Trabajo Fin de Grado

Grado en Ingeniería de Organización Industrial

Diseño e implementación de una herramienta informática para calcular las variables de un FMS con inventario en proceso constante

Autor: Sergio Diaz Aguilar

Tutor: José Manuel Framiñán Torres

Dpto. Organización Industrial y Gestión de Empresas I

Escuela Técnica Superior de Ingeniería

Sevilla, 2020

Trabajo Fin de Grado
Grado en Ingeniería de Organización Industrial

Diseño e implementación de una herramienta informática para calcular las variables de un FMS con inventario en proceso constante

Autor:
Sergio Diaz Aguilar

Tutor:
José Manuel Framiñán Torres
Catedrático de Universidad

Dpto. de Organización Industrial y Gestión de Empresas I
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla
Sevilla, 2020

Trabajo Fin de Grado: Diseño e implementación de una herramienta informática para calcular las variables de un FMS con inventario en proceso constante

Autor: Sergio Diaz Aguilar

Tutor: José Manuel Framiñán Torres

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

Presidente:

Secretario:

Acuerdan otorgarle la calificación de:

Sevilla, 2020

El Secretario del Tribunal

AGRADECIMIENTOS

Agradecer a mis padres todo el apoyo y paciencia que han tenido conmigo durante estos años en el grado.

A todos mis amigos que han confiado en mí y pudieron sacarme una sonrisa en los momentos difíciles.

Mostrar mi gratitud a cada uno de los profesores que he tenido y me han hecho aprender y mejorar en el día a día.

Sergio Diaz Aguilar

Sevilla, 2020

RESUMEN

En este trabajo hemos desarrollado una herramienta informática en C++ que nos ayuda a decidir qué cantidad de flujo externo debemos de introducir en un sistema de fabricación flexible *(FMS)* para mantener constante un valor de inventario en proceso *(wip)*. Además, recibimos información sobre distintas variables del sistema: tiempos de espera, tiempos de ciclo, *x-factor*, etc. También podemos obtener resultados de otras variables a nivel de estación. El FMS lo podemos analizar con una restricción general a nivel de sistema o con más de una restricción a nivel de estación.

Para la realización de este trabajo hemos usado principalmente los conocimientos adquiridos en la asignatura de "Sistemas Integrados de Producción" y conceptos de Teoría de Colas. También nos hemos apoyado en otra herramienta informática llamada ShopAnalyzer, desarrollada por José Manuel Framiñán, Catedrático de la Universidad de Sevilla.

ÍNDICE

ÍNDICE DE TABLAS

ÍNDICE DE IMÁGENES

1 OBJETO DEL PROYECTO

1.1 Objetivo

El objeto del proyecto es el desarrollo de una herramienta informática en lenguaje C++ que nos calcule la cantidad de flujo externo necesario para mantener constante el inventario intermedio (*Work in Progress o WIP*) de un sistema de fabricación flexible (FMS por sus siglas en inglés) y nos ofrezca las variables más importantes del sistema: tiempo de ciclo, el tiempo de espera, etc. Para ello, nos apoyaremos en el software informático ShopAnalyzer (Framiñan, 2020) que, además, nos permitirá extraer datos relevantes del proceso de producción. Para conseguir el objetivo propuesto hemos analizado el algoritmo EMPA (Framiñán, 2019) que consiste en la evaluación de los sistemas cerrados, en nuestro caso del FMS (véase punto 3.4.4). Además, sirve de base para el desarrollo de la aplicación informática que hemos creado con el nombre de *ConstantWip*.

1.2 Justificación

Uno de los mayores problemas que han tenido y siguen teniendo las empresas es la acumulación de inventario final que genera grandes costes de almacenamiento y por lo general pueden ser evitables o, al menos, minimizados. Cuando nos referimos a inventario imaginamos los productos finalizados acumulados en un almacén a la espera de demanda para darles salida. Sin embargo, suele ocurrir que no prestamos la atención requerida a un aspecto fundamental en el proceso productivo como es el estudio del inventario intermedio. Se genera entre estaciones y contribuye al aumento del coste de almacenamiento.

Tras el crecimiento de la filosofía Lean en Japón, llamada así por primera vez por John Krafcik en 1988 tras sus experiencias en las plantas de Toyota (Krafcik, 1988), y con la llegada de esta a Europa en los años ochenta y noventa, se desarrollan un conjunto de metodologías que proponen como solución a la problemática de almacenamiento producir solo cuando exista una demanda real. Este sistema de gestión de inventarios conocido

como *Just In Time* trata de fabricar los productos necesarios para satisfacer la demanda real (actual) y de esta forma mantener unos niveles de inventario mínimos.

Con el avance de esta metodología aparecen fundamentalmente dos sistemas de control de la producción. Por un lado, los sistemas *Kanban* que tratan de mantener constantes los niveles de inventario en proceso en cada estación o estaciones más importantes. Usan generalmente tarjetas o contenedores donde se almacena información sobre qué se va a producir, en qué cantidad, etc. Conjugan más de una restricción para llevar a cabo el mantenimiento constante del inventario en proceso. Por otro lado, los sistemas *Conwip* implantados con posterioridad (hay quién lo considera como una generalización de *Kanban*) tratan de mantener el nivel de inventario en proceso constante a lo largo de toda la línea o sistema productivo. En este caso tenemos una restricción general para todo el sistema, por lo que disponemos de más libertad para distribuir los productos a lo largo de las distintas estaciones.

En general, el uso de alguno de los anteriores métodos lo aplicamos cuando contamos con líneas de producción *(flow shop).* Como nuestro proyecto se basa en el análisis de un FMS, intentaremos coger y aplicar la idea fundamental de los métodos de control de la producción (limitación de wip en máquinas y en el sistema) en el FMS. En muchas ocasiones, los FMS cuentan con limitaciones de inventario debido principalmente a los sistemas de manejo de materiales (el conveyor o brazo robótico) que no permiten un inventario ilimitado, sino que están fuertemente limitados en muchos casos.

Ambos sistemas están enmarcados dentro de los procesos productivos de tipo *pull* que controlan el inventario en proceso. Dependiendo de características del sector, como la demanda, la variedad de productos con los que trabajamos y algunos otros, será recomendable utilizar un sistema u otro.

La herramienta informática que hemos desarrollado, llamada *ConstantWip*, calcula la cantidad de flujo externo necesario (*Input*) al sistema para mantener el nivel deseado de inventario en proceso. Además, nos dará información acerca de distintas variables fundamentales del sistema productivo gracias a la utilización conjunta con el programa informático *ShopAnalyzer*. Con el *ConstantWip* podremos efectuar el estudio del proceso productivo en un sistema de fabricación flexible donde buscamos mantener el *wip* constante a nivel de estación (*Kanban*) o a nivel de sistema (*Conwip*).

1.3 Sumario

El proyecto lo hemos estructurado en seis capítulos. A continuación, resumimos brevemente cada uno de ellos.

En el primer capítulo planteamos el objetivo del proyecto y el por qué hemos decidido desarrollar el *ConstantWip.* Además, cuenta con un sumario donde resume el contenido de cada uno de los capítulos.

El segundo capítulo comienza dando un repaso histórico por los diferentes sistemas de producción, desde la producción artesanal hasta el nacimiento y desarrollo de la filosofía Lean. Hemos tomado el sector automovilístico como referencia porque nos aporta una visión general bastante completa de la evolución de los sistemas productivos. Terminamos definiendo el FMS, cómo podemos clasificarlos, los diferentes elementos que los componen, etc.

En el tercero intentamos explicar los conceptos que más aparecen en el proyecto para facilitar la mejor comprensión del mismo. Además, describimos el material que hemos utilizado para desarrollar el *ConstantWip.* Desde las herramientas informáticas hasta las metodologías, fórmulas y demás elementos necesarios para comprender el funcionamiento de la aplicación informática.

En el cuarto capítulo explicamos de forma detallada el funcionamiento y manejo del *ConstantWip* para que el usuario consiga un resultado correcto del mismo. Además, hemos introducido varios ejemplos que nos permiten una visión más global y práctica de su funcionalidad.

En el quinto capítulo exponemos la conclusión con un resumen del proyecto y de las futuras líneas de investigación que podemos considerar tanto a nivel teórico como a nivel práctico y que sirvan para implementar o completar el funcionamiento del *ConstantWip*.

Al final del documento se muestra el conjunto de información consultada: libros, revistas, y páginas de internet usadas para la realización de este proyecto.

2 ANTECEDENTES Y CARACTERÍSTICAS DE LOS FMS

INTRODUCCIÓN

Como hemos comentado, el objetivo del *ConstantWip* es mantener constante el *wip* del sistema. Para conseguirlo, tomamos como referencia los sistemas *pull* que podemos considerarlos integrados en el control de materiales de la producción Lean. De ahí el breve recorrido que realizamos para conocer dónde y en qué contexto nacen estos tipos de sistemas, además de introducir algunos conceptos de la filosofía *Lean manufacturing* y cómo este nuevo modelo de gestión evoluciona.

La primera vez que encontramos el término Lean para referirnos al nuevo método, desarrollado en las plantas de Toyota en Japón, que nos permite hacer más con menos (menos esfuerzo, menos espacio, menos recursos, menos tiempo, menos desarrollo, menos errores) (*¿Por Qué La Metodología Lean Se Llama Lean?*, 2018) sucede en el artículo "Triumph of the Lean Production System" escrito por el ingeniero americano John Krafcik en 1988.

El hilo conductor entre el Lean manufacturing, (Japón-Toyota) y la fabricación en serie (Estados Unidos) es el sector automovilístico. Por esto, resulta interesante seguir estudiando la metodología *Lean* en este ámbito para verla como contrapunto a la producción en cadena.

En el primer punto de este capítulo veremos cómo han ido evolucionando los sistemas de fabricación desde la producción artesanal hasta la llegada del nuevo método para optimizar el sistema de producción: la filosofía *Lean manufacturing*. Tomaremos como ejemplo, principalmente, el sector de la automoción que tiene una estrecha relación con el desarrollo de las distintas formas de gestión y fabricación.

En el segundo punto de este capítulo, analizaremos los FMS, los elementos que los conforman y los distintos tipos que podemos encontrar en la vida real. Por último, observaremos la evolución de la robótica que tanta importancia tiene para los FMS.

La búsqueda para mantener el *wip* constante es aplicada en nuestro proyecto concretamente a un FMS por lo que es imprescindible conocer en profundidad todo lo relacionado con estos.

2.1 Desarrollo de los sistemas de fabricación en el sector automovilístico

A finales del siglo XIX y comienzos del siglo XX la fabricación en serie comienza a coger forma. En el año 1901, *Ransom Olds*, pionero de la industria automovilística estadounidense y fundador de las compañías *Oldsmobile* y *REO motor Car Company*, inaugura la primera cadena de montaje de todas las que se irían implantando a principios del siglo XX (Palacios, 2004) & (*Ransom Eli Olds*, 2020).

Imagen 2.1 Oldsmobile Production Line Production

Fuente: (Avila, 2016)

En 1911, *Frederick Taylor* en su obra *''Principles o Scientific Management''* sienta las bases del método de organización científica del trabajo conocido como taylorismo a partir de los siguientes principios (Barba Álvarez, 2010):

- Organización científica del trabajo.
- Selección y entrenamiento del trabajador.
- Cooperación y remuneración por rendimiento individual.
- Responsabilidad y especialización de los directivos en la planeación del trabajo.

El trabajo de Taylor puede considerarse como la racionalización de los procesos productivos al diferenciar las tareas de creación y ejecución. Así, los aspectos mentales quedan separados por completo de las tareas manuales. Esto constituyó una ruptura total con los métodos de producción del pasado, cuando la producción se organizaba en función del tipo de artesanía y los artesanos creaban, organizaban y completaban las tareas manuales (Jáuregui, 2001).

Henry Ford, en 1908, inspirado en la cadena de montaje de *Ransom*, comienza a sacar al mercado el famoso modelo T que tiene su año cumbre de producción en 1923 y que introduce la era del consumo en masa al ser vehículos baratos y sencillos de fabricar (Womack et al., 2017).

Tras las guerras mundiales, la cadena de montaje llegó a su punto álgido, sobre todo, en EEUU. Así, en el año 1955, la gran mayoría de compañías del resto de sectores industriales intentaban asemejarse al sistema de producción industrial en serie implementado por *Henry Ford* que da lugar al concepto fordismo (Womack et al., 2017).

Aunque se comienza a implementar a principios del siglo XX, no sería hasta la década de 1930 cuando comenzaría a instaurarse como un sistema de producción generalizado. Se puede considerar como una etapa del capitalismo moderno que abarca desde la década de 1940 hasta la década de 1970, la denominada edad dorada del capitalismo (Barciela, 2005).

El método de producción fordista implica la combinación del taylorismo con la creciente mecanización de grandes empresas con muchas líneas productivas, asociadas con la aplicación de la cadena de montaje, la selección uniforme de los componentes y de los productos finales (Jáuregui, 2001)

La finalidad primordial de este proyecto y en el que se basa el ConstantWip es conseguir constante el *wip* de un FMS. Esta idea subyace en los sistemas *pull* que a su vez se enmarcan en el concepto *Just in Time (sistema de* gestión de inventarios) que conforma uno de los muchos pilares de la filosofía Lean nacida en Japón. Nosotros analizaremos con mayor detalle el *JIT*; pero es conveniente tener una visión general de la filosofía *Lean manufacturing* para entender la gran influencia y repercusión que tuvo y sigue teniendo este modelo de gestión.

Mientras en Estados Unidos, de la mano de Taylor y Ford, introducen las primeras técnicas de optimización de la producción a principios del siglo XX, en Japón, a finales del siglo XIX, surge el primer

pensamiento *Lean Manufacturing* por parte de *Sakichi Toyoda*, el fundador del grupo Toyota. Aunque no es hasta finalizada la Segunda Guerra Mundial cuando comienza a tomar forma (*Origen y Evolución Del Lean Manufacturing*, 2015)

El concepto de *Lean* va más allá de una metodología o de un conjunto de herramientas que se aplican de forma aislada. Es una filosofía o forma de pensar para la mejora continua cuya idea central es crear máximo valor posible, desde el punto de vista del cliente, con el menor consumo de recursos usando el conocimiento y habilidades de la gente que realiza el trabajo. Aporta una visión totalmente distinta no solo en la producción propiamente dicha, sino en factores como la relación con los trabajadores o los clientes (Berenguer, 2015)

Detrás de *la filosofía Lean* podemos distinguir pilares bien definidos:

- Búsqueda de reducción de costes de fabricación, eliminando operaciones innecesarias.
- Un grado elevado de calidad, disminuyendo el número de piezas defectuosas e intentando que estos errores no se vuelvan a reproducir, abordando el problema raíz de estos.
- Aumentar el grado de satisfacción del cliente, no solo por la mejora en la calidad, sino entregando los pedidos a tiempo.
- Reducción del inventario y del *wip*, de esta forma disminuimos los costes y el espacio necesario en la zona de trabajo y maquinaria.

El *Lean Manufacturing* se fue desarrollando y perfeccionando en el seno de la empresa Toyota a lo largo de los años. Además de su fundador, personajes como su hijo *Kiichiro Toyoda*; el sobrino de éste, *Eiji Toyoda*, ingeniero de reconocido prestigio en la compañía; los también ingenieros *Taiichi Ohno, Shigeo Shingo y Seiichi Nakajima*, introdujeron conceptos como ***Jidoka*** (automatización con un toque humano), ***Just-in-time*** (ningún componente se fabrica antes de que sea estrictamente necesario), ***Kaizen*** (mejora continua), ***Poka-yoke*** (a prueba de fallos), ***Muda*** (eliminación de las actividades que no ofrecen valor), el ***TPM*** (Mantenimiento Productivo Total) o el ***OEE*** (Efectividad Total de los Equipos) que generaron una metodología y un conjunto de herramientas que conformaron el *Toyota Production System* (TPS), como comenzaría a ser conocido el *Lean Manufacturing* tras la crisis del petróleo de 1973 (Liker, 2010)

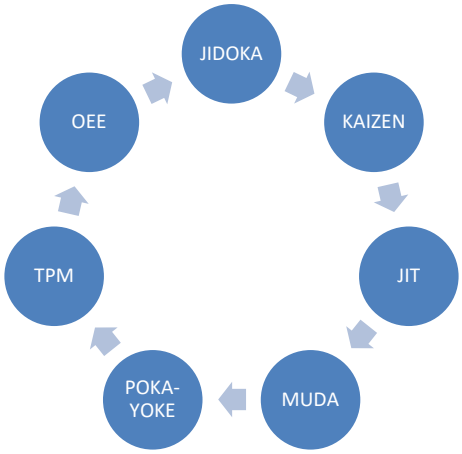

Imagen 2.2 Conceptos Fundamentales de Lean Manufacturing

Fuente: Elaboración Propia

Así, gracias a los positivos resultados económicos cosechados por la compañía en una época de recesión económica, el gobierno japonés fomentó su extensión a otras empresas del país. Sin embargo, su exportación a Occidente no llegaría hasta principios de los años noventa de la mano del libro de ***Womack, Jones y Roos*** titulado, *'The Machine that Changed the World: The Story of Lean Production, Toyota's Secret Weapon in the Global Car Wars that is now Revolutionizing World Industry'*, donde se habla por primera vez de *Lean Manufacturing.* Obviamente, el TPS no es un sistema cerrado, es decir, está en constante evolución y mejora continua. En la actualidad se aplica en su totalidad o con variaciones a todo tipo de empresas. Encontramos *Lean Thinking, Lean Management, ErgoLean, Lean Health, Lean Construction, Lean Office*, etc. (Berenguer, 2015)

2.2 Introducción a los Sistemas de Fabricación Flexibles

Un FMS es una celda altamente automatizada con un grupo de estaciones para procesado (generalmente máquinas herramienta CNC) interconectadas mediante un sistema automatizado de manejo y almacenamiento de material y controladas por un sistema integrado de ordenadores para dirigir los elementos anteriores (Groover, 2007)

Al FMS se le denomina flexible porque es un sistema capaz de adaptarse a los cambios en la demanda, productos y procesos de forma rápida y eficaz. Sin embargo, ningún sistema puede considerarse completamente flexible, siempre encontraremos límites en el grado de flexibilidad. Por ello, seremos capaces de diseñar y producir piezas dentro de un rango de estilos, tamaños y procesos, es decir, el FMS puede producir una familia única o un rango limitado de familias de piezas (Groover, 2007).

