



Universidad de Valladolid

# Desarrollo de una herramienta para la programación de proyectos con restricciones de recursos basada en la heurística P-SGS/MINSLK

Agustin Araujo Rodriguez

MÁSTER EN DIRECCIÓN DE PROYECTOS  
Departamento De Organización De Empresas Y C.I.M.  
Universidad De Valladolid  
España



**INSISOC**  
SOCIAL SYSTEMS  
ENGINEERING CENTRE  
2024





**Universidad de Valladolid**

# **Desarrollo de una herramienta para la programación de proyectos con restricciones de recursos basada en la heurística P-SGS/MINSLK**

Agustín Araujo Rodriguez

MÁSTER EN DIRECCIÓN DE PROYECTOS  
Departamento De Organización De Empresas Y C.I.M.  
Universidad De Valladolid

Valladolid, Junio 2024

## **Tutores**

David Jesús Poza García  
Félix Antonio Villafañez Cardeñoso



## **AGRADECIMIENTOS**

Deseo expresar mi más sincero agradecimiento a todas aquellas personas que hicieron que fuera posible que hoy yo esté aquí.

Me gustaría también agradecer a David Poza y Félix Villafañez, por su dedicación y esfuerzo como tutores de este trabajo.



## **RESUMEN**

El presente trabajo de fin de máster se enfoca en el desarrollo de una herramienta de resolución de problemas multiproyectos con limitación de recursos (Problemas RCMPSP). Esta herramienta mejora la eficiencia en la programación de proyectos optimizando el uso de recursos y garantizando su viabilidad.

Su algoritmo de resolución se basa en las características de la heurística P-SGS/MINSLK, la que ha demostrado ser eficaz para la resolución de problemas RCMPSP. Es desarrollada en lenguaje VBA, integrada a Microsoft Project, de uso intuitivo y eficiente. Además, automatiza procesos relevantes en la gestión de carteras e incorpora la gestión de los recursos financieros de los proyectos.

Validada en problemas de la librería MPSPLib, ha mostrado obtener excelentes resultados frente a la herramienta ya disponible en MS Project.

El uso de la herramienta permitirá la gestión eficiente de los proyectos, lo que redundará en organización más productivas y competitivas en el mercado.

### **Palabras claves**

RCMPSP, Heurística, Programación multiproyecto, Microsoft Project, P-SGS/MINSLK, Dirección de Carteras de Proyectos.

## **ABSTRACT**

This master's thesis focuses on the development of a tool for solving multi-project problems with resource limitations (RCMPSP Problems). This tool improves efficiency in project scheduling by optimizing the use of resources and guaranteeing their viability.

Its resolution algorithm is based on the characteristics of the P-SGS/MINSLK heuristic, which has proven to be effective for solving RCMPSP problems. It is developed in VBA language, integrated with Microsoft Project, intuitive and efficient to use. In addition, it automates relevant processes in portfolio management and incorporates the management of project financial resources.

Validated in problems of the MPSPLib library, it has shown to obtain excellent results compared to the tool already available in MS Project.

The use of the tool will allow efficient project management, which will result in more productive and competitive organizations in the market.

### **Keywords**

RCMPSP, Heuristic, Multi-project Scheduling, Microsoft Project, P-SGS/MINSLK, Project Portfolio Management.





# INDICE

<b>INTRODUCCIÓN.....</b>	<b>1</b>
Objetivo del Proyecto .....	2
Alcance del Proyecto .....	2
Motivación del Proyecto.....	3
Estructura del Documento .....	3
<b>Capítulo 1 Introducción a la programación de proyectos .....</b>	<b>5</b>
1.1 Descripción del problema .....	5
1.2 Gestión estratégica y proyectos .....	6
1.3 Revisión histórica de la programación de proyectos .....	7
1.4 Holgura y criticidad de una actividad .....	8
1.5 Recursos de un proyecto .....	9
1.6 Evaluación de la programación de proyectos .....	10
1.6.1. TMS ( <i>Total Makespan</i> ).....	10
1.6.2. Índice de ocupación de recursos .....	10
1.6.3. Biblioteca MPSPLib .....	11
<b>Capítulo 2 Programación avanzada de proyectos.....</b>	<b>13</b>
2.1 Método metaheurístico .....	13
2.2 Reglas de prioridad.....	15
2.3 Enfoques C-RCMPSP y D-RCMPSP .....	15
2.4 Heurística P-SGS/MINSLK.....	16
<b>Capítulo 3 Programación en software de proyectos .....</b>	<b>19</b>
3.1 Introducción.....	19
3.2 Microsoft Project .....	19
3.3 Programación con restricción de recursos en Microsoft Project .....	20
3.4 Desarrollo de automatizaciones en Microsoft Project .....	21
3.4.1. VBA (Visual Basic for Applications).....	21
3.4.1.1 Definir variables.....	22
3.4.1.2 Instrucción <i>Sub</i> .....	22
3.4.1.3 Instrucción <i>Function</i> .....	22
3.4.1.4 Instrucción <i>For</i> .....	23
3.4.1.5 Instrucción <i>For Each</i> .....	23
3.4.1.6 Instrucción <i>If Then Else</i> .....	23
3.4.1.7 Instrucción <i>While</i> .....	24
3.4.1.8 Instrucción <i>InputBox</i> .....	24
<b>Capítulo 4 Herramienta desarrollada.....</b>	<b>25</b>

---

4.1 Módulo Crear Tareas .....	26
4.1.1. Definición de la instancia que representa el proyecto.....	26
4.1.2. Declaración de variables .....	26
4.1.3. Obtención de información de archivo Excel.....	27
4.1.4. Creación de tareas .....	27
4.1.5. Incorporación de sucesoras .....	28
4.1.6. Incorporación de recursos .....	28
4.1.7. Obtención de los recursos del proyecto .....	29
4.1.8. Cierre de libro de Excel .....	30
4.2 Módulo Obtener información de proyectos .....	30
4.2.1. Declaración de variables .....	31
4.2.2. Ingreso de cantidad de proyectos de la cartera .....	31
4.2.3. Ingreso de la ubicación del proyecto.....	31
4.2.4. Crear instancia de proyecto.....	31
4.2.5. Actualizador de Id de tareas.....	32
4.2.6. Migrar información básica de las tareas .....	32
4.2.7. Migrar predecesoras.....	33
4.2.8. Migrar asignación de recursos .....	35
4.2.9. Migrar disponibilidad de recursos.....	35
4.3 Modulo Gestor de Recursos.....	36
4.3.1. Declaración e inicialización de variables.....	36
4.3.2. Iniciación de archivo de registro de acciones .....	37
4.3.3. Definir horizonte temporal a nivelar recursos.....	37
4.3.4. Iteración en cada momento .....	38
4.3.5. Actividades ya comenzadas .....	38
4.3.6. Actividades candidatas.....	39
4.3.7. Regla de prioridad por holgura .....	39
4.3.8. Arreglo de disponibilidad de recursos .....	40
4.3.9. Descuento de tareas comenzadas .....	40
4.3.10. Nivelación de recursos.....	41
4.4 Modulo Gestión Financiera .....	43
4.4.1. Declaración de variables .....	43
4.4.2. Línea base de financiación.....	43
4.4.3. Costo por periodo.....	44
4.4.4. Línea base de costo y evaluación de disponibilidad financiera .....	46
4.5 Instalación de la herramienta en MS Project .....	46
4.6 Uso de la herramienta .....	47
4.6.1. Crear tareas .....	47
4.6.2. Obtener información de proyectos.....	48
4.6.3. Gestor Recursos .....	50
4.6.4. Gestión financiera .....	51
<b>Capítulo 5 Análisis de resultados .....</b>	<b>53</b>
5.1 Metodología de evaluación de resultados .....	53
5.2 Resultados obtenidos .....	55

<b>CONCLUSIONES.....</b>	<b>61</b>
<b>Futuros pasos.....</b>	<b>63</b>
<b>INDICE DE FIGURAS .....</b>	<b>71</b>
<b>INDICE DE TABLAS.....</b>	<b>73</b>



---

## INTRODUCCIÓN

Los proyectos permiten a las organizaciones enfocar sus recursos para lograr un avance sustancial en temas que son de su interés, así como la resolución de problemas complejos que la estructura habitual de la organización no es capaz de resolver de forma eficiente. El avance que se logra mediante un proyecto permite un crecimiento real de la organización en el tema en cuestión, y eleva su gestión sobre su estado actual.

En la práctica, cuando las organizaciones deciden avanzar sobre una temática en particular realizan una gestión multiproyecto, donde se ejecutan varios proyectos simultáneamente usando un conjunto limitado de recursos (Pajares Gutierrez & Otros, 2009).

Cada uno de los proyectos que conforman una cartera cuenta con un conjunto de actividades que son necesarias realizar para alcanzar el o los objetivos propuestos. Estas actividades utilizan recursos (humanos, materiales, etc.) que generalmente son limitados (es decir, existe un número máximo de recursos que pueden emplearse, de manera simultánea, en la ejecución de estas actividades). En consecuencia, para obtener una programación (es decir, para establecer las fechas planificadas de ejecución de las actividades) que sea viable es necesario considerar la disponibilidad real de cada recurso y su uso a lo largo del tiempo. Estas consideraciones permiten evitar momentos en la ejecución del proyecto en los que se tengan recursos sobreasignados, lo que haría inviable la ejecución de las actividades según el programa establecido.

Obtener una programación viable de un proyecto considerando la escasez de sus recursos no es una tarea sencilla, tanto es así que se ha convertido en una línea de investigación bajo el nombre RCPSP (por sus siglas en inglés: *Resource Constrained Project Scheduling Problem*) y se ha demostrado que es un problema NP-Complejo<sup>1</sup> (Blazewicz, 1983). Cuando se analizan la gestión de recursos en múltiples proyectos, la dificultad de obtener una programación factible de la cartera de proyectos aumenta considerablemente, la línea de investigación para el estudio de este tipo de problemas se conoce como RCMPSP (por sus siglas en inglés: *Resource-Constrained Multi-Project Scheduling Problem*), como es de esperar, este problema tampoco se puede resolver de forma directa ya que también es considerado un problema NP-Complejo.

Entonces, dada la complejidad del problema en análisis, se han desarrollado distintos algoritmos capaces de obtener soluciones “suficientemente buenas”. Los softwares comerciales de programación de proyectos utilizan alguno de estos algoritmos que permiten una programación factible, pero dada la dificultad computación que implica el problema, las soluciones obtenidas son alejadas a la solución óptima. Microsoft Project es uno de los principales softwares de administración de proyectos que cuenta con una herramienta de nivelación de recursos que permite obtener una solución factible a los problemas RCMPSP, sin embargo, su algoritmo de resolución es desconocidos y obtiene resultados evidentemente alejados del óptimo.

---

<sup>1</sup> Problema NP-Complejo: son una clase de problemas computacionales extremadamente difíciles que se consideran intratables en la práctica, ya que no se pueden resolver de manera eficiente en tiempo polinomial.

En la búsqueda del desarrollo de algoritmos más inteligentes y eficientes, investigadores de la Universidad de Valladolid han desarrollado la heurística P-SGS/MINSLK (Villafañez & Otros, 2018) para la resolución de problemas RCMPSP. Se ha demostrado que esta heurística permite obtener muy buenos resultados en la programación de los proyectos, a pesar de su sencillez.

Por más que este tipo de trabajos demuestran potencial, éstos suelen quedar en el ámbito académico, por lo que su aplicación a problemas reales o su implementación en soluciones informáticas es limitado. En este marco de conocimiento, el primer objetivo de este trabajo Fin de Máster (TFM) consiste en el desarrollo de un módulo de Microsoft Project que integre a la herramienta informática la heurística P-SGS/MINSLK y así obtener una programación del proyecto factible según sus pasos, y que además sea de fácil uso y proporcione información relevante para el usuario.

Si bien la heurística P-SGS/MINSLK (Villafañez & Otros, 2018) permite obtener programaciones viables para carteras de proyectos mediante nivelación de los recursos, ésta no tiene en cuenta los costos de ejecución de las actividades ni la financiación real para la ejecución del proyecto. La línea base de costo debe ser analizada en detalle para garantizar el financiamiento necesario en cada periodo de tiempo. De esta forma se mitiga el riesgo de descalce de financiación, que provocaría un retraso en la programación del proyecto.

Por este motivo, el segundo objetivo de este TFM es incorporar la visión financiera de la gestión de proyecto mediante la elaboración de un módulo que permita el análisis de la línea base de costo del proyecto y anticipe los periodos de mayor riesgo de financiación insuficiente.

Este trabajo redundará en organizaciones más eficientes, mediante la programación y ejecución de proyectos viables en un menor plazo, mediante la optimización del uso de los recursos.

## **Objetivo del Proyecto**

El objetivo de este trabajo es el desarrollo de una herramienta completamente integrada a Microsoft Project capaz de obtener una programación de una cartera de proyectos según la heurística P-SGS/MINSLK, así como, facilitar al usuario la gestión de su cartera mediante el desarrollo de módulos que permiten automatizar la carga de información y, finalmente, incorporar la gestión de los recursos financieros de los proyectos en el análisis de su programación. Los desarrollos realizados deben ser eficiente, de uso intuitivo para los usuarios y fácilmente compartible entre los mismos.

## **Alcance del Proyecto**

A partir del objetivo planteado, se establece que las tareas necesarias para su cumplimiento son esencialmente las siguientes:

- Revisión bibliográfica y estado del arte de los problemas RCPSP y RCMPSP, particularmente de la heurística P-SGS/MINSLK presentada en el trabajo “*A generic heuristic for multi-project scheduling problems with global and local resource constraints (RCMPSP).*”

- 
- Desarrollo de módulos completamente integrados a Microsoft Project capaces de obtener una programación de proyecto según la heurística P-SGS/MINSLK.
  - Validación del funcionamiento de los módulos desarrollados y análisis de resultados obtenidos utilizando problemas de la librería MPSPLib.
  - Elaborar conclusiones y establecer posibles futuras líneas de trabajo para el crecimiento de los desarrollos realizados.

## **Motivación del Proyecto**

La motivación principal del presente trabajo es obtener una herramienta de fácil acceso y uso para los directores de proyecto. Las herramientas comerciales, principalmente Microsoft Project, incorporan módulos que permiten obtener una programación viable de los proyectos considerando las restricciones de recursos, pero se considera que son soluciones evidentemente alejadas del óptimo, y no se conoce el algoritmo de solución. Por otro lado, se ha demostrado que utilizando heurísticas se obtienen soluciones suficientemente buenas, pero estas suelen ser de complejidad de uso práctico para los directos de proyecto. Entonces, combinando ambas líneas de trabajo, se espera desarrollar una herramienta que incorpore heurísticas eficientes completamente integradas en Microsoft Project, que permita obtener programaciones de proyectos viables y eficientes, para de esta forma, lograr la optimización del uso de los recursos de las organizaciones.

## **Estructura del Documento**

El presente trabajo tiene la siguiente estructura:

- En el capítulo 1 se describe el problema en análisis en este TFM. Posteriormente, se realiza una breve revisión historia de la programación de proyectos, donde se mencionan los principales métodos desarrollados a lo largo del tiempo. Finalmente se presentan conceptos claves vinculados a la programación de proyectos, que son utilizados a lo largo de este trabajo.
- En el capítulo 2 se profundiza en el uso métodos metaheurísticos y basados en reglas de prioridad para la resolución de problemas complejo. Posteriormente, se desarrolla la heurística P-SGS/MINSLK, principal punto de partida de este trabajo. También se presenta dos enfoques de los problemas RCMPSP, el enfoque C-RCMPSP y el enfoque DRCMPSP.
- En el capítulo 3 se desarrolla sobre los sistemas de gestión de proyectos, particularmente la herramienta Microsoft Project y su uso para la programación de proyecto teniendo en cuenta las restricciones de recursos. Se busca la comprensión de la utilidad del desarrollo de automatizaciones en la herramienta para la resolución de problemas complejos.
- En el capítulo 4 se describe en detalle el código desarrollado en cada uno de los módulos objeto de este trabajo, con la intención de que el lector sea capaz de entender la codificación realizada y los pasos necesarios para su uso en la práctica. Además, es relevante que los lectores más avanzados sean capaces de incluirle mejoras y aumentar su capacidad.

- En el capítulo 5 se describe el trabajo realizado con un conjunto de proyectos de la librería MSPSLib y un análisis pormenorizado de los resultados, comparándolos principalmente con los obtenidos a partir de la nivelación de recursos que se obtiene en Microsoft Project. Aquí se espera mostrar el valor real del trabajo realizado frente a aquel ya disponible para el usuario.
- Finalmente, se cierra este trabajo con una serie de conclusiones y cuáles, se estima, son los futuros pasos a seguir para el desarrollo de la herramienta.



---

## Capítulo 1 Introducción a la programación de proyectos

En este capítulo, en primer lugar, se presenta la descripción del problema de análisis en este TFM. Posteriormente se presenta una revisión bibliográfica sobre la programación de proyectos. En la sección 1.2, se muestra la relevancia de la gestión de proyectos como mecanismo para la implementación de los lineamientos estratégicos de las organizaciones, las que, en un entorno cada vez más competitivo, buscan optimizar el uso de sus recursos. En la sección 1.3 se presenta una revisión histórica de la programación de proyectos. Finalmente, las secciones 1.4, 1.5 y 1.6 presentan conceptos claves vinculados a la programación de proyectos que son utilizados a lo largo de este TFM.

### 1.1 Descripción del problema

La programación de proyectos considerando la restricción de recurso son problemas de optimización combinatoria por lo que son considerados problemas NP-Complejo, este tipo de problemas conllevan una alta complejidad de resolución dado que no es posible construir algoritmos que en tiempo polinomial puedan resolver cualquier instancia del problema (Oliveira, 2004), por lo que se han desarrollado distintas líneas de investigación con el objetivo de encontrar una forma eficiente para obtener su solución óptima. La línea de investigación que trata el problema de programación de un proyecto con restricción de recursos es conocido como RCPSP (por sus siglas en inglés: *Resource Constrained Project Scheduling Problem*), mientras que el problema de programación de una cartera de proyecto con restricción de recursos es conocido como RCMPSP (por sus siglas en inglés: *Resource-Constrained Multi-Project Scheduling Problem*).

Los problemas RCPSP y RCMPSP pueden ser modelados mediante un problema de optimización con distintas funciones objetivos, como puede ser, minimizar la duración del proyecto, minimizar el costo, optimizar el uso de los recursos, etc. Sin duda, minimizar el tiempo total (TMS) del proyecto es el modelo más utilizado por las organizaciones, en la práctica se busca proyectos con plazos mínimos utilizando la menor cantidad de recursos posible. Este objetivo debe obtenerse cumpliendo al menos las siguientes restricciones:

- Duración mínima estimada de cada actividad.
- Relaciones de precedencia entre las actividades.
- Disponibilidad de recursos suficiente para cada actividad.

Por lo cual lo que se busca programar una solución viable al problema de optimización que se presenta en la figura 1.

