Revista de la Construcción
ISSN: 0717-7925
revistadelaconstruccion@uc.cl
Pontificia Universidad Católica de Chile
Chile

PONZ TIENDA, J. L.; BENLLOCH MARCO, J.; ANDRÉS ROMANO, C.; GIL SENABRE, DORIA

Un algoritmo matricial RUPSP / GRUPSP "sin interrupción" para la planificación de la producción bajo metodología Lean Construction basado en procesos productivos

Revista de la Construcción, vol. 10, núm. 2, abril, 2011, pp. 90-103
Pontificia Universidad Católica de Chile
Santiago, Chile

A matrix algorithm RUPSP / GRUPSP "no splitting allowed" for production planning under Lean Construction methodology based on production processes

Un algoritmo matricial RUPSP / GRUPSP "sin interrupción" para la planificación de la producción bajo metodología Lean Construction basado en procesos productivos

Autores

Ph. PONZ TIENDA, J. L.
Dep. of Archit. Constructions
Polytechnic University of Valencia, Valencia (Spain)
jopontie@csa.upv.es

Ph. BENLLOCH MARCO, J.
Dep. of Archit. Constructions
Polytechnic University of Valencia, Valencia (Spain)
jabenllo@csa.upv.es

Ph. ANDRÉS ROMANO, C.
Dep. of Management
Polytechnic University of Valencia, Valencia (Spain)
candres@omp.upv.es

M.Sc. DORIA GIL SENABRE
Dep. of Archit. Constructions
Polytechnic University of Valencia, Valencia (Spain)
dogilse@edificacion.upv.es

Fecha de recepción 22/1/2011

Fecha de aceptación 25/7/2011

Resumen

La planificación y programación de la producción en el ámbito de la construcción es, quizás, la gran asignatura pendiente del *construccction scheduling*, y uno de los grandes objetivos de la metodología *Lean Construction* o *Construcción sin pérdidas*.
En este artículo se pretende esclarecer las limitaciones de los algoritmos de cálculo utilizados, y ofrecer un nuevo algoritmo para la programación de proyectos en entornos *Lean Constucction* mediante grafos *PDM* (*Precedence Diagramming Method*) sin interrupción (no *splitting allowed)* y *precedencias generalizadas* (*GPR´s*) basadas en procesos constructivos, que sirvan para la correcta aplicación de modelos de optimización y control de la producción.

Palabras clave: Programación de la construcción, programación de proyectos, PDM, RUPSP, GRUPSP, RCPSP, GRCPSP, planificación de producción.

Abstract

Production Planning and Scheduling in the field of construction may be the great pending construction scheduling subject, and one of the major objectives of the Lean Construction methodology.
In this article, is intended to clarify the limitations of the calculation algorithms used, and offer a new algorithm for programming projects in Lean Construction environments through graphs PDM (Diagramming Method Precedence) without interruption (not splitting allowed) and generalized precedence relations (GPR´s) based on constructive processes for the correct application of scheduling, optimization and production control models.

Keys words: *Lean Construction, Construction scheduling, project scheduling, PDM, RUPSP, GRUPSP, RCPSP, GRCPSP, Production Planning, no splitting allowed.*

1. Introducción

Las nuevas tendencias en la gestión de proyectos y, más concretamente, en el ámbito del *Construction scheduling*, evolucionan de forma prácticamente imparable hacia las metodologías *Lean Construction*, gracias a las iniciativas del *Lean Construction Institute* (Lean Construction Institute, 2010), y a los trabajos de Alarcón (Alarcón, 1997), Koskela (Koskela, 2000) (Koskela, 1992), Ballard (Ballard & Howell, 1994) (Ballard G., 1999) (Ballard G., 2000) (Ballard & Howell, 2003), Howell (Ballard & Howell, 1994) (Ballard & Howell, 2003), Pellicer (Alarcón & Pellicer, 2009) y Formoso (Tzortzopoulos & Formoso, 1999) (Peixoto & Formoso, 1998). Esta nueva metodología está basada en la filosofía Kay-Zen y sistema de producción Toyota o *Producción sin pérdidas*, teniendo como una de sus más importantes herramientas la aplicación del *LPDS* (*Lean Proyect Delivery System*) (Ballard & Howell, 2003) y del *LPS* (*Last Planner System of Production Control*) (Ballard G., 2000) (Ballard & Howell, 2003) por la contrastada eficiencia y versatilidad como nuevo paradigma de la gestión de proyectos de construcción.

El sistema *LPS* (*Last Planner System of Production Control*) establece diferentes fases para la secuenciación de los proyectos de construcción. La primera fase correspondería al ***Master and Phase scheduling***, en donde se establecen en líneas generales los grandes objetivos de la programación del proyecto, afrontado este en su globalidad y con tareas definidas en los primeros y más altos niveles del *WS* (*Work Structuring*) (Ballard & Howell, 2003), una segunda fase corresponde al ***Lookahead schedule***, en donde las tareas pertenecientes ya a niveles inferiores del *WS*, son secuenciadas normalmente en un plazo de dos a tres meses vista y con un mayor nivel de definición; la última fase corresponde al ***weekly work plans***, en el cual las tareas pertenecientes ya al último y más bajo nivel del *WS* son secuenciadas con un altísimo nivel de detalle (ver Figura 1).

Para definir el *WS* (*Work Structuring*), se ha de establecer *Activity Definition Model* (*ADM*), en donde se definen las responsabilidades, prerrequisitos, restricciones, directivas, duración programada y los recursos asignados de aquellos disponibles (ver Figura 2).

