

# Evaluación bajo Simulación de un Sistema *Just In Time*

Christian Peter Winter, Francisco Muñoz Contreras

Ingeniería Técnica Industrial, Especialidad en Electrónica Industrial

**Resumen—** En las últimas décadas, el fuerte y constante crecimiento de la automatización en los procesos industriales ha dado lugar al desarrollo de diferentes sistemas integrados de fabricación con el fin de maximizar la eficiencia de la producción. Uno de los sistemas de fabricación más utilizados en la industria automovilística (industria líder en lo que a automatización de procesos se refiere) es el denominado sistema *Just-In-Time* (JIT) que fue definido por la *Toyota Motor Company* en los años 70, siendo parte del conocido *Toyota Production System* (TPS). El objetivo de este artículo será, pues, explicar detalladamente el comportamiento y la utilidad de los sistemas JIT a partir de diferentes simulaciones por ordenador.

## I. INTRODUCCIÓN

Para entender el comportamiento y la utilidad de los componentes JIT del TPS se realizarán varios experimentos a partir de un sistema simulado por ordenador, cosa que primeramente implicará la implementación de las rutinas de simulación sobre un entorno de programación como lo es Matlab. A partir de los mencionados experimentos nos será posible obtener resultados estadísticamente significativos sobre el comportamiento del sistema y así poder extraer conclusiones sobre el funcionamiento básico del TPS.

## II. EL TOYOTA PRODUCTION SYSTEM

El TPS es un sistema de producción altamente integrado que se centra en varios aspectos de la fabricación buscando, entre otros, el uso eficiente del personal y de los recursos. Sus principales objetivos son:

- Reducir los costes de producción.
- Aumentar la calidad de los productos.
- Reducir los tiempos de entrega.

A partir de aquí, se identificaron los elementos que impedían alcanzar los mencionados objetivos y se agruparon en 3 grandes grupos conocidos como “las 3 M” (del japonés *Muda*, *Mura* y *Muri*):

- *Muda*: Todo trabajo que no añade ningún tipo de valor al producto. Algunos ejemplos serían la corrección de defectos, la sobreproducción, el transporte o la espera.
- *Mura*: Cualquier tipo de desequilibrio en la producción como lo serían volúmenes de producción irregulares o planes de producción variables. Dan lugar a problemas en la planificación y el control.
- *Muri*: Situaciones que llevan a las máquinas o al personal al límite. Provoca problemas personales, averías en las máquinas y una consecuente reducción de la calidad del producto.

### A. *Just-In-Time*

Es un sistema de fabricación que pretende reducir el inventario y los costes de transporte y almacenamiento. Se basa en la idea de solo producir los productos necesarios en las cantidades necesarias y al ritmo que marque la demanda. Es decir, se basa en una filosofía tipo *PULL* en contraposición a la filosofía clásica o *PUSH*. A continuación se describen de manera detallada las dos filosofías:

1) Filosofía *PUSH*: Se basa en una planificación previa de los materiales requeridos (metodología llamada MRP, del inglés “*Material Requirements Planning*”) y en un control de inventario. Una vez realizada la planificación se comienza a producir. Al tener un flujo unidireccional (tanto la materia prima como la información viajan en una misma dirección, ver figura 1) tiene el inconveniente de requerir replanificación en caso de producirse cambios, cosa que genera retrasos y latencia. Además puede provocar acumulación de WIP en los cuellos de botella (máquinas cuya tasa de producción es inferior a la del resto de máquinas que componen el sistema).

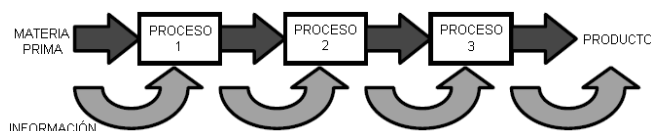


Fig. 1 Flujo unidireccional de un sistema *PUSH*.