Para saber si un sistema lo podemos calificar como flexible debe cumplir cuatro criterios en mayor o menor medida (Groover, 2007):

- Procesar distintos estilos de piezas, pero no por el modelo de lotes.
- Aceptar cambios en el programa de producción.
- Responder de forma inmediata ante averías y errores.
- Aceptar la introducción de nuevos diseños de piezas.

2.2.1 Tipos de Sistemas de Producción

Según la variedad y el volumen de producción que vamos a manejar, podemos dividir los sistemas de producción en tres formas básicas diferentes (Framiñán, 2019):

- Organización por proyecto: El producto está inmóvil
- Organización por proceso (*job shop*): Se agrupan las operaciones similares y el producto se mueve.
- Organización por producto (*flow shop*): Se agrupan las operaciones en la secuencia que necesita el producto.

El FMS trata de cubrir el hueco entre la producción unitaria de gran variedad de piezas y la producción en serie. Por ello, es aconsejable aplicarlo en una producción de volumen medio y variedad intermedia. En caso de tener

una cantidad muy elevada de producto y escasa variedad, es preferible el uso de líneas de flujo o líneas transfer; mientras que en el caso de una alta variedad y cantidad de producto baja, es recomendable el uso de un sistema tipo taller (Groover, 2007).

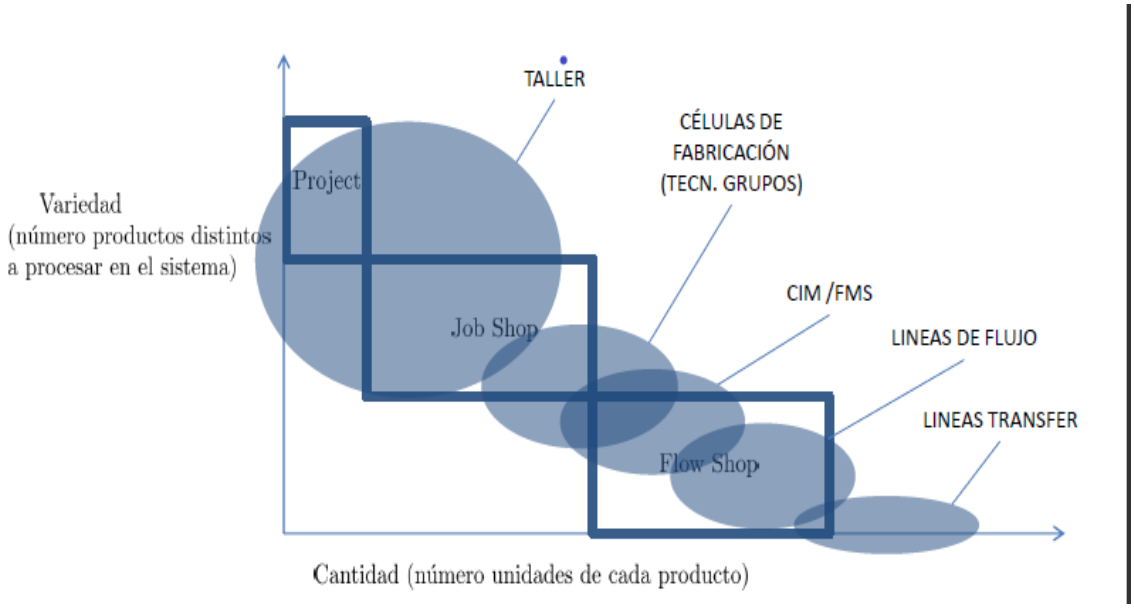

Imagen 2.4 Características de la aplicación de los distintos sistemas productivos.

Fuente: (Framiñán, 2019)

2.2.2 Componentes de un FMS

Ateniéndonos a la definición del FMS del apartado 2.2, podemos diferenciar distintos componentes (Framiñán, 2019):

- La estación de carga/descarga por donde los trabajos entran y salen.
- Máquinas de control numérico, normalmente automáticas.
- Sistema de transporte de materiales.
- Un ordenador o sistema de ordenadores que controlen los elementos anteriores.

2.2.2.1 Máquinas de Control Numérico

Una Máquina de Control Numérico (CNC, por sus siglas en inglés) es un dispositivo habilitado para tratar diferentes tipos de materiales y controlado por un ordenador. Su funcionamiento se basa en un sistema de coordenadas que especificará el movimiento de la herramienta de corte sobre la pieza. Estos movimientos están controlados por un programa informático ejecutado por un ordenador. La invención de este tipo de máquina ha reducido considerablemente los tiempos de proceso y han mejorado la eficiencia de los trabajos que anteriormente se realizaban de forma artesanal. Además, al aceptar opciones personalizadas por parte del operador, permiten satisfacer la gran mayoría de peticiones de los clientes (*Máquinas CNC: Todo Lo Que Necesitas Saber*, 2020).

El uso de *softwares* asistidos por computadora para el diseño, como CAD (*Computer-Aided Design)* y para la fabricación, como CAM *(Computer-Aided Manufacturing),* van de la mano de las máquinas CNC. Mientras que el CAD trabaja en el diseño de los modelos y los ensamblajes, el CAM hace uso de estos y genera las rutas de herramientas, es decir, transforma los diseños en información (código de programación g-code) para la máquina CNC (*CAD/CAM | Diseño y Fabricación Asistidos Por Ordenador*, 2020).

Imagen 2.5 Máquina CNC

Fuente: (*Control Numérico Por Computadora*, 2020)

2.2.2.2 Sistemas de Manejo de Materiales

Un sistema de manejo de materiales es un conjunto de métodos, instalaciones y equipamiento con capacidad limitada para transporte, embalaje y almacenaje. Estos sistemas pueden aplicarse para aumentar la productividad y lograr una ventaja competitiva en el mercado (Kulweic, 1985).

Según *Material Handling Institute,* (*The Ten Principles of Material Handling*, 2020) los diez principios para reducir costes y aumentar la productividad de la planta industrial haciendo uso de los sistemas de manejo de materiales son:

- Tener definidas las necesidades y objetivos de los métodos propuestos.
- Estandarización de los distintos equipos, controles, métodos y software dentro de los límites que logran los objetivos de desempeño.
- Minimización del trabajo de manejo de materiales sin eliminar productividad.
- Asegurar las operaciones para los humanos.
- Cargas unitarias adecuadas para lograr flujo de materiales y objetivo de inventarios.
- Utilización del espacio efectivo y eficiente.
- Movimiento y almacenaje de materiales deben estar integrados por completo para formar un sistema operativo.
- Automatizar las operaciones de manejo de materiales cuando sea posible para mejorar la eficiencia, consistencia y predictibilidad y disminuir los costos operativos.
- Tener en cuenta el impacto ambiental y el consumo de energía a la hora de seleccionar los equipos.
- Tener en cuenta todo el ciclo de vida de los sistemas de manejo de materiales, desarrollando un análisis económico.

Existen distintos tipos de sistemas de manejo de materiales:

- **Robot Industrial**. Según la norma ISO 8373 (*Norma Internacional ISO 8373. Robots and Robotic Devices*, 2012), un robot industrial es un manipulador multifuncional, controlado automáticamente, reprogramable en tres o más ejes, que puede estar fijo o móvil para uso

en aplicaciones de automatización industrial. Depende de una serie de parámetros como son (Framiñán, 2019):

- Grados de libertad. Son la suma de articulaciones que lo componen.
- Espacio de accesibilidad. Es el número de puntos accesibles al punto terminal.
- Capacidad de posicionamiento. Mide el grado de exactitud de los movimientos.
- Capacidad de carga. Peso que puede transportar.
- Velocidad. Máxima velocidad que puede alcanzar.

Podemos clasificar los robots industriales en cinco tipos diferentes:

- Robot articulado. Se clasifican según el número de articulaciones. Lo más frecuente entre cuatro y seis.
- SCARA. Muy usado en el ensamblado de productos.
- Configuración polar. Tiene varias articulaciones, cada una de ellas puede realizar un movimiento distinto: rotacional, angular y lineal.
- Coordenadas Cartesianas. Realiza tres desplazamientos ortogonales. Suele usarse para carga-descarga en las máquinas.
- Robot Delta. Suelen usarse para mover objetos pequeños a gran velocidad.

Imagen 2.6 Robot Industrial, SCARA

Fuente: (*Los 5 Tipos de Robots Industriales Más Utilizados Por Las Empresas*, 2020)

Existen otras muchas clasificaciones teniendo en cuenta otros factores. Una muy reconocida es la propuesta por T.M. Knasel, según a la generación a la que pertenecen, como podemos ver en la imagen 2.7.

Generación	Nombre	Tipo de control	Grado de movilidad	Usos más frecuentes
1ª (1982)	Pick & place	Fines de carrera, aprendizaje	Niguno	Manipulación, servicio de máquinas
2ª (1984)	Servo	Servocontrol, trayectoria continua, progr. codicional	Desplazamiento por vía	Soldadura, pintura
3ª (1989)	Ensamblado	Servos de precisión, visión, tacto, prog. off-line	AGV Guiado por vía	Ensamblado Desbarbado
4ª (2000)	Móvil	Sensores inteligentes	Patas Ruedas	Construcción Mantenimiento
5ª (2010)	Especiales	Controlados con técnicas de IA	Andante Saltarín	Uso militar Uso espacial

Imagen 2.7 Clasificación según Knasel

Fuente: (Barrientos et al., 2014)

- **AGV (*Automated Guided Vehicles System*).** Es un conjunto de vehículos con la posibilidad de programarles el destino, la selección de trayectoria y posicionamiento (BAWA, 2007).

  A la hora de realizar el diseño se deben tener en cuenta las trayectorias, el número de vehículos que comprende el sistema, los requerimientos que deben tener los vehículos para adaptarse fácilmente al sistema; que tipos de materiales va a transportar el AGV y cómo cuándo y dónde realizara las cargas y descargas de material; cómo se va a realizar el manejo de las baterías de los vehículos y cuáles van a ser los sistemas de seguridad empleados para evitar las posibles colisiones y daños del sistema (Le-Anh & De Koster, 2004). Por lo general, los pesos de transporte que soportan se mueve en un amplio rango, con baja velocidad y normalmente con sensores de ultrasonidos para evitar las colisiones. Nos podemos encontrar (Framiñán, 2019):

- Navegación por un camino definido.
- Navegación por un camino no definido. Estos últimos pueden subdividirse en:
  - Guiados por láser. Se posicionan etiquetas reflectantes en la planta para guiar al vehículo.
  - Navegación inercial. Usan giroscopios y sensores para determinar la posición actual del vehículo.

Imagen 2.8 Ejemplo de AGV

Fuente: (*AGV-Automated Guided Vehicles (Vehículos De Guiado Automático)*, 2020)

- ***Conveyors.*** Son sistemas transportadores utilizados para pesos medios o bajos, con una trayectoria fija y para un volumen de producción alto. Algunos de los tipos más usados son (Framiñán, 2019):
    - De ruedas
    - De rodillos
    - De remolque
    - De transportador aéreo
    - Cinta

Imagen 2.9 Ejemplo de Conveyor

Fuente: (Hill, 2020)

- **Grúas.** Sistemas usados para movimiento de cargas muy pesadas y con volúmenes de producción bajos. Dependiendo del tipo las podemos encontrar con trayectoria fija o variable. Las más comunes son (Framiñán, 2019):
    - Puente grúa.
    - Grúa semipórtico.
    - Grúa de pluma.

Imagen 2.10 Ejemplo de Grúa
Fuente: (*Grúas Para Manejo de Residuos*, 2020)

2.2.3 Clasificación de un FMS

Podemos clasificar los FMS dependiendo del número de CNC, el nivel de flexibilidad y la distribución en planta. Esta última característica depende en gran medida de los sistemas de manejo de materiales que se utilizan en planta.

Según el número de CNC (Groover, 2007):

- Celda de fabricación flexible. El sistema tiene una máquina CNC.
- Célula de fabricación flexible. El manejo de materiales es llevado a cabo por un robot, con un número reducido de máquinas CNC, entre dos y cuatro.

- Líneas de fabricación flexible. Las máquinas CNC están organizadas en un flujo regular. A veces se añaden buffers intermedios para dar una mayor flexibilidad.
- Sistema de manufactura flexible. Se le denominan sistemas cuando nos encontramos con más de cuatro máquinas CNC.

Estudiando la flexibilidad del sistema:

- FMS dedicado de manufactura especial. Están diseñados para producir una variedad limitada de estilos y cantidades de piezas.
- FMS de orden aleatorio. Cuando la familia es muy grande y hay variaciones substanciales en las configuraciones de partes. La programación de la producción puede estar sujeta a cambios diarios.

Según Groover (Groover, 2007), dependiendo de la distribución en planta:

- Distribución en línea. Las máquinas y los trabajos siguen una dirección lineal (uso de sistema de transferencia lineal). Son usados para trabajos que siguen un mismo orden. Por lo general, tienen la capacidad de movimiento en dos direcciones.
- Distribución en bucle. Sus estaciones de trabajo se encuentran en la periferia del transportador. Este sistema permite cualquier secuencia ya que podemos acceder a cualquier estación desde otra. El mayor problema de estos es el coste de transporte ya que la pieza tiene que dar en ocasiones muchas vueltas.
- Distribución en escalera. Su forma de trabajo es idéntica a la de bucle, pero en este caso las estaciones se ubican en los peldaños de la escalera.
- Distribución centrada en robot. La más eficiente y la más costa. Se utiliza para trabajos con distinto flujo.
- Distribución abierta. Es la distribución más compleja. La principal característica es el uso del AGV.

2.2.4 Posibles beneficios de la implantación del FMS

Como hemos visto en el apartado 2.1, la filosofía *Lean* trata de introducir una serie de métodos y herramientas para mejorar constantemente nuestro sistema de producción. Una vez implantado el FMS podremos beneficiarnos de ventajas como (Hernández & Vizán, 2013)

- Adaptabilidad en los programas de fabricación y cambios de diseño.
- Reducción de la mano de obra, de los *stocks* y, por tanto, del coste.
- Aumento en los niveles de calidad, minimizando los defectos.
- Posibilidad de realizar mantenimiento preventivo/predictivo para reducir tanto las averías como las paradas en los equipos; por lo que aumentamos el rendimiento de la máquina.
- Mayor capacidad de entrega en tiempos, lo que conlleva menos penalizaciones económicas.

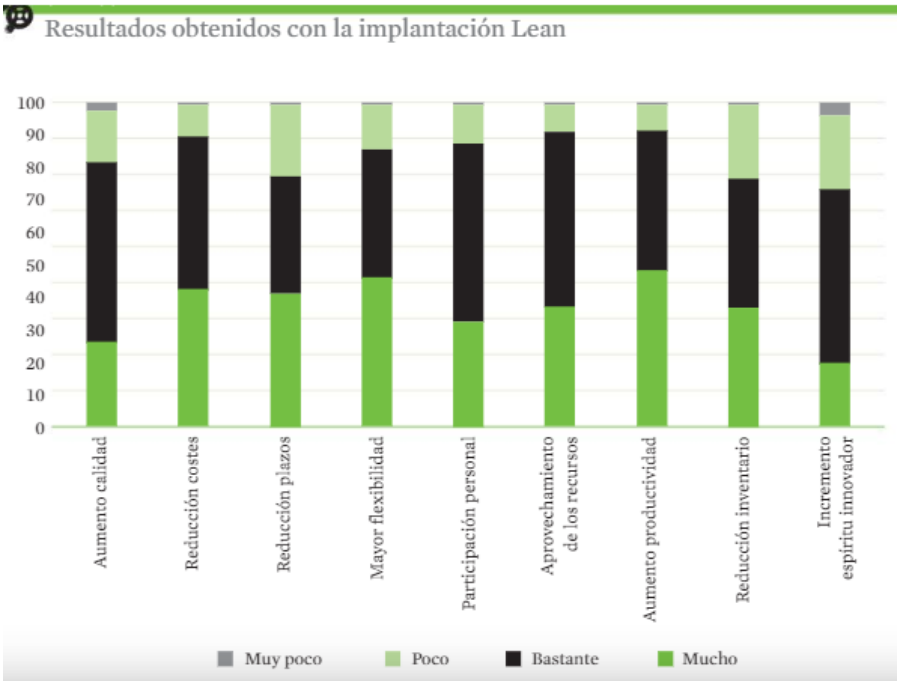

Imagen 2.11 Beneficios de la aplicación del Lean Manufacturing

Fuente: (Hernández & Vizán, 2013)

Sin embargo, existen algunos aspectos que se deben tratar con detenimiento antes de implantar FMS. Para obtener sus beneficios hemos constatado en el punto 2.2.2 la necesidad de contar con un nivel muy alto de tecnología: a nivel de máquina (uso de las máquinas CNC), de software informático que pueda diseñar y convertirlo en código para la máquina CNC (uso de CAD/CAM), y de mano de obra donde necesitaremos empleados altamente cualificados.

Por ello, antes de implantar un FMS debemos ser conscientes del gran coste inicial que conllevará. Si se consiguen conjugar los elementos examinados en el apartado 2.2.2 y llevar a cabo una mejora constante de nuestro sistema gestionándolo de forma eficaz podremos alcanzar los beneficios considerados anteriormente y acercarnos a las ideas expuestas en el punto 2.1 de la metodología *Lean.*

2.2.5 Evolución histórica de la róbotica y los FMS

De lo analizado en el apartado 2.2.2, los FMS se basan en el uso de las máquinas CNC, un sistema de transporte de materiales y un sistema de ordenadores que controlan los elementos anteriores. Podemos afirmar que para conseguir un FMS eficiente necesitaremos un gran uso de la robótica.

A continuación, veremos un pequeño resumen de cómo ha ido evolucionando el uso y la compra de los robots industriales. Esto nos dará una idea de cómo evolucionaron paralelamente los FMS, aunque con cierto retraso en el tiempo.

En 1961 se instala el primer robot industrial en el mundo, concretamente en General Motors (Estados Unidos); posteriormente le siguieron otros países como Japón y Suecia. Con la mejora de la relación nivel tecnológico-precio, en la década de los 80 hubo un gran aumento del mercado de la robótica (Pérez Pérez & Martínez Sánchez, 2000).

Según los datos que se muestran en la tabla 2.1 en el año 1998 existían un total de 720.392 robots en funcionamiento en todo el mundo.