Objective Function:

$$\text{find } S_I = \{ST_{I_{start}}, ST_{11}, \dots, ST_{1n_1}, \dots, ST_{ij}, \dots, ST_{m1}, \dots, ST_{mn_m}, ST_{I_{end}}\}$$

such that

$$\text{minimize } TMS_I \quad (\approx \min. ST_{I_{end}}) (\approx \min. (\max_{j \in I_i, i \in I} (ST_{ij} + d_{ij})))$$

Subject to

$$\begin{cases} ST_{ij} \geq ST_{iq} + d_{iq} & \forall i \in I; j, q \in J_i / q \in P_{ij} \\ ST_I = \min_{i \in I} ST_i = \min_{i \in I} ST_{I_{start}} = AD_I & / AD_I \geq 0 \\ \sum_{j \in J_{it}} r_{ijit} \leq R_{it} = R_{il}(t) = R_{il} = \text{const.} & \forall l \in L_i; i \in I; t \in \{AD_I, \dots, \bar{T}_I\} \\ \sum_{i \in I, j \in J_{it}} r_{ijkt} \leq R_{kt} = R_k(t) = R_k = \text{const.} & \forall k \in K_i; t \in \{AD_I, \dots, \bar{T}_I\} \end{cases}$$

**Figura 1:** Modelo matemático del problema RCMPSP. Fuente: (Villafañez & Otros, 2018)

Dada la complejidad de este tipo de problemas, únicamente los casos con pocas actividades y recursos son posibles de resolver mediante métodos exactos. Hoy en día, no se ha encontrado un método eficiente en todos los casos que permita obtener la programación óptima de un proyecto considerando las restricciones de recursos, por lo que dependiendo de las características del problema será mejor utilizar un método u otro.

Uno de los métodos de resolución de este tipo de problemas ampliamente estudiado es la heurística, permite obtener una solución existente en el espacio de soluciones del problema que sea suficientemente buena con un bajo uso de recursos asociados a la búsqueda de soluciones. Se han desarrollado múltiples heurísticas para la resolución de este tipo de problemas, basadas en distintas metaheurísticas y utilizando variados factores de priorización, obteniendo resultados dispares de acuerdo con la característica propia de los problemas a resolver.

De forma paralela a la investigación de heurísticas para la programación de proyectos, se han desarrollado sistemas de gestión de proyectos comerciales que brindan herramientas para la programación de proyectos con restricciones de recursos. Estos sistemas utilizan algoritmos internos desconocidos y que permiten obtener soluciones factibles, pero, en muchas ocasiones, alejadas de la solución óptima.

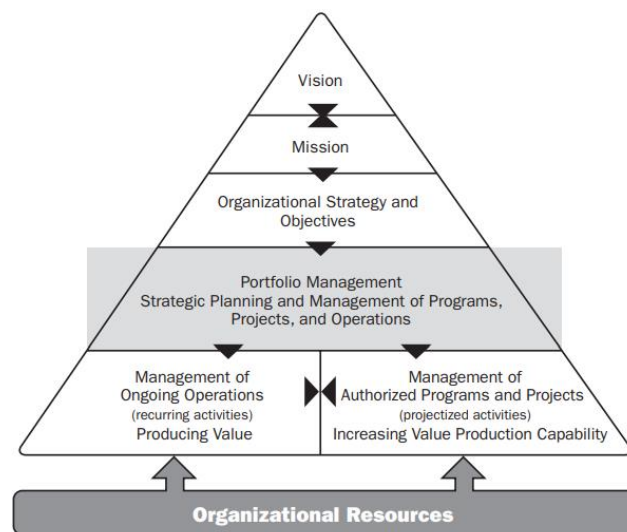
La integración de los avances en el desarrollo de heurísticas eficientes para la programación de proyectos en sistemas de gestión comerciales de uso expandido por los directores de proyectos es una oportunidad de obtener proyectos con una programación más eficiente, que permita la optimización del uso de recurso de las organizaciones, esto es, la problemática principal que intenta resolver este TFM.

## 1.2 Gestión estratégica y proyectos

Hoy en día, los altos niveles de competitividad de los mercados provocan que las organizaciones deban ser cada vez más eficientes en sus procesos y en el uso de sus recursos, por lo que una óptima

programación de los proyectos se ha convertido en algo esencial en la profesión de Director de Proyectos.

En este contexto de alta competitividad de los mercados, las organizaciones establecen sus lineamientos estratégicos a largo plazo en base a sus valores, misión, visión, análisis de entorno y análisis internos; la figura 2 muestra como estos lineamientos son desplegados en la práctica mediante los procesos operacionales propios de la organización y mediante proyectos. Estos proyectos se gestionan en torno a una estructura de trabajo eficiente y multidisciplinaria, donde es posible tratar de forma efectiva los temas que surgen de los lineamientos estratégicos y aquellos de compleja ejecución por la estructura organizacional habitual. Los proyectos relacionados que comparten un objetivo estratégico común se agrupan en programas de proyectos, estos programas y/o proyectos, relacionados o no, se agrupan en carteras de proyectos (Pajares, 2024).



**Figura 2:** Visión estratégica de las carteras de proyectos. Fuente: (PMI, 2017)

Existen distintas metodologías o estándares de gestión de carteras de proyectos, algunos de los más destacados son “The Standard for Portfolio Management” en su 4ta edición del Project Management Institute (PMI), “Individual Competence Baseline for Portfolio Management” en su versión 4.0 de la International Project Management Association (IPMA) y la metodología “PM<sup>2</sup> Portfolio Management Guide 1.5” de Open Project Management Methodology (PM<sup>2</sup>). Todos estos estándares presentan la relevancia de la programación de los proyectos para mantener una cartera optimizada y equilibrada en cuanto plazos, riesgos, recursos y necesidad de financiación. Particularmente es de estudio los criterios de programación de los proyectos dentro de la cartera y los criterios de distribución de recursos, donde se presentan métodos como los modelos jerárquicos, modelos heurísticos o de cadena crítica (Pajares, 2024).

### 1.3 Revisión histórica de la programación de proyectos

Se considera que la programación de proyectos inicia en los principios del siglo XX con los trabajos de Frederick Taylor, quien promovía una visión científica del trabajo, procurando un aumento de

la productividad a través de la mejora de la eficiencia y no de los recursos o “trabajar más duro”. En 1910 Henry Gantt crea el llamado Diagrama de Gantt, en el cual se representa la duración prevista de las actividades en intervalos de tiempo, así como los principales hitos o entregas del proyecto. Esta herramienta aun es utilizada ampliamente ya que permite una rápida visualización de la programación de las actividades del proyecto a lo largo de su vida.

Posteriormente comienza una etapa en la cual se desarrollan métodos de programación de proyectos basados en grafos. En el año 1957 se desarrolla el Método CPM (*Critical Path Method*) por parte de la empresa DuPont, y en 1958, la Armada de Estados Unidos desarrolla el método PERT (*Program Evaluation and Review Technique*), ambos métodos tienen en cuenta la relación de precedencia de las actividades y su duración estimada para obtener una programación del proyecto viable. Tanto CPM como PERT son métodos aún vigentes.

Estos métodos no tienen en cuenta la disponibilidad de los recursos necesarios para llevar a cabo las actividades, por lo que permiten una programación viable únicamente si consideramos que la disponibilidad de recursos es suficiente en todo momento del proyecto, lo que no es aplicable a la realidad de las organizaciones. Por este motivo, es necesario el desarrollo de métodos que consideren la necesidad y disponibilidad de los recursos.

Para resolver problemas de programación considerando las restricciones de recurso es que surgen métodos avanzados de programación de proyectos, como son los métodos heurísticos, que han permitido desarrollar algoritmos modernos de fácil uso para obtener soluciones aproximadas.

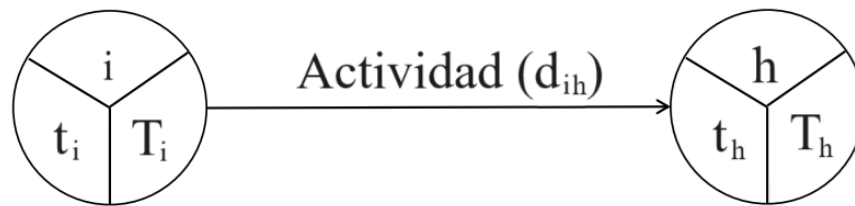
## 1.4 Holgura y criticidad de una actividad

Luego de que se identifican las actividades, se estima su duración y se identifican sus relaciones de precedencia, es posible programar el proyecto y obtener información relevante para la gestión y control del cronograma. Un ejemplo de esto es la holgura total ( $H_i$ ) de una actividad, lo que es la cantidad de tiempo que puede retrasarse una actividad  $i$  sin que afecte la duración total del proyecto, mientras que la holgura libre ( $H_i^*$ ) es la cantidad de tiempo que puede retrasarse una actividad  $i$  sin comprometer la fecha de comienzo de sus actividades sucesoras<sup>2</sup>.

Las actividades con holgura total igual a cero forman el camino crítico del proyecto, por lo que son consideradas actividades críticas. Estas actividades son de especial atención al momento del seguimiento y control del proyecto, ya que un retraso en la ejecución de una actividad crítica significa un retraso en la duración total del proyecto.

---

<sup>2</sup> Las actividades sucesoras son aquellas que tienen una dependencia actividad  $i$  y ocurren luego de esta, en función de la relación de precedencia.



**Figura 3:** Actividad en un diagrama AOA<sup>3</sup>

En la figura 3 se representa una actividad en un diagrama AOA entre el nodo  $i$  y el nodo  $h$ , la duración de la actividad es  $d_{ih}$ . Las variables  $t_i$  y  $t_h$  representan el momento más temprano en que todas las actividades predecesoras a la actividad pueden haber finalizado, mientras que las variables  $T_i$  y  $T_h$  representan el momento más tardío en el que todas las actividades predecesoras deben haber finalizado.

Por lo tanto, la holgura total ( $H_{ih}$ ) y la holgura libre ( $H_{ih}^*$ ) de una actividad entre los nodos  $i$  y  $h$  se calculan de la siguiente manera:

$$H_{ih} = T_h - t_i - d_{ih} \quad (1)$$

$$H_{ih}^* = t_h - t_i - d_{ih} \quad (2)$$

Entonces, el camino crítico  $C$  es el conjunto de actividades  $k_{ih}$ , con nodo de inicio  $i$  y nodo de fin  $h$ , que la sumatoria de su holgura total  $H_{ih}$  es 0:

$$C = \{k_{ih} / \sum H_{ih} = 0\} \quad (3)$$

## 1.5 Recursos de un proyecto

Los recursos son todos los activos necesarios para ejecutar las actividades de un proyecto según su planificación. Pueden ser personas, equipos, dinero o tiempo asignados a cada una de las actividades.

La gestión de los recursos es una de las tareas claves en la gestión de proyectos, el PMBOK en su sexta edición contiene un área de conocimiento específica para la gestión de recursos, la que incluye seis procesos para la planificación, ejecución y control de los recursos (PMI, 2017).

Existen recursos asignados a un proyecto en particular, llamados recursos locales, o recursos pertenecientes a la cartera de proyectos, llamados recursos globales, los que pueden ser utilizados por cualquier proyecto de la cartera. Los recursos son costosos, no contar con un recurso en el

---

<sup>3</sup> *Activity on Arcs*: Representación en grafo de un proyecto donde los arcos representan las actividades y los nodos los eventos temporales.

momento que es necesario puede llevar al proyecto a retrasos o sobrecostos, por lo que optimizar el uso y disponibilidad de estos es relevante para el éxito del proyecto.

## 1.6 Evaluación de la programación de proyectos

Es importante describir algunos indicadores o metodologías que son de utilidad para poder comparar las programaciones de proyectos obtenidas con distintos métodos, y así tener un factor de decisión objetivo que nos de información para concluir que método es mejor.

### 1.6.1. TMS (*Total Makespan*)

El TMS (*Total Makespan*): Es la diferencia entre la fecha máxima de finalización del conjunto de actividades y la fecha más temprana de comienzo del conjunto de proyectos de la cartera, lo que es:

$$\text{TMS} = \max_q\{f_q\} - \min_i\{AD_i\} \quad (4)$$

Donde:

$q$ : actividad

$f_q$ : tiempo de finalización de la actividad  $q$

$AD_i$ : tiempo de inicio del proyecto  $i$

### 1.6.2. Índice de ocupación de recursos

El índice de ocupación de recursos ( $OR_r$ ): Representa la utilización de cada uno de los recursos de la cartera de proyectos, es mejor programación de los proyectos aquella que logre mayor índice de ocupación, ya que permite la optimización del uso de los recursos.

$$OR_r = \frac{\sum_q(d_q * k_{qr})}{TM * a_r} \quad (5)$$

Donde:

$r$ : recurso

$q$ : actividad

$d_q$ : duración de la actividad  $q$

$k_{qr}$ : capacidad del recurso  $r$  asignado a la actividad  $q$ , por unidad de tiempo

TMS: *Total Makespan*

$a_r$ : disponibilidad del recurso  $r$  por unidad de tiempo

### 1.6.3. Biblioteca MPSPLib

MPSPLib (*Multi-Project Scheduling Problem Library*) es una librería de problemas RCMPSP que permite evaluar los algoritmos de solución a estos problemas, se compone de programas de proyectos estándares que permiten comparar distintos indicadores resultantes de la ejecución de los algoritmos propuesto por los investigadores.

En MPSPLib se incluyen un total de 140 problemas de distintas características como tamaño, complejidad y recursos; por lo que se pueden recrear distintos escenarios para poner a prueba los algoritmos de resolución. Los problemas constan de cuatro recursos (R1, R2, R3 y R4) de los que al menos uno es un recurso global y los restantes son recursos locales, también puede estar formado por 30, 90 o 120 tareas y 2, 5, 10 o 20 proyectos. La combinación de estos factores forma los 140 escenarios distintos.

Luego de acceder a la librería (<http://www.mpsplib.com>) se puede descargar la información de cada problema a partir de dos archivos. El primero de ellos se muestra en la figura 4, figura 5 y figura 6. En (1) se muestra la información general del problema como su nombre e identificador.

```

*****:
file with basedata      : j9015_.bas
initial value random generator: 65205087
*****:
projects                : 1
jobs (incl. supersource/sink ): 92
horizon                 : 498
RESOURCES
- renewable             : 4   R
- nonrenewable         : 0   N
- doubly constrained   : 0   D
*****:
PROJECT INFORMATION:
prn. #jobs rel.date duedate tardcost MPM-Time
1    90    3    0    93    69    93
*****:
PRECEDENCE RELATIONS:
jobnr. #modes #successors successors
1      1      3      2 3 4
2      1      1      5

```

**Figura 4:** Información de proyectos en biblioteca MPSPLib I

También contiene información de la cantidad de actividades del proyecto, incluidas las actividades ficticias de inicio y fin de cada proyecto (2) y sin incluir estas (3), así como, el horizonte temporal del problema sin la limitación de recursos (4). Por otro lado, se muestra toda la información necesaria para la programación de los proyectos, como las actividades sucesoras (5), y la duración de las actividades y uso de recursos de cada una (6).

REQUESTS/DURATIONS:						
jobnr.	mode	duration	R 1	R 2	R 3	R 4
1	1	0	0	0	0	0
2	1	7	2	2	6	2

**Figura 5:** Información de proyectos en biblioteca MPSPLib II

RESOURCEAVAILABILITIES:				
R 1	R 2	R 3	R 4	
52	48	53	46	

**Figura 6:** Información de proyectos en biblioteca MPSPLib III

Al final del archivo se tiene información sobre la disponibilidad de cada uno de los recursos utilizados en el proyecto, lo que se muestra en la figura 6.

En la figura 7 se presenta el segundo archivo en formato .xml donde nuevamente se tiene el nombre del proyecto (8), información de cada uno de los proyectos de la cartera como su fecha de inicio (9) e información sobre los recursos globales del proyecto (10).

Para la gestión de los recursos globales en MPSPLib se ignora la información contenida en el archivo de cada proyecto sobre ese recurso, la que se muestra en el punto (7), y se tiene en cuenta únicamente la que surge del archivo xml, punto (10).

```

<!DOCTYPE mp-list SYSTEM "mp.dtd">
<mp-list>
  <mp>
    <name>mp_j120_a2_nr3</name>
    <project-list>
      <project>
        <filename>KolischInstanzen/j120/j12058_10.sm</filename>
        <start>0</start>
      </project>
      <project>
        <filename>KolischInstanzen/j120/j12059_1.sm</filename>
        <start>5</start>
      </project>
    </project-list>
    <resources>
      <resource>29</resource>
      <resource>0</resource>
      <resource>0</resource>
      <resource>26</resource>
    </resources>
  </mp>
</mp-list>

```

**Figura 7:** Información de proyectos en biblioteca MPSPLib IV



---

## Capítulo 2 Programación avanzada de proyectos

Una heurística o método heurístico permite obtener una solución, existente en el espacio de soluciones factibles, suficientemente buena para los problemas NP-Complejos, en un límite de tiempo aceptable y/o utilizando una capacidad computacional aceptable.

El método heurístico surge de la necesidad de obtener una solución a los problemas complejos, como son los problemas RCMPSP, los que requieren una importante cantidad de recursos para obtener una solución exacta o bien no son posibles de resolver con la capacidad computacionales actuales. La utilización de método heurístico para estos casos consiste en analizar las características del problema y elaborar un conjunto de reglas o pasos a seguir que permiten obtener una solución válida, suficientemente buena y con bajo uso de recursos al problema en estudio.

En la sección 2.1 de este capítulo se desarrolla tres de los métodos metaheurísticos más utilizados para la resolución de problemas RCMPSP. En la sección 2.2 se describe el funcionamiento de las heurísticas basadas en regla de prioridad y se presentan ejemplo de priorización de decisiones para la programación de las actividades en problemas RCMPSP. La sección 2.3 describe los enfoques C-RCMPSP y D-RCMPSP. Finalmente, en la sección 2.4 se profundiza en la heurística P-SGS/MINSLK, se analiza su funcionamiento y principales características

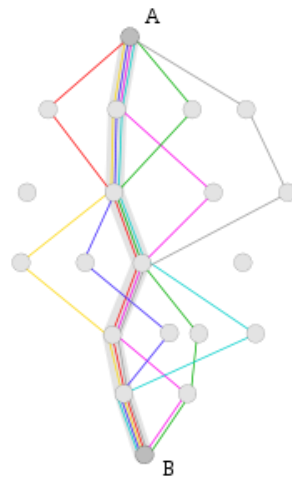
### 2.1 Método metaheurístico

Los métodos metaheurísticos son algoritmos que permiten obtener soluciones aproximadas a los problemas de optimización mediante una búsqueda iterativa en el espacio de soluciones. La búsqueda se realiza utilizando distintos conceptos generales, motivados de distintos conocimientos o comportamiento naturales, que permiten explorar el espacio de soluciones según sus preceptos (Herrera, 2010).

La mayoría de los algoritmos utilizados para la resolución de problemas RCMPSP son basados en el método metaheurístico de Algoritmos Genéticos (*Genetic Algorithm*) (Bredael & Vanhoucke, 2024). Este método simula el comportamiento de una población de individuos, la cual, mediante mutaciones de una generación a la siguiente, logra desarrollar individuos más preparados para sobrevivir en su entorno. Replicando este comportamiento a los problemas de optimización, se espera mejorar en cada iteración una solución factible obtenida en primera instancia. Para esto se definen qué factores afectar obtener una mejor solución al problema (denominados operadores genéticos) y se alterarán los mismos (mutación) de forma aleatoria para obtener nuevas soluciones al problema, se seleccionan las mejores soluciones obtenidas y se descartan aquellas que no logren buenos resultados. Si no se cumple con el criterio de parada se repite nuevamente el procedimiento tomando del grupo de soluciones seleccionado como generación de partida. De esta forma generación a generación se realizan mutaciones para mejorar las soluciones obtenidas (Sanchez, 2011).

Otro de los métodos utilizados para la generación de algoritmos es denominado Optimización de Colonia de Hormigas (*Ant Colony Optimization*), como el desarrollado en (Naihui He & Zhang, 2022). Las hormigas salen de su colonia y comienzan la búsqueda de alimento alrededor, cuando encuentran alimento dejan feromonas que hacen que el resto de las hormigas sigan su camino.

