz. El "conductor" guía al usuario en la dirección correcta mientras este desarrolla las tareas que se le van asignando.

A.10 Patrones y reglas de diseño

Extraído de [24, 40, 43, 44, 49]

A.10.1 Introducción

Un tema recurrente en los años 90 fue el estudio de diseño y desarrollo de interfaces de usuarios. Comenzaron a escribirse documentos guías para aplicaciones generales. Muchos proyectos optan por escribir sus propias guías, que tratan específicamente problemas relacionados a su ambiente.

Existen principios de diseño generales, reglas más específicas y los más recientes estudios hablan de patrones, todos ellos tienen el mismo propósito, ayudar en el diseño de interfaces de usuarios.

A.10.2 Reglas de diseño

Extraído de [24, 40, 43, 44, 49]

En su libro, Designing Visual Interfaces, Mullet y Sano plantean puntos básicos para el diseño de interfaces y técnicas para lograrlos. Jeff Johnson realizó una recopilación de errores comunes en la realización de una interfaz y propone las reglas de diseño para evitarlos. Esta es la bibliografía básica de las reglas que se exponen a continuación.

Elegancia y Simplicidad

Entre los beneficios de los diseños simples se encuentran:

- Accesibilidad (approachability): pueden ser rápidamente entendidos tanto para su uso inmediato, como para invitar a exploraciones adicionales.
- Reconocimiento: pueden ser reconocidos más fácilmente que otros más elaborados. Al presentar menos información visual, son asimilados, entendidos y recordados más fácilmente.
- Rapidez de percepción: Diseños simples pueden ser inmediatamente reconocidos y entendidos con un mínimo de esfuerzo consciente.
- Usabilidad: mejorando la accesibilidad y memorización de un producto, necesariamente mejoramos su usabilidad.

Aparte de la minimización de los componentes y la simplificación de la relación entre ellos, la simplicidad de un diseño depende de tres principios estrechamente relacionados:

- Los elementos deben estar unificados, para producir un todo coherente.
- Las partes deben ser refinadas, para que el usuario preste atención a los aspectos esenciales.
- La aptitud de la solución al problema de comunicación se debe asegurar a todos los niveles.

La elegancia no puede ser determinada por regla alguna, depende del gusto y éste se obtiene por el estudio continuo de interfaces de buena calidad que permite tener puntos de comparación. Se encuentra como patrón común que los diseños complejos raramente son elegantes, por tanto se intenta diseñar aplicaciones simples. Se detallan algunas técnicas que ayudan a la simplicidad de las interfaces:

- *Reducir el diseño a su esencia*: es un proceso que tiene tres pasos
  - Determinar las cualidades esenciales (típicamente una lista de adjetivos) que pueden ser incluidas en el diseño usando elementos formales como: etiquetas, controles, colores, texturas, patrones o imágenes.
  - Examinar críticamente cada elemento y cuestionar si se puede prescindir de él.
  - Tratar de eliminar elementos del diseño, si el sistema colapsa deben permanecer, de otra forma se deben omitir.

- *Regularizar los elementos de un diseño*
  - Utilizar formas geométricas regulares, contornos simples y colores opacos dónde sea posible.
  - Si se requiere el uso de formas similares, se debe tratar de hacerlas idénticas, por ejemplo, en tamaño, forma, color y alineación.
  - Limitar la variación de fuentes y tamaños de letras.
  - Los elementos críticos que deban resaltar en el diseño no deben estar regularizados.
- *Combinar adecuadamente los elementos*: Se debe realizar una revisión sistemática del diseño, para asegurar que no se mantiene redundancia innecesaria:

  - Revisar el papel funcional que desarrolla cada elemento en el diseño.
  - Buscar situaciones dónde múltiples elementos jueguen papeles similares
  - Analizar la forma de unificarlos, si es posible, con algunas modificaciones.
  - Combinar elementos redundantes en uno sólo.
  - Tener en cuenta que esta fórmula no siempre se puede aplicar.

A.10.3 Escala, contraste y proporción

Extraído de [24, 40, 43, 44, 49]

Escala, contraste y proporción de los elementos permiten diferenciarlos entre sí, logrando mayor énfasis en los elementos o las áreas que tengan mayor importancia.
Es importante mantener la relación entre todos los elementos, ya que la modificación de los atributos de uno de los elementos puede impactar en el balance, la unidad y la armonía de todos.

Se presentan tres técnicas simples que son relevantes a la mayoría de los problemas del diseño de la interfaz:

- *Establecer regiones de percepción*: El establecimiento de las regiones se desarrolla con un proceso de cuatro etapas:
  - Agrupar los elementos en un pequeño grupo de categorías (alrededor de siete), teniendo en cuenta su origen o intención de uso. Un grupo puede ser formado por un conjunto de elementos que sean independientes, lo importante es no dejar elementos fuera de los grupos.
  - Determinar la importancia de cada uno de éstos grupos y volver a agruparlos de acuerdo a esta categorización, se obtendrá un número aún menor (3-5)
  - Usar variables de percepción claras para establecer estas prioridades (tamaño y color son un buen ejemplo)
  - Realzar las diferencias entre distintos grupos y minimizarlas dentro de cada grupo.
  - Verificar con el "squint test" (cerrar un ojo y entornar el otro para reducir la luz y desestabilizar el foco) que los elementos de una región se vean como una unidad, pero que el grupo mismo pueda ser separado visualmente del resto de la interfaz.

- *Agudizar las distinciones visuales*, de manera que se reconozcan fácilmente.
  - Hay que determinar cuantos rangos de información vamos a querer diferenciar.
  - Determinar las variaciones disponibles de color o tamaño para poder utilizar.
  - Usar escalonamiento logarítmico y no lineal para asegurar la distinción de los elementos
  - Verificar con el "squint test" los resultados

- *Integrar las figuras con sus fondos:*
  - Hay que determinar el tamaño completo de la combinación figura/fondo.
  - Igualar el peso visual de la figura y el fondo.
  - Dejar espacio suficiente en los márgenes de la figura para eliminar tensión visual. Se requiere un margen mayor que en diseños para impresión.
  - Centrar la figura al fondo (salvo casos en los que los requerimientos de comunicación no lo permitan)

A.10.4 Organización y Estructura Visual

Extraído de [24, 40, 43, 44, 49]

Organización y estructura visual son características que el usuario utiliza para evaluar desde el primer momento los aspectos más relevantes de una interfaz. Una estructura organizada no es algo intuitivo y requiere el establecimiento de relaciones entre los componentes del diseño. Un diseño difícilmente se pueda entender sin la integridad que provee una estructura visual coherente.

La estructura introduce algunos de los siguientes beneficios:

- Unidad, incluso de elementos dispares del diseño, permitiendo que trabajen en conjunto con una meta común de comunicación.
- Integridad: una estructura fuerte y coherente mantiene el diseño enfocado en la meta de la comunicación, creando una forma que contribuye al significado del conjunto.
- Legibilidad, la estructura la realza, dividiendo el contenido de la información en subconjuntos que pueden ser procesados separadamente o en paralelo.
- Control: predice los campos de interés y facilita su navegación a través de la composición.

La organización y la estructura visual se basan en métodos constantes, confiables que se pueden aplicar repetidamente de una misma forma. Cuando la jerarquía es clara, el "display" puede fácilmente reflejar las relaciones entre los elementos, manteniendo el balance en la composición resultante.

La organización y estructura visual dependen de un planeamiento cuidadoso y una implementación meticulosa. Hay cuatro técnicas importantes que pueden ser aplicadas para mejorar la estructura:

- *Usar simetría para asegurar el balance:* No es una condición necesaria en todos los diseños, es una técnica fácil de aprender y buen punto de partida para todo diseño.
  - Identificar los ejes sobre los cuales establecer la simetría (la vertical prevalece en la percepción humana),
  - Balancear cuidadosamente la información en cada lado del eje.
  - Asegurar que el eje de simetría quede centrado en el contexto total de despliegue.
  - Verificar con el "squint test" los resultados

- *Uso de alineación para establecer relaciones visuales*: La alineación de elementos reduce la complejidad de la pantalla, haciéndola más clara y entendible. Los elementos alineados crean una fuerte atracción, estableciendo fácilmente una relación.
  - Hay que identificar las fronteras del área de diseño y buscar caminos para alinear elementos con ella.
  - Identificar elementos y márgenes que se puedan alinear, y modificar tamaño o posición para alinearlos.
  - Buscar elementos aislados y alinearlos con algún otro elemento del diseño.
  - Si un elemento no puede ser alineado con otro, tratar de relacionarlo a las proporciones del "display".

- *Ajuste óptico*: El diseño visual es percibido por la visión humana, y a veces se requiere realizar algunas compensaciones para contemplar peculiaridades de la visión. Por ejemplo, los caracteres con bordes curvos deben extenderse un poco por fuera del límite del margen, sino se hiciera así, a los efectos de la visión humana parecerían más pequeños. Hay que determinar el punto de alineación, unidad de espaciamiento requerida. El ajuste óptico es un proceso de tres pasos:
  - Determinar el punto verdadero de alineación y unidad de espaciamiento requerida
  - Verificar con respecto a los márgenes
  - Realizar acercamientos sobre los elementos a evaluar

- *Configuración del diseño con espacios negativos*: No siempre es necesario que todo el espacio tenga información, al contrario, espacios negativos (blancos) juegan un rol crucial en conducir la atención a regiones dónde se proporciona la información relevante. Se presenta la técnica para lograr esta meta:
  - Revisar la organización de la información.
  - Separar unidades independientes de información por espacios blancos adicionales.
  - Determinar los elementos que requieren énfasis visual
  - Aumentar el espacio blanco que rodea a los elementos críticos
  - Recordar que éstos espacios blancos no son espacios perdidos, sino que dan mayor importancia a los espacios adyacentes. Es la variable de percepción más poderosa, pero hay que usarla con criterio por su costo.

A.10.5 Modularización y Programación

Extraído de [24, 40, 43, 44, 49]

Modularización y programación son especialmente relevantes en el uso de interfaces. Hay tres técnicas importantes que pueden ser usadas para ayudar a que los usuarios lo encuentren previsible:

- *Reforzar la estructura a través de la repetición:* La repetición, puede ser por ejemplo, botones comunes a todas las pantallas que estamos navegando. Es bueno tenerlas siempre en la misma posición en la pantalla, de manera que sean fácilmente identificables.
  - Iniciar con un borrador de la serie de diseños que se producirán.
  - Localizar márgenes comunes o unidades funcionales que deban ser claramente percibidos en el diseño. Los diseños individuales deben ser ajustados para ser consistentes con elementos estructurales mayores.

- Localizar elementos ampliamente espaciados.
- Localizar las rutas que el ojo necesita seguir en el diseño. El uso de elementos estructurales similares puede facilitar esta tarea del usuario.
- Usar ubicaciones estándar y estilos de presentación consistentes para controles, texto o imágenes.

- *Establecer unidades modulares*. Establecer unidades básicas de medida de los componentes.
  - Determinar la unidad vertical (tamaño de controles y espacio entre ellos)
  - La unidad vertical debe permitir a dos controles ser ubicados uno enseguida del otro, debe ayudar en el posicionamiento de controles multilínea, grupos de controles, separación entre grupos.
  - Determinar la unidad horizontal (debe permitir acomodar etiquetas de una palabra y debe ser por lo menos el triple que la unidad vertical)
  - La unidad horizontal óptima debe permitir de 5 a 7 divisiones en un diseño típico y debe permitir también la subdivisión de estas nuevas regiones

- *Creación de estructuras basadas en grillas*. Dividir la pantalla con líneas horizontales y verticales (formando una grilla ó "grid canónico"), de manera de ubicar los elementos dentro de las celdas.
  - Determinar restricciones de tamaño para el área de trabajo.
  - Determinar los módulos básicos verticales y horizontales. El vertical es determinado por el conjunto de "widgets", mientras que el horizontal es determinado por el número de controles. Estos parámetros definen el "grid canónico".
  - Desarrollar un borrador que aproxime los tamaños, posiciones y orientación de los elementos de control relevantes.
  - Usar el "grid canónico" para ajustar los tamaños y posiciones.
  - Para estructuras dinámicas, identificar el tamaño mínimo que pueda ser ubicado en el diseño y evitar el cálculo dinámico

A.10.6 Representación de Imágenes

Extraído de [24, 40, 43, 44, 49]

El uso de imágenes es esencial para la comunicación a través de una interfaz de usuario. Todas las propiedades vistas antes –simplicidad, estructura, escala- son aplicables también a las imágenes.

Las imágenes son particularmente importantes en tres áreas.

- Identificación: sirven como representación de objetos concretos del mundo real, permitiendo su fácil identificación. Cuando la imagen y el texto son opuestos, las imágenes dominan.
- Expresión: tienen un gran poder de expresión y personalización en lo diseñado.
- Comunicación: las representaciones ilustradas cruzan límites sociales y lingüísticos con facilidad cuando los objetos que son representados son relativamente constantes a través de culturas.

Hay técnicas simples que pueden ayudar a mejorar la calidad de las imágenes:

- *Selección del vehículo adecuado*: Un aspecto esencial de las imágenes visuales es la velocidad y la dirección con la cual son reconocidas e identificadas.
  - Si el concepto a ser comunicado es un objeto concreto, familiar y tangible, use un signo de icono. Esto no siempre es posible, las palabras transmiten una idea mucho mejor que la imagen, por ejemplo: no es claro diferenciar un bar, snack, autoservicio o restaurante por imágenes y sí por palabras. El tema es que las cuatro palabras se pueden asociar a los mismos símbolos (cubiertos, por ejemplo)
  - Si el concepto será usado repetidamente se debe usar su representación a lo largo de toda la aplicación o aplicaciones.
  - En muchos otros casos - en procesos abstractos - se recomienda usar etiquetas textuales.

- Evitar tanto como sea posible, la mezcla de signos de iconos y texto dentro de una sola imagen.

Los diseñadores de interfaz siguen esta trayectoria equivocada intentando hacer todo textual o gráficamente, siendo que esto depende del problema de comunicación. Una representación visual puede ser sujeta a un testeo de usuarios antes de tener el producto terminado.

- *Refinamiento a través de abstracciones progresivas*:
  - Hay que determinar el nivel adecuado de abstracción para el conjunto de imágenes.
  - Iniciar con imágenes que incluyan las principales características. El refinamiento de la imagen depende de un proceso continuo de simplificación.
  - Modificar la imagen original, evaluarla.
  - Simplificar formas complejas.
  - Eliminar contornos que no son importantes para la identificación del objeto y experimentar con espacios negativos para sugerir contornos.

- *Coordinación para asegurar consistencia visual*: Para que un conjunto de imágenes trabajen juntas con eficacia deben compartir un lenguaje coherente de forma que la relación de cada imagen en el grupo sea inmediatamente evidente. La consistencia es particularmente importante en un conjunto de imágenes.
  - Iniciar con bosquejos completos para establecer las características de cada imagen.
  - Usar puntos de vista similares para la evaluación de cada imagen.
  - Usar formas similares de representación y niveles de abstracción.
  - Usar consistentemente tamaños, orientación, estructura, color para cada imagen.
  - Si es posible usar los mismos elementos a través del conjunto de imágenes

A.10.7 El color

Extraído de [24, 40, 43, 44, 49]

Investigaciones que se han realizado sobre el estudio de la visión han permitido que hoy tengamos *guías* para la selección del color en las interfaces. Un ejemplo de estas guías es la siguiente:

En general:

- Elegir combinaciones de colores compatibles. Evitar rojo–verde, azul–amarillo, verde–azul, rojo–azul
- Usar contrastes altos de color entre la letra y el fondo
- Limitar el número de colores a 4 para los novatos y a 7 para los expertos
- Usar azul claro sólo para las áreas de fondo
- Usar el blanco para la información periférica
- Usar códigos redundantes (formas además de colores); de 6 a 10 por ciento de los varones tienen algún problema de visión del color. Es necesario tener presente que no se debe abusar de los colores como medios de codificación porque los problemas de visión del color son muy comunes.

Para la Pantallas de visualización de datos:

- La luminosidad disminuye en este orden: blanco, amarillo, cian, verde, magenta, rojo y azul
- Usar blanco, cian o verde sobre fondos oscuros
- Para vídeos inversos usar nada (negro), rojo, azul o magenta.
- Evitar colores muy saturados

A.10.8 El menú

Extraído de [24, 40, 43, 44, 49]

Debe hacerse una correcta selección de las opciones del menú. Se deben incluir instrucciones efectivas, que tengan accesos directos. El diseño gráfico debe tener una estructura jerárquica.

- No usar menú dinámico: Evitar los menús que en un mismo sitio colocan distintas operaciones, dependiendo del estado del sistema. Esto induce a cometer errores.
- *Duplicación de ítems en el menú: La duplicación de ítems, a veces, se da por no tener claro el lugar dónde ubicar un comando. Funcionalidades visibles: Las funcionalidades se deben acceder viendo y apuntando, en lugar de recordando y escribiendo. Las interfaces ocultas deben estar completadas por otras visibles. El camino a todos los comandos debe ser visible, a pesar de que el comando en sí mismo puede no estar visible. En el menú, por ejemplo, los comandos están ocultos; se trata entonces de lograr que el nombre del ítem que los contiene sugiera las funcionalidades a las que permite acceder.*
- *Equivalentes por teclado ("hot keys"): Dependiendo de las habilidades de usuarios y niveles de conocimiento de aplicaciones informáticas, éstos tendrán interés en usar distintos tipos accesos a los comandos. Es conveniente proveer mecanismos alternativos que cumplan las necesidades de distintos tipos de usuarios, considerando principiantes y expertos. Tener equivalente por teclado: las opciones del menú deben tener un equivalente por teclado, de forma de poder acceder a ellos sin necesidad de utilizar el ratón.*

A.10.9 Las ventanas

Extraído de [24, 40, 43, 44, 49]

Existen distintos tipos de ventanas. Las ventanas primarias de una aplicación tienden a estar desplegadas por mucho tiempo y poseen acceso a la ayuda de la aplicación. Pueden ser minimizadas y no deben bloquear a otras ventanas.
Otro tipo de ventanas son las cajas de diálogo ("dialog box"), estas se caracterizan por mostrar información transitoria, se cierran si se cierra la ventana padre, pueden bloquear la aplicación, tienen un botón cerrar o cancelar y no poseen barras de menú y de herramientas.

- Los títulos: Deben identificar a las ventanas, no puede usarse el mismo título en ventanas distintas. Deben corresponder al comando que las invocó.
- Las barras: La ventana primaria siempre tiene una barra de menú. La barra de herramientas es opcional. No es necesario que tenga todos los comandos del menú, solamente los más importantes. Los comandos que contiene podrían ser seteados por los usuarios de acuerdo a sus necesidades.
- Botones de las cajas de diálogo: En una caja de diálogo pueden existir distintos tipos de controles, algunos controlan datos específicos y otros afectan a toda la ventana (aceptar-cancelar).Deben diferenciarse los distintos tipos de botones de una caja de diálogo. Identificar por un lado los que controlan datos específicos, y por otro, los que afectan a la ventana.
- Presentación de la ventana: Se debe cuidar el orden de lectura. Tener en cuenta la frecuencia de uso de la información y los controles. Cuidar la relación que tienen los controles. Tener en cuenta las expectativas de los usuarios.
- Ubicación de las ventanas: Desplegar todas las ventanas en la misma coordenada. Las ventanas hijas se despliegan en el medio de la ventana padre. Decidir dónde debe aparecer cada ventana. La posición dependerá del tipo de ventana (principal, subordinada, de atención, de error) La ventana debe aparecer enteramente en la pantalla. No cubrir información importante (Buscar de MS-Word).

A.10.10 Opciones de selección

Extraído de [24, 40, 43, 44, 49]

- ❖ Casillas de verificación (check-box): Las casillas de verificación representan una elección encendido/apagado. Los valores entre las distintas casillas son independientes. Podrían ser dependientes si hay un máximo de opciones a elegir. Identifican una de dos opciones posibles, del tipo verdadero/falso.
- ❖ "Radio buttons" : Los "Radio buttons" posibilitan la selección de una opción entre varias. La dificultad que presentan es que requieren espacio en pantalla para desplegar cada una de las opciones. Deben agruparse de manera tal que a simple vista se vea las opciones que van juntas. En la agrupación se deben seguir las reglas de la Gestalt. La utilización de separadores ayuda a identificar los distintos grupos de opciones.
- ❖ Botones como teclas de conmutación binaria (toggles): Algunas aplicaciones utilizan botones para controlar seteos que son del tipo encendido/apagado, cambiando su nombre cuando se presiona. No queda del todo claro si el nombre del botón corresponde a la acción que se va a ejecutar o al estado actual del sistema. Esto confunde al usuario. No utilizar botones como teclas de conmutación, para estas operaciones se deben utilizar checkboxes, o click-on, click-off togglebuttons, u otros.

A.10.11 Falta de realimentación de información

Extraído de [24, 40, 43, 44, 49]

- ❖ Selecciones ambiguas: Si hay una selección primaria y muchas secundarias, solo la primaria debe estar seleccionada. Si tenemos más de una vista para un mismo objeto, la selección en una vista debe reflejarse en el resto
- ❖ Ejecución de botones: Los botones debe cambiar su apariencia a "presionado" cuando se ejecuta la acción "mouse-down " sobre ellos. Los botones debe cambiar su apariencia a "no - presionado" y ejecutar la acción que corresponda cuando se ejecuta la acción "mouse-up " sobre ellos.

A.10.12 Uso de lengüetas (tabs)

Extraído de [24, 40, 43, 44, 49]

- La finalidad de las lengüetas es navegar entre distintos paneles
- No debe usarse un número excesivo de lengüetas, si se necesitan demasiadas buscar otro mecanismo
- Tratar de usar dos líneas de texto en las etiquetas, si la GUI lo permite, para no tener abreviar las palabras, evitando que éstas pierdan su significado.
- No utilizar nunca lengüetas danzantes. Esto hace que el usuario pierda dominio del sistema.

A.10.13 Etiquetas

Extraído de [24, 40, 43, 44, 49]

- *Alineación*: No hay una regla de diseño que determine como debe ser la alineación, pero al elegir una forma se debe mantener la consistencia en toda la aplicación.
- *Debe colocarse próximo al objeto al que hace referencia*, que al resto de los objetos.
- *Debe estar alineado con el objeto al que hace referencia*.

A.10.14 El texto

Extraído de [24, 40, 43, 44, 49]

Debe haber consistencia en la fuente utilizada en todas las ventanas de la aplicación. El tamaño de la fuente debe permitir su lectura y si es posible cambiar el tamaño acorde al gusto del lector.

Se debe colocar "Tooltips" (etiquetas que aparecen al estar activo el elemento en pantalla) para ampliar la información que se está visualizando.

La terminología:

- La terminología empleada debe ser consistente. Un término no puede tener más de un significado en la aplicación.
- Desarrollar un vocabulario que explique el significado de cada término visible al usuario en el producto y documentación.
- Consultar el vocabulario empleado con los usuarios, ellos determinan si los términos son comprensibles.
- Si existen estándares en la industria para nombrar los elementos, se deben utilizar y contemplar para no ser utilizados para otras funcionalidades.
- Si un mensaje se va a repetir en distintos sitios en la implementación, no incluir el mensaje sino una referencia. Los mensajes deben estar en un lugar de la aplicación sin repetirse.
- Evitar términos similares para funciones diferentes.
- Cuidar que los términos empleados no tengan en el mundo real un significado completamente distinto al que se le da en la interfaz, pudiendo confundir al usuario.
- Los nombres de los componentes en la interfaz deben quedar fuera de la interfaz desarrollada.
- Usar el término "Crear" en lugar de "Nuevo"

A.10.15 Campos de texto

Extraído de [24, 40, 43, 44, 49]

El diseño de un campo de texto depende de su tipo:

- *Editable y activo*: dentro de una caja negra con fondo blanco.
- *Editable e inactivo*: luce como un campo editable pero con el texto, la etiqueta y el borde agrisado.
- *No editable*: No estará dentro de una caja negra y debe lucir igual que una etiqueta.

- *Solo utilizar campos de texto cuando la información no es estructurada*.
- *No utilizar campos de texto* con información del tipo: números, fechas, teléfonos, horas, dinero, etc.

A.10.16 Estado de la aplicación

Extraído de [24, 40, 43, 44, 49]

Utilizar el tipo de cursor, barras de progreso, indicadores de "Trabajando" como indicadores para reconocer el estado del sistema.

- *Mostrar el indicador "Trabajando"* en la pantalla primaria de la aplicación cuando la aplicación está realizando alguna operación.
- *Mostrar indicadores de progreso* en operaciones lentas.

- *Desplegar el cursor ocupado* mientras el sistema realiza las tareas.

A.10.17 Controles inactivos

Extraído de [24, 40, 43, 44, 49]

- *Los controles inactivos deben estar deshabilitados*, el color gris utilizado debe dejar bien claro que la operación no está disponible.

A.10.18 Mensajes de error

Extraído de [24, 40, 43, 44, 49]

Es recomendable tener buenos mensajes de error, que faciliten la recuperación.

- *Expresar los errores en el vocabulario del dominio de la tarea.*
- *Debe identificarse el problema y también aportarse una solución*.
- *Los errores deben tener significado para los usuarios*.
- *Los mensajes deben ser parametrizables* para incorporar datos que ayuden a identificar el problema durante la ejecución.
- *Al escribir un mensaje de error hay que tener en cuenta el tipo de usuario para el que está destinado*:
  - usuario
  - administración del sistema
  - desarrolladores (debug)

A.10.19 Los comandos

Extraído de [24, 40, 43, 44, 49]

En el etiquetado de los comandos existen ciertas convenciones para el uso de "..."

- "..." se utiliza para indicar que se va a desplegar una caja de diálogo para el ingreso de opciones adicionales
- *No se utilizan las elipsis para todos los comandos que abren una ventana*.
- *Sólo se usan para comandos etiquetados con texto*, no para los que tienen gráficos(íconos)

A.10.20 Presentación de la información

Extraído de [24, 40, 43, 44, 49]

- *La pantalla pertenece al usuario*.
  - Los usuarios perciben que la pantalla está bajo su control.
- *Preservar la inercia en el despliegue*
  - Cuando el usuario cambia algo en la pantalla, el resto de lo desplegado debe permanecer inalterado.

A.10.21 La implementación

Extraído de [24, 40, 43, 44, 49]

Se debe implementar de acuerdo a las necesidades de los usuarios.

- *Al tomar consideraciones en el diseño, consultar a usuarios*, buscando su conveniencia.
- *El control debe estar en el usuario* (orden para llenado de los campos)
- *Se deben enfatizar los controles que son importantes*, principalmente cuando se presenta demasiada información en pantalla.
- *Se debe permitir que los usuarios puedan setear la configuración de la interfaz* (color y fuentes)
- *Exposición progresiva*. Sólo mostrar la complejidad cuando sea necesario.
- *Debe promoverse la comunicación entre los programadores*, de forma tal que, por ejemplo, funciones similares tengan interfaces consistentes. Si es posible tener un coordinador de interfaces gráficas.
- *Sugerir el formato que tienen que tener los datos* ("Ver y apuntar" versus "recordar y escribir")
  - Por ejemplo, en un campo de fecha, se pueden adecuar la forma de ingreso con la cantidad de campos para rellenar la información, dejando tres campos para que completen día, mes y año o desplegando dispositivos de selección.

A.10.22 Modos de operación

Extraído de [24, 40, 43, 44, 49]

Ventajas	Dificultades
• El usuario especifica la acción una sola vez. • Disminuye la cantidad de comandos • Mayor guía para los usuarios. • Se puede usar como seguridad para bloquear ciertas funciones para prever accidentes. • Utilizarlo para reconocer operaciones excepcionales.	• Antes de necesitar un modo, se debe pensar en setearlo. • Los usuarios deben reconocer el modo en el que estén • Hace que los usuarios cometan errores. • El sistema debe permitir la recuperación de errores de modo. • Restringen las acciones a las permitidas en el modo. El usuario se debe rendir ante el control del sistema

- *Minimizar la existencia de modos*.
- *Evitar la existencia de secuencias forzadas y restricciones temporales sobre las acciones de los usuarios*.
- *Desplegar indicadores de modo* mediante la utilización de diferencias en cabezales, formatos, avisos, o etiquetas.

A.10.23 Información relevante

Extraído de [24, 40, 43, 44, 49]

La siguiente información debe estar resaltada en el diseño, de manera de ser fácilmente identificada por el usuario:

1. Indicador de estado del sistema.
2. Indicador de modo.
3. Prompts para entrada de información.
4. Resultados
5. Mensajes de error y de estado.

6. Controles para manipular la aplicación.

Los problemas que son más frecuentes en éste punto tienen que ver con el tamaño del texto, su ubicación, que no se destacan de una pantalla completamente uniforme o que cambian demasiado rápidamente.

"Según indica LILLO JOVER los estudios de ANSHEL proponen una estrategia para delimitar el tamaño mínimo de una imagen o información en pantalla para que sea correctamente legible, observándola a una distancia tres veces superior a la que debe ser percibida en condiciones normales para comprobar si sigue siendo identificable. En éste caso, ANSHEL afirma, que se reduce la fatiga visual y se mejora la percepción de los iconos".

Para ello se debe resaltar aplicando las siguientes reglas:

- *Usar tamaños más grandes*
- *Ubicar la información dónde el usuario mira*
- *Resaltar la información* por el uso de letras negritas, colores contrastantes, usar sonido, etc.
  - Resaltar la información con gráficos.

A.10.24 Instrucciones

Extraído de [24, 40, 43, 44, 49]

- *La información debe permanecer visible*. Una pantalla que detalla una secuencia de pasos para una determinada ejecución debe permanecer visible mientras el usuario ejecuta esos pasos. Los usuarios no deben tener que recordar múltiples instrucciones

A.10.25 Instalación de la aplicación

Extraído de [24, 40, 43, 44, 49]

- *Se debe dar alta prioridad para mejorar las experiencias de instalación de los clientes*.
- *Los profesionales de las interfaces deben tener interés en cómo se realiza la instalación*.
- *En general no están involucrados en los procesos de diseño y testeo de la instalación*.

A.11 Patrones de diseño

A.11.1 El concepto de patrones

Extraído de [24, 27, 40, 43, 44, 49]

El concepto de patrones surgió en los años 70, cuando el arquitecto Christopher Alexander presentó el libro "The Timeless Way of Building". Él expone que es posible alcanzar la excelencia en arquitectura utilizando un conjunto cuidadosamente seleccionado de *patrones*. Expresar con palabras la calidad de un edificio bien diseñado es muy difícil, sin embargo, los patrones que se deben utilizar para lograrlo son simples y fáciles de entender.

Alexander define un patrón como: *una regla de tres partes, que expresa una relación entre un cierto contexto, un problema y una solución.* Luego lo detalla de la siguiente manera: "*Cada patrón describe un problema que ocurre reiteradamente en nuestro ambiente, y describe la base de la solución a ese problema, en una manera tal que se puede usar esa solución más de un millón de veces...*".

No son principios abstractos que requieren volver a descubrir la forma de aplicarlos cada vez que se usan, tampoco son tan específicos para una situación o cultura particular. Son algo entre medio, *buenas soluciones posibles para un problema de diseño común, dentro de cierto contexto, describiendo las calidades invariantes de todas esas soluciones.*

Para determinar si realmente estamos ante un Patrón, y no ante una teoría o especulación escrita con un formato de patrón, aplicamos la "Regla de tres" ("Rule of Three"). Esta regla establece que un patrón debe encontrarse en al menos tres sistemas existentes.

A.11.2 Patrones de diseño en interfaces de usuario

Extraído de [24, 40, 43, 44, 49]

Los patrones de diseño son una manera formal de documentar una solución comprobada a un problema recurrente dentro de un contexto específico. Pensados inicialmente por Alexander para arquitectura, se volvieron populares en la construcción de software. En particular, los patrones de interfaces de usuarios son convenientes para describir problemas relacionados con la usabilidad.

De la misma manera que los patrones de Alexander hacen la vida más placentera para sus habitantes, los patrones de interfaces de usuario ayudan a diseñar sistemas fáciles de usar para las personas, logrando la representación de la información según las necesidades del usuario. Esto se traducirá en satisfacción, eficiencia y aceptabilidad de los usuarios al utilizar las aplicaciones informáticas.

Los patrones de interfaces de usuario se hicieron conocidos en la comunidad de ingeniería de software con el documento "Common Ground", de Jennifer Tidwell, que apareció en la web en 1998. Pero, las referencias más tempranas a los patrones de Alexander, asociados a las interfaces de usuario, aparecen en el libro "Diseño centrado en el usuario" de Norman y Draper (1986). Desde entonces se han publicado conjuntos de patrones de interfaces de usuario, algunos se tratan como colecciones de patrones, como la colección de Amsterdam (Welie, 2001), mientras que otros se describen como Lenguaje de patrones(Borchers, 2001, Duyne et al., 2003), estos conceptos se definen más adelante.

En los sistemas interactivos debemos tener en cuenta la retroalimentación, es decir, las expresiones de salida de la aplicación (ventanas, iconos, mapas, mensajes de error) que informan al usuario del estado del sistema y el efecto de sus acciones. Este es un concepto que contribuye ampliamente a la usabilidad de una aplicación. El uso de patrones de interacción ayuda a la retroalimentación con un alto nivel de usabilidad.

Los patrones son de suma utilidad para diseñadores de interfaces, ofrecen un compendio de la amplia bibliografía existente sobre el buen diseño de interfaces de usuario, exponiendo el problema, el contexto en el que se desarrolla y la solución.

A.11.3 ¿Patrones o Guías de diseño?

Extraído de [24, 40, 43, 44, 49]

Mientras que las guías capturan conocimiento de diseño en reglas que pueden ser usadas en la construcción de interfaces, los patrones capturan conocimiento de diseño probado y se describen en términos de un problema, contexto y solución. Las guías escritas en formato de patrón podrían llegar a confundirse con un patrón, la diferencia es que los patrones corresponden a soluciones ya probadas.

Se han encontrado problemas en la utilización de guías, algunos se enumeran a continuación:

- Son a menudo muy simplificadas o abstractas
- Pueden ser difíciles de seleccionar
- Pueden ser difíciles de interpretar
- Pueden ser conflictivas
- A menudo tienen puntos de vista autoritarios concernientes a su validez

Uno de los problemas que presentan es que las guías se presentan como generales, sin embargo, son aplicables en determinado contexto solamente. El propósito de una guía no es mostrar el contexto, solo la regla que se debe aplicar.

Comparado a las guías, los patrones contienen un conocimiento más complejo del diseño, incluyendo la información del contexto, y varias guías se integran a menudo en un sólo patrón. Los patrones del diseño pueden ayudar convenientemente a diseñadores a colocar reglas abstractas del diseño en el contexto de proyectos concretos en los que están trabajando.

A.11.4 Formato de los patrones de diseño

Extraído de [24, 27, 40, 43, 44, 49]

Un patrón debe describir siempre las tres componentes fundamentales: contexto- problema y solución, y puede agregar otros campos que clarifiquen su uso.
Se detallan a continuación elementos que se pueden encontrar en los patrones de diseño de interfaces de usuario:

- Nombre: un patrón debe tener un nombre significativo.
- Problema: corresponde al problema que el patrón intenta resolver. Los problemas en el diseño de interfaces están orientados a las tareas del usuario, ya que se interesa fundamentalmente en la usabilidad.
- Principio de usabilidad: Principio o criterio ergonómico en el que se basa el patrón
- Contexto: También enfocado en el usuario. Incluye contexto de uso, las tareas, usuarios y ambiente para los cuales los patrones pueden ser aplicados. Corresponde a las precondiciones bajo las cuales el problema y la solución suelen ocurrir.
- Fuerzas: restricciones y fuerzas principales que deben ser analizadas. Como interactúan o entran en conflicto entre sí y con las metas deseadas.
- Solución: debe describirse muy concretamente y no imponer nuevos problemas. Describe solo la base de la solución. Pueden ser necesarios otros patrones para resolver sub-problemas.
- Razonamiento: una explicación de la solución.
- Ejemplos: una muestra ilustrativa de cómo el patrón es aplicable exitosamente que ayude a comprender el uso del patrón.

A.11.5 Lenguaje de patrones

Extraído de [24, 27, 40, 43, 44, 49]

Si un conjunto de patrones, se organizan y relacionan en una o más jerarquías que se interconectan se pueden considerar como un Lenguaje de Patrones. Un lenguaje de patrones debe proporcionar la guía de cómo utilizar con éxito combinaciones de patrones de una colección.

A los conjuntos de patrones que no tienen las condiciones para ser considerados un Lenguaje nos referimos como Colecciones de Patrones.

Un lenguaje de patrones es más que un catálogo de patrones. Un catálogo es como un diccionario, no provee una escritura, no tiene ninguna regla para el flujo ni para las conexiones internas. Un catálogo de patrones carece de la validación esencial que proviene de reconocer las características combinatorias de los lenguajes Un lenguaje dice qué patrones se pueden combinar y en qué manera, para crear un patrón de más alto nivel.

Otra definición de lenguaje, presentada en establece que una colección de patrones se convierte en un Lenguaje cuando una comunidad acuerda utilizar el conjunto de patrones dado por la colección. Lo cierto es que para poder llegar a la obtención de un buen lenguaje de patrones, se debe primero tener buenos patrones para incluir. La obtención de un lenguaje de patrones es la meta en el estudio de patrones.

Tidweel argumenta los beneficios de los patrones en la comunidad de diseño de HCI de la siguiente manera:

- Será un nuevo vocabulario, más descriptivo para comunicarse sobre el funcionamiento de ciertas interfaces.
- Nos habilita a tener mayor maestría en temas relacionados, como el diseño de paneles de control (de autos, centrales eléctricas), juegos de video, la Web e hipertexto.
- Aislando las cualidades características que hacen que metáforas, idiomas de moda, "widgets" tengan tan buenos resultados, podremos aprender de ellos y actuar de acuerdo a ellos, sin perder las lecciones de nuestra memoria colectiva. Muchas ideas excelentes se han perdido porque quedan fuera de moda, o porque no logran aceptación debido a razones económicas o malas puestas en práctica.
- Puede servir como una base sólida y práctica sobre la cual construir nuevas herramientas para las interfaces de usuario o nuevos conceptos, como realidad virtual, interfaces de sonido o sitios Web de última generación.

A.11.6 Colección de Patrones de Van Welie

Extraído de http://www.welie.com/patterns/ último acceso: 20/11/2010

Existe un conjunto muy grande de patrones de interfaces de usuario, entre ellos se encuentra la colección de patrones de Martijn van Welie. Contiene un conjunto de patrones muy interesante de aplicar en el diseño de interfaces de usuario. Van Welie, también tiene otra colección de patrones clasificada para aplicaciones Web, que se encuentra disponible en el mismo sitio.

La colección para interfaces gráficas de usuario desarrollada por van Welie, está clasificada en seis grupos: Modo, Navegación, Guía y Feedback, Presentación, Interacción física y Selección. El contexto de cada patrón define la situación funcional en la cual el patrón puede ser usado, sin referenciar a patrones de más alto nivel. Van Welie se refiere a ellos como una Colección de Patrones. Se describen a continuación algunos de estos patrones.

En la Tabla A-5 se presentan algunos de sus patrones y como fueron utilizados en el proyecto.

Patrón	Aplicación en FingLab
Nombre: Modo del cursor. **Tipo**: Patrón de modo. **Motivación:** El usuario está creando o modificando un objeto y necesita conocer el modo actual. **Principio**: Retroalimentación ("feedback"). **Solución**: El cursor puede ser cambiado a un icono que le indica al usuario sobre el estado actual de la interfaz. Se debe cambiar el cursor a un cursor neutral cuando la función haya terminado o se haya desactivado.	Para ejecutar una función el usuario mira el cursor, por lo tanto es el lugar más apropiado para mostrar el modo. Esto incrementa la satisfacción y puede disminuir los errores. Ejemplo al armar para indicar que se puede arrastrar.
Nombre: Magnetismo. **Tipo**: Patrón de selección. **Motivación:** Los usuarios necesitan posicionar los objetos con precisión. **Principio**: Conformidad de tareas. **Solución**: Hacer que los objetos se acomoden hacia otras posiciones o hacia otros objetos. Cuando los usuarios están ubicando los objetos, éstos se dibujan sobre otros objetos que estén lo suficientemente cerca. El objeto destino podría actuar como una "pared" que no permita que el movimiento del objeto pase de ahí.	No se necesita mucha precisión para alcanzar el destino. Esto reduce el tiempo necesario para posicionar tareas. Ejemplo al armar servicio que se marcan los que ya son parte del servicio al seleccionar un viaje en la lista de resultados de una consulta.
Nombre: Menú contextual. **Tipo**: Patrón de selección.	Opciones de botón derecho. Habilitación/deshabilitación de funciones,

Motivación: El usuario necesita saber, en cualquier momento del uso, las funcionalidades que tiene disponibles para decidir que hacer. **Principio**: Visibilidad. **Solución**: Poner las opciones en un menú. Mostrar la lista de las funciones que están disponibles en el contexto actual. Hacer que las funciones estén accesibles en una sola acción. Si el número de funciones es alto, agrupar las funciones distinguiéndolas en grupos. Si el número total de funciones es bajo (en general o dentro de un grupo), la lista debe mostrar todas las funciones y mostrar cuales se pueden seleccionar en el contexto actual. Para los usuarios que no están familiarizados con el nombre de la función debe haber una descripción disponible que explica la función. Las funciones se deben ordenar de acuerdo a uno o más de los criterios siguientes: semántica, semejanza, frecuencia de uso o alfabéticamente. Si la lista de posibilidades no diferencia mucho entre los contextos, la lista debe demostrar todas las posibilidades y destacar las posibilidades del contexto actual.	agrupaciones de funciones con líneas en el menú. La lista de funciones da al usuario una visión inmediata de todas las posibilidades. Los seres humanos están familiarizados con estas listas (Ej. menú de restaurante) y reconocerán rápidamente su función. La solución mejora la memorización y la satisfacción. Disminuye el rendimiento porque son necesarias acciones adicionales.
Nombre: Favoritos. **Tipo**: Patrón de selección. **Motivación:** Los usuarios necesitan encontrar un ítem regularmente usado en un conjunto grande de ítems. **Principio**: Minimizar las acciones (flexibilidad). **Solución**: Permitir que el usuario utilice favoritos que apunten a los ítems de la opción Introduzca una posibilidad para crear y para quitar de favoritos. Un favorito es una etiqueta que apunta directamente a un ítem particular. Una etiqueta puede ser una descripción textual o un algo más que ayude al usuario a identificar el ítem. Los favoritos deben ser accesibles en un número mínimo de acciones del usuario, típicamente directamente o en menú. Sin embargo, el número de favoritos puede llegar a ser alto. Por lo tanto, debe permitirse ordenarlos jerárquicamente, en caso de necesidad.	Para los espacios de trabajo recientes Es probable que el usuario sepa este concepto de sus experiencias en libros de texto usando marcadores. El favorito permite que el usuario consiga el ítem sin recordar la localización exacta del mismo. Encontrarlo se reduce otra vez a buscar en el sistema de favoritos que es mucho más pequeño que la colección entera, por lo tanto el tiempo de búsqueda es reducido. La solución aumenta la velocidad y la satisfacción del funcionamiento.
Nombre: Foco. **Tipo**: Patrón de selección. **Motivación:** Usuarios quieren obtener rápidamente información de los objetos y también modificarlos. **Principio**: Restricciones. **Solución**: Introducir foco en la aplicación**.** El foco debe mostrarse visualmente al usuario, cambiando por ejemplo su color o dibujando un rectángulo al rededor de él. Cuando un objeto tiene el foco, se convierte en el objetivo de todas las funcionalidades que son relevantes al objeto.	Cuando se selecciona un resultado y se marca en tabla y gráfico El foco es el equivalente de "agarrar un objeto". Por lo tanto es natural que la funcionalidad que pertenece al objeto está activada y presentada a los usuarios. Esto reduce el número de las acciones necesitadas para seleccionar la función y para ejecutarla para un objeto especificado. La solución mejora la velocidad y la memorización de la tarea.
Nombre: formato no ambiguo. **Tipo**: Patrón de selección. **Motivación:** El usuario necesita proveer a la aplicación datos, pero puede ser que no esté familiarizado con los datos requeridos o la sintaxis que debe usar. **Principio**: Restricciones. **Solución**: Sólo permitir que el usuario ingrese los datos en el formato correcto. Presentar al usuario la cantidad de campos requerida para cada elemento de datos, de acuerdo a su estructura. Indicar en cada campo la unidad de datos, si existe duda sobre la semántica del campo. El campo no debe permitir que los datos incorrectos sean incorporados. Evitar campos donde los	Selección de la fecha de ejecución. La idea principal es evitar el ingreso incorrecto de datos haciendo que esto sea imposible. Se reducen las posibilidades de error si se muestra el formato requerido. El rendimiento en tiempo se verá reducido, pero aumentará la satisfacción.

usuarios pueden mecanografiar el texto libremente. Además, se debe explicar la sintaxis con un ejemplo o una descripción del formato. Es bueno proporcionar opciones por defecto para los campos requeridos. Evitar el uso de campos opcionales, si se requiere su uso se deben marcar como opcionales.	
Nombre: seteo de atributos. **Tipo**: Patrón de selección. **Motivación:** Los usuarios quieren ver los atributos del objeto con el que están trabajando y necesitan saber como modificarlos. **Principio**: Visibilidad. **Solución**: Crear estructuras especiales que muestren los valores de los atributos. Estructurar el documento con elementos de diálogo para mostrar los distintos tipos de atributos. Incluir solo los más comunes y dejar que al usuario personalice los elementos disponibles. Se deben mostrar los atributos del elemento seleccionado, en caso de selección múltiple se mostrarán los atributos comunes.	Muchas cualidades se visualizan en parte en el panel del documento principal, pero no con la misma precisión que un elemento específico de diálogo puede proporcionar. Proporcionando elementos especializados de diálogo para cada atributo, se mejora la precisión. Se puede ahorrar espacio si se proporciona un estilo pop-up o pull-down de diálogo para ingresar los datos. Proporcionando los elementos especializados del diálogo para cada tipo de atributo, los valores son más fáciles de leer y de entender. Al fijar el usuario las cualidades utilizando los mismo elementos del diálogo, adquiere un sentido de manipulación directa.
Nombre: área de comandos. **Tipo**: Patrón de selección. **Motivación:** El usuario necesita saber donde puede encontrar los posibles comandos y como activarlos. **Principio**: Consistencia, visibilidad. **Solución**: Poner los comandos en un área específica y reconocible. Reservar un área en la pantalla para acceso a las funciones. Podría estar ubicada arriba o a la izquierda para asegurarnos que tenga buena visibilidad. Estas áreas deben ser distinguibles usando características visuales. Si el número de funciones es elevado se pueden agrupar conceptualmente.	Barra de menú y de botones agrupados. Con el área de comando una ubicación fija, se crea una forma consistente para que el usuario encuentre las funcionalidades. También facilita la velocidad de la interacción. Se incrementan la memorización y la satisfacción.
Nombre: Preferencias. **Tipo**: Patrón de selección. **Motivación:** Cada usuario es distinto y prefiere hacer las cosas de manera diferente. **Principio**: Adaptabilidad, flexibilidad. **Solución**: Permitir que los usuarios puedan realizar ajustes según sus preferencias. En lugar de forzar una configuración para cada usuario, permitir a los usuarios cambiar sus preferencias. Las opciones pueden agruparse, cuando el número de opciones es grande se puede usar un árbol.	Panel de preferencias. La estructura de navegación es usada para elegir la categoría de adaptación. Los árboles pueden acomodar más categorías que las tabs. Mostrar todos los seteos juntos puede dar al usuario la impresión de un perfil que pertenece a ese usuario. Estas adaptaciones incrementan la satisfacción del usuario y modifican el rendimiento, pero disminuyen el aprendizaje y la memorización.
Nombre: formato tabular. **Tipo**: Patrón de presentación. **Motivación:** Los usuarios necesitan entender rápidamente la información, las acciones a realizar dependen de esa información. **Principio**: Consistencia, previsibilidad (modelos conceptuales) **Solución**: Arregle todos los objetos en una grilla usando un número mínimo de filas y de columnas, haciendo las celdas tan grandes como sea posible. Los objetos se arreglan en una matriz usando el mínimo número de filas y columnas. Los que son de igual tipo deben estar alineados y exhibidos de la misma forma. Si muchos objetos pueden ser agrupados, la grilla se aplica en el nivel de grupos también. Los elementos cortos pueden ser estirados, comenzando y terminando en los límites de la rejilla. Los	Estructura de los formularios. La reducción al mínimo del número de filas y de columnas mejora el tiempo necesario para explorar la información y para tomar la acción apropiada (ley de Fitts). La información es percibida sin molestar al usuario. El tiempo necesario para leer la información se reduce, esto puede aumentar la performance de las tareas. La disposición que resulta es agradable de ver y aumenta la satisfacción.