Figura 1. *LPS. Ballard & Howell, 2003.*

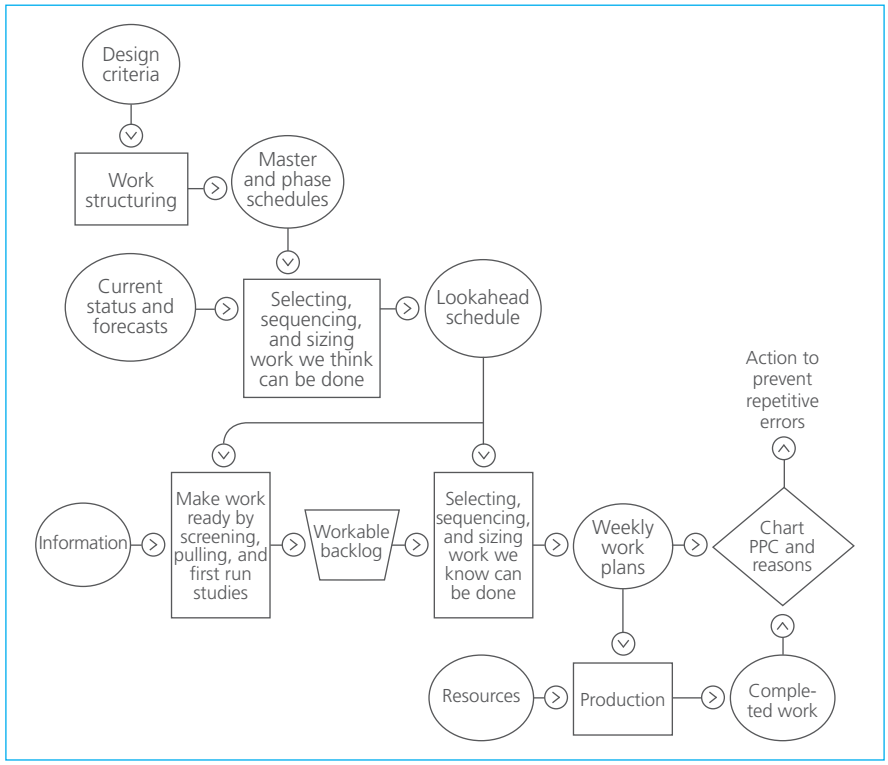
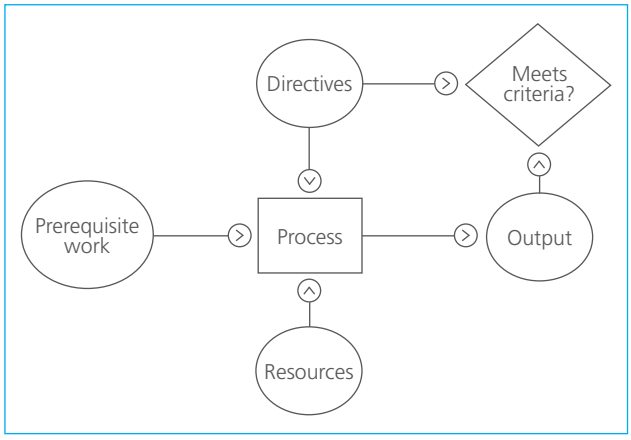


Figura 2. *ADM. Ballard & Howell, 2003*
Lean project management



La aplicación de la metodología Lean Construction debe ser afrontada desde dos enfoques diferenciados pero inseparables, un enfoque cualitativo que establece los objetivos y principios metodológicos a partir de las debilidades detectadas en los modelos tradicionales, y otro enfoque cuantitativo, que deberá implementar las anteriores directrices metodológicas, mediante la aplicación de los modelos matemáticos del *Project scheduling* y de la programación óptima de la producción, ofrecidos por la Investigación Operativa, y que tan excelentes resultados están dando en el mundo de la secuenciación de procesos industriales *Lean Manufacturing*.

La puesta en práctica de las herramientas cuantitativas necesarias para poder acometer con rigor la secuenciación óptima de un proyecto de construcción o edificación, requiere de la algorítmica y de los programas informáticos que la implementen, que contemplen las particularidades del sistema productivo de la construcción. ¿Pero están los programas comerciales de gestión de proyectos preparados para ofrecer soluciones no ya eficientes, sino solamente eficaces o aceptables? Parece que los importantísimos avances de las últimas décadas en *scheduling* y planificación óptima de la producción han permanecido ajenos a los programas comerciales, y aún más a las especiales particularidades que tienen los procesos constructivos, el *Lean Construccction* y el *LPS*.

En el presente artículo se va a analizar el estado del arte de la algorítmica para la secuenciación de proyectos sin restricciones de recursos *RUPSP* (*Resource Unconstrained Project Scheduling Problem*) y G*RUPSP* (*Generalized Resource Unconstrained Project Scheduling Problem*) con grafos *PDM* (*Precedence Diagramming Method*), para analizar su adecuación a cada una de las fases del *LPS* y proponer un nuevo algoritmo que permita la aplicación de modelos cuantitativos avanzados para la secuenciación de proyectos de construcción mediante el sistema *LPS*.

2. Estado del arte de la programación de proyectos con grafos *PDM*

Un grafo de proyecto *G*es un par denotado como *G*(*V,A*), donde *V*(*v1, v2, …, vN*) es el conjunto finito no vacío de vértices y *A*(*a*11, *a*12, …, *a*1*N*, *a*21, *a*22, …, *a*2*N*, …, *aNN*) es el conjunto de pares (*i, j*) de *V* correspondiente a los arcos del grafo. Los grafos de proyectos se diferencian básicamente del resto de grafos, en que son dirigidos y que no existen bucles ni circuitos, es decir, que los arcos tienen una dirección, no puede entrar y salir de un mismo nodo, teniendo con un comienzo y final de grafo perfectamente diferenciados.

Existen dos sistemas fundamentales para representar los grafos de proyectos, uno que asemeja las tareas a los arcos del grafo, correspondiendo los nodos a los hitos o instantes en que las tareas comienzan o finalizan, y conocida como *AoA* (*Activity on Arrow*) y otra en la que las tareas son asimiladas a los nodos, siendo las aristas las relaciones de dependencia existente entre ellas y conocida como *AoN* (*Activity on Node*).

Los primeros trabajos que contemplaron la representación de grafos con las tareas en los nodos se los debemos al ingeniero francés B. Roy (Roy, Graphes et ordonnancements, 1962) (Roy, 1959), que los denominó como *Método de los Potenciales*. En los Estados Unidos se siguieron líneas de trabajo similares, aunque de forma independiente. Estos trabajos fueron iniciados con el *Sistema de Actividades en los Nodos* por John Fondahl (Fondahl, 1961), considerado como el precursor del moderno *PDM* o *Precedence Diagramming Method*, aunque la primera descripción detallada con múltiples tipos de precedencias apareció en 1964 con el manual de uso del *Project Control System* del IBM 1440 (IBM, 1964), con J. Craig como investigador principal, aunque la autoría no está muy clara pues IBM cita ya en 1963 a HB Zahry Company de San Antonio como creador del formato de Precedencias.