2) Filosofía *PULL*: Se utiliza el MPS (“*Master Production Schedule*”, muy parecido al MRP) como guía pero la tasa de producción de cada una de las estaciones viene únicamente dada por la tasa de demanda de la siguiente estación. A modo de ejemplo, si la última estación de una cadena de producción recibe la orden de producir una pieza y no dispone de material, enviará una orden de arranque a la estación que le precede y así consecutivamente. De esta manera se obtiene un flujo bidireccional ya que la información viaja en sentido contrario a la materia prima (ver figura 2).

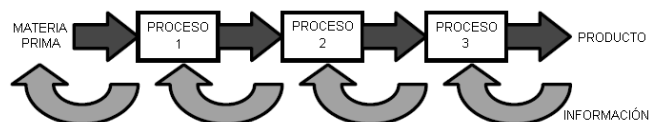


Fig.2 Ejemplo de un sistema *PULL* (el flujo es bidireccional).

A efectos prácticos, la filosofía tipo *PULL* disminuye considerablemente el WIP (disminuye el Muda), motivo por el cual es la filosofía en la que se basan los sistemas *Just-In-Time*.

### B. Los Kanbans (del japonés, “Tarjeta visual”).

El sistema de *Kanbans* es un sistema de tarjetas utilizado para controlar el transporte y la producción de materia y/o productos. Colocados en los distintos contenedores que se trasladan entre estaciones consecutivas, conllevan a la toma de decisiones locales generando así un comportamiento emergente, es decir, un sistema *PULL*. Existe gran variedad de *Kanbans* pero los más importantes son:

- *Kanbans* de Transporte (ver figura 3): Se utilizan para autorizar movimientos entre dos puntos de trabajo. Pueden contener información sobre el identificador del producto, su nombre, su enrutamiento, el tamaño de lote, etc.
- *Kanbans* de Producción (ver figura 3): autoriza la producción en una estación. Suele informar sobre el proceso a realizar.

Store Shelf No. 5E215 Item Back No. A42-15	Preceding Process	Store Shelf No. 726-18 Item Back No. A45-34	Process
Item No. 35670507	Forging IS-2	Item No. 56790-321	Machining SIS-8
Item Name Drive Pinion	Subsequent Process	Item Name Crankshaft	
Car Type SX5013C	Machining M-6	Car Type SX5013C-150	
Box Capacity 20 Box Type B Issued No. 418			

Fig. 3 Ejemplos de un *Kanban* de transporte (izquierda) y un *Kanban* de producción (derecha).

### C. Flujo de *Kanbans* entre dos procesos.

El flujo de *Kanbans* será siempre entre una estación precedente (*Preceding Stage*) y una siguiente (*Subsequent Stage*), teniéndose además una estación intermedia que hará la función de almacén (*Stacking Area*). El *Preceding Stage* se encargará de producir material en función de la tasa de producción actual. El material producido se almacenará en el *Stacking Area* y se irá consumiendo en el *Subsequent Stage* según la tasa de demanda del momento (ver figura 4).

A partir de los *Kanbans* se controlará todo el proceso anterior. Éstos se irán moviendo entre las estaciones o se dejarán en su caja correspondiente según unas reglas aplicables a cada una de las estaciones. Éstas reglas se especifican en el apartado III y deberían dejar más claro la funcionalidad del flujo de *Kanbans*.

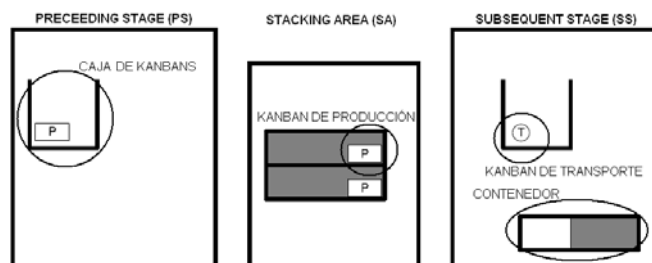


Fig. 4 Flujo de *Kanbans* entre dos procesos.

### D. Planificación y control de *Kanbans*

La clave para el correcto funcionamiento de un sistema JIT es el número total de *Kanbans* del que dispone. En concreto, el número de *Kanbans* de producción nos determinará la tasa de producción de nuestro sistema. Además el número de *Kanbans* de producción está en relación directa con el número de *Kanbans* de transporte. En consecuencia, será necesario determinar, mediante algún modelo de planificación, el número óptimo de *Kanbans*.