Dependiendo la cantidad de feromona del camino es que tantas hormigas lo tomarán, por lo tanto, se priorizan aquellos caminos más cortos y de mayor tránsito que son lo que contienen mayor cantidad de feromonas. En la figura 8 se representa este comportamiento, donde el camino más corto entre A y B es el más transitado. Llevando esto a un problema de optimización, se tiene un agente computacional (hormiga) que genera una solución factible del problema en cada iteración, de esta manera se forma un grafo que representa los posibles caminos que las hormigas pueden recorrer. Entre cada nodo existe información para decidir la conveniencia de tomar ese camino en base a la calidad de la solución obtenida, finalmente, los caminos que conducen a peores soluciones no se priorizan y si aquellos caminos que llevan a mejores soluciones, y sobre estos se enfoca la generación de soluciones nuevas en la siguiente iteración (Pérez & Otros, 2019).



**Figura 8:** Representación del método Optimización de Colonias de Hormigas. Fuente: (Berzal, 2023)

Además, se han desarrollado algoritmos de resolución de problemas RCMPSP basados en Optimización por Enjambre de Partículas (*Particle Swarm Optimization*) como el desarrollado en (Peng & Liu, 2024). Este método de resolución de problemas se basa en el comportamiento de los enjambres de insectos, los que al momento de recolectar polen priorizan la región de mayor densidad de flores. Llevando esto a la resolución de problemas de optimización, se comienza a trabajar con un conjunto de soluciones factibles, posteriormente se crean nuevas soluciones a partir de estas, entonces a medida que se encuentran mejores se prioriza la creación de nuevas soluciones en su entorno, generando un área de alta densidad de soluciones.

Todos estos métodos permiten facilitar la búsqueda de soluciones a los problemas complejos, se ha demostrado que permiten obtener soluciones de calidad y en tiempos razonables.

## 2.2 Reglas de prioridad

Los métodos heurísticos basados en reglas de prioridad definen distintos criterios para ordenar las decisiones relevantes para la obtención de una solución aproximada al problema de optimización, como puede ser la programación de las actividades y la asignación de recursos.

Las reglas de prioridad tienen como objetivo obtener una solución suficientemente buena o mejorar la solución obtenida en una siguiente iteración, por lo que los trabajos vinculados al desarrollo de heurísticas de este tipo pretenden identificar las reglas de prioridades más efectivas para la resolución del problema en análisis.

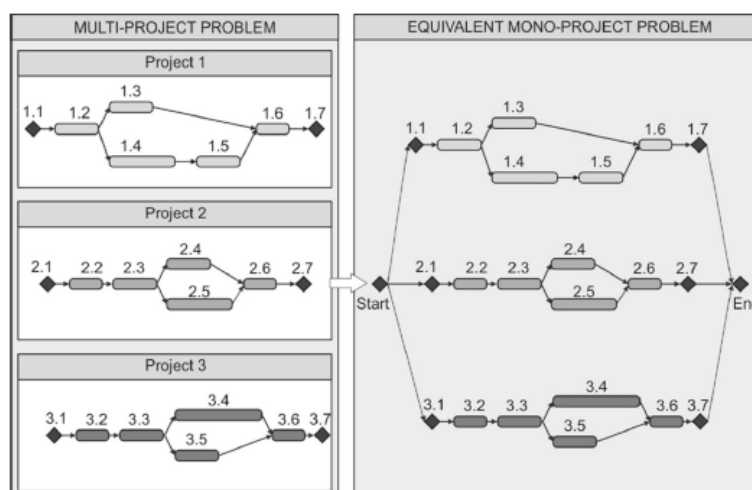
En el contexto de los problemas RCMPSP, en caso de no tener recursos suficientes para la programación simultánea de dos actividades, una regla de prioridad define un criterio para priorizar las actividades que reciben los recursos que necesitan frente a otras, menos prioritarias, que deberán esperar a que los recursos sean liberados. Algunos de estos criterios son:

- Actividad que pertenezca al proyecto que finaliza último.
- Actividad que tenga holgura mínima, calculada como el tiempo de finalización más tardío menos el tiempo de comienzo más tardío.
- Actividad que tenga mayor número de tareas sucesoras.
- Actividad que consuma menor cantidad de recursos.

## 2.3 Enfoques C-RCMPSP y D-RCMPSP

Profundizando en los problemas RCMPSP, se pueden distinguir dos casos, los problemas centralizados (C-RCMPSP) y los problemas descentralizados (D-RCMPSP). Los problemas centralizados consideran la creación de un único proyecto a partir de las actividades de los proyectos individuales, por lo que este enfoque funciona bien para obtener soluciones globales como el TMS (*the total makespan*), en este caso se obtiene un problema RCPSP donde existe un único camino crítico de la cartera de proyecto. El enfoque descentralizado considera a los proyectos de forma individual, por lo que la holgura de las actividades y el camino crítico es obtenido del proyecto particular, lo que permite mayor control sobre cada proyecto (Villafáñez & Otros, 2018).

En la figura 9 se representa a la izquierda el enfoque D-RCMPSP donde existen tres proyectos individuales, y el enfoque C-RCMPSP a la derecha donde estos tres proyectos conforman un único mono proyecto equivalente.



**Figura 9:** Enfoque D-RCMPSP y C-RCMPSP. Fuente: (Villafáñez & Otros, 2018)

## 2.4 Heurística P-SGS/MINSLK

La heurística P-SGS/MINSLK es el punto de partida de este trabajo, es necesario comprender su funcionamiento en profundidad para luego ser codificado e integrado cómo un módulo de funcionamiento independiente en Microsoft Project, lo que es, el objetivo de este TFM.

P-SGS/MINSLK surge de un trabajo realizado por el grupo de investigadores ISISOC de la Universidad de Valladolid, la misma fue publicada en el trabajo “*A generic heuristic for multi-project scheduling problems with global and local resource constraints (RCMPSP)*” con fecha 16 de enero de 2018 en el volumen 23 la revista Soft Computing (Villafáñez & Otros, 2018).

Esta heurística ha sido aplicada a los problemas de la librería MPSPLib donde logró obtener buenos resultado al momento de la resolución de los problemas. La tabla 1 presenta los resultados obtenidos en comparación con los obtenidos por otros algoritmos publicados en MPSPLib, para los 140 problemas existente en la librería.

**Tabla 1:** Resultados de P-SGS/MINSLK en casos de la librería MPSPLib. Fuente: (Villafáñez & Otros, 2018)

	Resultado mejor	Resultado igual o mejor	Diferencia con el mejor resultado inferior a:			
			1%	2%	5%	10%
Cantidad de problemas	22	38	54	71	108	135
Porcentaje de problemas	16%	27%	39%	51%	77%	96%

---

La heurística P-SGS/MINSLK ha demostrado un resultado igual o mejor en el 27% de los casos, en el 96% de los casos la solución obtenida se encuentra dispersa únicamente un 10% de la mejor solución obtenida hasta el momento. Su excelente comportamiento en los casos aplicados, y la simplicidad de uso, demuestra su potencial para obtener soluciones suficientemente buenas de los problemas RCMPSP.

La heurística P-SGS/MINSLK puede ser dividida en los seis pasos siguientes, los que no son los mencionados en la publicación original, pero describen de forma más comprensible y modulable el su funcionamiento:

1. Crear una programación temporal: En primer lugar, se debe crear una programación temporal de la cartera de proyecto sin considerar la restricción de recursos, utilizando las metodologías tradicionales de programación como CPM.
2. Obtener un listado de actividades candidatas: Para cada momento de tiempo  $t$  se debe obtener un listado de actividades candidatas a programar. Las actividades candidatas son aquellas que potencialmente pueden comenzar en el momento  $t$ , es decir, que no tienen actividades predecesoras o bien todas sus predecesoras ya han finalizado.
3. Priorizar las actividades candidatas por holgura: Las actividades candidatas se ordenan de menor a mayor holgura, se deben programar primero aquellas actividades que tengan menor holgura.
4. Evaluar la disponibilidad de recursos: Para cada actividad candidata, se evalúa si se cuenta con recursos suficiente para su programación en ese momento. Si para todos los recursos necesarios para la ejecución de la actividad en análisis se tiene disponibilidad suficiente, se programa la actividad. Si no se tiene disponibilidad en al menos uno de los recursos necesarios para ejecutar la actividad, se reprograma la actividad para el momento  $t+1$ , posteriormente se debe pasar a la siguiente actividad de la lista de candidatas.
5. Ajuste de disponibilidad de recursos: En caso de que la actividad se programe para el momento  $t$ , se debe actualizar la disponibilidad de los recursos, restando los utilizados por la actividad, posteriormente se debe pasar a la siguiente actividad de la lista de candidatas.
6. Pase a momento  $t+1$ : Cuando no se tienen más actividades candidatas por analizar, se debe pasar el momento  $t+1$  y repetir el procedimiento.

Repitiendo este procedimiento en todos los momentos del proyecto se tendrá una programación con los recursos nivelados, es decir, que no exista sobreasignación de recursos en ningún momento.

La heurística parte de una solución inicial no factible construida a partir de programación individual de los proyectos sin considerar la restricción de recursos, y utiliza como regla de prioridad, para ordenar que actividad programar, la holgura total de las actividades. Este esquema de programación es el que le brinda buenos resultados a la heurística.

Es clave también el concepto de actividad candidata, este es un conjunto de actividades que cumplen cierto criterio para poder pertenecer (las que pueden comenzar en el momento bajo

análisis) y los siguientes pasos de la heurística se ejecutan únicamente sobre las actividades de este conjunto, por lo que es importante que el mismo sea correctamente formado. Al momento de llevar a código la heurística este aspecto es de especial atención, por lo que debe ser correctamente testeado asegurando su correcto funcionamiento.

El enfoque que utiliza la heurística P-SGS/MINSLK es el de problema centralizado (C-RCMPSP), por lo tanto, se genera la programación de un único proyecto a partir de la programación inicial de cada proyecto individual. Esto implica que se pierda información de las tareas en el entorno de su proyecto particular, como es la criticidad, y se gestiona la cartera de proyecto como un problema RCPSP, es decir, modelando toda la cartera como un único proyecto. Esto también es de destacar, este enfoque debe ser tenido en cuenta para la codificación de la heurística.

En resumen, hay cuatro aspectos claves de la heurística P-SGS/MINSLK que definen la misma y deben ser especialmente considerados al momento de su codificación:

1. Enfoque C-RCMPSP.
2. Parte de una primera programación que no tienen en cuenta la restricción de recursos.
3. Se forma un conjunto de actividades candidatas.
4. La principal regla de prioridad es la holgura de las actividades.

---

## Capítulo 3 Programación en software de proyectos

En el presente capítulo se presenta la programación de proyectos en herramientas informáticas. Se comienza en la sección 3.1 con una introducción al uso de herramientas para la gestión de proyecto, y en la sección 3.2 se introduce Microsoft Project. En la sección 3.3 se describen funcionalidades relevantes de Project para la gestión de proyectos con restricción de recursos. En la sección 3.4 se describe el desarrollo de automatizaciones para la agilizar la gestión de proyectos, se profundiza en el lenguaje VBA, utilizado para el desarrollo de la herramienta objetivo de este TFM.

### 3.1 Introducción

Un software de gestión de proyectos es una herramienta que permite la planificación, seguimiento y control de los proyectos. Toma como datos de partida información relevante sobre las tareas planificadas, su relación de precedencia, duración, costo, uso de recursos, entre otras, y de manera sencilla obtiene la planificación del proyecto e información relevante para su seguimiento y control. Hoy en día el uso de este tipo de herramientas es esencial para todo director de proyecto.

Existen varios sistemas de gestión de proyectos disponibles en el mercado. De los más utilizados son Asana, Trello, Microsoft Project y Jira (Torres Sipion, 2024). Algunos de los factores más relevantes que debe considerar una organización al momento de seleccionar su sistema de gestión de proyecto son (Bernal, 2020):

- Metodología de gestión de proyectos: Es importante tener en cuenta la metodología de gestión de la organización, existen sistemas mejores adaptados para metodologías tradicional o para metodologías ágiles.
- Procesos que soporta el sistema: Son las funcionalidades que necesitamos que el sistema cuente, por ejemplo, la gestión de costos, gestión de recursos, documentación, gestión de incidentes, etc.
- Accesibilidad: Factor que toma en cuenta la disponibilidad de la herramienta en distintos soportes como web, aplicación móvil, trabajo colaborativo, etc. Aspecto clave para los equipos de trabajo actuales, donde los miembros se encuentran dispersos.
- Integración: Se debe evaluar la facilidad de flujo de información entre el sistema de gestión de proyectos y los restantes sistemas utilizados en la organización.
- Facilidad de aprendizaje: Es importante que el sistema sea de fácil aprendizaje para los miembros del equipo de gestión de proyectos, se priorizaran aquellos sistemas con documentación sobre su uso, formaciones y que cuenten con foro de usuarios.

### 3.2 Microsoft Project

Microsoft Project (MS Project) es un software desarrollado y comercializado por Microsoft, la primera versión fue lanzada en el año 1984 para el sistema operativo DOS y en el año 1990 fue lanzada la primera versión para Windows. A lo largo de los años el sistema ha ido mutando e

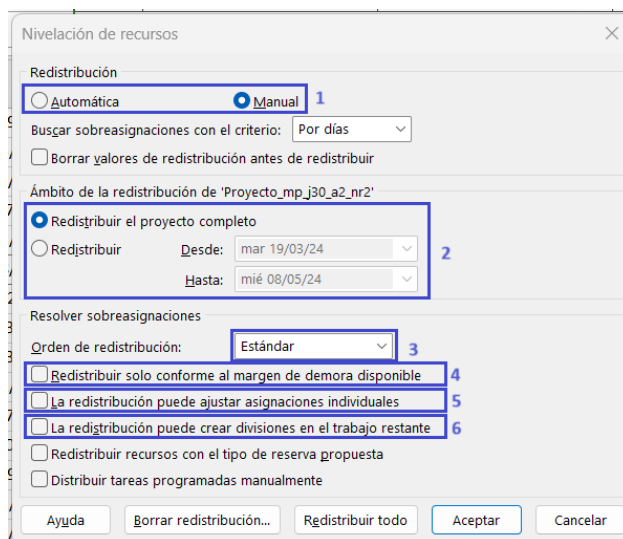
incorporando nuevas funcionalidades, hoy en día es un sistema de gestión de proyectos potente con funcionalidades claves para la dirección de proyectos.

Las principales funcionalidades que incluye Microsoft Project son la creación y asignación de tareas, planificación y programación, informes y análisis, diagramas de Gantt, obtención de KPI, gestión de recursos, entre otras. Originalmente creado para la gestión de proyectos con una metodología de cascada, ha ido integrando funcionalidades que hoy en día le permiten la gestión de proyectos ágiles, utilizando metodologías como Kanban o Scrum.

Microsoft Project se encuentra disponible en su versión de escritorio y en versión web. Se integra fácilmente con cualquier otra herramienta del ecosistema Microsoft, por este mismo motivo, es un sistema de fácil aprendizaje para el usuario, ya que su experiencia de usuario y usabilidad es alineada con todas las herramientas de Microsoft, de amplio uso personal o en organizaciones.

### 3.3 Programación con restricción de recursos en Microsoft Project

Respecto a la programación de proyecto con limitación de recursos, Microsoft Project incluye una herramienta que permite obtener una programación factible de los proyectos considerando el uso de los recursos de sus actividades y la disponibilidad de estos. Esta herramienta es conocida como “Nivelación de recursos” o “Redistribución de recursos”. En la figura 10 se muestra la pantalla que accede el usuario para la nivelación de recursos.



**Figura 10:** Ventana de nivelación de recursos en Microsoft Project.

MS Project ofrece dos opciones de nivelación de los recursos, manual o automática (1). La opción de distribución de recursos manual permite al usuario elegir el momento en el cual realizar la nivelación de los recursos, mientras que la distribución automática la realiza de forma instantánea siempre que se realice un cambio en los recursos asignados a las tareas o la disponibilidad de estos. A lo largo de este trabajo desarrollaremos el caso de nivelación de forma manual, esto permite un mayor control sobre la programación de las tareas.



---

Entonces, el usuario debe identificar en primera instancia si existe sobreasignación de recursos y en qué periodo de tiempo. Microsoft Project permite realizar una reprogramación total del proyecto o en un intervalo de tiempo (2).

Además, ofrece otras opciones sobre las que se debe decidir para tener en cuenta al momento de realizar la distribución. En primer lugar, el orden de distribución (3) donde se puede seleccionar tres opciones “Sólo identificador”, permite reprogramar las actividades de acuerdo con su identificador de forma ascendente, “Estándar”, toma en cuenta varios factores como dependencias, fechas y prioridades para reprogramar las actividades, y “Prioridad/Estándar”, la que en primer lugar evalúa criterios de prioridad de la actividad y luego los criterios estándares para reprogramarlas.

También se puede decidir si la redistribución debe hacerse únicamente dentro de las fechas de proyecto ya establecidas (4), para así evitar retraso de este. También permite la modificación de la asignación de recursos a las actividades para resolver la sobreasignación, por ejemplo, permitiendo que se ajuste la asignación de un recurso a una actividad, extendiendo la duración de la tarea o reasignando parte del trabajo a otro recurso disponible (5).

En ocasiones las opciones que ofrece para la resolución de los conflictos de sobreasignación de recursos no son siempre aplicables en la práctica, como por ejemplo dividir una actividad, cortándola en un momento luego de iniciada su ejecución y reprogramando lo restante para cuando se tenga recursos necesarios (6).

El algoritmo que utiliza Microsoft Project para la nivelación de los recursos es desconocido, por lo tanto, el director del proyecto pierde control sobre la programación del proyecto, ya que desconoce las reglas de prioridad que permiten decidir que tarea programar en lugar de otra.

### **3.4 Desarrollo de automatizaciones en Microsoft Project**

Microsoft Project, como todo sistema del ecosistema de Microsoft, permite el desarrollo de automatizaciones llamadas macros. Una macro es utilizada principalmente para automatizar un proceso o tarea repetitiva que lleva adelante el usuario, o bien un proceso complejo que es de difícil ejecución manual. Los principales beneficios de desarrollar macros son la eficiencia, los procesos automatizados se ejecutan en mucho menor tiempo que de forma manual, y la minimización de riesgo de error del usuario, una automatización debe ser probada y una vez se tiene confianza de su calidad repite el exactamente el mismo algoritmo todas las veces que es ejecutada.

El desarrollo de las automatizaciones en el ecosistema de Microsoft se realiza con el lenguaje de programación VBA (Visual Basic for Applications) utilizando como entorno de programación el propio sistema sobre el cual se ejecutará el proceso automatizado.

#### **3.4.1. VBA (Visual Basic for Applications)**

VBA incluye conceptos de un lenguaje orientado a objetos y algunos de sus principales beneficios. Un objeto es una representación de una entidad del entorno del sistema, como puede ser una actividad, que tiene distintas propiedades a las que se puede acceder para obtener información o

modificar la misma. Los lenguajes orientados a objetos son de más fácil escritura, mantenimiento y escalabilidad.

Para una mejor comprensión el trabajo realizado en este TFM es necesario conocer algunos aspectos relevantes del lenguaje de programación VBA, ya que con este se realizarán los desarrollos que son parte principal del entregable de este trabajo. A continuación, se presenta la estructura de los elementos más relevantes, en cursiva se presentan las palabras claves de VBA, las que son reservadas por el lenguaje y no deben modificarse, mientras que de color azul los datos de la estructura que el usuario debe modificar teniendo en cuenta las buenas prácticas para el desarrollo de códigos.