elementos largos pueden extenderse utilizando celdas múltiples de la rejilla. Los botones estándar de respuesta pueden estar en posiciones predefinidas y pueden estar fuera de la rejilla.	
Nombre: diseño líquido. **Tipo**: Patrón de presentación. **Motivación:** Los usuarios necesitan visualizar la presentación de forma proporcional al tamaño de la ventana y los elementos de la misma deben reacomodarse o cambiar su tamaño acorde sin dejar espacios vacíos. **Principio**: Consistencia, visualización **Solución**: Arregle todos los objetos de forma que se adapten al cambio de tamaño del área que los contiene. También se deben desplegar barras de desplazamiento horizontales y verticales según sea necesario. De esta manera se le da control al usuario sobre la presentación.	Se despliegan barras de desplazamiento cuando la ventana se hace más chica.
Nombre: Protección. **Tipo**: Patrón de dirección, retroalimentación. **Motivación:** El usuario puede accidentalmente seleccionar una función que tiene efectos irreversibles. **Principio**: Manejo de errores (Seguridad). **Solución**: Proteja al usuario insertando un protector. Agregar una capa de protección extra a las funciones para proteger a los usuarios de cometer errores. Los usuarios deberán confirmar la acción.	Con esta capa extra el usuario debería cometer dos errores repetitivos en lugar de uno. Una respuesta segura por defecto reduce las posibilidades del segundo error. La solución incrementa la seguridad, reduce errores y mejora la satisfacción. Sin embargo, esto requiere acciones extra por parte de los usuarios, lo cual reduce el rendimiento.
Nombre: Sugerencias. **Tipo**: Patrón de dirección, retroalimentación. **Motivación:** El usuario necesita saber la forma de seleccionar funciones **Principio**: Descubrimiento incremental (visibilidad). **Solución**: Dar al usuario otras formas de acceso a la misma función. En una de las formas de acceso a la función se puede dar las indicaciones de los demás caminos para la misma función. Se puede utilizar etiquetas múltiples, una indicar cada modo de acceso. Por ejemplo, si la función tiene un "shortcut", muestre la combinación correspondiente. Si hay un icono para la función, muestre el icono. Muestre la etiqueta principal y muestre siempre otras etiquetas directamente, si es posible dentro de las restricciones. Otras posibilidades son utilizar los "tooltips" que se activan cuando el usuario sostiene el ratón sobre un widget por aproximadamente dos segundos.	Avisar de las hot keys en el menú, poner íconos en el menú. Es muy probable que el usuario esté familiarizado con por lo menos una dirección de acceso a una función específica. Mostrando otras etiquetas como el "shorcut" o un icono, el usuario aprenderá más asociaciones para el acceso de la función. La solución acelera el aprendizaje y la memorización. El rendimiento de las tareas puede mejorar.
Nombre: mensajes de advertencia. **Tipo**: Patrón de dirección, retroalimentación. **Motivación:** Situaciones donde el usuario realiza una acción que pueda conducir intencionalmente a un problema. **Principio**: Prevención de errores (seguridad). **Solución**: Advertir al usuario antes de continuar la tarea y dar al usuario la opción de abortar las tareas. Indicar cual es la condición que accionó la advertencia para que el usuario sepa la causa. Una vez que se entienda eso, la pregunta debe hacer al usuario con solamente dos opciones. La solución disminuye errores y aumenta la satisfacción. Las opciones deben incluir un verbo que refiera a la acción deseada. No utilice Si/No como opciones. En algunos casos puede haber más de dos opciones. Quizás un mayor número de opciones sea válido en algunos casos, pero se debe tratar de poner el mínimo.	Indicando la condición que acciona el usuario puede entender porqué aparece la advertencia. Una vez que se entienda eso la pregunta deja al usuario con solamente dos opciones. Si se proporcionan solamente dos opciones la opción es simple para el usuario: continúe o aborte. Más opciones hacen cada vez más difícil la decisión del usuario. Usando un verbo en las opciones el usuario sabe inmediatamente para lo que está eligiendo, mientras que las opciones de Si/No requieren que el usuario recuerde exactamente cuál era la pregunta. La solución disminuye errores y aumenta la satisfacción.
Nombre: deshacer.	Ofreciendo la posibilidad de deshacer

Tipo: Patrón de dirección, retroalimentación. **Motivación:** Los usuarios pueden realizar acciones que después querrán deshacer. **Principio**: Manejo de errores (Seguridad). **Solución**: Dejar que los usuarios deshagan las acciones realizadas. Mantener una cola de ejecución con los comandos y permitir deshacer al menos la última pareja de acciones realizadas. Es importante que todos los comandos se puedan deshacer. Si un comando no se puede deshacer, se debe advertir al usuario antes de la ejecución.	siempre, los usuarios tienen un sentimiento de confort. Los usuarios pueden cometer errores y volver fácilmente un paso atrás, esto es una facilidad para aprender las funcionalidades de la aplicación. Esto elimina la necesidad de mensajes de error.
Nombre: Softkeys. **Tipo**: Patrón de navegación. **Motivación:** Los usuarios necesitan conocer como acceder a las funcionalidades. **Principio**: **Solución**: La solución es utilizar los mismos botones para distintas funciones mientras se mantiene la localización Colocar los botones al borde del espacio de trabajo y las etiquetas dentro del área de trabajo, o si es posible, en los botones mismo. Un softkey es un botón cuya función cambia de acuerdo a lo que se despliega encima de él. Cuando el espacio en la pantalla es limitado, usar botones físicos, sino usar "soft-buttons".	Fijando la localización de los botones, la funcionalidad siempre es accesible de la misma manera. Los botones solo ocupan un espacio mínimo y el espacio de la pantalla es optimizado.
Nombre: Espacios de navegación. **Tipo**: Patrón de navegación. **Motivación:** Los usuarios necesitan acceder a información que no puede ser desplegada completamente en el espacio disponible. **Principio**: Mapeo natural. **Solución**: Mostrar la información en distintos espacios y permitir a los usuarios navegar entre ellos. Agrupar los elementos de información en espacios separados. Permitir a los usuarios seleccionar solo un espacio a la vez. Cada espacio debe ser etiquetado con el nombre de la categoría. Todos los espacios individuales deben ser accesibles con una acción en el área especial para la navegación. Las áreas de navegación deben tener lugar arriba o a la izquierda del espacio y debe estar conectada con las áreas de información. Si el número de espacios es chico (menos de 8) la navegación puede estar arriba. Cuando el número es más grande el área de navegación deben colocarse en el lado izquierdo del espacio, usando una estructura de árbol.	Panel de planificación/configuración. Agrupar los elementos hace que sea más fácil para los usuarios encontrar un elemento particular. Colocando la navegación arriba o a la izquierda reduce el espacio necesario en la pantalla. En la sociedad occidental la lectura es de izquierda a derecha y de arriba hacia abajo. La solución mejora el tiempo del funcionamiento.
Nombre: Contenedor de navegación. **Tipo**: Patrón de navegación. **Motivación:** Los usuarios necesitan encontrar un ítem en una colección de contenedores. **Principio**: Agrupar los elementos (mapeo natural). **Solución**: Dividir la pantalla en tres áreas mostrando los contenedores y la selección final. Dividir una ventana en tres paneles, uno para ver una colección de contenedores, uno para ver el contenido de un contenedor, y uno para visualizar los ítems individuales. La selección de un contenedor debe determinar el contenido del contenedor, y el ítem seleccionado debe determinar el contenido del ítem. Las selecciones se pueden utilizar como parámetros a las funciones invocadas. Cada panel se debe especializar al tipo de contenido que presenta. Los paneles	Árbol de ejecuciones. Configurando los paneles según la manera occidental de la lectura (izquierda a derecha, arriba hacia abajo), apoyamos la relación causal basada en la selección. La disposición debe ayudar a entender la causalidad entre los paneles. Cada panel se puede adaptar al dominio y al usuario. Los paneles de tamaño modificable dan mayor libertad del usuario.

deben ser individualmente "scrollables" y se debe poder modificar su tamaño.	
Nombre: List browser. **Tipo**: Patrón de navegación. **Motivación:** Los usuarios necesitan examinar o procesar distintos ítem fuera de un conjunto o una lista. **Principio**: Minimizar acciones (mapeo natural). **Solución**: Buscar un orden natural para permitir a los usuarios navegar directamente desde un ítem al siguiente y regresar De acuerdo con el conocimiento de la tarea, se puede imponer un orden en el conjunto o lista. Para cada ítem que se presente al usuario, un widget de navegación permite que el usuario elija el ítem siguiente o anterior en la lista. El criterio de ordenación debe ser visible (cuando es configurable). Se debe apoyar la orientación, permitiendo ver el ítem actual y una lista parcial del índice. Si la complejidad no es un problema, el indicador del número del ítem actual podría ser un diálogo que permite saltar a cualquier ítem. También se podrían agregar elementos para ir al primer y último ítem.	Lista de algoritmos a elegir al lanzar una ejecución. Dando al usuario un modelo simple de navegación y permitiendo que el usuario vaya directamente al artículo siguiente o anterior, el usuario no necesita ir de nuevo al índice a seleccionar el artículo siguiente. La solución mejora la velocidad y la satisfacción del funcionamiento.
Nombre: Paneles colapsables. **Tipo**: Patrón de presentación. **Motivación**: los usuarios necesitan acceder a información o funcionalidades, pero solo bajo ciertas circunstancias (temporalmente). **Principio**: visibilidad. **Solución**: Crear paneles que puedan ser abiertos o cerrados independientemente uno de otro.	Los paneles de la presentación.
Nombre: Paneles movibles. **Tipo**: Patrón de presentación. **Motivación**: los usuarios pueden visualizar diferentes partes de la interfaz que no necesitan estar en un determinado orden. **Principio**: visibilidad. **Solución**: Crear paneles que puedan reubicarse en la interfaz independientemente uno de otro. De esta manera se le da control al usuario sobre la presentación. Para hacerlo es recomendable que el usuario lo haga haciendo click sobre el título del panel que desea mover, y que pueda arrastrarlo y soltarlo en la sección de la ventana donde desea ubicarlo. Se recomienda dar también la opción de volver a la presentación por defecto.	Los paneles de la presentación se pueden hacer movibles.
Nombre: historia de comandos. **Tipo**: Patrón de navegación. **Motivación**: el usuario realiza acciones y desea recordar qué acciones realizó. **Principio**: navegación. **Solución**: mantener un registro de las acciones realizadas.	Historial de espacios de trabajo. Historial del undo/redo..
Nombre: cuadro de comentario. **Tipo**: **Motivación**: los usuarios desean comentar sobre un determinado artículo o producto. **Principio**: retroalimentación. **Solución**: Agregar un cuadro para que el usuario pueda ingresar comentarios.	Comentarios al agregar módulos, algoritmos.
Nombre: selector de fecha. **Tipo**: Patrón de selección. **Motivación**: los usuarios necesitan ingresar una fecha o	Cuando se selecciona la fecha de ejecución.

período de tiempo sin saber exactamente la sintaxis. **Principio**: visibilidad. **Solución**: Agregar un selector de calendario y deshabilitar el campo en el que se ingresa la fecha. Esto evita errores de sintaxis y de elegir una fecha que no exista.	
Nombre: página de ayuda. **Tipo**: **Motivación**: los usuarios pueden tener problemas buscando lo que necesitan o tienen problemas con las funcionalidades. **Principio**: manejo de errores, retroalimentación. **Solución**: colocar un link o botón al índice de la ayuda que contiene los problemas más usuales.	La página de ayuda.
Nombre: Títulos en secciones. **Tipo**: Patrón de navegación. **Motivación**: los usuarios tienen muchas secciones para visualizar a la vez y puede ser difícil de entender. **Principio**: Visibilidad, mapeo natural. **Solución**: Poner títulos que definan exactamente lo que se ve en cada sección de la interfaz.	Títulos de cada panel.
Nombre: tablas ordenables. **Tipo**: Patrón de selección. **Motivación**: los usuarios visualizan mucha información que el usuario debe analizar. **Principio**: visibilidad, mapeo natural. **Solución**: Ofrecer la posibilidad de ordenar los datos por una cierta columna en la tabla. Esto facilita el análisis y exploración.	Tabla por orden y visualización de columnas.
Nombre: filas alternadas en la tabla. **Tipo**: Patrón de selección. **Motivación**: los usuarios visualizan mucha información que el usuario debe analizar y se puede confundir. **Principio**: visibilidad, mapeo natural. **Solución**: Ofrecer la posibilidad de alternar los colores de una fila y otra de las tablas, de esta manera el usuario puede seguir las filas sin confusión.	Esto se puede usar en las tablas de datos de un diario.
Nombre: completitud. **Tipo**: Patrón de selección. **Motivación**: el usuario necesita saber cuanto falta para completar una tarea o para que termine. **Principio**: **Solución**: mostrar una barra que indique cuánto se ha hecho de un proceso y cuánto falta.	Esto si bien puede ser útil para las ejecuciones de las heurísticas, no se puede usar porque no se puede saber con exactitud cuánto va a demorar una ejecución.
Nombre: visualización adaptable. **Tipo**: Patrón de selección. **Motivación**: permitir la presentación de la información según las necesidades de los usuarios. **Principio**: mapeo natural. **Solución**: Permitir mecanismos para cambiar la forma en que se muestra la información.	Control de zoom in out. Control de tamaño de fuente. Control de colores. Control de columnas que se visualizan en la tabla. Personalización de avisos sonoros o visuales. La experiencia del usuario se maximiza porque se realiza a la medida de sus necesidades.
Nombre: habilitación bajo demanda. **Tipo**: Patrón de selección. **Motivación**: existen varias opciones a seleccionar y algunas dependen de otras, si se selecciona una opción sin activar otra primero puede llevar a errores. **Principio**: manejo de errores. **Solución**: evitar ingresar errores en campos que no deberían tener datos, a menos que se haya ingresado información en un campo previo del cual depende. Los campos se deben habilitar	Cuando se habilitan secciones en los paneles de preferencias.

o deshabilitar según el estado de la aplicación, si se ingresaron datos en otros campos previos o si se seleccionó una determinada opción previamente.	
Nombre: mostrar bajo demanda. **Tipo**: Patrón de selección. **Motivación**: existen varias opciones a seleccionar y algunas dependen de otras, si se selecciona una opción sin activar otra primero, puede llevar a errores. **Principio**: manejo de errores. **Solución**: evitar ingresar errores en campos que no deberían tener datos, a menos que se haya ingresado información en un campo previo del cual depende. Los campos se deben mostrar u ocultar según el estado de la aplicación, si se ingresaron datos en otros campos previos o si se seleccionó una determinada opción previamente.	Barra de botones para el armado.

Tabla A-5:Patrones de diseño de interfaces

B Anexo: Planificación del transporte

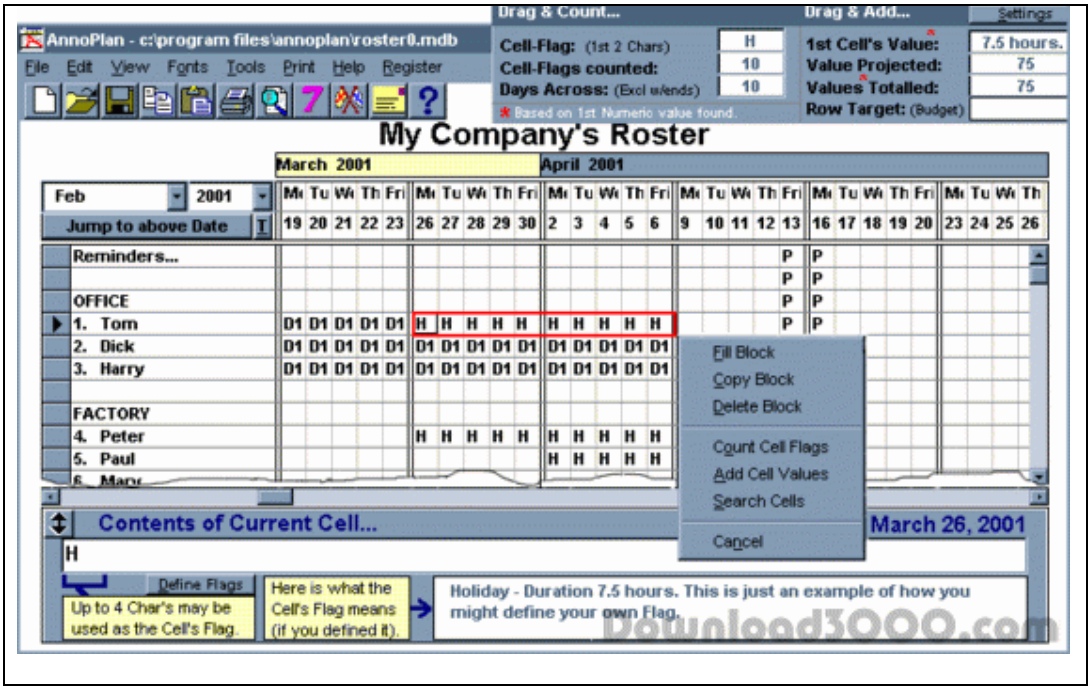
En este anexo se presenta la información técnica sobre las herramientas de planificación del transporte relevadas y las especificaciones de los algoritmos implementados por el departamento de Investigación Operativa.

B.1 Herramientas generales

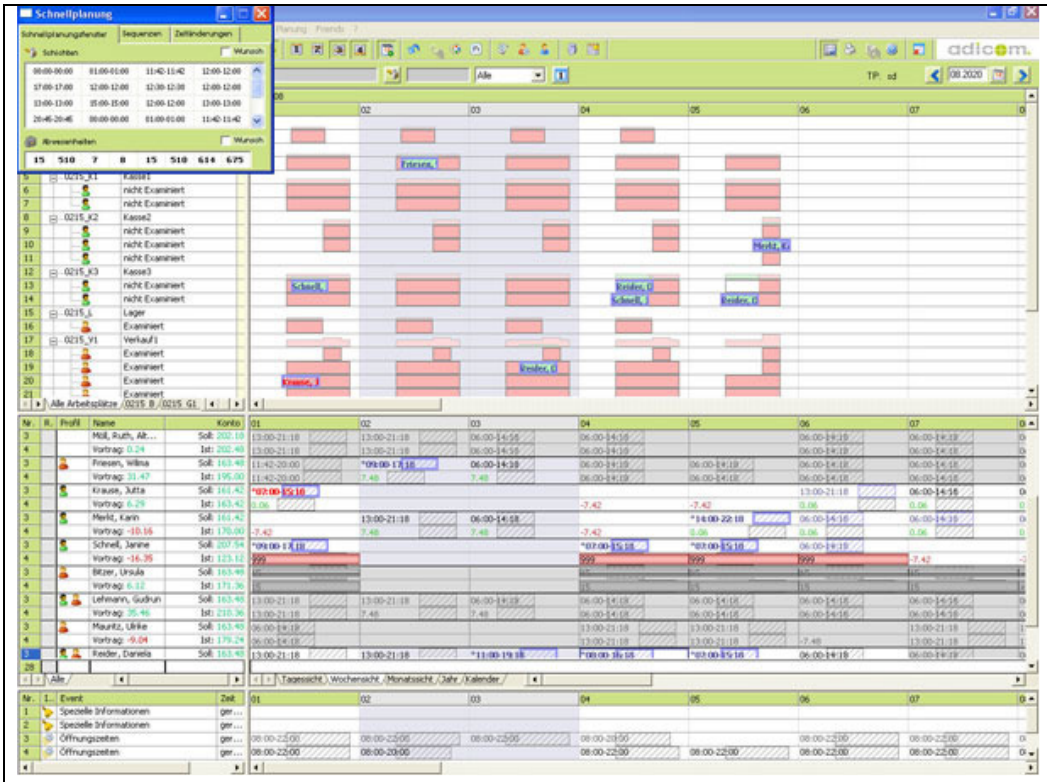
En esta sección se presentan algunas herramientas de planificación de propósito general relevadas para tomar ideas para el proyecto. Toda la información e imágenes presentadas en cada ficha fue extraída del link indicado en la misma.

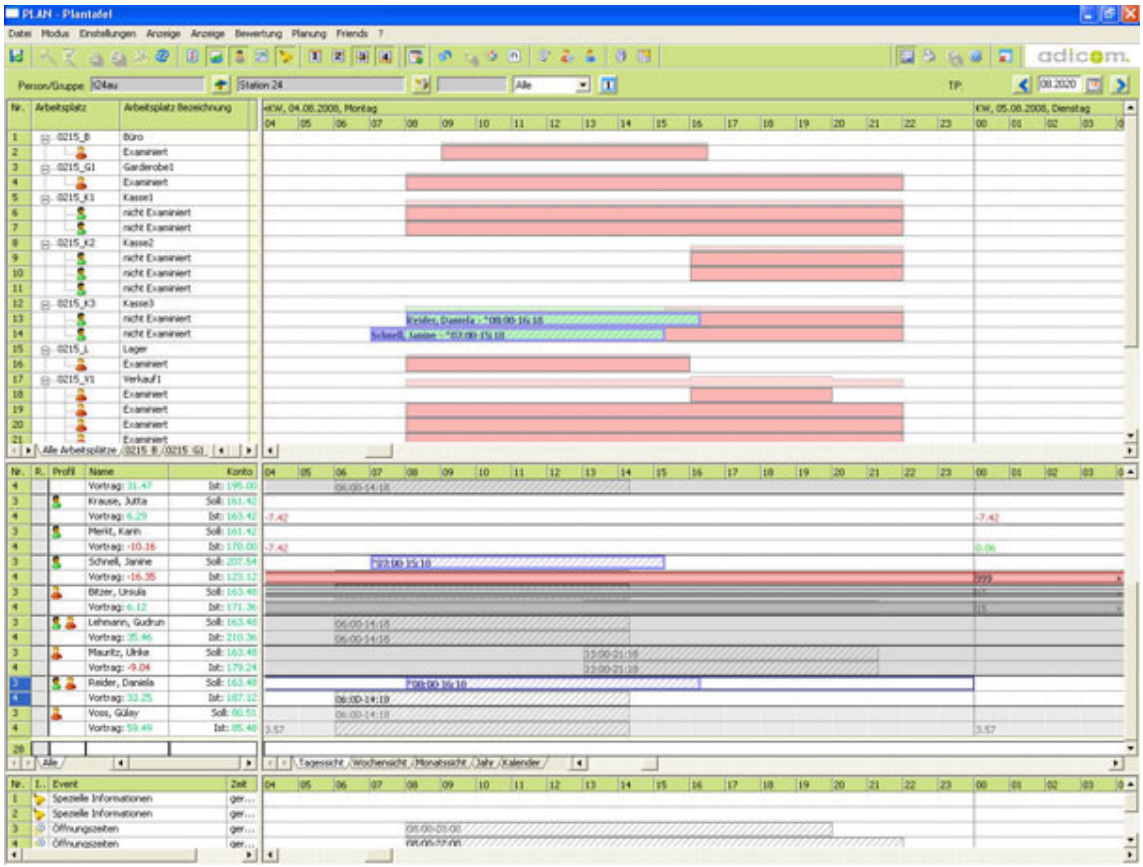
ACS Heim
Características
Link: http://www.alphacomputer.de/software/software.php?id=heim (último acceso: 22/03/2010) *Generalidades*: ACS Heim es una solución de software claramente estructurado para instalaciones fijas de cualquier tamaño y de la propiedad. A través de su diseño modular puede ser adaptado a su organización. 

Annoplan
Características
Link: http://www.download3000.com/download_24124.html (último acceso: 22/03/2010) *Generalidades*: AnnoPlan es una hoja de cálculo poderosa, basada en la fecha, diseñado para flexibilizar turnos de personal, equipos, etc. de habitaciones. También es bueno para la planificación y cálculos con el tiempo. Al igual que con la mayoría de las hojas de cálculo, la pantalla se rellena con células de ancho variable, pero las células del AnnoPlan están vinculadas a un calendario automático en la parte superior. Arrastre el puntero sobre las celdas para obtener la suma o la proyección. Defina sus propias marcas o simplemente introduzca texto/valores en las celdas de ancho variable. Este programa tiene un intervalo de fechas de 1928 a 2097, y puedes hacer búsquedas de la gama de todo el año. Los datos se guardarán en una base de datos de red. La cuadrícula es conmutable - 5 o 7 días de la semana. Listas de impresiones pueden verse fácilmente y su significado. Botón de desplazamiento horizontal muestra la fecha deseada conforme mueva. Nuevo botón de correo electrónico crea una vista JPEG de tu lista actual, lo auto-atribuye a una composición de correo electrónico en blanco, lista para poder enviar. Baja utilización de espacio en disco - no tiene que contener todas las celdas en el disco - sólo los que contienen datos. Esto significa que usted puede instantáneamente saltar hacia adelante de 50 años y introducir datos y deslizarse hasta el presente.



ADICOM
Características
Link: http://www.freudenberg-it.de/Personaleinsatzplanung.FIT?ActiveID=1193 (último acceso: 22/03/2010) *Generalidades*: En la organización del tiempo de trabajo personal y operacional, la planificación de estrategias es cada vez más importante. Una de nuestras fortalezas es reflejar fielmente con el mínimo esfuerzo. Porque con el personal adecuado y la estrategia de implementación adecuada se pueden crear beneficios inmediatos de costo y productividad. Permite insertar sus diferentes necesidades y obtener los empleados que encajan y sus calificaciones.





Esto requiere un software flexible y abierto como adicom. Esto también garantiza sus necesidades y ofertas incluyendo:

- Perfiles de habilidades de planificación de necesidades de materiales.
- Propuesta de planes mensuales basados en la gestión del tiempo integrado.
- Interfaces para los diversos sistemas de salario/sueldo.
- Diarios de tiempo para todos los tipos de transacción.
- Arancelarias de industria-neutral y libres de las normas y los informes compatibles con el BAT.

Crear una base excepcional que se adapte a sus necesidades y seguirá de ello, en el largo plazo, a sus ideas y objetivos con la planificación de despliegue de personal de adicom.

Schichtplaner 4

Características

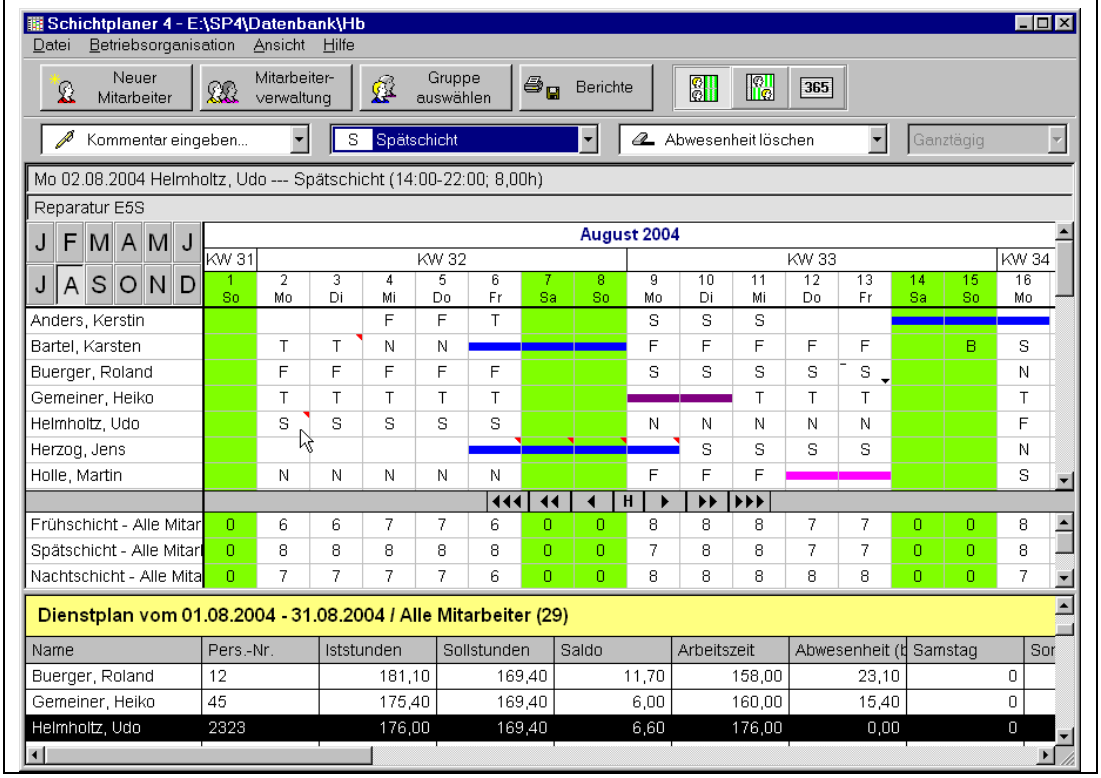
Link: http://www.schichtplaner.de/schichtplanung.htm (último acceso: 22/03/2010)

Generalidades:

Schichtplaner 4 es un programa planificación de recursos humanos cómodo, para empresas de todos los tamaños y sectores. Ningún conocimiento especial es necesario para la operación.

Características clave:

- Administrar cualquier número de empleados.
- Datos relacionados con los empleados (reales / horas planificadas, etc.).
- Cálculo de las primas del tiempo de trabajo.
- Definir sus propios tipos de capas y motivos de ausencia.
- Formación de grupos y subgrupos (grupo de trabajo, las calificaciones, etc.).
- Valor predeterminado de dotación de personal necesario para cualquier grupo.
- Marcado de las capas insuficientes.
- Servicios de registro y ausencias con el mouse.
- Servicio de conexión y entradas de oficina con comentarios.
- Definición de modelos de capa.
- Informes significativos (planes, estadísticas, listas).
- Guardar las listas en formato HTML.
- Exportación de datos e importación.
- Creación de redes.
- Administración de usuario integrada a través de la interfaz de usuario de habla alemana y documentación.



B.1.1 Observaciones:

Del relevo de sistemas de planificación de tripulación y vehículos se desprenden las siguientes conclusiones:

- Permitir múltiples vistas de los datos, en diferentes formatos para distintos tipos de análisis.
- Guardar solo los datos necesarios, no ocupar mucho espacio de disco.
- Proveer de datos siempre actualizados.
- Proveer de mecanismos de definición de reglas y parámetros específicos del caso.
- Personalización de las vistas.
- Exportación de datos y expedición de reportes.

B.2 Herramientas de transporte

En esta sección se presentan las herramientas de planificación del transporte relevadas para tomar como ejemplo para el proyecto. Toda la información e imágenes presentadas en cada ficha fue extraida del link indicado en la misma.

EMME/Dynameq/STAN

Características

Link: http://www.inro.ca/en/products/index.php (último acceso: 22/03/2010)
Generalidades:
Emme proporciona un enfoque unívocamente flexible y abierto a la elaboración de modelos que permite a los usuarios la libertad de aprovechar técnicas establecidas o crear nuevos métodos para satisfacer las necesidades locales. Modeladores y planificadores han dependido de Emme durante más de dos décadas, y hoy en día Emme impulsa parte del transporte más sofisticado del mundo con sus modelos de planificación.
El Framework de modelado de Emme Core permit:

- Análisis de la ruta de transporte privado – asignación de tráfico.
- Análisis de la estrategia de transporte público – asignación de tránsito.
- Matriz de modelado de demanda – matriz de cálculos, equilibrio de procedimiento.
- Análisis y automatización – red de calculadoras, marco de automatización.

El marco de modelado de Emme Core ofrece flexibilidad inigualable para abordar las preocupaciones locales y oportunidades para ampliar e innovar. El núcleo de Emme es un kit de herramientas de módulos diseñados para facilitar el descubrimiento y desarrollo de modelos de planificación de transporte. Ofrece al planificador un conjunto completo e integral de herramientas para el modelado de la demanda, red multimodal de modelado y análisis y la aplicación de procedimientos de evaluación.

Dynameq es una nueva raza de equilibrio dinámico de asignación de tráfico (DTA) para su uso en grandes redes congestionadas. Dynameq proporciona a los planificadores de una vista de las condiciones del tráfico dinámico y ofrece comparaciones de escenario racional que sólo son posibles con una solución basada en el equilibrio.

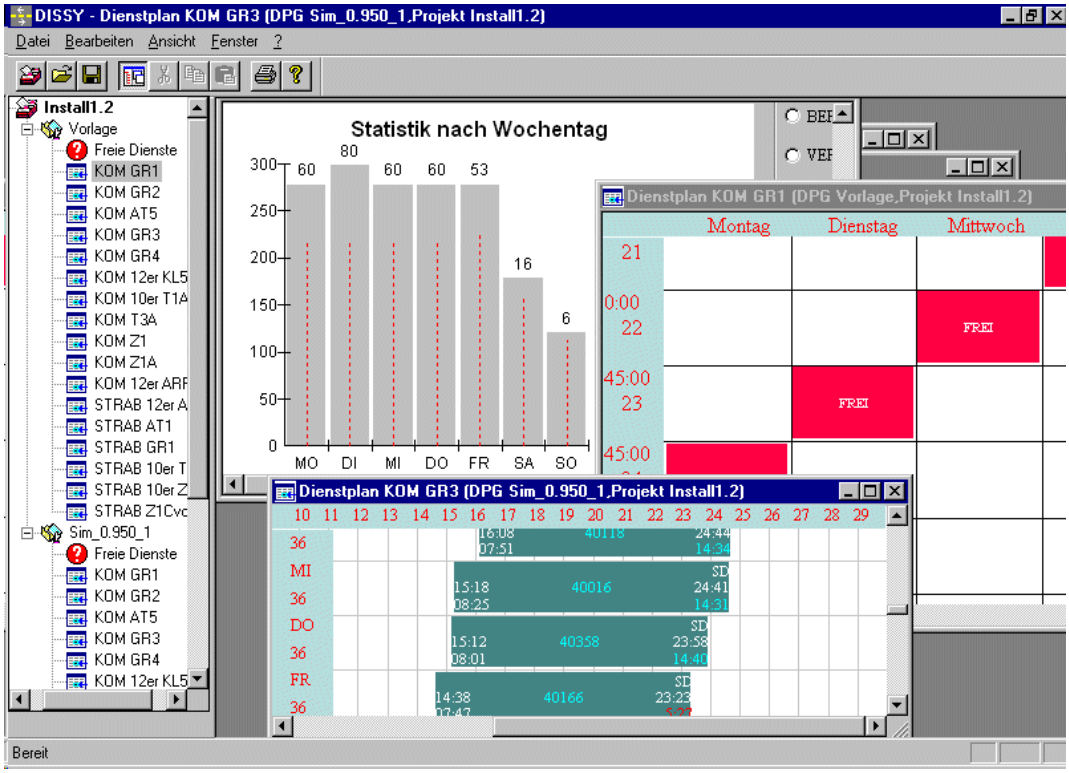
STAN es una herramienta de planificación de gráfico interactivo que responde a las necesidades de cargadores y transportistas eficientemente para evaluar el impacto de cambios importantes en la infraestructura de transporte, el entorno reglamentario y patrones de demanda sobre su costo, tiempo, fiabilidad y otras medidas de rendimiento. Proporciona un marco flexible y general para la representación de la infraestructura y el modelado de las actividades de los sistemas de transporte o distribución de transporte multimodal y multiproducto dinámico. Es un sistema de soporte de decisión que proporciona un manejo de datos uniforme y eficiente, incluida la validación de consistencia de datos completos sobre la

entrada. Su banco de datos está estructurado para permitir el análisis simultáneo de varias alternativas de planificación.

DISSY

Características

Link: http://www2.informatik.hu-berlin.de/alcox/forschung/DISSY/ (último acceso: 22/03/2010)
Generalidades:
Los objetivos del proyecto DISSY son el desarrollo, la evaluación y la difusión de una simulación basada en HPCN y un sistema de soporte a las decisiones para el problema de asignación de turnos en el transporte público urbano. Permite simular y evaluar diferentes escenarios de planificación, lo que permite adaptar mejor la gestión de recursos humanos a las nuevas condiciones de mercado. Multiplica la productividad de los usuarios del departamento de planificación y produce una transparencia en la gestión y los controladores. Conduce a una nueva flexibilidad en transporte público urbano, la herramienta integrada ofrece nuevos impulsos para innovar el proceso de negocio de las empresas de transporte y otras industrias mediante el trabajo por turnos.



HASTUS ("Horaires et Assignations pour Système de Transport Urbain et semi-urbain")

Características

Link: http://www.giro.ca/en/products/hastus/index.htm (último acceso: 22/03/2010)
Generalidades:
HASTUS es una solución de software integrada y modular para la planificación de tránsito, sus operaciones y la información del cliente. La eficacia y la flexibilidad de HASTUS queda demostrado por las instalaciones en Norte y Sudamérica, Europa, Australia y Asia. Gracias a la incorporación de más de 30 años de experiencia de GIRO y sugerencias de nuestros clientes, HASTUS es una de las mejores inversiones que pueden hacer las autoridades de tránsito y los operadores privados. El sistema básico de HASTUS proporciona las herramientas necesarias para producir planificaciones de vehículos y tripulación eficientes, utilizando optimizadores líderes en la industria. Módulos totalmente integrados trabajan desde horarios para gestionar

las operaciones diarias de forma precisa. Otros módulos disponibles ofrecen soluciones para la planificación de la red de tránsito, la información del cliente y el análisis de rendimiento. La base subyacente, fácilmente accesible a otros sistemas a través de herramientas de integración global, ofrece una fuente sencilla, fiable y completa de datos de las operaciones de tránsito. Además, nuestro equipo de profesionales brindan soporte oportuno y un amplio conocimiento de las mejores prácticas en la industria del transporte público.

HASTUS ofrece una solución de software completa, modular e integrada para las operaciones de tránsito. Módulos básicos cubren las operaciones diarias y planificación, mientras que ofrece módulos complementarios disponibles para la planificación y análisis, así como información de los clientes. Para la planificación, herramientas flexibles le ayudarán a describir su red de rutas y la definición de horarios deseados. Potentes algoritmos crean viajes de ahorro de costos y horarios de la tripulación, así como listas de uno a varios días. Soluciones automáticas pueden ser validadas y afinadas, según sea necesario, con vistas basadas en listas y gráficas. Módulos de operaciones diarias desde los horarios previstos le permiten administrar los cambios de servicio diarios, así como las asignaciones de vehículo y operador. Información completa y actualizada está disponible en todo momento, lo cual lo hace líder en la mejor toma de decisiones "en vivo". Puede comparar los horarios reales frente a los previstos, supervisar las estadísticas de rendimiento de operador y exportar los datos de la hora exacta para la nómina. El módulo de SelfService, basado en Web, permite a los empleados acceder a su información personal y realizar diversas tareas, tales como ingresar en horas extraordinarias, manejar disponibilidad y preferencias. Beneficios adicionales se consiguen a través de una reducción significativa en los servicios, ahorrando tiempo y minimizando errores.

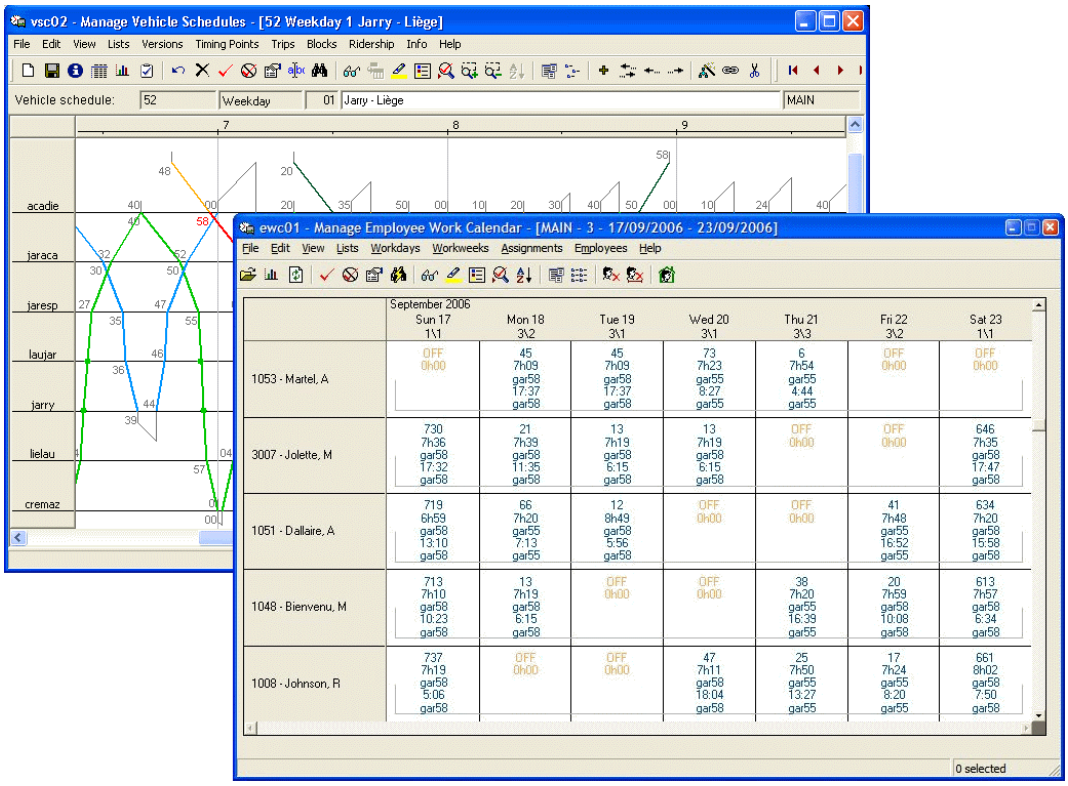
Provee información de atención al cliente de calidad, lo que refleja que tan bien planeado está o brindando información en tiempo real. Esto incluye la planificación de viajes o manejar el nivel de información mostrada en el sitio Web o en carteles de parada, así como a través de otras tecnologías, como sistemas IVR y señales electrónicas. Comentarios de clientes también pueden ser grabados e inmediatamente asignados a la información de servicio correspondiente.
Análisis y herramientas de planificación proporcionan la red y horarios de modelado de funciones, incluyendo sincronización mientras minimiza los requisitos del vehículo. Módulos de ejecución de análisis de datos de tiempo y el número de pasajeros contribuyen a mejorar la utilización de su flota y al servicio que mejor se adaptan a las necesidades del cliente. En pruebas repetidas y las implementaciones en todo el mundo, los algoritmos de planificación HASTUS generan ahorros que van desde 2 hasta el 5% y más, comparado a los sistemas de la competencia y métodos manuales. Estos potentes algoritmos hacen uso de un motor de reglas flexibles que producen excelentes resultados en muchos contextos diversificados de funcionamiento con el software estándar.

HASTUS es el fruto de las investigaciones en curso y desarrollo en GIRO, en colaboración con de la Universidad de Montreal internacionalmente reconocido como centro de investigación en transporte. Constantes mejoras en los métodos de optimización y características del sistema garantizan que HASTUS sigue siendo el líder en este campo altamente especializado, aportando beneficios adicionales a los clientes con cada actualización. El software utiliza el servidor Windows y la plataforma de escritorio, proporcionando un entorno de escritorio estable, productivo y familiar. La base de datos relacional puede alojarse en un servidor de base de datos compatible con ODBC Oracle o MS SQL Server. La arquitectura escalable es compatible con cualquier tamaño de organización, desde usuario independiente, a las instalaciones de toda la empresa con cientos de usuarios.

Modelos de entrega de aplicaciones
Las aplicaciones abarcan el espectro de modelos de configuración cliente-servidor, tres niveles y basada en la Web. Las instalaciones del cliente y el servidor se ejecutan en Intel - AMD o Windows, a través de una red de área local. Permite múltiples sitios remotos y los usuarios pueden utilizarlo con Windows Terminal Server, opcionalmente con Citrix Presentation Server para facilitar la administración adicional. Esta configuración de tres niveles permite clientes "thin" y acceso basado en navegador. Procesos de optimización por lotes se gestionan a través de un trabajo integrado con una cola de instalación. Se pueden ejecutar localmente en la

estación de trabajo de un usuario, o ser conducidas entre una "granja informática", compuesta por cualquier número de equipos.

Aplicaciones basadas en Web
Las aplicaciones basadas en la Web están disponibles para las funciones de hacer frente a una mayor base de usuarios, tales como operadores de tránsito o el público en general. Estos incluyen un módulo de SelfService para operadores y tripulaciones, así como el planificador de viajes HASTINFO, en el contexto del transporte público. Las aplicaciones basadas en la Web también están disponibles para nuestros GeoRoute (datos y actualización de ruta de prestación de servicios) y productos de software GIRO/Access (solicitudes de viaje en el contexto de paratránsito). Las aplicaciones Web se validan con todos los navegadores actuales y que siguen las directrices de accesibilidad. Se desarrollaron usando las tecnologías .NET de Microsoft y se ejecutan en Windows IIS Server con ASP.NET. Se recurrió a CSS -hojas de estilo en cascada– para proveer una fácil personalización, satisfacer el color corporativo y las guías de diseño y su integración en los sitios corporativos de intranet e Internet.



PlanOp

Características

Link: http://www.jeppesen.com/industry-solutions/land/postal/transportation-network-planning.jsp (último acceso: 22/03/2010)
Generalidades:
El objetivo es planear redes de distribución local, nacional o mundial rentables y de alta calidad, proveer software de optimización y modelado de la red de transporte para empresas de courier, integradores y reenviadores y otras organizaciones de logística.
PlanOp ofrece:

- Un diseño de red robusto y eficiente.
- Costos de distribución reducida.
- Escalas de tiempo de planificación reducidas.
- Capacidad de modelar, cuantificar, analizar y comunicar las opciones de red.
- Conocimiento de la red mejorada.

- ❖ Tamaño de flota de vehículo optimizado y mezcla.
- ❖ Capacidad para desarrollar planes de contingencia.
- ❖ Personalización para requerimientos específicos.

PlanOp y Jeppesen pueden ayudarle a:

- ❖ Diseño de redes complejas de transporte.
- ❖ Tomar decisiones estratégicas y tácticas sobre la estructura de red y sus opciones
- ❖ Desarrollar un repositorio de datos y el modelo de red detallada y realista.
- ❖ Planificar y ejecutar operaciones eficientes de vehículos y la tripulación.
- ❖ Facilitar la planificación de operaciones diarias.
- ❖ Reune los detalles de los tipos de vehículos, tiempos de viaje, ubicaciones y horarios de ordenación, tipos de productos y normas de servicio en el modelo de una amplia red que permite el acceso fácil a los diagramas visuales e informes. Los motores de optimización incluyen:
  - o Optimizador de flujo de red.
  - o Optimizador de enrutamiento de vehículo.
  - o Optimizador de planificación de tripulación y vehículos.
  - o Optimizador de selección de concentrador.

Utilice PlanOp para estos y otros servicios:

- ❖ Agregación de red.
- ❖ Optimización de funciones del controlador.
- ❖ Contenedorización.
- ❖ Comparar y resolver las diferencias entre los planes.
- ❖ Capacidad de animación y simulación.

DIVA

Características

Link: http://www.mentzdv.de/englisch/products/diva/ (último acceso: 22/03/2010)

Generalidades:

Los operadores de transporte y las autoridades tienen que realizar una multitud de tareas. No sólo hacer la planificación, con cuestiones tales como el calendario y la planificación de personal y vehículos, sino que deben hacerlo adecuadamente y de manera eficiente. Además, un conjunto de medios de comunicación debe ser capaz de proporcionar convenientemente a pasajeros con la mejor y más actualizada información acerca de los servicios prestados por la autoridad/operador de transporte que van desde los horarios a la información a través de terminales móviles.

El tipo de operador también debe incluirse en la ecuación: un operador urbana con niveles de alta frecuencia tiene diferentes requisitos que un operador que opera en las zonas más rurales. Sin embargo, a pesar de las diferencias en las necesidades operacionales, una cosa que todos los operadores tienen en común es la competencia, la constantemente creciente presión para reducir costos y proporcionar a los clientes con la más alta calidad de servicio. DIVA ayuda a los operadores de todo el mundo hacer frente a estos y otros desafíos operacionales.

Es un sistema con las siguientes funciones:

- Grabación de planificación.
- Conducción y planificación de circulación.
- Optimización del actual.
- Planificación de servicio.
- Plan de optimización de servicio.
- Administración de personal.
- Cartografía.
- Libros de planificación.
- Boletines de planificación.
- Optimización de la conexión.
- Geografía.
- Integración de distintos proveedores.
- Suministro de los sistemas operativos basados en el equipo (RBL) y ordenador a bordo.

- El calendario (EPT) de suministro.

DIVA es desarrollado desde 1987. DIVA-Windows es la tercera generación de programas DIVA después de DIVA en sistemas de Nixdorf (tecnología de datos medio) y DIVA en UNIX. Mdv está actualmente trabajando en la cuarta etapa de DIVA. Además de una formación más reciente, desarrollo tecnologías y la mayor integración de las aplicaciones Web, DIVA 4 proporcionará un mayor nivel de integración para satisfacer a los usuarios de diferentes objetivos.