CUADRO 1
NUMERO DE INSTALACIONES ANUALES Y STOCK DE ROBOTS EN OPERACION EN EL MUNDO EN LOS AÑOS NOVENTA

	1991	1992	1993	1994	1995	1996	1997	1998
Número de robots instalados	76.258	56.891	54.376	55.524	71.516	79.615	85.081	71.173
Número total de robots en operación	512.000	543.900	565.700	586.101	615.089	657.175	699.371	720.392
Porcentaje de instalación de reposición	—	43,9	59,9	63,2	59,4	47,1	50,4	70,4

Tabla 2.1 Número de instalaciones anuales y stock de robots en operación en los años 90

Fuente: (Pérez Pérez & Martínez Sánchez, 2000)

Observando la tabla 2.2 vemos como Japón lideraba con diferencia (un 57%) el *ránking*, seguido por Estados Unidos (11,34%) y Alemania (10,15%); mientras, España cuenta con el 1,2%. La ventaja de Japón en el uso de robots, frente al resto de países, está motivado por el nacimiento de la filosofía Lean en la automoción y el cambio de metodología que conlleva como expusimos en el punto 2.1.

CUADRO 2
NUMERO DE ROBOTS EN OPERACION EN 1998 EN DISTINTOS PAISES

País	Nº de robots	% sobre total	País	Nº de robots	% sobre total
1. Japón	411.812	57,16	7. Reino Unido	10.765	1,49
2. Estados Unidos	81.746	11,34	8. Rusia	10.000	1,39
3. Alemania	73.155	10,15	9. España	8.633	1,20
4. Italia	31.517	4,38	10. Benelux	7.245	1,00
5. Rep. de Corea	31.430	4,36	11. Taiwan	5.835	0,81
6. Francia	16.211	2,25	Resto de países	32.043	0,47

Tabla 2.2 Número de robots en operaciones en 1998 en distintos paises

Fuente: (Pérez Pérez & Martínez Sánchez, 2000)

De los datos de la tabla 2.3, se desprende que el sector de la automoción junto al de maquinaria son los más habituados en el empleo de los robots a finales del siglo XX.

DISTRIBUCIÓN DEL PARQUE DE ROBOTS POR SECTORES POR PAISES EN 1998

	Total	Alemania	España	Francia	Japón	Noruega	Reino Unido
Agricultura	0,1	—	—	—	0,1	—	0,1
Agroalimentaria	1,1	2,3	0,9	4,1	0,7	4,6	2,0
Textil	0,3	1,0	0,3	0,6	0,1	—	0,2
Madera y papel	0,9	1,0	0,2	0,4	0,8	4,0	0,5
Químico y plásticos	12,2	9,0	8,8	17,9	12,5	6,2	19,7
Metales y minerales	1,7	6,5	1,6	4,4	0,7	6,2	0,8
Transformados metálicos	6,3	7,1	7,0	1,7	4,9	46,8	5,4
Maquinaria	26,2	13,0	8,6	15,3	30,9	8,4	6,7
Telecomunicación	8,5	—	0,5	—	11,7	0,2	0,7
Instrumentación	1,1	1,1	—	—	1,2	0,6	0,2
Automoción	29,7	48,3	61,4	50,0	24,0	4,1	51,9
Muebles	0,6	3,4	0,9	—	—	7,4	4,5
Otros sectores	11,3	7,3	9,8	6,0	12,4	11,5	7,3

Tabla 2.3 Distribución del parque de robots por sectores por paises en 1998

Fuente: (Pérez Pérez & Martínez Sánchez, 2000)

Esto se debe principalmente al tipo de operaciones que tienen que realizar y en gran medida al coste de implantación de un FMS. Ya hemos puesto de manifiesto que la implantación de un FMS requiere una inversión muy elevada. En datos económicos, a finales de los años ochenta el 55% de los FMS habían costado menos de 3 millones de dólares y el 10% en torno a 20 millones de dólares (Pérez Pérez & Martínez Sánchez, 2000). Esta inversión inicial sólo es posible llevarla a cabo por algunas empresas pertenecientes al sector de la automoción, aeronáutica y maquinaria. (ver tabla 2.3)

Por otro lado, como ya anticipábamos, el desarrollo de los FMS siguió un camino similar al de los robots. El primer FMS se implementa en 1965 en Estados Unidos. A principios de los noventa, había 402 FMS, con mayor número de ellos en Japón y Estados Unidos con 112 y 81 respectivamente (Pérez Pérez & Martínez Sánchez, 2000).

En España estos sistemas fueron apareciendo con un importante retraso respecto a otros países, además de los mencionados anteriormente. Se instaló el primero en el año 1985 en la empresa Construcciones Aeronáuticas S.A. Hasta 1999 se habían utilizado los FMS en 39 empresas. En la tabla 2.4 podemos ver la distribución por sectores en España del uso del FMS y en la tabla 2.5 la distribución hasta 1999 por regiones del número de empresas con un sistema FMS (Pérez Pérez & Martínez Sánchez, 2000).

DISTRIBUCION SECTRIAL DE LOS SISTEMAS FLEXIBLES EN ESPAÑA	
Automoción	14
Fabricación maquinaria mecánica	12
Aeronáutica	5
Fabricación maquinaria eléctrica	5
Transformados metálicos	3
TOTAL EMPRESAS	39

Tabla 2.4 Distibución sectorial de los sistemas flexibles en España

Fuente: (Pérez Pérez & Martínez Sánchez, 2000)

DISTRIBUCION REGIONAL DE LAS EMPRESAS ADOPTANTES E INSTALADORAS DE SISTEMAS DE FABRICACION FLEXIBLE EN 1998		
	Empresas adoptantes	Empresas instaladoras
Andalucia	2	—
Aragón	3	—
Cantabria	1	—
Castilla-La Mancha	1	—
Castilla-León	1	—
Cataluña	11	3
Com. Valenciana	3	—
La Rioja	1	—
Madrid	9	2
Navarra	4	—
País Vasco	3	7
TOTAL EMPRESAS	39	12

Tabla 2.5 Distribución regional de las empresas adoptantes de sistemas de fabricación flexibles en 1998

Fuente: (Pérez Pérez & Martínez Sánchez, 2000)

Como era de esperar, con la llegada del siglo XXI la adquisición y el uso de los robots industriales ha seguido creciendo a nivel mundial. España no ha quedado al margen de este incremento como podemos comprobar en los siguientes gráficos que nos ofrece, además, información de los sectores líderes en el uso de los robots industriales.

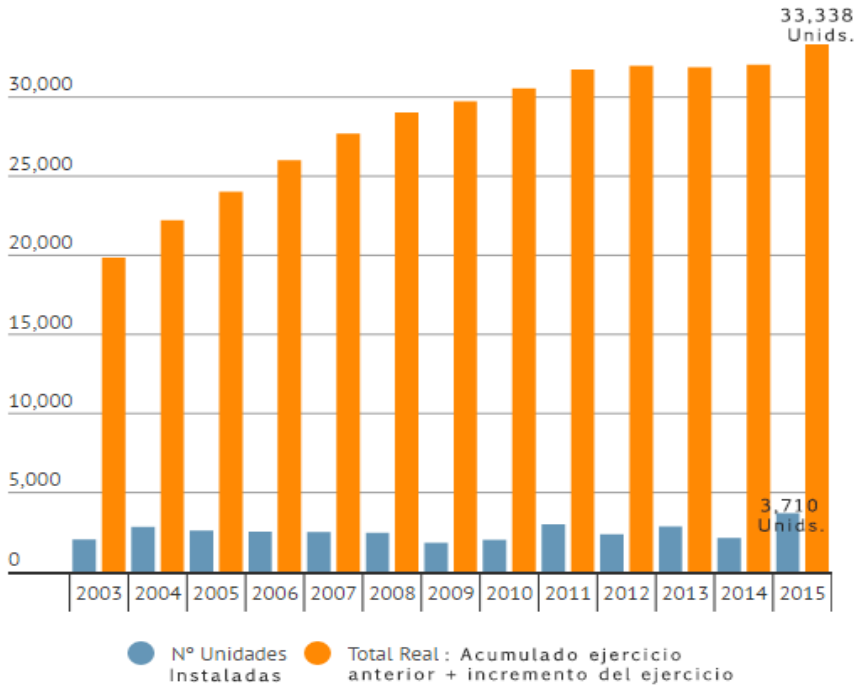

Tabla 2.6 Robots Industriales entre 2003-2015

Fuente: (Estadísticas Sobre Incorporación de Robots Industriales En España En 2015, 2016)

Apreciamos como España bate récord en el año 2015 en la instalación de robots industriales con 3.710 unidades adquiridas en 2015. Con estas incorporaciones el total acumulado es de 33.338 unidades.

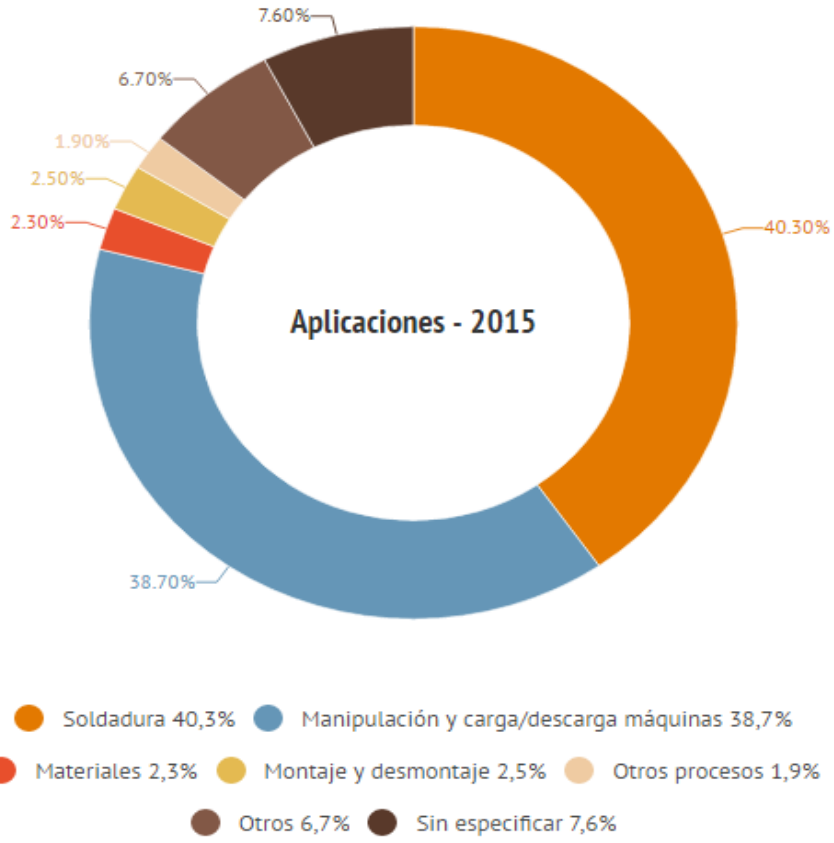

Tabla 2.7 Aplicaciones de Robots Industriales en el 2015

Fuente: (España Vuelve Al "Top10" de Robots Industriales Instalados, 2019)

Las principales aplicaciones se centran en la soldadura con un 40,3% y en la manipulación y carga/descarga de máquinas 38,7%. Esta última aplicación es muy común en los FMS donde en muchas ocasiones es necesario contar con robots industriales para la carga/descarga de los productos en las máquinas.

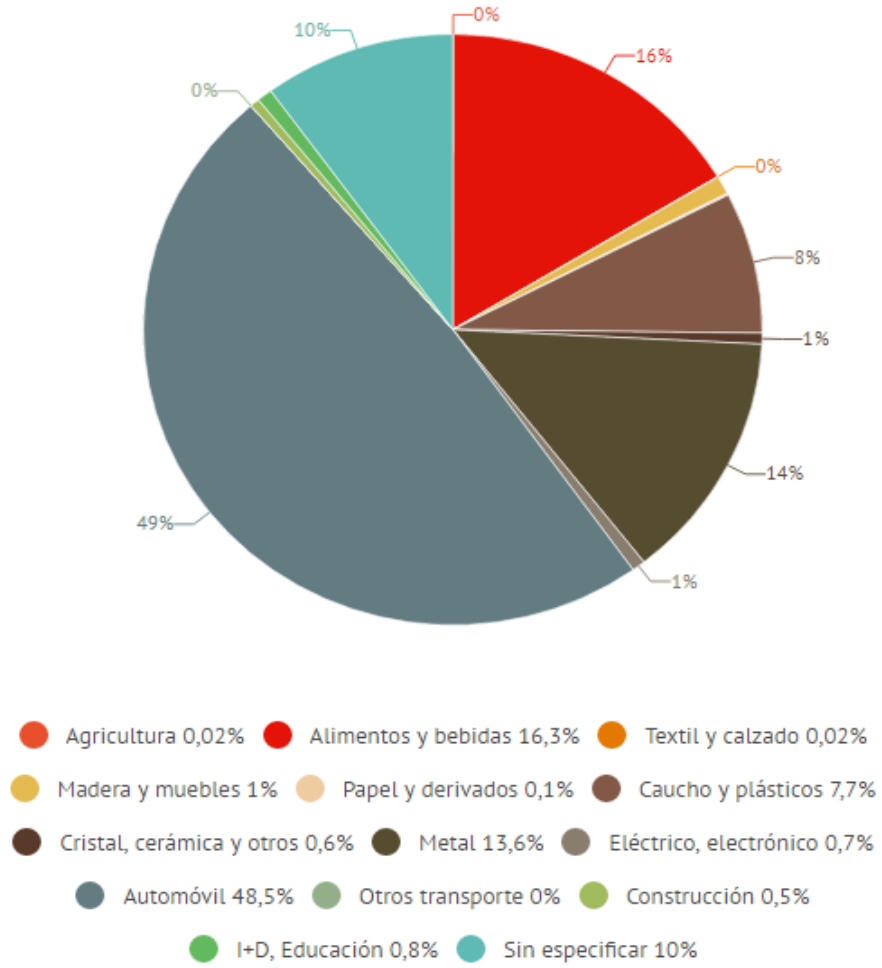

Tabla 2.8 Sectores de aplicación de los robots industriales en el 2015

Fuente: (España Vuelve Al "Top10" de Robots Industriales Instalados, 2019)

Como hemos señalado en la tabla 2.3 el sector del automóvil es el que más utiliza los robots industriales con un 49% en el año 2015.

A continuación, se ofrecen algunos datos de cómo ha ido evolucionando el uso de robots industriales en los últimos años tanto en España cómo a nivel mundial.

Con la instalación en el año 2018 de 5.266 nuevos robots industriales, España bate un nuevo record por cuarto año consecutivo. En este mismo año, España se sitúa en el cuarto mercado más importante a nivel europeo para la robótica. Este incremento se debe principalmente a los sectores automovilístico, el de

la alimentación y del metal/maquinaria. (*España Vuelve Al "Top10" de Robots Industriales Instalados*, 2019).

Al igual que ocurre en España, en el resto de países el aumento de robots industriales también sigue al alza. Asia sigue siendo el líder con un crecimiento en el 2018 del 1%, mientras que el mercado europeo, que es el segundo, creción un 14% y el americano lo hizo en un 20%. En el 2018 los cinco mercados principales: China, Japón, República de Corea, Estados Unidos y Alemania, representaron el 74% de los robots industriales adquiridos a nivel mundial. (*Crece Un 6% El Número de Robots Industriales Instalados En Todo El Mundo En 2018*, 2019).

Como sucede a nivel nacional, la industria auomovilística sigue siendo el sector con más demanda de robots industriales a nivel mundial con casi el 30% del suministro total en el 2018; seguido de cerca por la industria eléctrica/electrónica y la del metal y maquinaria (*Crece Un 6% El Número de Robots Industriales Instalados En Todo El Mundo En 2018*, 2019).

Si en el año 2018 en España contábamos con 5.266 nuevos robots industriales, a nivel mundial tuvimos un incremento del 6% respecto el año anterior con 422.000 nuevos robots industriales. Según las estimaciones, para el año 2022 alcanzaremos las 584.000 unidades a nivel mundial (*Crece Un 6% El Número de Robots Industriales Instalados En Todo El Mundo En 2018*, 2019).

CONCLUSIÓN

El crecimiento de la robótica en paralelo con los FMS a lo largo de las últimas décadas, hace entrever la importancia del estudio conjunto del FMS y el ConstantWip. Los beneficios para las empresas observados en el trascurso del tiempo no son meramente anecdóticos o irreales sino que, como hemos observado en la imagen 2.11, la implantación de la filosofía *Lean manufacturing* (en nuestro proyecto en particular el concepto del mantenimiento del *wip* constante incluido en el método de producción JIT) las mejora considerablemente en la mayoría de casos. Finalmente, tras conocer la disposición en que podemos encontrar estos sistemas y los principales componentes que forman los FMS podemos avanzar en el estudio del ConstantWip. En el siguiente capítulo introducimos conceptos y terminología frecuentemente usados cuando estudiamos y trabajamos con los sistemas de fabricación y el ConstantWip.

3 DESCRIPCIÓN DE LA METODOLOGÍA

INTRODUCCIÓN

Tras el breve resumen histórico del avance de la robótica, del desarrollo de nuevos métodos de gestión en la producción y fabricación, de cómo intervienen ambos en la creación de los sistemas FMS, cuáles son sus componentes y cómo podemos clasificarlos, en este capítulo, introduciremos conceptos importantes que nos ayudarán a entender el conjunto de herramientas y metodologías utilizadas para el desarrollo del programa informático y las fórmulas usadas en este.

3.1 El Stock

Nos referimos a *stock* como el material que encontramos inmovilizado temporalmente a la espera de ser usado o vendido.

Tenemos que diferenciar entre el *stock* o inventario en curso y el *stock* final. Los primeros son los productos que han terminado una fase (semi acabados) y están a la espera de pasar a la siguiente etapa de fabricación, mientras los segundos son aquellos productos terminados que se guardan en los almacenes para después distribuirse en los puntos de venta finales. También podemos incluir las materias primas procedentes de los proveedores que, en muchas ocasiones, generan gran inventario y coste esperando ser utilizadas (Onieva et al., 2006).