Los modelos más comunes son el modelo heurístico (ideado por *Toyota*) y el modelo probabilista. Al presentar éstos ciertos inconvenientes poco deseables, será una tercera alternativa la que usaremos en los experimentos posteriores. Esta tercera alternativa es la simulación y se explicará más en detalle en los apartados siguientes.

### E. Ventajas e inconvenientes de JIT

Como cualquier sistema de producción, el sistema *Just-In-Time* presenta ciertas ventajas e inconvenientes a tener en cuenta:

#### Ventajas:

- Aumento de la productividad.
- Aumento de la calidad.
- Reducción del tiempo de producción.
- Reducción del WIP.
- Menores requisitos de espacio.

#### Inconvenientes:

- Suele ser un sistema inadecuado en entornos con cambios grandes y frecuentes en la planificación de la producción.
- Poco preparado, en general, para responder a problemas o malfuncionamientos de máquinas y/o personal.

### F. Suposiciones y Simplificaciones

Para facilitar la implementación de las rutinas de simulación, haremos las siguientes suposiciones:

- Punto de trabajo simple: nos centraremos en un único *Preceding Stage*, *Stacking Area* y *Subsequent Stage*.
- Flujo básico: Aunque existan varios tipos de *Kanbans*, en la simulación solo utilizaremos *Kanbans* de producción y transporte.
- Funcionamiento síncrono: El TPS real funciona de manera totalmente asíncrona ya que cualquier acción se puede realizar en cualquier momento. En cambio, en nuestro caso, aplicaremos de manera secuencial y síncrona todas las reglas dentro de un mismo paso de simulación.
- Costes de sobreproducción y falta de material: Supondremos que un contenedor tiene una capacidad total de 1 por lo que las tasas de producción y consumo variarán entre 0 y 1. El coste de sobreproducción será igual a la suma de los contenidos de cada uno de los contenedores del sistema y el coste de falta de material se corresponderá, exactamente, con la cantidad de demanda que no se ha podido satisfacer.

- Número de *Kanbans* y contenedores: Supondremos que el número total de contenedores será igual al número total de *Kanbans* de transporte. Además, el número de *Kanbans* de producción será igual al número de *Kanbans* de transporte más dos.

### III. IMPLEMENTACIÓN

En el apartado anterior se ha dado una visión general del TPS y del flujo de información y materiales que involucra la utilización de los *Kanbans* de transporte y producción. Sin embargo, para la implementación de una filosofía *PULL*, además será necesario definir un conjunto de reglas aplicables a cada una de las 3 áreas que componen el sistema total (*Preceding Stage*, *Stacking Area* y *Subsequent Stage*):

#### A. Reglas aplicables al *Preceding Stage*

- Si tenemos un contenedor vacío con un *Kanban* de transporte, movemos el *Kanban* a un contenedor lleno del *Stacking Area*.
- Si disponemos de un contenedor vacío sin ningún *Kanban* y hay *Kanbans* de producción en la caja correspondiente, sacamos un *Kanban* de la caja y se lo ponemos al contenedor.
- Si tenemos un contenedor que aún no está lleno, seguimos llenándolo al ritmo que marque la tasa de producción.
- Si tenemos un contenedor lleno con un *Kanban* de producción, movemos el contenedor junto con su *Kanban* al *Stacking Area*.

#### B. Reglas aplicables al *Stacking Area*

- Si tenemos un contenedor lleno con los dos *Kanbans* (producción y transporte), movemos los *Kanbans* a sus cajas correspondientes y el contenedor al *Subsequent Stage*.

#### C. Reglas aplicables al *Subsequent Stage*

- Si tenemos un contenedor que aún no está vacío, seguimos vaciándolo al ritmo que marque la tasa de consumición.
- Si tenemos un contenedor vacío sin *Kanbans* y hay *Kanbans* de transporte en la caja correspondiente, sacamos un *Kanban* de transporte de la caja y se lo ponemos al contenedor.
- Si disponemos de un contenedor vacío con un *Kanban* de transporte, movemos el contenedor junto a su *Kanban* al *Preceding Stage*.