Planificación de servicios
La planificación de servicios DIVA hace posible construir los horarios para los operadores de transporte. Hace posible crear funciones tomando disposiciones legales y acuerdos operativos en consideración. Es adecuado para tanto los operadores de transporte regional como urbano. Las servicios bien pueden crearse por bloques de vehículos o directamente interconexión de viajes.

La planificación de servicios DIVA incluye las siguientes funciones:
- Servicio de transferencia de pool de calendario y la planificación de vehículos.
- Ajuste de los reglamentos legales y operativas.
- Servicios de edición (edición de funciones, creación de capacidad de repuesto).
- Análisis estadísticos para controlar (estadísticas de planificación de servicios, vista de registro de servicios, curva de material de transporte).
- Planificación de servicios regional.
- Planificación de la lista de edición (lista teórica de edición, ocupación de posición de lista).
- Generación de impresión (impresión de planificación de servicios, lista de salida de almacén etc.).
- Planificación de tren.

La planificación de servicios DIVA hace posible trabajar en una forma eficiente y fácil de usar cuando mediante técnicas tales como arrastrar & soltar para la edición y gestión de servicios, comprobación de tipos de cambio, los reglamentos legales y operativos durante la creación de los servicios, etc.. Es posible trabajar en las vistas gráficas y tabulares. El usuario es proporcionado con la mejor asistencia posible por las muchas instalaciones de automatización diferentes y la utilización de procedimientos de optimización matemática.

Servicio de transferencia
Si un calendario ha sido aprobado, sus datos pueden transferirse a la planificación de servicios. Durante la transferencia, el planificador de servicios puede decidir si desea generar una planificación completamente nueva o basarlo en una planificación de servicios existente de otro proyecto. Los servicios, a continuación, son reimportados en la planificación y actualizados. Los servicios creados previamente se reproducen si es posible cuando se transfiere el pool de servicios. Se pueden definir libremente derechos adicionales para viajes especiales y pueden agregarse a la agrupación de servicio en la planificación.

Ajuste de los reglamentos legales y operativas
La funcionalidad puede ser adaptada a las necesidades del usuario durante la creación de los servicios por medio de parameterización. Los parámetros existen para la comprobación de los requisitos legales y operacionales, para calcular automáticamente los suplementos de tiempo de trabajo y para la creación automática de servicios secundarios. También se pueden insertar servicios secundarios adicionales, libremente definibles. Se puede ajustar el cálculo de los valores de destino si es necesario. Otra instalación de ajuste es proporcionada por los tipos de cambio, que se utilizan como herramientas durante la creación del servicios. Sin embargo, la creación de servicios puede también llevarse a cabo sin el uso de tipos de cambio.

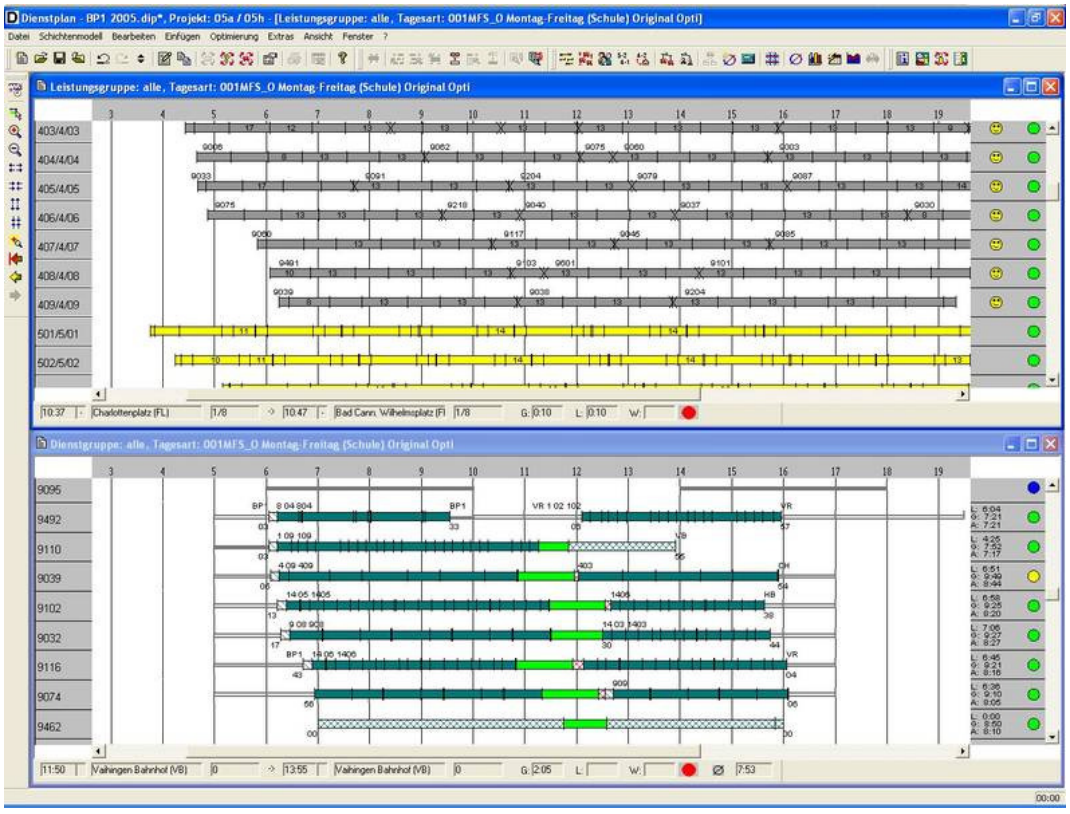
Edición de servicios
Los servicios y sus agrupaciones pueden ser manejados con arreglo a depósitos y funcionamiento de ramas y proporcionarán al usuario una manera de crear un entorno de

trabajo que se adapte a sus necesidades individuales. El cálculo de servicios de acuerdo a tarifas diferentes para grupos de controlador diferentes también es fácil de hacer de esta manera. Los servicios se editan directamente en los gráficos.

El usuario es asistido por funciones de automatización:

- Planificaciones de servicios pueden mostrarse automáticamente basándose en planificaciones existentes de destino utilizando el imitador.
- Una amplia variedad de funciones de copiar lo hacen posible copiar servicios o grupos de servicios y sus tipos de cambio de otros proyectos.
- La búsqueda anterior/siguiente ayuda al usuario en el establecimiento de servicios individuales y evalúa las alternativas que están disponibles.
- Las reglas de creación de servicios son monitoreadas durante la creación y se muestra el resultado de la comprobación.
- La creación automática de servicios secundarios agrega elementos del servicio de conformidad con las reglas definidas por el usuario, que también contribuyen a la actualización de los kilómetros.
- Formas de optimización de planificación de servicios completan horarios de servicios.

Las funciones de optimización y automatización que se proporcionan dan al usuario una herramienta eficaz para la gestión flexible de los datos de recursos. Así como la funcionalidad de creación de servicios real, instrumentos analíticos también están disponibles para el planificador. Funciones de cálculo puede usarse para estimar el número mínimo de servicios que son necesarios, o la función de curva de material de transporte puede utilizarse para mostrar gráficamente los servicios cubiertos y compararlos con funciones existentes o la selección del tipo de cambio para un grupo de destino.



El programa de ayuda al usuario durante la planificación antes de que se editan los servicios.

- Una estimación inicial es proporcionada por una función para calcular el número de cambios que son necesarios.
- La curva de material de transporte proporciona una visión general de los tipos de

cambio que se necesitan.

Otras vistas de datos estadísticos muestran el resultado de la construcción y ayudan al usuario en su evaluación:

- La obligación de registrar la vista y las estadísticas del plan de servicios proporcionan una amplia gama de datos relativos a los derechos individuales y la planificación general de servicios y, por lo tanto, hacen posible evaluar el plan según criterios tales como la eficiencia, compatibilidad social etc..

Planificación de servicios regional
La planificación de servicios DIVA también hace posible construir servicios junto con bloques (servicios = bloque). El usuario puede trabajar con tablas o gráficos. Es posible trabajar en varias pantallas de visualización. Los viajes y funciones se muestran en las vistas de gráficas o en formato tabular. Viajes y servicios se muestran como diagramas de Gantt para la edición gráfica de funciones y pueden modificarse en estas listas. Las funciones que se crean se comprueban inmediatamente y los resultados muestran la utilización de símbolos y colores apropiados. Las funciones que se crean se pueden imprimir como un gráfico de Gantt. Un folleto de planificación de servicios también puede ser extraído directamente del resultado en un diseño elaborado en formato PostScript.

Turnos
La etapa final en el proceso de planificación de servicios es los turnos, que también con frecuencia se conoce como la creación de la lista. Los servicios acabados de los grupos de derecho individual ahora se almacenarán en listas de grupos, como son llamados, por lo que se crea una secuencia de destino lógico para el conductor. Aquí es importante respetar el período de descanso nocturno y garantizar que las funciones se organizan con una cierta regularidad. Los parámetros relevantes y funciones para ambos requisitos están disponibles en DIVA.

En la planificación de servicios, el usuario puede definir diferentes listas teóricas para las diferentes listas de grupos. El administrador de la lista ayuda al usuario en la creación de la lista de destino para los grupos de las diferentes listas. Las listas que se crean inicialmente esquemáticamente pueden representarse en el calendario como una matriz de listas con roster a fin de proporcionar una asignación de controlador/servicios concreto.

Generación de documentos impresos
Las funciones que se han creado pueden imprimirse en la forma de una tabla o gráfico de Gantt. Un folleto de planificación de servicios también puede ser extraído directamente del resultado en un diseño elaborado en formato PostScript. El diseño del folleto de planificación de servicios estándar puede ser adaptado al usuario y adaptado por el propio usuario más tarde.

Planificación de tren
La planificación de servicios DIVA puede utilizarse para la planificación del tren. Cuando se importa el pool de servicios, los carros pertenecientes a una empresa de ferrocarril se reconocen y se combinan en consecuencia. Permite configurar cuando los trenes están siendo divididos o montan.

IVU.plan
Características
Link: http://www.ivu.com/products-and-solutions/busses-trains/planning/ivuplan.html (último acceso: 22/03/2010) *Generalidades*: *IVU.plan-o planes de despliegue óptimos para los vehículos y personal* IVU.plan gestiona la red y elabora el calendario, desde la planificación estratégica a través de los cambios realizados en un plazo breve. IVU.Plan utiliza algoritmos inteligentes para mejorar la circulación de vehículos y servicios – también integrados en un solo paso a petición – y al hacerlo, tiene en cuenta todas las disposiciones operativas y las especificaciones del planificador. IVU.plan es compatible con las necesidades operacionales complejas tales como

desvinculación de tren, vuelos charter y también planes para realizar un seguimiento de ocupación. El software también ofrece interfaces con otros sistemas y permite una variedad de estadísticas y documentos impresos para estar preparado, tales como horarios de salida, mostrar planes, calendarios, copias impresas de la lista y el calendario así como informes completos.

Planificación con IVU.plan.timetable.map de la red:

- Tablas y registro basado en GIS y procesamiento de la red de transporte con:
  - Los datos básicos y los parámetros necesarios.
  - Paradas y varios otros puntos de la red.
  - Rutas y las líneas con planificación diferenciada en tiempo de ejecución.
  - Registro y gestión de atributos detallados para las estadísticas etc. y suministro de interfaces a sistemas downstream (ITCS, tickets, información sobre los pasajeros).
- Integrado de enrutamiento de conexiones de viaje operacional y vacía.
- Visualización cartográfica de la red de transporte, con una función que permite la integración de mapas en línea actualizados.

Calendario de planificación con IVU.plan.timetable:

- Planificación tabular y gráfica y la modificación de los horarios con todos los detalles para los pasajeros y las operaciones.
- Validaciones precisas para el día de operación / basado en el calendario.
- Pequeños ajustes a la planificación de las obras de construcción o eventos a gran escala.
- Posibilidad de planificar las variantes para análisis de calendario y la planificación de la oferta.
- Planificación de conexión integrada.
- Impresiones de calendario, como planes de visualización, el horario de salida, el libro de calendario.
- Evaluación de calendario y estadísticas.

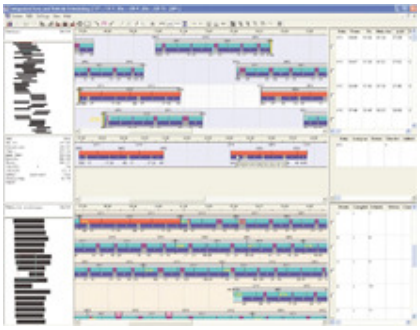
Planificación de vehículos y optimización con IVU.plan.vehicle:

- Análisis de las necesidades de planificación detallada de vehículos sumadas, tiempos de montaje y requisitos de descanso.
- Formación de planificación de vehículos: gráfico (diagrama de tiempo de la ruta y gráfico de barras, así como tabla).
- Optimización de vehículos integrada con mezcla de vehículos detallada y requisitos para la distribución de energía, análisis de sensibilidad con la modificación del calendario de planificación.
- Impresiones de planificación del vehículo, como tarjeta de coches de tren, vehículo gráfico, la planificación impresa, extracción / listas de pull-in, visión general de las líneas.
- Evaluación de planificación de vehículo y estadísticas.

Planificación de tareas con IVU.plan.duty:

- Análisis de todos los parámetros necesarios y requisitos, incluso reglas extremadamente complejas pueden ser registradas.
- Generación de tareas manual y automáticamente optimizada basándose en las planificaciones de vehículos.
- Generación de planificación de vehículos manual y automáticamente optimizada en un procedimiento integrado sobre la base de los viajes.
- Se cumplen todas las reglas definidas y elementos de derecho adicional, como la puesta en marcha y tiempos de take-down, saltos, tiempos de viaje, las tarifas y los suplementos que agrega automáticamente por el sistema.
- Posibilidad de tripulación múltiple y formación de equipos, por ejemplo, para el ferrocarril y el tráfico de envío.
- Impresiones de tareas como copia impresa de las tareas de servicios urbanos y regionales, copia impresa de gráfico de tareas y salto de planificaciones.

- Análisis y estadísticas de tareas.



Tracs II

Características

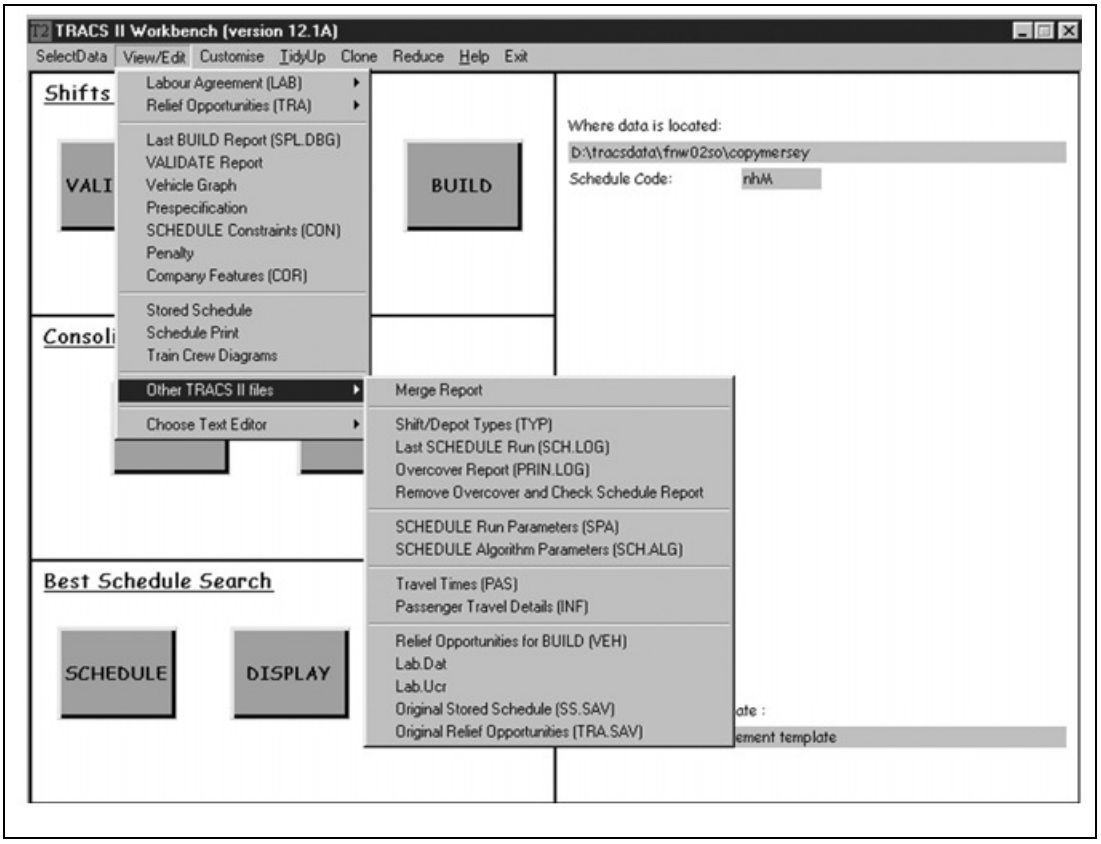
Link: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.16.112 (último acceso: 22/03/2010)

Generalidades:

Tracs II es un sistema de planificación de tripulación consistente en un conjunto de módulos. Está diseñado para ser tan portable como sea posible, por lo que su componente optimizador principal es independiente de las reglas de construcción de cualquier turno. Una versión anterior llamada IMPACS fue instalada en los ómnibus de transporte de Londres en 1984 y ha sido usado como el principal planificador de horarios de tripulación en Londres desde ese entonces. IMPACS y Tracs II juntos han sido usados para construir horarios operacionales reales en unas 100 compañías de ómnibus y trenes, tanto como herramienta de planificación de tripulación regular como para testear estrategias alternativas o para realizar mejoras a los horarios. Recientemente, Tracs II ha sido instalada en cada una de las 25 compañías operadoras del grupo de transporte más grande del Reino Unido, con unos 10.000 vehículos. Cada una de estas compañías tiene un historial diferentes, algunas son antiguamente dependientes del gobierno, otras son compañias privadas regionales, mientras que dos son divisiones de la antes conocida London Transport. Las condiciones de planificación, por lo tanto, varían enormemente según la empresa.

Esencialmente, Tracs II ha tenido dos fases principales: generación de un gran conjunto de turnos potenciales, posiblemente hasta algunos millones o más, y selección de un subconjunto de esos turnos que compondrán la planificación. En la práctica, existen muchas etapas en el proceso de solución de Tracs II, controladas a través de un workbench o interfaz de usuario:

- Preparación de datos, entrada y validación.
- Especificación previa.
- Identificación de los posibles viajes.
- Generación de uno o más conjuntos de turnos potenciales válidos.
- Reducción del conjunto de turnos.
- Mezcla de los conjuntos de turnos.
- Ejecución de reglas.
- Selección de un subconjunto de turnos que cubren el trabajo vehicular y minimiza una función objetivo.
- Post-procesamiento.
- Salidas.
- Validación de turnos.



Vera
Características
Link: http://www.mentzdv.de/englisch/products/vera/ (último acceso: 22/03/2010) *Generalidades*: El nuevo procedimiento VERA realiza registros parciales del viaje actual y examina las paradas de origen y destino del viaje en general. La posibilidad de bajar en una parada antes o después se simula, restringido por las condiciones de aplicabilidad del viaje parcial en cuestión. Los programas están disponibles para la gestión básica, el estudio, el recuento y los datos del viaje, así como los programas para la inclusión y el control de la encuesta y los datos de conteo. Los programas de lista y control realizan una comprobación de plausibilidad a fondo. La información pertinente secundaria, como entradas, el propósito del viaje, la ruta con la 1ª clase, los tipos de vehículos, conjuntos de vehículos (trenes), el calendario y la red son verificadas. Se presta atención a los siguientes criterios, entre otras cosas: ❖ Toda la ruta se almacena en los tiempos del calendario, y toda la información de la red conocidos relacionados con las transferencias se tienen en cuenta. ❖ Relación de la línea recta entre el origen y destino del viaje según la longitud de la ruta de viaje. ❖ Velocidades de transporte. ❖ Veces que se realizan los viajes. ❖ Relación entre los tiempos de espera necesarios durante las transferencias a los viajes posteriores. ❖ Uso de múltiple enlaces y paradas. ❖ Datos de la encuesta se comparan con los resultados del recuento como preparación para la proyección.

Se verifica la verosimilitud de la codificación del grupo de asientos y el cumplimiento de las especificaciones.

La proyección de un día de semana normal, sábado o domingo y la asignación a la red se llevarán a cabo teniendo en cuenta diversos factores, tales como:

- Balance de los datos de la encuesta con los datos del contador.
- Proyección de piezas de encuestados del tipo de vehículo, teniendo en cuenta los cambios combinación.
- Proyección de la muestra al azar.
- Ponderación de los días de encuesta varias para rutas individuales.
- Ponderación de los valores medios anuales (relacionados al ticket).
- Ponderación de los diferentes días, por ejemplo, Sábado (relacionados al ticket).
- Ponderación de los puntos de origen y de destino y el origen y las rutas de destino.
- Ponderación de los flujos individuales.
- Factores específicos de viaje
- Encuestas parciales también se pueden simular en esta asignación de tráfico.

- Evaluación de programas.
- Las personas transportadas / km por persona.
- Relaciones de origen / destino.
- Uso de rutas.
- Uso de paradas.
- Usuarios del sistema / rendimiento del sistema.
- Tiempos de viaje / distancias.
- Cálculo de ingresos en rutas específicas.
- Gráficos interactivos.

Varias evaluaciones se pueden realizar sobre la base de los resultados del tráfico de asignación.

En todos los programas de evaluación, las relaciones que se están examinando pueden limitarse a flexibilidad definibles en partes físicas de la red (zonas efecto). Esta definición puede llevar a cabo los enlaces y los viajes si es necesario.

Los tickets, motivo del viaje o cualquier conjunto de ellos, las zonas de origen y destino, tiempos de viaje y las agrupaciones de flujo se pueden utilizar como filtros para todos los programas de evaluación.

Las etapas de evaluación están disponibles en Windows NT en un programa integrado. Las evaluaciones diferentes orientadas a gráficas y a la lista pueden ser llamadas y se imprimen para la selección de datos elaborado por el simple cambio del contexto de presentación.

Las personas transportadas / km por persona
El número de personas transportadas, kilómetros por persona y la distancia promedio en las rutas individuales, las sucursales de la empresa, los operadores y los totales se calculan y se muestran en la forma de una lista. La evaluación se lleva a cabo para tickets específicos con una facilidad para la creación de los totales a través de las estructuras tarifarias (o grupo de tickets) o según el motivo del viaje, y una facilidad para la creación de los totales de acuerdo con los propósitos de los grupos de viaje. Las clases se pueden mostrar por separado si es necesario. Se muestran los viajes según el área seleccionada de la actividad. Se realizan cálculos anuales, mensuales y valores diarios.

Relaciones origen / destino
Se pueden evaluar las relaciones entre las paradas individuales, grupos de paradas definidos flexiblemente o grupos de origen/destino definidos flexiblemente. La carga de origen / destino, la frecuencia de la transferencia y el número de kilómetros por persona se calcula en ambos sentidos. Los totales se calculan para cada origen y destino. Se calculan los viajes en el área

seleccionada de la actividad y mismo grupo origen/destino.

Uso de ruta
La carga (número de personas transportadas) se calcula para cada ruta. Las proporciones de carga individuales se diferencian según el número de personas que entran y salen ya en las paradas individuales y la carga del enlace entre las paradas. El resultado (opcional) puede ser una lista que contiene la orden de parada o una pantalla gráfica (en formato PostScript).

Una opción de ruta especificada puede ser utilizada como un ejemplo para resumir las rutas paralelas en la misma ruta, la misma rama de operación o todas las rutas.

Uso de paradas
La carga causada por el número de personas que entran, salen y la transferencia se pueden enumerar en detalle o como un resumen de cada parada. Todas las relaciones de transferencia en la parada se pueden enumerar con amplios detalles, incluida la información sobre las transferencias de los peatones.

Los usuarios del sistema/rendimiento del sistema
La carga y el rendimiento de los sistemas de tráfico (rutas individuales, las ramas de explotación o agrupación de ellas, u operadores definibles se pueden utilizar en este caso) y las relaciones entre estos sistemas, debido a las transferencias se calculan y se muestran en formato de lista.

Distancias de viaje / tiempos de viaje
La distribución de las distancias de los viajes puede ser calculada de acuerdo a clases definibles distancia. Los valores individuales y acumulados se enumeran (absoluto y relativo).

Tiempos de recorrido
La distribución del tiempo de viaje se puede enumerar.

Cálculo de ingresos en rutas específicas
Un programa para el cálculo de los ingresos de ruta específica se ha escrito para la VVS y la SBB. Los ingresos de las rutas individuales de un operador se pueden calcular de forma individual o con las relaciones de transferencia a otras rutas que pertenecen al mismo operador o de un operador diferente sobre la base del modelo tarifario, según los tickets que se utilizan, el número calculado de kilómetros por persona y la distribución conocida del ingreso.

Los gráficos interactivos
Los resultados de una asignación de tráfico puede ser desplegada gráficamente como una red o el plan de uso de paradas y se imprime en formato PostScript. Varios parámetros están disponibles para el control de la pantalla, tales como:

- Red de secciones, las escalas de producción.
- Los formatos de salida.
- Escalas de carga.
- Fuentes.
- Los medios de transporte, las superposiciones, los cálculos totales.
- Todos los filtros anteriores.
- Varios niveles para tomar en consideración las zonas efecto.
- Flujo de clúster con opciones de definición de niveles múltiples para la selección de enlace.

Rucus (Run Cutting and Scheduling System)
Características
Link: http://onlinepubs.trb.org/onlinepubs/trnews/rpo/rpo.trn118.pdf (último acceso: 22/03/2010) *Generalidades*:

La agencia de investigación comenzó mediante la preparación de un amplio análisis de todos los pasos a seguir en la planificación y ejecución del proceso. Esto fue seguido por el desarrollo de un sistema modular que cubría todas las fases de la planificación de actividades del departamento, sin embargo, fue propicio para la aplicación por etapas.
El software fue probado RUCUS contra de las soluciones de programación manual en Akron, Baltimore y San Diego. La asignación de controlador de viajes generados por el sistema automatizado Se encontró que ahorrar cerca de 2 por ciento en controlador de pagar horas sobre el manual producido horarios. Después de extensas las pruebas, el software fue desarrollado RUCUS hasta el punto en el que se puesto en uso operativo completo en mediados de 1975 en el centro de Nueva York Autoridad Regional de Transporte en la selección de enlaces Syracuse.for Aplicación y Beneficios 1979, 44 agencias de transporte en América del Norte se analiza, prueba o investigando activamente RUCUS software o derivados de la misma. Las 25 agencias de transporte utilizando RUCUS o el software-como RUCUS sistemas de ahorro anual estimado de alrededor de $ 3,3 millones, con un ahorro en la operación los costos que varían de 1 a 3 por ciento.

Un estudio realizado por el Tránsito Metropolitano Comisión (MTC), St. Paul, Minnesota, documentado ahorros específicos en uno de los casos. El MTC análisis costo-beneficio demostrado las siguientes ventajas de RUCUS:

• Aliviar el fabricante del horario de repetición tareas y propenso a errores;
• Producción eficiente y rentable se ejecuta, y
• Hacer una contribución importante a la gestión de datos del sistema de información base.

La programación de los vehículos de transporte, la elaboración de horarios, y la asignación de pistas a los conductores es muy compleja y requiere mucho tiempo. Son tareas que implican el uso de una gran cantidad de datos. Después que RUCUS está instalado y la base de datos esté establecida, se puede realizar: (a) tratamiento automatizado de carga de pasajeros datos de conteo, (b) la interfaz automatizada phototypesetter con los horarios públicos; (c) de interfaz a un sistema de nómina; y (d) kilometraje programado por vehículo para el control de costes.

Los recientes acontecimientos RUCUS -como de la comunidad de consultoría- han producido ayudas a la programación automática que operan en mini y microcomputadoras. El original desarrollo de RUCUS, patrocinado por la Administración de Transporte Urbano, ha llevado a una nueva pequeña industria en la comunidad de la consultoría con varias empresas ofreciendo RUCUS como sistema para la industria del tránsito.

NOVA

Características

Link: http://novascheduling.com/products.html (último acceso: 22/03/2010)
Generalidades:
NOVA es un completo sistema para la planificación de una tripulación. Permite a los planificadores mantener horarios de vuelo, hacer emparejamientos eficientes de construcción, dar cabida a las solicitudes de oferta y automatizar la lista de generación.

La ventaja clave de NOVA es la capacidad de automatizar la generación de lista mediante licitación preferencial. Esto permite a las líneas aéreas ahorrar tiempo mediante la generación de listas mucho más rápidamente y a los miembros de la tripulación obtener una mayor satisfacción por poder especificar sus preferencias.

Acceso multi-usuario *Se adapta al crecimiento, NOVA maneja multitarea y edita con facilidad.*
• Un equipo de planificadores pueden cada uno trabajar en diferentes tareas al mismo tiempo.
• Todos los datos de vuelo, la pareja y la lista común es compartida.
La información se mantiene en una base de datos central y todos los planificadores tienen acceso a hacer cambios.

Programación de vuelo *Fácil de mantener si se cambia el horario, NOVA indica que emparejamientos y listas requieren modificaciones.*
• Agregar y modificar la línea principal, charter, ad hoc, y posicionamiento de información de vuelo, en cualquier momento.
• Complemento de tripulación de Alter y otros especiales requisitos en los días seleccionados y vuelos.
• Pista de vuelos y emparejamientos a través de períodos sin tener que volver a introducir datos.

Programa *Maximizar la eficiencia, NOVA le permite construir múltiples escenarios, predecir y comparar los costos y seleccionar la combinación más eficiente.*
• Acomodar vuelos y aterrizajes, capacitación, vacaciones, y otras variables en la construcción del emparejamiento.
• Manejar sin problemas los cambios de zona horaria y tiempo de ahorro de luz del día.
• Validar el plan de vuelo con emparejamientos para garantizar cobertura.

Opciones de licitación *Aumentar la satisfacción, NOVA admite solicitudes de oferta, una solución dinámica y capacidad de respuesta que beneficia tanto a la tripulación como a la aerolínea.*
• Satisfacer las peticiones de la tripulación para los destinos preferidos, dentro de miembros de la tripulación fuera de tiempo y sus compañeros, límites operacionales y legales.
• Las aerolíneas se benefician de la mayor satisfacción de la tripulación, en última instancia, que afecta a sus clientes.
• Captura ofertas de tripulación desde ubicaciones remotas.

NOVA OnLine es un sistema de comunicación de Internet que permite la entrada de la oferta y la tripulación, utilizado para apoyar la fase referencial del sistema de planificación de NOVA Crew. Permite que los planificadores distribuyan información de paquetes de oferta y publicar listas en Internet rápida y fácilmente. Con NOVA OnLine, las tripulaciones pueden presentar sus ofertas preferenciales en línea por Internet, desde cualquier ubicación remota en el mundo. Obtienen acceso instantáneo a la oferta de información del paquete tan pronto como esté disponible y puede presentar sus ofertas desde una ubicación más conveniente, como desde sus hogares.

Listas automáticas generadas por el sistema
NOVA ofrece la generación de listas automatizadas basadas en la oferta y asigna premios por antigüedad, la prioridad o la rotación.
• Previsión de los requisitos de tripulación más rápido y con mayor rapidez, reduciendo la reserva de número de tripulantes necesaria.
• Establecer su propia construcción y los parámetros de destino.
• Especificar las directrices de la lista para cada base de tripulación.
• Integrar varias asignaciones de lenguaje, y mezcla las calificaciones automáticamente. La comprobación de reglas asegura que sus operaciones cumplen con todas las reglas del Gobierno y la industria.
• Se comprueban automáticamente todas las asignaciones de emparejamientos y lista para el cumplimiento legal. Se muestra un mensaje de advertencia si se producen violaciones.
• Comprueba conjuntos de reglas que se aplican a períodos específicos, lo que facilita la integración de nuevos contratos o cambios reglamentarios.

Informes *Según demanda, NOVA ofrece más de 20 informes actualizados, incluyendo:*
• Individual lista de premios.
• Hotel, transporte terrestre y requisitos de posicionamiento.
• Asignación de costos resumen.

Registros *Configurados a sus necesidades, los registros de planificación de NOVA son fáciles de acceder y administrar en una base de datos central.*
• Los registros de tripulación registros contienen información sobre las calificaciones y bases,

requerimientos de capacitación y las estadísticas.
• Emparejamientos y la lista de parámetros jurídicos ofrecen una amplia gama de reglas que son fáciles de modificar y mantener.

Exportación/importación de datos *Compartir información, NOVA puede intercambiar datos con otros sistemas, asegurando velocidad y precisión en la transferencia.*
• Plan de vuelo de importación en formato de SIMM de IATA.
• Importación de información de tripulación, incluidos los datos de la tripulación y de la historia, para un día, desde sistema de operaciones tales como Magallanes.
• Exportar e importar emparejamientos desde un sistema de optimización de emparejamiento como Cygnus.
• Exportación de información de tripulación, incluidos los datos de tripulación y listas, para un día, de sistemas de operaciones tales como Crewtrac y Magallanes.

Atención al cliente NOVA proporciona asistencia sensible y fácil de usar.
• Manuales de formación y ayuda en línea.
• 24 Horas asistencia telefónica al cliente.

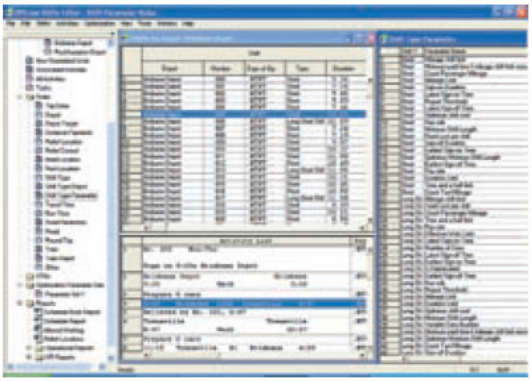
Complementos opcionales *Mejorar el rendimiento, mejorar su proceso de planificación mediante la adición de:*
• Sistema de optimización de Cygnus pareado – Ahorre tiempo sustancial y dinero al automatizar y optimizar la generación de emparejamiento.
• Sistema de entrada de pujas de Internet - permite la tripulación enviar ofertas a través de la Internet desde cualquier ubicación remota.

OpCrew

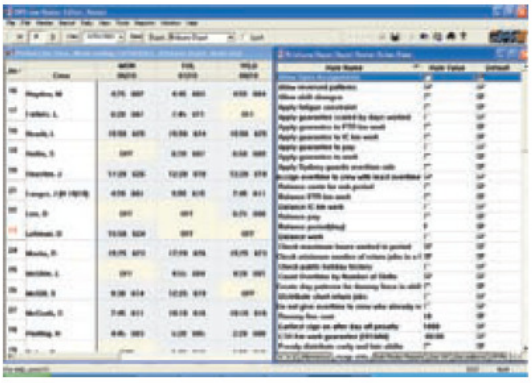
Características

Link: (último acceso: 22/03/2010)
Generalidades:

OPCrew Shifts



OPCrew Rosters



Gestión de la tripulación ferroviaria eficaz - Un requisito fundamental
La gestión eficaz de la tripulación de ferrocarril es una capacidad básica que todos los ferrocarriles requieren. De hecho, la tendencia de desregulación de la industria y la reestructuración es el aumento de la exigencia de todos los aspectos del funcionamiento de un ferrocarril para ser competitivo y eficiente. Un ferrocarril deben tener los sistemas de gestión de la tripulación y procesos que le permiten:

• Uso eficiente de la tripulación de los trenes de personal.
• Plan de Operaciones de la tripulación de forma rápida y eficaz.
• Minimizar las interrupciones de la tripulación del tren relacionados.
• Proporcionar un entorno de trabajo justo y equitativo para la tripulación.
• Controlar la fatiga de la tripulación y la seguridad.
• Responder rápidamente a los cambios de tren y de la tripulación.
• Eficiente administración de la tripulación de ferrocarril.
• Evaluar las opciones estratégicas para la tripulación (por ejemplo, lugares de base de la tripulación).

Un equipo completo de gestión del sistema ferroviario.
Para proporcionar los ferrocarriles con la capacidad de gestión de la tripulación que requieren, OPCOM desarrollado OPCrew. OPCrew es un sistema modular que permite a los ferrocarriles gestionar todos los aspectos de las operaciones de la planificación de la tripulación a largo plazo.

En el sistema se destacan:
• Central de gestión de datos de la tripulación.
• La generación automática de los turnos de la tripulación optimizado y listas.
• La gestión de los cambios período a los cambios y las listas.
• El día de la operación de cambio y reparación de lista.
• El tiempo y la administración de la asistencia de la tripulación.

OPCrew ofrece:
• Aumento de la eficiencia de la tripulación.
• Reducción del esfuerzo de la tripulación de planificación.
• La tripulación equitativa planes.
• Mejora de la gestión de los cambios periodo minimiza el uso de horas extras y garantizando la cobertura de tren.
• Mejora de las decisiones operativas relacionadas con la reducción de las interrupciones de la tripulación y de reserva y requisitos del equipo de socorro.
• Generación automática y controlada de los requisitos de pago de la tripulación de la reducción de errores, disputas y los costes administrativos.
• Pulso cardiaco rápido, un análisis preciso y robusto de las opciones de la tripulación de facilitar una mejor toma de decisiones.

OPCrew dispone de lo siguiente:
• Una completa solución de gestión de tripulación ferrocarril.
• Construcción automática de los cambios de la tripulación y las listas.
• La gestión de las operaciones diarias de la tripulación.
• Cálculo de las necesidades de la nómina de la tripulación.
• Una estructura de datos flexible que puede integrarse con la planificación actual del tren y los sistemas administrativos de la tripulación.
• Una arquitectura modular que permite la implementación selectiva y progresiva.
• Versiones para el ferrocarril urbano y de larga distancia.
• Una interfaz fácil de usar.
• Un enfoque ferroviario específico para modelar las reglas de trabajo de la tripulación y las prácticas operativas.
• Preferencia de la tripulación de captura.

B.2.1 Observaciones:

Del relevo de sistemas de planificación de tripulación y vehículos se desprenden las siguientes conclusiones:

- Consideran diferentes tipos de usuario.
- Ofrecen diferentes vistas de los datos y expedir reportes de formato personalizable.
- Permiten acceso a los datos por diferentes medios.
- Permiten la retroalimentación constante a los datos.
- Consideran leyes laborales y permiten personalizar requerimientos específicos.
- Permiten personalización de los resultados y realizar cambios manuales.
- Ofrecen diferentes formas de interacción con los gráficos y de edición.
- Permiten configuración de los datos de entrada para adaptarse a cada situación.
- Permiten ejecutar algoritmos de forma local o remota.
- Permiten ejecutar validaciones sobre los datos de entrada y los resultados.
- Permiten incorporar experiencias y resultados previos para mejorar los resultados actuales.
- Brindan soporte a las decisiones y estrategias alternativas.
- Reducen costos y aumentan la eficiencia de los recursos.
- Son multipropósito: realizan planificación de tripulación, planificación de vehículos, ruteo, etc.

B.3 Especificación de E-S de Algoritmos

El Departamento de Investigación Operativa del Instituto de Computación (InCo) de la Facultad de Ingeniería desarrolla algoritmos para resolver el problema de **planificación y ordenamiento de vehículos y tripulación** para una compañía de transporte colectivo urbano y suburbano de pasajeros. En esta sección se especifica el formato de las entradas y salidas de los mismos.

B.3.1 Entradas

Estos datos se extraen del sistema de la empresa de transporte.

El algoritmo de optimización tiene como entrada un archivo de texto con varias tablas de datos. Dichas tablas tienen un orden específico que se muestra a continuación:

```
#Sectores
CodSector   CodPlace    CodCalidad  Nombre

#Calidad
CodCalidad  Nombre

#Servicios
CodBus      Servicio    CodSector

#Lineas
CodLinea    CodCalidad  Corredor    Nombre

#Lugares
CodLugar    Nombre

#Expresos
CodOrigen   CodDestino  Distancia   Tiempo

#Viajes
Nro.Viaje   Linea CodOrigen   CodDestino  Atraque     Salida      Llegada
```

```
#TiemposPideOrden
CodLug      TiempoPOMIN

#KmsLineas
CodLinea    CodOrigen   Kms
```

Los datos de cada tabla son los siguientes:

B.3.1.1 Sectores:

En esta tabla se tienen cuales son los depósitos de los ómnibus. Se tienen los siguientes campos:

- *CodSector*: es el código del sector.
- *CodPlace*: es el código del lugar en el que esta el deposito.
- *CodCalidad*: es el código de calidad de los ómnibus del sector.
- *Nombre*: es el nombre que tiene el sector.

B.3.1.2 Calidad

En esta tabla se definen los códigos de calidad de los ómnibus. Contiene los siguientes campos:

- *CodCalidad*: es el código que identifica la calidad de un coche.
- *Nombre*: nombre del tipo de coche.

B.3.1.3 Servicios

Contiene para cada coche de la compañía, el número del servicio que se le asignará en la primer rotación del sector:

- *CodBus: código del coche.*
- *Servicio: servicio que cubre en el primer día de rotación.*
- *CodSector: sector al que pertenece el coche.*

B.3.1.4 Lineas

En esta tabla se definen las líneas. Los campos de esta tabla son los siguientes:

- *CodLinea*: el código de la línea.
- *CodCalidad*: la calidad de los viajes de la línea.
- *Corredor*: el corredor al que pertenece la línea.
- *Nombre*: el nombre de la línea.

B.3.1.5 Lugares

Aquí se definen todos los lugares que son visitados por la empresa de transporte (seria deseable excluir los carteles). Los campos utilizados son los siguientes:

- *CodLugar*: el código del lugar.
- *Nombre*: el nombre del lugar.

B.3.1.6 Expresos

En esta tabla se tienen los expresos que se utilizaran para crear el libro. Típicamente todos los que se tengan disponibles. Se tienen los siguientes campos:

- *CodOrigen*: es el código del lugar de origen.
- *CodDestino*: es el código del lugar de destino.
- *Distancia*: la distancia en Kms -multiplicados por 0.8-. (en números enteros).
- *Tiempo*: es el tiempo en minutos (en números enteros).

B.3.1.7 Viajes

En esta tabla se tienen los viajes con los que se conformará el libro de servicios. Los campos de esta tabla son:

- *Nro.Viaje*: el numero del viaje.
- *Linea*: el código de la línea al que pertenece el viaje.
- *CodOrigen*: el código del lugar de donde sale el viaje.
- *CodDestino*: el código del lugar a donde llega el viaje.
- *Atraque*: la hora a la cual el viaje debe atracar antes de salir.
- *Salida*: la hora a la cual el viaje debe salir.

- *Llegada*: la hora a la cual el viaje debe llegar a su destino.

B.3.1.8 TiemposPideOrden

Esta tabla contiene los tiempos mínimos que pueden asignarse a un 'Pide Orden' según el lugar en el que se realice:

- *CodLug*: Código del lugar en el que se realiza el Pide Orden.
- *TiempoPOMIN*: Tiempo mínimo para un Pide Orden en dicho lugar.

B.3.1.9 KmsLineas

En esta tabla se tienen los kilómetros en que recorre cada línea. Los campos que se tienen son:

- *CodLinea*: el código de la línea.
- *CodOrigen*: el código del lugar del que sale la línea.
- *Kms*: la cantidad de kilómetros que recorre un viaje de la línea (numero entero).

B.3.2 Salidas

Vista general de la gramática del archivo de salida de los algoritmos (tanto el de soluciones iniciales como el integrado):

```
Datos de <path>
Servicios encontrados con horizonte <hora> minutos
Costo = <entero>
Costo EMPRESA = <entero>
#Viajes cubiertos = <int>
#Viajes NO cubiertos = <int>
Cant. servicios = <int>

Servicio <acc(1)> : DATOS DEL SERVICIO
Hora comienzo = <hora>
Hora fin =  <hora>
Lugar del servicio = <int>
kms viaje = <int>
kms expresos = <int>
Cantidad de turnos = <int>
Costo(por ahora incluye kms expresos, viático y pideOrden) = <int>
Capacidad restante = <hora>

Lista de turnos:
-<acc(2)>--    Datos generales del turno:
Cantidad de Kms = <int>
Evento<\t>Duracion<\t>Origen<\t>Destino<\t>Atraque<\t>Sale<\t>Llega<\t>Descripcio
n Linea<\t>Numero Viaje
Viaje<\t><int><\t><lug><\t><lug><\t><hora><\t><hora><\t><hora><\t><desc_ln><\t><i
nt>
Expreso<\t><int><\t><lug><\t><lug>
Pide_Orden<sp(5)><int>
Descanso<sp(11)>DESCANSO INTERMEDIO <hora> a <hora>
Person<sp(11)>Personal                de                PASAJERO                a
<lug><sp(15)><hora><sp(10)><hora><sp(10)><hora><sp(10)><desc_ln>

Gramatica

salida = inicio lista_serv
lista_serv = serv lista_serv | vacio
serv = ini_serv lista_turnos
lista_turno = turno lista_turno | vacio
turno = ini_turno lista_evento
```

```
lista_evento = evento lista_evento | vacio
evento = viaje | expreso | pide_orden | descanso | person


Tipos de datos:

indefinido = #
digito = dig
entero = int | #
hora = dig dig ':' dig dig | #
tabulacion = \t
retorno de linea = \n
espacios = sp(int)
path = ruta de archivos, directorios con /
descripcion_linea = desc_ln
lugar = lug
acumulado "es un tipo int que cumulativo y se resetea por niveles superior:
cuando aparece un nuevo acumulado del nivel n, se resetean los acumulados de
nivel n+1" = acc(int)
inicio =
Datos de <path><\n>
Servicios encontrados con horizonte <hora> minutos<\n>
Costo = <entero><\n>
Costo EMPRESA = <entero><\n>
#Viajes cubiertos = <int><\n>
#Viajes NO cubiertos = <int><\n>
Cant. servicios = <int><\n>
<\n>
ini_serv =
Servicio <acc(1)> : DATOS DEL SERVICIO<\n>
Hora comienzo = <hora><\n>
Hora fin =  <hora><\n>
Lugar del servicio = <int><\n>
kms viaje = <int><\n>
kms expresos = <int><\n>
Cantidad de turnos = <int><\n>
Costo(por ahora incluye kms expresos, viático y pideOrden) = <int><\n>
Capacidad restante = <hora><\n>
<\n>
Lista de turnos:<\n>
ini_turno =
-<acc(2)>--<sp(4)>Datos generales del turno:<\n>
Cantidad de Kms = <int><\n>
Evento<\t>Duracion<\t>Origen<\t>Destino<\t>Atraque<\t>Sale<\t>Llega<\t>Descripcio
n Linea<\t>Numero Viaje<\n>
viaje                                                                    =
Viaje<\t><int><\t><lug><\t><lug><\t><hora><\t><hora><\t><hora><\t><desc_ln><\t><i
nt><\n>
expreso = Expreso<\t><int><\t><lug><\t><lug><\n>
pide_orden = Pide_Orden<sp(5)><int><\n>
descanso = Descanso<sp(11)>DESCANSO INTERMEDIO <hora> a <hora><\n>
person        =        Person<sp(11)>Personal        de        PASAJERO        a
<lug><sp(15)><hora><sp(10)><hora><sp(10)><hora><sp(10)><desc_ln><\n>

Ejemplo

Datos de instances/data_file_example_2007_12_24.txt
Servicios encontrados con horizonte 24:00 minutos
Costo = #
Costo EMPRESA = #
#Viajes cubiertos = 25
```

```
#Viajes NO cubiertos = 5
Cant. servicios = 2

Servicio 0 : DATOS DEL SERVICIO
Hora comienzo = 7:00
Hora fin = 16:00
Lugar del servicio = 2
kms viaje = 517
kms expresos = 10
Cantidad de turnos = 2
Costo(por ahora incluye kms expresos, viático y pideOrden) = 2501
Capacidad restante = 4:45

Lista de turnos:
-0--    Datos generales del turno:
Cantidad de Kms = 154
Evento      Duracion    Origen      Destino     Atraque     Sale  Llega
      Descripcion Linea Numero Viaje
Viaje 120   PANDO PANDO 7:00  7:05  9:05  7A-Pando-Pando(C.MAL)   129
Expreso     180   PANDO MONTEVIDEO

-0--    Datos generales del turno:
Cantidad de Kms = 0
Evento      Duracion    Origen      Destino     Atraque     Sale  Llega
      Descripcion Linea
Pide_Orden     240
Descanso           DESCANSO INTERMEDIO 14:05 a 15:05
Person           Personal de PASAJERO a PANDO               15:05          15:10
16:00          7A-Mdeo-Pando(C.MAL)

Servicio 1 : DATOS DEL SERVICIO
Hora comienzo = 15:05
Hora fin = 17:00
Lugar del servicio = 1
kms viaje = 517
kms expresos = 0
Cantidad de turnos = 1
Costo(por ahora incluye kms expresos, viático y pideOrden) = 2501
Capacidad restante = #

Lista de turnos:
-0--    Datos generales del turno:
Cantidad de Kms = 154
Evento      Duracion    Origen      Destino     Atraque     Sale  Llega
      Descripcion Linea
Viaje 120   PANDO MONTEVIDEO  15:05 15:10 16:00 7A-Pando-Mdeo(CIUD.)
Expreso     60    MONTEVIDEO PANDO
```

C Anexo: Documentación de proyecto

C.1 Introducción

En este anexo se presenta la documentación general del proyecto. La documentación técnica y de usuario de FingLab se entrega en un documento aparte con el CD que contiene la implementación.