El *Sistema de Actividades en los Nodos* tan solo consideraba precedencias del tipo *final-comienzo* (*FCz ij*), que significa que la tarea *i* ha de estar totalmente finalizada con anterioridad al comienzo de la tarea *j*, debiendo existir al menos un desfase de *z* lapsos entre los dos sucesos, pudiendo ser cero o negativo. La re-

presentación de las tareas en los nodos de los grafos se realizaba por medio de cajetines, que contenían la información relativa a los tiempos más tempranos y más tardíos de comienzo y terminación de cada una de ellas, desarrollando un algoritmo de cálculo basado en el principio de que si conocemos el instante de terminación más desfavorable del conjunto *i* de tareas predecesoras de *j*, estaremos en condiciones de establecer su instante de comienzo más temprano (*esj*):

comienzoJ= máximo finali
esj= máx(*efi*); ∀ *j, i precedente de j*

De forma análoga, pero en sentido inverso, se calculan los instantes de comienzo y terminación más tardía, estableciendo así los cuatro tiempos característicos de una tarea, y que son:

- *Tiempo más pronto de empezar* (*esio early start*): representa el instante de tiempo en el que como muy pronto puede empezar una tarea, cumpliendo las relaciones de precedencias impuestas en el proyecto.
- *Tiempo* más pronto de terminar (*efi o early fini h*): representa el instante de tiempo en el que como muy pronto puede finalizar una tarea, cumpliendo las relaciones de precedencias impuestas en el proyecto.
- *Tiempo más tarde de empezar* (*lsi o latest start*): representa el instante de tiempo en el que como muy tarde puede comenzar una tarea, cumpliendo las relaciones de precedencias impuestas en el proyecto, más tarde del cual se retrasaría la fecha de finalización del proyecto.
- *Tiempo más tarde de terminar* (*lfi o latest finish*): representa el instante de tiempo en el que como muy tarde puede finalizar una tarea, cumpliendo las relaciones de precedencias impuestas en el proyecto, más tarde del cual se retrasaría la fecha de finalización del proyecto.

Los anteriores tiempos o instantes serán representados dentro del Nodo tal y como se muestra en la Figura 3.

Figura 3. *Tiempos* ***esi***, ***efi***, ***lsi*** *e* ***lfi*** *del Nodo* ***i*** *correspondiente a la tarea* ***i*** *con duracion* ***di***

Donde *di* es la duración de la tarea *i*, que ha de ser ejecutada de forma continua y con intensidad constante, quedando el algoritmo de cálculo de los tiempos de las tareas de un grafo de proyecto y su duración mínima (*makesp an*) de la siguiente forma:

Forward calculation
Paso 1.
esstart=0
efstart= esstart + dstart
For (*i*=1; *end*; *i*++)
esi=–∞;
Paso 2.
For (*j*=1, *end*, *j*++)
esj= δj + máx{*esj*; *efi + zij*} ∀ {*ij*} ∈ *A*
efj= esj + d j;
Backward calculation
Paso 1.
lfend= makespan
lsend= lfend – dend
For (*i*= 1; *end*; *i*++)
lfi= ∞;
Paso 2.
For (*i*= *end*, 1, *i*– –)
lfi= min lfi; *lsj – zij* ∀ {*ij*} ∈ *B*
lsi= lfi – di;
Siendo: *A el conjunto de tareas precedentes de j.*
B el conjunto de tareas sucesoras de i.
δj el desplazamiento con respecto a esj.

Consideramos que una tarea es crítica cuando la diferencia entre el tiempo más tarde de terminar (*lfi*) y el tiempo más pronto de empezar (*esi*) es su duración (*di*), llamando *holgura total* (*hti*) a la cantidad de tiempo que podemos retrasar el comienzo de una actividad sin afectar a la duración del proyecto. La *holgura total* (*hti*) se calculará como la diferencia entre el tiempo más tarde de terminar (*lfi*) y el tiempo más pronto de empezar (*esi*) menos su duración (*di*). La *holgura libre* (*hli*) de una tarea, es la cantidad de tiempo que podemos retrasar el comienzo de una actividad sin afectar al tiempo más pronto de comienzo de las actividades sucesoras, y se calculará como *Hli= minesj–zij–efi*} ∀ {*ij*} ∈ *B*

3. La representación matricial de los grafos de proyecto

Una característica muy importante y poco conocida de los grafos es la posibilidad de representarse mediante matrices. Esta representación está basada en una indexación bidimensional de los nodos del grafo, de tal manera que los arcos entre los nodos se pueden ver como relaciones entre los índices, con un peso que será la duración del arco (desfase *z*). Es decir, una matriz *A*,

donde los elementos o entradas de la matriz *aji* son el desfase *z* si existe conexión entre los nodos *i* y *j* y *Null* en el caso contrario, siendo *i* la tarea sucesora de *j*. Como los grafos de proyecto son grafos dirigidos, por cada entrada *aij≠ Null* se encontrará una entrada *aji= Null*, y al carecer de bucles, en todas las entradas *ajj* encontraremos un valor *Null*. Para evitar los circuitos será condición suficiente que las entradas *aij* en las que *i>j* sean *Null* (ver Figura 4).

Figura 4. *Indexación de un grafo*

	1	2	*i*	*n*
1				
2	a_{21}			
j	a_{j1}	a_{j2}	a_{ji}	
n	*an1*	a_{n2}	a_{ni}	

Los primeros trabajos para el cálculo de los tiempos de las tareas a partir de su representación matricial se los debemos a Zaderenko (Zaderenko, 1968), que en su algoritmo original tan solo ofrecía valores para los tiempos más pronto y más tarde de empezar, además de requerir de una tarea ficticia de comienzo y otra de final (Ponz Tienda, 2009). El algoritmo original de Zaderenko puede ser fácilmente mejorado, eliminando la necesidad de incluir las dos tareas ficticias e incluyendo, además del cálculo de los tiempos, los valores de desplazamiento (δ) para las tareas tal y como se muestra en la Figura 5.