Uno de los puntos más importantes y, al mismo tiempo, unas de las claves de la filosofía *Lean* es la reducción de los *stocks*. En las últimas décadas han intentado proponer sistemas que utilicen el mínimo inventario posible (cero *stock*) (Onieva et al., 2006). En *Lean* está el llamado *JUST IN TIME* (JIT) que trata de producir únicamente el número de productos por los que tenemos una demanda real. De esta forma, estamos asegurando vender el total de las unidades producidas y, así dejar el inventario a cero. La principal ventaja de este sistema a la par que su inconveniente más importante reside en no disponer de *stock*. En caso de algún

contratiempo con los proveedores o en la cadena de producción podría originar el descontento de nuestros clientes ya que no llegaríamos a tiempo en los pedidos (*Definición de Metodología Just in Time o Justo a Tiempo y Cómo Aplicarla*, 2020).

3.2 Work in Progress

El *Wip* son aquellas materias primas que han entrado en nuestro sistema productivo, pero aún no están preparadas para su venta ya que todavía no son un producto terminado. En el apartado 3.1 lo hemos definido como inventario en curso. Al igual que hemos puesto de manifiesto la importancia que tiene el nivel de *stock*, el *Wip*, como parte de nuestro inventario, también necesita un estudio a fondo (Muñoz, 2012).

En este tercer capítulo nombraremos herramientas que podemos emplear en los sistemas *JIT* para tratar de mantener un *Wip* constante en nuestro sistema y conseguir minimizar el *stock.*

3.2.1 Sistemas con *Wip* constante

Podemos diferenciar dos métodos de *JIT* (Framiñán, 2019)*:*

- Sistema *Conwip*, tratará de mantener el *Wip* constante imponiendo una restricción en todo el sistema.
- Sistema *Kanban*, buscará conseguir un *Wip* constante imponiendo restricciones en algunas de las estaciones.

Hay numerosas variantes de sistemas que utilicen el *wip* constante como pieza fundamental en su producción. Algunos de ellos:

- *Workload control* de Bertrand (Bertrand, 1983)
- *C-WIP* de Glassey y Resende (Glassey & Resende, 1988)
- *Long pull* de Lambrecht y Segaert (Lambrecht & Segaert, 1990)
- *Globaly flexible line* de So (So, 1990)
- *Single Stage Kanban* al que se refiere Spearman (Spearman, 1992) y Di Mascolo (Di mascolo et al., 1996)

Todos estos son variantes del sistema *Conwip*, aunque existan diferencias entre ellos (Rodríguez et al., 2002).

Tanto los sistemas *Conwip* como los sistemas *Kanban* son considerados como subsistemas del método *JIT*. Tal y como hemos comentado en el punto 3.1, el método *JIT*, trata de producir el mínimo número de unidades en las menores cantidades posibles y en el último momento posible, eliminando la necesidad de inventarios (Hay, 1989). Ambas formas de controlar la producción las podemos encuadrar en los sistemas denominados *pull*, frente a los sistemas tipo *push*.

El modelo *Push*, trata de predecir las demandas futuras para comprar y producir lo necesario. Normalmente suele usarse un *software MRP* (*Material Requirement Planning*) que calcula la cantidad necesaria de producto. Este sistema es útil cuando tenemos un mercado o consumidor garantizado (Daniel et al., 1998). El inconveniente se presenta cuando nos encontramos en entornos o sectores donde la demanda varía aleatoriamente y no existe una repetitividad en los patrones o hay ausencia de tendencia.

El modelo *Pull*, adapta la producción a la demanda en tiempo real, es decir, solo comenzaremos a producir en caso de existir un pedido real. Aquí el flujo físico va en dirección opuesta al flujo de información (Daniel et al., 1998). Es el modelo utilizado para conseguir la idea de *JIT*. Por tanto, este modelo *Pull*, necesitará una reposición ágil de materia prima por parte de sus proveedores.

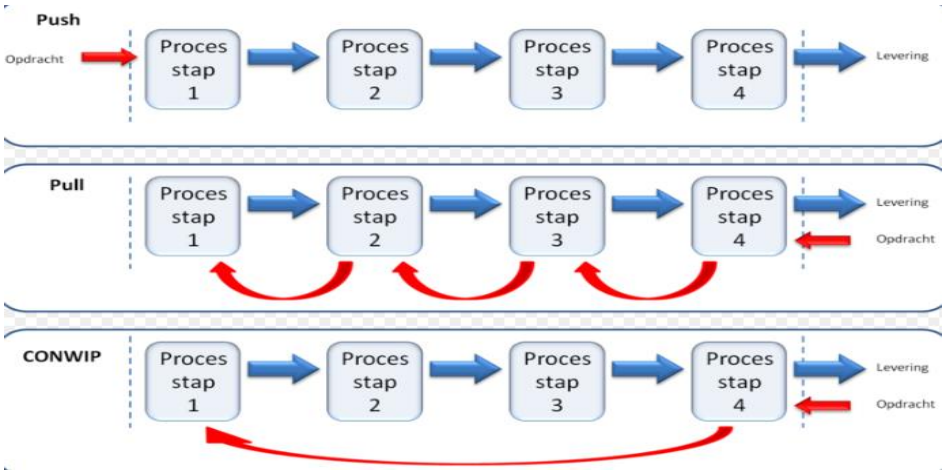

Imagen 3.1 Distintos tipos de Sistemas de Producción

Fuente: (Formación En Sistemas de Producción y Fabricación, 2014)

En la imagen 3.1 están representados los dos tipos de producción *pull* y el tipo *push*. En azul encontramos el flujo físico de materiales o productos mientras que en rojo observamos el flujo de información.

Un sistema *Pull* trae muchos beneficios, entre ellos la reducción del inventario y por tanto de los costes. Pero para (Flórez Pareja et al., 2016), los costes de la empresa podrían incrementarse mediante el uso del sistema *Pull*. Algunas empresas que tengan escasos beneficios, pueden sentir que no tengan ningún incentivo financiero para implantar un proceso tipo *Pull* (Flórez Pareja et al., 2016). Por otro lado, no siempre será una buena herramienta. Por ejemplo, en los casos donde haya un elevado número de rechazos, existencia de tiempos de *setup*, órdenes de trabajo con tiempos de proceso muy breves o con una demanda con mucha fluctuación (Ruiz-Usano, R. Framiñán, J.M., Crespo, A., Muñoz, 2020)

En algunos casos, el implantar un sistema híbrido puede resultar más beneficioso. Llamamos a un sistema híbrido aquel que utiliza en su sistema los dos enfoques de los que hemos hablado anteriormente (*push-pull*). Su uso es recomendable según algunas simulaciones (Rodríguez et al., 2002) para los sistemas donde existe una estación mucho más congestionada (estación cuello de botella). En este tipo de sistema la línea la dividimos en dos partes diferenciadas:

- Aguas arriba del cuello de botella aplicamos un sistema *pull*, para maximizar la ocupación del cuello de botella y evitar acumulación de inventarios excesivos.
- Aguas abajo del cuello de botella aplicaríamos un sistema *push*, para maximizar el *thoughput* (tasa de salida).

Este enfoque es el explicado por Goldratt (Goldratt, 2005) & (Ruiz-Usano, R. Framiñán, J.M., Crespo, A., Muñoz, 2020) sobre el uso de la *TOC* (Teoría de las restricciones).

Como hemos puesto en relieve en los puntos 2.1 y 3.1, la nueva filosofía surgida en Japón nos será útil para mejorar nuestro sistema de producción en algunos casos. Nosotros pondremos énfasis en la idea de mantener un *wip* constante dentro del sistema, y, por tanto, no generar un *stock* innecesario que, como ya hemos visto, es una de las partes que más problemas y costes pueden generar a una empresa. En los dos próximos capítulos la herramienta informática que hemos desarrollado nos ayudará a mantener los niveles deseados de *wip* constantes para intentar estar lo más cerca posible de la idea de cero stocks.

3.3 Presentación de las posibles herramientas

3.3.1 Herramientas para analizar líneas de producción

Existen diversas herramientas para realizar el estudio de una línea de producción.

- Modelos de simulación. Experimentan con un modelo muy cercano al real para ver los cambios que se producen en el sistema. Es una forma flexible de sintetizar un problema real complejo; aunque la optimización resulta difícil con un tiempo de proceso largo y complicado a la hora de desarrollar el modelo

  Algunos ejemplos de instrumentos de simulación son Writness, Arena, Ithink (Fullana Belda & Urquía Grande, 2009).

- Métodos analíticos. Tratamos de aproximar o simplificar un modelo conceptual con un conjunto de ecuaciones y fórmulas, para posteriormente encontrar un método de resolución numérica de estas (*¿Método Numérico, Analítico y Experimental: Concurrentes o Complementarios En La Ingeniería?*, 2017). El programa informático que hemos desarrollado utiliza las fórmulas y conceptos de la teoría de colas y lo visto en la asignatura de Sistemas Integrados de Producción.

3.3.2 Lenguajes informáticos y entornos.

Al planificar el programa informático que presentamos en este capítulo, comprobamos que en la actualidad disponemos de un gran número de lenguajes de programación donde escoger: C/C++, Python, Java, JavaScript, Fortran, ADA, Delphi... Lógicamente, nos referimos a lenguajes llamados “de alto nivel” o de “tercera generación” que utilizan el lenguaje natural para realizar los programas y expresan los algoritmos de forma adecuada a la capacidad cognitiva humana (*Lenguaje de Alto Nivel*, 2020)

En un principio hemos dudado de qué lenguaje utilizar, aunque siempre hemos tenido presentes el C, el C++ y Python. De este último, no teníamos conocimientos previos pero es un lenguaje de propósito general muy interesante porque ofrece una sintaxis bastante sencilla y cercana a la lógica humana, con un código bastante cómodo de leer y relativamente sencillo de generar, depurar y mantener.

C, es un lenguaje de nivel medio, potente, eficiente, que puede utilizarse para programar casi cualquier tipo de tarea y que ofrece un control muy importante de lo que sucede en el ordenador. Sin embargo, no es un lenguaje sencillo de aprender y requiere experiencia para sacarle rendimiento (Salas, 1991).

Por último, C++ nace como una extensión de C con la idea de que pudiese manejar objetos, bastante didáctico, ya que facilita el aprendizaje de otros lenguajes, y que ofrece un alto rendimiento. Sin embargo, también puede resultar más complicado de usar que otros lenguajes (Deitel & Deitel, 2004).

Al final, nos hemos decidido por este último debido a la escasez de tiempo de que disponemos para realizar el TFG, a los conocimientos previos que nos aporta C, ya que lo hemos tenido que trabajar en alguna asignatura del Grado, y a la estructura del programa informático a diseñar.

Para terminar, comentar que para su programación hemos utilizado el entorno de desarrollo integral DEV-C con el compilador g++ de MinGW, también empleado, en su momento, con el lenguaje C (*C (Lenguaje de Programación)*, 2020).

3.4 Metodologías aplicadas

3.4.1 Conceptos

Para entender el programa informático es necesario conocer algunos conceptos fundamentales de la teoría de colas y de los sistemas productivos.

El **cuello de botella** es la parte del proceso productivo más congestionada, que genera paradas y retrasos en el mismo. Todo sistema productivo tiene uno y es el que marca el ritmo o límite de producción. A veces, podemos determinar cuál es la actividad cuello de botella viendo la cantidad de stock generado delante de la estación conflictiva. Los cuellos de botella, cómo atacarlos y hacerles frente fueron estudiados por Goldratt (Goldratt, 2005) con la Teoría de las restricciones.

Los **sistemas de manejo de materiales** vistos en el punto 2.2.2 (MHS, por sus siglas en inglés) serán

parte fundamental del estudio de nuestro FMS. Por ellos pasarán cada uno de los trabajos, ya que todo el flujo que entra y sale de cada estación es movido por el MHS.

La **congestión** nos indica el porcentaje de tiempo que nuestra máquina o sistema está ocupado (Framiñán, 2019). Podemos definirla como:

$$\rho = \frac{\lambda}{\mu}$$

Donde:

λ: $Número\ de\ llegadas\ por\ unidad\ de\ tiempo$

μ: $Número\ de\ servicios\ por\ unidad\ de\ tiempo$

Para el caso de c máquinas idénticas en una misma estación, el cálculo de la congestión de la estación sería:

$$\rho = \frac{\lambda}{c * \mu}$$

Para el caso estable, la congestión siempre será ≤ 1, siendo $\rho = 1$ el caso límite.

Distinguimos dos casos (Framiñán, 2019):

1. $\boldsymbol{\rho < 1}$ → En este caso, $\lambda < \mu$ y por tanto th = λ. Se produce la conservación de tasas medias. La salida de productos va al mismo ritmo que la entrada de productos.

Tasa de producción (*throughput) th.* Es el número medio de unidades por unidad de tiempo que salen del sistema.

2. $\boldsymbol{\rho > 1}$ → *En este caso, $\lambda > \mu$ y por tanto $\mu = th$. Al entrar más unidades de las que se van produciendo (saliendo), se van acumulando unidades ilimitadas (Proceso Inestable).*

3.4.2 Teoría de Colas

La herramienta informática creada hace uso de la Teoría de Colas. En este apartado veremos qué es la teoría de colas, conceptos importantes de esta teoría y cómo la hemos utilizado para hacer el programa.

La teoría de colas la podemos representar de manera gráfica como la llegada de un conjunto de clientes que esperan ser servidos para poder marcharse (imagen 4.1). En nuestro caso, los clientes que entran en el sistema son los trabajos, mientras que el "servicio" que ofrecemos al cliente es el proceso que le realizaremos al producto en las estaciones.

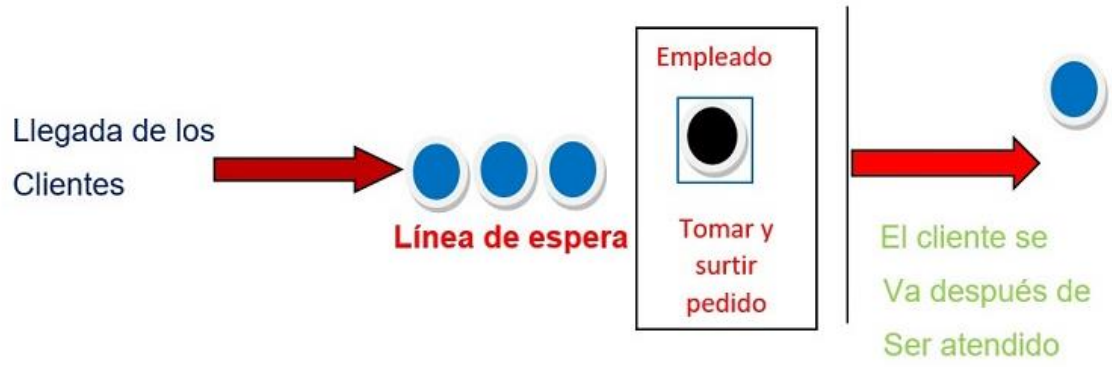

Imagen 4.1 Representación de una cola en un sistema
Fuente: (*Teoría de Colas o de Líneas de Espera*, 2020)

Para estudiar este tipo de problema debemos tener en cuenta (Pedro et al., 2015):

- La llegada de los clientes
- El tiempo de servicio
- Disciplina de la cola
- Capacidad del sistema

La **llegada de clientes.** En general es estocástica, es decir, depende de una variable aleatoria. Necesitamos conocer a que distribución probabilística representa y si la llegada entre clientes se da de forma individual o lo hacen en grupos en cuyo caso, podemos hablar de lotes en sistemas de producción (Pedro et al., 2015).

El **tiempo de servicio.** Tenemos que analizar las mismas características que en el tiempo de llegadas de clientes. Por un lado, la distribución probabilística que sigue y, por otro, si la máquina trata los productos de

forma individual o por lotes (Pedro et al., 2015).

Cuando existe la posibilidad de fabricar más de un tipo de producto, como es el caso que estamos abordando con un FMS, es necesario determinar la secuencia de entrada de los trabajos (Rodríguez et al., 2002) o lo que es lo mismo, el orden a través del cual seleccionamos a los "clientes" para que reciban el servicio. Esta secuenciación también es conocida como **disciplina de la cola.** Los dos métodos más conocidos son: (imagen 4.2)

- FIFO, donde atenderemos al cliente que primero llega.
- LIFO, atendemos al último cliente que llega.

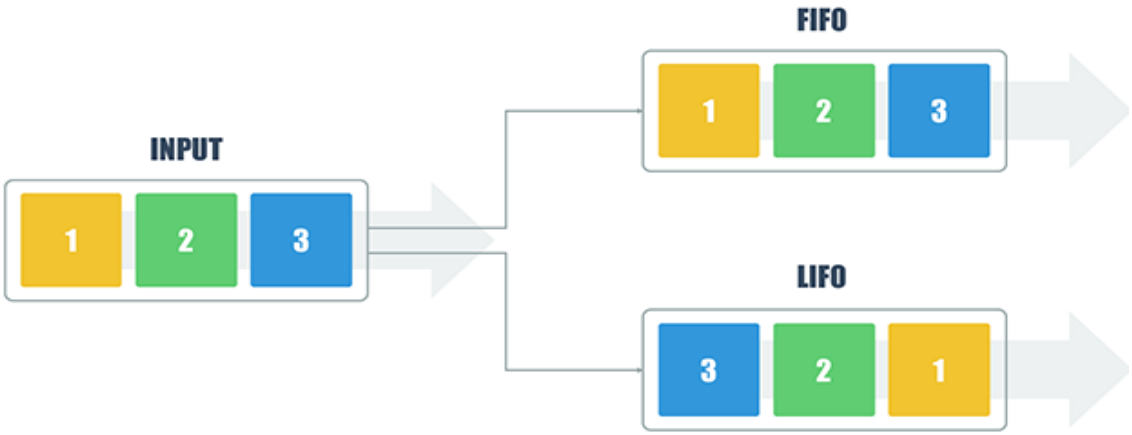

Imagen 4.2 Representación de dos formas de ordenación de las entradas en el sistema

Fuente: (*LIFO FIFO Para PowerPoint*, 2020)

Existen otros sistemas de ordenación: dependiendo del tiempo que tarde en ser procesados, dependiendo de la importancia del producto, etc. Para González (Rodríguez et al., 2002) en los sistemas *conwip,* es importante tener en cuenta dos cuestiones: Si los distintos trabajos comparten un mismo cuello de botella o si el cuello de botella es dependiente de la secuencia y por otro lado si la demanda es contra stock o contra pedido.