C.2 Plan de Tesis

Se trabajará con referencias dadas por los tutores, así como también el grupo deberá realizar búsquedas por sus propios medios. Cualquier consulta sobre algún material, se puede enviar la información a los tutores para pedir su opinión al respecto.

En caso de que un documento no esté disponible para bajar de Internet se debe enviar la información a alguno de los tutores para que éste lo pida a biblioteca. El grupo también podrá consultar el catálogo colectivo de biblioteca disponible en Internet (http://www.fing.edu.uy/inco/pedeciba/bibpm/field.php/Main/Cat%e1logos) que incluye tesis de grado ya entregadas. También se consultarán tesis de otras universidades (http://www.fi.uba.ar/materias/7500/).

Se propone citar todas las **fuentes de búsqueda** utilizadas (Google, Citeseer, etc.).

La investigación se realizará según la siguiente metodología:

- Búsqueda
  - Buscar por distintas fuentes: primero las que son gratuitas, luego para las cuales es necesario algún registro
    - Google
    - Citeseer
    - Biblioteca facultad
    - Referencias de otros libros y papers
    - Por los mismos autores de los que ya se encontró material
  - Registro de palabras clave de búsqueda (ver Palabras claves)
  - Registro de los autores más importante o comunes al tema
  - Recolección de material
- Selección de material
  - Leer por encima para tener una idea de cuánto aporta según:
    - Índice
    - introducción
    - Abstract
    - resumen si lo tiene
    - resultados
    - conclusiones
  - Clasificar según:
    - Utilidad: cuáles tienen algo que aportar al tema y cuáles no
    - Novedad: si es algo reciente, alguna innovación respecto al tema
    - De qué fuente viene
    - Longitud: si son demasiados cortos (y no aportan nada) o demasiado extensos (y solo quedan para profundizar en el tema)
    - Profundidad
    - Material gráfico (diagramas, esquemas)
  - Poner comentarios en el título del archivo para recordar
  - Almacenarlo en diferentes carpetas por tema y según si fue útil o no
  - Chequeo de autores (ver Autores)

    - Agregar a referencias
        - Agregar qué referencias fueron útiles y cuáles se descartaron y por qué (ver Registro de Referencias)
- Resumir o extraer lo más importante de cada referencia útil
    - Hacerlo en el informe directamente o
    - tomar notas o
    - subrayar o
    - anotar página en que se encuentra la información útil o
    - editarlo si el documento lo permite
- Agregar al glosario (ver Glosario):
    - Términos nuevos
    - definiciones nuevas
    - acepciones nuevas
    - especificaciones de un concepto
    - traducciones nuevas de un término
- Agregar la información obtenida al informe [1]
    - Guardar versiones del informe
- Citar en el informe (ver Registro de Referencias)
- Archivar el material
- Respaldar el material en el repositorio

C.2.1 Documento de Tesis

El documento final de la tesis consistirá en un **documento maestro** (ejecutivo) que referirá a otros **apéndices e informes** donde se desarrollan los temas. Cada etapa del proyecto tendrá como resultado un informe al cual el documento maestro hará referencia y se irán haciendo a lo largo de la duración del proyecto [2, 3].

No se requerirá documentación referente a la administración interna del grupo, pero se pedirá que se tenga un cronograma de actividades específico.

Para tener como ejemplo en el catálogo de la biblioteca del Inco [4] hay tesis de proyectos de grado que se pueden bajar para verlas y utilizarlas como referencia [5].

C.2.2 Estructura

La estructura del informe final es la que se recomienda en casi toda la bibliografía [2]. El Informe deberá contener:

- **Carátula** indicando título del proyecto, nombres de los estudiantes, nombres de los tutores, cometido, lugar y año de su publicación.

- **Resumen del trabajo** (*200-500 palabras*). Seguido de palabras claves o temáticas relevantes (**glosario**).
    - El resumen debe dar una idea completa de todo el proyecto, se supone que un lector comienza por leer el resumen y puede no leer nada más. Por lo tanto el resumen debe ser una buena enumeración de todas las etapas del proyecto.
    - Es importante que el resumen mencione claramente los formalismos, técnicas, herramientas, lenguajes utilizados.
    - El resumen no debe limitarse a describir el problema abordado, sino que debe describir la solución del problema, con una evaluación de la misma. El resumen no es como "el pie publicitario de una novela" que deja en suspenso al lector para que lea el libro.

- No es necesario incluir en el resumen una descripción contextual al estilo de "... en los últimos años el campo de xxx ha resultado importante debido a .....".
- No incluya referencias bibliográficas ni referencias a otras partes del informe.
- No utilice acrónimos sin explicar su significado, salvo que sean muy conocidos.

- **Tabla de contenido**.

- El **Informe (documento maestro)** estará organizado en capítulos y dentro de éstos, en secciones y sub-secciones.

  - Capítulo 1: Consiste básicamente en la **Introducción**:
    - se plantea y define el problema
      - Problemática del transporte
      - Problema VSP y CSP
        - Solución por medio de heurísticas, algoritmos genéticos, etc.
    - se motiva el trabajo
      - Visualización de datos
    - se deja claro cuales son los objetivos planteados (general del proyecto, si correspondiese o está inmerso en un proyecto más ambicioso, etc. y los específicos)
      - proveer las interfaces para los algoritmos
    - se plantean los resultados esperados
    - se establecen resumidamente las conclusiones
    - se describe la organización general del documento, destacándose secciones que se deben leer para entender el trabajo y cuales no (si las hay)

  - Capítulo 2: revisión del **estado del arte**.
    - Investigación del **estado del arte de interfaces**
    - breve introducción a los conceptos necesarios para entender el trabajo (**glosario**)
    - **Herramientas similares de scheduling** (incluye cuadro de comparación)

  - Capítulo 3: **Parte central del trabajo**. Se refiere a lo que es **producción** propia o aporte del proyecto de grado. Si es muy extenso o abarca distintas temáticas puede dividirse en más capítulos. Se incluyen las decisiones tomadas.
    - **Requerimientos**
    - alcance
    - **Herramientas de visualización** (estado del arte)

  - Capítulo 4: **documentación técnica**
    - **diseño**.
    - **Implementación**: describirla en términos de decisiones tomadas en ese sentido (pueden repetirse si algunas se expresaron en el capítulo anterior), tipos de datos utilizados, restricciones establecidas, funcionalidades, etc. Los detalles de programación se dejan para los anexos.
    - **prototipos**
    - **Plan preliminar de pruebas**. Resultados esperados.

  - Capítulo 5: Puede ser un capítulo aparte del 4. Incluye:
    - las **pruebas realizadas** (casos de prueba)
    - los **resultados obtenidos**
    - **comparaciones** (pueden incluirse gráficas, tablas)

  - Capítulo 6: **Conclusiones y Trabajo Futuro**.
    - aquí se evalúan los **resultados alcanzados**, dificultades encontradas,
    - **lo que se planteó hacer y lo que se hizo realmente**, aportes
    - se muestran posibles **extensiones al trabajo**
    - se realiza una **autocrítica** de lo que se hizo y lo que faltó (por problemas de tiempo, recursos, como se puede continuar, que cosas hacer, prioridades, etc.)

- Anexo de **referencias**: contiene todas las referencias de todo el informe. Puede incluir los **chequeos de los autores** más importantes.

- Anexo de **glosario**: contiene todas las definiciones de los términos utilizados en todo el informe. Puede incluir las **palabras clave** más importantes utilizadas en las búsquedas.

- Anexo de **manual de usuario**: es el manual de usuario de la aplicación construida.

- Anexo **Administrativo**: contiene todo lo referente a la administración del proyecto:
  - **Plan de proyecto**
    - **Cronograma**
  - Cómo se desarrolló la **tesis**
  - **Registro de actividades**
  - **Actas de reunión**

C.2.3 Metodología

La metodología se establecerá en la sección "Estandar de Implementación y Documentación técnica y de usuario". [1, 5-21]

C.2.4 Formato

La documentación será generada en su mayoría en Microsoft Word o Powerpoint para las presentaciones pero se podrá generar copias en formato pdf o algún otro que los tutores especifiquen.

Las hojas serán en papel tamaño A4 y márgenes de al menos 1,5 cm con letra tipo Tahoma de tamaño 10 pts. El espaciado será simple. Cada sección tendrá título numerado [13].

Cada documento tendrá carátula con el título, autor, fecha y nombre del proyecto e índice. Cada página tendrá un encabezado con el título del documento y pie de página con el número de página.

C.2.5 Estilo de redacción

La documentación debe ser clara para permitir un buen seguimiento tanto del grupo como de los tutores de lo ya realizado. Se debe poder seguir las etapas del pensamiento y desarrollar la investigación manteniendo el objetivo de la misma. El lenguaje debe ser comprensible para aquellos involucrados, según su dominio del tema y capacitación [1].

Las imágenes pueden o no estar numeradas pero en la redacción debe quedar claro a qué imagen se está haciendo referencia.

C.2.6 Respaldos

Se realizarán respaldos en diversos medios:

- Luego de cada sesión de trabajo (o luego de cada modificación importante) con un documento o código se realizará un respaldo (identificado con la fecha) que se guardará en:
  - Dos directorios distintos del computador del grupo dedicado exclusivamente a los respaldos.

    - Una copia de dicho archivo se guardará en un pendrive donde se mantendrán las 10 o 15 versiones más recientes.
- Cada dos o tres semanas se realizará:
    - Copia de los respaldos generados en un pendrive y a un disco externo.
- Cada dos meses se realizará:
    - Copia de los respaldos generados en un CD
- Los respaldos más importantes también se mantendrán en:
    - Una dirección de correo electrónico creada exclusivamente para mantener respaldos.
    - El repositorio de facultad.

Se enviará copias o notificaciones de la ubicación de las copias de los documentos más importantes por mail a los tutores.

C.2.7 Registro de Referencias

Luego de comparar herramientas para registro de referencias [22-28], las referencias se registrarán por medio del programa Endnote [29], ya que este permite mantener las referencias y luego citarlas fácilmente en Microsoft Word.

En principio se establecen como pautas para las referencias:

- Especificación completa de todas las referencias utilizadas. Cada referencia debe contener todos los detalles que se tengan de la misma: título, autores, fecha y lugar de publicación [30-32], si es un libro se debe especificar en la medida de lo posible el ISBN, si es un paper encontrado en Internet se debe especificar el link y la última fecha de acceso al mismo [33-36].
- El formato ideal es el de usar números y listar las referencias al final del documento. Para más información consultar el **formato bibliográfico y de Web que usan en la biblioteca del InCo** [2, 3].
- El orden no importa.

Se propone clasificar las referencias por:

1. Referencias que se conocen pero no pudieron ser ubicadas: por ejemplo libros que aparecen en el catálogo de la biblioteca pero que estaban prestados y no se pudieron ver, papers de los que se pudo conocer el abstract pero no se consiguió el texto completo, etc.
2. Referencias que se conocen y ubicaron pero que no contenían información adecuada para este proyecto y se desestimaron, ya que se tomó el tiempo de evaluarlas aunque no se utilizaron.
3. Referencias que se conocen, ubicaron y utilizaron para este proyecto.
4. Referencias que se conocen, ubicaron, que pueden resultar útiles, pero que no se utilizaron porque van más allá del alcance del proyecto, eran demasiado profundas o quedan para trabajo futuro.

C.2.8 Autores

En la medida de lo posible se realizará el chequeo de los autores de libros y papers utilizados para este proyecto. Se tratará de por lo menos hacer una lista de los mismos y proveer de un link a sus páginas personales.

C.2.9 Glosario

Se mantendrá un glosario con las definiciones de toda la terminología utilizada durante el proyecto. Dicho glosario contiene los términos que se traen del glosario del proyecto de la compañía de transporte urbano.

C.2.10 Palabras claves

Se harán listas de palabras claves utilizadas para la búsqueda de información sobre cada tema. Cada lista estará adjunta al correspondiente informe. Las palabras clave se asociarán en la medida de lo posible según las fuentes en las que fueron utilizadas y se considerarán en distintos idiomas: español, alemán, inglés, francés, portugués e italiano .

C.2.11 Referencias adicionales para el plan de tesis

Mas referencias de consulta para realizar el plan de tesis fueron:

1. *Biblioteca PECECIBA-InCo - Main-Catálogos* http://www.fing.edu.uy/inco/pedeciba/bibpm/field.php/Main/Cat%e1logos *Último acceso: 20/12/2009*.
2. Caivano, J.L., *GUÍA PARA REALIZAR, ESCRIBIR Y PUBLICAR TRABAJOS DE INVESTIGACIÓN*. 1995.
3. FING, *GUÍA DE INFORME DE PROYECTO DE GRADO*. p. 2.
4. PEDECIBA, F.d. Ingeniería, and U.d.l.R.d. Uruguay, *Guía para la presentación de publicaciones del Instituto de Computación y del Area Informática del PEDECIBA*. p. 10.
5. Ingeniería, F.d. and U.d.B. As., *75.00 TESIS DE GRADO EN INGENIERÍA INFORMATICA* http://materias.fi.uba.ar/7500/ *Último acceso: 20/12/2009*.
6. *Cómo Elaborar una Tesis de Grado - Apuntes de Comunicación y Periodismo* http://www.elprisma.com/apuntes/comunicacion_y_periodismo/comoelaborarunatesisdegrado/ *Último acceso: 20/12/2009*.
7. *Cómo elaborar y asesorar una investigación de tesis - Monografias.com* http://www.monografias.com/trabajos3/comotesis/comotesis.shtml *Último acceso: 20/12/2009*.
8. *Cómo escribir una tésis de grado - Monografias.com* http://www.monografias.com/trabajos/tesisgrado/tesisgrado.shtml *Último acceso: 20/12/2009*.
9. Rothberg, J.M., *Como escribir y publicar un artículo científico.*
10. *Dissertation-Thesis Guide* http://www.learnerassociates.net/dissthes/ *Último acceso: 20/12/2009*.
11. *ESTRUCTURA DEL PROYECTO DE GRADO guia-de-estructura-del-proyecto-de-grado.pdf*. p. 6.
12. Grundy, J.C., *Experiences with Facilitating Student Learning in a Group Information Systems Project Course*. 1996. p. 8.
13. EAFIT, U., *GUÍA PARA LA PRESENTACIÓN DE PROYECTOS Y TESIS DE GRADO*. 2007. p. 25.
14. Melero, R., *How to start to write a scientific paper*.
15. Chandrasekhar, R., *How to Write a Thesis: A Working Guide*. 2000. p. 33.
16. *IUB Writing Tutorial Services Pamphlets* http://www.indiana.edu/~wts/pamphlets.shtml *Último acceso: 20/12/2009*.
17. *LearnerAsociates.net LEARNSITE* http://www.learnerassociates.net/ *Último acceso: 20/12/2009*.
18. *Manual para la elaboración de Tesis Doctorales, Trabajos de Grado y Trabajos Especiales*. 2007. p. 82.
19. *Recomendaciones para la preparación de propuestas de tesis*.
20. Wilson, J.R., *Some guidelines on technical writing*. 1999. p. 8.
21. García-Martínez, D.R., *UNA PROPUESTA PARA LA ESTRUCTURA DE LA TESIS DE GRADO EN INGENIERÍA INFORMÁTICA*. p. 3.
22. Valduriez, P., *Algunas ideas para mejorar la escritura de informes técnicos*. 1994.
23. *Comparison of reference management software - Wikipedia, the free encyclopedia* http://en.wikipedia.org/wiki/Comparison_of_reference_management_software *Ultimo acceso: 20/12/2009*.
24. *Dell'Orso Bibliography formatting software an evaluation template* http://www.burioni.it/forum/ors-bfs4/ors-bfs.htm *Último acceso: 20/12/2009*.
25. *JabRef reference manager* http://jabref.sourceforge.net/ *Último acceso: 20/12/2009*.
26. *Managing references* http://www.library.unisa.edu.au/infoskills/manreferences.asp *Último acceso: 20/12/2009*.
27. *ProCite* http://www.procite.com/ *Último acceso: 20/12/2009*.
28. *Reference Manager* http://www.refman.com/ *Último acceso: 20/12/2009*.

29. *Thomson ResearchSoft - Compare Products* *http://search.thomsonreuters.com/search?q=reference+manager&allAreas=on&site=default_collection&client=default_frontend&proxystylesheet=default_frontend&filter=p&getfields=*&output=xml_no_dtd&num=20* *Último acceso: 20/12/2009*.
30. *EndNote - Bibliographies Made Easy http://www.endnote.com/ Último acceso: 20/12/2009*.
31. *ISO 690 12ª ed. 1987 Documentación - Referencias bibliográficas Contenido, forma y estructura*.
32. Pourailly, M.J.L., *Cómo elaborar referencias bibliográficas.* 2007: p. 10.
33. *Ejemplos de cómo registrar bibliografía: del formato o estilo ISO 690 - 2*.
34. *Norma ISO 690-2 SO/TC 46/SC 9 Referencias a documentos electronicos.htm*.
35. *ISO 690-2 Primera edición 1997-11-15 Norma Internacional ISO 690-2.htm*.
36. Estivill, A. and C. Urbano, *FBD Cómo citar recursos electrónicos http://www.ub.es/biblio/citae-e.htm Último acceso: 20/12/2009*.
37. Aldana, J.S., *LAS REFERENCIAS BIBLIOGRÁFICAS DE RECURSOS ELECTRÓNICOS Y SUS PARTES*. 2000. p. 7.

C.3 Prototipos

Crear prototipos en las fases tempranas del proyecto es de gran importancia, ya que al principio de un proyecto es cuando se toman ciertas decisiones (de diseño, de herramientas a utilizar, etc.) y el prototipado ayuda a examinar si dichas decisiones son viables. Especialmente cuando se trata de crear una interfaz gráfica se debe crear un prototipo que el usuario pueda inmediatamente examinar para indicarle al programador si va en la dirección correcta o si hay algo que definitivamente no representa lo que el cliente desea (*User Interface Prototypes http://www.agilemodeling.com/artifacts/uiPrototype.htm Último acceso: 20/12/2009*.).

Por medio de estos prototipos podemos determinar qué no se está contemplando, qué es de agrado al usuario y definitivamente qué es lo que se debería cambiar antes de comenzar el proceso de desarrollo. Los prototipos presentados en esta sección se utilizaron como una primera forma de atacar el problema y como un medio para la recolección de requerimientos.

C.3.1 Prototipo 1: Ventanas

Este prototipo fue realizado en Visual Basic .NET y su meta es dar la sensación general de cómo la aplicación va a lucir. Se eligió este lenguaje ya que es fácil de manejar y crear ventanas que el usuario puede identificar inmediatamente. El ejecutable se encuentra en Prototipos\Prototipo1_Ventanas\ PrototipoInterfaz.rar del CD entregado con la documentación técnica y de usuario de FingLab.

En este prototipo se da una idea de:

- Cómo va a lucir el menú

Figura C-1: Menú

- Los estados del sistema
- Algunos formularios básicos
    - Alta de algoritmo

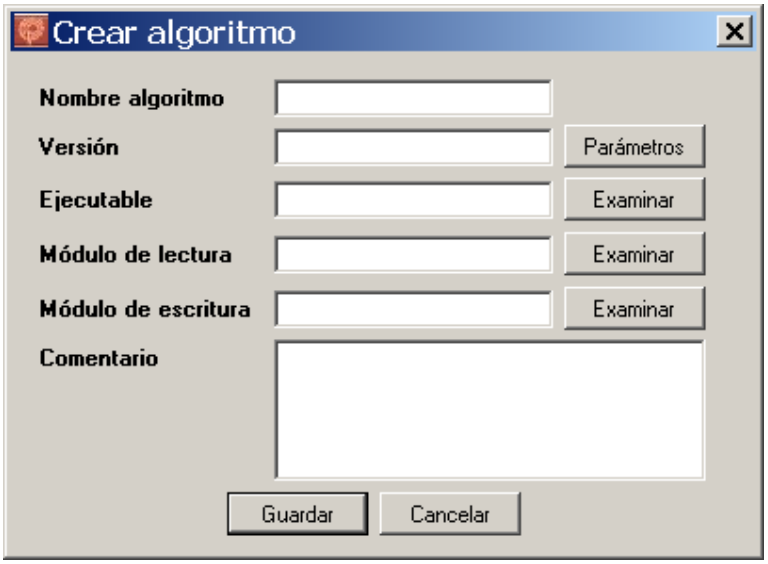


Figura C-2: Formulario de crear algoritmo

- Ingreso de parámetros

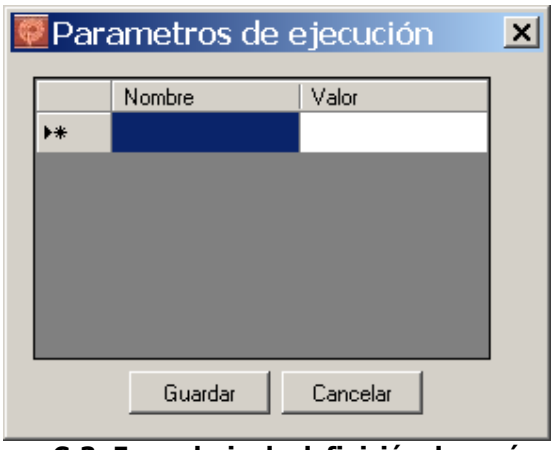


Figura C-3: Formulario de definición de parámetros

- Lanzamiento de ejecución

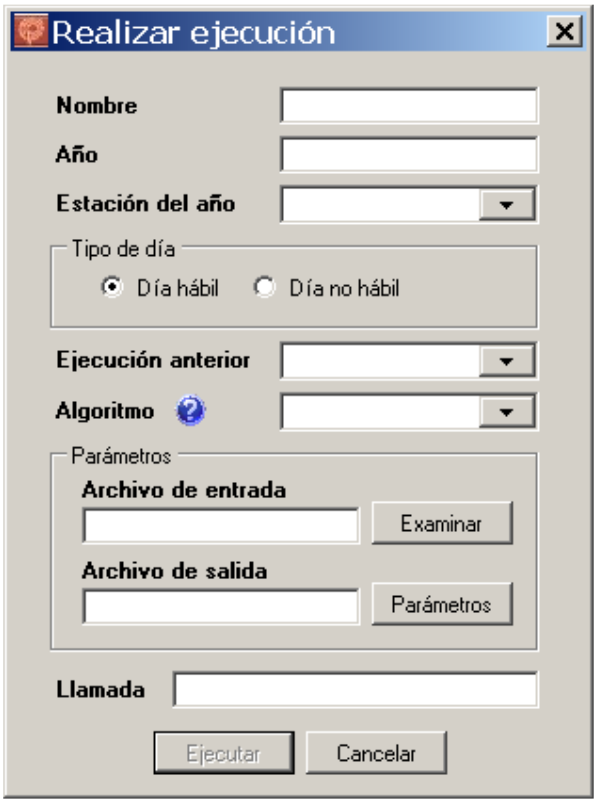


Figura C-4: Formulario de lanzar ejecución

- Manejo del versionado

Figura C-5: Árbol de planificaciones

- Cómo se abre un diario

Figura C-6: Formulario para abrir diario

- Cómo se hacen las consultas

Figura C-7: Formulario para hacer una consulta

- Cómo se hacen los armados manuales de servicios

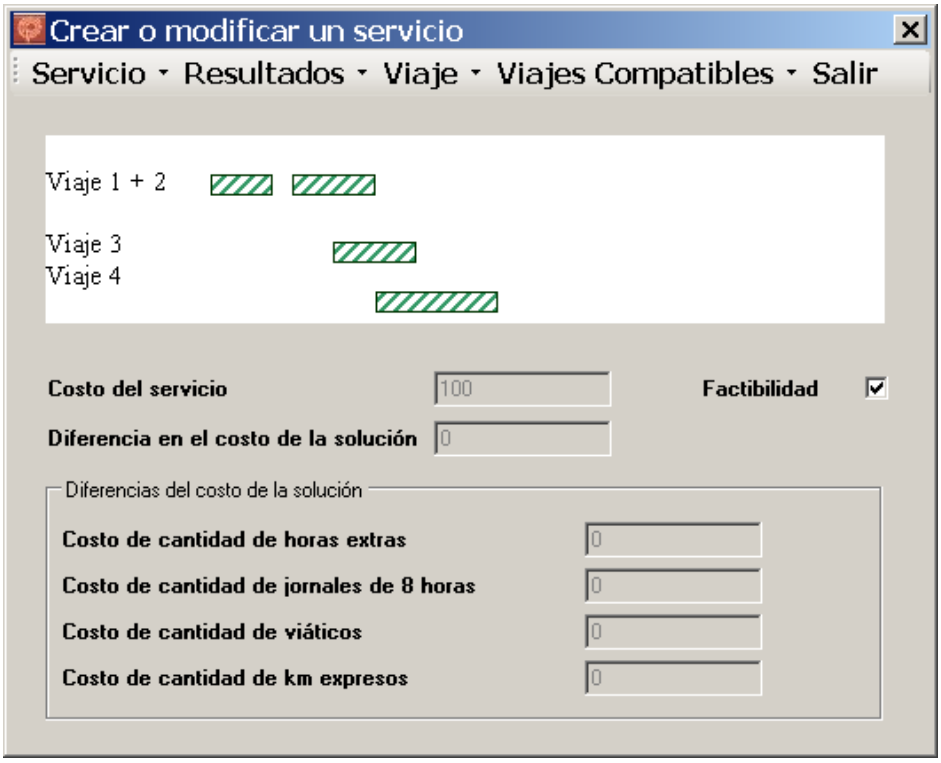


Figura C-8: Formulario para crear o modificar un servicio

La respuesta frente a este prototipo fue positiva y sirvió mucho para comprobar con los usuarios la visión global del sistema. También sirvió para chequear que las interacciones con el sistema fueran las que los usuarios esperaban según los requerimientos descriptos.

C.3.2 Prototipo 2: Arrastrar y soltar (drag and drop)

Este prototipo fue realizado en html con javascript y su meta es dar la sensación general de cómo se pueden armar los servicios manualmente utilizando la visualización de diagrama de Gantt por medio de arrastrar y soltar. Se eligió esta forma de implementar el prototipo ya que es fácil de hacer. El ejecutable se encuentra en Prototipos\Prototipo2_DragAndDrop\ Prototipo2_DragAndDrop.html del CD entregado con la documentación técnica y de usuario de FingLab.

En este prototipo se da una idea de cómo se puede armar un viaje de forma visual.

Figura C-9: Diagrama de Gantt con los viajes para armado de un nuevo servicio

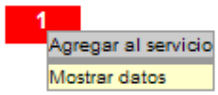


Figura C-10: Menú de botón derecho del mouse

Zonas de la interfaz

Como se puede ver en la Figura C-9, se presenten dos zonas en la interfaz, una donde se realiza el armado del nuevo servicio y otra zona en la que se despliegan resultados de consultas de donde se sacan los viajes a incorporar al nuevo servicio. La zona de resultados debe proveer de scroll (barras de desplazamiento) ya que las consultas necesarias para obtener viajes que puedan incorporarse al nuevo servicio pueden ser muchos.

Datos de los resultados de la consulta

Los resultados se muestran ya ordenados cronológicamente para permitir una mejor visualización del lugar dentro de la zona de armado que pueden ocupar. Al pasar el Mouse sobre un resultado de una consulta (un viaje) se despliegan los datos del mismo, como se muestra en amarillo en la Figura C-9.

Dichos datos también pueden mostrarse si se da la opción en un menú que se despliega al presionar el botón derecho del Mouse sobre el viaje, como muestra la Figura C-10 (la opción para mostrar todos los datos es "Mostrar datos"). Al presionar dicha opción se despliegan los datos del viaje en una ventana aparte, como muestra la Figura C-11

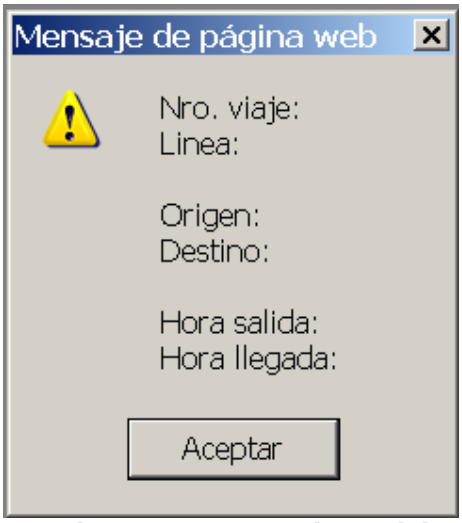


Figura C-11: Datos de un viaje

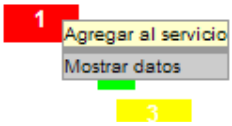


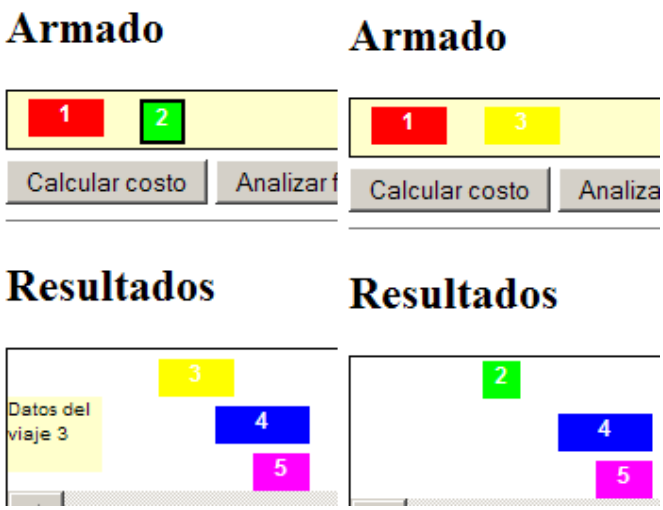
Figura C-12: Opción de agregar un viaje al servicio

Figura C-13: Armado de nuevo servicio

Incorporar un viaje al nuevo servicio

Los viajes se pueden ir incorporando al nuevo servicio mediante:

- Botones en la barra del menú del sistema. Al presionar dicha opción el viaje se saca de la zona de resultados y se coloca en la zona de armado, como muestra la Figura C-13.
- Opciones de botón derecho del Mouse. Al presionar el botón derecho del Mouse sobre un viaje se despliegan opciones para realizar acciones sobre dicho viaje, una de ellas es la opción "Agregar al servicio". Al presionar dicha opción el viaje se saca de la zona de resultados y se coloca en la zona de armado, como muestra la Figura C-12.
- Presionando el botón izquierdo del Mouse (y manteniéndolo apretado), se puede arrastrar el viaje con el Mouse y soltarlo en la zona de armado. Al arrastrar el viaje a la zona de armado, tan pronto como el viaje se superpone con dicha zona, se puede soltar el botón del Mouse y la interfaz se encarga de acomodar el viaje en la zona de armado, como muestra la Figura C-13.



Antes **Después**

Figura C-14: Chequeo de factibilidad

Chequeo de factibilidad

Si al pasar el Mouse por un viaje dentro de la zona de resultados, dicho viaje se superpone con un viaje que ya corresponde a la secuencia del nuevo servicio, dicho viaje se marca con un borde para notar la diferencia.

En el ejemplo de la Figura C-14, el Mouse se coloca sobre el viaje 3, pero el mismo se superpone con el viaje 2 que ya corresponde a la secuencia del nuevo servicio. El borde resaltado del viaje 2 advierte al usuario que si intenta incorporar el viaje 3 al nuevo servicio, el viaje 2 se sacará de la secuencia del nuevo servicio y volverá a la zona de resultados como muestra la parte derecha de la Figura C-14.

Figura C-15: Chequeo de compatibilidad

Chequeo de compatibilidad

Si al pasar el Mouse por un viaje dentro de la zona de resultados, dicho viaje no se superpone con un viaje que ya corresponde a la secuencia del nuevo servicio, pero ambos no son compatibles, es decir, no es posible que un ómnibus pueda ir desde el destino de un viaje y llegar a tiempo al origen de otro.

En el ejemplo de la Figura C-15, el Mouse se coloca sobre el viaje 5, pero el mismo no es compatible con el viaje 3 que ya corresponde a la secuencia del nuevo servicio. El borde punteado del viaje 3 advierte al usuario que no es posible incorporar el viaje. Si dicho viaje se arrastra a la zona de armado (o si se presiona la opción correspondiente de botón derecho del Mouse), al soltarlo se despliega un aviso como muestra la Figura C-16 y el viaje 5 se devuelve a la zona de resultados.

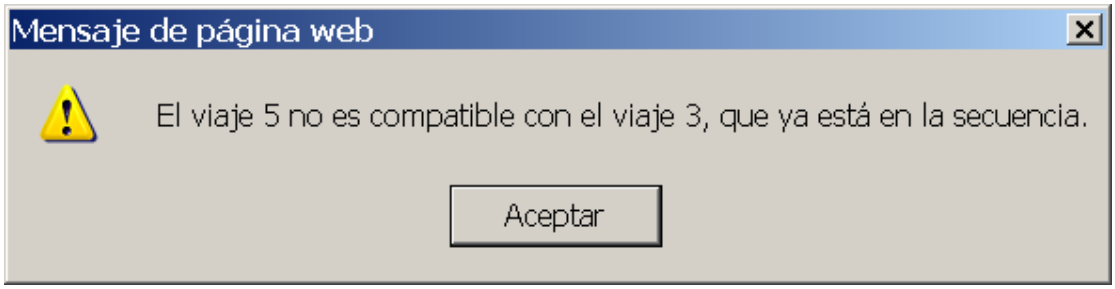


Figura C-16: Mensaje de advertencia de compatibilidad

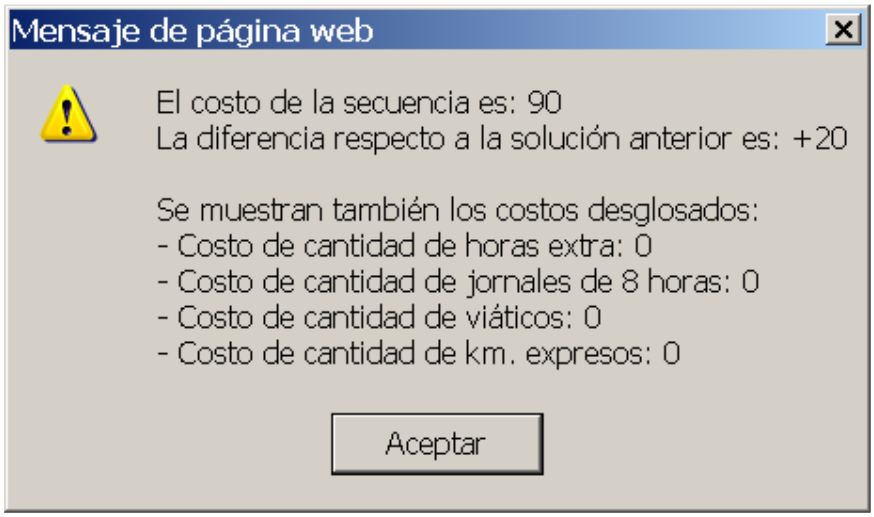


Figura C-17: Costos de la solución y el nuevo servicio

Cálculo de costos

En cualquier momento se puede calcular el costo tanto de la secuencia del nuevo servicio que se está armando, como de la solución con el nuevo servicio incorporado. De esta manera se puede apreciar si el costo de la solución ha mejorado o no, como muestra la Figura C-17.

Figura C-18: Autocompletar con un servicio expreso

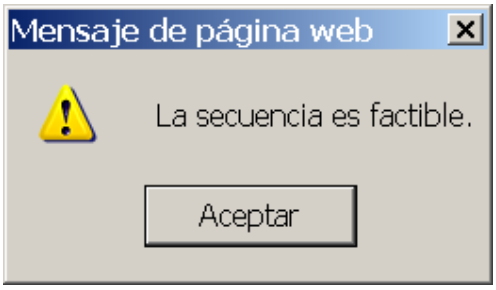


Figura C-19: Mensaje de secuencia factible

Autocompletar viajes expresos

Al finalizar el armado se puede chequear la factibilidad completa, esto agrega los viajes expresos necesarios para que todo el servicio pueda realizarse como muestra la Figura C-18. Luego se despliega el mensaje de la Figura C-19 y la secuencia está lista para ser guardada. Al guardar se deben actualizar todos aquellos servicios de los cuales se removieron viajes para formar el nuevo servicio. Esta nueva solución se debe guardar como un nuevo diario, para así mantener las versiones originales producidas por el algoritmo.

La respuesta frente a este prototipo fue positiva y sirvió mucho para que los usuarios pudieran comprobar cómo se puede mediante la visualización realizar el armado de servicios que es de particular interés en el proyecto.

C.3.3 Prototipo 3: Llamadas a programas de C desde java

Este prototipo fue realizado en C y Java y su meta es comprobar que se puede hacer llamadas a programas de C desde Java, lo cual es requerimiento para realizar las ejecuciones desde la aplicación en Java de los algoritmos que están programados en C.

Se implementó un programa en C que recibe como parámetros la cantidad de segundos que debe demorar antes de terminar su ejecución. El ejecutable se encuentra en Prototipos\Prototipo3_Llamada_a_C_desde_Java\ pruebac.rar del CD entregado con la documentación técnica y de usuario de FingLab.

Desde el punto de vista técnico esta prueba permite a los programadores seguir avanzando con Java como un lenguaje viable para realizar el proyecto.

C.3.4 Prototipo 4: Intercambiar clases en Java (reflection)

Este prototipo fue realizado en Java y su meta es comprobar que se pueden intercambiar dos clases que implementan una misma interfaz en tiempo de ejecución y ejecutar funciones de las mismas, lo cual es requerimiento si deseamos intercambiar los módulos de cálculo de costos, factibilidad, etc.

Se implementó un proyecto en Java que contiene dos clases que implementan una misma interfaz y en tiempo de ejecución intercambia las clases. El ejecutable se encuentra en Prototipos\Prototipo4_Crear_nuevas_clases_en_Java\ PruebaImpl.rar del CD entregado con la documentación técnica y de usuario de FingLab.

Desde el punto de vista técnico esta prueba permite a los programadores seguir avanzando con Java como un lenguaje viable para realizar el proyecto. En el ejemplo de la Figura C-20, el sistema deja de utilizar el módulo M, por el módulo N que implementa la misma interfaz. El sistema cuando llama a una función definida en la interfaz, ejecuta la del módulo al que esté vinculado en ese momento.

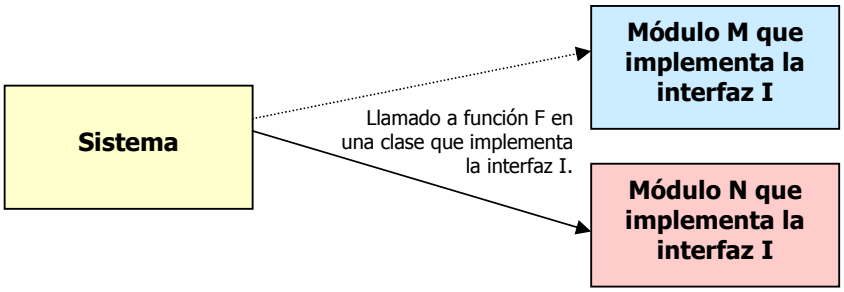


Figura C-20: Sustitución de módulos

C.4 Análisis de visualizaciones

Esta sección contiene el análisis de dimensiones del proyecto presentado. Para realizar este estudio nos basamos en la siguiente información:

1. requerimientos del problema
2. entrevistas con el cliente
3. investigación en el campo (análisis de herramientas en el mercado enfocadas al área de planificación de transporte)

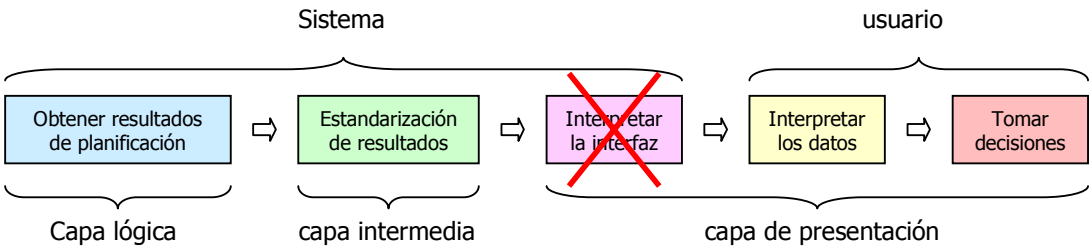
La meta de este estudio es analizar todas las posibles representaciones visuales según las dimensiones del problema y encontrar aquella(s) que cumplan con los requerimientos establecidos, así como ofrecer al usuario el apoyo necesario para visualizar información así como dar soporte a la toma de decisiones.

C.4.1 Dimensiones del problema

Se entiende por dimensión del problema una posible métrica, variable o elemento que se desea representar visualmente o que se puede deducir por medio de la interpretación de la visualización de datos en forma gráfica. Según los requerimientos, las dimensiones que podemos identificar son:

- Tiempo: al tratarse de scheduling o planeamiento la dimensión del tiempo es vital para saber qué está ocurriendo en cada momento con los recursos que se están manejando.
- Servicios: los conjuntos de viajes que hemos construido o que estamos construyendo manualmente.
- Turnos: la separación de los servicios según las normas de trabajo.
- Eventos: viajes o períodos de tiempo en los que los recursos están cumpliendo una función determinada:
    - Descansos de la persona
    - Tiempos de descanso
    - Viajes comunes / Personas trabajando
    - Viajes expresos
    - Viajes de chofer como pasajero
    - Disponibilidad por un período
    - Disponibilidad por el día
- Lugares: los lugares donde se realizan los eventos.
- Ómnibus: los medios en los cuales se realizan los servicios.
- Viajes compatibles: poder comparar viajes para saber si se pueden concatenar para formar una secuencia.

La percepción de la interfaz debe ser natural, para que la interpretación de la misma propiamente no sea un problema, sino que el problema se centre realmente en lo que se debe resolver, que es interpretar los datos y poder tomar las decisiones basado en ellos. Para ello la interfaz debe ser intuitiva, corresponderse con la visión mental que la persona tiene del problema y los datos deben visualizarse de forma clara y su entendimiento debe ser inmediato ("estoy viendo X", un viaje o lo que sea). Lo que le queda al usuario es interpretar los datos ("qué significa X", qué puedo hacer con esta información, cómo puedo usarla para comparar, intercambiar, etc.). nuestro objetivo con la visualización es que sea un pasaje natural y no un obstáculo:



En base a las dimensiones identificadas anteriormente existen otras dimensiones que se pueden deducir según lo que se represente visualmente:

- Persona-tiempo: lo que hace una persona en el correr del tiempo
- Persona/tiempos de descanso-tiempo: lo que hace una persona o los tiempos de descanso a lo largo del tiempo.
- Persona-lugar: cómo una persona se mueve de un lugar a otro.
- Ómnibus-tiempo: cómo un ómnibus se mueve a lo largo del tiempo.
- Ómnibus-lugar: cómo se mueve un ómnibus de un lugar a otro.
- Persona/ómnibus-tiempo: cómo se mueve una persona o un ómnibus a lo largo del tiempo.
- Persona/ómnibus-lugar: cómo se mueve una persona o un ómnibus de un lugar a otro.
- Lugar/ómnibus-tiempo: si un ómnibus está en un lugar en un momento determinado (esto define si un viaje es compatible).

Lugar y tiempo son dos medidas que no tienen sentido por sí solas, es su resultado lo que define una persona, un viaje, un ómnibus, un tiempo de descanso, etc.

Los tiempos de descanso y los viajes compatibles comparten características como que un ómnibus no puede estar en un lugar si no tiene tiempo de llegar y una persona no puede estar en un lugar si no está allí, o no puede llegar allí a tiempo o si no descansó lo suficiente.

Un ómnibus, una persona y un servicio son lo mismo, para la realidad de nuestro problema, todos los viajes de un servicio son realizados por un mismo ómnibus y la(s) misma(s) persona(s), con lo cual podemos reducir dimensiones.

A continuación analizaremos las posibles visualizaciones que podemos realizar con estas dimensiones según el problema de transporte que estemos enfrentando.

C.4.2 Visualizaciones del problema

En esta sección analizaremos las posibles visualizaciones que podemos realizar con estas dimensiones según el problema de transporte que estemos enfrentando (ISP, VSP o CSP).

C.4.2.1 ISP

Para el caso del problema de scheduling integrado nos interesa ver los resultados por servicio y por persona en el tiempo. Entonces si tenemos un panel visual que nos permita seleccionar qué servicio queremos ver, podemos tener un gráfico por lugares a lo largo del tiempo y por persona a lo largo del tiempo:

Panel de selección de servicios:

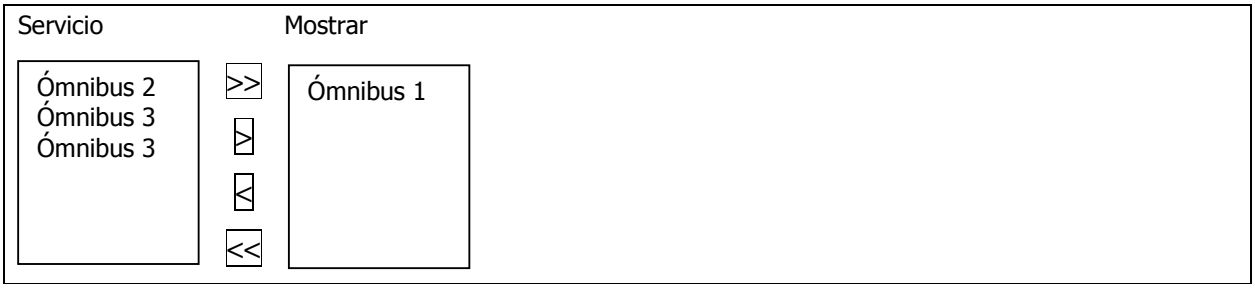


Figura C-21: Selección de servicios

Resultado de la visualización:

Si dibujáramos los puntos en un mapa se perdería la dimensión del tiempo, por eso proponemos este tipo de visualización:

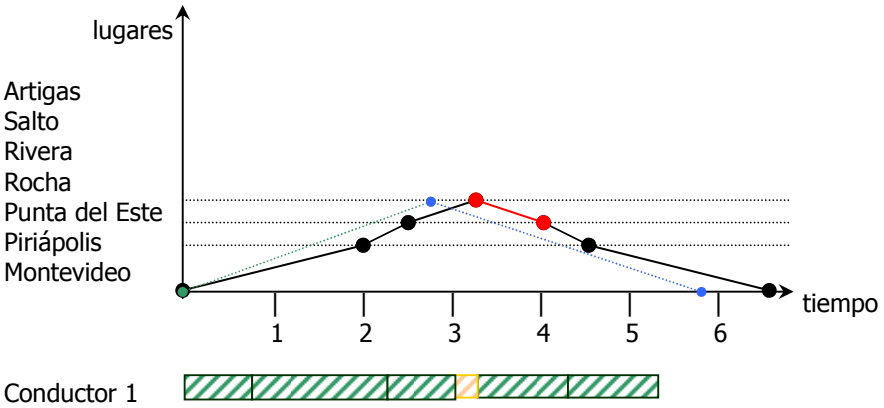


Figura C-22: Diagrama de puntos mostrando resultados

Es un gráfico de puntos, donde cada punto representa un ómnibus, un conductor, un lugar y una hora. Pero no se puede dejar los puntos solos, se los debe unir para que quede la idea de que todos los puntos corresponden a un mismo ómnibus.

Si quisiéramos en un gráfico así hacer una consulta de viajes compatibles podríamos, por ejemplo, al seleccionar el viaje en rojo da como resultado de viajes compatibles el primer tramo en verde que ya está como parte de otro servicio (la secuencia en azul).

En la gráfica de conductores podemos ver los siguientes eventos:

- Trabajando
- Descansado o en espera
- Viajando como pasajero

En la gráfica de lugares podemos ver los siguientes eventos:
- Viaje común
- Viaje expreso
- Descanso (cuando el ómnibus no cambia de lugar)

Fortalezas	Debilidades
El paso del tiempo en el eje x es más natural e intuitivo.	Las líneas no representan nada: el que se crucen no representa que los trayectos realmente se corten, los largos de las líneas no guardan la proporcionalidad respecto a la distancia o tiempo que representan.
Al mostrar los lugares en el eje y se tiene la sensación de movimiento de espacio en relación al tiempo.	No sirve para comparar por las proporciones ni se puede deducir nada de estos gráficos. Además cuando hay muchos queda demasiado entreverado.
Los lugares pueden estar ordenados por distancia para que las gráficas queden mejor.	El orden no evita los cruces de más.