#### **3.4.1.1 Definir variables**

La instrucción *Dim* permite declarar y asignar espacio de almacenamiento a una o varias variables.

*Dim* nombreVariable As tipoVariable

El valor de *nombreVariable* hace referencia al nombre que el usuario decide asignar a la variable que está declarando, mientras que *tipoVariable* hace referencia al tipo de la variable en cuestión. Dentro de los tipos de variables más comunes se encuentran *String*, *Date*, *Collection*, *Project*, *Integer*, *Double*, *Workbook*, *Worksheet*.

#### **3.4.1.2 Instrucción Sub**

La instrucción *Sub* permite declarar un procedimiento el cual contiene un bloque de código que no devuelve un valor, a diferencia de una función.

*Sub* nombreProcedimiento()

[Conjunto de funcionalidades del código]

*End Sub*

El valor de *nombreProcedimiento* es el nombre que se le da al procedimiento que se está declarando, este valor es relevante ya que luego dará nombre al módulo de Microsoft Project que contiene estas funcionalidades. Cada macro está formado por un módulo que a su vez contiene un procedimiento *Sub*.

#### **3.4.1.3 Instrucción Function**

La instrucción *Function* es similar a la instrucción *Sub*, pero *Function* permite devolver un valor obtenido a partir de las funcionalidades en su estructura.

*Function* nombreFuncion() As tipoVariable

[Conjunto de funcionalidades del código]

---

### *End Function*

El valor de **nombreFuncion** es el nombre que se le da a la función que se está declarando. El **tipoVariable** es el tipo de dato que la función va a devolver, de los más comunes se encuentran *String*, *Integer* y *Double*. *End Function* marca el final de la función.

#### **3.4.1.4 Instrucción For**

La instrucción *For* permite repetir la ejecución de un bloque de código un numero establecido de veces.

*For* **nombreVariable** = **limiteInferior** *To* **limiteSuperior**

[Conjunto de funcionalidades del código]

*Next* **nombreVariable**

El valor de las variables **limiteInferior** y **limiteSuperior** al momento de ejecutar la instrucción *For* definirá la cantidad de veces que se ejecutará las funcionalidades que incluye. Las tres variables presentes deben ser definidas del tipo *Integer* para el correcto funcionamiento de la instrucción.

#### **3.4.1.5 Instrucción For Each**

La instrucción *For Each* permite recorrer los elementos de un conjunto, por ejemplo, una colección, un arreglo o un objeto, resulta particularmente útil cuando se trabaja con este tipo de datos para ejecutar líneas de código para cada uno de sus elementos, de forma más sencilla y optima que utilizando *For*.

*For Each* **nombreVariable** *In* **nombreConjunto**

[Conjunto de funcionalidades del código]

*Next* **nombreVariable**

La variable **nombreVariable** representa cada uno de los elementos del conjunto **nombreConjunto**, por lo que se puede obtener sus propiedades utilizando la sintaxis de programación basada en objetos a partir de la variable **nombreVariable**.

#### **3.4.1.6 Instrucción If Then Else**

La instrucción conocida como *If* o *If Then Else* es el condicional más conocido en la programación, permite ejecutar bloques de código si se cumple o no cierta condición establecida. Su sintaxis en VBA es la siguiente.

*If* **condición** *Then*

[Conjunto de funcionalidades del código si se cumple la condición]

*Else*

[Conjunto de funcionalidades del código si no se cumple la condición]

*End If*

#### **3.4.1.7 Instrucción *While***

La instrucción *While* permite la ejecución repetida de un bloque de código siempre que se esté cumpliendo una condición dada, para cada vez que se ejecuta el código corrobora el cumplimiento de esta condición y si se cumple lo ejecuta, de lo contrario no ejecuta y continua con el código fuera del bloque *While*.

*While* [condición](#)

[Conjunto de funcionalidades del código]

*Wend*

#### **3.4.1.8 Instrucción *InputBox***

La instrucción *InputBox* permite la interacción con el usuario para que ingrese datos en tiempo de ejecución de código.

`variableDatos = InputBox(InformaciónParaUsuario)`

De esta forma aparecerá en pantalla una ventana emergente donde el usuario ingrese el dato que se le solicite según la [InformaciónParaUsuario](#), y se guardará en la variable `variableDatos` para ser utilizado en el código restante.

---

## Capítulo 4 Herramienta desarrollada

En este capítulo se describe el entregable principal de este TFM: una herramienta desarrollada para programar proyectos con restricción de recursos siguiendo los pasos y reglas de prioridad de la heurística P-SGS/MINSLK. Esta herramienta ha sido desarrollada utilizando el lenguaje de programación VBA para ser ejecutada en Microsoft Project de forma totalmente integrada al sistema.

En un inicio, este capítulo se divide en función de los cuatro módulos principales que incluye esta herramienta. Cada sección describe en detalle las principales líneas de código que componen cada uno de los módulos, con el objetivo de una mejor comprensión del funcionamiento de la herramienta, y que aquel lector más avanzado sea capaz de entender la base de programación y realizar modificaciones o mejoras en caso de que entienda necesario. Los módulos son:

1. **Crear tareas:** El objetivo de este módulo es que el usuario obtenga una programación inicial de un proyecto en Microsoft Project a partir de un archivo Excel que contenga información sobre las tareas que componen el proyecto, es decir, nombre tarea, sucesoras, duración, uso de recurso de cada tarea y disponibilidad de cada recurso del proyecto. Este módulo automatiza la creación de proyectos, particularmente para este TFM se tiene necesidad de agilizar este proceso para realizar pruebas eficientes de los escenarios que proporciona la librería MPSPLib.
2. **Obtener información de proyectos y unificar cartera:** Este código permite obtener la información de distintos proyectos ya programados en Microsoft Project y unificar toda la programación en un único proyecto nuevo, el de la cartera, al que luego se nivelará sus recursos. Este módulo implementa el enfoque C-RCMPSP de la heurística P-SGS/MINSLK obteniendo una programación inicial de la cartera sin considerar la restricción de recursos.
3. **Gestor de recursos:** Este es el módulo principal de la herramienta, permite realizar la nivelación de los recursos del proyecto. Incorpora la selección de actividades candidatas, su evaluación según criterio de prioridad y las evaluaciones necesarias para decidir si programarla o no en el periodo.
4. **Gestión financiera:** Este módulo incorpora la gestión de recursos económicos en la evaluación de la programación de la cartera. A partir de la línea base de financiación planificada permite conocer si el costo de la cartera de proyectos excede la financiación para cada momento de tiempo. El módulo devuelve un archivo indicando para cada momento si la financiación es suficiente o no, pero no ajusta la programación en caso de que la misma sea insuficiente. Este funcionamiento se entiende conveniente ya que se prioriza la evaluación y disponibilidad de los recursos humanos y materiales del proyecto sobre los financieros.

Además, en este capítulo se incluye dos secciones finales. La sección 4.5, en la que se presenta los pasos a seguir para compartir entre usuarios los archivos que conforman la herramienta y su instalación en Microsoft Project, y la sección 4.6 que contiene un breve instructivo de uso.

## 4.1 Módulo Crear Tareas

El módulo llamado Crear Tarea, como se menciona antes, tiene como principal funcionalidad obtener la información de las tareas de un proyecto contenida en una plantilla de Excel y, a partir de esta, crear la programación en Microsoft Project.

El pseudocódigo de este módulo es esencialmente el siguiente:

- ⇒ Se abre una instancia para leer datos del archivo Excel
- ⇒ Usuario ingresa información de la plantilla a cargar
- ⇒ Se crea las tareas en MS Project con información básica (nombre y duración)
  - ⇒ Incorpora información de tareas sucesoras
  - ⇒ Incorpora información de uso de recursos
  - ⇒ Incorpora información de disponibilidad de recursos
- ⇒ Se cierra instancia de archivo Excel

A continuación, se describen los principales bloques de código que componen este módulo con el objetivo de una mejor comprensión de su estructura.

### 4.1.1. Definición de la instancia que representa el proyecto

```
(1) Dim proyecto As Project
(2) Set proyecto = ActiveProject
```

En la línea de código (1) se crea una nueva variable de tipo proyecto de MS Project, en la línea (2) se asigna a la variable creada el proyecto actualmente activo. La variable *proyecto* hace referencia a el proyecto de Microsoft Project actualmente activo, operando con ella se puede acceder a toda la información del proyecto. Este procedimiento se repite al inicio de cada módulo.

### 4.1.2. Declaración de variables

```
(3) Dim rutaArchivo As String
(4) Dim libro As Workbook
(5) Dim hoja As Worksheet
(6) Dim fila As Integer
(7) Dim tareaNueva As Object
(8) Dim texto As String
(9) Dim cantSucesoras As Integer
(10) Dim sucesoras As String
(11) Dim recursoUno As String
(12) Dim recursoDos As String
```

```
(13) Dim recursoTres As String
(14) Dim recursoCuatro As String
```

En las anteriores líneas de código se definen las principales variables que se utilizarán a lo largo del módulo. Las que destacan:

- rutaArchivo (3), representa el directorio del archivo de Excel donde se encuentra la información del proyecto.
- libro (4) y hoja (5), representan el archivo de Excel y la hoja del archivo donde se encuentra la información de las tareas del proyecto.

#### 4.1.3. Obtención de información de archivo Excel

```
(15) rutaArchivo = InputBox("Ingrese la ruta del archivo de excel del cual
se obtendrá la información del proyecto", "Ingresar datos")
(16) Set libro = Workbooks.Open(rutaArchivo)
(17) Set hoja = libro.Sheets("Hoja1")
```

Para la obtención de la información de la plantilla de Excel se definen las anteriores tres líneas de código. En la línea (15) se genera un cuadro de dialogo para que el usuario ingrese la dirección del archivo de Excel que contiene la información del proyecto, en la línea (16) se abre el archivo y se guarda su información en la variable libro y en la línea (17) se accede a la información de la hoja "Hoja1" del archivo de Excel. Operando con la variable hoja se puede acceder a toda la información obtenida del archivo Excel.

En la sección ANEXO I Plantillas Excel se presenta el formato de la plantilla de Excel para guardar la información del proyecto, el que debe ser respetado para un funcionamiento correcto del proceso de obtención de datos.

#### 4.1.4. Creación de tareas

```
(18) n = 1
(19) For fila = 2 To hoja.Cells(Rows.Count, 1).End(xlUp).Row
(20) Set tareaNueva = proyecto.Tasks.Add()
(21) tareaNueva.Name = "Tarea " & hoja.Cells(fila, 1).Value
(22) tareaNueva.Duration = hoja.Cells(fila, 4).Value & "d"
(23) tareaNueva.Manual = False
(24) n = n + 1
(25) Next fila
```

Las líneas de código anteriores recorren cada una de las filas del archivo de Excel y crea las tareas mediante la sentencia `proyecto.Tasks.Add()`, se le asigna nombre (21), duración (22) y programación automática (23).

#### 4.1.5. Incorporación de sucesoras

```
(26) cantSucesoras = CInt(hoja.Cells(fila, 2).Value)
(27) sucesoras = hoja.Cells(fila, 3).Value
(28) If cantSucesoras <> 0 Then
(29)     If cantSucesoras = 1 Then
(30)         tarea.Successors = sucesoras
(31)     Else
(32)         If cantSucesoras = 2 Then
(33)             tarea.Successors = Left(sucesoras, 2) & ";" &
Right(sucesoras, 2)
(34)         Else
(35)             tarea.Successors = Left(sucesoras, 2) & ";" & Mid(sucesoras,
4, 3) & ";" & Right(sucesoras, 2)
(36)         End If
(37)     End If
(38) End If
```

Las líneas anteriores de código se ejecutan dentro de un bucle *ForEach* que recorre todas las actividades existentes en el proyecto. Para cada actividad creada anteriormente se obtiene la cantidad de sucesoras y una cadena de texto con la identificación de que actividades son sucesoras (líneas (26) y (27)), posteriormente se comienza a trabajar en la cadena de texto de acuerdo con la cantidad de tareas sucesoras para obtener la información de los identificadores de las actividades separados por punto y coma, para realizar la asignación en la línea (35).

Por ejemplo, la cadena de texto original muestra que las actividades sucesoras son las que tienen identificación 1, 5 y 24 con el siguiente formato "1 5 24", para la asignación de las actividades sucesoras en Project es necesario convertir la cadena de texto al siguiente formato "1; 5; 24".

Es importante destacar que este código funciona correctamente para un máximo de tres tareas sucesoras para una tarea.

#### 4.1.6. Incorporación de recursos

```
(39) recursoUno = hoja.Cells(fila, 5).Value
(40) recursoDos = hoja.Cells(fila, 6).Value
(41) recursoTres = hoja.Cells(fila, 7).Value
(42) recursoCuatro = hoja.Cells(fila, 8).Value

(43) Dim textoRecursos As String
(44) textoRecursos = ""

(45) If CInt(recursoUno) > 0 Then
(46)     textoRecursos = hoja.Cells(1,5).Value & "[" & recursoUno & "00%];"
(47) End If

(48) If CInt(recursoDos) > 0 Then
```



```

(49)     textoRecursos = textoRecursos & hoja.Cells(1,6).Value & "[" &
recursoDos & "00%];"
(50)     End If

(51)     If CInt(recursoTres) > 0 Then
(52)         textoRecursos = textoRecursos & hoja.Cells(1,7).Value & "[" &
recursoTres & "00%];"
(53)     End If

(54)     If CInt(recursoCuatro) > 0 Then
(55)         textoRecursos = textoRecursos & hoja.Cells(1,8).Value & "[" &
recursoCuatro & "00%]"
(55)     End If

(56)     tarea.ResourceNames = textoRecursos
(57)     fila = fila + 1

```

Las anteriores líneas de código se ejecutan a continuación del bloque 2.1.5, dentro del mismo bucle *For Each* para cada tarea del proyecto, y a su vez recorre cada fila del archivo Excel. En las líneas (39) a (42) se obtiene el valor a asignar a cada recurso a partir de la información del Excel, esta información es guardada en variables, las que luego se utilizan para crear la cadena de texto `textoRecursos` que define la asignación final que se le debe realizar a la tarea. Se incorpora el recurso a la cadena `textoRecursos` únicamente si el mismo tiene asignada una cantidad mayor a 0, líneas (45) a (55).

A modo de ejemplo, según la información de la plantilla de Excel de la asignación de recursos que se muestra en la figura 11, la actividad 3 debe tener asignado 900%, 0%, 800% y 800% de los recursos R 17, R 18, R 3 y R 19 respectivamente. Entonces a partir de la información del Excel la variable `textoRecursos` tomará el valor "R 17[900%]; R 3[800%]; R 19[800%]" para asignar los recursos en Project, omitiendo la información del recurso con asignación 0.

	A	B	C	D	E	F	G	H
1	jobnr.	#successors	successors	duration	R17	R18	R3	R19
4	3	3	5 6 7	4	9	0	8	8

**Figura 11:** Asignación de recursos de una actividad

Es importante destacar que este código funciona correctamente para un máximo de cuatro recursos por proyecto.

#### 4.1.7. Obtención de los recursos del proyecto

```

(58)     Dim cantidadRecurso As Integer
(59)     Dim costoEstandar As Integer
(60)     Dim costoPorUso As Integer

(61)     If recursoProy.Name = hoja.Cells(1, 9).Value Then
(62)         cantidadRecu = hoja.Cells(2, 9).Value

```

```

(63)         recursoProy.MaxUnits = cantidadRecu & "00"
(64)     End If

(65)     If recursoProy.Name = hoja.Cells(1, 10).Value Then
(66)         cantidadRecu = hoja.Cells(2, 10).Value
(67)         recursoProy.MaxUnits = cantidadRecu & "00"
(68)     End If

(69)     If recursoProy.Name = hoja.Cells(1, 11).Value Then
(70)         cantidadRecu = hoja.Cells(2, 11).Value
(71)         recursoProy.MaxUnits = cantidadRecu & "00"
(72)     End If

(73)     If recursoProy.Name = hoja.Cells(1, 12).Value Then
(74)         cantidadRecu = hoja.Cells(2, 12).Value
(75)         recursoProy.MaxUnits = cantidadRecu & "00"
(76)     End If

```

En esta sección del módulo se obtiene la información de disponibilidad de los recursos del proyecto. Por ejemplo, la información del nombre del primero de los recursos se obtiene en la fila 1 y columna 9, línea (62), y su disponibilidad de la fila 2 y la columna 9, línea (63). Se procede de igual forma para el resto de los recursos según la ubicación de la información en la plantilla de Excel.

#### 4.1.8. Cierre de libro de Excel

```
(77)     libro.Close
```

Al finalizar la obtención de datos del libro de Excel se cierra la instancia abierta al inicio, esto es importante para liberar la capacidad de procesamiento que se está ocupando en el archivo.

## 4.2 Módulo Obtener información de proyectos

Este módulo tiene como objetivo llevar a la práctica dos de los puntos más destacados de la heurística P-SGS/MINSLK, su enfoque C-RCMPSP y obtener una primera programación de proyectos sin tener en cuenta la restricción de recursos. Para lograr esto el usuario debe ingresar que proyectos son los que componen la cartera y el módulo unificará todos estos proyectos en un nuevo mono proyecto.

El pseudocódigo general de este módulo es el siguiente:

- ⇒ Usuario ingresa la cantidad de proyectos de la cartera
- ⇒ Para cada proyecto de la cartera
  - ⇒ Usuario ingresa la ubicación del archivo de MS Project del proyecto
  - ⇒ Se migra información básica de las tareas (nombre, inicio, fin, duración)
  - ⇒ Se migra información de taras predecesoras

- ⇒ Se migra información de uso de recursos
- ⇒ Para cada recurso del proyecto
  - ⇒ Usuario ingresa la disponibilidad, costo de uso y costo estándar
  - ⇒ Se asigna los datos ingresados al proyecto

A continuación, se describen las principales líneas de código que forman el módulo.

#### 4.2.1. Declaración de variables

```
(1) Dim cantidadDeProyectosUser As String
(2) Dim cantidadDeProyectosInt As Integer
(3) Dim nombreDeProyectoUser As String
```

En primer lugar, se declaran las variables más relevantes que se utilizarán en el módulo. En este caso son variables relacionadas a datos proporcionados por el usuario respecto a la cantidad de proyectos que conforman la cartera y la ubicación del archivo de Project de cada proyecto.

#### 4.2.2. Ingreso de cantidad de proyectos de la cartera

```
(4) cantidadDeProyectosUser = InputBox("Ingrese la cantidad de
proyectos", "Ingresar datos")
(5) cantidadDeProyectosInt = CInt(cantidadDeProyectosUser)
```

Las líneas de código (4) y (5) permiten que el usuario ingrese la cantidad de proyectos que componen la cartera, esta información se utiliza luego repetir la información a solicitar para cada proyecto. En la línea (5) se convierte el dato de cantidad de proyectos ingresado por el usuario de una *string* a una variable tipo *integer* para poder operar luego con el número.

A partir de aquí el código opera dentro de un bucle *For* de 1 al valor de la variable *cantidadDeProyectosInt*, donde la variable *i* representa el número de proyecto en el cual se encuentra la iteración.

#### 4.2.3. Ingreso de la ubicación del proyecto

```
(6) nombreDeProyectoUser = InputBox("Ingrese la ubicación del proyecto "
& i, "Ingresar datos")
```

La línea de código identificada con el número (6) genera un cuadro de diálogo donde el usuario ingresa el directorio donde se encuentra el archivo de MS Project de cada proyecto de la cartera.