En nuestro caso, estudiaremos el sistema de producción aplicando el método FIFO. Conforme vayan entrando los productos los iremos procesando sin darle más importancia a unos u otros.

La **capacidad del sistema**. La podemos definir como la máxima tasa de producción (*throughput) th* que conseguimos sin que nuestro sistema se convierta en inestable; en cuyo caso, estaríamos acumulando un número infinito de trabajos dentro del sistema (Framiñán, 2019).

Uno de los procedimientos para calcular cuantitativamente la capacidad es conseguir que la congestión del cuello de botella sea de un 100% en nuestro sistema, es decir, conseguir que la estación más congestionada, debido a que sus tiempos de proceso son muy grandes o todos o parte de los productos pasan por ella, esté constantemente en funcionamiento (congestión 100%). La tasa de entrada necesaria para conseguir esto es la capacidad del sistema (Framiñán, 2019).

En ocasiones, debemos tener en cuenta reprocesos en el sistema. Esto es, hay productos que necesitan pasar por la misma máquina varias veces a lo largo del sistema productivo. Por ejemplo, un producto que recibe una primera capa de pintura es llevado a otra estación para someterlo a otros procesos; por último, es transportado de nuevo a la máquina de pintura para aplicarle una segunda capa. En estos casos, hay que tener en cuenta que los tiempos de proceso no tienen que ser iguales en ambas etapas (Framiñán, 2019).

3.4.3 Sistemas Integrados de producción

Para realizar el programa informático, además del uso de la Teoría de Colas, hemos recurrido a los conocimientos adquiridos en la asignatura de SIP (Sistemas Integrados de Producción). A continuación, veremos las fórmulas y nomenclaturas básicas para el entendimiento del proyecto.

λ: $Tasa\ de\ llegada. Número\ de\ llegadas\ por\ unidad\ de\ tiempo$

μ: $Tasa\ de\ producción. Número\ de\ servicios\ por\ unidad\ de\ tiempo$

Th. Throughput: *Número medio de unidades que salen del sistema.*

$\rho: Congestión$: *Nivel de ocupación del sistema*

c: *Número de máquinas idénticas, en la misma estación.*

$\omega: Wip$ *Número medio de unidades que hay en el sistema.*

Ct. Tiempo de Ciclo. *Tiempo medio entre la entrada del producto al sistema y su salida.*

Qt. Tiempo de Espera. Tiempo medio de espera de los trabajos dentro del sistema

Ts: *Tiempo de servicio.*

θ_a*:* *Variabilidad de las llegadas*

θ_s*:* *Variabilidad del servicio*

α(i): *Es la proporción de flujo externo que pasa por la estación i.*

th:* *Capacidad del sistema*

b: *Es la estación cuello de botella*

$\lambda = th \; si \; \rho < 1$ *Conservación de tasas medias*

$\rho = \frac{\lambda}{c * \mu}$ $Congestión$

$\omega = \lambda * ct$ *Fórmula de Little*

$ct = qt + ts$ *Relación tiempo de ciclo y tiempo de espera*

$qt = V * U * T$ *Fórmula de Kingman (si c > 1 →V * U' * T)*

- $V = \frac{\theta_a + \theta_s}{2}$ *Variabilidad del proceso*
- $U = \frac{\rho}{1-\rho}$ *Ratio de ocupación*
- $T = ts$ *Tiempo de servicio*
- $U' = \frac{\rho^{\sqrt{2C+2}-1}}{C*(1-\rho)}$ *Ratio de ocupación (si c>1)*

Fórmula de Little: La fórmula de Little que utilizamos a nivel de sistema, también es aplicable a nivel de subsistema. Si por ejemplo, quisiéramos calcular el número de unidades que hay en el buffer de una estación, podríamos aplicar la fórmula, donde λ sigue siendo la tasa de llegada y *ct* lo sustituiríamos por *qt*. Ahora queremos saber el número de unidades en el buffer y no en todo el sistema (Framiñán, 2019).

Fórmula de Kingman: Es una aproximación del tiempo de espera medio cuando existen variabilidades en las llegadas o servicios. Para sistemas cercanos al nivel de inestabilidad, obtenemos valores muy precisos según (Harrison & Patel, 1993).

La búsqueda para ofrecer la mínima espera al cliente nos exige plantear como objetivo ideal que tiempo de ciclo y de servicio sean iguales, es decir, ausencia de tiempos de espera. Si observamos la variabilidad del proceso (la V en la fórmula de Kingman) verificamos que está formada por variabilidad en la llegada de los productos y en el servicio. Si ambos datos son cero, el primer término sería nulo, y, por tanto, la fórmula de Kingman formada por productos nos daría como resultado cero. En consecuencia, la única forma de eliminar el tiempo de espera en el proceso es conseguir la ausencia de variabilidad.

El trabajo manual ejecutado por un trabajador, por muy experimentado que este sea, tendrá variabilidades en la calidad del producto. La variabilidad será mayor o menor en función de muchos factores: Anímicos, fatiga, conocimiento del trabajo realizado, etc. La única forma de anular estas variabilidades, o al menos reducirlas considerablemente, es introduciendo la automatización (Vaughn, 1990). Para el estudio de este trabajo, cuando nos referimos a variabilidad, debemos entenderla referida a la relacionada con los tiempos del sistema (de llegada, de salida, de servicio, etc).

A nivel de sistema:

Algunas de las **variables más importantes del sistema** las podemos calcular una vez obtenidas las variables de cada estación.

$$ct = \sum_{i=1}^{m} \alpha(i) * ct(i) \qquad Tiempo\ de\ ciclo$$

$$t_0 = \sum_{i=1}^{m} \alpha(i) \cdot t_s(i) \qquad Tiempo\ de\ ciclo\ mínimo$$

$$\omega = \sum_{i=1}^{m} \omega(i) \qquad Inventario\ en\ proceso$$

$$\text{x-factor} = \frac{\text{ct}}{\text{t}_0} \qquad \textit{X-factor}$$

$$th^* = \frac{c(b) * \mu(b)}{\alpha(b)} \qquad Capacidad$$

Calcular el factor x en un sistema de producción puede resultar de gran utilidad, ya que informa de la eficiencia de nuestro sistema (Framiñán, 2019). El elemento *ct* es el tiempo de servicio más el tiempo de espera. Si buscamos un sistema ideal no debe de existir tiempo de espera. En cuyo caso el *ct* coincide con la variable t_0 o tiempo de ciclo mínimo de nuestro sistema. En general, el *x-factor* es el número de veces que mi *ct* es el *ct* mínimo (t_0).

Cuando calculamos **α(i)** podemos pensar que será menor o igual a uno porque es la proporción del flujo total externo que pasa por una máquina i. No obstante, hay que tener en cuenta los casos de reentrada que hacen que el resultado sea mayor a 1. Estos casos, son muy habituales en los FMS, en concreto en los MHS, ya que son los responsables de cualquier movimiento de los trabajos dentro del sistema (Framiñán, 2019).

Los sistemas FMS pueden ser estudiados como un sistema tipo taller. Sin embargo, tienen algunas características propias que hace conveniente su estudio con otras fórmulas integradas en el software ShopAnalyzer:

$$\lambda(i) = \sum_{j=1}^{n} v^j(i) * \gamma^j \qquad (1)\ Tasa\ de\ entrada\ a\ estación\ i$$

$$\lambda(MHS) = \gamma + \sum_{i=1}^{m} \lambda(i) \qquad (2)\ Tasa\ de\ entrada\ a\ MHS$$

$$t_s(i) = \frac{1}{\alpha(i)} \sum_{j=1}^{n} t_v^j(i) * ({}^{\gamma^j}/_{\gamma}) \qquad (3)\ Tiempo\ de\ servicio\ de\ estación\ i$$

$$t_s(MHS) = \sum_{j=1}^{n} p^j(MHS) * t_s^j(MHS) \qquad (4)\ Tiempo\ de\ servicio\ del\ MHS$$

$$p^j(i) = \frac{v^j(i)}{\alpha(i)} * \left(\gamma^j/_{\gamma}\right) \qquad (5)\ Proporción\ de\ j\ en\ i$$

$$p^j(MHS) = \gamma^j \frac{1 + \sum_{i=1}^{m} v^j(i)}{\lambda(MHS)} \qquad (6)\ Proporción\ de\ j\ en\ MHS$$

$$\theta_s(i) = \frac{\sum_{j=1}^{n} p^j(i) * \left(t_s^j(i)\right)^2 * \left(\theta_s^j(i) + 1\right)}{t_s^2(i)} - 1 \quad (7)$$

A continuación, explicamos las variables que pueden resultar de mayor complejidad de las anteriores fórmulas:

(1) $v^j(i)$: *Número de veces que un producto j visita la estación i.*

γ^j: *Flujo externo del producto j.*

(2) γ: $Flujo\ externo\ total.$

(3) t_v^j: *Tiempo total que el producto j pasa en la estación i.*

$\gamma^j/_{\gamma}$: *Mix del producto j*

$\alpha(i)$: $La\ proporción\ del\ flujo\ externo\ que\ entra\ en\ la\ estación\ i$

(4) En ConstantWip el tiempo de servicio del MHS es independiente al tipo de trabajo, la fórmula (4) es usada para el caso de que el tiempo sea dependiente del trabajo j.

(7) La variabilidad agregada del tiempo de servicio.

Para el cálculo de la variabilidad entre llegadas de cada estación $\theta_a(i)$ es necesario calcular las ecuaciones de balance de variabilidad.

$$\theta_a(i) = \mathrm{f}\,(\vec{\theta}_a, \vec{\lambda}, \vec{t}_s, p)$$

Ya hemos comentado el significado de todos los elementos salvo el de *p.* Este representa la matriz de probabilidades agregada. Para realizarla debemos tener distintas consideraciones:

- El flujo entre estaciones es 0, ya que el MHS es el encargado de mover cada producto entre las distintas estaciones. Por tanto, la probabilidad de que un producto en una estación i vaya al MHS es 1 para todas las estaciones.
- El flujo desde el MHS a una estación i:

$$p(MHS,i) = \frac{\lambda(i)}{\lambda(MHS)}$$

3.4.4 FMS abierto y FMS cerrado

Habitualmente el estudio de un FMS lo realizamos sobre un FMS abierto. Esto significa que el número de trabajos dentro del sistema no está limitado. Aunque en la realidad es difícil encontrarnos con sistemas de este tipo, su estudio simplifica los cálculos y fórmulas utilizadas (Framiñán, 2019).

Cuando tratamos de estudiar un FMS cerrado el procedimiento a seguir consiste en ir reduciendo la tasa de entrada y en cada iteración ir evaluando los resultados del sistema hasta encontrar y satisfacer las limitaciones de *wip* (prueba-error) (Framiñán, 2019).

Aplicando esta idea hemos desarrollado el programa informático. Este va realizando iteraciones, comprobando en cada una de ellas los datos del *wip* del sistema. Dependiendo del *wip* obtenido en cada iteración, aumentaremos o disminuiremos la tasa de entrada. Como el *wip* es directamente proporcional al input del sistema, un incremento de la tasa de entrada propiciará un aumento del *wip* en el sistema.

Además, la herramienta informática desarrollada, está automatizada. De forma que el usuario tan solo deberá crear un fichero de texto con una serie de datos del sistema que quiere estudiar e insertar la limitación de *wip* que desee. El programa, tras realizar una serie de iteraciones, mostrará por pantalla la tasa de entrada necesaria para mantener el *wip* deseado constante dentro del sistema; además de otros datos más específicos como Tiempo de Ciclo (CT), *X-factor*, Tiempo de espera (QT) entre otros.

4 APLICACIÓN DE LA METODOLOGÍA

INTRODUCCIÓN

Introducidos los conceptos y métodos más importantes utilizados para el desarrollo de nuestro programa pasamos a explicar con más detalle los requerimientos y funcionamiento del mismo. Lo ilustraremos con la resolución de dos ejercicios paso a paso. El primero de ellos resuelve un sistema *Conwip* y el segundo es una combinación de este y el sistema *Kanban.*

4.1 Funcionamiento del programa

Los sistemas para el control de la producción, o sistemas *pull*, nos permiten mantener constante el *wip* dentro de nuestro sistema. Como ya vimos en el apartado 3.2.1, los dos sistemas más utilizados son el sistema *Conwip* y el sistema *Kanban.* En el programa desarrollado contamos con la posibilidad de elegir cualquiera de las dos opciones.

El método de cálculo de input necesario para mantener el *wip* constante en ambos casos es parecido. Pasamos a explicarlos de manera detallada.

4.1.1 Sistema Conwip

Lo primero que debemos de conocer es la forma de inicializar nuestro programa. Su ejecución requiere de dos parámetros o argumentos:

- El primero es el nombre de un fichero de texto
- El segundo es un número: 1 ó 2 en función de si queremos analizar el sistema *Conwip o kanban*

C:\> *ConstantWip* <<parametro1>> <<parametro2>>

Por ejemplo:

C:\> *ConstantWip fichero.txt 1*

El primer parámetro es un fichero de texto que debemos crear con el formato y los datos que veremos más adelante en el apartado 4.1.2. Contiene esencialmente la información más relevante del sistema con el que vamos a trabajar: número de máquinas, tiempos de las máquinas, variabilidades de tiempo, etc.

El segundo parámetro nos permite elegir el tipo de sistema que queremos estudiar. Si introducimos el número 1 analizamos el sistema *Conwip* y si es el número 2 el sistema *Kanban.*

Una vez ejecutado el programa, nos muestra en pantalla el valor mínimo de *wip* que nos puede ofrecer ateniéndonos a los datos del sistema introducidos en el fichero de texto. A su vez, nos pide que introduzcamos el *wip* que deseamos obtener y un pequeño épsilon para dar un intervalo de valores aceptables al sistema.

$\omega_c + \varepsilon$ (1)

$\omega_c - \varepsilon$ (2)

(1) Valor máximo del *wip* deseado.

(2) Valor mínimo del *wip* deseado.

```
C:\Users\usuario\Desktop\Programas\Ejercicios C>ConstantWip prueba.txt 1
ShopAnalyzer (v0.3) -  © Jose M Framinan 2018-2020
This is a prototype console executable to analyse the main state variables of a general shop.

The tag STAGE_NAMES has not been found in file nuevo.txt.
The tag TIME_TRAVELLING_MATRIX has not been found in file nuevo.txt.
Bottleneck is workstation MHS. Capacity of the system: th*= 0.2352941 units/time unit.

El valor minimo de wip que puede ofrecerse es 1.726653
Introduce epsilon: 0.1
Introduce wip: 15
```

Imagen 4.1 Entrada al sistema

Fuente: Elaboración Propia

A partir de este momento, el programa inicia la fase de iteración automática hasta encontrar un resultado que esté dentro del rango especificado.

Finalmente, nos muestra en pantalla el input necesario para obtener un valor de *wip* que cumpla con los requisitos solicitados y los valores de las variables de *ct,* t_0 *y x-factor.*

También nos genera un fichero de salida (solución.txt) que contiene gran cantidad de información: datos de sistema, de estación o para cada tipo de trabajo.

```
ShopAnalyzer (v0.3) -  © Jose M Framinan 2018-2020
This is a prototype console executable to analyse the main state variables of a general shop.

The tag STAGE_NAMES has not been found in file nuevo.txt.
The tag TIME_TRAVELLING_MATRIX has not been found in file nuevo.txt.
Bottleneck is workstation MHS. Capacity of the system: th*= 0.2352941 units/time unit.
0.231395

Para obtener un wip de 15.061790 necesitaria un input de: 0.231395

Su QT= 53.591251
Su CT= 65.091248
Su X_factor= 5.660109

El numero de iteraciones para encontrar un resultado ha sido de 11
Si desea mas informacion sobre los datos obtenidos, abra el archivo solucion.txt generado en su carpeta
```

Imagen 4.2 Salida del Sistema

Fuente: Elaboración Propia

4.1.2 Sistema Kanban

Como hemos adelantado al comienzo de este capítulo, la forma de trabajar de este segundo sistema es muy parecida a la expuesta en el punto 4.1.1.

El programa se ejecutará pasándole dos parámetros (igual que en el sistema anterior).

El primer parámetro es un fichero de texto. Podemos utilizar el mismo que hemos creado para trabajar en el sistema anterior *(Conwip)* o generar uno nuevo. En todo caso, debe recoger los datos de nuestro FMS y añadirlos al formato establecido.

El segundo parámetro, que nos permite elegir el tipo de sistema que queremos estudiar, es el número 2 que se corresponde con el sistema *Kanban.*

Por ejemplo:

C:\> *ConstantWip* fichero.txt 2

En este caso nos muestra en pantalla una tabla con los valores mínimos de *wip* que nos puede ofrecer; en este sistema, a nivel de estación. Podemos ir introduciendo estación por estación la restricción de *wip* deseada. Si no deseamos restricción en alguna de ellas debemos introducir un 0.