Conclusiones:

Es mejor que los viajes se representen en el tiempo como barras horizontales (tal como en la visualización de los conductores) ya que el graficar los lugares pierde el sentido en la gráfica. El tipo de gráfico por lugar puede que se utilice solo para las manipulaciones manuales y para la información total (mostrar un diario entero) que sean solo tablas comunes como muestra la siguiente figura:

servicio	Turno 1			Turno 2
Servicio 1	MVD – Salinas 9:30 – 11:30	Evento 2	…	
Servicio 2	…	…		
…				

Figura C-23: Tabla de servicios

De esta manera el tiempo transcurre de forma horizontal en un servicio. Queda como el gráfico pero no proporcional (las celdas de la tabla son siempre del mismo tamaño), lo que permite desplegar los datos correctamente.

Estas tablas se tienen que poder ver en pantalla lo más posibles (poder hacer zoom o que de alguna manera se puedan ver todas las columnas) y poder imprimir bien cuando está la hoja horizontal.

Quedaría ver cómo se podrían hacer los cambios cuando se quiere intercambiar viajes entre unidades y lo que pasa con los conductores, si se intercambian también o no.

C.4.2.2 CSP

Para el caso del problema de scheduling de conductores nos interesa ver los resultados por persona en el tiempo. Entonces si tenemos un panel visual que nos permita seleccionar qué conductor queremos ver, podemos tener un gráfico por lugares a lo largo del tiempo y por persona a lo largo del tiempo:

Vista 1

Panel de selección de personas:

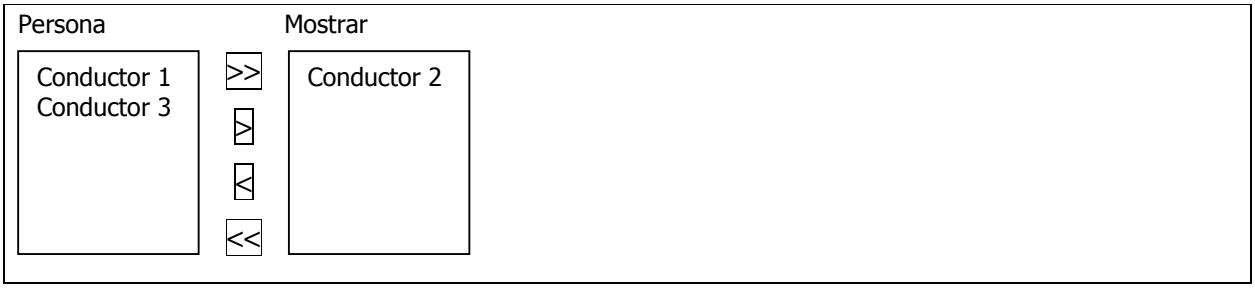


Figura C-24: Selección de personal

Resultado de la visualización:

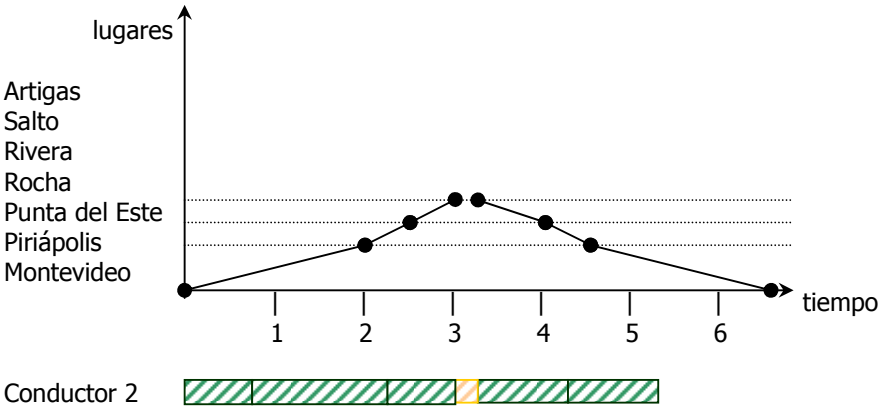


Figura C-25: Diagrama de puntos mostrando los resultados

Al seleccionar en la lista al conductor 2 se muestra los cambios de viaje que hace y una gráfica con los tiempos de descanso que tiene. Cada rectángulo verde tiene información del viaje que representa.

Esta visualización por lugares tiene los mismos problemas de interpretación que se especificaron para el caso ISP.

Vista 2 (especialmente para comparar conductores)

Panel de selección de personas:

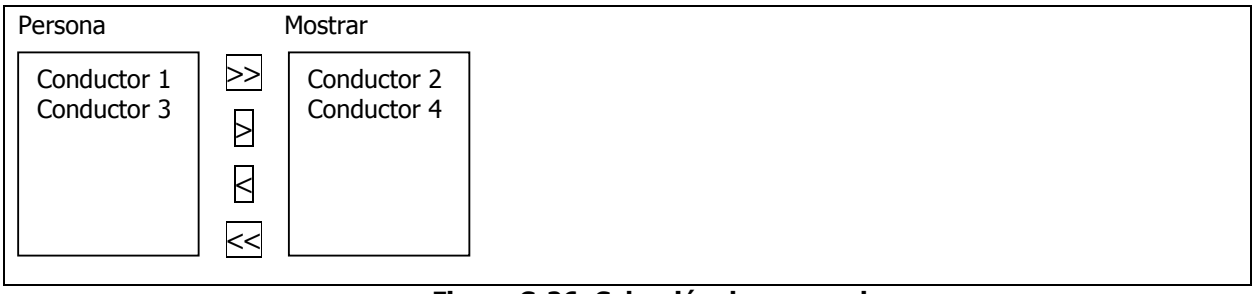


Figura C-26: Selección de personal

Resultado de la visualización:

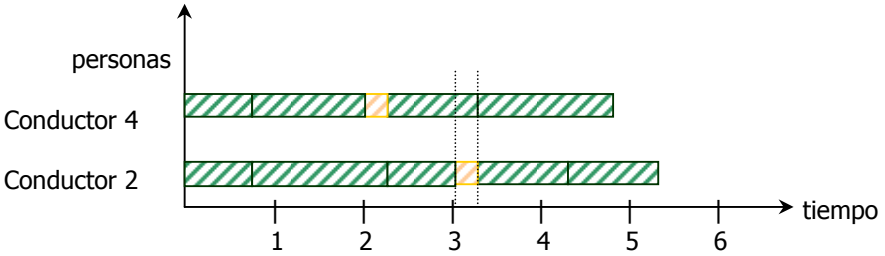


Figura C-27: Diagrama de Gantt con la visualización de resultados

Al seleccionar en la lista a los conductores se muestra una gráfica con los tiempos de descanso que tienen. Cada rectángulo verde tiene información del viaje que representa.

Conclusiones:

La vista 1 sigue presentando los problemas de mala interpretación que ya se observaron en el caso ISP.

En la vista 2 vemos que el paso del tiempo en el eje x es más natural e intuitivo. Al mostrar las personas en el eje y se puede ver la duración de los eventos y se tiene la sensación de movimiento de espacio en relación al tiempo sin necesariamente graficar los lugares. Es fácil de comparar los tiempos de descanso compatibles para intercambiar personal. Esta vista también funciona si en vez de conductores consideramos servicios u ómnibus, por lo que constituye una forma de visualización muy viable para nuestro caso.

C.4.2.3 VSP

Para el caso del problema de scheduling de vehículos tiene las dos vistas como CSP pero con ómnibus en vez de personas.

C.4.2.4 Otras herramientas

Como hemos visto en el estudio de herramientas similares en el mercado, dirigidas al scheduling de vehículos, todas incluyen visualizaciones de tipo diagrama de Gantt. Solo incluyen mapas cuando se trata de problemas de ruteo. Ver el anexo Herramientas transporte.

C.4.2.5 Opinión de expertos

En una reunión con Eduardo Fernández (docente de Facultad de Ingeniería y encargado del curso "Interacción Persona-Computadora"), discutimos todas estas visualizaciones y finalmente la visualización en forma de diagrama de Gantt es la única que coincide en que refleja todas las dimensiones del problema.

C.4.3 Conclusiones

Como resultado de el análisis de posibles visualizaciones tenemos que:

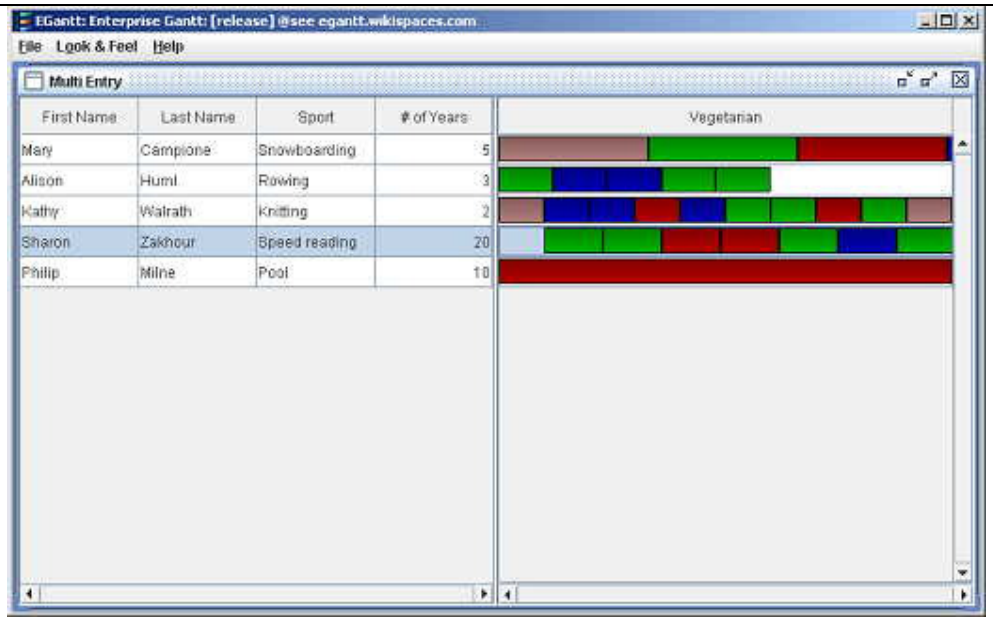
1. Al dibujar los recorridos en un mapa perdemos la dimensión del tiempo, la cual es vital si estamos trabajando con scheduling (si estuviéramos trabajando con problemas de ruteo entonces sí sería viable una visualización de tipo mapa).
2. Al dibujar los lugares en un eje se pierde el orden, las distancias, se pueden confundir los largos de los trayectos en el gráfico con la duración o la distancia de dichos trayectos, lo cual es erróneo.
3. El paso del tiempo en el eje x es más natural e intuitivo.
4. Al dibujar sobre una línea de tiempo en el eje x, si se dibujan personas u ómnibus sobre el eje y, sí se tiene una correcta intuición de la duración de los eventos y se obtiene la sensación de movimiento en el tiempo (y también en el espacio).
5. Al dibujar sobre una línea de tiempo en el eje x, es fácil de ver por qué un evento es compatible con otro. El que estén en el mismo lugar se puede manejar filtrando los resultados en la consulta de eventos compatibles sin necesariamente graficar dichos lugares (el manejo de lugares se hace por medio de lógica y visualmente se reconoce que si un viaje es compatible es porque comparten lugares de origen o destino).

En conclusión, la visualización más efectiva para el problema de scheduling es un **diagrama de Gantt**. Permite visualizar o intuir todas las dimensiones del problema de forma clara y provee de todos los elementos que ayudan a seguir la línea de pensamiento del usuario.

C.5 Herramientas Gantt

En esta sección se presentan las herramientas relevadas que realizan gráficos de Gantt. Toda la información e imágenes presentadas en cada ficha fue extraída del link indicado en la misma.

E-Gantt
Características
Link: http://egantt.wikispaces.com/ (último acceso: 22/03/2010) *Versión*: 0.5.3 *Lenguaje*: Java *Generalidades*: E-Gantt es una librería de gráficos Gantt implementada en Java Swing para la visualización de tablas de ordenamiento. La librería es tipicamente usada para editar y visualizar horarios de trabajo complicados. Fue desarrollada por Keith Long y ya no está en desarrollo. La librería E-Gantt ha sido integrada con éxito en muchos proyectos de código abierto y en proyectos comerciales en las áreas de: investigación de ordenamiento, investigación médica, proyectos de defensa militar de EEUU y herramientas de administración de redes. Es de licencia LGPL, por lo cual es libre para uso comercial y no comercial.



Ventajas	Desventajas
❖ Es de licencia GLPL. ❖ La interfaz es simple. ❖ Es portable, ya que está programado en Java.	❖ La documentación es escasa. ❖ Visualmente es simple y útil, pero no muy atractiva. ❖ El código fue muy complejo de entender para realizar las modificaciones necesarias para nuestro proyecto.

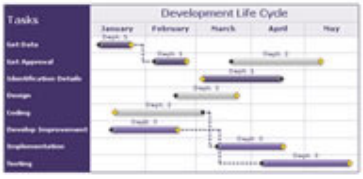
Resultado

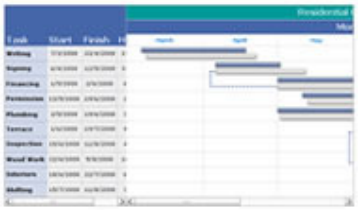
Tuvimos problemas para poder realizar un ejemplo simple de prueba para verificar su validez, por lo cual lo descartamos. La complejidad de la implementación y la documentación que no es muy amplia no permitieron que se pudiera realizar un prototipo usando esta librería.

FusionCharts

Características

Link: http://www.fusioncharts.com/ (último acceso: 22/03/2010)
Versión: 3
Lenguaje: existen diferentes distribuciones para ser utilizadas con diferentes tecnologías.
Generalidades: Permite crear gráficos interactivos y animados para aplicaciones. Permite su ejecución en PCs, Macs, iPads, iPhones y la mayoría de dispositivos móviles. Utiliza Flash y JavaScript para crear gráficos y funciona con datos en formato XML y JSON. Puede ser integrado con cualquier tecnología de servidor (ASP, ASP.NET, PHP, JSP, ColdFusion, Ruby, etc.) y bases de datos. Es usado por más de 17.000 clientes y 330.000 usuarios en más de 110 países. No es gratuito, solo se puede usar bajo licencia.





Ventajas	Desventajas
❖ Visualmente es excelente, como se puede ver en las imágenes y en las demos. ❖ Permiten un alto grado de personalización. ❖ Gran compatibilidad. ❖ Muy buena documentación.	❖ No es gratuito. ❖ No lo pudimos probar. ❖ Funciona si se utiliza en Web y con un servidor (nosotros preferimos algo que funcione como aplicación de escritorio).

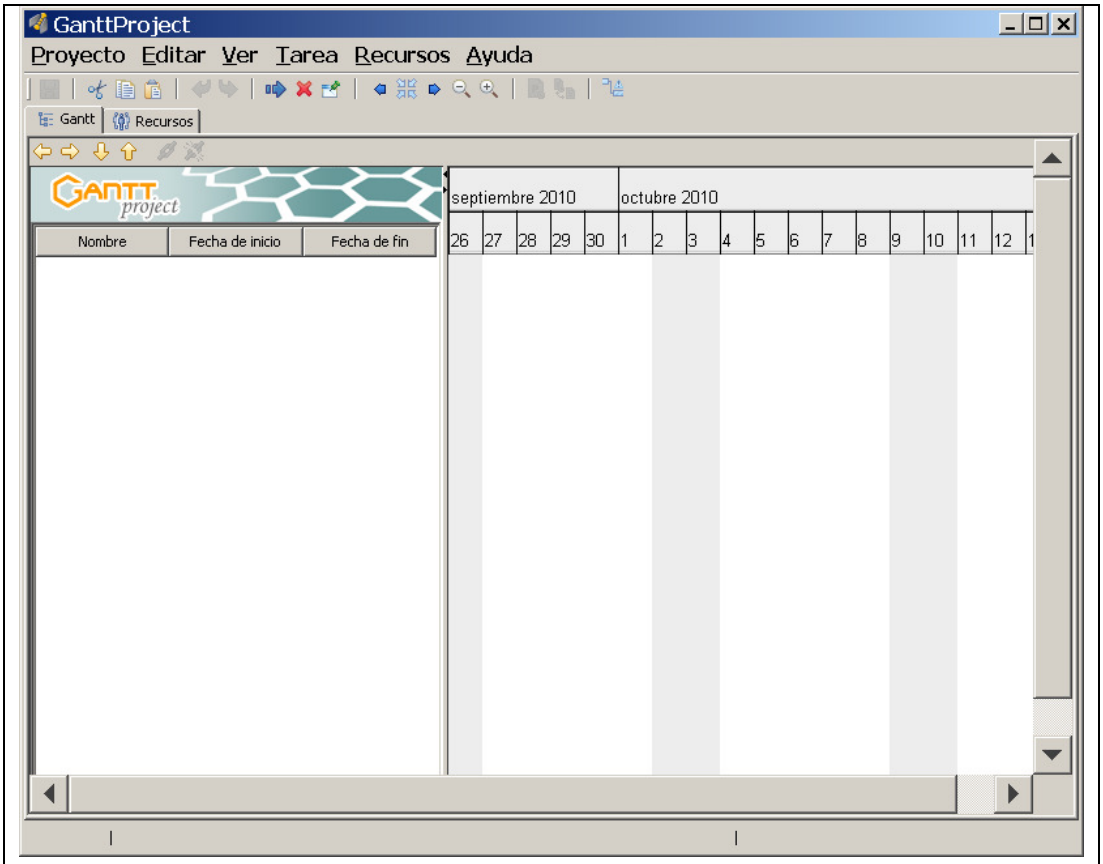
Resultado

Al no ser gratuito no lo hemos podido utilizar, pero las demos en línea son excelentes y la calidad visual es excelente.

GanttProject

Características

Link: http://www.ganttproject.biz/ (último acceso: 22/03/2010)
Versión: 2.0.7
Lenguaje: Java
Generalidades: GanttProject es una herramienta de escritorio multi-plataforma para administración y ordenamiento de proyectos. Es de licencia libre y su código es abierto. Permite crear gráficos de Gantt, dibujar dependencias, marcar metas, asignar recursos humanos, generar un gráfico Pert a partir del gráfico Gantt, exportar imágenes y generar reportes pdf y HTML.



Ventajas	Desventajas
❖ Es de licencia libre y código abierto. ❖ Es una aplicación de escritorio. ❖ Buena documentación. ❖ Muy buenas funcionalidades, muy completas. ❖ Es portable, ya que está programado en Java.	❖ El código fue un poco difícil de entender para realizar las modificaciones necesarias para el proyecto. ❖ Tiene muchas librerías extra que necesita para poder correr.

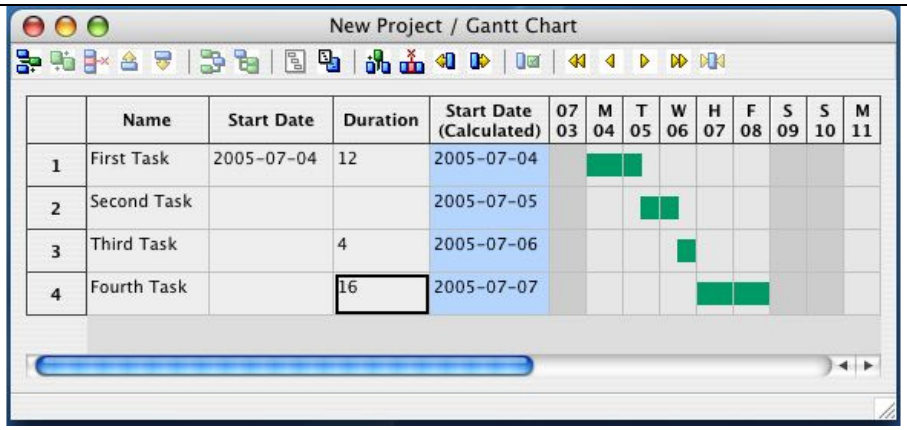
Resultado

Fue considerado para realizar el proyecto, pero finalmente se eligió otra herramienta.

GanttPV

Características

Link: http://www.pureviolet.net/ganttpv/ (último acceso: 22/03/2010)
Versión: 0.11
Lenguaje: Python
Generalidades: GanttPV permite definir tareas, duraciones de las mismas, dependencias, fechas de comienzo y feriados. Basado en esta información calcula las fechas de fin de cada tarea y crea el gráfico Gantt que puede ser usado para indicar cuando las tareas son completadas. También permite la administración de los recursos humanos asignados a las tareas. Es multi-plataforma y de licencia libre.

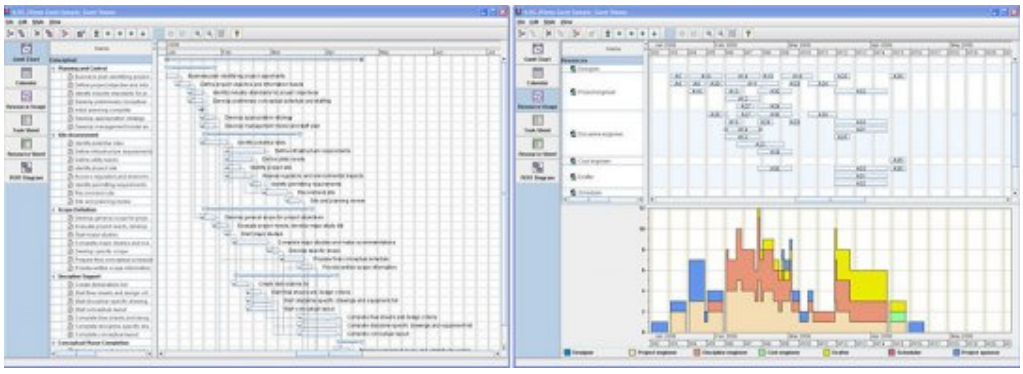


Ventajas	**Desventajas**
❖ Es de código abierto. ❖ Es multi-plataforma. ❖ Visualmente es agradable. ❖ Buena documentación.	❖ El lenguaje Python no es familiar para los integrantes del grupo por lo que implica una tarea extra. Sigue siendo una preferencia el lenguaje Java.
Resultado	
No se utilizó por preferir una herramienta que estuviese implementada en Java.	

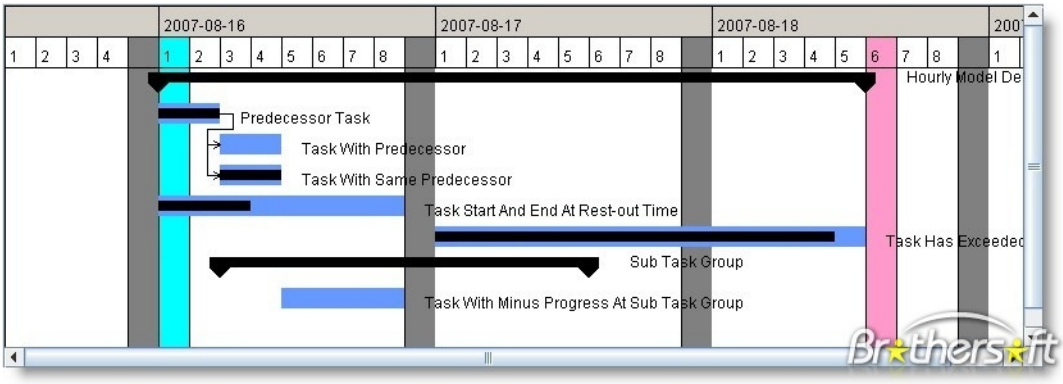
ILOG

Características

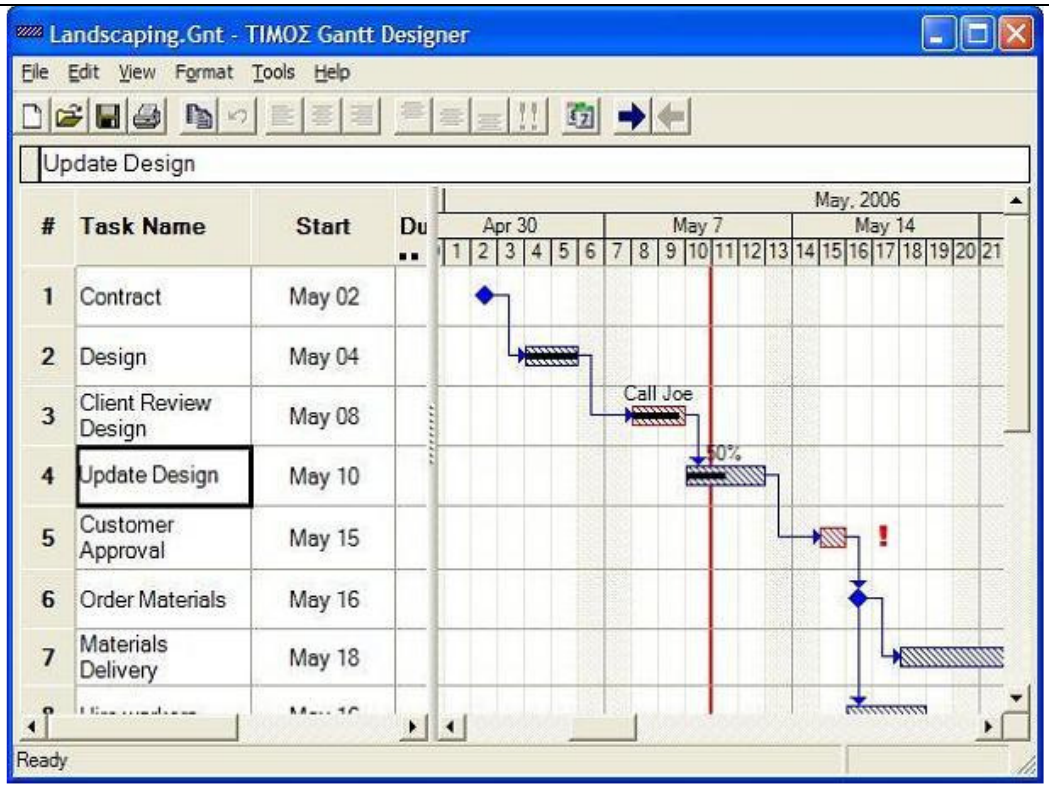
Link: http://www-01.ibm.com/software/integration/visualization/java/ (último acceso: 22/03/2010)
Versión: 8.7
Lenguaje: Java
Generalidades: IBM ILOG son componentes para visualización en Java que proveen, componentes gráficos y personalizables para visualizaciones de escritorio, Ajax y Eclipse. Estos componentes permiten a los desarrolladores Java agregar gráficos intuitivos para sus aplicaciones. Permiten realizar diferentes tipos de gráficos, incluyendo de Gantt, llamado JViews Gantt. Provee configurar los tiempos, tareas, recursos, calendarios y vistas de camino crítico. Incluye un diseñador de gráficos de tipo "apuntar y hacer click" que permite configurar visualizaciones con código mínimo. No es de código abierto y no es gratuito.



Ventajas	**Desventajas**
❖ Visualmente es excelente. ❖ Altamente configurable. ❖ Es portable, ya que está programado en Java.	❖ No es gratuito. ❖ No es de código abierto. ❖ No lo pudimos probar, solo ver unas demos que sí están muy buenas.
Resultado	
No es viable su utilización ya que buscamos herramientas de tipo gratuitas y de código abierto.	

SwiftGantt	
Características	
Link: http://www.bluechillies.com/details/45130.html (último acceso: 22/03/2010) *Versión*: 0.3.6 *Lenguaje*: Java *Generalidades*: SwiftGantt es un componente de gráficos de Gantt implementado en Java Swing de código abierto. Permite definir tareas, unidades de tiempo, personalizar los colores y exportar imágenes del gráfico.	
	
Ventajas	**Desventajas**
❖ Es de código abierto y gratuito. ❖ Es simple. ❖ Muy personalizable y configurable. ❖ Es portable, ya que está programado en Java.	❖ Tuvimos problemas para poder realizar un ejemplo simple de prueba para verificar su validez, por lo cual lo descartamos. ❖ La documentación es muy escasa.
Resultado	
Fue considerado para realizar el proyecto, pero finalmente se eligió otra herramienta.	

Timios Gantt Designer
Características
Link: http://download.cnet.com/Gantt-Chart-Designer/3000-2076_4-10507121.html (último acceso: 22/03/2010) *Versión*: 2.0 *Lenguaje*: .Net *Generalidades*: Timios Gantt Designer se concentra únicamente en los diagramas de Gantt. La gestión de proyecto es un producto secundario. Los datos de texto se ingresan en una planilla. Para los datos que pueden representarse gráficamente, como son la hora de comienzo o la duración, puede usar el mouse para arrastrar y redimensionar las barras de tiempo. Tiene asistencia para la diagramación con calendario real. Especifique la cantidad de días necesarios para completar una tarea y Gantt Designer se encarga automáticamente de considerar los fines de semana y los días feriados. Se actualiza automáticamente si usted cambia la fecha de inicio. Utilice arrastrar y soltar para vincular tareas, con profusas dependencias, y las tareas afectadas se reprograman automáticamente, visual e instantáneamente. Se evitan los vínculos innecesarios. Tiene opciones de impresión flexibles, muy fáciles de configurar. Distribuye grandes diagramas en varias páginas con facilidad y tal como usted lo espera. También puede imprimir a un archivo de imagen compacto para incluirlo en páginas web. Gantt Designer es totalmente compatible con Unicode. El programa está traducido al alemán, el chino, el japonés, el italiano, el español, el francés y el portugués. Es de licencia libre.



Ventajas	Desventajas
❖ Es muy configurable. ❖ Tiene funcionalidades de arrastrar y soltar.	❖ No es portable, ya que está programado en .Net y funciona solo en sistemas Windows.

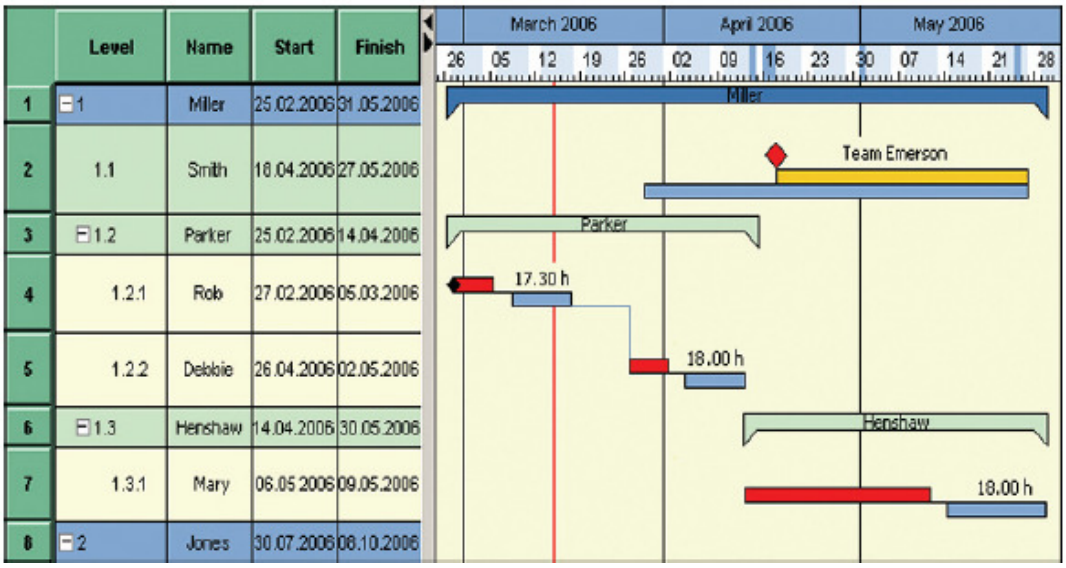
Resultado

No se utilizó por preferir una herramienta que estuviese implementada en Java.

VARChart

Características

Link: http://www.hallogram.com/jgantt/jgantt.html (último acceso: 22/03/2010)
Versión: -
Lenguaje: Java
Generalidades: VARChart JGantt, permite interacciones del usuario como crear, borrar y arrastrar y soltar actividades, modificar la escala de tiempo y tablas, colapsar o expandir grupos de actividades. Permite que el desarrollador personalice las interacciones a ciertos eventos del gráfico. Utiliza la funcionalidad arrastrar y soltar para pasar información de un gráfico a otro. No es gratuito y el código no es abierto.



Ventajas	Desventajas
❖ Visualmente es muy bueno. ❖ Altamente configurable. ❖ Permite agregar funcionalidades. ❖ Es portable, ya que está programado en Java.	❖ No es gratuito. ❖ No es de código abierto. ❖ No lo pudimos probar.

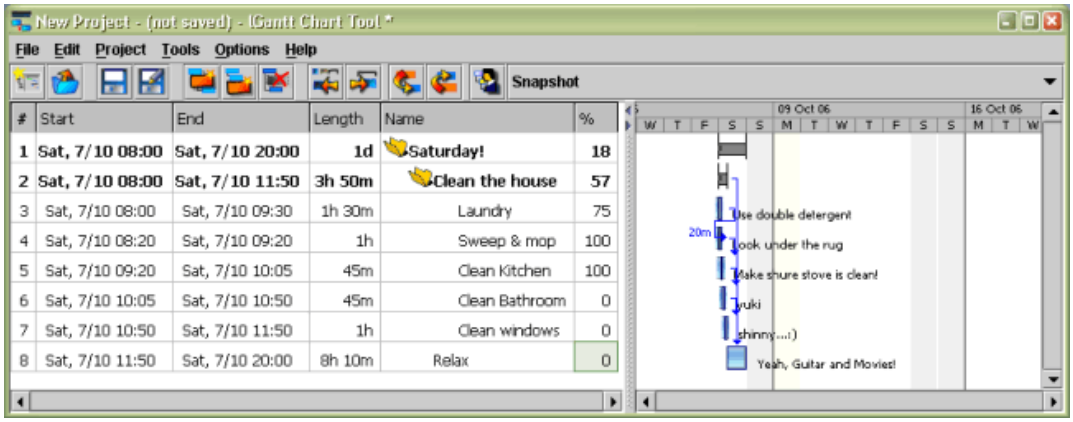
Resultado

No es viable su utilización ya que buscamos herramientas de tipo gratuitas y de código abierto.

LGantt

Características

Link: http://ghannid.homeip.net/wiki/LGanttPrd_en.wiki (último acceso: 22/03/2010)
Versión: 0.4.7
Lenguaje: Java
Generalidades: LGantt es una aplicación implementada en lenguaje Java que permite crear gráfico de Gantt y evaluar actividades basadas en el tiempo. Permite administrar tareas, restricciones, hacer y rehacer cambios, ver notas como tooltips, administrar recursos, exportar imágenes y registrar snapshots de una línea base determinada. Es de licencia libre y código abierto.



Ventajas	Desventajas
❖ Es portable, ya que está implementada en	❖ Tiene funcionalidades que no se adaptan

Java. ❖ Es gratuito y de código abierto. ❖ Posee los niveles de escala y detalle que más se adaptan a nuestro proyecto. ❖ Altamente personalizable y configurable. ❖ El código es muy claro. ❖ Funcionalidades adaptables a nuestras necesidades. ❖ Podemos agregarle funcionalidades fácilmente.	a nuestras necesidades y que deberán ser removidas sin alterar el funcionamiento de lo que sí necesitamos. ❖ Documentación escasa.
Resultado	
Está implementado en Java y es gratuito. El código es relativamente fácil de entender y seguir para poder realizar las modificaciones necesarias. Debido a que pudimos tener en poco tiempo un prototipo funcionando con este componente, fue el que elegimos para nuestro proyecto.	

C.5.1 Observaciones

Como conclusiones generales de las herramientas analizadas, buscamos una herramienta:

- ❖ Que esté implementada en Java para proveer mayor portabilidad.
- ❖ Que sea de código abierto para permitir las modificaciones necesarias al caso de ordenamiento de viajes.
- ❖ Que sea gratuito, de licencia libre.
- ❖ Que la documentación sea clara y amplia.
- ❖ Que se pueda realizar un prototipo de prueba en poco tiempo para comenzar el trabajo inmediatamente.
- ❖ Que el código esté claro y se pueda modificar lo más fácilmente posible.
- ❖ Que permita la visualización de datos en formato de tabla y gráfico. Que la interacción con la tabla se vea reflejada de alguna forma en el gráfico y viceversa.
- ❖ Que permita las funcionalidades de arrastrar y soltar para permitir el manejo de los elementos visibles en el gráfico.
- ❖ Que permita cambiar las escalas de tiempo.
- ❖ Que permita personalizar los colores, fuentes y tamaños del gráfico.
- ❖ Que permita exportar imágenes y reportes pdf o HTML.

C.6 Análisis de lenguajes e IDEs

C.6.1 Lenguaje Java

En el análisis de herramientas que implementan diagramas de Gantt encontramos a Lgantt que está implementado en Java. Pero esta no es la única razón por la cual seleccionamos este lenguaje para desarrollar este proyecto [1].

Java ofrece muchas ventajas en comparación a otros lenguajes [2]:

- Es portable, podemos ejecutarlo en otros sistemas operativos que no sean necesariamente en el que se desarrolló la aplicación.
- Se pueden crear aplicaciones que se pueden ejecutar dentro de un explorador Web
- Permite la sustitución de módulos en tiempo de ejecución (como vimos en el desarrollo de prototipos)
- Podemos ejecutar desde java cualquier programa en otro lenguaje (como comprobamos en el desarrollo de prototipos)

Cumple con todos los requisitos para nuestro proyecto y por eso decidimos seguir el desarrollo de Lgantt en Java.

Si bien otros lenguajes fueron considerados, como por ejemplo Visual Basic .NET (que fue utilizado para construir uno de los prototipos del proyecto), por experiencia y por la necesidad de portabilidad elegimos Java.

C.6.2 IDEs para Java

A la hora de implementar necesitamos un IDE que nos permita desenvolvernos con facilidad a la hora de programar. La principal diferencia entre uno y otro es si los usuarios están acostumbrados a uno en particular y si el IDE ofrece todo lo que necesitamos.

Existen IDEs que permiten construir interfaces gráficas de forma dinámica [3-7], pero estos en general resultan muy pesados a la hora de trabajar y pueden tener problemas durante su utilización (utilizan mucho el arrastrar y soltar con el Mouse y hay veces que esto puede hacer que los elementos que se tratan de construir se superpongan o desacomoden y que el editor no pueda deshacer el problema dando un error).

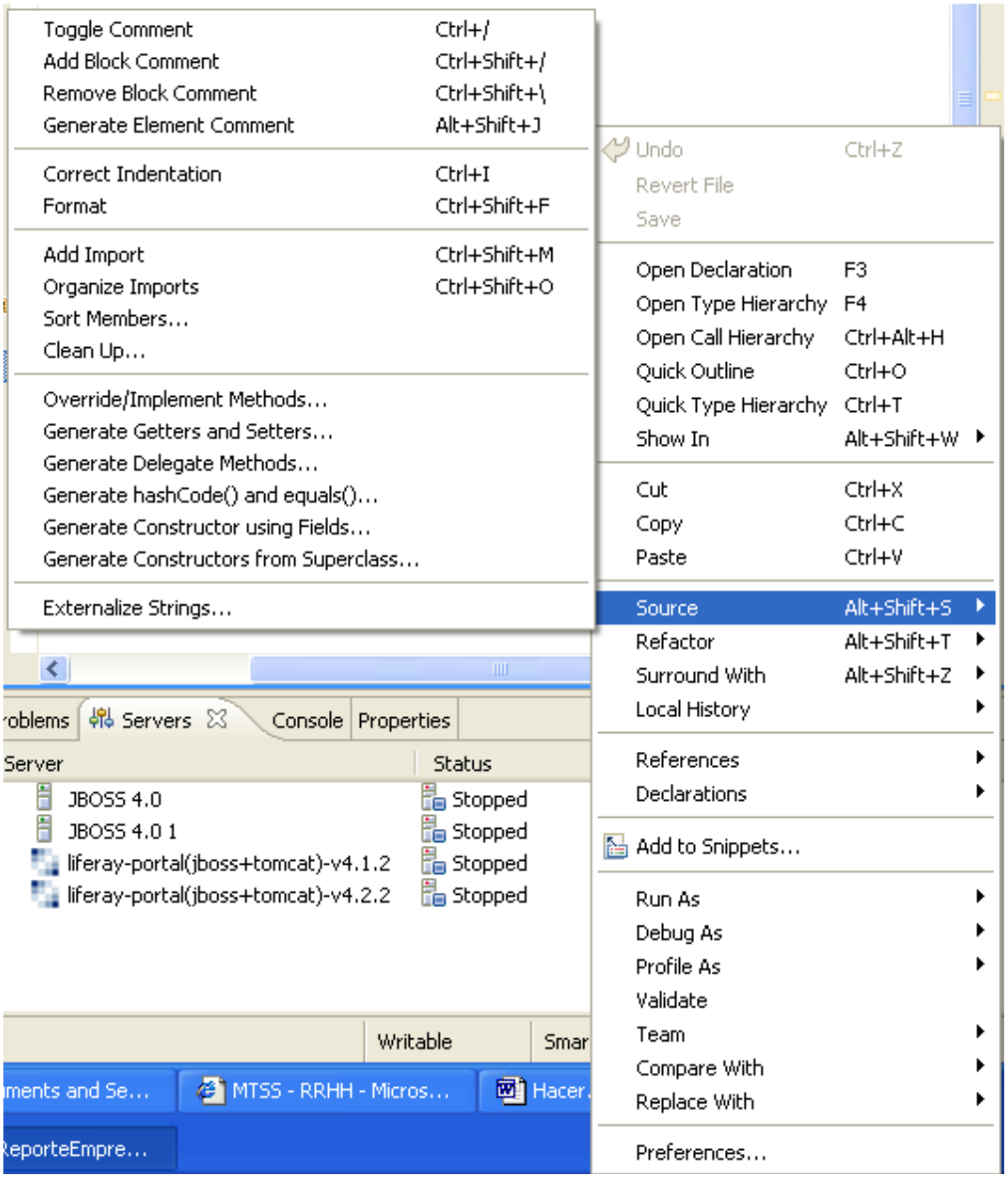
También tenemos la necesidad de generar diagramas UML para la documentación técnica del proyecto, por lo cual evaluamos varias herramientas que permitan generar estos diagramas desde el código (ya que tenemos el código de Lgantt de donde partir) y viceversa [8].

Al ver las comparaciones de IDEs [9-12] decidimos utilizar Eclipse para programar y ocasionalmente utilizar NetBeans para obtener los diagramas necesarios para la documentación a partir del código [13-16].

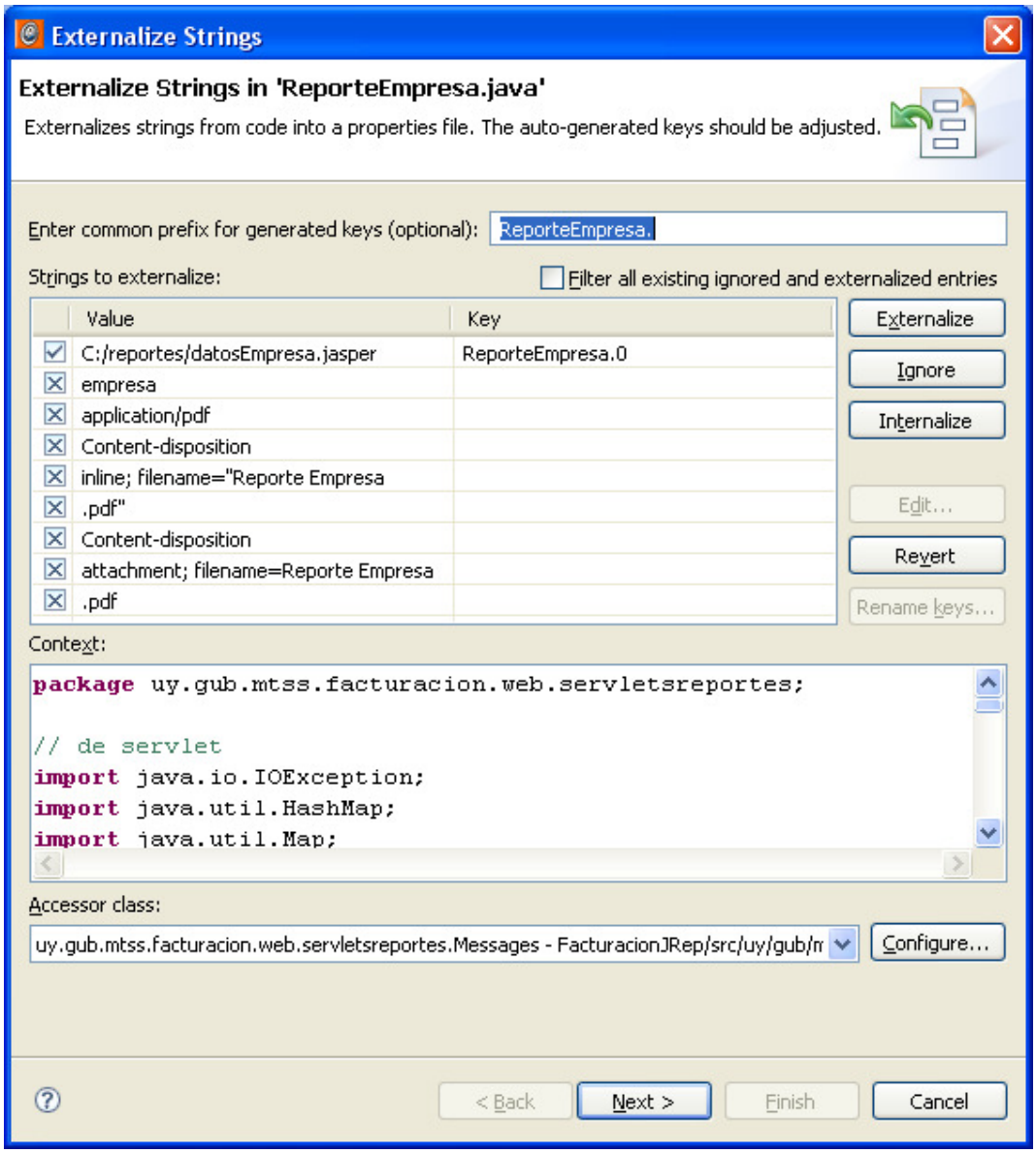
C.6.3 Utilidades de Eclipse

Algunas utilidades ofrecidas por Eclipse que nos resultaron interesantes para programar son:

- Fácil de crear proyectos
- Búsquedas en el código
- Completa los nombres de las clases, funciones, etc. A medida que se va escribiendo.
- Es liviano al ejecutar en comparación a otros IDEs (NetBeans es mucho más pesado, por eso lo usamos solo para diagramas)
- Soporte de CVS
- Con botón derecho sobre el texto de cualquier clase: se despliega un menú que permite
  - Source
    - Agregar / borrar comentario
    - Corregir indentación
    - Agregar import
    - Implementar métodos getters y setters
    - Externalize strings: al seleccionar un texto y presionar esta opción se despliega la ventana:


Toggle Comment Ctrl+/
Add Block Comment Ctrl+Shift+/
Remove Block Comment Ctrl+Shift+\
Generate Element Comment Alt+Shift+J
Correct Indentation Ctrl+I
Format Ctrl+Shift+F
Add Import Ctrl+Shift+M
Organize Imports Ctrl+Shift+O
Sort Members...
Clean Up...
Override/Implement Methods...
Generate Getters and Setters...
Generate Delegate Methods...
Generate hashCode() and equals()...
Generate Constructor using Fields...
Generate Constructors from Superclass...
Externalize Strings...
Undo Ctrl+Z
Revert File
Save
Open Declaration F3
Open Type Hierarchy F4
Open Call Hierarchy Ctrl+Alt+H
Quick Outline Ctrl+O
Quick Type Hierarchy Ctrl+T
Show In Alt+Shift+W
Cut Ctrl+X
Copy Ctrl+C
Paste Ctrl+V
Source Alt+Shift+S
Refactor Alt+Shift+T
Surround With Alt+Shift+Z
Local History
References
Declarations
Add to Snippets...
Run As
Debug As
Profile As
Validate
Team
Compare With
Replace With
Preferences...
Servers
Console
Properties
Server
Status
JBOSS 4.0 Stopped
JBOSS 4.0 1 Stopped
liferay-portal(jboss+tomcat)-v4.1.2 Stopped
liferay-portal(jboss+tomcat)-v4.2.2 Stopped
Writable
MTSS - RRHH - Micros...