#### 4.2.4. Crear instancia de proyecto

```
(7) Dim objApp As Object
(8) Dim proyectoUnitario As Object
(9) Set objApp = CreateObject("MSProject.Application")
(10) objApp.FileOpenEx nombreDeProyectoUser
(11) Set proyectoUnitario = objApp.ActiveProject
```

En las anteriores líneas se genera una nueva instancia de proyecto y se asigna la misma al proyecto ingresado por el usuario. La variable `proyectoUnitario` es la que almacena toda la información del proyecto individual que conforma la cartera y se debe operar con ella para obtener la información de este.

En las líneas (7) y (8) se declaran dos nuevas variables del tipo *Object*, posteriormente a la variable `objApp` se le asigna un objeto de MS Project (9) y abre el proyecto ingresado por el usuario (10).

Es importante mencionar aquí que la variable `proyecto` hace referencia al mono proyecto de la cartera de proyectos, mientras que la variable `proyectoUnitario` a cada uno de los proyectos individuales que conforman la cartera.

#### 4.2.5. Actualizador de Id de tareas

Previo a comenzar a migrar la información de un archivo de Project a otro que tiene toda la cartera de proyectos es necesario conocer si el archivo de la cartera ya cuenta con tareas creadas o no, esto es relevante al momento de asignar las tareas sucesoras, ya que en el proyecto individual el identificador de cada tarea es interno a ese proyecto, pero en la cartera su identificador es considerando las tareas ya creadas de otros proyectos.

```
(12) Dim contadork As Integer
(13) contadork = 0
(14) For Each tareak In proyecto.Tasks
(15)     contadork = contadork + 1
(16) Next tareak
```

En las líneas (12) y (13) se inicia la variable `contadork` que es la que va a guardar la cantidad de tareas ya existentes en el archivo de la cartera de proyectos. Para obtener el dato se realiza un bucle *For Each* sobre todas las tareas ya existentes y se suma 1 por cada tarea a la variable `contadork`.

#### 4.2.6. Migrar información básica de las tareas

Ya se está en condiciones de empezar a migrar la información del proyecto a la cartera de proyecto. Es importante mencionar que el siguiente bloque de código se ejecuta dentro del siguiente bucle *For Each*:

```
(17) For Each tareaU In proyectoUnitario.Tasks
```

Entonces la variable `tareaU` representa cada una de las tareas del proyecto unitario o proyecto individual, utilizando sus propiedades de objeto se puede acceder a datos relevantes.

```
(18) Dim tareaNueva As Object
(19) Set tareaNueva = proyecto.Tasks.Add(tareaU.Name)
(20) tareaNueva.Name = "P" & i & "-" & tareaNueva.Name
(21) tareaNueva.Start = tareaU.Start
(22) tareaNueva.Finish = tareaU.Finish
(23) tareaNueva.Duration = tareaU.Duration
```

En la línea (18) se define una nueva tarea y se crea la misma (19) en `proyecto`, es decir en el proyecto que representa la cartera. A esa nueva tarea se le asigna un nuevo nombre (20) con el formato P[número de proyecto]-[nombre de la tarea en su proyecto], luego se migra la fecha de inicio, fecha de fin y duración de las tareas.

#### 4.2.7. Migrar predecesoras

El trabajo para migrar las predecesoras es algo complejo, principalmente para lograr asignar a las predecesoras según su Id en la cartera de proyecto a partir de su Id en el proyecto individual. Aquí es donde se hace uso de la variable `contadrok`.

```
(24) Dim predecesorasTexto As String
(25) Dim nuevasPredeTexto As String
(26) predecesorasTexto = tareaU.Predecessors
(27) If Len(predecesorasTexto) > 0 Then
(28)     Dim arreglo() As String
(29)     arreglo = Split(predecesorasTexto, ";")
(30)     For j = LBound(arreglo) To UBound(arreglo)
(31)         Dim acumulador As Integer
(32)         If j = 0 Then
(33)             acumulador = arreglo(j) + contadrok
(34)             nuevasPredeTexto = acumulador
(35)         Else
(36)             acumulador = arreglo(j) + contadrok
(37)             nuevasPredeTexto = acumulador & ";" & nuevasPredeTexto
(38)         End If
(39)     Next j
(40)     Dim longitud As Integer
(41)     Dim ultimoElemento As String
(42)     longitud = Len(nuevasPredeTexto)
(43)     If longitud > 0 Then
(44)         ultimoElemento = Mid(nuevasPredeTexto, longitud)
(45)         If ultimoElemento = ";" Then
(46)             nuevasPredeTexto = Left(nuevasPredeTexto, longitud -
(47) 1)
(48)         End If
(48)     Else
```

```
(49)         nuevasPredeTexto = ""
(50)             End If
(51)     Else
(52)         nuevasPredeTexto = ""
(53)     End If
(54)     tareaNueva.Predecessors = nuevasPredeTexto
```

En las líneas (24) y (25) se definen dos variables relevantes, `predecesorasTexto` que representa las predecesoras según su id en el proyecto individual, y `nuevasPredeTexto` que representa las tareas predecesoras según el id de la cartera de proyectos.

En la línea (27) se evalúa el largo del texto `predecesorasTexto` para conocer si la tarea en análisis tiene tareas predecesoras, en caso de no tener se finaliza la ejecución en la línea (52). En caso de que la actividad tenga actividades predecesoras se genera un arreglo de tipo *string* (29) donde se va a guardar en cada posición el id de la actividad predecesora, según el proyecto individual. Posteriormente se recorre todos los elementos del arreglo actualizando este valor para que tenga el id correcto según la cartera de proyectos, y volviendo a formar una cadena de texto con los valores separados por punto y coma.

En la línea (31) se define la variable `acumulador` que es un entero que guarda el valor del id actualizada. Posteriormente esta variable es convertida a una cadena de texto donde cada id actualizado es separado por punto y coma, esto se realiza en las líneas (32) a (38). Finalmente puede suceder casos donde el final del carácter sea un “;”, para esto en las líneas (40) a (47) se evalúa el caso y en caso de que se cumpla que el ultimo carácter es “;” se lo quita. De esta forma se obtiene la variable `nuevasPredeTexto` con el formato correcto y se asigna las predecesoras en la línea (54).

A modo de ejemplo, un caso de modificación de predecesoras de este bloque de código puede ser el siguiente:

1. Se supone que las actividades predecesoras de una actividad de un proyecto individual son las actividades con Id 1, 5 y 9, entonces se tiene el dato de predecesoras según el siguiente formato: “1; 5; 9”
2. Si la cartera ya incluye actividades de otros proyectos, supongamos 20 actividades, ahora las actividades con Id 1, 5 y 9 en su proyecto tendrán los Id 21, 25 y 29. Entonces en primer lugar se forma el siguiente arreglo de números enteros [1, 5, 9], y luego a cada elemento se le suma 20, obteniendo el arreglo de Id actualizados [21, 25, 29].
3. Finalmente es necesario volver al formato de números separados por punto y coma para poder asignar las actividades predecesoras correctamente, entonces se convierte el arreglo anterior a la siguiente cadena de texto: “21; 25; 29”.

De esta forma se tiene la asignación de actividades predecesoras según la identificación de cada actividad en la cartera.

#### 4.2.8. Migrar asignación de recursos

```
(55) tareaNueva.ResourceNames = tareaU.ResourceNames
```

En la línea (55) se asignan los recursos de las tareas del proyecto unitario a la del proyecto de la cartera.

Hasta aquí se han migrado todas las tareas de cada uno de los proyectos unitarios y se ha formado un nuevo archivo de Microsoft Project que representa la cartera.

#### 4.2.9. Migrar disponibilidad de recursos

```
(56) Dim cantidadRecursosTotal As Integer
(57) cantidadRecursosTotal = proyecto.ResourceCount

(58) For Each recursoProy In proyecto.Resources
(59)     Dim cantidadRecurso As Integer
(60)     Dim costoEstandar As Integer
(61)     Dim costoPorUso As Integer

(62)     cantidadRecu = InputBox("Ingrese la cantidad de recurso
disponible del recurso " & recursoProy.Name & " en porcentaje: ", "Ingresar
datos")
(63)     costoEstandar = InputBox("Ingrese el costo por hora del
recurso " & recursoProy.Name & ": ", "Ingresar datos")
(64)     costoPorUso = InputBox("Ingrese el costo por uso del recurso
" & recursoProy.Name & ": ", "Ingresar datos")

(65)     recursoProy.MaxUnits = cantidadRecu
(66)     recursoProy.StandardRate = costoEstandar
(67)     recursoProy.CostPerUse = costoPorUso

(68) Next recursoProy
```

Las anteriores líneas de código permiten que el usuario ingrese la disponibilidad de cada uno de los recursos asignados a las actividades migradas (62), también su costo estándar (63) y costo por uso (64). El costo estándar es el costo en euros por hora de uso, el costo por uso es el costo base por uso del recurso, independiente del tiempo de uso.

Luego de obtenida la información se asignan los valores al proyecto en las líneas (65) a (67).

### 4.3 Modulo Gestor de Recursos

Este módulo es el núcleo de la herramienta desarrollada, es el que permite realizar la distribución de los recursos. Analiza y obtiene las actividades candidatas en cada momento y evalúa su programación según la regla de prioridad establecida por la heurística P-SGS/MINSLK.

Este módulo también genera un registro en formato .txt con las principales decisiones tomadas en cada momento.

El pseudocódigo general de este módulo es el siguiente:

- ⇒ Obtiene datos de la programación inicial del proyecto
- ⇒ Usuario ingresa si nivela todo el proyecto o un parte inicial
- ⇒ Para cada día de la programación del proyecto
  - ⇒ Obtiene el conjunto de actividades ya comenzadas
  - ⇒ Resta a la disponibilidad de recursos el consumido por actividades comenzadas
  - ⇒ Obtiene el conjunto de actividades candidatas
    - ⇒ Evalúa disponibilidad de recursos a actividades candidatas
    - ⇒ Si tiene disponibilidad suficiente se ajusta disponibilidad de recursos
    - ⇒ Si no tiene disponibilidad suficiente reprogramar la actividad
  - ⇒ Actualiza fecha de fin de proyecto

#### 4.3.1. Declaración e inicialización de variables

```
(1)Dim proyecto As Project
(2)Set proyecto = ActiveProject

(3)Dim fechaInicio As Date
(4)fechaInicio = proyecto.ProjectStart

(5)Dim fechaFin As Date
(6)fechaFin = proyecto.ProjectFinish

(7)Dim tarea As Task
(8)Dim fechaActual As Date
(9)fechaActual = fechaInicio
(10)    Dim fechaNueva As Date
(11)    Dim fechaDos As Date
(12)    Dim proyectoUltimo As String
(13)    Dim nombreArchivoGenerado As String
(14)    nombreArchivoGenerado = "proyecto1003"

(15)    Dim tareasCandidatas As Collection
(16)    Set tareasCandidatas = New Collection
```



```
(17) Dim tareasComenzadas As Collection
(18) Set tareasComenzadas = New Collection
```

Las anteriores líneas de código representan la declaración e inicialización de las principales variables del módulo. En las líneas (3) y (4) se obtiene la fecha de inicios del proyecto y en las filas (5) y (6) se obtiene la fecha de fin prevista para el proyecto, estas variables definirán luego entre que fecha se debe iterar inicialmente para aplicar la heurística. En las filas (8) a (12) se definen variables auxiliares que luego se utilizarán en funcionalidades de la herramienta. La variable `nombreArchivoGenerado` representa el nombre con el que se generará el archivo de registro de las acciones.

En las líneas (15) y (16) se define la variable `tareasCandidatas` como una colección de elementos, esta variable agrupará para cada momento el conjunto de actividades que cumplen con el criterio para ser candidatas para programar, según lo desarrollado en la sección 1.4. En las líneas (17) y (18) se define, de igual manera, el conjunto de actividades que ya han comenzado y deben seguir en el momento actual. A la disponibilidad de recurso inicial hay que restarle el consumo de recursos de estas actividades para obtener la disponibilidad real en cada momento.

#### 4.3.2. Iniciación de archivo de registro de acciones

```
(19) Set fs = CreateObject("Scripting.FileSystemObject")
(20) Set a = fs.CreateTextFile("C:\Users\agust\Desktop\" &
nombreArchivoGenerado & ".txt", True)
(21) a.WriteLine ("ESTE ARCHIVO ES UN RESUMEN DE LA EJECUCIÓN")
```

Estas líneas de código permiten generar el archivo .txt que deja registro de las acciones que realiza la heurística en cada actividad, y deja el primer registro en el archivo (21). Es importante aquí que el usuario modifique la dirección donde se creará el archivo, para que sea una carpeta de su directorio, se debe modificar la fila (20).

La sentencia `a.WriteLine("texto")` escribe el texto entre comillas en el archivo de texto definido en la variable `a`. Esta nomenclatura se utiliza en todos los módulos de la herramienta.

#### 4.3.3. Definir horizonte temporal a nivelar recursos

Al igual que Microsoft Project, la herramienta desarrollada cuenta con la posibilidad de nivelar todo el proyecto o bien una franja de tiempo deseada desde el día de inicio.

```
(22) Dim respuesta As VbMsgBoxResult
(23) respuesta = MsgBox("¿Desea nivelar todo el proyecto? Si selecciona
< No > puede nivelar los primeros dias del proyecto que usted indique.",
vbQuestion + vbYesNo, "Confirmación")
(24) If respuesta = vbYes Then
```

```
(25)         fechaDos = fechaFin
(26)     Else
(27)         Dim cantidadDias As Integer
(28)         cantidadDias = InputBox("Ingrese la cantidad de días que desea
nivelar ", "Ingresar datos")
(29)         fechaDos = DateAdd("d", cantidadDias, fechaInicio)
(30)     End If
```

En la línea (23) se le solicita al usuario si desea nivelar todo el proyecto o unos días en particular, desde el inicio del proyecto. En caso de nivelar todo el proyecto se realiza la siguiente igualdad `fechaDos = fechaFin`, de lo contrario se obtiene la cantidad de días que el usuario desea nivelar los recursos (28) y se calcula la nueva fecha de fin de la nivelación (29).

Esta funcionalidad es particularmente útil cuando el tiempo de procesamiento de la nivelación de recursos de todo el proyecto es demasiado largo, ya que le permite al usuario ir nivelando en secciones, lo que tiene un tiempo de procesamiento menor y permite visualizar resultados antes.

#### 4.3.4. Iteración en cada momento

A partir de aquí el código que se describe se ejecuta dentro de una sentencia *While* en función de las fechas obtenidas antes (31).

```
(31)     While fechaActual < fechaDos
```

Dentro del *While* se ejecuta un bucle *For* para recorrer todos los momentos del proyecto. Al final del bucle se actualiza el valor de la variable `fechaDos` (32).

```
(32)         For fechaActual = fechaInicio To fechaDos
```

Es necesario este esquema de programación ya que la fecha de fin del proyecto puede modificarse en cada iteración al reprogramar una tarea que no tiene recursos suficientes. Lo que sucede es que el bucle *For* mantiene fijo el rango de iteración y no es susceptible al cambio de la fecha de fin del proyecto, pero la sentencia *While* si, entonces una vez finalizada la ejecución del bucle *For* y si aún no se ha llegado al final del proyecto, la sentencia *While* permite volver al bucle *For* para continuar con la iteración en cada momento que falta.

#### 4.3.5. Actividades ya comenzadas

Previo a trabajar en el análisis de disponibilidad de recursos es necesario identificar que actividades son las que ya han comenzado en momentos anteriores y en el momento actual si o si deben ejecutarse, ya que a la disponibilidad de recurso total se le debe restar la cantidad de recurso ya comprometido para estas actividades.

```

(33)     For Each tarea In proyecto.Tasks
(34)         If tarea.Start < fechaActual And tarea.Finish >= fechaActual
Then
(35)             tareasComenzadas.Add tarea
(36)         End If
(37)     Next tarea

(38)     a.WriteLine ("LAS TAREAS YA COMENZADAS SON:")
(39)     For Each tarea In tareasComenzadas
(40)         a.WriteLine (tarea.Name)
(41)     Next tarea

```

Entonces para cada actividad del proyecto se evalúa si su fecha de inicio es anterior a la fecha actual y si su fecha de fin es igual o posterior a la fecha actual (34), de esta forma sabremos que es una actividad que si o si debe ejecutarse en el momento en análisis. En caso de cumplir con la doble condición la tarea es agregada al conjunto tareasComenzadas. En las líneas (38) a (41) se deja registro de todas las tareas que se identificaron como comenzadas en el archivo .txt de la ejecución.

#### 4.3.6. Actividades candidatas

Sobre el conjunto de actividades candidatas es que se ejecutará los pasos de la heurística, por lo que su correcta composición es clave para el funcionamiento de la herramienta.

```

(42)     For Each tarea In proyecto.Tasks
(43)         If tarea.Start = fechaActual Then '
(44)             tareasCandidatas.Add tarea
(45)         End If
(46)     Next tarea

```

La condición para que una actividad sea incorporada al conjunto de actividades candidatas es que su fecha de inicio sea igual a la fecha actual, entonces se recorre todas las actividades del proyecto y se agregan al conjunto tareasCandidatas aquellas que cumplen con dicha condición.

#### 4.3.7. Regla de prioridad por holgura

```

(47)     For i = 1 To tareasCandidatas.Count - 1
(48)         For j = 1 To tareasCandidatas.Count - 1
(49)             If tareasCandidatas(j).TotalSlack >
tareasCandidatas(j + 1).TotalSlack Then
(50)                 Set temp = tareasCandidatas(j)
(51)                 tareasCandidatas.Remove (j)
(52)                 tareasCandidatas.Add temp
(53)             End If
(54)         Next j

```

```

(55)     Next i

(56)     a.WriteLine ("LAS TAREAS CANDIDATAS SON:")
(57)     For Each tarea In tareasCandidatas
(58)         a.WriteLine (tarea.Name & ", holgura: " & tarea.TotalSlack)
(59)     Next tarea

```

En las líneas (47) a (55) se reordena el conjunto de actividades candidatas según la holgura de cada una, de esta forma se aplica la regla de prioridad que establece la heurística. Luego al momento de evaluar la disponibilidad de recursos y tomar la decisión de programar una actividad o no se realiza respetando el orden establecido en esta sección del código. Para obtener la información de holgura de una actividad se utiliza la propiedad TotalSlack.

#### 4.3.8. Arreglo de disponibilidad de recursos

Para guardar e ir actualizando la disponibilidad de cada recurso en cada iteración he definido un arreglo en el que guarda en cada posición la disponibilidad de cada recurso, este arreglo es llamado recursoDispArray.

```

(60)     Dim recursoDispArray() As Double
(61)     ReDim recursoDispArray(1 To cantidadRecursos)

(62)     Dim recursoDisp As Double
(63)     Dim k As Integer

(64)     k = 1
(65)     For Each recursot In proyecto.Resources
(66)         recursoDisp = recursot.MaxUnits
(67)         recursoDispArray(k) = recursoDisp
(68)         a.WriteLine ("El recurso " & recursot.Name & " tiene
disponibilidad de " & recursoDispArray(k))
(69)         k = k + 1
(70)     Next recursot

```

En las líneas (60) y (61) se define el arreglo, mientras que en las líneas (64) a (70) se inicializa sus valores a partir de la disponibilidad inicial total de cada recurso.