Figura 5. *Indexación de un grafo de proyecto*

	Dur	δ	*Es*	*Ef*	*Ls*	*Lf*	*1*	*2*	*i*	*n*
1	d_1	δ_1	es_1	ef_1	ls_1	lf_1				
2	d_2	δ_2	es_2	ef_2	ls_2	lf_2	a_{21}			
j	d_j	δ_j	es_j	es_j	ls_j	lf_j	a_{j1}	a_{j2}	a_{ji}	
n	d_n	δ_n	es_n	es_n	ls_n	lf_n	a_{n1}	a_{n2}	a_{ni}	

A partir de la indexación del grafo de proyecto, podemos establecer el siguiente algoritmo en pseudocódigo:

Forward calculation
For (*j*=1, *n*, *j*++)
  For (*i*=1, *j*–1, *i*++)
    esj= δj + maxesj, efi + aji; ∀ *aij <> Null*
  afj= esj + dj
  makespan= max makespan, efj;
Backward calculation
For (*i*=1, *n*, *i*++)
  lfi= makespan
  For (*j=i*+1, *n*, *j*++)
    lfi= minlfi, lsj – aji; ∀ *aji <> Null*
  lsi= lfi – di

El anterior algoritmo puede ser implementado en una macro para ser ejecutado en una hoja de cálculo, estableciendo las duraciones y los desplazamientos como variables de decisión de la formulación de las celdas, quedando de esta forma un modelo totalmente dinámico, que en *VBA* para Excel© de Microsoft© puede ser descargada junto con la generadora de la matriz y las instrucciones de uso en la url: *http://personales.upv/jopontie/rupsp_nsa_sp.zip*

Esta formulación es muy práctica y adecuada para secuenciaciones destinadas a estandarizar y optimizar procesos constructivos mediante ***daily work plans***, para la elaboración de ***weekly work plans***, y muy excepcionalmente para ***lookahead Schedules***, no debiendo ser utilizada en ***Master and Phase scheduling*** al requerir del uso de otro tipo de dependencias llamadas generalizadas (*GPR´s*) para contemplar la simultaneidad en la ejecución de las tareas.

4. Las precedencias generalizadas y la criticidad inversa

En la formulación del *RUPSP* expuesta hasta ahora, tan solo se han contemplado dependencias del tipo *final-comienzo* (*FCzij*), pero existe otro tipo de dependencias llamadas generalizadas o *GPR* (*Generalized Precedence Relations*), que permiten condicionar y simultanear la ejecución de las tareas entre sí. Estas dependencias generalizadas son:

- *Comienzo-comienzo* (*CCzij*): significa que la tarea *i* debe de estar iniciada, para que comience la tarea *j*, debiendo existir al menos un desfase de *z* lapsos entre los dos sucesos, pudiendo ser cero o negativo.
- *Final-final* (*FFzij*): significa que la tarea *i* debe de estar finalizada, para que finalice la tarea *j*, debiendo existir al menos un desfase de *z* lapsos entre los dos sucesos, pudiendo ser cero o negativo.

- *Comienzo-final* (*CFzij*): significa que la tarea *i* debe de estar iniciada, para que finalice la tarea *j*, debiendo existir al menos un desfase de *z* lapsos entre los dos sucesos, pudiendo ser cero o negativo.

Para contemplar las anteriores dependencias, tan solo tendremos que sustituir en el *Forward Calculation* la expresión:

$esj = \delta j + máx\{esj;\ efi + zij\}\ \forall\ \{ij\} \in A$

Por:

$$\begin{array}{lll} esj = \delta j + máx\{esj;\ efi + zij\} & \forall\ \{FCzij\} \in A & \\ \quad esi + zij; & \forall\ \{CCzij\} \in A & \\ \quad efi + zij - dj; & \forall\ \{FFzij\} \in A & [1] \\ \quad esi + zij - dj\} & \forall\ \{CFzij\} \in A & [2] \end{array}$$

Y sustituir en el *Backward Calculation* la expresión:

$lfi = min\ lfi;\ lsj - zij \qquad \forall\ \{ij\} \in B$

Por:

$$\begin{array}{lll} lfi = min\ \{lfi;\ lsj - zij; & \forall\ \{FCzij\} \in B & \\ \quad lsj - zij + di; & \forall\ \{CCzij\} \in B & [3] \\ \quad lfj - zij; & \forall\ \{FFzij\} \in B & \\ \quad lfj - zij + di\} & \forall\ \{CFzij\} \in B & [4] \end{array}$$

Donde por medio de [1], [2], [3] y [4] garantizamos la continuidad, la no interrupción, en la ejecución de las tareas.

Estas dependencias generalizadas pueden ser contempladas en la anterior formulación matricial transformándolas en dependencias de tipo *final-comienzo* (*FCzij*) (Bartusch, Möhring, & Radermacher, 1988), y que se muestran en la Figura 6.

Figura 6. *Transformación de dependencias generalizadas a dependencias* ***FCzij***

Una dependencia del tipo:	como final-comienzo (FCz_{ij}) será:
Comienzo-comienzo (CCz_{ij})	$FC\ (z\text{-}d_i)ij$
Final-final (FFz_{ij})	$FC\ (z\text{-}d_j)ij$
Comienzo-final (CFz_{ij})	$FC\ (z\text{-}d_i\text{-}d_j)ij$

Las anteriores transformaciones darán como resultado valores de *z* negativos para las dependencias resultantes, algo que no supone ningún problema desde el punto de vista de la aplicación del algoritmo, ofreciendo exactamente los mismos resultados en ambos modelados. El problema surge porque al imponer la relajación de no interrupción, se puede provocar resultados paradójicos e incorrectos, a causa de la criticidad inversa, fenómeno poco estudiado pero de enorme trascendencia. Una tarea es *crítica inversa* cuando un incremento (decremento) en la duración de esta produce un desplazamiento (adelanto) en sus tiempos de comienzo, afectando de manera contraria a la duración del proyecto.