Externalize strings: al seleccionar un texto y presionar esta opción se despliega la ventana:

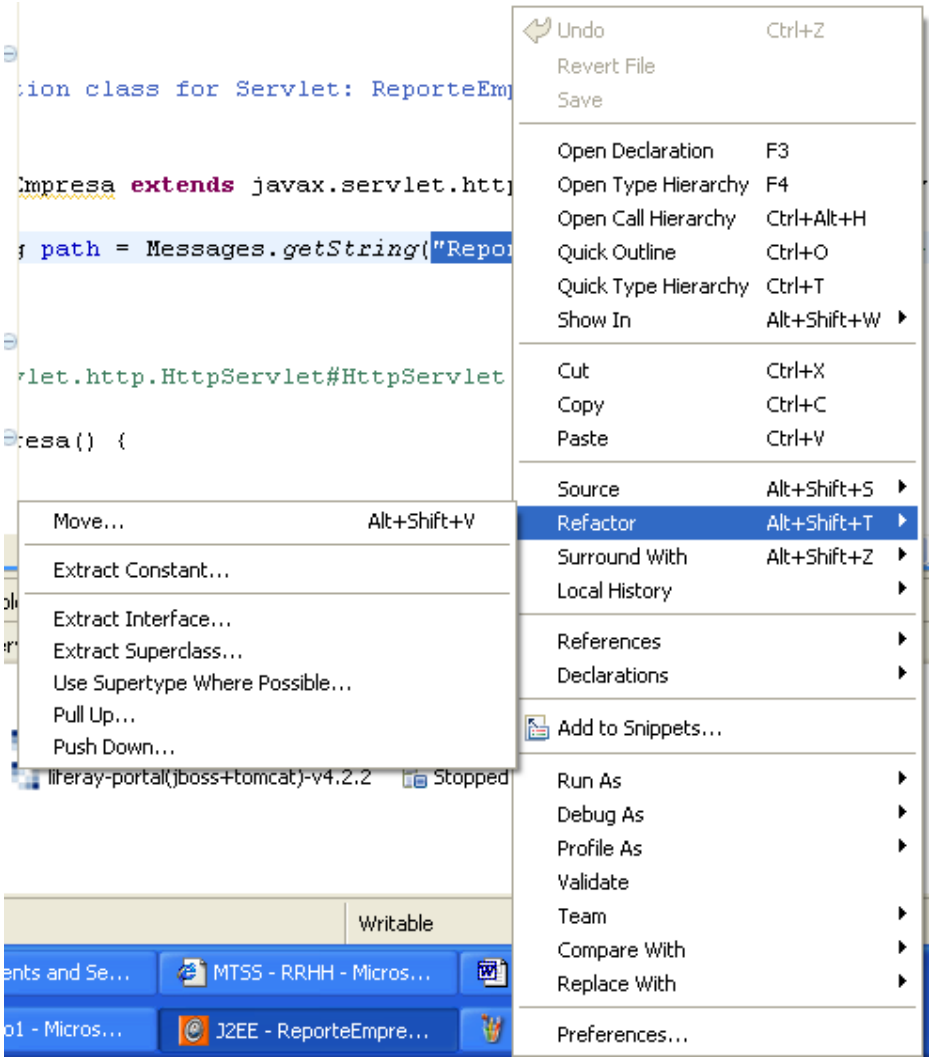


se selecciona el string que se quiere incluir y genera las clases:

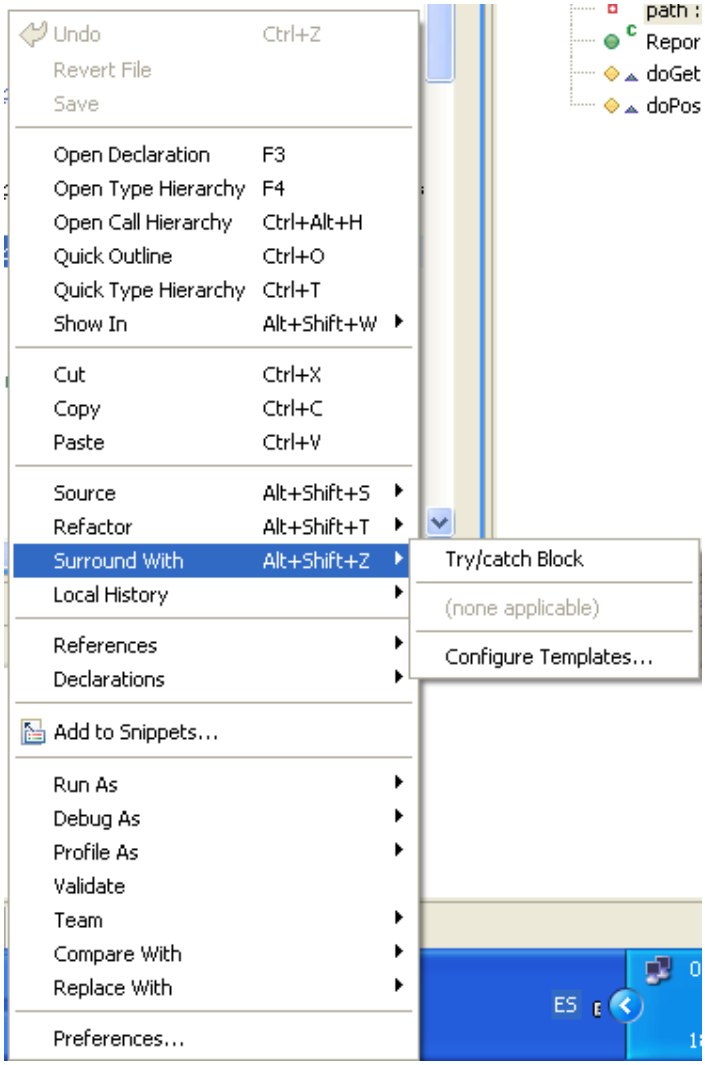
- Messages.java
- Messages.properties

para que sea más fácil de modificar los strings y dar soporte a la internacionalización.

- Refactor
  - extraer constantes



- surround with
    - try/catch



C.6.4 Otras herramientas

Otras herramientas que se utilizaron para el proyecto son:

- SSH Secure Shell: para transferir archivos al repositorio de facultad.
- Endnote: para registrar referencias.
- IReport: para implementar plantillas de impresión.
- BloodShed DevC++: para implementar algunas pruebas en lenguaje C.
- Microsoft Visio: para realizar algún diagrama adicional.
- Microsoft Word: para realizar la documentación.
- Microsoft PowerPoint: para realizar las presentaciones.
- SQL Express: para realizar pruebas con bases de datos
- SQL Server Management Studio Express CTP: para las pruebas con la base de datos SQL EXPRESS.
- Wink 2.0: para generar videos instructivos.

C.6.5 Referencias adicionales

Estas son algunas referencias adicionales sobre Java e IDEs.

1. *Why Java's Hot* http://www.ibiblio.org/java/books/jdr/chapters/01.html *Último acceso: 20/12/2009*.
2. *Programming Language Comparison* http://www.jvoegele.com/software/langcomp.html *Último acceso: 20/12/2009*.
3. *Free Java Gui Builder Download - The complete Java development setting.* http://shareme.com/showtop/freeware/java-gui-builder.html *Último acceso: 20/12/2009*.
4. *Graphical user interface builder - Wikipedia, the free encyclopedia* http://en.wikipedia.org/wiki/Graphical_user_interface_builder *Último acceso: 20/12/2009*.
5. *Introduction to GUI Building - NetBeans IDE 6.0 Tutorial* http://www.netbeans.org/kb/60/java/gui-functionality.html *Último acceso: 20/12/2009*.
6. *Nabble - Netbeans IDE Users - GUI builder - JPanel gets huge unexpectedly* http://www.nabble.com/GUI-builder---JPanel-gets-huge-unexpectedly-td16766297.html *Último acceso: 20/12/2009*.
7. *NetBeans Wiki NetBeansUserFAQ* http://wiki.netbeans.org/NetBeansUserFAQ *Último acceso: 20/12/2009*.
8. *Open Source UML & Modeling in Java* http://java-source.net/open-source/uml-modeling *Último acceso: 20/12/2009*.
9. *Eclipse, NetBeans, and IntelliJ Assessing the Survivors of the Java IDE Wars* http://www.devx.com/Java/Article/34009 *Último acceso: 20/12/2009*.
10. *Java IDE Review* http://www.mactech.com/articles/mactech/Vol.12/12.08/JavaIDEReview/index.html *Último acceso: 20/12/2009*.
11. *javaidecomparison [MojaveWiki]* http://www.mojavelinux.com/wiki/doku.php?id=javaidecomparison *Último acceso: 20/12/2009*.
12. *NetBeans and Eclipse Comparison Building Web Apps - No Contest [cld.blog-city.com]* http://cld.blog-city.com/netbeans_and_eclipse_comparison__web_apps__no_contest.htm *Último acceso: 20/12/2009*.
13. Glinert, E., *Visual programming environments: applications and issues*. 1990: IEEE Computer Society Press. 686.
14. Glinert, E., *Visual programming environments: paradigms and systems*. 1990: IEEE Computer Society Press. 661.
15. Hák, R., *Editors Comparison (NetBeans IDE, Eclipse, IntelliJ IDEA)*. 2008. p. 11.
16. Quatrani, T., *Visual modelling with rational rose and UML*. 1998: Addison Wesley. 222.
17. *Reingeniería con Netbeans* http://cnx.org/content/m17590/latest/ *Último acceso: 20/12/2009*.

C.7 Estándar de Implementación y Documentación

Estas son algunas convenciones para la documentación técnica:

1. Definir qué idea o concepto global se trata de abarcar con el documento.

   Investigar:
   - La herramienta.
   - Otros manuales o materiales.
   - Experiencias previas.
   - Manuales o herramientas parecidas en el área.

2. Elaborar un índice enumerando las unidades y posibles secciones del manual.

   I. Se pueden incluir las siguientes unidades:
   - Introducción.
   - Presentación.
   - Unidades de desarrollo
   - Resumen.
   - Respuestas a las autoevaluaciones y ejercicios.
   - Referencias.
   - Anexos.
   - Glosario.
   - Índice Alfabético.

   II. Se pueden incluir las siguientes secciones en cada unidad:
   - Introducción
   - Desarrollo.
   - Ejercicios.
   - Auto-evaluación.
   - Resumen.
   - Referencias particulares de la unidad.

3. Definir una estructura más detallada y hacer las correcciones del índice basado en el feedback.

4. Definir un vocabulario común.

5. Establecer gestión:
   - Un plan de trabajo (en tiempo y personal).
   - Alcance.
   - Estimaciones: en tiempo y personal.
   - Políticas de reporte y corrección de errores: con comentarios, correos, listas, documentos aparte, etc.
   - Políticas de consulta: con los autorizadores.
   - Versionado: hacer un zip por día (más si se hacen cambios muy grandes), usar CVS, respaldo de versiones parciales importantes.
   - Registro de cambios: todos los días enumerar los cambios más grandes, correcciones, nuevas adiciones, etc.

6. Elaborar pautas de diseño:

   Formato del documento que se genera: Word, pdf, texto plano, etc. Si es un documento maestro y unidades aparte o si son varios documentos chicos y uno que referencia a los demás.

   En qué herramienta se desea hacerlo.

   Diseño de página:
   - Tamaño de página.
   - Márgenes.
   - Encabezado y pie de página.
   - Números de página.
   - Carátula.
   - Diseño para encuadernación doble faz o simple.

   Utilización de:
   - Links a lugares del documento o a otros documentos.
   - Referencias cruzadas.
   - Comentarios.

   Diseño de títulos:
   - Tipo de fuente
   - Color
   - numeración.

   Diseño de etiquetas:
   - definiciones.
   - listas.
   - tablas.
   - procedimientos.
   - código.
   - Ilustraciones.
     a. figuras.
     b. diagramas.
   - Casos:
     a. ejemplos.
     b. ejercicios.
     c. preguntas.
   - Recuadros:
     a. Nota.
     b. Importante.
     c. Advertencia.
     d. Consejo.
     e. Truco.
     f. Más información.
     g. Información relacionada.
     h. Referencia
     i. Pie de página.

NOTA: Una nota indica un concepto que puede ser de interés para el usuario pero que no es algo esencial que debe ser recordado para utilizar el sistema.

IMPORTANTE: Una nota importante indica un concepto que el usuario debe recordar cuando use el sistema y que puede aclarar las ideas presentadas en la sección.

ADVERTENCIA: Una advertencia le indica al usuario que un procedimiento puede involucrar algún riesgo como pérdida de datos, bloqueo del sistema o incluso daño del hardware. Estos cuadros indican al usuario cómo prevenir dichas situaciones o cómo recuperarse una vez que ocurrieron.

CONSEJO: Un consejo o buena práctica indica al usuario como tomar ventaja del sistema en un aspecto particular. Puede mostrar como mejorar el desempeño del sistema o hacer una tarea más fácil o rápido. Los consejos pueden ayudar al lector a recordar más fácilmente un procedimiento largo o incorporar una buena práctica.

TRUCO: Un truco puede ayudar al usuario a obtener los resultados deseados sin necesariamente seguir todos los pasos indicados por la sección.

MÁS INFORMACIÓN: Estas cajas indican referencias o enlaces a otros recursos de información donde el lector puede profundizar sobre el tema de la sección.

INFORMACIÓN RELACIONADA: Estas cajas indican referencias o enlaces donde el lector puede encontrar información relacionada al tema de la sección que puede ser de interés, pero no necesariamente sobre el mismo tema.

Definir:

- Tipo de fuente que se usará.
- Iconos especiales.
- Color del recuadro
- Tipo de numeración.
- Ubicación de las etiquetas respecto al elemento.
- Ubicación del elemento en la página.

7. Formato de las unidades:

I. Se pueden incluir en las siguientes unidades:

- Introducción
  a. presentación del manual
  b. organización del manual
  c. audiencia objetivo
  d. convenciones
  e. temas no tratados
  f. documentos relacionados
- Presentación.
- Unidades de desarrollo
- Resumen.
  a. Metodologías.
  b. Buenas prácticas.
  c. Sugerencias.

- Respuestas a las auto-evaluaciones y ejercicios.
- Referencias
    a. chequear todas las referencias
    b. poner últimas fechas de acceso a páginas Web
    c. versiones del documento referenciado
    d. chequear autores.
- Anexos: incluyen todo lo que es demasiado largo para incluir en las unidades. Contenido complementario, para profundizar.
- Glosario.
- Índice Alfabético.

II. Se pueden incluir las siguientes secciones en cada unidad:
- Introducción: enumerar lo que se va a desarrollar en la unidad.
- Desarrollo.
- Ejercicios: ejercicios de baja, mediana y gran complejidad que abarquen lo descrito en la unidad.
- Auto-evaluación: preguntas de múltiple opción o simples que testean la comprensión general de la unidad.
- Resumen: enumerar lo presentado y lo que se dará en la próxima unidad.
- Referencias particulares de la unidad.

C.7.1 Normas de utilidad

Las normas de utilidad tienen el propósito de asegurar que una documentación satisfaga las necesidades prácticas de información de las audiencias [1-26].

Identificación de la audiencia: las audiencias que participan o son afectadas por la documentación deben identificarse, de manera que se puedan satisfacer sus necesidades.

Se debe identificar y ordenar las audiencias de acuerdo a las limitaciones de tiempo y recursos. Evita desviarse del propósito del documento. El criterio apropiado para clasificar las audiencias incluye el interés potencial expresado por cada audiencia, así como sus propuestas para usar la documentación y tomar decisiones o influir en ellas.

Lineamientos:
- identifique a los líderes desde el principio de la documentación, pues éstos pueden ayudar al documentador a elegir otras audiencias.
- Establezca contacto con los representantes de las audiencias y conozca la importancia que éstos otorgan a la documentación, el uso que harán de los resultados y la información de sus intereses particulares.
- Llegue a un acuerdo con el cliente respecto de la importancia de las audiencias potenciales y de la información que éste desea obtener; asimismo, planee y utilice los datos recopilados e informes de cada actividad.
- Manténgase alerta con respecto a audiencias adicionales consideradas a lo largo de la documentación y, entre las limitaciones de tiempo y recursos, mantenga la flexibilidad y capacidad para responder a sus necesidades.

Errores:
- Generalizar excesivamente la documentación por tratar de abarcar en un solo informe las diversas necesidades de audiencias muy heterogéneas.

Advertencias:
- Al ajustar los informes complementarios a las audiencias específicas, debe tener cuidado en no distorsionar los resultados de la documentación.

- Después de identificar las audiencias potenciales, ordene las necesidades de información, dentro de las limitaciones de tiempo y recursos.
- Después de convenir con las audiencias potenciales, por el contrario ayude a que se desarrollen expectativas reales, en las que deberá tener en cuenta las limitaciones metodológicas, financieras y políticas en la documentación.

Confiabilidad del documentador: las personas que dirigen la documentación deben ser confiables y competentes al realizar la documentación, de modo que sus resultados alcancen la credibilidad y aceptación máximas.

Se debe tener un grupo de documentadores capacitado, con la habilidad técnica, conocimiento sustancial, experiencia, integridad, habilidad y de relación pública y otras características consideradas necesarias por el cliente y usuarios de la documentación. Si no se cumple con la capacitación el resultado es que la documentación sea inútil, rechazada y que no fundamente la adecuación técnica, integridad, utilidad y sentido práctico de la misma.

Lineamientos:

- Manténgase al tanto de las fuerzas sociales y políticas respecto de la documentación, especialmente de aquellas ligadas a los grupos interesados.
- Asegúrese de que tanto la planeación como la información del grupo respondan a las preocupaciones de las principales audiencias.
- Someta a revisión el plan de documentación y encárguese de que otros documentadores con títulos aceptados por el cliente sometan el trabajo a una auditoria.
- Considere algunos procedimientos alternativos en el proceso de planeación de la documentación.
- Sea claro al describir el plan de documentación a las diversas audiencias, y demuestre que el plan es realista y técnicamente correcto.
- Mantenga informadas a las audiencias en cuanto al progreso de la documentación por medio de cartas, informes, memorandos, folletos y juntas.

Errores:

- No lograr persuadir que el documentador es confiable tanto en los contenidos como en la metodología de la documentación.
- Llevar a cabo una documentación que otorgue prioridad a los valores de las audiencias particulares y, por lo tanto, que ignore o minimice los valores de otras.
- No autoeliminarse del proceso evaluativo por falta de aptitudes, experiencia u objetividad necesaria para efectuar la documentación.
- No asegurar que los miembros del grupo de documentación dediquen su tiempo y esfuerzo, y no sólo su buena imagen, a la documentación.

Advertencias:

- Comprender que las fuentes se agotan en el esfuerzo para alcanzar confiabilidad y aceptación, de manera que el énfasis exagerado de estos factores tiene como resultado una baja en la relación costo-producto (eficiencia) de la documentación.

Selección y alcance de la información: la información recopilada debe ser de tal alcance y selección que dirija las preguntas pertinentes al objeto de documentación y responda a las necesidades y los intereses de las audiencias específicas.

Se debe lograr plantear los diversos elementos de mayor importancia para las principales audiencias y responder a ellos con información clara y útil. Se deben abarcar todas las variables importantes para cada elemento (efectividad, efectos secundarios negativos, costos, etc.), aunque las audiencias no lo especifiquen. Se debe considerar lo que es significativo para el cliente, ya que no se puede abarcar todo, pero también identificar lo que el cliente ha pasado por alto. Se debe revisar la bibliografía para poder establecer juicios que permitan identificar y jerarquizar lo que se debe documentar. La continua comunicación con el cliente establecerá juicios cada vez más específicos para cada tema. Se debe tratar de responder al usuario, más allá de las preferencias del documentador.

Lineamientos:

- Dialogue con los representantes de las audiencias principales para comprender los diversos puntos de vista y su posible problemática, y para conocer las necesidades de información de las diversas audiencias.
- Confirme que las preguntas formuladas a los entrevistados se relacionen con los propósitos de la documentación.
- Evite que las audiencias crean que se responderán todas sus preguntas.
- Sea lo suficientemente flexible para permitir que se añadan las preguntas que surjan durante la documentación.
- Permita al cliente jerarquizar en orden de importancia las audiencias potenciales y trabajar con los representantes de cada audiencia en la jerarquización de los temas según la importancia que tendrán para cada audiencia.
- Trabaje con el cliente para cotejar el orden de los temas de cada audiencia, elimine los temas al final de la lista y añada aquellos que el documentador considere importantes aunque no se hayan propuesto.
- Distribuya el esfuerzo total para la documentación y asigne en la lista final de temas, el mayor esfuerzo a los temas jerárquicamente mayores.

Errores:

- No considerar el complejo nivel técnico de las audiencias al decidir qué información debe recopilarse y cómo se debe analizar e informar.
- No actualizar las necesidades de información mediante el contacto periódico con el cliente.
- Recopilar información sólo porque la considera conveniente, en vez de hacerlo porque es necesaria.
- No tener en cuenta la dinámica entre comprensión y selección en cada etapa de la documentación: desarrollo del plan, establecimiento del presupuesto, selección de instrumental, y acopio, análisis, interpretación y publicación de la información.

Advertencias:

- Reconocer que la comprensión se opone a la selectividad y tener en cuenta que la forma de comprometerse es mediante la selección de la información que tanto el documentador como el cliente consideren más importante para las audiencias más significativas.
- Debe dirigirse, en primera instancia, al propósito fundamental de la documentación y llevar a cabo otras actividades de interés – como la investigación de la documentación, la investigación de un sujeto dado y la capacitación en la documentación-, sólo si los recursos lo permiten y si tales actividades auxiliares tienen a contribuir al mejoramiento de a comprensión y/o desarrollo de la educación y/o documentación.

Interpretación valorativa: las perspectivas, los procedimientos y la fundamentación que se utilicen al interpretar los resultados deben describirse con cuidado, de manera que los fundamentos de los juicios de valor sean claros.

La valoración es la escala de utilidad de un objeto, su importancia o valor global. Es complejo determinar quien hace los juicios de valor. El objetivo es que los documentadores y sus clientes puedan determinar de manera reflexiva cuál será la aproximación adecuada para valorar la información recopilada, y éstos deben revelar y justificar la aproximación que elija.

Lineamientos:

- Considere las bases alternativas para interpretar los resultados; por ejemplo: objetivos del proyecto, especificación de los procedimientos, leyes y reglamentos, metas institucionales, desempeño de los grupos por comparación, medición de las necesidades de un grupo consumidor, expectativas de desempeño de los grupos por comparación, medición de las necesidades de un grupo consumidor, expectativas de desempeño en un grupo muestra e informe de juicios por medio de los diversos grupos referenciales.
- Considere quién hará la interpretación valorativa; por ejemplo: los documentadores el cliente, las diversas audiencias, algún grupo regulador específicamente encargado de estas tareas, o alguna combinación de éstos.
- Considere las técnicas alternativas que se podrían utilizar para medir el valor significativo de la información recopilada; por ejemplo, cuente con grupos paralelos de expertos que elaboren informes por escrito, responsabilice de manera directa al cliente y a otras audiencias para que elaboren las

interpretaciones del valor; dirija un juicio o peritaje administrativo sobre el objeto de investigación; busque las convergencias existentes por medio de un estudio Delphi, o responsabilice al documentador directamente para obtener un juicio de mérito o valor.

Errores:

- Suponer que las documentaciones deben ser objetivas en el sentido de estar exentas de juicios de valor.
- No determinar las perspectivas de valor que el cliente y las audiencias consideran importantes al interpretar los resultados de la documentación.
- Diseñar la recopilación de datos y procedimientos de análisis, sin considerar qué criterio será necesario para interpretar las respuestas, por ejemplo, el desempeño de un grupo comparativo.

Advertencias:

- No se debe destinar demasiado tiempo a los problemas relacionados con la determinación del valor significativo de la información recopilada, pues se reducirán el tiempo y el esfuerzo necesarios para recopilar, promover y apoyar tales juicios de valor.
- Si se vislumbra que las variables adicionales no previstas en el plan de documentación original son pertinentes, el esfuerzo destinado a juzgar el mérito y valor de un objeto no se deberá limitar a aquellas variables proyectadas al inicio del plan de documentación.
- Se debe reconocer que las reglas para tomar decisiones suelen ser arbitrarias y, por ello, deben ser sujetas a debate.

Claridad del informe: el informe de la documentación debe describir el objeto por evaluar y su contexto, los propósitos, procedimientos y resultados de la documentación, de manera que las audiencias puedan entender con rapidez lo que se ha hecho, por qué se ha realzados, qué información se obtuvo, a qué conclusiones se llegó y qué recomendaciones se hicieron.

Lineamientos:

- Cuando sea apropiado, trate de completar el informe formalmente escrito, con otras técnicas de comunicación, como diagramas, etc.
- Comuníquese regularmente con los participantes de mayor importancia y con las audiencias.
- Dirija los informes lo más directamente posible a las cuestiones evaluadas.
- Aporte información contextual suficiente que dé sentido a la documentación global, de tal forma que se proporcionen fundamentos firmes para las conclusiones y recomendaciones.
- Defina todos los términos técnicos, junto con el contexto y enfoque de la documentación, y considere la complejidad técnica y estadística de que puede constar la audiencia.
- Los enunciados sumarios deben complementarse con una explicación de los problemas, objetivos y preguntas.
- Escriba de forma simple y directa, pero no simplista.
- Sírvase de ejemplos para ayudar a la audiencia a relacionar los resultados con circunstancias prácticas.
- Invite a los representantes de las audiencias a revisar el informe o informes.
- Ayude a las audiencias a comprender los términos técnicos utilizados en los informes y anexe un glosario, trace informes sumarios y técnicos y/o proporcione a las audiencias oportunidades de capacitación.

Errores:

- No adecuar el informe de acuerdo con la habilidad que tenga la audiencia para utilizar el lenguaje técnico.

Advertencias:

- Informe las diversas percepciones del objeto por evaluar, aún cuando surjan ambigüedades, sin embargo, debe señalar dichas ambigüedades.
- No permita que en el esfuerzo por alcanzar la claridad, se distorsionen los resultados.

Difusión del informe: Los resultados de la documentación se deben difundir a clientes y otras audiencias con derecho a la información para que éstos puedan ponderarlos y utilizarlos.

Una audiencia con derecho a la información es aquella que se nombra oficialmente con derecho a saber los resultados de la documentación por las razones siguientes:

- Ser el cliente.
- Ser legalmente responsable del objeto por evaluar.
- Ser quien financia el objeto de la documentación.
- Ser quienes proporcionan una cantidad sustancial de datos para la documentación.
- Ser determinantes en otro sentido, por ejemplo, ser quien diseña el programa por evaluar, ser el afectado en la carrera o status profesional, etc.

Se debe trabajar en colaboración con el cliente para reportar la documentación a todas las audiencias con derecho a la información para que puedan utilizarla o apoyarla.

Lineamientos:

- Defina, desde el principio y con ayuda del cliente, las audiencias con derecho a la información.
- Informe al cliente, durante la negociación inicial, que todas las audiencias con derecho a la información deben tener acceso a los resultados y observaciones, por lo menos de manera resumida.
- Informe al cliente, durante la negociación inicial, que se debe proporcionar a todas las audiencias con derecho a la información el acceso directo al informe total.
- Cuando sea factible, encárguese de que participen personas ajenas a los clientes y de que los documentadores cuiden que los informes sean de calidad y comuniquen sus resultados a las audiencias con derecho a la información.
- Los representantes de las audiencias más importantes deben proponer qué resultados y observaciones y que formato de informe les interesaría y les sería útil.
- Llegue a un acuerdo con los clientes al concluir el plan de difusión, especifique las audiencias y detalle los contenidos, formatos y fechas de informes para audiencias con derecho a la información, incluidos los medios masivos de comunicación.
- Al concluir, llegue a un acuerdo con el cliente con respecto al control editorial acerca de quién expondrá los informes intermedios y finales; sin embargo, deberá reservarse el derecho a exponer la información sobre las prácticas ilegales empleadas para obtener dicha información.
- Rectifique con los representantes de las audiencias con derecho a la información si los esbozos del informe son apropiados y claros y si los datos son precisos.
- Elabore un resumen del informe total y, si éste es más extenso de lo contemplado, resúmalo, aún cuando hacerlo no forme parte del acuerdo inicial.
- Al planear la difusión de los resultados, deberá considerar los diversos métodos.
- Reúnase con quienes elaborarán los resúmenes de los resultados para asegurarse de que hayan comprendido los resultados. Ofrezca la revisión de sus resúmenes por escrito o telefónicamente, antes de su publicación.
- Elabore los informes en un formato que permita al cliente reproducirlos fácil, rápida y económicamente.

Errores:

- Dirigir el informe al cliente o financiador, omitiendo a las demás audiencias con derecho a la información.

Advertencias:

- Debe tener en cuenta que, con frecuencia, se mejorará el desempeño de la gente si los procedimientos y materiales de los que son personalmente responsables se critican de forma privada.
- Realice los pasos necesarios para asegurar que la información sobre prácticas ilegales descubiertas durante la documentación se proporcione, aún bajo el riesgo de exponer los acuerdos contractuales existentes.

Oportunidad del informe: la publicación de los informes se debe hacer a tiempo para que las audiencias aprovechen mejor los resultados. La demora hace que se pierda tiempo útil de los materiales.

Lineamientos:

- Al elaborar la documentación, investigue qué audiencias planean tomar decisiones para las que solicitan información y averigüe en que momento la requieren.

- Empiece por planear retrospectivamente, señale la fecha de entrega de los informes y trace proyectos realistas que incluyan los pasos por seguir para cumplir con la fecha límite, mantenga un margen de tiempo que le permita examinar los problemas no contemplados en los requisitos de los informes intermedios.
- Los documentadores con experiencia deben revisar el calendario programado y seguir haciéndolo sucesivamente.
- En caso de que las actividades de la documentación se retrasen y no se puedan ajustar al calendario, identifique los pasos alternativos por seguir.
- Dentro de los límites posibles y a lo largo de la documentación, debe mantenerse alerta y responder a los cambios de horario de las audiencias.
- A lo largo del trabajo, recuerde cuáles son las necesidades de información más importantes, busque satisfacerlas, aún cuando excluya las de menor importancia.

Errores:
- Comprometer fechas de entrega de resultados, sin elaborar un cronograma detallado y sin considerar si los recursos son adecuados para la documentación.
- Sin hacer un examen cuidadoso, suponer que los demás individuos o grupos pueden cumplir con el trabajo que les corresponde dentro de los límites de tiempo estipulados.
- No verificar el progreso durante las etapas intermedias, ni informar de os retrasos al cliente que, a su vez, pueden ocasionar el retraso del informe.

Advertencias:
- Considere cambios que garanticen que los informes se entreguen en el tiempo estipulados; por ejemplo, utilice una muestra de población más pequeña, reduzca las preguntas de las entrevistas, omita las partes secundarias de la documentación o abrevie el cuerpo principal del informe; sin embargo, la precisión técnica no se debe sacrificar por mantener la fecha límite de entrega.

Trascendencia de la documentación: las documentaciones se deben planear y dirigir para estimular su cumplimiento por los miembros de las audiencias. La trascendencia de una documentación se refiere a la influencia que ésta tiene en las decisiones y actos de los miembros de las audiencias. El mejoramiento no es automático sino que debe guiarse y estimularse.

Lineamientos:
- Al comienzo de la documentación, muestre a las audiencias principales de qué manera serian útiles los resultados de la documentación a su trabajo.
- Encárguese de que los representantes de otras audiencias participen en la determinación de las preguntas y en la planeación de la puesta en marcha de los procedimientos de la documentación.
- Sea abierto, franco y concreto al informar a as audiencias y muéstrese dispuesto y deseoso de ayudar a esclarecer los informes.
- Elabore periódicamente informes de los resultados interinos, en los cuales destaque cómo se pueden aplicar al papel que desempeñan los miembros de las audiencias.
- Trace los méritos de los posibles cursos de acción alternativos por tomar y someta a análisis aquellos que aparezcan en el informe final.
- Complemente los informes escritos con una comunicación oral constante.
- Dentro de los límites de tiempo y recursos, planee ayudar a las audiencias a medir, interpretar y aplicar los resultados de la documentación que excedan el tiempo de entrega del informe final.

Errores:
- Perder el interés en la documentación en cuanto se entrega el informe final.
- Exhibir su falta de confianza en las audiencias, en cuanto al uso práctico de los resultados; por ejemplo, mencionar públicamente que las audiencias sólo respaldarán aquellas secciones de la documentación que refuercen sus creencias y prácticas comunes.
- Preocuparse del valor teórico de los resultados a expensas de otorgar importancia a su aplicación.
- No considerar los valores de las audiencias al hacer las recomendaciones.

Advertencias:
- No intente asumir la responsabilidad de los clientes de modo que actúe sobre los resultados de la documentación.

- Considere que gran parte de los documentadores no tienen la capacitación necesaria para ser agentes del cambio y que requieren una ayuda especializada para ser competentes en ese papel.
- Considere que las acciones de las audiencias se suponen basadas en los resultados de la documentación. Por ello, infórmelos de posibles aplicaciones incorrectas de los resultados.
- Evite dejarse influir por audiencias con poder que apoyen recomendaciones no fundamentadas en los resultados.

C.7.2 Diagramas

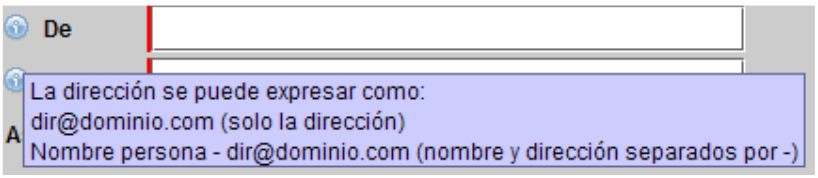
Estas son algunas convenciones específicas para los diagramas de la documentación:

- Los diagramas se deben realizar por medio de una herramienta que permita extraerlos directamente del código. De esta manera los diagramas pueden mantenerse fácilmente actualizados.
- La herramienta seleccionada es NetBeans como se especifica en la sección "Análisis de lenguajes e IDEs".

C.7.3 Convenciones Generales

Para la implementación del código se debe cumplir las siguientes pautas:

- Para la implementación se utilizará la herramienta Eclipse dada la familiaridad del programador y las utilidades provistas por la misma como se especifica en la sección "Análisis de lenguajes e IDEs".
- Abstraer los conceptos para así realizar una implementación que sea parametrizable y fácil de modificar y reprogramar.
- Utilizar archivos de mensajes para configurar todos los strings de la aplicación y proveer de una forma fácil de internacionalización y mantenimiento en caso de cambios.
- Las funciones deben ser lo más generales posibles (parametrizables) para así convertirlas en utilidades que pueden ser llamadas para varias tareas.
- Las funciones deben ser lo más breves posibles. De esta manera se facilita su visualización, revisión, verificación, generalización y reutilización.
- Realizar revisiones de código periódicas para mantener la visión general del proyecto, mejorar la implementación y analizar nuevas metas a establecer o proponer para trabajo futuro.
- Cada vez que el usuario realice una revisión de código se deberá analizar:
  - Ver qué partes del código son fijas (harcodeadas) y ver cómo parametrizarlas.
  - Cómo parametrizar una función o porción de código (por más pequeña que sea) para que sea generalizable a varios casos.
  - Ver qué partes de código o funciones se pueden convertir en una utilidad para poder reutilizarlas.
- Establecer una única forma en todo el sistema de:
  - Mostrar una operación que termina en éxito.
  - Mostrar una operación que termina en advertencia.
  - Mostrar una operación que termina en error.
  - Mostrar un error del sistema.
- Reducir los mensajes que salen por consola y siempre mostrarlos por medio de una ventana.
- Proveer la mayor guía o ayuda posible al usuario por medio de mensajes claros al ejecutar una acción o al desplegar un elemento en pantalla (botones, campos, etc.). Por ejemplo:



- Trabajar con rutas utilizando el separador del sistema para así mantener la portabilidad a diferentes sistemas operativos. Windows utiliza como separador el carácter '\', mientras que Linux usa '/', por lo que este aspecto se debe tomar en cuenta en la implementación.
- Reducir el número de warnings en el código lo más posible para evitar problemas a futuro (agregar tipos a las colecciones de datos, etc.).
- Proveer de mensajes de advertencia antes de cerrar una ventana por si se desea guardar las modificaciones. Confirmar cualquier acción que no se pueda deshacer antes de hacerla.
- Indicar claramente cuáles son los campos obligatorios en cada formulario para realizar una acción.
- Realizar respaldos de código cada vez que:
    - se realice una modificación importante
    - se cierre una funcionalidad (se haya implementado y verificado)
    - se termine con una jornada de trabajo

  y mantener un log con lo que contenga dicha versión.

C.7.4 Convenciones Específicas

Estas son algunas convenciones específicas para implementar ciertos elementos:

C.7.4.1 Subsistemas

Estas son algunas convenciones específicas para implementar subsistemas (packages):

- Los subsistemas se deben nombrar completamente en minúsculas. Deben estar contenidos dentro de los packages: uy.edu.fing.lab.
- Los subsistemas deben agrupar clases referentes a funcionalidades similares del sistema. Se pueden hacer tantos subpackages como sea necesario.
-
- Agruparlas según:
    - Principales: package para la aplicación principal del sistema y los componentes globales (archivo de mensajes, etc.).
    - Entrada/salida: package para las clases que se encargan de la entrada y salida y la configuración general del proyecto. Se puede hacer un subpackage para cada grupo de funcionalidades:
        - Lectura / escritura
        - Paleta
        - Modelo del proyecto
    - Recursos: package de recursos estáticos del sistema. Se puede hacer un subpackage para cada grupo de recursos:
        - Imágenes
        - Archivos de ayuda
        - Plantillas de impresión
        - Programas externos
    - Utilidades: package para utilidades del sistema. Se puede hacer un subpackage para cada grupo de utilidades.
    - Visualización: package para funcionalidades del sistema (agrupado parecido al menú). Se puede hacer un subpackage para cada grupo de funcionalidades.
        - Algoritmos
        - Ejecuciones
        - Gráfico
        - Preferencias
        - resultados

C.7.4.2 Componentes:

Estas son algunas convenciones específicas para implementar componentes (clases):

- Los componentes se deben nombrar siempre empezando con mayúscula y cada palabra que se concatena al nombre debe empezar con mayúscula. Por ejemplo: LabUtilsBD.java

- Se puede usar '_' en caso de ser necesario para diferenciar clases similares. Por ejemplo: LabUtilsIO_BD.java, LabUtilsIO_TXT.java, etc.

C.7.4.3 Interfaces:

Estas son algunas convenciones específicas para implementar interfaces:

- Las interfaces se deben nombrar siempre empezando con mayúscula, cada palabra que se concatena al nombre debe empezar con mayúscula y terminar en Interface. Por ejemplo: DialogoInterface.java

C.7.4.4 Variables:

Estas son algunas convenciones específicas para implementar variables:

- Las variables deben definirse al principio de una función o clase.
- Solo se deben hacer públicas en una clase si es imprescindible que otras clases las accedan y si son muchas como para implementar las funciones de get y set de cada una.
- En caso de ser públicas en una clase incluir comentarios en formato javadoc [27]. Por ejemplo:

```
/**
 * Vector con todos los menúes
 */
public Vector<JMenu> todosMenu = new Vector<JMenu>();
```

C.7.4.5 Constantes:

Estas son algunas convenciones específicas para implementar constantes:

- Convertir en constantes aquellas variables cuyo valor no cambia. Así se mantiene su valor en una sola clase y no es necesario redefinirlas cada vez. Si su valor cambia, solo hay que modificar el valor en un solo lado.
- Las constantes deben tener nombres lo más explicativos posible. Deben definirse siempre con todas las letras en mayúsculas y cada vez que se concatena una palabra, se debe hacer con '_'. Por ejemplo: *FILE_CONFIG_PRIVADO_PREF*
- Deben incluir comentarios en formato javadoc [27]. Por ejemplo:

```
/**
 * Nombre del archivo de configuración del directorio privado confPreferencias
 */
public static String FILE_CONFIG_PRIVADO_PREF = "confPreferencias";
```

C.7.4.6 Funciones:

Estas son algunas convenciones específicas para implementar funciones:

- Las funciones deben tener nombres lo más explicativos posible.
- Deben comenzar con minúscula y cada vez que se concatena una palabra, ésta debe comenzar con mayúscula. Por ejemplo: leerMatrizBD
- Se puede usar '_' en caso de ser necesario para diferenciar funciones similares. Por ejemplo: leerMatrizBD_SQL
- Mantener una correcta indentación del código para poder hacer un seguimiento del mismo. Los caracteres de comienzo de sección deben estar al final de la sentencia que inicia el flujo y el carácter de fin de sección debe estar alineado con el primer carácter de la sentencia que inicia el flujo. Por ejemplo:

```
<sentencia que inicia el flujo>{

}
```

- Se deben comentar las áreas más importantes o complejas dentro de una función ya sea con comentarios del estilo '/**<comentario> */' o '// <comentario>'. Todas las funciones deben

estar documentadas en formato javadoc [27] (sobre el encabezado) con sus parámetros, retornos y excepciones. Las explicaciones deben ser lo más claras posibles y con ejemplos si aplica. Por ejemplo:

```
        /**
         * Obtiene una matriz con los datos de una tabla de base de datos
         * @param archivoJar                dirección del driver (.jar)
         * @param claseDriver               nombre del driver (Ejemplo:
com.microsoft.sqlserver.jdbc.SQLServerDriver)
         * @param usuario                           usuario de la base de datos
         * @param password                          contraseña de la base de datos
         * @param nombreBD                          nombre de la base de datos
         * @param url                               url de conexión
         * @param nombreTabla                       nombre de la tabla de la base de datos
         * @param columnas                          arreglo con los nombres de las columnas a leer
         * @param order                                     nombre de la columna por la cual ordenar los
datos
         * @param ascendente                        true si los datos deben ser devueltos en orden
ascendente
         * @param maximo                            valor máximo que debe tener la tupla
         * @return                                          matriz con los datos de una tabla según lo
especificado
         * @throws Exception
         */
        public static String[][] leerMatrizBD(String archivoJar, String claseDriver, String usuario,
                        String password, String nombreBD, String url, String nombreTabla, String[] columnas,
                        String order, boolean ascendente, String maximo) throws Exception{
```

C.7.5 Referencias adicionales

Estas son algunas referencias adicionales sobre estándares de implementación y documentación.

1. *Reglas gramaticales* http://www.indiana.edu/~call/reglas.html *Último acceso: 20/12/2009*.
2. *RAE* http://www.rae.es/ *Último acceso: 20/12/2009*.
3. *La Página del Idioma Español* http://www.elcastellano.org/ *Último acceso: 20/12/2009*.
4. *Horrores Ortográficos Más Comunes » Alquimistas del Diseño - Cuestionar, transmutar y diseñar* http://alquimistas.evilnolo.com/2005/11/13/horrores-ortograficos-mas-comunes/ *Último acceso: 20/12/2009*.
5. Valduriez, P., *Algunas ideas para mejorar la escritura de informes técnicos*. 1994.
6. *Cómo Elaborar una Tesis de Grado - Apuntes de Comunicación y Periodismo* http://www.elprisma.com/apuntes/comunicacion_y_periodismo/comoelaborarunatesisdegrado/ *Último acceso: 20/12/2009*.
7. *Cómo elaborar y asesorar una investigación de tesis - Monografias.com* http://www.monografias.com/trabajos3/comotesis/comotesis.shtml *Último acceso: 20/12/2009*.
8. *Cómo escribir una tésis de grado - Monografias.com* http://www.monografias.com/trabajos/tesisgrado/tesisgrado.shtml *Último acceso: 20/12/2009*.
9. *Dissertation-Thesis Guide* http://www.learnerassociates.net/dissthes/ *Último acceso: 20/12/2009*.
10. *ESTRUCTURA DEL PROYECTO DE GRADO guia-de-estructura-del-proyecto-de-grado.pdf*. p. 6.
11. Grundy, J.C., *Experiences with Facilitating Student Learning in a Group Information Systems Project Course*. 1996. p. 8.
12. Ingeniería, F.d. and U.d.l.R.d. Uruguay, *GUÍA DE INFORME DE PROYECTO DE GRADO*. p. 2.
13. EAFIT, U., *GUÍA PARA LA PRESENTACIÓN DE PROYECTOS Y TESIS DE GRADO*. 2007. p. 25.
14. PEDECIBA, F.d. Ingeniería, and U.d.l.R.d. Uruguay, *Guía para la presentación de publicaciones del Instituto de Computación y del Area Informática del PEDECIBA*. p. 10.
15. Caivano, J.L., *GUÍA PARA REALIZAR, ESCRIBIR Y PUBLICAR TRABAJOS DE INVESTIGACIÓN*. 1995.
16. Melero, R., *How to start to write a scientific paper*.

17. Chandrasekhar, R., *How to Write a Thesis: A Working Guide*. 2000. p. 33.
18. *IUB Writing Tutorial Services Pamphlets* http://www.indiana.edu/~wts/pamphlets.shtml *Último acceso: 20/12/2009*.
19. *LearnerAsociates.net LEARNSITE* http://www.learnerassociates.net/ *Último acceso: 20/12/2009*.
20. Sharkey, E. and J. Paynter, *Lessons to be learnt from students software estimation exercises*. p. 8.
21. *Manual para la elaboración de Tesis Doctorales, Trabajos de Grado y Trabajos Especiales*. 2007. p. 82.
22. Thomas, R., *A Practical Experiment in Teaching Software Engineering Metrics*. 1996. p. 7.
23. *Recomendaciones para la preparación de propuestas de tesis*.
24. Wilson, J.R., *Some guidelines on technical writing*. 1999. p. 8.
25. Ingeniería, F.d. and U.d.B. As., *75.00 TESIS DE GRADO EN INGENIERÍA INFORMATICA* http://materias.fi.uba.ar/7500/ *Último acceso: 20/12/2009*.
26. García-Martínez, D.R., *UNA PROPUESTA PARA LA ESTRUCTURA DE LA TESIS DE GRADO EN INGENIERÍA INFORMÁTICA*. p. 3.
27. *How to Write Doc Comments for the Javadoc Tool* http://java.sun.com/j2se/javadoc/writingdoccomments/ *Último acceso: 20/12/2009*.

D Glosario, bibliografía y referencias

D.1 Glosario

En esta sección se encuentran las definiciones de los términos utilizados en el informe y contexto de este proyecto. Los términos se encuentran ordenados alfabéticamente. Para cada concepto se presenta la información necesaria para que el término sea entendido sin ambigüedad, agregando si es necesario posibles sinónimos del mismo. El objetivo es tener un vocabulario que sea comprendido de la misma forma por todos ya que fue imprescindible en la etapa de estudio y búsqueda de información.

También se proporciona para cada término:

- *Similares*: sinónimos u otros términos similares o relacionados, algunos de los cuales también están definidos en este glosario.
- *Traducciones posibles*: se presentan algunas de las traducciones posibles a otros idiomas del término y sus similares.
- *Tema relacionado*: indica que dicho término fue utilizado en su mayoría en los informes relacionados a un tema específico que puede ser "Transporte", "Interfaces" o "Sistemas".
- *Links relacionados*: contiene enlaces a páginas Web con definiciones amplias del término.

Las definiciones fueron extraídas y compiladas de las siguientes fuentes (Último acceso: 27/11/2009):

- http://es.wikipedia.org/wiki/Portada
- http://www.wordreference.com/
- glosarios en español (con traducciones a inglés) de términos generales:
  - http://glosario.panamacom.com/
  - http://www.tugurium.com/gti/
  - http://www.geocities.com/Athens/2693/glosario.html
  - http://www.ati.es/novatica/glointv2.html
  - http://www.mallorcaweb.net/mostel/glosario.htm
  - http://personal.telefonica.terra.es/web/karmentxu/glosario/glosario.html
  - http://e-words.us/w/
  - http://www.alegsa.com.ar/Diccionario/diccionario.php
- glosarios de HCI
  - http://es.tldp.org/ORCA/glosario.html
  - http://ei.cs.vt.edu/~cs5714/glossary.html
  - http://www.interaction-design.org/encyclopedia/
  - http://id00156.id.tue.nl/hci/
  - http://www.usabilitybok.org/glossary
- glosarios en inglés:
  - http://en.wikibooks.org/wiki/CACS:Glossary
  - http://www.cnet.com/Resources/Info/Glossary/index.html?hhTest
  - http://www.sharpened.net/glossary/
  - http://whatis.techtarget.com/
  - http://www.geocities.com/ikind_babel/babel/babel.html
  - http://www.nuhorizons.com/glossary/computerconcepts.asp
  - http://www.directron.com/videoglossary.html
  - http://www.directron.com/glossary.html
  - http://www.whitecanyon.com/computer-security-glossary-a-d.php (de seguridad)
  - http://www.iwebtool.com/computer_glossary/
  - http://www.trinity.edu/~rjensen/245glosf.htm#Pointer
  - http://www.saugus.net/Computer/Terms/
- glosarios de transporte

- http://www.glossarist.com/glossaries/transport/
- http://www.idea.gov.uk/idk/core/page.do?pageId=81121
- http://www.nottingham.ac.uk/transportissues/glossarypage.shtml

Los términos señalados con (*) corresponden a definiciones sacadas de la literatura. El resto son conceptos utilizados exclusivamente en el ámbito de la compañía de transporte urbano.