Una vez definidas conceptualmente todas las reglas, ahora se puede pasar a la implementación de éstas a nivel de código, en nuestro caso mediante *Matlab*.

Primeramente será necesario definir la estructura de los datos que se utilizarán para simular los distintos elementos que componen el sistema. Para ello utilizaremos un *Struct* global llamado *theSystem* en el que se guardará toda la información.

Aparte del programa principal se utilizarán varias subfunciones para implementar la simulación:

- *init\_system.m*: inicializa todos los parámetros del sistema y resetea *theSystem*.
- *filter\_containers.m*: Busca contenedores dentro del sistema según unas características preestablecidas.
- *update\_preceding.m*: aplica las reglas correspondientes al *Preceding Stage*.
- *update\_stacking.m*: aplica las reglas correspondientes al *Stacking Area*.
- *update\_subsequent.m*: aplica las reglas correspondientes al *Subsequent Stage*.
- *draw\_system.m*: visualiza de forma gráfica los resultados de la simulación y el flujo de *Kanbans* dentro del sistema (ver figura 5).
- *compute\_cost.m*: calcula los costes.

El programa principal (llamado *tps.m*) únicamente se encargará de realizar los diferentes pasos de simulación dependiendo de las tasas de producción y demanda previamente introducidas por el usuario. Para ello se realizará un bucle que iterará secuencialmente las funciones de *update\_preceding*, *update\_stacking* y *update\_subsequent*.

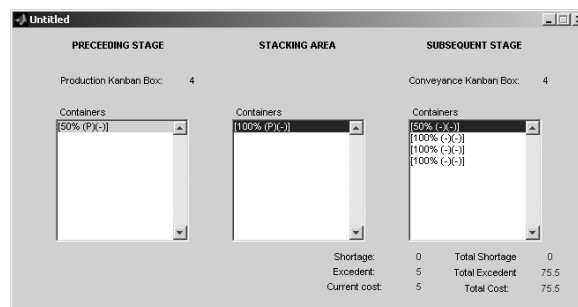


Fig. 5 Visualización de costes. También se visualiza la situación actual de los contenedores y *Kanbans* dentro del sistema.

### IV. EVALUACIÓN EXPERIMENTAL

Como la finalidad de todo sistema de producción es la reducción de costes, los experimentos que se llevarán a cabo se basarán en los conceptos de coste de sobreproducción y coste de falta de material.

Se calcularán el coste de sobreproducción y de falta de material para cada paso de simulación. Además, se calcularán el coste de sobreproducción total y el coste de falta de material total que se corresponderán con los costes que se han producido durante un cierto periodo de tiempo (en este caso un año).

#### A. Experimento I: Niveles de demanda estables

Observaremos la evolución de los costes de sobreproducción y falta de material según distintos niveles de demanda y diferente número de *Kanbans* de transporte. Para ello supondremos dos cosas:

- La tasa de demanda será constante durante un periodo de simulación (en nuestro, caso un año). En definitiva, se simularán varios años con distintas tasas de demanda.
- La tasa de producción se mantendrá constante durante toda la simulación.

En nuestro caso hemos optado por una tasa de producción de 0.5 y tasas de demanda entre 0.1 y 1 a pasos de 0.1 (un total de 10 años diferentes) y variando el número de *Kanbans* entre 1 y 10.

### B. Experimento II: Niveles de demanda variables

En este segundo experimento se procederá de la misma manera que en el apartado anterior con la única diferencia de no mantener constante la tasa de demanda dentro de un mismo año. La tasa de demanda variará de manera aleatoria según una distribución Normal (con media  $\mu=0.5$  y desviación típica  $\sigma=0.2$ ).