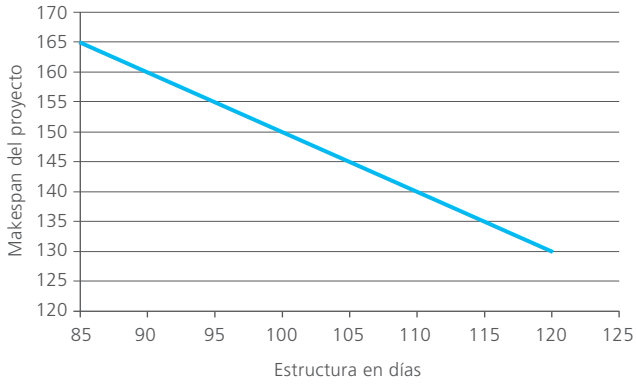
La existencia de tareas críticas inversas produce efectos anómalos en el del proyecto del *RUPSP* con *GPR´s o GRUPSP*, y fue estudiado por primera vez en 1981 por Jerome D. Wiest (Wiest, 1981), llegándolo a denominar como *efecto perverso*, escribiendo: "*Las tareas críticas inversas son un concepto que entierra el sentimiento natural de las consecuencias de alargar o retrasar una tarea*".

Supongamos la instancia presentada en la Figura 7, de un ***Master scheduling***, formado por tres tareas: Cimentación, Estructura y Resto de obra, de 20, 100 y 80 días de duración respectivamente.

Figura 7. *Instancia con dependencias generalizadas*

El obtenido es de 150 días, pero si cambiamos la duración de la estructura a 95 días, obtenemos un nuevo de 155 días, y si la cambiamos a 105 días el resultante será de 145 días (ver Figura 8). Es decir, se produce un efecto en el del proyecto contrario a la dirección del cambio en la duración de la tarea crítica inversa, y puede ser fácilmente comprobado introduciendo la anterior instancia en Microsoft Project©.

Figura 8. *Evolución del vs de estructura con dependencias simples*



Actualmente se sigue considerando inherente a los problemas con GPR´s (Herroelen, 1999) (Valls & Lino, 2001), aunque puede reducirse su efecto permitiendo múltiples dependencias simultáneas entre dos tareas, acotando inferior o superiormente el efecto de la criticidad inversa, tal y como hace Primavera Project Planner©, quedando el *Paso 2 del Forward calculation* de la siguiente forma:

$$
\begin{aligned}
&For\ (j{=}1,\ end,\ j{+}{+})\\
&\quad esj = \delta j + máx\{esj;\ efi + zij\ k; && \forall\ \{FCzij\ k\} \in A\\
&\qquad esi + zij\ k; && \forall\ \{CCzij\ k\} \in A\\
&\qquad efi + zij\ k - dj; && \forall\ \{FFzij\ k\} \in A \quad [1]\\
&\qquad esi + zij\ k - dj\} && \forall\ \{CFzij\ k\} \in A \quad [2]\\
&\quad efj = esj + dj;
\end{aligned}
$$

Y el del Backward calculation:

$$
\begin{aligned}
&For\ (i{=}\ end,\ 1,\ i{-}{-})\\
&\quad lfi = min\ lfi;\ lsj - zij\ k; && \forall\ \{FCzij\ k\} \in B\\
&\qquad lsj - zij\ k + di; && \forall\ \{CCzij\ k\} \in B \quad [3]\\
&\qquad lfj - zij\ k; && \forall\ \{FFzij\ k\} \in B\\
&\qquad lfj - zij\ k + di\} && \forall\ \{CFzij\ k\} \in B \quad [4]\\
&lsi = lfi - di;
\end{aligned}
$$

Donde por medio [1], [2], [3] y [4] seguimos garantizando la continuidad, la no interrupción, en la ejecución de las tareas.

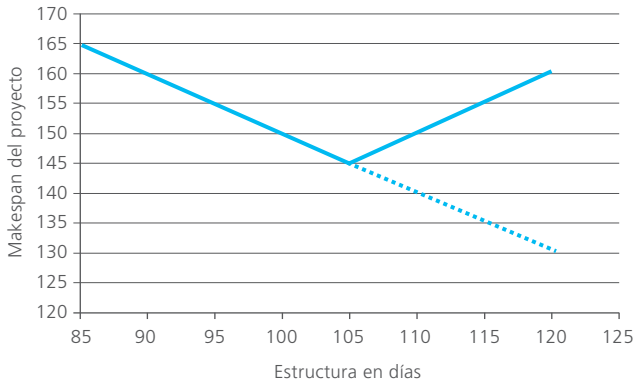
Quedando la anterior instancia de la siguiente forma:

Figura 9. *Instancia con dependencias generalizadas simultáneas.*

El obtenido es de 150 días igual al obtenido anteriormente, que en este caso con *GPR´s* múltiples soluciona parcialmente la criticidad inversa, aunque para valores de estructura inferiores a 105 días seguiríamos observando su efecto, desapareciendo para el resto de valores (ver Figura 10).

La algorítmica actual y la criticidad inversa imposibilitan la utilización eficaz de las GPR´s, y consecuentemente condicionan la elaboración de los ***lookahead Schedules*** y del ***Master scheduling*** exclusivamente con relaciones del tipo *final-comienzo* (*FCzij*), incrementando enormemente el número de tareas y la operatividad de la secuenciación. Además, la criticidad inversa implica la imposibilidad de aplicar rigurosamente modelos de optimización de la producción con *GPR´s* al obtener como resultado soluciones incorrectas, especialmente en aquellos modelados como los *MRCPSP* (*Multi Mode Resource Constrained Project Scheduling Problem*) o *TCTP* (*Time Cost Trade Off Problem*), que consideran diferentes modos de ejecución para las tareas, además de actuar sobre el traslado de las tareas (δ) como los *G/RCPSP* (Generalized/*Resource Constrained Project Scheduling Problem*).

Figura 10. *Evolución del vs de estructura con dependencias generalizadas simultáneas*



5. La solución de la criticidad inversa

Jerome D. Wiest (Wiest, 1981) apuntó que la solución parcial al problema de la criticidad inversa consistía en la aplicación de algoritmos con fragmentación *spliting allowed*, y concretamente el algoritmo de Crandall (Crandall, 1973), posteriormente fue corregido por Valls y Martí (Valls, Martí, & Lino, 1996) y modificado por Ponz (Ponz Tienda, 2010).