A modo de ejemplo, el proyecto con recursos R1, R2 y R3 con disponibilidad 5, 10 y 12 respectivamente, el arreglo recursoDispArray tendrá el valor de [5, 10, 12].

#### 4.3.9. Descuento de tareas comenzadas

```

(71)     For Each tareac In tareasComenzadas
(72)         k = 1
(73)         For Each recursot In proyecto.Resources
(74)             For Each asignacionc In tareac.Assignments

```

```

(75)         Set recursoc = asignacionc.Resource
(76)         unidadesc = asignacionc.Units
(77)         If recursoc.ID = recursot.ID Then
(78)             recursoDispArray(k) = recursoDispArray(k) -
unidadesc
(79)             a.WriteLine ("El recurso " & recursoc.Name & "
tiene disponibilidad de " & recursoDispArray(k) & " ?" & k & "? " & unidadesc)
(80)         End If
(81)     Next asignacionc
(82)     k = k + 1
(83) Next recursot
(84) Next tareac

```

Luego de obtener inicializado la variable con la disponibilidad total inicial de cada recurso, es necesario restar el consumo de recursos ya comprometido a las actividades comenzadas y que deben ejecutarse en el momento en análisis. Para esto se recorre todos los elementos del conjunto `tareasComenzadas` y se resta los recursos asignados a cada actividad del conjunto. De esta forma se actualiza los valores de disponibilidad del arreglo `recursoDispArray()`.

Continuando con el ejemplo anterior, si suponemos que se tiene actividades ya comenzadas que utilizan 4 y 5 unidades de los recursos R1 y R3, al finalizar estas líneas de código el arreglo `recursoDispArray` tendrá el valor [1, 10, 7].

#### 4.3.10. Nivelación de recursos

```

(85)     For Each tarea1 In tareasCandidatas
(86)         a.WriteLine ("Trabajando en la tarea: " & tarea1.Name)
(87)         Dim m As Integer
(88)         m = 1

(89)     For Each recursot In proyecto.Resources
(90)         Dim totalRecursos As Double
(91)         totalRecursos = 0
(92)         If tarea1.Assignments.Count > 0 Then
(93)             Dim asignacion As Assignment
(94)             For Each asignacion In tarea1.Assignments
(95)                 Dim recurso As Resource
(96)                 Set recurso = asignacion.Resource
(97)                 Dim unidades As Double
(98)                 unidades = asignacion.Units
(99)                 If recurso.ID = recursot.ID Then
(100)                    totalRecursos = totalRecursos + unidades
(101)                End If
(102)            Next asignacion
(103)        End If
(104)        If totalRecursos > recursoDispArray(m) Then
(105)            contadorSobreAsignado = contadorSobreAsignado + 1
(106)        End If

```

```

(107)         m = m + 1
(108)     Next recursot

(109)     If contadorSobreAsignado >= 1 Then
(110)         fechaNueva = DateAdd("d", 1, fechaActual)
(111)         tarea1.Start = fechaNueva
(112)     Else
(113)         k = 1
(114)         For Each recursot In proyecto.Resources
(115)             For Each asignacion In tarea1.Assignments
(116)                 Set recurso = asignacion.Resource
(117)                 unidades = asignacion.Units
(118)                 If recurso.ID = recursot.ID Then
(119)                     recursoDispArray(k) =
recursoDispArray(k) - unidades
(120)                 End If
(121)             Next asignacion
(122)             k = k + 1
(123)         Next recursot
(124)     End If

(125)     contadorSobreAsignado = 0
(126) Next tarea1

```

Esta sección del código realiza la nivelación de recursos, evalúa la disponibilidad de recursos necesarios para ejecutar cada actividad y la programa en caso de contar con los recursos suficientes.

El bloque de código se ejecuta dentro de un bucle *For Each* que recorre todas las actividades del conjunto de actividades candidatas. Entre las líneas (89) y (108) evalúa la disponibilidad de recursos de cada actividad, si al menos uno de los recursos asignados a la actividad no tiene disponibilidad suficiente la variable `contadorSobreAsignado` toma valor mayor o igual a uno, lo que indica que la actividad debe ser reprogramada.

En las líneas (109) a la (124) se evalúa el valor de la variable `contadorSobreAsignado` y decide en función al mismo. Si el valor es mayor o igual a 1 reprograma el inicio de la actividad para el día siguiente (110), en caso contrario, la actividad no se reprograma y ajusta la disponibilidad de recursos restando los consumidos por la actividad (119).

Este procedimiento se realiza para todas las actividades del conjunto de actividades candidatas.

Luego de realizar la nivelación de recursos de un momento, previo de pasar al siguiente, se vacían los conjuntos de actividades comenzadas y actividades candidatas.

```

(127)         Set tareasCandidatas = New Collection
(128)         Set tareasComenzadas = New Collection

```

## 4.4 Modulo Gestión Financiera

Este módulo incorpora la gestión de recursos económicos en la evaluación de la programación de la cartera. A partir de la línea base de financiación planificada permite conocer si el costo de la cartera de proyectos excede la financiación para cada momento de tiempo. El módulo devuelve un archivo indicando para cada momento si la financiación es suficiente o no, pero no ajusta la programación en caso de que la misma sea insuficiente, este funcionamiento se entiende conveniente ya que se prioriza la evaluación y disponibilidad de los recursos humanos y materiales del proyecto sobre los financieros.

### 4.4.1. Declaración de variables

```
(1) Dim recursoSumaCosto() As Double
(2) Dim recursoCostoAcumulado() As Double
(3) Dim recursoFinancieroDisp() As Double
(4) ReDim recursoSumaCosto(1 To m)
(5) ReDim recursoCostoAcumulado(1 To m)
(6) ReDim recursoFinancieroDisp(1 To m)
```

En las anteriores líneas se definen variables particulares de este módulo, tres arreglos que tienen tantos elementos como días planificados el proyecto. `recursoSumaCosto(t)` guarda la información de costos para cada día `t`, sumando los costos individuales de las actividades planificadas para ese día; la variable `recursoCostoAcumulado(t)` es el valor de costo acumulado de la variable anterior, representa la línea base de costo del proyecto; mientras que la variable `recursoFinancieroDisp(t)` es la financiación planificada para cada momento `t`, se obtiene de datos de una plantilla de Excel proporcionada por el usuario.

### 4.4.2. Línea base de financiación

```
(7) Dim rutaArchivo As String
(8) Dim libro As Workbook
(9) Dim hoja As Worksheet
(10) Dim fila As Integer

(11) rutaArchivo = InputBox("Ingrese la ruta del archivo de excel del cual
se obtendrá la financiación disponible en cada periodo", "Ingresar datos")
(12) Set libro = Workbooks.Open(rutaArchivo)
(13) Set hoja = libro.Sheets("Hoja1") ' Cambia "Sheet1" por el nombre de
tu hoja

(14) n = 1
(15) For fila = 2 To hoja.Cells(Rows.Count, 1).End(xlUp).Row
(16)     recursoFinancieroDisp(n) = hoja.Cells(fila, 2).Value
(17)     n = n + 1
(18) Next fila
```

```

(19)    n = n - 1
(20)    If m > n Then
(21)        Dim p As Integer
(22)        p = m - n
(23)        For i = 1 To p
(24)            recursoFinancieroDisp(n + i) = recursoFinancieroDisp(n + i -
1)
(25)        Next i
(26)    End If

```

El usuario debe ingresar los datos de la línea base de financiación para cada momento en plantilla de Excel, la que se muestra en la sección ANEXO I Plantillas Excel. La herramienta le solicitará la ubicación de su directorio de la plantilla y guarda en la “Hoja1” la información obtenida, líneas (11) a (13).

En las líneas (14) a (18) inicializa la variable `recursoFinancieroDisp(t)` a partir de los valores de la plantilla. Puede suceder que inicialmente se planificó la financiación para todo el proyecto, sin embargo, al reprogramar actividades para obtener los recursos nivelados el plazo total del proyecto se extiende y quede un periodo de tiempo sobre el final sin financiación planificada, para resolver esto en las líneas (19) a (26) se completa la etapa sin financiación con el ultimo valor ingresado por el usuario.

A modo de ejemplo, un proyecto inicialmente programado para tres días tiene la siguiente financiación (1, 5, 7), luego de nivelar los recursos el proyecto pasa a durar cinco días y su financiación planificada pasa a ser (1, 5, 7, 7, 7).

#### 4.4.3. Costo por periodo

```

(27)    For fechaActual = fechaInicio To fechaDos
(28)        If Weekday(fechaActual) <> vbSaturday And Weekday(fechaActual) <>
vbSunday Then

(29)            For Each tarea In proyecto.Tasks

(30)                If tarea.Start <= fechaActual And tarea.Finish >=
fechaActual Then
(31)                    tareasComenzadas.Add tarea
(32)                End If
(33)            Next tarea

(34)            For Each tareac In tareasComenzadas
(35)                For Each asignacionc In tareac.Assignments
(36)                    Set recursoc = asignacionc.Resource
(37)                    unidadesc = asignacionc.Units
(38)                    If unidadesc > 0 Then

(39)                        Dim posicion As Integer

```



```

(40)                                     Dim parteAntes As String
(41)                                     posicion = InStr(recursoc.StandardRate,
",")
(42)                                     parteAntes = Left(recursoc.StandardRate,
posicion - 1) ' Parte antes del carácter

(43)                                     If fechaActual = tareac.Start Then
(44)                                     recursoSumaCosto(k) = recursoc.CostPerUse *
unidadesc + recursoSumaCosto(k)
(45)                                     End If

(46)                                     recursoSumaCosto(k) = CInt(parteAntes) *
unidadesc * 8 + recursoSumaCosto(k)

(47)                                     End If
(48)                                     Next asignacionc
(49)                                     Next tareac

(50)     Else
(51)         recursoSumaCosto(k) = 0
(52)     End If

(53)     fechaFin = proyecto.ProjectFinish
(54)     Set tareasCandidatas = New Collection
(55)     Set tareasComenzadas = New Collection
(56)     k = k + 1

(57)     Next fechaActual

```

Para obtener el costo por día,  $\text{recursoSumaCosto}(t)$ , se recorren todos los días del proyecto, para cada día en primer lugar se evalúa si es sábado o domingo, en caso de que lo sea el costo asignado para ese día es 0 (51), de lo contrario se realiza el cálculo de costos.

En las líneas (30) a (33) se forma el conjunto de actividades que se están ejecutando en la fecha en análisis, sobre este conjunto se realiza el cálculo de costos. El costo total de cada actividad se compone del costo por uso, que se da el primer día de ejecución de la actividad, y el costo estándar que se da cada día que la actividad está en ejecución.

El costo estándar se obtiene (46) a partir de multiplicar los siguientes factores:

- 1) Costo por hora de una unidad de recurso, para obtenerlo es necesario operar sobre el dato para eliminar decimales y utilizar el número entero, líneas (39) a (42).
- 2) Unidades del recurso asignada a la actividad.
- 3) Las 8 horas laborales del día.

El costo por uso se obtiene en las líneas (43) a (45), cuando la fecha de evaluación es la del día de comienzo de la actividad se suma el costo por uso a la variable  $\text{recursoSumaCosto}(t)$ .

Finalmente se inicializan los conjuntos y variables y se pasa al cálculo del día siguiente.

#### 4.4.4. Línea base de costo y evaluación de disponibilidad financiera

```

(58)   For i = 1 To m
(59)       If i > 1 Then
(60)           recursoCostoAcumulado(i) = recursoSumaCosto(i) +
recursoCostoAcumulado(i - 1)
(61)           If recursoCostoAcumulado(i) < recursoFinancieroDisp(i) Then
(62)               a.WriteLine ("Costo acumulado periodo " & i & ": " &
recursoCostoAcumulado(i) & " -> RECURSO FINANCIERO SUFICIENTE ")
(63)           Else
(64)               a.WriteLine ("Costo acumulado periodo " & i & ": " &
recursoCostoAcumulado(i) & " -> RECURSO FINANCIERO INSUFICIENTE")
(65)           End If
(66)       Else
(67)           recursoCostoAcumulado(1) = recursoSumaCosto(1)
(68)           If recursoCostoAcumulado(i) < recursoFinancieroDisp(i) Then
(69)               a.WriteLine ("Costo acumulado periodo " & i & ": " &
recursoCostoAcumulado(i) & " -> RECURSO FINANCIERO SUFICIENTE")
(70)           Else
(71)               a.WriteLine ("Costo acumulado periodo " & i & ": " &
recursoCostoAcumulado(i) & " -> RECURSO FINANCIERO INSUFICIENTE")
(72)           End If
(73)       End If
(74)   Next i

```

Esta sección del módulo obtiene la línea base de costo `recursoCostoAcumulado(t)`, es decir el costo acumulado, a partir del costo diario total `recursoSumaCosto(t)`. Se itera en cada posición de las variables que representan la línea base de costo y la línea base de financiación, comparando los valores para cada momento, y se deja registro si la financiación es suficiente o no en el archivo `.txt` de salida.

## 4.5 Instalación de la herramienta en MS Project

La herramienta desarrollada consta de cinco elementos, cada uno contenido en un archivo distinto. Cuatro elementos son cada uno de los módulos desarrollados, el quinto elemento es la cinta de opciones con botones que permiten la ejecución de los módulos. Estos archivos se pueden compartir entre usuarios y ser instalados de forma que las funcionalidades queden completamente integradas en MS Project.

Los archivos que representan las funcionalidades de cada uno de los módulos tienen extensión `.bas`, para su instalación se debe acceder a MS Project y proceder según los siguientes pasos:

1. Acceder a menú Programador > Visual Basic
2. Clic derecho en "Módulos"

3. Clic en "Importar archivo"
4. Importar cada uno de los archivos .bas

Para instalar la cinta de opciones que contiene los botones para la ejecución de cada módulo se debe proceder según los siguientes pasos:

1. Clic derecho en la cinta de opciones, en cualquier pestaña o se puede crear una nueva
2. Clic en "Personalizar cinta de opciones"
3. Clic en el botón "Importar o Exportar"
4. Clic en la opción "Importar archivo de personalización"
5. Importar el archivo .exportedUI

De esta forma se accede a todas las funcionalidades de la herramienta desde Microsoft Project.

## 4.6 Uso de la herramienta

Para un correcto uso de la herramienta se elabora el siguiente instructivo en el que se detallan los pasos necesarios para la ejecución de los cuatro módulos desarrollados.

Al generar la cinta de opciones según se detalla anteriormente, se crean los cuatro botones, los que se visualizan en la figura 12. Cada uno de ellos permite ejecutar cada uno de los módulos desarrollados.



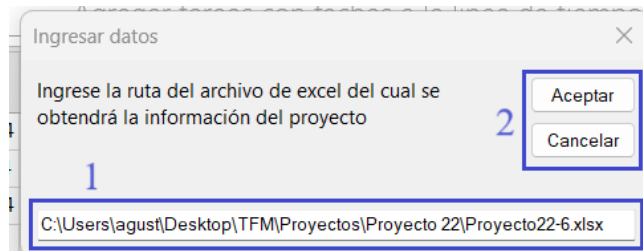
**Figura 12:** Botones de la cinta de opciones

### 4.6.1. Crear tareas

Previo a utilizar el módulo Crear Tareas, que permite crear las tareas de un proyecto a partir de información contenida en la plantilla de Excel, se debe tener disponible la información de las actividades que conforman el proyecto en dicha plantilla, según formato que se detalla en ANEXO I Plantillas Excel.

Para ejecutar el módulo se abre un nuevo archivo de MS Project y posteriormente se realiza clic sobre el botón CrearTareas de la cinta de opciones.

El usuario visualiza la ventana emergente que se presenta en la figura 13. En primer lugar, ingresa la ubicación del directorio de la plantilla de Excel sin entrecomillar el texto (1), luego realiza clic en el botón “Aceptar” para iniciar la carga de la información.



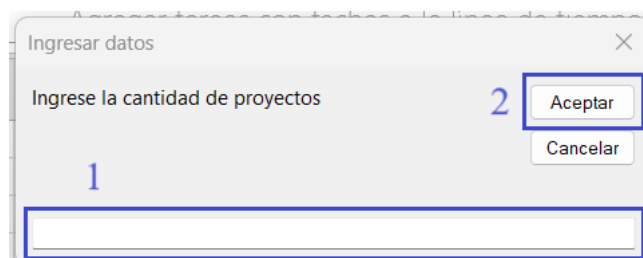
**Figura 13:** Ventana de ingreso de datos de módulo Crear Tareas

El resultado de este proceso es el archivo de MS Project con la información de las actividades que conforman el proyecto, nombre, duración, predecesoras, uso de recursos y disponibilidad de recursos del proyecto.

#### 4.6.2. Obtener información de proyectos

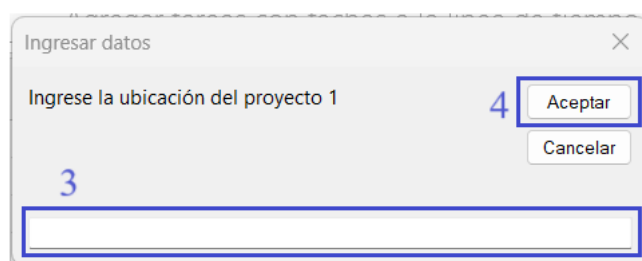
Para ejecutar el módulo Obtener Información, el que unifica los proyectos que conforman la cartera en un único mono proyecto, el usuario abre a un nuevo archivo de MS Project, ingresa a la cinta de opciones y realiza clic en el botón ObtenerInformacionProyectos.

El usuario visualiza la ventana emergente que se presenta en la figura 14. En esta ingresa en numero la cantidad de proyectos que conforman la cartera (1), luego realiza clic en el botón “Aceptar”.



**Figura 14:** Ventana de ingreso de cantidad de proyecto de módulo Obtener Información

Posteriormente para cada proyecto de la cartera ingresa la ubicación del archivo de MS Project que contiene toda la información del proyecto (3), para esto el usuario accede a la ventana emergente que se presenta en la figura 15.

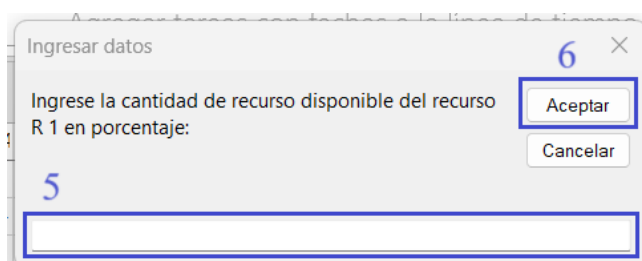


The screenshot shows a dialog box titled "Ingresar datos" with a close button (X) in the top right corner. The main text reads "Ingrese la ubicación del proyecto 1". Below this text is a large empty text input field. To the right of the input field, there are two buttons: "Aceptar" and "Cancelar". A blue number "3" is positioned to the left of the input field, and a blue number "4" is positioned to the left of the "Aceptar" button.

**Figura 15:** Ventana de ingreso de ubicación de cada proyecto que conforma la cartera

Para migrar la información del archivo MS Project del proyecto al nuevo archivo MS Project de la cartera realiza clic en el botón “Aceptar”.

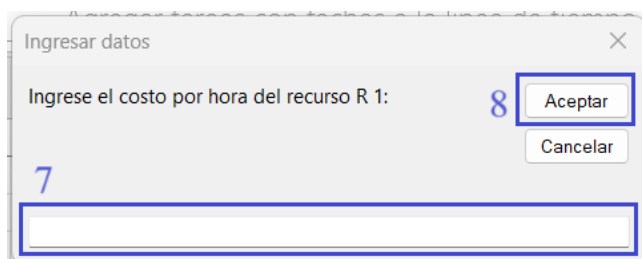
Luego de ingresar la ubicación de todos los proyectos que conforman la cartera, ingresa la información de disponibilidad de cada uno de los recursos asignados a las tareas y sus costos. Para esto el usuario visualiza las ventanas emergentes de la figura 16, figura 17 y figura 18.