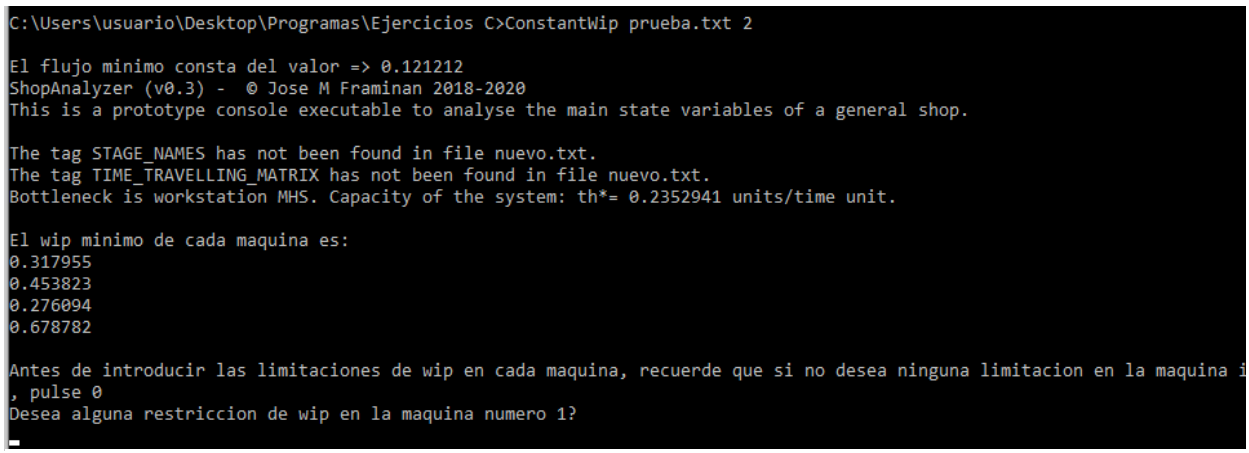

Imagen 4.3 Entrada al Sistema I

Fuente: Elaboración Propia

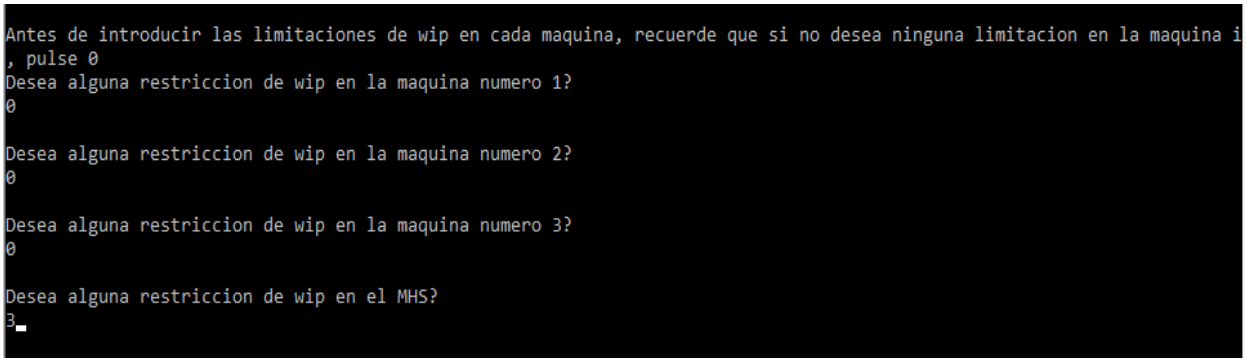

Imagen 4.4 Entrada al Sistema II

Fuente: Elaboración Propia

El programa inicia la fase iterativa expuesta en el apartado 4.1.4 hasta encontrar un valor de input que cumpla con todas las restricciones de *wip* a nivel de estación. Al igual que en el sistema *Conwip,* cuando se encuentra una solución nos la muestra por pantalla junto a los valores de las variables *ct, t0 y x-factor* y nos generará el fichero de salida "solución.txt".

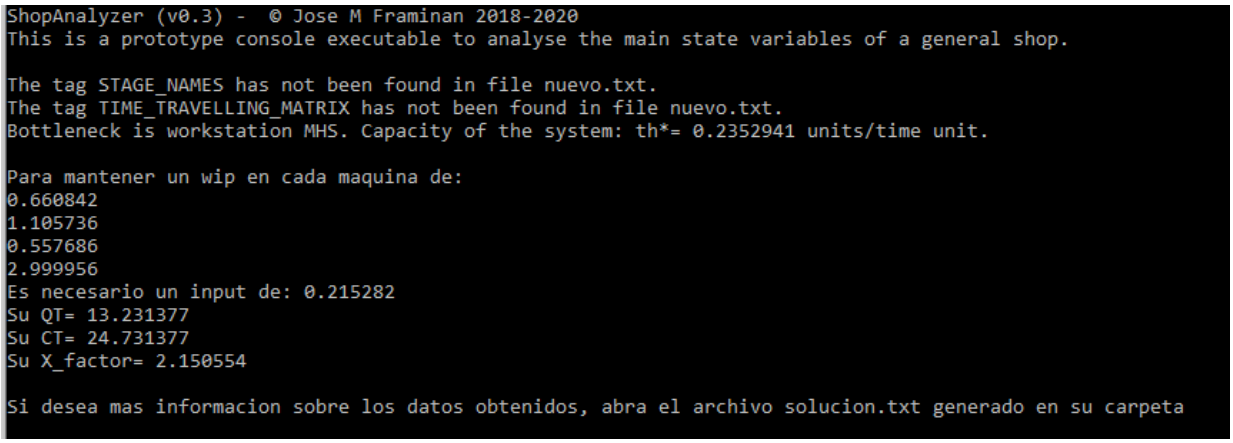

Imagen 4.5 Salida del Sistema Kanban

Fuente: Elaboración Propia

4.1.3 Fase de iteración. Sistema Conwip

En el apartado 3.4.4 explicamos qué era un sistema abierto y cómo la forma de analizarlo con este programa nos ayuda a convertirlo en un sistema cerrado.

En el punto anterior hemos mencionado que en esta fase iterativa necesitamos ofrecer al usuario el valor mínimo de *wip* que puede solicitar dependiendo de los datos del sistema introducidos en el fichero de texto.

Para obtenerlo, utilizamos la fórmula (3) que calcula la tasa mínima de entrada (Framiñán, 2019). Con esta tasa de entrada y aplicando las fórmulas vistas en el apartado 3.4.3 que utiliza el programa *ShopAnalyzer* logramos el *wip* mínimo que podemos conseguir en el sistema.

$$\underline{\gamma} = \frac{1}{ts(MHS)+\sum_{i=1}^{n} ts(i)} \quad (3)$$

En el denominador (3) nos encontramos la suma de todos los tiempos de servicio de las estaciones y el tiempo de servicio del MHS.

Entre los datos que obtenernos cuando ejecutamos por primera vez el *ShopAnalyzer* encontramos el parámetro de la capacidad del sistema *(th^*),* es decir, la tasa de entrada máxima que el sistema puede manejar sin convertirse en un sistema inestable.

El *wip* máximo puede obtenerse de forma análoga al *wip* mínimo. Sin embargo, al introducir el input máximo (capacidad del sistema), obtenemos valores de *wip* infinitos, ya que este input es el valor límite. Este input máximo (4) se obtiene cuando conseguimos una congestión del 100% en la estación cuello de botella (b).

$$\overline{\gamma}\ tal\ que\ \rho(b) = 1 \quad (4)$$

Como hemos anticipamos en el punto anterior, este valor de *wip* mínimo que hemos calculado se muestra por pantalla al usuario y nos pide que introduzcamos tanto el *wip* como el épsilon que deseamos obtener. Ambos marcan un rango donde obtener el resultado:

$$[\omega_c - \varepsilon]$$

$$[\omega_c + \varepsilon]$$

En el proceso interno, el programa se irá moviendo entre el valor mínimo del input $(\underline{\gamma})$ y el valor máximo del input (th^*). Como explicamos más adelante, se va comprobando el *wip* para cada uno de estos valores hasta conseguir alguno que esté dentro del rango que hemos solicitado.

Para entrar en este rango y calcular un valor de *wip* válido podemos utilizar, al menos, dos procesos de búsqueda: lineal o secuencial y binaria.

La primera consiste en ir incrementando poco a poco el valor del input mínimo hasta entrar en el rango deseado. Este tipo de búsqueda es sencillo de implantar pero, por lo general, poco práctica para listas de gran tamaño.

La segunda es más rápida y eficiente. Introducida por primera vez por John Mauchly en 1946 (*Búsqueda Binaria*, 2020), también recibe el nombre de búsqueda de intervalo medio (Williams, 2020) o búsqueda logarítmica por Knuth 1998 (*Algoritmos de Ordenación y Búsqueda*, 2020), consiste en comparar el valor buscado con el elemento medio del segmento que tenemos; si no son iguales, la mitad donde no puede estar el valor es eliminada y continuamos la búsqueda en la mitad restante hasta encontrar el valor.

Nuestro programa utiliza la búsqueda binaria y calcula, en primer lugar, el punto medio de ambas tasas

de entrada: mínima *(γ)* y máxima *(th)*. Esta tasa (valor del punto medio calculado) se pasa al programa *ShopAnalyzer* que nos proporciona el valor del *wip* del sistema. Este valor conduce a tres posibles casos:

Caso 1. El valor de *wip* extraído se encuentra dentro del rango deseado. El programa finaliza devolviendo los datos comentados en el apartado 4.1.1 junto al fichero solución.

Caso 2. El valor de *wip* extraído se encuentra por encima del rango deseado, y por tanto, necesitamos reducir el *wip* general. Para ello, disminuimos la tasa de entrada. El nuevo valor de la tasa de entrada es el punto intermedio entre la tasa mínima y el input anterior.

Caso 3. El valor de *wip* extraído se encuentra por debajo del rango deseado, y por tanto, necesitamos aumentar el *wip* general. Para ello, aumentamos la tasa de entrada. El nuevo valor de la tasa de entrada será el punto intermedio entre la tasa máxima y el input anterior.

Podemos comprobar que esta forma de búsqueda binaria del rango deseado es más eficiente que la búsqueda lineal, que busca el rango variando mínimamente el input.

En cada iteración analizamos un FMS abierto (no hay limitación de trabajos en el sistema) pero con el proceso iterativo generado conseguimos analizar el FMS como un sistema cerrado (con limitación de trabajos en el sistema).

4.1.4 Fase de iteración. Sistema Kanban

El proceso para obtener la tasa de entrada mínima y máxima es el mismo que en el apartado 4.1.3. Sin embargo, los valores mostrados por pantalla y solicitados son diferentes. Así, en este sistema, se nos presenta por pantalla para cada estación los valores mínimos de *wip* que podemos alcanzar e introducimos los que deseamos para cada una de ellas. Internamente, *ConstantWip* suma los *wip* de cada estación obteniendo un *wip* general del sistema.

El modo de proceder para calcular este *wip* es muy parecido al sistema anterior. Disponemos de un rango de valores de input que se mueve entre el valor mínimo del input *(γ)* y el valor máximo del input (th^*): $[\gamma, th^*]$. Sin embargo, en este caso no conocemos un rango de posibles valores de *wip*, sino un único valor de *wip*, el general de todas las estaciones.

Seguimos utilizando el método de búsqueda binaria para ir acotando el valor o valores de input que cumplan las restricciones del sistema.

El modo de proceder es el siguiente: calculamos el punto medio (pm) del rango de input mencionado anteriormente. Si el valor *wip* de este (obtenido tras someter al programa *ShopAnalyzer* el valor de input del punto intermedio conseguido) satisface las restricciones de cada estación se convierte en una primera solución y lo guardamos.

Continuamos repitiendo el proceso buscando más soluciones. La diferencia en esta segunda iteración se encuentra en que el rango donde buscamos la solución es el valor del punto medio de input anterior y el input máximo: **[pm,** $\boldsymbol{th^*}$**].**

Si en esta segunda iteración, seguimos obteniendo una posible solución de *wip* con el valor del punto medio de intervalo utilizado *(pm2),* se guarda y continuamos con otra iteración pero en este caso el rango es: **[pm2,** $\boldsymbol{th^*}$**].** Así continuamos hasta que los valores máximo y mínimo del rango se cruzan.

En este caso particular el punto medio se convierte siempre en el mínimo del rango y la solución definitiva es la última antes de producirse el cruce.

La evolución de los rangos es:

$[\gamma, th*]$ → $[pm, th*]$ → $[pm2, th*]$ → $[pm3, th*]$ →….

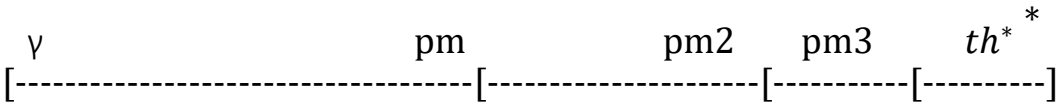

Imagen 4.6 Ejemplo de Rangos en Sistema Kanban I

Fuente: Elaboración Propia

Durante el proceso iterativo hemos considerado que el desplazamiento siempre se realiza hacia la derecha de la línea imaginaria que mostramos en la figura. Esto es así porque en cada iteración hemos encontrado una posible solución.

Pero puede ocurrir todo lo contrario, es decir, el desplazamiento siempre se produce hacia la izquierda. ¿Cuándo ocurre esto? Pues cuando desde la primera iteración no encontramos soluciones que cumplan con las restricciones del sistema. En este caso, cada vez que realizamos una iteración, con la búsqueda del punto medio, este se convierte en el valor máximo del rango. El proceso finaliza cuando se cruzan los valores máximo y mínimo del rango.

Utilizando los nombres con que hemos designado los extremos de los rangos, la evolución es:

$[\gamma , th*] \rightarrow [\gamma,pm] \rightarrow [\gamma ,pm2] \rightarrow [\gamma, pm3] \rightarrow \ldots.$

Y representándolos en una línea:

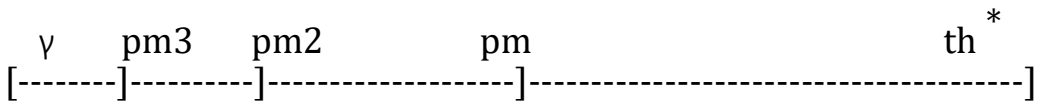

Imagen 4.7 Ejemplo de Rango Sistema Kanban II

Fuente: Elaboración Propia

Obviamente, durante el proceso se pueden presentar situaciones en que se alternen los dos casos extremos vistos anteriormente.

Así, el punto medio del rango en que buscamos se convierte en extremo inferior o superior en función de si se encuentra o no un valor de input que consideremos solución al problema.

Si hemos conseguido más de una solución, el valor del input más alto sería la solución definitiva.

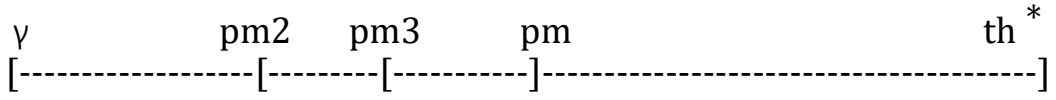

Imagen 4.8 Ejemplo de Rango Sostema Kanban III

Fuente: Elaboración Propia

4.2 Interacción usuario-programa

En esta parte del capítulo 4, se explica el manejo del ***ConstantWip*** para su correcto funcionamiento y se describen las distintas etiquetas que se utilizan en el mismo.

4.2.1 Apertura del programa

Como vimos en el apartado 4.1.1 para inicializar el programa es necesario pasarle dos parámetros o argumentos. Es decir, para ejecutar el programa, abrimos una sesión DOS (interprete de comandos) en Windows, nos posicionamos en el directorio donde se encuentre e introducimos el nombre del programa y separados por espacios los dos argumentos:

C:\ruta del programa> *ConstantWip* <<fichero_entrada.txt>> << tipo_analisis>>

El primer parámetro <<fichero_entrada.txt>> es un fichero de texto que contiene todos los datos necesarios que describen el sistema productivo que vamos a analizar. En apartado 4.2.2 lo veremos más detenidamente.

El segundo parámetro de entrada es un número:

Si es 1, nuestro FMS tendrá una limitación de capacidad general en el sistema. Sistema *Conwip.*

Si es 2, nuestro FMS tendrá limitaciones de capacidad en las máquinas que el usuario seleccione. Sistema *Kanban.*

4.2.2 Fichero de entrada

El fichero de texto de entrada está compuesto por datos etiquetados. Las etiquetas pueden ser obligatorias u opcionales. Las obligatorias y, por tanto, más importantes son (Framiñán, 2020):

- MACHINES_STAGE: Indica en un vector el número de máquinas idénticas que tiene una estación. En este vector excluimos la cantidad de sistemas de manejo de materiales idénticos que hay en el FMS.

- JOBS: Indica el número de trabajos distintos que se van a manejar.
- MHS: Indica el número de sistemas de manejo de materiales idénticos con los que contamos.
- PRODUCT_MIX: Indica en un vector el mix de productos que vamos a utilizar. Es decir, la proporción del total de la tasa de entrada que entra en el FMS de cada trabajo.
- ROUTING_INFO: La matriz routing info muestra en cada una de sus filas la secuencia (estaciones que visita) que sigue cada trabajo a través del FMS. Esta matriz la escribimos en forma vectorial donde el signo ';' representa el final de la ruta de un trabajo.

$$\begin{matrix} 0 & 1 & 2 \\ 1 & 2 & \\ 1 & 0 & 1 \end{matrix} \qquad (5)$$

Nos encontramos con una matriz que contiene el mismo número de filas que número de trabajos (cada fila corresponde con un tipo de trabajo) y un número de columnas variable.

Puede ocurrir que el mismo trabajo pase varias veces por la misma estación. Como ocurre en la fila tres (5).

A la hora de introducir esta matriz en el fichero de texto la etiqueta quedaría:

[ROUTING_INFO = 0,1,2;1,2;1,0,1]

- AVG_PROCESSING_TIMES: Una matriz con la misma dimensión que la anterior etiqueta, donde cada elemento representa el tiempo de servicio medio que tarda cada estación para cada uno de los distintos tipos de trabajo.

$$\begin{matrix} 5 & 7 & 12 \\ 6 & 2 & \\ 9 & 3 & 7 \end{matrix} \qquad (6)$$

Al igual que ocurría en la anterior etiqueta, podemos encontrarnos con un producto que vuelve a una estación con un tiempo de proceso distinto. Fila tres (5) (6), donde la estación 1 tarda primero nueve unidades de tiempo y posteriormente siete unidades de tiempo.

Así mismo, al introducir la etiqueta en el fichero de texto, la fila quedaría:

[AVG_PROCESSING_TIMES = 5,7,12;6,2;9,3,7]

- VAR_PROCESSING_TIMES: De nuevo una matriz con la misma dimensión que las anteriores matrices. Cada elemento muestra la variabilidad en el tiempo de servicio de cada estación para cada tipo de trabajo.
- MHS_AVG_PROCESSING_TIMES: Indica un número entero que representa el tiempo de transporte del sistema de manejo de materiales.
- MHS_VAR_PROCESSING_TIMES: Número entero que representa la variabilidad en el tiempo de transporte del sistema de manejo de materiales.

  Para facilitar el análisis y desarrollo del programa hemos considerado que el tiempo de transporte del sistema de manejo de materiales no depende del tipo de trabajo, sino que es el mismo para cada uno de los productos.

- SUM_AVG_INPUT_FLOW: Corresponde a un valor numérico entero. Es la tasa de entrada total que recibe nuestro FMS, es decir, la suma de las tasas de entrada media de los productos.
  Nuestro programa hace uso de esta etiqueta para introducir el dato al *ShopAnalyzer* en cada iteración, por tanto, es imprescindible que aparezca aunque el valor aportado por el usuario sea intrascendente.
- VAR_INPUT_FLOW: Es un vector de tantas posiciones como número de trabajos existentes que representan la variabilidad en la tasa de entrada.