La fragmentación óptima de las tareas, aunque no soluciona por sí misma la criticidad inversa, ha de ser considerada seriamente, debiendo ser potestativo del secuenciador del proyecto la fragmentación o no de las tareas, y no resultado de la relajación del problema, pues está demostrado que la fragmentación de las tareas ofrece mejores soluciones y valores para el proyecto que los algoritmos sin fragmentación (Valls, Martí, & Lino, 1996) (Ponz Tienda, 2010).

Para solucionar definitivamente la criticidad inversa se ha de volver al origen y estudiar nuevamente la verdadera naturaleza del problema, que no es más que la esencia del *Lean thinking*: la secuenciación de procesos productivos y, como tal, pensar en términos de producción.

6. Un algoritmo matricial *Lean thinking* con precedencias generalizadas

Cuando se establecen las duraciones y el tipo de dependencia entre las tareas de un determinado proceso constructivo, se analizan las directivas generales, los prerrequisitos necesarios para su ejecución, y los recursos disponibles para cada un de las tareas (ver Figura 2). Así, de esta forma, la duración normalmente vendrá determinada por las directrices y el rendimiento de los recursos disponibles, y las dependencias entre las tareas por los prerrequisitos. Estos prerrequisitos son posteriormente representados en la secuencia de los trabajos como dependencias del tipo *final-comienzo* (*FCzij*), *comienzo-comienzo* (*CCzij*), *final-final* (*FFzij*) o *comienzo-final* (*CFzij*), siendo el lapso mínimo entre los sucesos de comienzo y finalización de las tareas.

El problema surge porque tradicionalmente se ha considerado a los prerrequisitos como lapsos *z*, y de forma inercial así se han seguido considerando, cuando realmente y especialmente en los procesos constructivos, su verdadera naturaleza suele ser la de determinados niveles de producción necesarios para comenzar o finalizar una determinada tarea o que no podrán ser ejecutados mientras no finalice su precedente.

Así, de esta forma, cualquier modificación en la disponibilidad o rendimiento de los recursos asignados a una tarea, llevará aparejado una modificación en su duración y en la producción realizada por unidad de tiempo, y consecuentemente al modificarse la producción por unidad de tiempo, se deberá modificar el efecto que produce sobre las dependencias, algo que no se contempla en la algorítmica.

Así pues, a las dependencias tradicionales las denominaremos *desfases* y estableceremos tres nuevos tipos de dependencias que llamaremos *relaciones de producción* entre las tareas, y que son:

- *Relación de producción de comienzo-comienzo* (*RCCpij k*): La *relación de producción de comienzo-comienzo* entre *i* y *j* (*RCCpij k*) significa que la tarea *i* debe de estar iniciada y haber transcurrido al menos una *p* centésima parte de producción, para que comience la tarea *j*, siendo necesariamente mayor que cero y menor que la unidad (*0<RCCpij k*<1).
- *Relación de producción de final-final* (*RFFpij k*): La *relación de producción de final-final* entre *i* y *j* (*RFFpi k*) significa que la tarea *i* debe de estar finalizada, y deberá quedar pendiente al menos una *p* centésima parte de producción de la tarea *j* sin ejecutar para cuando *i* finalice, siendo necesariamente mayor que cero y menor que la unidad (*0<RFFpij k*<1).
- *Relación de producción de comienzo-final* (*RCFpij k*): La *relación de producción de comienzo-final* entre *i* y *j* (*RCFpij k*) significa que la tarea *i* debe de estar iniciada y haber transcurrido al menos una *p* centésima parte de producción, para que finalice la tarea *j*, siendo necesariamente mayor que cero y menor que la unidad (*0<RCFpij k*<1).

Las *relaciones de producción* y los *desfases* no son excluyentes entre si, sino que habitualmente están íntimamente unidas, como por ejemplo cuando los prerrequisitos para comenzar una tarea de albañilería no solo es el de haber finalizado una determinada planta de la estructura expresado como una *p* centésima una parte de su producción total, sino que además ha de transcurrir un determinado lapso (*z*) para su fraguado y desapuntalado, surgiendo las *dependencias mixtas* o *relaciones de producción con desfase*:

- *Relación con Desfase de comienzo-comienzo* (*RDCCp(z)ij k*): La *relación con desfase de comienzo-comienzo* entre *i* y *j* (*RDCCp(z)ij k*) significa que la tarea *i* debe de estar iniciada y haber transcurrido al menos una *p* centésima parte de producción con un desfase adicional de *z* lapsos para que comience la tarea *j*, siendo necesariamente mayor que cero y menor que la unidad la *relación*, pudiendo ser cero o negativo el *desfase*.
- *Relación con Desfase de final-final* (*RDFFp(z)ij k*): La *relación con desfase de final-final* entre *i* y *j* (*RDCCp(z)ij k*) significa que la tarea *i* debe de estar finalizada y deberá transcurrir al menos una *p* centésima parte de producción, con un desfase adicional de *z* lapsos para que finalice la tarea *j*, siendo necesariamente mayor que cero y menor que la unidad la *relación*, pudiendo ser cero o negativo el *desfase*.
- *Relación con Desfase de comienzo-final* (*RDCFp(z)ij k*): La *relación con desfase de comienzo-final* entre *i* y *j* (*RDCFp(z)ij k*) significa que la tarea *i* debe de estar iniciada y haber transcurrido al menos una *p* centésima parte de producción con un desfase adicional de *z* lapsos para que finalice la tarea *j*, siendo necesariamente mayor que cero y menor que la unidad la *relación*, pudiendo ser cero o negativo el *desfase*.

La indexación del grafo de proyecto puede ser adaptada para contemplar las anteriores precedencias generalizadas (*GPR´s*), representando cada tarea *j* por la fila (*2j–1*), y sus precedentes por la columna (*2i–1*), de tal manera que la primera fila o columna relativa de la tarea corresponderá a su comienzo, y la segunda a su final, y cada uno de los índices de la matriz corresponderá a un tipo de precedencia en función de su posición relativa (ver Figura 11).

A partir del hecho de que las *relaciones de producción* son expresadas en tantos por uno, y los *desfases* en valores enteros, no es imprescindible diferenciarlas en los índices, y la representación matricial del grafo de proyecto adaptada a las relaciones generalizadas basadas en producción y desfases quedará tal y como se aprecia en la Figura 12.