## V. RESULTADOS

### A. Resultados experimento I

Los resultados obtenidos se muestran en las figuras 6 y 7. Se observa que los costes de falta de material se mantienen relativamente constantes respecto al número de *Kanbans* (los resultados obtenidos con un único *Kanban* se han omitido ya que no fueron satisfactorios, produciéndose errores debido a la secuencialidad de la simulación). Siendo la tasa de producción de 0.5, toda tasa de demanda inferior o igual a ésta produce un coste de falta de material nulo. A partir de tasas superiores al 0.5, este coste aumenta de manera constante.

El coste de sobreproducción es relativamente bajo a tasas de demanda mayores a 0.5 y se ve mínimamente afectado por el número de *Kanbans*. Sin embargo, a partir de una tasa inferior a 0.5, el coste aumenta considerablemente y especialmente para un número de *Kanbans* elevado. Teniendo pocos *Kanbans* conlleva a tener pocos contenedores, por lo cual existe menor capacidad de almacenaje. Estando los contenedores llenos, la producción se detiene dejando de generarse coste de sobreproducción. En cambio, cuantos más contenedores haya, más tiempo pasará hasta detener la producción y más coste de sobreproducción se generará.

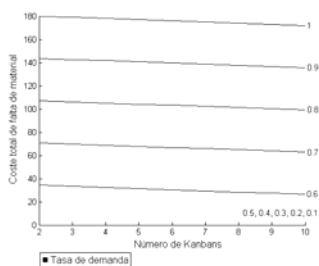


Fig. 6 Costes de falta de material.

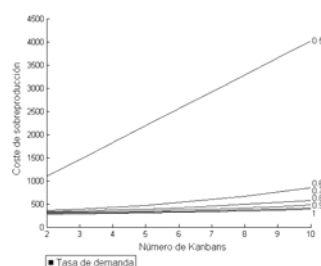


Fig. 7 Costes de sobreproducción.

### B. Resultados experimento II

Trabajando con niveles de demanda variables, los resultados que se obtienen (figuras 8 y 9) son también bastante variables cosa que implica una interpretación algo complicada (como ya mencionado en el apartado de inconvenientes, el sistema JIT no es del todo adecuado en entornos muy cambiantes). De todas maneras se observa que a nivel que aumenta el número de *Kanbans*, el coste de falta de material disminuye y el coste de sobreproducción aumenta.

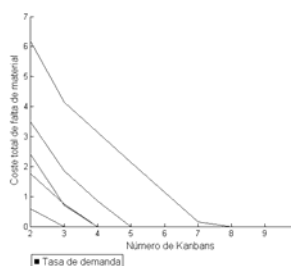


Fig. 8 Costes de falta de material.

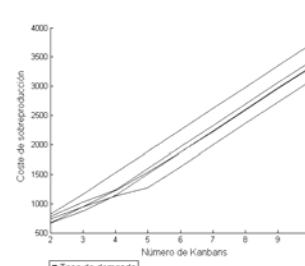


Fig. 9 Costes de sobreproducción.

## VI. CONCLUSIONES

A partir de los resultados obtenidos en los dos apartados anteriores se puede observar que el sistema JIT es bastante más efectivo para bajas variaciones de demanda. En nuestro caso concreto, el número ideal de *Kanbans* solía coincidir con un término intermedio, es decir, alrededor de cinco ya que en ese caso se obtenía un coste combinado mínimo.

Aunque esta haya sido únicamente una simulación muy simple del flujo básico del TPS, si fuera de interés se podría utilizar como base para una posterior extrapolación al sistema real.

Asignatura: Sistemas Integrados de Fabricación.

Impartida por : Antoni Burguera Burguera.



Christian Peter Winter (christian@kpwinter.com)

es estudiante de tercer curso de Ingeniería Técnica Industrial (especialidad en electrónica industrial) en la UIB. El trabajo aquí presentado se corresponde con el informe de una práctica de la asignatura de Sistemas Integrados de Fabricación.



Francisco Muñoz Contreras (feco\_m@msn.com) es estudiante

de tercer curso de Ingeniería Técnica Industrial (especialidad en electrónica industrial) de la UIB. El trabajo aquí presentado se corresponde con el informe de una práctica de la asignatura de Sistemas Integrados de Fabricación.