Las etiquetas opcionales:

- STAGE_NAMES: Escribimos en un vector los nombres para referirnos a cada estación del FMS.
- JOB_NAMES: Es un vector que contiene los nombres de los diferentes trabajos.

A continuación, mostramos un ejemplo de cómo quedaría el fichero de entrada de los datos del sistema. (Imagen 4.9)

```
[MACHINES_STAGE=1,1,1]
[STAGE_NAMES=NC1,NC2,NC3]
[JOBS=3]
[JOB_NAMES=A,B,C]
[MHS=1]
[PRODUCT_MIX=0.4,0.35,0.25]
[ROUTING_INFO=1,2,1,3;1,3,2;2,3]
[AVG_PROCESSING_TIMES=25,36,45,10;20,25,30;35,100]
[VAR_PROCESSING_TIMES=0,0,0,0;0,0,0;0,0]
[MHS_AVG_PROCESSING_TIMES=1]
[MHS_VAR_PROCESSING_TIMES=0]
[SUM_AVG_INPUT_FLOW=1]
[VAR_INPUT_FLOW=0,0,0]
```

Imagen 4.9 Ejemplo de Fichero de Entrada

Fuente: Elaboración Propia

En el ejemplo de la imagen 4.9, contamos con un sistema de tres estaciones (NC1, NC2, NC3). En cada una de ellas una máquina. Queremos fabricar tres tipos de productos (A, B, C) cuya proporción de tasa de entrada es 0.4, 0.35 y 0.25 respectivamente. La ruta de fabricación del producto A es la estación 1,2,1,3. La ruta del producto B es 1,3,2 y la del producto C es 2,3. Contamos con un sistema de manejo de materiales que tarda 1 unidad de tiempo en desplazar los productos de forma determinista (no hay variabilidad). Finalmente, los tiempos de servicio de las estaciones se muestran en la línea avg_processing_times. Cómo ya avanzábamos, nos podíamos encontrar con productos que revisitaran estaciones y tuvieran tiempos de servicio diferentes, cómo el producto A en la estación 1, primero la duración es de 25 y posteriormente es de 45. Estos tiempos también son deterministas.

El número introducido en la etiqueta sum_avg_input_flow por parte del usuario no lo tendremos en cuenta. Sin embargo, es necesario que aparezca en el fichero de entrada para el correcto funcionamiento del programa.

4.3 Ejemplos y resolución

En este apartado vamos a resolver varios ejemplos paso a paso.

El primer ejercicio servirá para utilizar el primer sistema de nuestro programa (sistema *Conwip*); mientras que para el segundo ejercicio utilizaremos el segundo sistema (sistema *Kanban*). Recorreremos todo el proceso, desde la entrada del archivo hasta la obtención de las variables más importantes del programa informático.

Los siguientes ejercicios son tomados del boletín de problemas y apuntes (Framiñán, 2019).

Ejercicio 1.

Tenemos una célula de fabricación flexible como la de la figura 2, donde un robot alimenta tres estaciones de soldadura. Se procesan tres tipos de productos (puertas), cuyos tiempos en segundos y mix son:

Tipo puerta	*Tiempo soldadura*	*Mix*
A	*10*	*30 %*
B	*5*	*40 %*
C	*14*	*30 %*

El tiempo del robot es de dos segundos (determinista) para desplazar las piezas de la estación de carga y descarga a las estaciones de soldadura. Debido a las características de espacio que tiene el sistema, el inventario no puede ser superior a cuatro (una pieza por cada estación y otra por el robot). El flujo de llegada de las piezas es exponencial, ya que son transportadas por operarios. Asumimos que los tiempos de soldadura son deterministas.

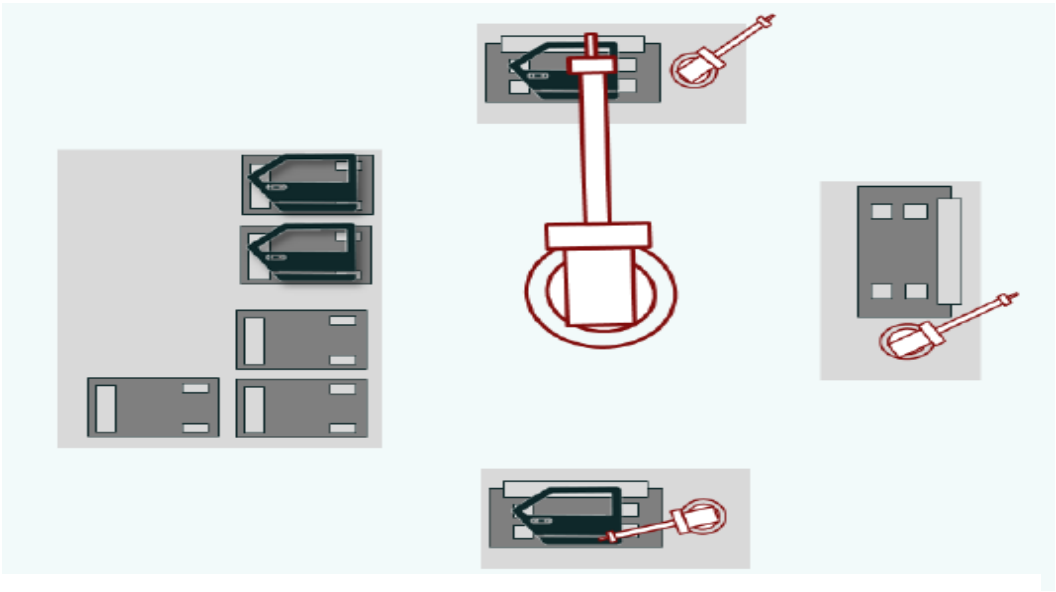

Imagen 4.10 FMS

Fuente: (Framiñán, 2019)

1. *Determinar el cuello de botella.*
2. *Tasa de entrada para mantener constante el sistema con 4 piezas y ct del sistema.*
3. *Tasa de entrada para mantener constante el sistema con 3 piezas y ct del sistema.*

Resolución:

En primer lugar, debemos preparar el fichero de texto con los datos de nuestro sistema. Siguiendo el apartado 4.2.2. escribimos las etiquetas necesarias, quedando el texto como se muestra en la imagen 4.11.

```
[MACHINES_STAGE=3]
[JOBS=3]
[JOB_NAMES=A,B,C]
[MHS=1]
[PRODUCT_MIX=0.3,0.4,0.3]
[ROUTING_INFO=0;0;0]
[AVG_PROCESSING_TIMES=10;5;14]
[VAR_PROCESSING_TIMES=0;0;0]
[MHS_AVG_PROCESSING_TIMES=2]
[MHS_VAR_PROCESSING_TIMES=0]
[SUM_AVG_INPUT_FLOW=1]
[VAR_INPUT_FLOW=1,1,1]
```

Imagen 4.11 Fichero de Entrada. Ejercicio 1

Fuente: Elaboración Propia

Machines_stage: Tenemos tres estaciones de soldadura idénticas.

Jobs: Realizamos tres piezas distintas o puertas diferentes.

Mhs: Tenemos un robot que desplaza los productos.

Product_mix: Lo obtenemos de la tabla del ejercicio con los datos. Cada número representa un tipo de producto diferente.

Routing_info: Solo tenemos un tipo de máquina (estación de soldadura). Cada tipo de producto pasa solamente por una de estas. Representamos con un cero la primera estación de nuestro sistema. Si hubiese un segundo tipo distinto de máquina, la representaríamos con uno y así sucesivamente.

Avg_processing_times: Cada número representa el tiempo de proceso de la pieza con la estación de soldadura.

Var_processing_times: La variabilidad del tiempo de proceso de las distintas piezas es cero, ya que los tiempos de soldadura son deterministas.

Mhs_avg_processing_times: El tiempo que tarda el robot en desplazar las piezas. Tenemos que recordar que la programa asume que el tiempo de desplazamiento no depende del tipo de trabajo ni de pieza. En caso de encontrarnos con un ejercicio donde nos indiquen tiempos de desplazamientos distintos para cada tipo de trabajo, deberíamos hacer una aproximación para calcular el tiempo de desplazamiento del sistema de manejo de materiales.

Mhs_var_processing_times: La variabilidad es determinista.

Sum_avg_input_flow: Recordemos que esta etiqueta es necesaria escribirla, aunque el número escrito no será tenido en cuenta.

Var_input_flow: Al ser un trabajo humano, lo normal es encontrarnos variabilidades. En este caso, el texto nos dice que consideremos una variabilidad exponencial, es decir, una variabilidad de uno.

Preparado el fichero de texto, inicializamos el programa con los parámetros presentados en el apartado 4.2.1. Para solucionar los apartados de este ejercicio necesitaremos utilizar como segundo parámetro el sistema

tipo 1 (sistema *Conwip*).

Después de escribir los parámetros, nos mostrarán por pantalla las preguntas del épsilon y del *wip* que queremos para nuestro sistema. Generalmente, el épsilon será un número muy pequeño que permite al programa la búsqueda del *wip* deseado con un pequeño margen de error. En nuestro caso, hemos introducido un épsilon de 0.1.

Por otro lado, el sistema debe tener cuatro piezas en el sistema, por lo que introducimos un *wip* de 4 (imagen 4.12). De esta forma, nuestro programa buscará la solución dentro del rango 3.9 y 4.1

A la vez que el programa nos solicita introducir el épsilon y el *wip,* nos muestra el valor mínimo de *wip* que puede encontrar. Para los datos introducidos de nuestro ejemplo este valor (1.2909) es el encontrado al aplicar la fórmula de tasa mínima de entrada del apartado 4.1.3.

Como observamos en la imagen 4.12, gracias al uso del *ShopAnalyzer* sabemos cuál de las estaciones de nuestro sistema es el cuello de botella (Bottleneck is Workstation MHS). Además, nos indica la capacidad de nuestro sistema. En nuestro caso, de 0.25 unidades/segundo.

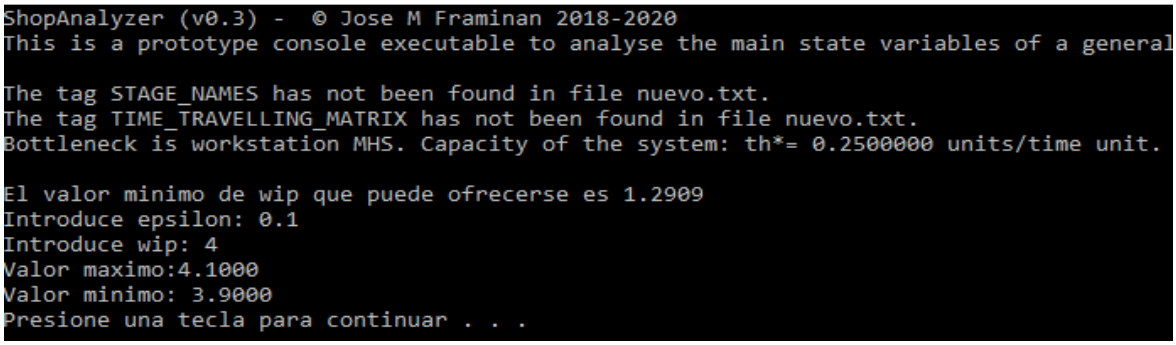

Imagen 4.12 Entrada al Sistema. Ejercicio 1

Fuente: Elaboración Propia

El programa realizará las iteraciones necesarias hasta encontrar un valor del *wip* que esté en nuestro rango deseado. Al encontrarlo, el programa mostrará por pantalla el input necesario para mantener el *wip* deseado constante. En este caso, para conseguir el *wip* de 3.9136 necesitaremos una tasa de entrada de 0.1948 unidades/segundo. Finalmente, muestra algunos datos importantes del sistema por pantalla e invita al usuario a abrir el archivo solución que generado donde encontraremos más datos específicos del sistema (imagen 4.13).

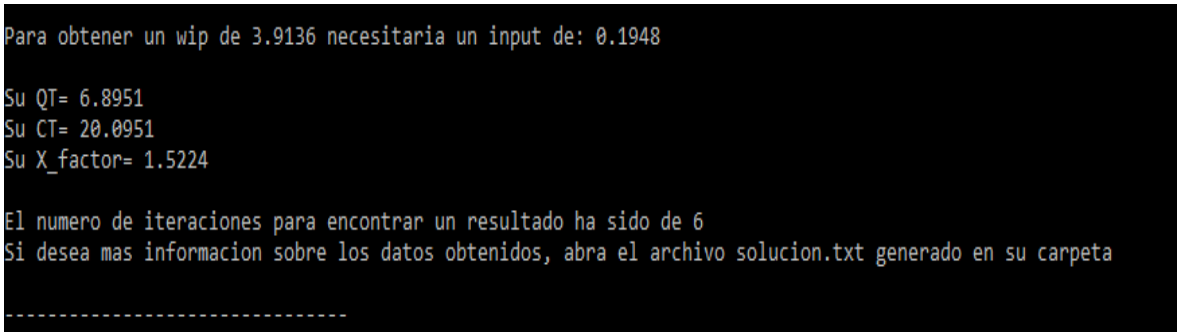

Imagen 4.13 Salida del Sistema. Ejercicio1

Fuente: Elaboración Propia

Hasta este momento, los datos extraídos del programa nos permiten contestar a las dos primeras preguntas. El cuello de botella del sistema es el MHS. La tasa de entrada para mantener 4 unidades en el sistema de forma constante es 0.1948 unidades/segundo y el tiempo de ciclo del sistema es de 20.0951

El archivo de texto que contiene las soluciones completas del sistema es el de la imagen 4.14. Las primeras seis etiquetas son datos que mostramos por pantalla con el programa informático. Las etiquetas con el encabezado de Workstation hacen referencia a las estaciones y al MHS, mientras que las etiquetas con el inicio de Job hacen referencia a cada uno de los distintos tipos de producto.

```
[SYSTEM_CAPACITY=0.25]
[SYSTEM_T0=13.2]
[SYSTEM_QT=6.89506393592899]
[SYSTEM_CT=20.095063935929]
[SYSTEM_WIP=3.91359408177791]
[SYSTEM_X_FACTOR=1.52235332847947]
[WORKSTATION_LAMBDA=0.194754,0.389508]
[WORKSTATION_ALPHA=1,2]
[WORKSTATION_THETA_A=0.658095413155849,0.804282429424216]
[WORKSTATION_THETA_D=0.608564858848431,0.316190826311698]
[WORKSTATION_RHO=0.5972456,0.779016]
[WORKSTATION_QT=1.22452777910012,2.83526807841444]
[WORKSTATION_CT=10.4245277791001,4.83526807841444]
[WORKSTATION_WIP=2.03021848309086,1.88337559868705]
[WORKSTATION_X_FACTOR=1.13310084555436,2.41763403920722]
[WORKSTATION_PROBABILITY_MATRIX=0,1;0.5,0]
[JOB_LAMBDA=0.0584262,0.0779016,0.0584262]
[JOB_QT=6.89506393592899,6.89506393592899,6.89506393592899]
[JOB_CT=20.095063935929,20.095063935929,20.095063935929]
[JOB_T0=13.2,13.2,13.2]
[JOB_X_FACTOR=1.52235332847947,1.52235332847947,1.52235332847947]
[JW_LAMBDA=0.0584262,0.1168524;0.0779016,0.1558032;0.0584262,0.1168524]
```

Imagen 4.14 Fichero "solución.txt". Ejercicio 1

Fuente: Elaboración Propia

Observando estos datos determinamos el cuello de botella en la etiqueta *Workstation_Rho*. Esta muestra la congestión de las estaciones y del MHS. Como vimos en el capítulo 3.4, el cuello de botella será aquel que tiene más congestión. En nuestro ejercicio es el MHS con una congestión del 77,90%

Finalmente, para resolver el apartado 3 procedemos de la misma forma. Ahora, en cambio, introducimos un *wip* de 3. Los datos obtenidos son los de la imagen 4.15. Necesitaremos una tasa de entrada de 0.1696 unidades/segundo con un tiempo de ciclo del sistema de 17.5880 segundos.

```
Para obtener un wip de 2.9837 necesitaria un input de: 0.1696

Su QT= 4.3880
Su CT= 17.5880
Su X_factor= 1.3324

El numero de iteraciones para encontrar un resultado ha sido de 2
Si desea mas informacion sobre los datos obtenidos, abra el archivo solucion.txt generado en su carpeta
```

Imagen 4.15 Salida del Sistema. Ejercicio 1 II

Fuente: Elaboración Propia

Ejercicio 2.

Tenemos un FMS con tres estaciones cuyas tasas de servicio son de 2 unidades/minuto, 3 y 2 respectivamente. En cada una de estas estaciones existe 1 máquina CNC. Actualmente contamos con un brazo robótico cuyo tiempo de desplazamiento de los distintos tipos de producto es de 1 minuto. Se está considerando implementar algún sistema tipo pull para procesar los dos tipos de productos con un mix de 75% y 25% respectivamente. Ambos productos deben pasar por todas las estaciones, además, el producto B, tiene que volver a reprocesarse en su última etapa en la segunda estación con un tiempo de 1 minuto. Asumir un comportamiento determinista, debido a la automatización de las estaciones.

1. *Máxima tasa de entrada para que un sistema Kanban no supere, en promedio, las tres tarjetas por estación.*
2. *Máxima tasa de entrada para que un sistema Conwip no supere, en promedio, las nueve tarjetas.*

Resolución:

En primer lugar, realizamos el fichero de entrada.

```
[MACHINES_STAGE=1,1,1]
[JOBS=2]
[JOB_NAMES=A,B]
[MHS=1]
[PRODUCT_MIX=0.75,0.25]
[ROUTING_INFO=0,1,2;0,1,2,0]
[AVG_PROCESSING_TIMES=2,3,2;2,3,2,1]
[VAR_PROCESSING_TIMES=0,0,0;0,0,0,0]
[MHS_AVG_PROCESSING_TIMES=1]
[MHS_VAR_PROCESSING_TIMES=0]
[SUM_AVG_INPUT_FLOW=1]
[VAR_INPUT_FLOW=0,0]
```

Imagen 4.16 Fichero Entrada. Ejercicio 2

Fuente: Elaboración Propia

Recordamos introducir como parámetros el nombre de nuestro fichero de texto y el numero dos para trabajar con el sistema *Kanban*.