Figura 11. *Indexación relativa de las dependencias GPR´s de i con j*

		i	
		$2_i - 1$	2_i
j	$2_j - 1$	$CC_{ij} = a_{fila\ (2j\ -\ 1)\ columna\ (2i-1)}$	$FC_{ij} = a_{fila\ (2j\ -\ 1)\ columna\ (2i-1)}$
	2_j	$CF_{ij} = a_{fila\ (2j)\ columna\ (2i-1)}$	$FF_{ij} = a_{fila\ (2j)\ columna\ (2i)}$

Figura 12. *Indexación de un grafo de proyecto con GPR´s*

						1		2		i		n	
	Dur	δ	Early	Last		C	F	C	F	C	F	C	F
1	d_1	δ_1	es_1	ls_1	C								
			ef_1	lf_1	F								
2	d_2	δ_2	es_2	ls_2	C	cc_{21}	fc_{21}						
			ef_2	lf_2	F	cf_{21}	ff_{21}						
j	d_j	δ_j	es_j	lsj	C	cc_{j1}	fc_{j1}	cc_{j2}	fc_{j2}	cc_{ji}	fc_{ji}		
			ef_j	lf_j	F	cf_{j1}	ff_{j1}	cf_{j2}	ff_{j2}	cf_{ji}	ff_{ji}		
n	d_n	δ_n	es_n	ls_n	C	cc_{n1}	fc_{n1}	cc_{n2}	fc_{n2}	cc_{ni}	fc_{ni}		
			ef_n	lf_n	F	cf_{n1}	ff_{n1}	cf_{n2}	cc_{n2}	cfni	ff_{ni}		

Y el algoritmo para el cálculo de los tiempos será:

Forward calculation

For (j=1, n, j++)
  $fila = 2j - 1$
  For (i= $1j - 1$, i++)
    $columna = 2i - 1$
    $pccij = afila, columna$ — $\forall\ afila, columna$ *no entero*
    $zccij = afila, columna$ — $\forall\ afila, columna$ *entero*
    $zccij = afila, columna + 1$ — $\forall\ afila, columna + 1$ *entero*
    $pcfij = afila + 1, columna$ — $\forall\ afila + 1, columna$ *no entero*
    $zcfij = afila + 1, columna$ — $\forall\ afila + 1, columna$ *entero*
    $pffij = afila + 1, columna + 1$ — $\forall\ afila + 1, columna + 1$ *no entero*
    $zffij = afila + 1, columna + 1$ — $\forall\ afila + 1, columna + 1$ *entero*
    $esj = \delta j + max\ \{esj, esi + pccij \cdot di;$
      $esi + zccij;$
      $efi + zfcij;$
      $esi + pcfij \cdot di - dj;$
      $esi + zcfij - dj;$
      $efi + pffij \cdot di - dj;$
      $efi + zffij - dj\};$
  $efj = esj + dj$
  $makespan = max\ efj;$

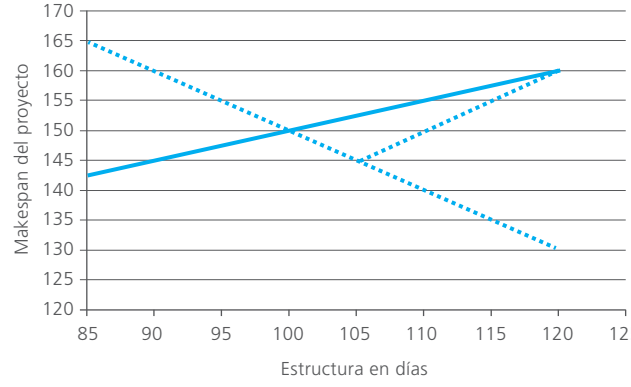
Backward calculation

For (i=n, 1, i−−)
  $lfi = makespan$
  For (j=$i + 1$, n, j++)
  $lfi = min\ (lfi, lsj - pccji \cdot di + di;$
    $lsj - zccij + di;$
    $lsj - zfcij;$
    $lfj - pcfij \cdot di + di;$
    $lfj - zcfij + di;$
    $lfj - pffij \cdot di;$
    $lfj - zffij\};$
  $lsi = lfi - di;$

La implementación del anterior algoritmo en una macro escrita en *VBA* para Excel© de Microsoft© puede ser descargada junto con la generadora de la matriz y las instrucciones de uso en la url: *http://personales.upv/jop nt ie/rupsp_nsa_lean.zip*. La anterior implementación no contempla las dependencias de relación y desfase simultáneas, aunque puede ser fácilmente solventando considerando una nueva tarea cuya duración sea el desfase y su dependencia la relación.

Como los valores de las dependencias son directamentemente proporcionales a la producción, los tiempos de las tareas evolucionarán necesariamente en la misma dirección que las modificaciones realizadas en las duraciones de las tareas, desapareciendo completamente las tareas críticas inversas (ver Figura 13), ofreciendo siempre valores correctos para él, posibilitando la aplicación de modelos de optimización que actúen no solo sobre el desplazamiento (δi) de las tareas, sino también sobre su duración (di).

Figura 13. *Evolución del vs de estructura con dependencias Lean generalizadas simultáneas*



7. Experimentación, validación y conclusiones

No obstante lo anterior, se ha experimentado y validado el algoritmo *Lean* propuesto con dependencias generalizadas (*GPR´s*) de desfase y de producción, para poder comparar los resultados obtenidos tras alterar en ambas direcciones las duraciones de un proyecto de referencia y volver a calcularlo con el algoritmo tradicional y el algoritmo propuesto, para establecer la relevancia de la criticidad inversa para determinar la bondad de los resultados y su adecuación a los problemas reales de construcción.