The screenshot shows a dialog box titled "Ingresar datos" with a close button (X) in the top right corner. The main text reads "Ingrese la cantidad de recurso disponible del recurso R 1 en porcentaje:". Below this text is a large empty text input field. To the right of the input field, there are two buttons: "Aceptar" and "Cancelar". A blue number "5" is positioned to the left of the input field, and a blue number "6" is positioned to the left of the "Aceptar" button.

**Figura 16:** Ventana de ingreso de disponibilidad de recurso de módulo Obtener Información

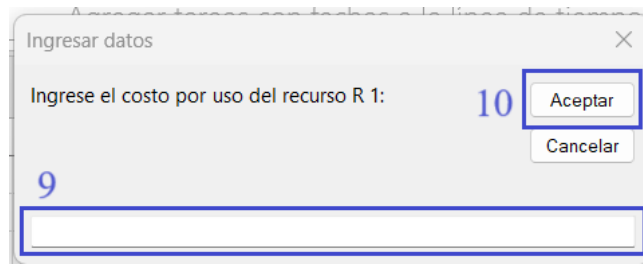
En la ventana emergente que se visualiza en la figura 16 ingresa la disponibilidad de cada recurso en porcentaje (5), es decir, si se cuenta con disponibilidad de 5 unidades de un recurso, se debe ingresar el valor 500. Posteriormente realiza clic en el botón “Aceptar” para continuar con el ingreso de información.



The screenshot shows a dialog box titled "Ingresar datos" with a close button (X) in the top right corner. The main text reads "Ingrese el costo por hora del recurso R 1:". Below this text is a large empty text input field. To the right of the input field, there are two buttons: "Aceptar" and "Cancelar". A blue number "7" is positioned to the left of the input field, and a blue number "8" is positioned to the left of the "Aceptar" button.

**Figura 17:** Ventana de ingreso de costo por hora de cada recurso

En la ventana emergente que se visualiza en la figura 17 ingresa el costo por hora de uso de cada recurso en valor numérico. Para continuar con el ingreso de información se realiza clic en el botón “Aceptar”.



**Figura 18:** Ventana de ingreso de costo por uso de cada recurso

Finalmente, en la figura 18 se visualiza la ventana emergente que permite al usuario el ingreso del costo por uso de cada recurso. El costo por uso es aquel que se da por única vez cada vez que se utiliza el recurso y se contabiliza el primer día de uso. Para finalizar el ingreso de información realiza clic en el botón “Aceptar”.

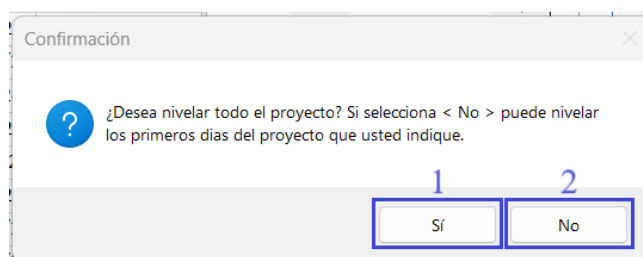
En cualquier paso del proceso el usuario tiene la posibilidad de cancelar la carga de información realizando clic en el botón “Cancelar” de la ventana emergente que se encuentre disponible.

El resultado de este proceso es un archivo de MS Project que contiene la información de todos los proyectos que conforman la cartera.

#### 4.6.3. Gestor Recursos

El módulo Gestor Recursos es aquel que realiza la nivelación de los recursos, obteniendo una programación de la cartera que cuente con disponibilidad de recursos en todo momento. Para ejecutar este módulo el usuario ingresa al archivo de MS Project del proyecto o cartera que desee nivelar los recursos, accede al menú de la cinta de opciones y realizar clic en el botón GestorRecursos.

A continuación, visualiza la ventana emergente que se presenta en la figura 19. En esta ventana se presenta las opciones de nivelar los recursos de todo el proyecto en un solo paso, si selecciona la opción “Sí” (1), o nivelar los recursos de un primer intervalo de tiempo del proyecto, si selecciona la opción “No” (2).

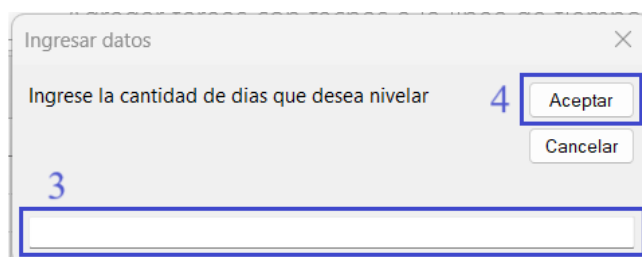


**Figura 19:** Ventana de selección de intervalo de tiempo de nivelación de recursos

Es importante tener en cuenta la opción de nivelación de recursos en intervalos de tiempo, principalmente en aquellos proyectos con muchas actividades o conflictos de uso de recursos a

resolver, ya que puede permitir obtener la programación del proyecto nivelada en menor tiempo de ejecución.

Si el usuario selecciona la opción (2) debe ingresar la cantidad de días iniciales del proyecto que desea nivelar el uso de recursos (3). Para esto tiene acceso a la ventana emergente que se presenta en la figura 20. Para iniciar la nivelación de recursos el usuario realiza clic en el botón “Aceptar” (4).

Una ventana emergente con el título "Ingresar datos" y un botón de cerrar "X" en la esquina superior derecha. El texto principal dice "Ingrese la cantidad de días que desea nivelar". A la izquierda de este texto hay un número "4" en azul. A la derecha del texto hay un campo de entrada con el número "4" escrito dentro. Debajo del campo de entrada hay un botón "Aceptar" y un botón "Cancelar". En la parte inferior izquierda de la ventana hay un número "3" en azul y un campo de entrada vacío.

**Figura 20:** Ventana de ingreso de días a nivelar recursos

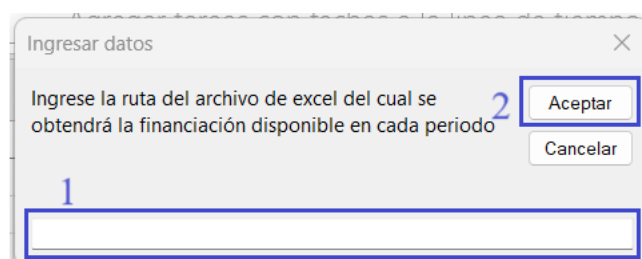
El resultado esperado de este módulo es el proyecto que representa la cartera con una programación sin conflicto de uso de recursos, y además un archivo en formato .txt con un resumen de las decisiones tomadas en cada momento de tiempo.

#### 4.6.4. Gestión financiera

El módulo Gestión Financiera permite evaluar la disponibilidad de recursos financieros en cada momento del proyecto, comparando la línea base de costos obtenida en la programación de la cartera, con la línea base de financiación planificada.

Para ejecutar este módulo, el usuario ingresa al archivo MS Project del proyecto que desea evaluar los recursos financieros, accede a la cinta de opciones y realiza clic en el botón GestionFinanciera.

El usuario tiene acceso a la ventana emergente presentada en la figura 21, en la que ingresa la ubicación de la plantilla de Excel que tiene información de la línea base de financiación planificada para cada momento (1). El contenido de esta plantilla debe ser en el formato establecido en ANEXO I Plantillas Excel.

Una ventana emergente con el título "Ingresar datos" y un botón de cerrar "X" en la esquina superior derecha. El texto principal dice "Ingrese la ruta del archivo de excel del cual se obtendrá la financiación disponible en cada periodo". A la izquierda de este texto hay un número "2" en azul. A la derecha del texto hay un campo de entrada con el número "2" escrito dentro. Debajo del campo de entrada hay un botón "Aceptar" y un botón "Cancelar". En la parte inferior izquierda de la ventana hay un número "1" en azul y un campo de entrada vacío.

**Figura 21:** Ventana de ingreso de ubicación de plantilla de datos

Para iniciar el proceso de evaluación financiera a partir de los datos de la planilla Excel realiza clic en el botón “Aceptar” (2).

El resultado esperado de este módulo es un archivo formato .txt con la evaluación de disponibilidad de recursos financieros en cada momento de tiempo.



---

## Capítulo 5 Análisis de resultados

A lo largo de este TFM se presenta la problemática de obtener una programación viable suficientemente buena de proyectos o carteras de proyectos, considerando la restricción de recursos.

En el capítulo 4 se presenta en detalle los principales componentes de la herramienta desarrollada que permite obtener una solución a este problema. Esta herramienta se encuentra completamente integrada a MS Project y se basa en las principales características de la heurística P-SGS/MINSLK.

Se ha visto también que Microsoft Project ya incluye su propia herramienta para la nivelación de recursos de los proyectos. En este punto, surge la necesidad de saber que tan buenos resultados proporciona la herramienta desarrollada en este trabajo, en comparación con la herramienta ya disponible en MS Project.

Este capítulo está dedicado a este fin, comparar los resultados proporcionados por la herramienta desarrollada con los que proporciona la herramienta ya disponible de Microsoft Project. La metodología que se utilizará se basa en realizar la programación de una muestra de las distintas carteras de proyectos que ofrece la librería MPSPLib con las dos herramientas y comparar los resultados obtenidos.

### 5.1 Metodología de evaluación de resultados

Para evaluar la calidad de resultados que proporciona la herramienta desarrollada, se comparan los resultados obtenidos con esta, con los que proporciona la herramienta incluida en MS Project. Para ello, se considera una muestra de 25 problemas estándar de la biblioteca MPSPLib, los que se programan con las dos herramientas. A partir de las programaciones obtenidas se compararán indicadores como el TMS de cada solución obtenida.

La selección de 25 problemas de la librería se realiza procurando obtener una muestra representativa en cuanto a cantidad de proyectos, cantidad de tareas y cantidad de recursos globales de los problemas seleccionados, de forma de tener problemas suficientemente heterogéneos para crear escenarios distintos que permiten obtener conclusiones sobre el funcionamiento de las herramientas en cada caso.

La librería MPSPLib presenta problemas con 2, 5, 10 o 20 proyectos; con 30, 90 o 120 actividades cada proyecto; y 1, 2 o 3 recursos globales. Combinando los valores de estas variables se tiene una combinación de 36 problemas posibles, de estos se seleccionan 25 problemas.

Los 25 problemas seleccionados se presentan en la tabla 2. Se muestra la cantidad de proyectos que contiene cada problema, la cantidad de tareas de cada proyecto, la cantidad de recursos globales y el identificador de la librería MPSPLib del problema que cumple con estas condiciones, el que ha sido seleccionado para probar las herramientas.

**Tabla 2:** Problemas de MPSPLib seleccionados para realizar pruebas.

Proyectos	Tareas / Proyecto	Recursos globales	Id MPSPLib
2	30	1	7
2	90	1	27
2	120	1	47
2	30	2	8
2	90	2	26
2	120	2	46
2	30	3	9
2	90	3	29
2	120	3	49
5	30	1	20
5	30	2	18
5	30	3	19
5	90	3	39
10	30	1	2
10	90	1	22
10	120	1	42
10	30	2	3
10	90	2	23
10	120	2	43
10	30	3	4
10	90	3	24
10	120	3	44
20	30	1	12
20	30	2	13
20	30	3	14

Entonces, luego de obtener una programación de estos 25 problemas seleccionados con la herramienta desarrollada y con la que proporciona Microsoft Project, se registra el TMS obtenido con cada herramienta y comparando este se evaluará la calidad de los resultados que proporcionan

Para comparar los valores de TMS obtenidos con cada herramienta se propone el indicador de la ecuación (6). Este indicador proporciona en porcentaje el beneficio de tiempo “ganado” por la programación obtenida en la herramienta de desarrollo propio, en comparación con el TMS obtenido por la herramienta de nivelación de recurso de Microsoft Project.

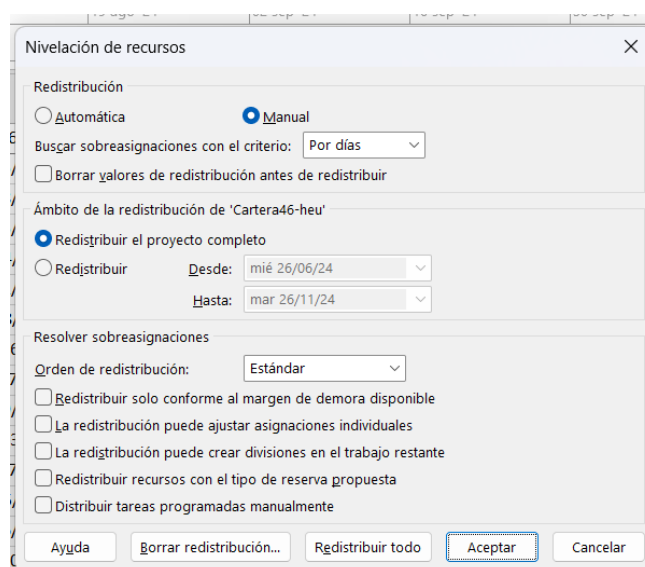
$$I_{TMS} = \frac{TMS_p - TMS_h}{TMS_p} \quad (6)$$

Donde:

- $TMS_h$ : Es el TMS obtenido por la ejecución de la herramienta de desarrollo propio.
- $TMS_p$ : Es el TMS obtenido por la ejecución de la herramienta de Microsoft Project.

Valores positivos del indicador  $I_{TMS}$  muestran que la programación obtenida por la herramienta de desarrollo propio tiene un tiempo de ejecución del proyecto menor al obtenido por la herramienta de Microsoft Project, por lo tanto, la herramienta de desarrollo proporciona una programación más óptima. Mientras que, valores negativos del indicador, muestran que la herramienta de Microsoft Project proporciona mejor programación.

Es importante mencionar que la herramienta de nivelación de recursos de Microsoft Project es utilizada con la configuración que se muestra en la figura 22.



**Figura 22:** Configuración de herramienta de MS Project en la ejecución de pruebas

De igual forma, la ejecución de la herramienta de elaboración propia se realiza replicando esta configuración, esto es, resolviendo en una única instancia la programación del proyecto completo y sin uso de algún tipo de metodología particular para resolver las asignaciones más que la propia del algoritmo desarrollado.

## 5.2 Resultados obtenidos

En la tabla 3 se presentan los resultados obtenidos para cada uno de los 25 problemas seleccionados. La columna denominada TMS muestra el valor de este indicador en la programación inicial del proyecto, cuando aún no se ha realizado la nivelación de recursos.

**Tabla 3:** Resultados obtenidos

Id MPSPLib	Proyectos	Actividades	Recursos globales	TMS	TMS <sub>h</sub>	TMS <sub>p</sub>	I <sub>TMS</sub>
2	10	30	1	89	119	132	9,85%
3	10	30	2	88	243	288	15,63%
4	10	30	3	96	146	166	12,05%
7	2	30	1	47	68	83	18,07%
8	2	30	2	64	64	64	0,00%
9	2	30	3	49	61	77	20,78%
12	20	30	1	117	276	336	17,86%
13	20	30	2	119	303	361	16,07%
14	20	30	3	115	177	193	8,29%
18	5	30	2	72	118	155	23,87%
19	5	30	3	72	72	72	0,00%
20	5	30	1	74	92	120	23,33%
22	10	90	1	128	128	129	0,78%
23	10	90	2	143	211	224	5,80%
24	10	90	3	145	145	155	6,45%
26	2	90	2	88	88	88	0,00%
27	2	90	1	107	130	148	12,16%
29	2	90	3	101	102	108	5,56%
39	5	90	3	123	129	140	7,86%
42	10	120	1	115	266	352	24,43%
43	10	120	2	138	146	138	-5,80%
44	10	120	3	148	371	562	33,99%
46	2	120	2	110	169	173	2,31%
47	2	120	1	107	146	170	14,12%
49	2	120	3	100	146	245	40,41%

Como se observa, la herramienta desarrollada obtuvo mejor o igual valor del índice  $I_{TMS}$  en 24 de los 25 problemas analizados. Llegando a un beneficio máximo en el problema 49, donde la programación obtenida es un 40,41% mejor que la obtenida con la herramienta de Microsoft Project. El valor promedio del índice  $I_{TMS}$  obtenido en los 25 problemas es de 12,55%.

En los problemas 8, 19, 26 y 43 el resultado obtenido con la herramienta desarrollada es iguales o peores que los obtenidos con la herramienta de Microsoft Project. Como se puede observar estos problemas tienen como característica que el TMS de la programación inicial, previo a la nivelación de recursos, se encontraba igual o muy próximo al TMS obtenido luego de nivelar los recursos. Es decir, la nivelación de recurso se realiza casi sin necesidad de alterar la fecha final del proyecto.

Es lógico que para estos casos la variación entre los TMS obtenidos sea mínima, ya que se espera obtener como resultado un tiempo de ejecución del proyecto similar al inicial. Por lo que, quitando

los problemas 8, 19, 22 y 26, lo que tienen la particularidad que  $TMS = TMS_h$ , el promedio del indicador  $I_{TMS}$  es 14,91%.

MPSPLib define el indicador OLF (*Overload Factor*) como el factor de sobre carga de un problema, a mayor valor de OLF mayor complejidad en la nivelación de los recursos del problema. En la tabla 4 se presenta el valor de OLF para cada uno de los problemas seleccionados.

Los problemas 9, 18, 20, 42 y 44 son los que presentan mejor valor en el índice  $I_{TMS}$ . La clasificación de OLF de MPSPLib de todos estos problemas es “High”, es decir, son problemas de compleja solución. Si extendemos esta idea, de los mejores 13 valores, 11 son problemas clasificados como “High”.

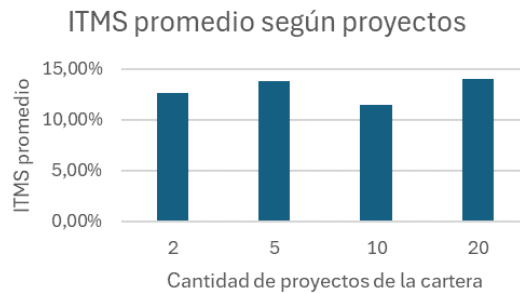
**Tabla 4:** Clasificación OLF de los problemas según MPSPLib

Id MPSPLib	TMS	$TMS_h$	$TMS_p$	$I_{TMS}$	OLF	Clasificación
49	100	146	245	40,41%	1,33	High
44	148	371	562	33,99%	2,36	High
42	115	266	352	24,43%	1,71	High
18	72	118	155	23,87%	1,11	High
20	74	92	120	23,33%	1,01	High
9	49	61	77	20,78%	0,79	Low
7	47	68	83	18,07%	0,85	Low
12	117	276	336	17,86%	2,25	High
13	119	303	361	16,07%	2,46	High
3	88	243	288	15,63%	2,64	High
47	107	146	170	14,12%	1,03	High
27	107	130	148	12,16%	1,02	High
4	96	146	166	12,05%	1,42	High
2	89	119	132	9,85%	0,89	Low
14	115	177	193	8,29%	1,51	High
39	123	129	140	7,86%	0,79	Low
24	145	145	155	6,45%	0,57	Low
23	143	211	224	5,80%	1,40	High
29	101	102	108	5,56%	0,59	Low
46	110	169	173	2,31%	1,17	High
22	128	128	129	0,78%	0,69	Low
8	64	64	64	0,00%	0,59	Low
26	88	88	88	0,00%	0,33	Low
19	76	72	72	0,00%	0,27	Low
43	138	146	138	-5,80%	0,75	Low

Al analizar el promedio del valor del indicador  $I_{TMS}$  para los problemas agrupados por la cantidad de proyectos que conforman la cartera, se obtiene la información de la tabla 5 y figura 23.