El *ConstantWip* nos muestra los valores mínimos que puedo escoger para cada estación. Aunque nosotros, tal y como dice el ejercicio, exigiremos un *wip* de tres para cada una. (imagen 4.17)

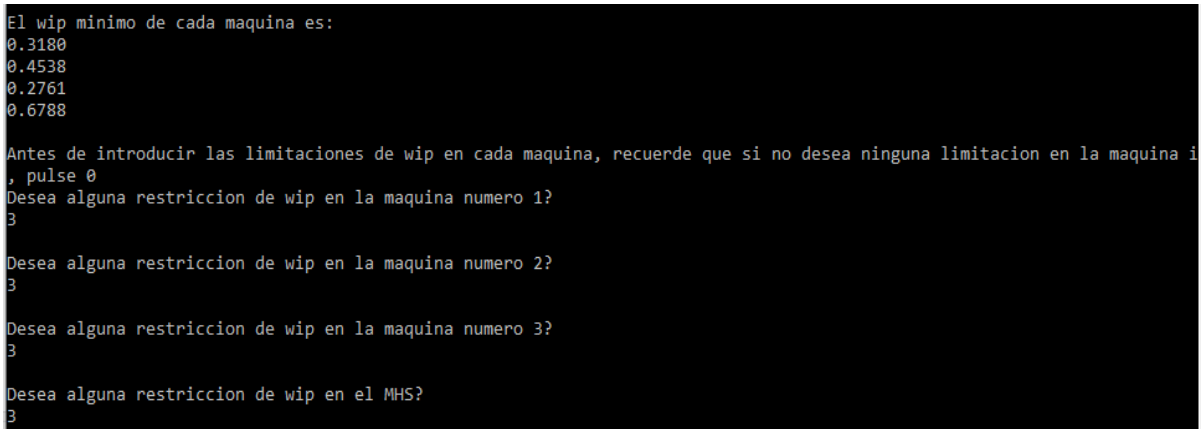

Imagen 4.17 Entrada al Sistema. Ejercicio 2

Fuente: Elaboración Propia

Como podemos observar, el *ConstantWip* nos pregunta por cada estación y, en último lugar, por la limitación del MHS.

Cuando encuentra la solución final nos genera el archivo solución.

Hemos resaltado la fila donde viene el *wip* para cada estación (imagen 4.18) para observar que para un input de 0.2153 (valor sacado por pantalla) hemos cumplido la restricción de tener menos de tres unidades por estación.

```
[WORKSTATION_LAMBDA=0.2691025,0.215282,0.215282,0.9149485]
[WORKSTATION_ALPHA=1.25,1,1,4.25]
[WORKSTATION_THETA_A=0.726177233750437,0.78094178700035,0.78094178700035,0.423668666697395]
[WORKSTATION_THETA_D=0.567382056942078,0.455197648121172,0.63616661416516,0.069002594751486]
[WORKSTATION_RHO=0.4843845,0.645846,0.430564,0.9149485]
[WORKSTATION_QT=0.655725626272736,2.13622377285176,0.590488517722832,2.27882524759576]
[WORKSTATION_CT=2.45572562627274,5.13622377285176,2.59048851772283,3.27882524759576]
[WORKSTATION_WIP=0.660841905344059,1.10573652626707,0.557685549072407,2.99995624204987]
[WORKSTATION_X_FACTOR=1.36429201459596,1.71207459095059,1.29524425886142,3.27882524759576]
[WORKSTATION_PROBABILITY_MATRIX=0,0,0,1;0,0,0,1;0,0,0,1;0.294117647058824,0.235294117647059,0.2
[JOB_LAMBDA=0.1614615,0.0538205]
[JOB_QT=12.4977389072304,15.4322897810989]
```

Imagen 4.18 Fichero "solución.txt". Ejercicio 2

Fuente: Elaboración Propia

Para realizar el apartado dos, nos sirve el mismo fichero de entrada. (Imagen 4.16) Debemos recordar pasar como segundo parámetro el 1.

Introducimos el nivel de *wip* que queremos mantener en el sistema, en nuestro caso, 9; y un épsilon pequeño, que para nosotros es, por ejemplo, el 0.01.

Comprobamos en la imagen 4.19 que el input necesario para mantener un *wip* constante en el sistema de 8.9987 unidades es de 0.2273 unidades/minuto.

```
Para obtener un wip de 8.9987 necesitaria un input de: 0.2273

Su QT= 28.0940
Su CT= 39.5940
Su X_factor= 3.4430

El numero de iteraciones para encontrar un resultado ha sido de 8
Si desea mas informacion sobre los datos obtenidos, abra el archivo solucion.txt generado en su carpeta
```

Imagen 4.19. Salida del Sistema. Ejercicio 2

Fuente: Elaboración Propia

5 CONCLUSIONES

5.1 Conclusiones

En este proyecto hemos desarrollado un programa informático en lenguaje C++ apoyándonos en otra aplicación informática, el ShopAnalyzer. El capítulo dos lo dedicamos a conocer, de forma resumida, el desarrollo de los sistemas productivos y las características de los FMS.

Hemos expuesto los conocimientos básicos para entender las variables más importantes del programa y bajo que fórmulas e ideas hemos conseguido desarrollarlo. El *ConstantWip* tiene como objetivo buscar una solución a la tarea de mantener constante el *wip* para cualquier valor. El programa devuelve el valor del flujo externo necesario para satisfacer el *wip* deseado por el usuario.

Aplicando el programa reducimos en gran medida los inventarios intermedios y finales y, en consecuencia, el coste de almacenamiento, ya que solo trabajamos con la cantidad de productos necesarios. Además, al estar desarrollado para un FMS, (aunque disponga de cierta flexibilidad) podemos adaptarnos más fácilmente a los cambios externos de la demanda y trabajar solamente con la cantidad necesaria para responder a la demanda real.

El *ConstantWip* ofrece la posibilidad de trabajar con 2 modelos de sistemas tipo *pull,* de mantenimiento del *wip*:

- Por un lado, contamos con un sistema tipo *Conwip*, que consigue mantener a nivel de sistema un *wip* constante. Con este sistema podemos disponer de más libertad dentro de cada estación, ya que la restricción es a nivel del sistema.
- Por otro lado, tenemos un sistema tipo *Kanban,* para restringir las estaciones deseadas a un wip deseado.

En el quinto capítulo hemos realizado dos ejercicios para explicar detalladamente cómo trabaja el programa informático y cómo nos encuentra una solución al ejercicio.

5.2 Líneas de investigación futuras

Podemos desarrollar algunas líneas de investigación para ampliar el trabajo expuesto en este proyecto.

- Implementar o añadir nuevas opciones al ConstantWip que otorguen de un catálogo más amplio de soluciones a problemas que se presentan actualmente en la actividad productiva de las empresas. Por ejemplo, averías en las estaciones, paradas planificadas o tiempos de *setup*. Mejorar el programa desarrollado teniendo en cuenta una o varias de estas variables conseguiría resolver problemas más complejos y comunes.

- Cuando mostramos el funcionamiento del programa en los dos ejercicios de ejemplo del capítulo cinco, podemos observar que pedimos al usuario que introduzca un valor deseado de *wip*. A nivel informativo mostramos por pantalla el *wip* minimo de referencia. Sin embargo, no podemos hacer lo mismo con un *wip* máximo que coincida con la tasa máxima que puede soportar el sistema porque no disponemos de ningún recurso (fórmula, procedimiento) que lo establezca. Si el usuario introduce un valor superior a la capacidad máxima del sistema, ConstantWip no encuentra una solución. Por tanto, una posible mejora de este sería buscar un valor de *wip* máximo o, al menos, una aproximación para facilitar una información más completa al usuario.

- Actualmente se considera que el tiempo de desplazamiento de los distintos productos por parte del MHS es idéntico. Pero en realidad vemos que dependiendo de alguna de sus características, tipo de producto, forma, peso, el tiempo de desplazamiento puede ser diferente para algunos de ellos. Además, puede darse el caso de que las diferentes estaciones no guarden la misma distancia ni entre ellas ni con respecto al MHS, por lo que el acceso a algunas de ellas puede ser mayor que otras. Por tanto, es recomendable

el estudio de cómo añadir la posibilidad de que el tiempo del MHS sea dependiente del tipo de trabajo realizado.

- Una de las variables decisionales más importantes para los sistemas tipo *pull* es establecer la cantidad de *wip* con la que vamos a trabajar. Existen distintos estudios que han intentado fijar un número óptimo de esta variable (Rodríguez et al., 2002). Para establecerla debemos conjugar la tasa de salida y el nivel de *wip* que vamos a mantener. Conforme aumentamos la tasa de salida (como máximo hasta la capacidad del sistema), elevamos el *wip*. Estas dos variables mantienen una relación de proporcionalidad de manera que el aumento o disminución de una produce un aumento o disminución de la otra. Una implementación de gran valor para este programa sería la búsqueda de la cantidad óptima de *wip* dependiendo de las características que introduzca el usuario sobre su sistema de producción.

BIBLIOGRAFÍA

¿Método numérico, analítico y experimental: concurrentes o complementarios en la ingeniería? (2017). https://www.esss.co/es/blog/simulacion-numerica-metodos-analitico-experimental-concurrentes-o-complementarios-en-la-ingenieria/

¿Por qué la metodología lean se llama lean? (2018). http://www.aretegestiona.es/metodologia_lean/

AGV-Automated Guided Vehicles (Vehículos De Guiado Automático). (2020). https://www.systemlogistics.com/spa/soluciones-y-proyectos/agv-automated-guided-vehicles-vehículos-de-guiado-automático

Algoritmos de ordenación y búsqueda. (2020).

Avila, S. (2016). *Ransom E. Olds: Drawing the line – StMU History Media*. https://stmuhistorymedia.org/ransom-e-olds-drawing-the-line/

Barba Álvarez, A. (2010). *Frederick Winslow Taylor y la administración científica: contexto, realidad y mitos*.

Barciela, C. (2005). La edad de oro del capitalismo, 1945-1973. In *Historia Económica Mundial, Siglos X-XX* (pp. 339–389).

Barrientos, A., Peñin, L., Blaguer, C., & Aracil, R. (2014). *Fundamentos de Robótica*.

BAWA, H. S. (2007). *Procesos de Manufactura* (1ª Edición). MCGRAW-HILL INTERAMERICANA.

Berenguer, J. M. (2015). *¿Qué es el modelo "LEAN" o de producción ajustada?* https://prevenblog.com/que-es-el-modelo-lean-o-de-produccion-ajustada/

Bertrand, J. W. M. (1983). The use of workload information to control job lateness in controled and uncontrolled release production systems. *International Journal of Operations Management*, *3*(2), 79–92.

Búsqueda Binaria. (2020). http://numerentur.org/busqueda/

C (lenguaje de programación). (2020). https://es.wikipedia.org/wiki/C_(lenguaje_de_programación)

CAD/CAM | Diseño y fabricación asistidos por ordenador. (2020). https://www.autodesk.es/solutions/cad-cam

Control numérico por computadora. (2020). https://he.wikipedia.org/wiki/Computer_Numerical_Control

Crece un 6% el número de robots industriales instalados en todo el mundo en 2018. (2019). https://www.interempresas.net/Robotica/Articulos/254914-Crece-un-6-por-ciento-el-numero-de-robots-industriales-instalados-en-todo-el-mundo-en-2018.html

Daniel, S., L., R., & Bulfin, J. (1998). *Planeación y control del aproducción*. MCGRAW-HILL INTERAMERICANA.

Definición de metodología just in time o justo a tiempo y cómo aplicarla. (2020). https://leanmanufacturing10.com/just-in-time

Deitel, H., & Deitel, P. (2004). *Cómo programa en C/C++ y Java* (4ª Edición). Pearson Educación.

Di mascolo, M., Frein, Y., & Y., D. (1996). An analytical method for performance evaluation of kanban controlled production systems. *Operations Research*, *44*(1), 50–64.

España vuelve al "Top10" de robots industriales instalados. (2019). http://www.automaticaeinstrumentacion.com/es/notices/2019/09/espana-vuelve-al-top10-de-robots-industriales-instalados-45793.php#.Xtt56sBS_cthttps://oasys-sw.com/poblacion-robotica-clave-

automatizacion-industria/

Estadísticas sobre incorporación de Robots Industriales en España en 2015. (2016). https://prograbox.com/estadisticas-robotica-industrial-espana-2016/

Flórez Pareja, O., Trejos Taborda, C., & Becerra Fernández, M. (2016). Modelo hibrido pull-push en el sector de las telecomunicaciones. *Revista Ingeniería Industrial*, *15*(2), 213–227.

Formación en sistemas de producción y fabricación. (2014). https://retos-operaciones-logistica.eae.es/formacion-especializada-y-optimizacion-del-control-de-los-sistemas-de-produccion-y-fabricacion/

Framiñan, J. M. (2020). *ShopAnalyzer* (0.3).

Framiñán, J. M. (2019). *"Sistemas Integrados de Producción", Escuela Técnica Superior de Ingeniería*.

Framiñán, J. M. (2020). *Manual de uso ShopAnalyzer*. *versión 03*.

Fullana Belda, C., & Urquía Grande, E. (2009). *Los modelos de simulación: Una herramienta multidisciplinar de investigación*.

Glassey, C. R., & Resende, M. G. C. (1988). A scheduling rule for release in semiconductor fabrication. *Operations Research Letters*, *7*, 213–217.

Goldratt, E. (2005). *La meta* (3ª Edición). Díaz de Santos, S.A.

Groover, M. (2007). *Fundamentos de manufactura moderna* (3ª Edición). McGraw-Hill.

Grúas para manejo de residuos. (2020). https://www.konecranes.com/es-es/industrias/residuos-en-energia/gruas-para-la-industria-de-la-produccion-energetica-mediante-residuos-y-biomasa/gruas-para

Harrison, P. G., & Patel, N. M. (1993). *Performance modelling of communication networks and computer architectures*. Addison-Wesley.

Hay, E. (1989). *Justo a Tiempo: La técnica japonesa que genera mayor ventaja competitiva*. Grupo Norma.

Hernández, J. C., & Vizán, A. (2013). *Lean manufacturing Conceptos, técnicas e implantación*. https://issuu.com/renataantunes7/docs/lean_manufacturing

Hill, K. (2020). *Imporancia de los transportadores en sistemas de automatización*. https://engmag.in/importance-of-conveyors-in-automation-systems/

Jáuregui, A. (2001). *Principios de la administración científica, Taylor y Ford - GestioPolis*. https://www.gestiopolis.com/principios-de-la-administracion-cientifica-taylor-y-ford/

Krafcik, J. F. (1988). *Triumph of the Lean Production System*.

Kulweic, R. (1985). *Materials Handling Handbook* (2ª Edición). Wiley.

Lambrecht, M., & Segaert. (1990). Buffer stock allocation and assembly type production lines. *International Journal of Operations & Production Management*, *10*(2), 47–61.

Le-Anh, T., & De Koster, M. B. M. (2004). *ERIM REPORT SERIES RESEARCH IN MANAGEMENT ERIM Report Series reference number A Review Of Design And Control Of Automated Guided Vehicle Systems*.

Lenguaje de alto nivel. (2020). https://es.wikipedia.org/wiki/Lenguaje_de_alto_nivel

LIFO FIFO para PowerPoint. (2020). https://hislide.io/product/lifo-fifo-ppt/

Liker, J. (2010). *Las claves del éxito de Toyota: 14 principios de gestión del fabricante más grande del mundo*. EDICIONES GESTION 2000.

Los 5 tipos de robots industriales más utilizados por las empresas. (2020). https://www.bfmx.com/automatizacion/tipos-de-robots-industriales-mas-utilizados/

Máquinas CNC: Todo lo que necesitas saber. (2020). https://www.stanser.com/como-funciona-una-maquina-cnc/

Muñoz, J. F. (2012). *Control de trabajo en proceso (WIP) para el aumento de la productividad*.

https://www.gestiopolis.com/control-trabajo-proceso-wip-aumento-productividad/

Norma Internacional ISO 8373. Robots and robotic devices. (2012). https://www.iso.org/obp/ui/#iso:std:iso:8373:ed-2:v1:en

Onieva, L., Cortés, P., Muñuzuri, J., Guadix, J., & Ibáñez, J. N. (2006). *METODOS CUANTITATIVOS Y ORGANIZACION DE LA PRODUCCION*. SINTESIS.

Origen y evolución del lean manufacturing. (2015). https://www.progressalean.com/origen-y-evolucion-del-lean-manufacturing/

Palacios, J. (2004, February 1). *Oldsmobile llega al final del camino*. https://www.elmundo.es/motor/2004/323/1075739684.html

Pedro, J., Sabater, G., & Rogle, G. (2015). *Teoría de colas*.

Pérez Pérez, M., & Martínez Sánchez, A. (2000). *Las tecnologías de automatización flexible en España Modelling the adoption of teleworking: An empirical study of resources and organisational factors View project Spanish Defence View project*. https://www.researchgate.net/publication/28120263

Ransom Eli Olds. (2020). https://es.wikipedia.org/wiki/Ransom_Eli_Olds

Rodríguez, P. L. G., Framiñán, J. M., & Ruíz-Usano, R. (2002). *Control de la Producción Mediante un Sistema con Inventario en Proceso Constante: CONWIP*.

Ruiz-Usano, R. Framiñán, J.M., Crespo, A., Muñoz, M. . (2020). Sistemas de control push-pull. un estudio comparativo. In *International Journal*. http://adingor.es/congresos/web/uploads/cio/cio2001/simulacion/US-5.pdf

Salas, A. (1991). *Curso de lenguaje "c."* https://es.slideshare.net/victdiazm/curso-de-lenguaje-c-angel-salas

So, K. C. (1990). The impact of buffering strategies on the performance of production line systems. *International Journal of Production Research*, 2293–2307.

Spearman, M. L. (1992). Customer service in pull production systems. *Operations Research*, 53–63.

Teoría de Colas o de Líneas de Espera. (2020). https://www.emprendices.co/teoria-colas-lineas-espera/

The Ten Principles of Material Handling. (2020). www.mhia.org

Vaughn, R. (1990). *Introducción a la Ingeniería Industrial* (2ª Edición). Editorial Reverté.

Williams, L. F. (2020). *A MODIFICATION TO THE HALF-INTERVAL*. 95–101.

Womack, J., Jones, D., & Roos, D. (2017). *La máquina que cambió el mundo*. Profit Editorial.