D.1.1 Accesibilidad

Capacidad de acceso y uso de una aplicación por todas las personas independientemente de la discapacidad (física, intelectual o técnica) que presenten o de las que se deriven del contexto de uso (tecnológicas o ambientales).
Similares: usabilidad, accesibilidad web, interacción, interfaz, interfaces, interfaz gráfica, Interacción Persona-Computadora, percepción, visión, visualización
Traducciones posibles: accesibility, usability, usefulness, interaction, interface, graphic interface, HCI (Human-Computer Interaction), perception, vision, visualization
Tema relacionado: Interfaces
Links relacionados: Accesibilidad en Wikipedia

D.1.2 Accesibilidad web

Capacidad de acceso a la Web y a sus contenidos por todas las personas independientemente de la discapacidad (física, intelectual o técnica) que presenten o de las que se deriven del contexto de uso (tecnológicas o ambientales).
Similares: usabilidad, interacción, interfaz, interfaces, interfaz gráfica, Interacción Persona-Computadora, percepción, visión, visualización
Traducciones posibles: accesibility, usability, interaction, interface, graphic interface, HCI (Human-Computer Interaction), perception, vision, visualization
Tema relacionado: Interfaces
Links relacionados: Accesibilidad web en Wikipedia

D.1.3 Administración

Disposición de recursos (ómnibus o personas) con objetivos definidos (como la optimización en el uso de dichos recursos, la reducción de costos, etc.).
Similares: manejo, administrar, organización, organizar, ordenar, orden, ordenamiento, asignación, logística, planeamiento, planificación
Traducciones posibles: administration, management, manage, logistics, scheduling, schedule, pairing, plan, planning, time-tabling, time table
Tema relacionado: Transporte, Sistemas
Links relacionados:

D.1.4 Algoritmo evolutivo / Algoritmo genético

Algoritmos que hacen evolucionar una población de individuos sometiéndola a acciones aleatorias semejantes a las que actúan en la evolución biológica (mutaciones y recombinaciones genéticas), así como también a una Selección de acuerdo con algún criterio, en función del cual se decide cuáles son los individuos más adaptados, que sobreviven, y cuáles los menos aptos, que son descartados.
Similares: algoritmo, algoritmia, heurística
Traducciones posibles: genetic algorithm, evolutive algorithm, heuristic
Tema relacionado: Sistemas
Links relacionados: Algoritmo genético en Wikipedia

D.1.5 Asignación

Disposición o adjudicación de recursos en el tiempo (ómnibus o personas).
Similares: asignado(s), asignada(s), asignaciones, asignar, empatar, emparejar, tupla, dupla, viaje(s) asignado(s), ordenamiento, organización, planeamiento, planificación, programa, programación, rotación, rotaciones
Traducciones posibles: assignment, assigned, assign, schedule, scheduling, pairing, plan, planning, roster, rostering, time table, time-tabling
Tema relacionado: Transporte, Sistemas

Links relacionados:

D.1.6 Bloque (*)

Término utilizado en el ámbito de ordenamiento de vehículos. Conjunto de viajes sucesivos asignados a un ómnibus, comenzando con un pull-out y terminando con un pull-in. Un coche que deja el depósito más de una vez consta de varios bloques.
Similares: bloques, turno(s), actividad(es), tarea(s), evento(s), pull-in-trip(s), pull-out-trip(s), servicio(s), viaje(s), rotación, rotaciones, servicio(s)
Traducciones posibles: duty, duties, block(s), shift(s), task(s), activity, activities, event(s), trip(s), roster(s), service(s)
Tema relacionado: Transporte
Links relacionados:

D.1.7 Break

Tiempo en que un empleado o conductor no está realizando tareas de manejo. Igual cuenta como tiempo trabajado.
Similares: breaks, spare time, pit stop, stopover, rest stop
Traducciones posibles: descanso(s)
Tema relacionado: Transporte
Links relacionados:

D.1.8 Chofer

Persona que maneja o conduce un ómnibus.
Similares: choferes, conductor(es), conductora(s), personal, tripulación, empleado(s), flota, cuadrilla, escuadra
Traducciones posibles: crew, staff, driver(s), manpower, human resources, employee(s)
Tema relacionado: Transporte
Links relacionados:

D.1.9 Circulación

Fenómeno causado por el flujo de vehículos en una vía, calle o autopista.
Similares: tráfico, tránsito, flujo
Traducciones posibles: circulation, traffic, transit
Tema relacionado: Transporte
Links relacionados: Ingeniería de tráfico en Wikipedia Tránsito vehicular en Wikipedia

D.1.10 Coche

Vehículo en el que se realizan los viajes. Tiene un tipo que determina el tipo de viaje que puede hacer.
Similares: ómnibus, coches, unidad(es) de transporte, vehículo(s), flota, cuadrilla, escuadra
Traducciones posibles: bus, vehicle(s), fleet
Tema relacionado: Transporte
Links relacionados: Autobús en Wikipedia Medio de transporte en Wikipedia

D.1.11 Conductor

Persona que maneja o conduce un ómnibus.
Similares: conductores, conductora(s), chofer(es), personal, tripulación, empleado(s), flota, cuadrilla, escuadra
Traducciones posibles: crew, staff, driver(s), manpower, human resources, employee(s)
Tema relacionado: Transporte
Links relacionados:

D.1.12 Crew

Personal, tripulación, recursos humanos o conjunto de choferes o conductores de ómnibus.
Similares: staff, driver(s), manpower, human resources, employee(s)

Traducciones posibles: chofer(es), conductor(es), conductora(s), personal, equipo, tripulación, empleado(s), flota, cuadrilla, escuadra
Tema relacionado: Transporte
Links relacionados:

D.1.13 Crew Scheduling Problem (CSP)

Problema consistente en averiguar la disposición en tiempo de los recursos humanos con objetivos definidos (como la optimización en el uso de dichos recursos, la reducción de costos, etc.). Problema de asignación de recursos humanos a tareas. Obedece reglas correspondientes a los tiempos de descanso, su duración, su distribución durante la jornada, etc.
Similares: VSP (Vehicle Scheduling Problem), ISP (Integrated Scheduling Problem), IVSP (Integrated Vehicle Scheduling Problem), administration, management, manage, logistics, scheduling, schedule, pairing, plan, planning, time table, time-tabling, itinerary
Traducciones posibles: problema de planificación de vehículos, problema de planificación de personal, problema integral de planificación de vehículos, manejo, administrar, organización, organizar, ordenar, orden, ordenamiento, asignación, logística, planeamiento, planificación, tabla(s) de horario, itinerario
Tema relacionado: Transporte, Sistemas
Links relacionados:

D.1.14 Cuadro

Tabla o planilla de servicios. Disposición de actividades o eventos en el tiempo.
Similares: cuadros, tabla(s), planilla(s), diario(s), asignación, asignaciones, plan(es), programa(s), programación, itinerario
Traducciones posibles: schedule, table(s), chart(s), journal(s), assignment(s), plan(s), program(s), time table, itinerary
Tema relacionado: Transporte, Sistemas
Links relacionados:

D.1.15 Depósito

Lugar en que se almacenan los ómnibus.
Similares: depósitos, garage(s), garaje(s), punta(s), almacén, almacenes
Traducciones posibles: depot(s), storage space
Tema relacionado: Transporte
Links relacionados: Almacén en Wikipedia

D.1.16 Depot

Lugar en que se almacenan los ómnibus.
Similares: storage place
Traducciones posibles: depósito(s), garage(s), garaje(s), punta(s), almacén, almacenes
Tema relacionado: Transporte
Links relacionados: Almacén en Wikipedia

D.1.17 Descanso

Tiempo en que un empleado o conductor no está realizando tareas de manejo. Igual cuenta como tiempo trabajado.
Similares: descansos
Traducciones posibles: break, spare time, pit stop, stopover, rest stop
Tema relacionado: Transporte
Links relacionados:

D.1.18 Destino

Lugar por donde una línea pasa o hace una parada.
Similares: destinos, punta(s), lugar(es), línea(s)
Traducciones posibles: destination, stop, bus stop
Tema relacionado: Transporte
Links relacionados:

D.1.19 Día hábil

Día de la semana que puede ser lunes, martes, miércoles, jueves o viernes siempre que no sea feriado.
Similares: días hábiles, jornada(s)
Traducciones posibles: work day, journey(s), week day
Tema relacionado: Transporte
Links relacionados:

D.1.20 Día no hábil

Día de la semana que puede ser sábado, domingo o cualquier día de la semana que sea feriado.
Similares: días no hábiles, jornada(s), fin(es) de semana
Traducciones posibles: weekend, holliday
Tema relacionado: Transporte
Links relacionados:

D.1.21 Día útil

Día de trabajo en que se realizan servicios. Puede ser un día hábil o no.
Similares: días útiles, jornada(s)
Traducciones posibles: work day, journey(s)
Tema relacionado: Transporte
Links relacionados:

D.1.22 Diagrama

Herramienta gráfica o forma de visualización de datos. Se refiere generalmente a una tabla o planilla de servicios en una representación gráfica.
Similares: cuadros, tabla(s), planilla(s), diario(s), asignación, asignaciones
Traducciones posibles: schedule, table, chart, journal, assignment
Tema relacionado: Sistemas
Links relacionados:

D.1.23 Diagrama de gantt

Herramienta gráfica cuyo objetivo es mostrar el tiempo de dedicación previsto para diferentes tareas o actividades a lo largo de un tiempo total determinado. En principio, no indica las relaciones existentes entre actividades, pero la posición de cada tarea a lo largo del tiempo hace que se puedan identificar dichas relaciones e interdependencias.
Similares: diagramas de Gantt, cuadro(s) de Gantt
Traducciones posibles: Gantt chart(s), Gantt diagram(s)
Tema relacionado: Sistemas
Links relacionados: Diagrama de Gantt en Wikipedia

D.1.24 Diario

Planilla o conjunto de planillas con las actividades, eventos u horarios que cumplen los servicios o el personal.
Similares: libro(s), planilla(s) de servicios, diario(s) de servicios, diarios de actividades, planilla(s), plan(es), programación, servicio(s), horario(s), tabla(s) de horario, itinerario, solución
Traducciones posibles: schedule, schedule table(s), activity chart(s), service(s), time table, itinerary
Tema relacionado: Transporte
Links relacionados:

D.1.25 Diario de actividad

Planilla o conjunto de planillas con las actividades, eventos u horarios que cumplen los servicios o el personal.
Similares: libro(s), planilla(s) de servicios, diario(s) de servicios, diarios de actividades, planilla(s), plan(es), programación, servicio(s), horario(s), tabla(s) de horario, itinerario
Traducciones posibles: schedule, schedule table(s), activity chart(s), service(s), time table, itinerary
Tema relacionado: Transporte

Links relacionados:

D.1.26 Diario de servicios

Planilla o conjunto de planillas con las actividades, eventos u horarios que cumplen los servicios o el personal.
Similares: libro(s), planilla(s) de servicios, diarios de servicios, diario(s) de actividades, planilla(s), plan(es), programación, servicio(s), horario(s), tabla(s) de horario, itinerario
Traducciones posibles: schedule, schedule table(s), activity chart(s), service(s), time table, itinerary
Tema relacionado: Transporte
Links relacionados:

D.1.27 Diseño

Composición o armado de un diario de servicios.
Similares: diseños, composiciones, armados, diseñar, planificar, planificación, planificamiento, plan, horario, diario
Traducciones posibles: design, composition, scheduling, time-tabling
Tema relacionado: Transporte, Sistemas
Links relacionados:

D.1.28 Diseño Centrado en el Usuario

Proceso de diseño en el que las necesidades, los deseos y las limitaciones del usuario final de una interfaz o documento toman una atención y relevancia considerable en cada nivel del proceso de diseño.
Similares: interacción, interfaz, interfaces, interfaz gráfica, visualización, visión
Traducciones posibles: User-centered design (UCD), design, interface, interaction, visualization, vision
Tema relacionado: Interfaces
Links relacionados: Diseño centrado en el usuario en Wikipedia

D.1.29 Driver

Persona que maneja o conduce un ómnibus.
Similares: crew, staff, driver(s), manpower, human resources, employee(s)
Traducciones posibles: conductores, conductora(s), chofer(es), personal, tripulación, empleado(s), flota, cuadrilla, escuadra
Tema relacionado: Transporte
Links relacionados:

D.1.30 Duty (*)

Término utilizado tanto en el ordenamiento de vehículos como en el ordenamiento de conductores. Según el lugar es equivalente a bloque o shift.
Similares: duties, block(s), shift(s), task(s), activity, activities, event(s), pull-in-trip(s), pull-out-trip(s), trip(s), roster(s), service(s)
Traducciones posibles: bloque(s), turno(s), actividad(es), tarea(s), evento(s), viaje(s), servicio(s), rotación, rotaciones, servicio(s)
Tema relacionado: Transporte
Links relacionados:

D.1.31 Ejecución

Cargar un programa en memoria y que la computadora realice las instrucciones en el mismo.
Similares: ejecuciones, experimento(s)
Traducciones posibles: execution(s)
Tema relacionado: Sistemas
Links relacionados:

D.1.32 Empleado

Persona que maneja o conduce un ómnibus. También puede referirse a los usuarios del sistema.
Similares: empleados, chofer(es), conductor(es), conductora(s), personal, equipo, tripulación, flota, cuadrilla, escuadra, usuario
Traducciones posibles: employee(s), crew, staff, driver(s), manpower, human resources, user

Tema relacionado: Transporte
Links relacionados:

D.1.33 Equilibrio de kilómetros realizados

Existe equilibrio de kilómetros realizados cuando todos los coches realizan durante el mes la misma cantidad de kilómetros.
Similares: balance
Traducciones posibles: balance, equilibrium
Tema relacionado: Transporte
Links relacionados:

D.1.34 Estación

Es uno de los períodos en que se divide el año. Puede ser verano, otoño, invierno o primavera. Determina el flujo de pasajeros y por ende la cantidad de viajes que realizan los servicios.
Similares: estaciones, estación del año, estaciones del año
Traducciones posibles: season
Tema relacionado: Transporte
Links relacionados:

D.1.35 Evento

Acontecimiento, hecho o actividad que ocurre durante un servicio. Puede ser: realizar un viaje útil, o un descanso, o un viaje expreso.
Similares: eventos, actividad(es), bloque(s), tarea(s), servicio(s)
Traducciones posibles: event(s), block(s), shift(s), task(s), activity, activities, duty, duties, service(s)
Tema relacionado: Transporte
Links relacionados:

D.1.36 Experimento

Secuencia de ejecuciones.
Similares: ejecución(es)
Traducciones posibles: experiment(s), execution(s)
Tema relacionado: Sistemas
Links relacionados:

D.1.37 Flight segment (*)

Término utilizado en el ámbito de la planificación aérea. Es un vuelo desde origen a destino.
Similares: travel, scale, trip
Traducciones posibles: viaje, vuelo, segmento de vuelo, escala
Tema relacionado: Transporte
Links relacionados:

D.1.38 Flota

Conjunto de ómnibus o conductores o ambos.
Similares: cuadrilla, escuadra, empleado(s), chofer(es), conductor(es), conductora(s), personal, equipo, tripulación
Traducciones posibles: fleet, squad, employee(s), crew, staff, driver(s), manpower, human resources
Tema relacionado: Transporte
Links relacionados: Autobús en Wikipedia Medio de transporte en Wikipedia

D.1.39 Framework

Es una estructura de soporte definida, en la cual otro proyecto de software puede ser organizado y desarrollado.

Similares:
Traducciones posibles:
Tema relacionado: Interfaces
Links relacionados: Framework en Wikipedia

D.1.40 Garage

Lugar en que se almacenan los ómnibus.
Similares: depósito(s), garage(s), garaje(s), punta(s), almacén, almacenes
Traducciones posibles: depot(s), storage space
Tema relacionado: Transporte
Links relacionados: Almacén en Wikipedia

D.1.41 Graphic User Interface (GUI)

Artefacto tecnológico de un sistema interactivo que posibilita, a través del uso y la representación del lenguaje visual, una interacción amigable con un sistema informático.
Similares: interface, graphic interface, user interface, visualization, vision, perception
Traducciones posibles: interfaz, interface(s), interfaz gráfica, interface(s) gráfica(s), interfaz de usuario, interface(s) de usuario (también puede aparecer como "interfase"), visualización, visión, percepción
Tema relacionado: Interfaces
Links relacionados: Interfaz gráfica de usuario en Wikipedia

D.1.42 Heurística

Algoritmo que abandona el tener buenos tiempos de ejecución y/o buenas soluciones; por ejemplo, normalmente encuentran buenas soluciones, aunque en ocasiones no hay pruebas de que la solución no pueda ser arbitrariamente errónea; o se ejecuta razonablemente rápido, aunque no existe tampoco prueba de que deba ser así.
Similares: heurísticas, meta-heurística(s), algoritmo(s) evolutivo(s), algoritmo(s) genético(s), algoritmo heurístico
Traducciones posibles: heuristic
Tema relacionado: Sistemas
Links relacionados: Heurística en Wikipedia Algoritmo heurístico en Wikipedia

D.1.43 Hora extra

Hora no comprendida en el jornal de 8 horas de un empleado.
Similares: chofer(es), conductor(es), conductora(s), personal, tripulación, empleado(s)
Traducciones posibles: overtime, extra hour, fleet, crew, staff, driver(s), manpower, human resources, employee(s)
Tema relacionado: Transporte
Links relacionados: Hora extra en wikipedia

D.1.44 Hora pico

Parte del día en el que la congestión del tránsito es mayor.
Similares: horas pico, tráfico, tránsito
Traducciones posibles: peak hour, traffic, transit
Tema relacionado: Transporte
Links relacionados: Rush hour en wikipedia

D.1.45 Human-Computer Interaction (HCI)

Disciplina que tiene como objetivo el Diseño, Implementación y Pruebas de sistemas interactivos para el uso humano.
Similares: interaction
Traducciones posibles: interacción persona-computador(a), interacción persona-ordenador, interacción
Tema relacionado: Interfaces
Links relacionados: Interacción Persona-Computador

D.1.46 Integrated Scheduling Problem (ISP)

Es el problema de resolver del VSP (Vehicle Scheduling Problem) y el CSP (Crew Scheduling Problem) simultáneamente, en vez de por separado.
Similares: VSP (Vehicle Scheduling Problem), CSP (Crew Scheduling Problem), IVSP (Integrated Vehicle Scheduling Problem), administration, management, manage, logistics, scheduling, schedule, pairing, plan, planning, time table, time-tabling, itinerary, bus, vehicle(s), fleet, crew, staff, driver(s), manpower, human resources, employee(s)
Traducciones posibles: problema de planificación de vehículos, problema de planificación de personal, problema integral de planificación de vehículos, manejo, administrar, organización, organizar, ordenar, orden, ordenamiento, asignación, logística, planeamiento, planificación, tabla(s) de horario, itinerario, chofer(es), conductor(es), conductora(s), personal, tripulación, empleado(s), unidad(es) de transporte, vehículo(s), flota, cuadrilla, escuadra
Tema relacionado: Transporte, Sistemas
Links relacionados:

D.1.47 Integrated Vehicle Scheduling Problem (IVSP)

Es el problema de resolver del VSP (Vehicle Scheduling Problem) y el CSP (Crew Scheduling Problem) simultáneamente, en vez de por separado.
Similares: VSP (Vehicle Scheduling Problem), CSP (Crew Scheduling Problem), ISP (Integrated Scheduling Problem), administration, management, manage, logistics, scheduling, schedule, pairing, plan, planning, time table, time-tabling, itinerary, bus, vehicle(s), fleet, crew, staff, driver(s), manpower, human resources, employee(s)
Traducciones posibles: problema de planificación de vehículos, problema de planificación de personal, problema integral de planificación de vehículos, manejo, administrar, organización, organizar, ordenar, orden, ordenamiento, asignación, logística, planeamiento, planificación, tabla(s) de horario, itinerario, chofer(es), conductor(es), conductora(s), personal, tripulación, empleado(s), unidad(es) de transporte, vehículo(s), flota, cuadrilla, escuadra
Tema relacionado: Transporte, Sistemas
Links relacionados:

D.1.48 Interacción

Proceso que establece un usuario con un dispositivo, sistema u objeto determinado.
Similares: interacción persona-computador(a), interacción persona-ordenador, percepción, comunicación
Traducciones posibles: interaction, HCI (Human-Computer Interaction), perception, comunication
Tema relacionado: Interfaces
Links relacionados: Interacción en Wikipedia

D.1.49 Interacción Persona-Computadora

Disciplina que tiene como objetivo el Diseño, Implementación y Pruebas de sistemas interactivos para el uso humano.
Similares: interacción persona-computador, interacción persona-ordenador
Traducciones posibles: interaction, HCI (Human-Computer Interaction)
Tema relacionado: Interfaces
Links relacionados: Interacción Persona-Computador

D.1.50 Interaccionar

Establecer una relación de comunicación entre un usuario y un dispositivo, sistema u objeto determinado.
Similares: interacción, interactuar, interacción persona-computador(a), interacción persona-ordenador, percepción, comunicación
Traducciones posibles: interact, interaction, HCI (Human-Computer Interaction), perception, comunication

Tema relacionado: Interfaces
Links relacionados: Interacción en Wikipedia

D.1.51 Interfaz

Conexión física y funcional entre dos aparatos o sistemas independientes. Parte del sistema que permite la circulación correcta y sencilla de información entre varias aplicaciones y entre el propio programa y el usuario. Conjunto de comandos y/o métodos que permiten la intercomunicación del programa con cualquier otro programa o entre partes (módulos) del propio programa o elemento interno o externo.
Similares: interface(s), interfaz gráfica, interface(s) gráfica(s), interfaz de usuario, interface(s) de usuario (también puede aparecer como "interfase(s)"), comunicación, interacción
Traducciones posibles: interface, graphic interface, user interface, GUI (graphic user interface), interaction, comunication
Tema relacionado: Interfaces
Links relacionados: Interfaz en Wikipedia

D.1.52 Interfaz de usuario

Forma en que los usuarios pueden comunicarse con una computadora, y comprende todos los puntos de contacto entre el usuario y el equipo.
Similares: interface(s), interfaces gráficas, interfaz de usuario, interface(s) de usuario (también puede aparecer como "interfase(s)")
Traducciones posibles: interface, graphic interface, user interface, GUI (graphic user interface)
Tema relacionado: Interfaces
Links relacionados: Interfaz de usuario en Wikipedia

D.1.53 Interfaz gráfica (de usuario)

Artefacto tecnológico de un sistema interactivo que posibilita, a través del uso y la representación del lenguaje visual, una interacción amigable con un sistema informático.
Similares: interface(s), interfaces gráficas, interfaz de usuario, interface(s) de usuario (también puede aparecer como "interfase(s)"), visión, percepción, visualización
Traducciones posibles: interface, graphic interface, user interface, GUI (graphic user interface), visualization, vision, perception
Tema relacionado: Interfaces
Links relacionados: Interfaz de usuario en Wikipedia Interfaz gráfica de usuario en Wikipedia

D.1.54 Itinerario

Planilla o conjunto de planillas con las actividades, eventos u horarios que cumplen los servicios o el personal.
Similares: libro(s), planilla(s) de servicios, diario(s) de servicios, diarios de actividades, planilla(s), plan(es), programación, servicio(s), horario(s), tabla(s) de horario, itinerario
Traducciones posibles: itinerary, schedule, schedule table(s), activity chart(s), service(s), time table
Tema relacionado: Transporte
Links relacionados:

D.1.55 Libro

Planilla o conjunto de planillas con las actividades, eventos u horarios que cumplen los servicios o el personal para una estación del año.
Similares: diario(s), planilla(s) de servicios, diario(s) de servicios, diario(s) de actividades, planilla(s), servicio(s), horario(s), tabla(s) de horario, itinerario
Traducciones posibles: schedule, schedule table(s), activity chart(s), time table, itinerary, service(s)
Tema relacionado: Transporte
Links relacionados:

D.1.56 Línea

Define un recorrido ficticio, y representa un conjunto de recorridos o trayectos transitados por los ómnibus de la empresa que pueden superponerse entre sí. Es un recorrido con origen y destino que tiene un vehículo

(pero no tiene hora de salida ni llegada, lo que lo diferencia de los viajes. Los viajes son instancias particulares de una línea).
Similares: líneas, línea(s) departamental(es), línea(s) derivada(s), línea(s) social(es), punto(s) de convergencia, destino(s), punta(s), viaje(s)
Traducciones posibles: bus line(s), trip(s)
Tema relacionado: Transporte
Links relacionados:

D.1.57 Línea departamental

Línea que no ingresa a Montevideo.
Similares: línea(s), línea(s) departamental(es), línea(s) derivada(s), línea(s) social(es)
Traducciones posibles: bus line(s), trip(s)
Tema relacionado: Transporte
Links relacionados:

D.1.58 Línea derivada

Es un tipo de línea departamental. Estas no tienen fines económicos, son líneas que le generan pérdidas a la empresa, existen por motivos sociales.
Similares: línea(s), línea(s) departamental(es), líneas derivadas, línea(s) social(es)
Traducciones posibles: bus line(s), trip(s)
Tema relacionado: Transporte
Links relacionados:

D.1.59 Línea social

Es un tipo de línea departamental. Estas no tienen fines económicos, son líneas que le generan pérdidas a la empresa, existen por motivos sociales.
Similares: línea(s), línea(s) departamental(es), línea(s) derivada(s), líneas sociales
Traducciones posibles: bus line(s), trip(s)
Tema relacionado: Transporte
Links relacionados:

D.1.60 Logística

Disposición de recursos (ómnibus o personas) con objetivos definidos (como la optimización en el uso de dichos recursos, la reducción de costos, etc.). Conjunto de medios y métodos necesarios para llevar a cabo la organización de una empresa, o de un servicio, especialmente de distribuciones.
Similares: administración, manejo, administrar, ordenar, orden, ordenamiento, asignación, organizar, organización, planeamiento, planificación, plan(es), tabla(s) de horario, itinerario
Traducciones posibles: logistics, management, manage, administration, scheduling, schedule, plan, planning, time-tabling, itinerary
Tema relacionado: Transporte
Links relacionados: Logística en Wikipedia

D.1.61 Management

Disposición de recursos (ómnibus o personas) con objetivos definidos (como la optimización en el uso de dichos recursos, la reducción de costos, etc.).
Similares: manage, administration, logistics, sheduling, schedule, plan, planning, itinerary
Traducciones posibles: manejo, administración, administrar, organización, organizar, ordenar, orden, ordenamiento, asignación, logística, planeamiento, planificación, plan(es), tabla(s) de horario, itinerario
Tema relacionado: Transporte, Sistemas
Links relacionados:

D.1.62 Manejo

Disposición de recursos (ómnibus o personas) con objetivos definidos (como la optimización en el uso de dichos recursos, la reducción de costos, etc.).

Similares: administración, administrar, organización, organizar, ordenar, orden, ordenamiento, asignación, logística, planeamiento, planificación, plan(es), tabla(s) de horario, itinerario
Traducciones posibles: management, manage, administration, logistics, scheduling, schedule, plan, planning, itinerary
Tema relacionado: Transporte, Sistemas
Links relacionados:

D.1.63 Manpower

Personal, tripulación, recursos humanos o conjunto de choferes o conductores de ómnibus.
Similares: crew, staff, driver(s), human resources, employee(s)
Traducciones posibles: chofer(es), conductor(es), conductora(s), personal, equipo, tripulación, empleado(s), flota, cuadrilla, escuadra
Tema relacionado: Transporte
Links relacionados:

D.1.64 Ómnibus

Vehículo en el que se realizan los viajes. Tiene un tipo que determina el tipo de viaje que puede hacer.
Similares: unidad(es) de transporte, coche(s), vehículo(s), flota, cuadrilla, escuadra
Traducciones posibles: bus, vehicle(s), fleet
Tema relacionado: Transporte
Links relacionados: Autobús en Wikipedia Medio de transporte en Wikipedia

D.1.65 Ómnibus alternativa

Ómnibus viejo que solo puede trabajar en líneas derivadas, salvo casos excepcionales.
Similares: ómnibus, ómnibus preferencial, ómnibus viaggio, unidad(es) de transporte, vehículo(s), flota, cuadrilla, escuadra
Traducciones posibles: bus, vehicle(s), fleet
Tema relacionado: Transporte
Links relacionados: Autobús en Wikipedia Medio de transporte en Wikipedia

D.1.66 Ómnibus preferencial

Ómnibus nuevo que recién se está pagando, por lo que se le asigna una mayor cantidad de Km. para hacer.
Similares: ómnibus, ómnibus alternativa, ómnibus viaggio, unidad(es) de transporte, vehículo(s), flota, cuadrilla, escuadra
Traducciones posibles: bus, vehicle(s), fleet
Tema relacionado: Transporte
Links relacionados: Autobús en Wikipedia Medio de transporte en Wikipedia

D.1.67 Ómnibus viaggio

Ómnibus más grande que los comunes, llevan 15 personas más.
Similares: ómnibus, ómnibus alternativa, ómnibus preferencial, unidad(es) de transporte, vehículo(s), flota, cuadrilla, escuadra
Traducciones posibles: bus, vehicle(s), fleet
Tema relacionado: Transporte
Links relacionados: Autobús en Wikipedia Medio de transporte en Wikipedia

D.1.68 Ordenamiento

Disposición en tiempo de recursos (ómnibus o personas) con objetivos definidos (como la optimización en el uso de dichos recursos, la reducción de costos, etc.).
Similares: administración, administrar, organización, organizar, ordenar, orden, manejo, asignación, logística, planeamiento, planificación, programa, programación, plan(es), tabla(s) de horario, itinerario
Traducciones posibles: management, manage, administration, logistics, scheduling, schedule, pairing, assignment, assigning, assign, plan, planning, itinerary
Tema relacionado: Transporte, Sistemas
Links relacionados:

D.1.69 Pairing (*)

Término utilizado en el ámbito de la planificación aérea. Es una secuencia de viajes a ser realizados por un conductor.
Similares: assignment, assigning, assign, crew, staff, driver(s), manpower, human resources, employee(s), schedule, scheduling, roster, rostering, service(s), itinerary
Traducciones posibles: par(es), asignación, asignaciones, asignar, asignado(s), asignada(s), viaje(s) asignado(s), empatar, emparejar, tupla, dupla, viaje(s), servicio(s), chofer(es), conductor(es), conductora(s), personal, tripulación, empleado(s), flota, programa(s), programación, planificación, planeamiento, horario(s), plan(es), rotación, rotaciones, proceso de rotación, itinerario
Tema relacionado: Transporte
Links relacionados:

D.1.70 Percepción

Función psíquica que permite al organismo, a través de los sentidos, recibir, elaborar e interpretar la información proveniente de su entorno.
Similares: percepción visual/ocular/óptica, visión, visualización, sentido, interpretación
Traducciones posibles: perception, sense, vision, visualization
Tema relacionado: Interfaces
Links relacionados: Percepción en Wikipedia

D.1.71 Personal

Personas, tripulación, recursos humanos o conjunto de choferes o conductores de ómnibus.
Similares: chofer(es), conductor(es), conductora(s), equipo, tripulación, empleado(s), flota, cuadrilla, escuadra
Traducciones posibles: crew, staff, driver(s), manpower, human resources, employee(s)
Tema relacionado: Transporte
Links relacionados:

D.1.72 Plan

Disposición en tiempo de recursos (ómnibus o personas) con objetivos definidos (como la optimización en el uso de dichos recursos, la reducción de costos, etc.).
Similares: administración, administrar, organización, organizar, ordenar, orden, manejo, asignación, logística, ordenamiento, planificación, planeamiento, programa, programación, servicio(s), horario(s), itinerario, diario(s), libro(s), tabla(s) de horario, itinerario
Traducciones posibles: management, manage, administration, logistics, scheduling, schedule, pairing, assignment, assigning, assign, plan, planning, service(s), itinerary
Tema relacionado: Transporte, Sistemas
Links relacionados:

D.1.73 Planeamiento

Disposición en tiempo de recursos (ómnibus o personas) con objetivos definidos (como la optimización en el uso de dichos recursos, la reducción de costos, etc.).
Similares: administración, administrar, organización, organizar, ordenar, orden, manejo, asignación, logística, ordenamiento, plan, planificación, programa, programación, servicio(s), horario(s), diario(s), libro(s), tabla(s) de horario, itinerario
Traducciones posibles: management, manage, administration, logistics, scheduling, schedule, pairing, assignment, assigning, assign, plan, planning, service(s), itinerary
Tema relacionado: Transporte, Sistemas
Links relacionados:

D.1.74 Planificación

Disposición en tiempo de recursos (ómnibus o personas) con objetivos definidos (como la optimización en el uso de dichos recursos, la reducción de costos, etc.).
Similares: administración, administrar, organización, organizar, ordenar, orden, manejo, asignación, logística, plan, planeamiento, ordenamiento, programa, programación, servicio(s), horario(s), diario(s), libro(s), tabla(s) de horario, itinerario
Traducciones posibles: management, manage, administration, logistics, scheduling, schedule, pairing, assignment, assigning, assign, plan, planning, service(s), itinerary
Tema relacionado: Transporte, Sistemas

Links relacionados:

D.1.75 Planilla

Calendario o asignación de actividades, eventos u horarios que cumplen los servicios o el personal.
Similares: planilla(s) de servicios, diario(s) de servicios, diario(s) de actividades, libro(s), programa(s), plan(es), cuadro(s), servicio(s), horario(s), tabla(s) de horario, itinerario
Traducciones posibles: schedule, schedule table(s), activity chart(s), plan, program, service(s), time table, itinerary
Tema relacionado: Transporte
Links relacionados:

D.1.76 Problema de planificación de personal

Problema consistente en averiguar la disposición en tiempo de los recursos humanos con objetivos definidos (como la optimización en el uso de dichos recursos, la reducción de costos, etc.). Problema de asignación de recursos humanos a tareas. Obedece reglas correspondientes a los tiempos de descanso, su duración, su distribución durante la jornada, etc.
Similares: problema de planificación de vehículos, problema integral de planificación de vehículos, manejo, administrar, organización, organizar, ordenar, orden, ordenamiento, asignación, logística, planeamiento, planificación, plan(es), tabla(s) de horario, horario(s) , itinerario, chofer(es), conductor(es), conductora(s), personal, tripulación, empleado(s), flota, cuadrilla, escuadra
Traducciones posibles: VSP (Vehicle Scheduling Problem), CSP (Crew Scheduling Problem), ISP (Integrated Scheduling Problem), IVSP (Integrated Vehicle Scheduling Problem), administration, management, manage, logistics, scheduling, schedule, pairing, plan, planning, time table, time-tabling, itinerary, crew, staff, driver(s), manpower, human resources, employee(s)
Tema relacionado: Transporte, Sistemas
Links relacionados:

D.1.77 Problema de planificación de vehículos

Problema consistente en averiguar la disposición en tiempo de las unidades de transporte con objetivos definidos (como la optimización en el uso de dichos recursos, la reducción de costos, etc.). Problema de asignación de unidades de transporte a tareas en el tiempo.
Similares: problema de planificación de personal, problema integral de planificación de vehículos, manejo, administrar, organización, organizar, ordenar, orden, ordenamiento, asignación, logística, planeamiento, planificación, plan(es), horario(s) , itinerary, tabla(s) de horario, unidad(es) de transporte, vehículo(s), flota, cuadrilla, escuadra
Traducciones posibles: VSP (Vehicle Scheduling Problem), CSP (Crew Scheduling Problem), ISP (Integrated Scheduling Problem), IVSP (Integrated Vehicle Scheduling Problem), administration, management, manage, logistics, scheduling, schedule, pairing, plan, planning, time table, time-tabling, itinerary, bus, vehicle(s), fleet
Tema relacionado: Transporte, Sistemas
Links relacionados:

D.1.78 Problema de ruteo de vehículos

Problema consistente en averiguar las rutas de una flota de transporte para dar servicio a unos clientes.
Similares: ruteo, ruteamiento, ruta(s), vehículo(s), Problema de rutas de vehículos
Traducciones posibles: VRP (Vehicle Routing Problem), routing
Tema relacionado: Transporte
Links relacionados: Problema de rutas de vehículos en Wikipedia

D.1.79 Programa

(1) Conjunto de instrucciones en un lenguaje de programación que pueden ser interpretadas por una computadora para ser ejecutadas.
(2) También puede ser el scheduling, planificación, u organización las actividades de ciertos recursos (ómnibus o personas). Disposición en tiempo de recursos (ómnibus o personas) con objetivos definidos (como la optimización en el uso de dichos recursos, la reducción de costos, etc.).
Similares: administración, administrar, organización, organizar, ordenar, orden, manejo, asignación, logística, planificación, planeamiento, ordenamiento, programación, servicio(s), horario(s), diario(s), libro(s), tabla(s) de horario, itinerario
Traducciones posibles: management, manage, administration, logistics, scheduling, schedule, pairing, assignment, assigning, assign, plan, planning, service(s), time table, time-tabling, itinerary
Tema relacionado: Transporte, Sistemas

Links relacionados:

D.1.80 Programación

(1) Hacer programas o aplicaciones.
(2) También puede ser el scheduling, planificación, armar los horarios u organizar las actividades de ciertos recursos (ómnibus o personas). Disposición en tiempo de recursos (ómnibus o personas) con objetivos definidos (como la optimización en el uso de dichos recursos, la reducción de costos, etc.).
Similares: administración, administrar, organización, organizar, ordenar, orden, manejo, asignación, logística, plan(es), planificación, planeamiento, ordenamiento, programa(s), servicio(s), horario(s), diario(s), libro(s), tabla(s) de horario, itinerario
Traducciones posibles: management, manage, administration, logistics, scheduling, schedule, pairing, assignment, assigning, assign, plan, planning, service(s), itinerary
Tema relacionado: Transporte, Sistemas
Links relacionados:

D.1.81 Public

Perteneciente a todas las personas, no solo a un grupo privado o particular.
Similares: public transport
Traducciones posibles: público(s), pública(s), transporte público
Tema relacionado: Transporte
Links relacionados: Transporte público en Wikipedia

D.1.82 Público

Perteneciente a todas las personas, no solo a un grupo privado o particular.
Similares: públicos, pública(s)
Traducciones posibles: public, public transport
Tema relacionado: Transporte
Links relacionados: Transporte público en Wikipedia

D.1.83 Pull-in-trip (*)

Término utilizado en el ámbito de ordenamiento de vehículos. Viaje que realiza un ómnibus desde el lugar de finalización de su último viaje útil, hasta su garaje.
Similares: travel, trip, return
Traducciones posibles: retorno, vuelta a depósito, viaje de retorno, regreso, viaje de regreso
Tema relacionado: Transporte
Links relacionados:

D.1.84 Pull-out-trip (*)

Término utilizado en el ámbito de ordenamiento de vehículos. Viaje que realiza un ómnibus desde su garaje hasta el punto de partida del primer viaje útil.
Similares: travel, trip, departure
Traducciones posibles: salida, salida de depósito, viaje de salida
Tema relacionado: Transporte
Links relacionados:

D.1.85 Punta

Se le denomina así a un garage, sobretodo a un garage del interior.
Similares: puntas, depósito(s), garage(s), garaje(s), punta(s)
Traducciones posibles: depot(s), storage space
Tema relacionado: Transporte
Links relacionados:

D.1.86 Punto de convergencia

Es donde convergen varias líneas. En estos puntos es importante que los ómnibus lleguen a una frecuencia constante.
Similares: puntos de convergencia, destino(s), lugar(es), línea(s)
Traducciones posibles: convergence point, convergence corner, destination
Tema relacionado: Transporte
Links relacionados:

D.1.87 Recorrido

Conjunto ordenado de esquinas, por las que pasan los coches.
Similares: recorridos, trayecto(s), ruta(s)
Traducciones posibles: path, route, trajectory
Tema relacionado: Transporte
Links relacionados:

D.1.88 Recursos humanos

Personas, tripulación o conjunto de choferes o conductores de ómnibus.
Similares: chofer(es), conductor(es), conductora(s), personal, tripulación, empleado(s), flota, cuadrilla, escuadra
Traducciones posibles: crew, staff, driver(s), manpower, human resources, employee(s)
Tema relacionado: Transporte
Links relacionados:

D.1.89 Roster / Rostering

Asignación de recursos humanos a un plan o diario de actividades o servicios. También puede ser turno que le toca realizar a una persona.
Similares: rosters, rostering, assignment, assigned, assign, schedule, scheduling, pairing, plan, planning, shift, duty, duties, block(s), shift(s), task(s), activity, activities, event(s), service(s), time table, time-tabling, itinerary
Traducciones posibles: rotación, rotaciones, proceso de rotación, asignado(s), asignada(s), asignación, asignaciones, asignar, empatar, emparejar, tupla, dupla, viaje(s) asignado(s), ordenamiento, organización, planeamiento, plan(es), planificación, programa(s), programación, bloque(s), turno(s), actividad(es), tarea(s), evento(s), viaje(s), servicio(s), horario(s), diario(s), libro(s), tabla(s) de horario, itinerario
Tema relacionado: Transporte
Links relacionados:

D.1.90 Rotación / Proceso de rotación

(1) Es una instancia del proceso de rotación. Es decir, es la asignación de coches a servicios de un determinado día.
(2) Durante el proceso de planificación se genera un conjunto de servicios para cada conjunto de ómnibus del mismo tipo dentro de cada garaje. Ambos conjuntos tendrán el mismo tamaño. El proceso de rotación se refiere a que a cada ómnibus se le asociará un servicio diferente cada día dentro del conjunto correspondiente.
Similares: rotaciones, asignado(s), asignada(s), asignación, asignaciones, asignar, empatar, emparejar, tupla, dupla, viaje(s) asignado(s), ordenamiento, organización, plan(es), planeamiento, planificación, programa(s), programación, bloque(s), turno(s), actividad(es), tarea(s), evento(s), viaje(s), servicio(s), horario(s), diario(s), libro(s), tabla(s) de horario, itinerary
Traducciones posibles: roster(s), rostering, assignment, assigned, assign, schedule, scheduling, pairing, plan, planning, shift, duty, duties, block(s), shift(s), task(s), activity, activities, event(s), service(s), itinerary
Tema relacionado: Transporte
Links relacionados:

D.1.91 Routing

Averiguar las rutas de una flota de transporte para dar servicio a unos clientes.
Similares: VRP (Vehicle Routing Problem), road
Traducciones posibles: ruteo, ruteamiento, ruta(s), vehículo(s), Problema de ruteo de vehículos, Problema de rutas de vehículos
Tema relacionado: Transporte
Links relacionados: Problema de rutas de vehículos en Wikipedia

D.1.92 Rush hour

Parte del día en el que la congestión del tránsito es mayor.
Similares: peak hour, traffic, transit
Traducciones posibles: hora pico, tráfico, tránsito
Tema relacionado: Transporte
Links relacionados: Rush hour en wikipedia

D.1.93 Ruteo

Averiguar las rutas de una flota de transporte para dar servicio a unos clientes.
Similares: ruteamiento, ruta(s), vehículo(s), Problema de ruteo de vehículos, Problema de rutas de vehículos
Traducciones posibles: routing, VRP (Vehicle Routing Problem), road
Tema relacionado: Transporte
Links relacionados: Problema de rutas de vehículos en Wikipedia

D.1.94 Schedule / Scheduling

Disposición en tiempo de recursos (ómnibus o personas) con objetivos definidos (como la optimización en el uso de dichos recursos, la reducción de costos, etc.).
Similares: administración, administrar, organización, organizar, ordenar, orden, manejo, asignación, logística, plan(es), planificación, planeamiento, ordenamiento, programa, programación, servicio(s), horario(s), diario(s), libro(s), tabla(s) de horario, itinerario
Traducciones posibles: management, manage, administration, logistics, scheduling, schedule, pairing, assignment, assigning, assign, plan, planning, service(s), itinerary
Tema relacionado: Transporte
Links relacionados: Planificación de transporte en Wikipedia

D.1.95 Servicio

Secuencia de viajes que realiza una unidad de transporte (ómnibus). Un servicio especifica la misión de un ómnibus durante todo un día útil y está compuesto por un conjunto de entre 0 a 3 turnos.
Similares: servicios, bloque(s), turno(s), actividad(es), tarea(s), evento(s), viaje(s), servicio(s), rotación, rotaciones, diario(s), libro(s), planilla(s) de servicios, diarios de servicios, diario(s) de actividades, planilla(s), plan(es), programación, planificación, planeamiento
Traducciones posibles: service(s), duty, duties, block(s), shift(s), task(s), activity, activities, event(s), pull-in-trip(s), pull-out-trip(s), trip(s), roster(s), schedule, schedule table(s), activity chart(s), time table
Tema relacionado: Transporte
Links relacionados:

D.1.96 Shift (*)

Término utilizado en el ámbito de ordenamiento de conductores. Es el trabajo planificado para ser realizado por un chofer en un día.
Similares: straight shift, duty, duties, block(s), task(s), activity, activities, event(s), trip(s), roster(s), service(s), time table
Traducciones posibles: bloque(s), turno(s), actividad(es), tarea(s), evento(s), pull-in-trip(s), pull-out-trip(s), servicio(s), viaje(s), rotación, rotaciones, servicio(s), diario(s), libro(s)
Tema relacionado: Transporte
Links relacionados:

D.1.97 Simulación

Experimentación con un modelo de una hipótesis o un conjunto de hipótesis de trabajo. Generalmente se refiere a la construcción de un modelo abstracto que representa algún sistema de la vida real.
Similares: modelo de simulación
Traducciones posibles: simulation, model
Tema relacionado: Sistemas
Links relacionados: Simulación en Wikipedia

D.1.98 Solución

Planilla o conjunto de planillas con las actividades, eventos u horarios que cumplen los servicios o el personal.
Similares: libro(s), planilla(s) de servicios, diario(s), diario(s) de servicios, diarios de actividades, planilla(s), plan(es), programación, servicio(s), horario(s), tabla(s) de horario, itinerario, solución
Traducciones posibles: schedule, schedule table(s), activity chart(s), service(s), time table, itinerary
Tema relacionado: Transporte
Links relacionados:

D.1.99 Soporte a las decisiones

Sistema que ayuda en el proceso de toma de decisiones.
Similares: sistema de soporte a las decisiones
Traducciones posibles: decision support, decision support system
Tema relacionado: Sistemas
Links relacionados: Sistemas de Soporte a Decisiones en Wikipedia

D.1.100 Staff

Personas, tripulación, recursos humanos o conjunto de choferes o conductores de ómnibus.
Similares: crew, driver(s), manpower, human resources, employee(s)
Traducciones posibles: chofer(es), conductor(es), conductora(s), personal, tripulación, empleado(s), flota, cuadrilla, escuadra
Tema relacionado: Transporte
Links relacionados:

D.1.101 Straight shift (*)

Término utilizado en el ámbito de ordenamiento de conductores. Son shifts que contienen descansos ('breaks').
Similares: shift, duty, duties, block(s), task(s), activity, activities, event(s), trip(s), roster(s), service(s)
Traducciones posibles: bloque(s), turno(s), actividad(es), tarea(s), evento(s), pull-in-trip(s), pull-out-trip(s), servicio(s), viaje(s), rotación, rotaciones, servicio(s)
Tema relacionado: Transporte
Links relacionados:

D.1.102 Tabla de horario

Planilla o conjunto de planillas con las actividades, eventos u horarios que cumplen los servicios o el personal.
Similares: diario(s), planilla(s) de servicios, diario(s) de servicios, diario(s) de actividades, libro(s), planilla(s), programación, logística, plan(es), planificación, planeamiento, ordenamiento, programa, servicio(s), horario(s), itinerario
Traducciones posibles: schedule, schedule table(s), activity chart(s), time table, itinerary
Tema relacionado: Transporte
Links relacionados:

D.1.103 Tabla de actividad

Planilla o conjunto de planillas con las actividades, eventos u horarios que cumplen los servicios o el personal.
Similares: diario(s), planilla(s) de servicios, diario(s) de servicios, diario(s) de actividades, libro(s), planilla(s), programación, logística, plan(es), planificación, planeamiento, ordenamiento, programa, servicio(s), horario(s), itinerario
Traducciones posibles: schedule, schedule table(s), activity chart(s), time table, itinerary
Tema relacionado: Transporte
Links relacionados:

D.1.104 Tarea

Evento que puede ser: realizar un viaje útil, o un descanso, o un viaje expreso.
Similares: bloque(s), turno(s), actividad(es), tareas, evento(s), pull-in-trip(s), pull-out-trip(s), servicio(s), viaje(s), rotación, rotaciones, servicio(s)
Traducciones posibles: task(s), activity, activities, event(s), shift, duty, duties, block(s), trip(s), roster(s), service(s)
Tema relacionado: Transporte