**Tabla 5:** ITMS promedio según cantidad de proyectos

Proyectos	$I_{TMS}$ promedio
2	12,60%
5	13,77%
10	11,46%
20	14,07%



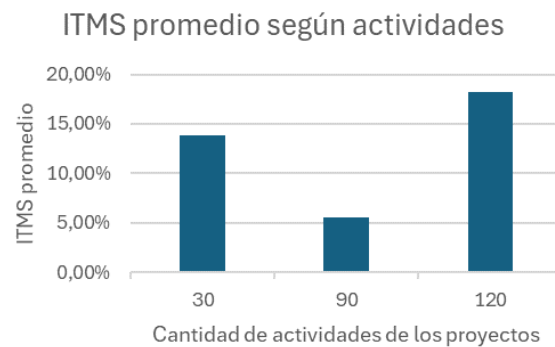
**Figura 23:**  $I_{TMS}$  promedio según cantidad de proyectos

Como se muestra en los resultados, no se identifica una relación entre los mejores resultados y la cantidad de proyectos que conforman la cartera, los resultados obtenidos son muy similares en los cuatro escenarios. Por lo que la herramienta desarrollada tiene comportamiento independiente de la cantidad de proyectos que conforman la cartera a la que se nivela los recursos.

Si se realiza en mismo análisis, pero comparando la cantidad de actividades que conforma cada proyecto de la cartera, se obtiene el resultado que se muestra en la tabla 6 y figura 24.

**Tabla 6:** ITMS promedio según cantidad de actividades

Actividades	$I_{TMS}$ promedio
30	13,82%
90	5,52%
120	18,24%



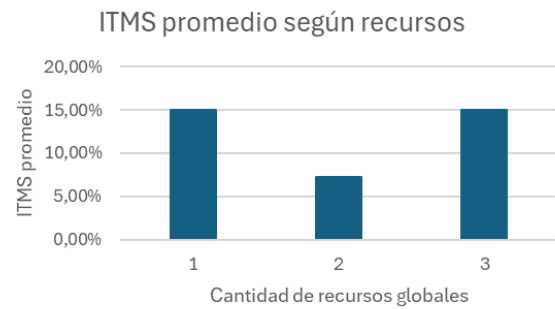
**Figura 24:**  $I_{TMS}$  promedio según cantidad de actividades

Se observa que para proyectos con mayor cantidad de actividades se obtiene mejores resultados, considerablemente por encima del valor promedio de todos los problemas (12,55%). Para proyectos con 30 actividades, la menor de las opciones, se obtienen resultados cercanos al promedio. Sin embargo, para proyectos con 90 actividades, el valor obtenido es considerablemente menor.

Al realizar el mismo análisis, pero en esta ocasión en función de la cantidad de recursos globales de la cartera, se obtiene la tabla 7 y figura 25.

**Tabla 7:** ITMS promedio según recursos globales

Recursos	$I_{TMS}$ promedio
1	15,07%
2	7,28%
3	15,04%



**Figura 25:**  $I_{TMS}$  promedio según recursos globales

El resultado obtenido es similar al caso anterior, en los valores extremos de la variable se obtiene mejores resultados que para el valor central. Para carteras con 1 o 3 recursos globales los valores obtenidos superan el promedio de todos los proyectos.

Es de destacar en esta sección el funcionamiento de los módulos Crear Tareas y Obtener Información, los que han sido de gran utilidad para el proceso de pruebas realizadas ya que, como se menciona en la sección correspondiente, automatizan los procesos de carga de información a Microsoft Project y la generación del proyecto de la cartera a partir de los proyectos individuales.

Además, la inclusión de la visión financiera en el módulo Gestión Financiera ha permitido obtener información sobre el riesgo de descalce financiero para cada momento de los proyectos programados. Este módulo ha sido probado ingresando información ficticia de los proyectos y ha mostrado excelente performance operativa.





---

## CONCLUSIONES

A lo largo de este trabajo se presentó una herramienta desarrollada capaz de obtener una solución factible a los problemas multiproyectos considerando la limitación de los recursos (RCMPSP). Su algoritmo de toma de decisiones es basado en las principales características de la heurística P-SGS/MINSLK desarrollada por el grupo de investigadores de la Universidad de Valladolid (Villafáñez & Otros, 2018).

La herramienta también es capaz de automatizar procesos tediosos como la carga de información a Microsoft Project y la generación de archivos que representan la cartera de proyectos, a partir de los proyectos individuales. Esto facilita considerablemente la gestión de la cartera de proyecto a los usuarios. Además, incorpora el análisis financiero de los proyectos, evaluando en cada momento los costos planificados contra la financiación prevista. Brindando al usuario información clave para la gestión de los recursos financieros.

Entonces, se tiene una herramienta completamente integrada a Microsoft Project y fácil de compartir entre usuarios del sistema. De uso intuitivo, cualquier usuario con conocimientos básicos en Microsoft Project es capaz de utilizarla sin complejidad alguna. A diferencia de la herramienta de nivelación de recursos disponibles en MS Project, se conoce en detalle su algoritmo de toma de decisiones, y todo proceso realizado por la herramienta genera archivos de registros que permiten conocer en detalle las acciones realizadas, lo que otorga mayor control al Director de Proyecto en la programación y uso de recursos.

Para verificar su utilidad al momento de programar proyectos, se realizaron pruebas en 25 proyectos seleccionados de la librería MPSPLib. La herramienta demostró obtener excelentes resultados en comparación con la herramienta de Microsoft Project. En 24 de las 25 pruebas realizadas se obtuvo igual o mejor resultado. Considerando los 25 problemas se obtuvo un  $I_{TMS}$  promedio de 12,55%, mientras que, si se considera únicamente los casos en los que fue necesario aplazar la fecha de finalización del proyecto, el  $I_{TMS}$  promedio pasa a ser 14,91%.

Se demostró también que, a mayor complejidad del problema a resolver, la herramienta desarrollada obtiene mejores resultados. Además, los resultados obtenidos no son sensibles a la cantidad de proyectos que conforman la cartera, lo que muestra ser una herramienta de utilidad para cartera de todas las dimensiones. Al analizar la cantidad de recursos globales de la cartera y la cantidad de actividades de los proyectos que la conforman, se encontró que para valores extremos de estas variables se obtienen mejores resultados que para valores medios.

El enfoque de desarrollo modular con el que se realizó la herramienta facilita su crecimiento, en un futuro integrar a su algoritmo de resolución otras heurísticas que han demostrado su buen funcionamiento puede otorgar mayor capacidad de decisión al usuario. Permitiendo evaluar el uso de algoritmo según el problema concreto que se pretende resolver.

Como conclusión final, se ha cumplido con el objetivo principal de este trabajo. La herramienta desarrollada ha demostrado ser eficaz y robusta en las pruebas realizadas, por lo que se encuentra en condiciones de ser utilizada por cualquier usuario que se enfrente a un problema RCMPSP.



## Futuros pasos

La herramienta desarrollada ha demostrado ser eficaz para la programación de un conjunto de problemas estándares pertenecientes a la librería MPSPLib. Surge la necesidad de evaluar su comportamiento en proyectos reales, en lo que sus características propias permitan concluir sobre la utilidad práctica de la herramienta.

Otro de los trabajos futuros que surgen, es la necesidad de mejorar la eficiencia del algoritmo de programación de proyectos incluido en el módulo Gestor Recursos. A lo largo del trabajo se ha mejorado considerablemente la eficiencia de este módulo, pero igualmente para proyectos con una cantidad considerable de tareas o bien de compleja distribución de los recursos entre las tareas, se obtienen tiempos de resolución altos en comparación con los tiempos de resolución que se obtiene con la herramienta de Microsoft Project.

La herramienta es desarrollada a partir de las características la heurística P-SGS/MINSLK, la cual ha demostrado obtener muy buenos resultados al momento de programar proyectos con restricción de recursos. Existen otros métodos de resolución publicados en la librería MPSPLib o desarrollados en otros trabajos por alumnos e investigadores de la Universidad de Valladolid, como por ejemplo (Álvarez-Campana, 2021), que también han demostrado excelentes resultados al momento de programar proyectos. Por lo que, uno de los futuros pasos que se establecen es la inclusión al módulo Gestor Recursos, la nivelación de recursos basados en distintas metodologías, brindando al usuario la posibilidad de decidir cual aplicar de acuerdo con las características propias del problema a resolver, o comparar resultados entre ellas.

Contar con reportes que muestren información relevante es clave para el director de proyectos al momento de la toma de decisiones. A pesar de que la herramienta hoy en día genera reporte de las decisiones tomadas en el proceso de programación de los proyectos, un paso futuro es la inclusión de reportes que de forma rápida muestre la información al usuario y en un formato en el cual la misma sea manipulable, ejemplo Excel.

Finalmente, la posibilidad de integrar a la herramienta tecnologías emergentes es algo que siempre debe tenerse presente. La creación de nuevos métodos de solución basados en nuevas tecnologías puede otorgar un aumento sustancial en la eficiencia de los algoritmos de resolución.



---

## Bibliografía

- Bernal, R. E. (2020). Comparación de Herramientas para Gestión de Proyectos. *Universidad San Jorge*, 15-16.
- Berzal, F. (2023). Optimización. *Departamento de Ciencias de la computación e IA, Universidad de Granada*.
- Blazewicz, J. L. (1983). Scheduling subject to resource constraints: classification and complexity. *Discrete applied mathematics*, 11 - 24.
- Bredael, D., & Vanhoucke, M. (2024). A genetic algorithm with resource buffers for the resource-constrained multi-project scheduling problem. *Elsevier*, 20.
- Herrera, F. (2010). Introducción a los Algoritmos Metaheurísticos. *Universidad de Granada*.
- Naihui He, D., & Zhang, B. (2022). Integrated multi-project planning and scheduling - a multiagent approach. *European Journal of Operational Research*.
- Olveira, A. (2004). *Heurísticas para Problemas de ruteos de vehículos*. Montevideo, Uruguay: Facultad de Ingeniería.
- Pajares Gutierrez, J., & Otros. (2009). Gestión eficiente de carteras de proyectos. *DYNA Ingeniería e Industria*, 761-772.
- Pajares, J. (2024). Introducción a la Dirección de Carteras de Proyectos. *Materia Estrategia Empresarial y Gestión de Carteras de Proyectos UVA*.
- Peng, J., & Liu, X. (2024). A study of comprehensive resource scheduling under public health events: An improved heuristic quantum algorithm. *IOS Press*.
- Pérez, O. d., & Otros. (2019). Colony of optimization algorithm for the automatic generation of schedules. *Revista Tecnología Digital*.
- PMI. (2017). PMBOK Guide 6ta edición. 307-357.
- PMI. (2017). The Standard for Portfolio Management fourth edition.
- Poza, D. (2024). Técnicas avanzadas de programación de proyectos. *Programación y Monitorización de Proyectos, Universidad de Valladolid*.
- Sanchez, D. (2011). DISEÑO ÓPTIMO DE LAMINADOS EN MATERIALES COMPUESTOS. APLICACIÓN DEL MEF Y EL MÉTODO DE LAS SUPERFICIES DE RESPUESTA. *Máster en Diseño Avanzado en Ingeniería Mecánica*, 100.
- Torres Sipion, C. (2024). Gestión de proyectos de sistemas de información en entornos virtuales y equipos. *Salud, Ciencia y Tecnología*.
- Villafañez, F., & Otros. (2018). A generic heuristic for multi-project scheduling problems with global and local resource constraints (RCMPSP). *Soft Computing*, 3465-3479.



## ANEXO I Plantillas Excel

### Plantilla de información de actividades de un proyecto

Esta plantilla contiene la información de todas las actividades que conforman un proyecto, como son nombre, sucesoras, duración y uso de recursos, además la disponibilidad de recursos del proyecto. Se utiliza en el módulo Crear Tareas para generar un archivo de MS Project a partir de la información que contiene.

Para un correcto procesamiento de los datos es imprescindible respetar el formato que se describe en esta sección y se muestra en la figura 22.

	A	B	C	D	E	F	G	H	I	J	K	L
1	jobnr.	#successors	successors	duration	R17	R18	R3	R19	R17	R18	R3	R19
2	1	3	2 3 4	0	0	0	0	0	79	81	66	69
3	2	3	14 16 38	6	1	9	6	6				
4	3	3	5 6 7	4	9	0	8	8				
5	4	3	11 24 78	1	6	6	8	4				
6	5	3	9 17 21	8	3	6	1	9				
7	6	2	8 13	7	1	9	7	8				
8	7	3	27 31 74	6	6	3	9	1				
9	8	2	23 82	10	10	9	2	10				
10	9	2	10 28	6	8	9	4	3				
11	10	1	12	5	10	3	8	1				
12	11	3	15 18 25	2	3	3	7	1				
13	12	2	45 57	3	1	4	3	6				
14	13	2	29 30	5	8	6	9	9				
15	14	3	22 40 58	2	3	6	10	6				

**Figura 26:** Plantilla Excel carga de datos de un proyecto

Cada fila de la plantilla contiene la información de una actividad del proyecto, los datos de cada columna se describen a continuación:

- Columna A: Contiene el nombre de la actividad. En MS Project se le agrega el texto “Tarea “ antes del contenido de la columna.
- Columna B: Contiene la cantidad de actividades sucesoras.
- Columna C: Contiene los Id de las actividades sucesoras separados por dos espacios.
- Columna D: Contiene la duración de la actividad en días.
- Columna E, F, G y H: Contienen la cantidad a asignar del recurso con nombre el valor de la fila 1 de la columna a cada actividad. El valor a asignar es en unidades, no en porcentaje, es decir, si cuento con 5 unidades del recurso se debe ingresar el valor 5.
- Columna I, J, K, L: Contienen la disponibilidad del recurso con nombre el valor de la fila 1 de la columna. El valor a asignar es en unidades.

El título de las columnas A, B, C y D es irrelevante, por lo que la fila 1 de estas columnas pueden tomar el valor que el usuario entienda conveniente. Mientras que los títulos de las columnas E a L son muy relevante ya que de ellos se obtienen los nombres de los recursos que se van a crear.

El nombre del archivo tampoco es relevante, pero si la hoja en la que se tiene la información a cargar debe tener como nombre “Hoja1”.

En el caso de contar con recursos globales, el nombre del recurso global debe permanecer igual en todos los archivos de los proyectos que conforman la cartera.

## Plantilla de información de recursos financieros

Esta plantilla contiene la información de la línea base de financiación planificada para cada momento del proyecto. Se utiliza en el módulo Gestión Financiera para comparar la financiación planificada con la línea base de costo, y de esta forma evaluar la disponibilidad de recursos financieros en cada momento.

Para un correcto procesamiento de los datos es imprescindible respetar el formato que se describe en esta sección y se muestra en la figura 23.

	A	B
1	Periodo	Financiacion disp
2	1	100
3	2	200
4	3	300
5	4	400
6	5	600
7	6	800
8	7	1000
9	8	1001
10	9	1002
11	10	1003
12	11	1010
13	12	1011
14	13	1012
15	14	1013
16	15	1014

**Figura 27:** Plantilla de información de disponibilidad de recursos financieros

Cada fila de la plantilla contiene la información de disponibilidad de recursos financieros de un día del proyecto, los datos de cada columna se describen a continuación:

- Columna A: Contiene el momento o periodo al cual se debe asignar la financiación planificada. En los hechos esta columna sirve únicamente de referencia para la carga de



datos para el usuario, ya que no se toma esta información en el procesamiento de la plantilla.

- Columna B: Contiene el valor de financiación acumulada planificada para cada periodo de tiempo.

Los títulos de las columnas no son relevantes para el procesamiento de la información, por lo que el usuario puede ingresar el valor que entienda conveniente.

El nombre del archivo tampoco es relevante, pero si la hoja en la que se tiene la información a cargar debe tener como nombre "Hojal".



**INDICE DE FIGURAS**

<b>Figura 1:</b> Modelo matemático del problema RCMPSP. Fuente: (Villafáñez & Otros, 2018) .....	6
<b>Figura 2:</b> Visión estratégica de las carteras de proyectos. Fuente: (PMI, 2017).....	7
<b>Figura 3:</b> Actividad en un diagrama AOA.....	9
<b>Figura 4:</b> Información de proyectos en biblioteca MPSPLib I .....	11
<b>Figura 5:</b> Información de proyectos en biblioteca MPSPLib II.....	12
<b>Figura 6:</b> Información de proyectos en biblioteca MPSPLib III.....	12
<b>Figura 7:</b> Información de proyectos en biblioteca MPSPLib IV .....	12
<b>Figura 8:</b> Representación del método Optimización de Colonias de Hormigas. Fuente: (Berzal, 2023).....	14
<b>Figura 9:</b> Enfoque D-RCMPSP y C-RCMPSP. Fuente: (Villafáñez & Otros, 2018) .....	16
<b>Figura 10:</b> Ventana de nivelación de recursos en Microsoft Project. ....	20
<b>Figura 11:</b> Asignación de recursos de una actividad .....	29
<b>Figura 12:</b> Botones de la cinta de opciones .....	47
<b>Figura 13:</b> Ventana de ingreso de datos de módulo Crear Tareas .....	48
<b>Figura 14:</b> Ventana de ingreso de cantidad de proyecto de módulo Obtener Información.....	48
<b>Figura 15:</b> Ventana de ingreso de ubicación de cada proyecto que conforma la cartera .....	49
<b>Figura 16:</b> Ventana de ingreso de disponibilidad de recuso de módulo Obtener Información .....	49
<b>Figura 17:</b> Ventana de ingreso de costo por hora de cada recurso.....	49
<b>Figura 18:</b> Ventana de ingreso de costo por uso de cada recurso .....	50
<b>Figura 19:</b> Ventana de selección de intervalo de tiempo de nivelación de recursos .....	50
<b>Figura 20:</b> Ventana de ingreso de días a nivelar recursos.....	51
<b>Figura 21:</b> Ventana de ingreso de ubicación de plantilla de datos.....	51
<b>Figura 22:</b> Configuración de herramienta de MS Project en la ejecución de pruebas .....	55
<b>Figura 23:</b> ITMS promedio según cantidad de proyectos .....	58
<b>Figura 24:</b> ITMS promedio según cantidad de actividades .....	58
<b>Figura 25:</b> ITMS promedio según recursos globales.....	59
<b>Figura 26:</b> Plantilla Excel carga de datos de un proyecto.....	67
<b>Figura 27:</b> Plantilla de información de disponibilidad de recursos financieros .....	68



**INDICE DE TABLAS**

<b>Tabla 1:</b> Resultados de P-SGS/MINSLK en casos de la librería MPSPLib. Fuente: (Villafañez & Otros, 2018).....	16
<b>Tabla 2:</b> Problemas de MPSPLib seleccionados para realizar pruebas.....	54
<b>Tabla 3:</b> Resultados obtenidos.....	56
<b>Tabla 4:</b> Clasificación OLF de los problemas según MPSPLib.....	57
<b>Tabla 5:</b> ITMS promedio según cantidad de proyectos.....	58
<b>Tabla 6:</b> ITMS promedio según cantidad de actividades.....	58
<b>Tabla 7:</b> ITMS promedio según recursos globales.....	59