Links relacionados:

D.1.105 Task

Evento que puede ser: realizar un viaje útil, o un descanso, o un viaje expreso.
Similares: tasks, activity, activities, event(s), shift, duty, duties, block(s), trip(s), roster(s), service(s)
Traducciones posibles: bloque(s), turno(s), actividad(es), tarea(s), evento(s), pull-in-trip(s), pull-out-trip(s), servicio(s), viaje(s), rotación, rotaciones, servicio(s)
Tema relacionado: Transporte
Links relacionados:

D.1.106 Time table / Time-tabling

Planilla o conjunto de planillas con las actividades, eventos u horarios que cumplen los servicios o el personal.
Similares: schedule, schedule table(s), scheduling, activity chart(s), plan, planning, itinerary
Traducciones posibles: planilla(s) de servicios, diario(s) de servicios, diario(s) de actividades, libro(s), planilla(s), plan(es), programación, planificación, planeamiento, ordenamiento, itinerario
Tema relacionado: Transporte
Links relacionados:

D.1.107 Tipo de viaje

Determina qué tipo de unidad de transporte puede realizarla (un viaje de tipo 1 puede ser realizado por unidades de transporte de tipo 1 o superior, un viaje de tipo 2 puede ser realizado por unidades de transporte de tipo 2 o superior pero no de tipo 1, etc.)..
Similares: tipos de viaje, viaje(s)
Traducciones posibles: travel, trip
Tema relacionado: Transporte
Links relacionados:

D.1.108 Traffic

Fenómeno causado por el flujo de vehículos en una vía, calle o autopista.
Similares: transit, circulation, transport, transportation
Traducciones posibles: tráfico, tránsito, circulación, flujo
Tema relacionado: Transporte
Links relacionados: Ingeniería de tráfico en Wikipedia Tránsito vehicular en Wikipedia

D.1.109 Tráfico

Fenómeno causado por el flujo de vehículos en una vía, calle o autopista.
Similares: tránsito, circulación, flujo
Traducciones posibles: traffic, transit, circulation, transport, transportation
Tema relacionado: Transporte
Links relacionados: Ingeniería de tráfico en Wikipedia Tránsito vehicular en Wikipedia

D.1.110 Transit

Fenómeno causado por el flujo de vehículos en una vía, calle o autopista.
Similares: traffic, circulation, transport, transportation
Traducciones posibles: tráfico, tránsito, circulación, flujo
Tema relacionado: Transporte
Links relacionados: Ingeniería de tráfico en Wikipedia Tránsito vehicular en Wikipedia

D.1.111 Tránsito

Fenómeno causado por el flujo de vehículos en una vía, calle o autopista.
Similares: tráfico, circulación, flujo

Traducciones posibles: traffic, transit, circulation, transport, transportation
Tema relacionado: Transporte
Links relacionados: Ingeniería de tráfico en Wikipedia Tránsito vehicular en Wikipedia

D.1.112 Transport

Traslado de personas o bienes de un lugar a otro. Dentro de esta acepción se incluyen numerosos conceptos, de los que los más importantes son infraestructuras, vehículos y operaciones.
Similares: transportation, urban tansport, urban
Traducciones posibles: transporte, transporte urbano, urbano(s), urbana(s)
Tema relacionado: Transporte
Links relacionados: Transporte en Wikipedia Medio de transporte en Wikipedia Transporte público en Wikipedia

D.1.113 Transporte

Traslado de personas o bienes de un lugar a otro. Dentro de esta acepción se incluyen numerosos conceptos, de los que los más importantes son infraestructuras, vehículos y operaciones.
Similares: transporte urbano, urbano(s), urbana(s)
Traducciones posibles: transport, transportation, urban transport
Tema relacionado: Transporte
Links relacionados: Transporte en Wikipedia Medio de transporte en Wikipedia Transporte público en Wikipedia

D.1.114 Transportation

Traslado de personas o bienes de un lugar a otro. Dentro de esta acepción se incluyen numerosos conceptos, de los que los más importantes son infraestructuras, vehículos y operaciones.
Similares: transport, urban transport
Traducciones posibles: transporte, transporte urbano, urbano(s), urbana(s)
Tema relacionado: Transporte
Links relacionados: Transporte en Wikipedia Medio de transporte en Wikipedia Transporte público en Wikipedia

D.1.115 Trayecto

Conjunto ordenado de esquinas, por las que pasan los coches.
Similares: trayectos, recorrido(s), ruta(s), viaje(s), ruteo
Traducciones posibles: path, route, routing, trajectory, trip, travel
Tema relacionado: Transporte
Links relacionados:

D.1.116 Trayectos

Sistema implementado por la empresa de transporte urbano para asistir a la planificación de servicios.
Similares:
Traducciones posibles:
Tema relacionado: Transporte
Links relacionados:

D.1.117 Trip

Recorrido efectuado por un ómnibus entre dos puntos geográficos de la ciudad. Cada viaje tiene asociado una hora de partida y una hora de llegada.
Similares: trips, travel, duty, duties, block(s), shift(s), task(s), activity, activities, event(s), roster(s), service(s)
Traducciones posibles: viaje(s), bloques, turno(s), actividad(es), tarea(s), evento(s), pull-in-trip(s), pull-out-trip(s), servicio(s), rotación, rotaciones, servicio(s)
Tema relacionado: Transporte
Links relacionados:

D.1.118 Tripulación

Conjunto de conductores.
Similares: cuadrilla, escuadra, empleado(s), chofer(es), conductor(es), conductora(s), personal, equipo, flota
Traducciones posibles: fleet, squad, employee(s), crew, staff, driver(s), manpower, human resources
Tema relacionado: Transporte
Links relacionados:

D.1.119 Turno

Término utilizado en el ámbito de ordenamiento de conductores. Es el trabajo planificado para ser realizado por un chofer en un día.
Similares: bloque(s), turno(s), actividad(es), tarea(s), evento(s), pull-in-trip(s), pull-out-trip(s), servicio(s), viaje(s), rotación, rotaciones, servicio(s)
Traducciones posibles: shift(s), straight shift, duty, duties, block(s), task(s), activity, activities, event(s), trip(s), roster(s), service(s)
Tema relacionado: Transporte
Links relacionados:

D.1.120 Unidad de transporte

Vehículo en el que se realizan los viajes. Tiene un tipo que determina el tipo de viaje que puede hacer.
Similares: ómnibus, unidad(es) de transporte, coche(s), vehículo(s), flota, cuadrilla, escuadra
Traducciones posibles: bus, vehicle(s), fleet
Tema relacionado: Transporte
Links relacionados: Autobús en Wikipedia Medio de transporte en Wikipedia

D.1.121 Urban

Que ocurre dentro de una ciudad.
Similares: urban transport, transport, transportation
Traducciones posibles: urbano(s), urbana(s), transporte urbano
Tema relacionado: Transporte
Links relacionados: Transporte público en Wikipedia

D.1.122 Urbano

Que ocurre dentro de una ciudad.
Similares: urbanos, urbana(s), transporte urbano
Traducciones posibles: urban, urban transport, transport, transportation
Tema relacionado: Transporte
Links relacionados: Transporte público en Wikipedia

D.1.123 Usabilidad

Facilidad o nivel de uso, grado en el que el diseño de un objeto facilita o dificulta su manejo.
Similares: accesibilidad, accesibilidad Web, interacción, interfaz, interfaces, interfaz gráfica
Traducciones posibles: usability, usefulness, accesibility, interaction, interface, graphic interface
Tema relacionado: Interfaces
Links relacionados: Usabilidad en Wikipedia

D.1.124 User-centered design (UCD)

Proceso de diseño en el que las necesidades, los deseos y las limitaciones del usuario final de una interfaz o documento toman una atención y relevancia considerable en cada nivel del proceso de diseño.
Similares: design, interface, interaction
Traducciones posibles: diseño centrado en el usuario, interacción, interfaz, interfaces, interfaz gráfica
Tema relacionado: Interfaces
Links relacionados: Diseño centrado en el usuario en Wikipedia

D.1.125 Usuario

Persona que utiliza un sistema o aplicación.
Similares: usuarios
Traducciones posibles: user
Tema relacionado: Interfaces, Sistemas
Links relacionados:

D.1.126 Usuario Planificador

Persona que utiliza un sistema o aplicación y que posee conocimientos de planificación de vehículos.
Similares: usuarios
Traducciones posibles: user, planner(s)
Tema relacionado: Interfaces, Sistemas
Links relacionados:

D.1.127 Usuario Técnico/Desarrollador/Programador

Persona que utiliza un sistema o aplicación y que posee conocimientos de programación.
Similares: usuarios
Traducciones posibles: user, developer(s), programmer(s)
Tema relacionado: Interfaces, Sistemas
Links relacionados:

D.1.128 Vehicle

Medio de transporte en el que se realizan los viajes. Tiene un tipo que determina el tipo de viaje que puede hacer.
Similares: bus, vehicles, fleet
Traducciones posibles: ómnibus, unidad(es) de transporte, coche(s), vehículo(s), flota, cuadrilla, escuadra
Tema relacionado: Transporte
Links relacionados: Autobús en Wikipedia Medio de transporte en Wikipedia

D.1.129 Vehicle Routing Problem (VRP)

Problema consistente en averiguar las rutas de una flota de transporte para dar servicio a unos clientes.
Similares: routing
Traducciones posibles: ruta(s), vehículo(s), Problema de ruteo de vehículos, Problema de rutas de vehículos, ruteamiento
Tema relacionado: Transporte
Links relacionados: Problema de rutas de vehículos en Wikipedia

D.1.130 Vehicle Scheduling Problem (VSP)

Problema consistente en averiguar la disposición en tiempo de las unidades de transporte con objetivos definidos (como la optimización en el uso de dichos recursos, la reducción de costos, etc.). Problema de asignación de unidades de transporte a tareas en el tiempo.
Similares: CSP (Crew Scheduling Problem), ISP (Integrated Scheduling Problem), IVSP (Integrated Vehicle Scheduling Problem), administration, management, manage, logistics, scheduling, schedule, itinerary, pairing, plan, planning, time table, time-tabling, bus, vehicle(s), fleet
Traducciones posibles: problema de planificación de vehículos, problema de planificación de personal, problema integral de planificación de vehículos, manejo, administrar, organización, organizar, ordenar, orden, ordenamiento, asignación, logística, planeamiento, planificación, plan(es), horario(s), tabla(s) de horario, itinerario, unidad(es) de transporte, vehículo(s), flota, cuadrilla, escuadra
Tema relacionado: Transporte, Sistemas

D.1.131 Vehículo

Medio de transporte en el que se realizan los viajes. Tiene un tipo que determina el tipo de viaje que puede hacer.
Similares: ómnibus, unidad(es) de transporte, coche(s), vehículo(s), flota, cuadrilla, escuadra
Traducciones posibles: bus, vehicle(s), fleet

Tema relacionado: Transporte
Links relacionados: Autobús en Wikipedia Medio de transporte en Wikipedia

D.1.132 Viaje

Recorrido efectuado por un ómnibus entre dos puntos geográficos de la ciudad. Es una instancia de una línea. Cada viaje tiene asociado una hora de partida y una hora de llegada.
Similares: viajes, bloques, turno(s), actividad(es), tarea(s), evento(s), pull-in-trip(s), pull-out-trip(s), servicio(s), rotación, rotaciones, servicio(s)
Traducciones posibles: trip(s), travel, duty, duties, block(s), shift(s), task(s), activity, activities, event(s), roster(s), service(s)
Tema relacionado: Transporte
Links relacionados:

D.1.133 Viaje asignado

Es un tipo de viaje. Viaje que pertenece a un servicio.
Similares: viajes asignados, viaje(s), viaje(s) compatible(s), bloques, turno(s), actividad(es), tarea(s), evento(s), pull-in-trip(s), pull-out-trip(s), servicio(s), rotación, rotaciones, servicio(s), asignado(s), asignada(s), asignaciones, asignar, empatar, emparejar, tupla, dupla, ordenamiento, organización, plan(es), planeamiento, planificación, programa, programación, rotación, rotaciones
Traducciones posibles: trip(s), duty, duties, block(s), shift(s), task(s), activity, activities, event(s), roster(s), service(s), assignment, assigned, assign, schedule, scheduling, pairing, plan, planning, roster, rostering
Tema relacionado: Transporte
Links relacionados:

D.1.134 Viaje compatible

Es un tipo de viaje. Dos viajes son compatibles si es posible realizar uno después del otro utilizando un mismo vehículo. Por ejemplo, dos viajes que se superponen en el tiempo no son compatibles entre si. Dos viajes son compatibles si la hora de llegada del primero y la hora de salida del segundo pueden acomodarse a las reglas del problema (que no se superpongan, que dejen suficiente tiempo para el descanso del personal, o para poder llegar al lugar de origen del segundo, etc.).
Similares: viajes compatibles, viajes, viaje(s) asignado(s), bloques, turno(s), actividad(es), tarea(s), evento(s), pull-in-trip(s), pull-out-trip(s), servicio(s), rotación, rotaciones, servicio(s), asignado(s), asignada(s), asignaciones, asignar, empatar, emparejar, tupla, dupla, ordenamiento, organización, plan(es), planeamiento, planificación, programa, programación, rotación, rotaciones
Traducciones posibles: compatible trip, trip(s), duty, duties, block(s), shift(s), task(s), activity, activities, event(s), roster(s), service(s), assignment, assigned, assign, schedule, scheduling, pairing, plan, planning, roster, rostering
Tema relacionado: Transporte
Links relacionados:

D.1.135 Viaje expreso

Es un tipo de viaje. No tienen fines económicos porque no recoge pasajeros, son viajes que le generan pérdidas a la empresa. Existen para poder trasladar un ómnibus del garage al lugar de salida del primer viaje útil, del lugar de destino del último viaje útil al garage y para llegar al lugar de salida de cualquier viaje útil que tenga asignado desde donde esté.
Similares: viajes expresos, línea(s), viajes asignados, viaje(s), viaje(s) compatible(s), bloques, turno(s), actividad(es), tarea(s), evento(s), pull-in-trip(s), pull-out-trip(s), servicio(s), rotación, rotaciones, servicio(s), asignado(s), asignada(s), asignaciones, asignar, empatar, emparejar, tupla, dupla, ordenamiento, organización, plan(es), planeamiento, planificación, programa, programación, rotación, rotaciones
Traducciones posibles: non-stop trip, bus line(s), trip(s), duty, duties, block(s), shift(s), task(s), activity, activities, event(s), roster(s), service(s), assignment, assigned, assign, schedule, scheduling, pairing, plan, planning, roster, rostering
Tema relacionado: Transporte
Links relacionados:

D.1.136 Viaje libre

Viaje que se debe realizar pero que no está asignado a ningún servicio.
Similares: viajes libres, viaje(s) no asignado(s), viaje(s) asignado(s), viaje(s), viaje(s) compatible(s), bloques, turno(s), actividad(es), tarea(s), evento(s), pull-in-trip(s), pull-out-trip(s), servicio(s), rotación, rotaciones, servicio(s), asignado(s), asignada(s), asignaciones, asignar, empatar, emparejar, tupla, dupla, ordenamiento, organización, plan(es), planeamiento, planificación, programa, programación, rotación, rotaciones

Traducciones posibles: unassigned trip(s), trip(s), duty, duties, block(s), shift(s), task(s), activity, activities, event(s), roster(s), service(s), assignment, assigned, assign, schedule, scheduling, pairing, plan, planning, roster, rostering
Tema relacionado: Transporte
Links relacionados:

D.1.137 Viaje útil

Viaje que genera ganancia porque recoge pasajeros y cumple con los horarios y destinos estipulados.
Similares: viaje(s) asignado(s), viaje(s), viaje(s) compatible(s), viaje(s) libre(s), viaje(s) expreso(s), bloque(s), turno(s), actividad(es), tarea(s), evento(s), pull-in-trip(s), pull-out-trip(s), servicio(s), rotación, rotaciones, servicio(s), asignado(s), asignada(s), asignaciones, asignar, empatar, emparejar, tupla, dupla, ordenamiento, organización, plan(es), planeamiento, planificación, programa, programación, rotación, rotaciones
Traducciones posibles: trip(s), duty, duties, block(s), shift(s), task(s), activity, activities, event(s), roster(s), service(s), assignment, assigned, assign, schedule, scheduling, pairing, plan, planning, roster, rostering
Tema relacionado: Transporte
Links relacionados:

D.1.138 Viático

Conjunto de provisiones para un viaje.
Similares:
Traducciones posibles: expenses, travel allowance
Tema relacionado: Transporte
Links relacionados:

D.1.139 Visión

Sentido que consiste en la habilidad de detectar la luz y de interpretarla (ver).
Similares: visualización, percepción, interfaz, interfaces
Traducciones posibles: visualization, perception
Tema relacionado: Interfaces
Links relacionados: Visión en Wikipedia

D.1.140 Visualización

Transformación de datos científicos pero abstractos en imágenes.
Similares: visión, percepción, interfaz, interfaces, interfaz gráfica
Traducciones posibles: visualization, perception, vision
Tema relacionado: Interfaces
Links relacionados: Visualizacón científica en Wikipedia

D.1.141 Traducciones de términos más comunes a diferentes idiomas

dominio	español	alemán	inglés	portugués	francés	italiano
sistemas	calidad	qualität	quality	qualidade	qualité	qualitá
	decisión	entscheidung / entscheidungsgrundlage	decision	decisão	décision	decisione
	manejo / gestión / administración	verwaltung	management	administração / gestão	direction / gestion	amministrazione/ direzione / dirigenza / gestione
	planeamiento / logística	planerisch / planung / logistik / verpflegungs	planning / logistics	planejamento / planificação	planning	progettazione / pianificazione

	programación / programa / horario / itinerario / planificación / tabla de horario / asignación / tarea / diario / libro / planilla	vorbereitung / planung / dienst / tabelle / liste	scheduling / schedule / time-tabling / time-table / task / tasking / assignment / duty / itinerary	planificação / horário / itinerário / diário de atividade	horaire / programme	programmazione / orario / piano / programma / tabella di marcia
	soporte / asistencia	betreuung	support	support	assistance / soutien	supporto / assistenza
	usuario	anwender / benutzer	user	usuário / utente	utilisateur / usager	utilizzatore / utente
transporte	depósito	depot	depot	depósito	dépôt	deposito
	flota / personal / tripulación / recursos humanos / manpower / empleado / conductor / chofer	mannschaft / besatzung / busfahrer / crew	crew / staff / human resources / employee / driver	tripulação / frota / empregado	personnel	equipaggio
	ómnibus	bus	bus / autobus	ônibus	bus	bus / autobus
	público	publik / publikum	public	público	publique	pubblico
	rotación / turno / asignación	stammrolle	roster / rostering			
	ruteo	leitwegsteuerung / straße	routing	roteirização	routage	rottamento
	tráfico	verkehr	traffic	tráfego	circulation	traffico
	transporte	verkehrsmittel / transportmittel	transport / transportation	transporte	transport	transporto
	urbano	stadt / städtisch	urban	urbano	urbain	urbano / cittadino
	vehículo	fahrzeug	vehicle	veículo	véhicule	veicolo
	viaje	reise / fahrt / bewegung	travel	viagem (viagens)	voyage	viaggio
interfaces	computadora	computer / rechner	computer	computador	ordinateur	computer
	diseño	gestaltung / ausführung	design	desenho	dessin	disegno / piano / progetto
	gráfica	grafik / grafische / grafische benutzeroberfläche	graphic	gráfico	graphique	grafico
	humano	mensch / personalabteilung / human	human	humano	humain	umano
	interacción	wechselwirkung / interaktive / interaktion	interaction	interação	interaction	interazione
	interface / interfaz	schnittstelle / benutzeroberfläche / anwendungsschnittstelle	interface	interface	interface	interfaccia
	percepción	wahrnehmung	perception	percepção	perception	percezione
	psicología	psychologie	psychology	psicologia	psycologie	psicologia
	visión		vision	visão	vision	visione
	visual		visual	visual	visuel	visivo

	visualización	sichtbarmachung / veranschaulichung / vorstellungskonzept	visualization	visualização	visualisation	visualizzazione

D.2 Autores

En esta sección se encuentran los enlaces a las páginas con información sobre los principales autores de papers, libros y materiales tomados como referencia a lo largo del proyecto. Los autores se encuentran ordenados alfabéticamente. El objetivo es hacer un chequeo de los autores para corroborar sus aportes al área y evaluar sus contribuciones. También para tener acceso a material recientemente publicado por los mismos.

También se proporciona para cada autor:

- *Tema relacionado*: indica que dicho autor se especializa en material que fue utilizado en su mayoría en los informes relacionados a un tema específico que puede ser "Heurísticas", "Optimización", "Transporte", "Interfaces" o "Sistemas";
- *Personas relacionadas*: son otros investigadores a los que está asociado el autor.

A continuación se dan los enlaces correspondientes.

D.2.1 A. Colorni

Tema relacionado: Heurísticas, Optimización
Personas relacionadas:

D.2.2 Alain Hertz

http://www.gerad.ca/~alainh/
Tema relacionado: Heurísticas, Optimización
Personas relacionadas:

D.2.3 Alan Dix

http://www.comp.lancs.ac.uk/~dixa/
http://en.wikipedia.org/wiki/Alan_Dix
http://www.alandix.com/blog/
Tema relacionado: Interfaces
Personas relacionadas:

D.2.4 Alison J. Head

http://www.ajhead.com/
http://www.boxesandarrows.com/person/1270-alison
Tema relacionado: Interfaces (diseño)
Personas relacionadas:

D.2.5 Amadeo/Amedeo R. Odoni

http://web.mit.edu/aeroastro/www/people/odoni/bio.html
http://transportation.section.informs.org/ANNOUNCEMENTS/amedeo.odoni.html
Tema relacionado: Transporte
Personas relacionadas:

D.2.6 Andrea Roli

http://www.lia.deis.unibo.it/~aro/
Tema relacionado: Heurísticas, Optimización
Personas relacionadas:

D.2.7 Andreas Löbel

http://www.zib.de/loebel/
Tema relacionado: Transporte
Personas relacionadas:

D.2.8 Belén Melián

http://webpages.ull.es/users/mbmelian/
Tema relacionado: Heurísticas, Optimización
Personas relacionadas:

D.2.9 Ben Shneiderman

http://www.cs.umd.edu/~ben/
http://www.cs.umd.edu/hcil/members/bshneiderman/umlpapers/
http://en.wikipedia.org/wiki/Ben_Shneiderman
Tema relacionado: Interfaces
Personas relacionadas:

D.2.10 Bruce L. Golden

http://www.rhsmith.umd.edu/faculty/bgolden/
Tema relacionado: Heurísticas, Optimización
Personas relacionadas:

D.2.11 Carol Righi

http://www.taskz.com/carol_righi.php
Tema relacionado: Interfaces (user-centered design)
Personas relacionadas:

D.2.12 Catherine C. McGeoch

http://www.cs.amherst.edu/ccm/
Tema relacionado: Heurísticas, Optimización
Personas relacionadas:

D.2.13 Celso C. Ribeiro

http://www-di.inf.puc-rio.br/~celso/grupo_de_pesquisa.htm
Tema relacionado: Heurísticas, Optimización
Personas relacionadas:

D.2.14 Edward J. Anderson

http://www.edanderson.info/

Tema relacionado: Heurísticas, Optimización
Personas relacionadas:

D.2.15 Fred Glover

http://spot.colorado.edu/~glover/
Tema relacionado: Heurísticas, Optimización
Personas relacionadas:

D.2.16 George Gray

Tema relacionado: Transporte
Personas relacionadas:

D.2.17 Gianni Di Caro

http://www.idsia.ch/~gianni/
Tema relacionado: Heurísticas, Optimización
Personas relacionadas:

D.2.18 Gregory Abowd

http://www.gregoryabowd.com/
http://www.cc.gatech.edu/~abowd/
http://www-static.cc.gatech.edu/~abowd/research/research-statement.html
Tema relacionado: Interfaces
Personas relacionadas:

D.2.19 Guy Desaulniers

http://www.crt.umontreal.ca/~guyd/gdeng.html
Tema relacionado: Transporte
Personas relacionadas:

D.2.20 Ibrahim H. Osman

http://sb-lb.aub.edu.lb/PersonalSites/DrOsman/index.html
Tema relacionado: Heurísticas, Optimización
Personas relacionadas:

D.2.21 Isabel Branco

Tema relacionado: Transporte
Personas relacionadas:

D.2.22 J. Marcos Moreno Vega

http://webpages.ull.es/users/jmmoreno/
Tema relacionado: Heurísticas, Optimización
Personas relacionadas:

D.2.23 Jakob Nielsen

http://es.wikipedia.org/wiki/Jakob_Nielsen
http://www.useit.com/

http://www.useit.com/jakob/
Tema relacionado: Interfaces (usabilidad)
Personas relacionadas:

D.2.24 José A. Moreno Pérez

http://webpages.ull.es/users/jamoreno/
Tema relacionado: Heurísticas, Optimización
Personas relacionadas:

D.2.25 Jacques Desrosiers

http://www.crt.umontreal.ca/~jacques/jdeng.html
http://neumann.hec.ca/pages/jacques.desrosiers/eng.html
Tema relacionado: Transporte
Personas relacionadas:

D.2.26 James P. Kelly

Tema relacionado: Heurísticas, Optimización
Personas relacionadas:

D.2.27 Janet Finlay

Tema relacionado: Interfaces
Personas relacionadas:

D.2.28 Jean Marc Rousseau

http://www.ustran.com/presentacion/jeanmarc.html
http://www.cirano.qc.ca/cv/court/1529_en.html
Tema relacionado: Transporte
Personas relacionadas:

D.2.29 Jesse James Garrett

http://en.wikipedia.org/wiki/Jesse_James_Garrett
http://www.boxesandarrows.com/person/5-jessejamesgarrett
http://www.elmundo.es/navegante/2006/05/29/entrevistas/1148920092.html
Tema relacionado: Interfaces (user-centered design)
Personas relacionadas:

D.2.30 Joachim R. Daduna

Tema relacionado: Transporte
Personas relacionadas:

D.2.31 José M. Pinto Paixão

Tema relacionado: Transporte
Personas relacionadas:

D.2.32 Joseph Leung

http://web.njit.edu/~leung/

Tema relacionado: Transporte
Personas relacionadas:

D.2.33 Karel Vredenburg

http://www.interaction-design.org/references/authors/karel_vredenburg.html
http://karelvredenburg.com/
Tema relacionado: Interfaces (user-centered design)
Personas relacionadas:

D.2.34 Lester Hoel

http://cts.virginia.edu/Hoel.htm
http://unjobs.org/authors/lester-a.-hoel
Tema relacionado: Transporte
Personas relacionadas:

D.2.35 Manuel Laguna

http://leeds-faculty.colorado.edu/laguna/
Tema relacionado: Heurísticas, Optimización
Personas relacionadas:

D.2.36 Marc Desrochers

Tema relacionado: Transporte
Personas relacionadas:

D.2.37 Marco Dorigo

http://iridia.ulb.ac.be/~mdorigo/HomePageDorigo/index.php
Tema relacionado: Heurísticas, Optimización
Personas relacionadas:

D.2.38 Mark D. Hickman

http://www.u.arizona.edu/~mhickman/
Tema relacionado: Transporte
Personas relacionadas:

D.2.39 Martin Helander (biblia)

http://www.ntu.edu.sg/home/martin/
Tema relacionado: Interfaces
Personas relacionadas:

D.2.40 Mauricio G. C. Resende

http://www.research.att.com/~mgcr/
Tema relacionado: Heurísticas, Optimización
Personas relacionadas:

D.2.41 Michael G.H. Bell

http://www.cts.cv.ic.ac.uk/html/Staff/staffDetails.asp?id=MGHB

Tema relacionado: Transporte
Personas relacionadas:

D.2.42 Nenad Mladenovic

http://www.brunel.ac.uk/about/acad/siscm/maths/people/acad/NenadMladenovic
Tema relacionado: Heurísticas, Optimización
Personas relacionadas:

D.2.43 Pierre Hansen

http://www.hec.ca/en/profs/pierre.hansen.html
Tema relacionado: Heurísticas, Optimización
Personas relacionadas:

D.2.44 Prasad V. Prabhu (biblia)

Tema relacionado: Interfaces
Personas relacionadas:

D.2.45 Rafael Martí

http://www.uv.es/rmarti/
Tema relacionado: Heurísticas, Optimización
Personas relacionadas:

D.2.46 Richard S. Barr

http://faculty.smu.edu/barr/
Tema relacionado: Heurísticas, Optimización
Personas relacionadas:

D.2.47 Russell Beale

http://www.cs.bham.ac.uk/~rxb/
Tema relacionado: Interfaces
Personas relacionadas:

D.2.48 Scott Isensee

http://members.aol.com/isensee/
http://www.taskz.com/scott_isensee.php
Tema relacionado: Interfaces (user-centered design)
Personas relacionadas:

D.2.49 Stefan Voss/Voß

http://iwi.econ.uni-hamburg.de/IWIWeb/Default.aspx?tabid=1010
Tema relacionado: Transporte
Personas relacionadas:

D.2.50 Thomas K. Landauer

http://en.wikipedia.org/wiki/Thomas_Landauer
http://www.cs.colorado.edu/courses/schedules/thomas_landauer.html

http://www.pearsonkt.com/bioLandauer.shtml
Tema relacionado: Interfaces
Personas relacionadas:

D.2.51 William H. K. Lam

http://www.cse.polyu.edu.hk/~cehklam/
Tema relacionado: Transporte
Personas relacionadas:

D.2.52 William R. Stewart

Tema relacionado: Heurísticas, Optimización
Personas relacionadas:

D.3 Bibliografía

En esta sección se presenta una lista de bibliografía que fue consultada como parte del estudio del área del transporte. http://home.arcor.de/Emden-Weinert/crew-scheduling+staff-rostering.html Último acceso: 20/11/2010

D.3.1 Lista de papers de algoritmos y optimización

Link: http://www.asap.cs.nott.ac.uk/publications/old/book-chapters.shtml
Último acceso: 27/11/2009

D.3.2 Lista de papers de grupos y personas que trabajan en optimización

Link: http://www.lancs.ac.uk/staff/letchfoa/co_people.htm
Último acceso: 27/11/2009

D.3.3 Lista de grupos, proyectos, herramientas y personas que trabajan en planificación y ordenamiento

Link: http://thomas.emden-weinert.de/crew-scheduling+staff-rostering.html
Último acceso: 27/11/2009

D.3.3.1 Links relacionados (Último acceso: 27/11/2009)

- Vehicle Routing (Tim Duncan); here is a local copy of the page as of 6/12/97.
- Transportation Resources (Princeton)
- Traffic Planning & Engineering, Transportation Links (Lund)
- Transportlinks (Jürgen Schenker, FH Heilbronn)
- The Transport Web - the Information Service for the Transport Industry (Bristol)
- Links to Logistics and Transport (Cranfield)
- Global Technology Base: Transportation
- Transportation Research Board (US NAS)
- Virtual Logistics Directory (Lee Shore Enterprises Ltd.)
- Yahoo! - Transportation
- WWW sites related to transport research (Y.L.Siu, Leeds)
- Aviation - World Wide Web Virtual Library
- Aviation human performance factors & crew resource management Web Site

- SINTA - Special Interest Network for Transport in Australia
- ITS Cooperative Deployment Network
- Workindex.com
- HR - Info

D.3.3.2 Grupos y proyectos de investigación (Último acceso: 27/11/2009)

- CRT - Center for Research on Transportation (Montreal)
- GERAD (Montreal)
- Operations Research Group at CSIRO Mathematical and Information Sciences (Melbourne / Adelaide / Canberra)
- INRETS - French Nation. Research Inst. Transport & Safety
- MIT Center for Transportation Studies
- Flugplanung (ZAIK, Köln)
- Helmut Baumgarten (TU Berlin)
- Institute for Transport Studies (Leeds)
- Scheduling and Constraint Management (Leeds)
- Rail Transportation Planning (Zimmermann, Braunschweig)
- Vehicle Scheduling in Handicapped People's Transport (ZIB)
- Multiple Depot Vehicle Scheduling in Public Mass Transit (ZIB)
- Leena Suhl (Paderborn)
- AUSIAS: Transport Telematics (Brunel)
- ESPRIT Project SuperbuS: Public Transport (Brunel)
- EU Transport RTD Program
- Intermodal Passenger Transport
- PAROS - Parallel large scale automatic scheduling
- PARROT - Parallel Crew Rostering
- Duty Scheduling in Public Transportation (ZIB Berlin, IVU, HanseCom)
- DISSY - Driver Scheduling System for Public Transport (HUB, VSS, BSAG)
- Air-crew scheduling (IBM Tokyo Research Lab)
- Crew Scheduling (Georgia Tech)
- Automated Scheduling and Planning Group (Nottingham)
- Verfahren zur Optimierung des Mitarbeitereinsatzes am Arbeitsplatz (Prof. Kleinschmidt, Passau)
- PAREO - Parallel processing in operation research
- Railway Research Group

D.3.3.3 Software & Benchmarks (Último acceso: 27/11/2009)

Sistema de software	Enfoque	Proveedor
AirCrews™	Airline	Sabre Inc.
	Airline	Carmen Systems
Cygnus, NOVA	Airline	Mercury Scheduling Inc.
IFES, DIES, REDI 2	Public Transit	Häni-Prolektron AG
Hastus	Public Transit	GIRO Inc.
HOT II, OPUS	Public Transit	HanseCom GmbH
MICROBUS, BERTA	Public Transit	IVU
Cityplan21	Public Transit	TTI Systeme
DIVA	Public Transit	Mentz
DISSY	Public Transit	VSS GmbH / BSAG
TR System	Public Transit	TR-Partner A/S
Vehicle Routing - Commercial Software		

BUSCH, OPCrew	public transport, rail	opcom
MICROSTER Personnel Rostering System		Accolade
Roster		Time & People Australia
ShiftTrack		Open Wave (Melbourne, New York)
AnnoPlan		
EasyStaff		Shibutzit Software Development LTD., FASTECH
SCAT, EWS	Railway, Plants	PS Technology, Inc (Boulder, CO)
Officer Scheduling		InTime Solutions Inc. (Burnaby, British Columbia)
WorkFORCE		Adaptiv
People Scheduler		IMSI
ScheduleSoft		ScheduleSoft Corporation
Visual Staff Scheduler PRO 4.0		ABS - Atlas Business Solutions Inc
ESCALAS, CREWS_NS	Rail Transport	SISCOG
SP-Expert		ASTRUM, ERlangen
Alpha Computer (Chemnitz)	Hospitals	
ACCURAT		AC-Service, Stuttgart
		AGP Standard-Software GmbH, Münster
		InVision Software GmbH
		ISGUS Homepage
		MBB Gelma Hengstler
		perbit Software GmbH
		ATOSS® Software AG
		ADICOM Unternehmensgruppe
		H.R. Management Software GmbH
		MHG Firmengruppe
Paisy		ADP
	Zeiterfassung	Interflex
Schichtplaner		
Pflegedienst 2000		COMfuture GmbH
ALTER Zeitwirtschaft - PZW PC		Alter
Complex	Hospitals	
Schichtplanassistent	Industrial Production	XIMES
MiniMatching		MiniMatching

D.3.3.4 Benchmarks (Último acceso: 27/11/2009)

- Planning & Scheduling Benchmarks
- Benchmarks for the Capacitated Vehicle Routing Problem
- Set partitioning instances from airline crew scheduling problems (K. Hoffman);
- OR-LIBRARY test data sets (J.E. Beasley)

D.3.4 Organizaciones y compañías

Último acceso: 27/11/2009

- Ad OPT Technologies Inc.
- AGIFORS
- ARRIVA
- Austin Analytics
- IATA
- Transportation Companies
- Adtranz
- Airports and Airlines
- Carmen Systems AB
- Circadian Technologies, Inc. (Cambridge, MA)
- GIRO Inc.(Le Groupe en Informatique et Recherche Opérationnelle, Montréal)
- HaCon Ingenieurges. mbH
- HanseCom
- ILOG
- Intranetz - Gesellschaft für Informationslogistik
- IVU GmbH
- MagicLogic Optimization Inc.(Vancouver)
- PTV - Planungsbüro Transport und Verkehr GmbH (Karlsruhe)
- SABRE Decision Tech.
- TLC - Transport-, Informatik- und Logistik-Consulting GmbH
- Airborne Express
- Burlington Air Express
- Danzas
- DPD Deutscher Paketdienst
- DHL Worldwide Express
- EMS Kurierpost
- Federal Express
- GP General Parcel Logistics
- KLM Cargo
- Kühne & Nagel
- Lufthansa Cargo AG
- Lufthansa Systems
- DB Cargo
- Omnibus Systems
- Post AG
- RGW-Express
- Schenker International
- Shiftwork Services
- Stagecoach
- Trans-o-flex Schnell-Lieferdienst
- Trans Tec
- United Parcel Service

D.4 Referencias

1. *Computer-aided scheduling of public transport*. International Conference on Computer-aided Scheduling of Public Transport CASPT 2000: Springer. p. 466, 2000. isbn 3540422439.
2. *Engenharia de tráfego e transportes 2000: avanços para uma era de mudanças*. Congresso Panamericano de Engenharia de Transito e Transporte: ANPET. p., 2000. isbn 85-87893-01-7.
3. *Graphical User Interface Gallery http://toastytech.com/guis/index.html Último acceso: 20/12/2009*.
4. *Interacción persona-computador http://es.wikipedia.org/wiki/Interacci%C3%B3n_persona-computador Último acceso: 20/12/2009*.
5. *Programming Language Comparison http://www.jvoegele.com/software/langcomp.html Último acceso: 20/12/2009*.
6. *User Interface Prototypes http://www.agilemodeling.com/artifacts/uiPrototype.htm Último acceso: 20/12/2009*.
7. *Why Java's Hot http://www.ibiblio.org/java/books/jdr/chapters/01.html Último acceso: 20/12/2009*.
8. *Wordle - Create http://www.wordle.net/create Último acceso: 20/12/2009*.
9. Andriole, S., *Storyboard prototyping: a new approach to user requirements analysis*: QED information sciences. p. 280, 1989. isbn 0-89435-246-6.
10. Balci, O., R. Sharda, and S. Zenios, *Computer science and operations research: new developments in their interfaces*: Pergamon. p. 536, 1992. isbn 0-08-040806-0.
11. Baldonado, M.Q.W., A. Woodruff, and A. Kuchinsky, *Guidelines for Using Multiple Views in Information Visualization.*
12. Blum, C., *Hybrid metaheuristics: an emerging approach to optimization*: Springer. p. 289, 2008. isbn 9783540782940.
13. Borndröfer, R., M. Grötschel, and M.E. Pfetsch, *Public transport to the fORe!* 2006.
14. Brasileiro, A. and W.K. Junior, *Panorama nacional da pesquisa em transportes 2004*. Congresso de Pesquisa e Ensino em Transportes: ANPET. p. 2 vol. + CD-ROM, 2004. isbn 8587893-09-2 (VOL. 1) y 8587893-10-6 (VOL. 2).
15. Cancela, H. and M. Urquhart, *Simulación del Transporte colectivo urbano*, INCO, PEDECIBA Informática. 1999.
16. Carroll, J.M., *Human-computer interaction: psychology as a science of design.* Int. J. Human-Computer Studies. 46, 1996.
17. Ceder, A. *Urban Transit Scheduling: Framework, Review and Examples*. in *JOURNAL OF URBAN PLANNING AND DEVELOPMENT*. 2002.
18. Daduna, J., I. Branco, and J.P. Paixao, *Computer-aided transit scheduling*. International Conference on Computer-aided Scheduling of Public Transport: Springer. p. 374, 1993. isbn 3-540-60193-7.
19. Dasgupta, P., P.P. Chakrabarti, and S.C. DeSarkar, *Multiobjective heuristic search:an introduction to intelligent search methods for multicriteria optimization*: Vieweg. p. 134, 1999. isbn 3-528-05708-4.
20. Deb, K., *Multi-objective optimization using evolutionary algorithms*: Wiley. p. 497, 2001. isbn 047187339X.
21. Desaulniers, G. and M.D. Hickman, *Public Transit*: Les Cahiers du GERAD G-2003-77. 2003.
22. Dumas, J. and J.C. Redish, *A practical guide to usability testing*: Intellect. p. 404, 1999. isbn 1841500208.
23. Duyne, D.V., J. Landay, and J. Hong, *The design of sites: patterns, principles and processes for crafting a customer-centered web experience*: Addison Wesley. p. 762, 2003. isbn 020172149X.
24. Fernandez, E. and T. Laurenzo, *Apuntes del curso Interacción Persona-Computadora http://www.fing.edu.uy/inco/cursos/inpercom/index-2009.html Último Accesso: 10/03/2010.* 2009.
25. Fowler, S., *GUI design handbook*: McGraw-Hill. p. 318, 1998. isbn 0-07-059274-8.
26. French, S., *Sequencing and scheduling: an introduction to the mathematics of the job-shop*: Horwood. p. 246, 1990. isbn 0-13-806365-6.
27. Gamma, E., *Design patterns:elements of reusable object-oriented software*: Addison-Wesley. p., 1995. isbn 0-201-63361-2.
28. Garrett, J.J., *The elements of user experience: user-centered design for the web*: New Riders. p. 189, 2002. isbn 0735712026.
29. Goldberg, D.E., *Genetic algorithms in search, optimization, and machine learning*: Addison Wesley. p. 412, 1989. isbn 0201157675.

30. Gruttner, E., et al. *Recorridos óptimos de líneas de transporte público usando algoritmos genéticos*. in *Jornadas Chilenas de Computación*. 2002.
31. Hahn, J. and J. Kim, *Why Are Some Diagrams Easier to Work With? Effects of Diagrammatic Representation on the Cognitive Integration*
Process of Systems Analysis and Design.
32. Hewett, B., Card, Carey, Gasen, Mantei, Perlman, Strong y Verplank, *HCI Bibliography: Human-Computer Interaction Resources* http://www.hcibib.org/ *Último acceso: 20/12/2009*. 1992.
33. Hooker, J., *Logic based methods for optimization: combining optimization and constraint satisfaction*: Wiley. p. 495, 2000. isbn 0471385212.
34. John, B.E. and D.E. Kieras, *The GOMS Family of User Interface Analysis Techniques: Comparison and Contrast.* 1996.
35. John, B.E. and D.E. Kieras, *Using GOMS for User Interface Design and Evaluation: Which Technique?* 1996.
36. Kim, J., J. Hahn, and F.J. Lerch, *HOW IS THE DESIGNER DIFFERENT FROM THE USER? - FOCUSING ON A SOFTWARE DEVELOPMENT METHODOLOGY -.* Proceedings of the 7th Empirical Studies of Programmers. 1997.
37. Leung, J.Y.-T., *Handbook of scheduling: algorithms, models and performance analysis*: Chapman and Hall. p., 2004. isbn 1584883979.
38. Mauttone, A., H. Cancela, and M. Urquhart, *Diseño y optimización de rutas y frecuencias en el transporte colectivo urbano: modelos y algoritmos*. 2003.
39. Mayhew, D., *The usability engineering lifecycle: a practitioner´s handbook for user interface design*: Morgan Kaufmann. p. 542, 1999. isbn 1558605614.
40. Meneghin, A. and A. Ferraz-Leite, *Interfaz Gráfica para Herramienta de Planificación de Transporte Público Urbano Colectivo.* 2003.
41. Miller, G.A., *The Magical Number Seven, Plus or Minus Two: Some Limits on Our Capacity for Processing Information.* The Psychological Review. 63, 1956.
42. Myers, B.A., *A Brief History of Human Computer Interaction Technology* http://www.cs.cmu.edu/~amulet/papers/uihistory.tr.html *Último acceso: 20/12/2009*, ACM interactions. Vol. 5, no. 2. 1998.
43. Nielsen, J., *Usabilidad: diseño de sitios web*: Prentice Hall. p. 416, 2000. isbn 84-205-3008-5.
44. Noyes, J. and C. Baber, *User-centered design of systems*: Springer. p. 222, 1999. isbn 3-540-76007.
45. Odoni, A.R., Rousseau, and Wilson, *Handbook in OR and MS: OR and the Public Sector*. 1994.
46. Pursula, M. and J. Nittymäki, *Mathematical methods on optimization in transportation systems*: Kluwer. p. 247, 2001. isbn 0-7923-6774-X.
47. Reynolds, C., *Generalization in User Interface Design.* 1999.
48. Sharkey, E. and J. Paynter, *Lessons to be learnt from students software estimation exercises*.
49. Shneiderman, B., *Designing the user interface: strategies for effective human computer interaction*: Addison-Wesley. p. 639, 1998. isbn 0-201-69497-2.
50. Silva, C., *LGantt* http://ghannid.homeip.net/wiki/LGanttPrd_es.wiki *Último acceso: 15/03/2010*.
51. Souza, M.J.F., G.P. Silva, and S.M.S. Mapa, *Métodos de pesquisa em vizinhança variável aplicados à soluçao do problema de programaçao diária de tripulaçoes de ônibus urbano*. Congresso de Pesquisa e Ensino em Transportes: ANPET. p. 1492-1501, 2004. isbn 8587893-09-2 (VOL. 1) y 8587893-10-6 (VOL. 2).
52. Svanæs, D., *Understanding Interactivity.*
53. Thomas, R., *A Practical Experiment in Teaching Software Engineering Metrics*. 1996.
54. Voss, S., *Meta-heuristics: advances and trends in local search paradigms for optimization*: Kluwer. p. 511, 1999. isbn 0-7923-8369-9.
55. Vredenburg, K., S. Isensee, and C. Righi, *User-centered design: an integrated approach*: Prentice Hall. p. 246, 2002. isbn 0130912956.
56. Wichary, M., *GUIdebook: Graphical User Interface Gallery* http://www.guidebookgallery.org/index *Último acceso: 20/12/2009*.
57. Wikipedia, *GOMS* http://en.wikipedia.org/wiki/GOMS *Último acceso: 10/03/2010*.
58. Wikipedia, *Gráficos en 3D* http://es.wikipedia.org/wiki/Gr%C3%A1ficos_3D_por_computadora *Último acceso: 03/03/2010*.
59. Wikipedia, *Historia de Amazon* http://es.wikipedia.org/wiki/Amazon.com *Último acceso: 03/03/2010*.
60. Wikipedia, *Historia de Facebook* http://es.wikipedia.org/wiki/Facebook *Último acceso: 03/03/2010*.
61. Wikipedia, *Historia de Google* http://en.wikipedia.org/wiki/Google *Último acceso: 20/12/2009*.

62. Wikipedia, *Historia de la mensajería instantánea* http://en.wikipedia.org/wiki/Instant_messaging *Último acceso: 20/12/2009*.
63. Wikipedia, *Historia de las redes sociales* http://en.wikipedia.org/wiki/Social_network_service *Último acceso: 20/12/2009*.
64. Wikipedia, *Historia de los blogs* http://en.wikipedia.org/wiki/Blog *Último acceso: 20/12/2009*.
65. Wikipedia, *Historia de los teléfonos móviles* http://en.wikipedia.org/wiki/History_of_mobile_phones *Último acceso: 20/12/2009*.
66. Wikipedia, *Historia de Microsoft Windows* http://en.wikipedia.org/wiki/History_of_Microsoft_Windows *Último acceso: 20/12/2009*.
67. Wikipedia, *Historia de MySpace* http://es.wikipedia.org/wiki/Myspace *Último acceso: 03/03/2010*.
68. Wikipedia, *Historia de Napster* http://es.wikipedia.org/wiki/Napster *Último acceso: 03/03/2010*.
69. Wikipedia, *Historia de P2P* http://en.wikipedia.org/wiki/File_sharing *Último acceso: 20/12/2009*.
70. Wikipedia, *Historia de PayPal* http://es.wikipedia.org/wiki/PayPal *Último acceso: 03/03/2010*.
71. Wikipedia, *Historia de Twitter* http://es.wikipedia.org/wiki/Twitter *Último acceso: 03/03/2010*.
72. Wikipedia, *Historia de YouTube* http://es.wikipedia.org/wiki/YouTube *Último acceso: 03/03/2010*.
73. Wikipedia, *Historia del IPad* http://es.wikipedia.org/wiki/Ipad *Último acceso: 03/03/2010*.
74. Wikipedia, *Historia del libro electrónico* http://es.wikipedia.org/wiki/Ebook *Último acceso: 03/03/2010*.
75. Wikipedia, *Historia del Transporte* http://es.wikipedia.org/wiki/Transporte *Último acceso: 01/02/2010*.
76. Wikipedia, *Historia del USB* http://es.wikipedia.org/wiki/Universal_Serial_Bus *Último acceso: 03/03/2010*.
77. Winder, S.A.J., *A Brief Survey of Central Mechanisms in Primate Visual Perception.* 2002.
78. Zurek, E.E., *Sistemas urbanos: Amplio campo para la investigación*.