Para la experimentación se ha utilizado el generador de instancias aleatorias ProGen (Kolisch, 2005). Se han generado 475 instancias en grupos 125 unidades para 10, 60 y 90 tareas cada uno de ellos, que han sido calculadas cada una de ellas de cinco formas diferentes:

- Cálculo 1: En el primer cálculo se han considerado todas las dependencias entre las tareas de la forma tradicional; es decir, con dependencias de desfase exclusivamente (*FCzij k*, *C C zijk*, *F F zijk* y *C F zijk*), del cual obtendremos un valor para el del proyecto que será usado como referencia de los ulteriores cálculos.
- Cálculos 2 y 3: En estos cálculos se ha reducido la duración de todas las tareas 5 unidades de tiempo, y se ha vuelto a resolver en el cálculo 2 con el algoritmo tradicional y dependencias de desfase exclusivamente (*F C zijk*, *C C zijk*, *F F zijk* y *C F zijk*). En el cálculo 3 se han cambiado las dependencias de desfase a relaciones de producción de tipo *RC Cpij k*, *RF Fpij k* y *RC Fpijk*.
- Cálculos 4 y 5: En estos cálculos se ha incrementado la duración de todas las tareas 5 unidades de tiempo, y se ha vuelto a resolver en el cálculo 4 con el algoritmo tradicional y dependencias de desfase exclusivamente (*F C zijk*, *C C zijk*, *F F zijk* y *C F zijk*). En el cálculo 5 se han cambiado las dependencias de desfase a relaciones de producción de tipo *RC Cpijk*, *RF Fpij k* y *RC Fpij k*.

Tras procesar los resultados de los 2.375 cálculos realizados, se han obtenido los siguientes estadísticos:

Figura 14. *Estadísticos cálculos 2 y 3 de la experimentación*

Número de tareas de cada instancia	% del inicial del cálculo 1 Reduciendo 5 días la duración de las tareas			
	Cálculo 2 con algoritmo tradicional		Cálculo 3 con algoritmo propuesto	
	μ		μ	
10	82,01	0,04794557	72,91	0,02492269
60	85,76	0,03916622	72,66	0,02517168
90	84,22	0,02508701	71,71	0,03372721

Figura 15. *Estadísticos cálculos 4 y 5 de la experimentación*

Número de tareas de cada instancia	% del inicial del cálculo 1 Aumentando 5 días la duración de las tareas			
	Cálculo 4 con algoritmo tradicional		Cálculo 5 Con algoritmo propuesto	
	μ		μ	
10	121,53	0,04698666	128,36	0,02567037
60	115,38	0,03759957	124,79	0,02517168
90	113,46	0,02408353	125,72	0,03406448

A partir de los anteriores estadísticos, se puede concluir que:

1. Los valores obtenidos con el algoritmo *Lean* propuesto tras reducir la duración de las tareas es, como mínimo, un 10% inferior al obtenido con el algoritmo tradicional, llegando hasta el 13% de diferencia, al no estar afectado por la criticidad inversa.
2. Los valores de obtenidos con el algoritmo *Lean* propuesto tras aumentar la duración de las tareas es, como mínimo, un 7% superior al obtenido con el algoritmo tradicional, llegando hasta el 12% de diferencia, al no estar afectado por la criticidad inversa.
3. La desviación típica () de los resultados obtenidos utilizando el algoritmo tradicional es sensiblemente mayor para proyectos de pocas tareas que para los de mayor número, poniendo de manifiesto que los proyectos con pocas tareas son más sensibles al efecto de la criticidad inversa que los proyectos con un mayor número de estas, pareciendo absorber parcialmente las aberraciones producidas por la criticidad inversa en el de proyectos con un mayor número de tareas.
4. La desviación típica () de los resultados obtenidos utilizando el algoritmo *Lean* propuesto es sensiblemente mayor para proyectos de muchas tareas que para los de menor número. Fenómeno absolutamente correcto, pues al aumentar el número de variables no homogéneas de una muestra aleatoria, debe aumentar la varianza.

8. Bibliografía

Alarcón, L. F. y Pellicer, E. (2009). Un nuevo enfoque en la gestión: la construcción sin pérdidas. *Revista de Obras Públicas,* (3496), 45-52.

Alarcón, L. (1997). *Lean Construction.* Rotterdam.: A.A. Balkem.

Ballard, G. (1999). Recuperado el 2010, de Lean Construction Institute, Las Vegas, NV; Work Structuring. White Paper #5: www.leanconstruction.org

Ballard, G. (2000). *Lean Construccction Insttute, Las Vegas, NV, Phase Scheduling White Paper #7*. Recuperado el 2010, de www.leanconstuccction.org

Ballard, G. (2000). *The last planner system of production control. PhD dissertation, Civil Engineering.* Birmingham: University of Birmingham.

Ballard, G., & Howell, G. A. (2003). Lean project management. *31(2), 119-133*.

Ballard, G., & Howell, G. (1994). Implementing lean construction: stabilizing work flow.

Bartusch, M., Möhring, R., & Radermacher, F. (1988). Scheduling project networks with resource constraits and time windows. *Annals of Operations Research*, *16*, 201-240.

Crandall, K. (1973). Project Planning with Precedence Lead/ Lag Factors. *Project Management Quarterly 4*, 18-27.

Demeulemeester., E. L. (2002). *Project scheduling: a research handbook*. International series in operations research & management science, Volume 49. Springer. ISBN 10207051, 9781402070518. 685 páginas.

Fondahl, J. W. (1961). A Non-Computer Approach to the Critical Path Method for the Construction Industry. *Department of Civil Engineering, Stanford University.*

Herroelen, W. (1999). The multi-mode resource-constrained project scheduling problem with generalized precedence relations. *European Journal of Operations Research, 119*, 538-556.

IBM. (1964). *Users Manual for IBM 1440 Project Control System (PCS).* IBM.

Kleim, R. (1999). *Scheduling of Resource-Constrained Projects.* Kluwer Academic Publishers.

Kolisch, R. &. (2005). *Project Scheduling Problem Library - PSPLIB.* Obtenido de http://129.187.106.231/psplib/

Koskela, L. (2000). An exploration towards a production theory and its application to construction. *Dissertation for the degree of Doctor of Building Technology. Helsinki University of Technology (Finland)*.

Koskela, L. (1992). Aplication of the Production Philosophy to Construction. *Technical Report #72, Center for Integradted Facility Engineering Departament of Civil Engineering, Stanford University, CA*.

Lean Construction Institute. (2010). *Lean Construction Institute*. Obtenido de http://www.leanconstruction.org/

Malcolm, C. R. (1965). Applications of a Technique for Research and development Program Evaluation".. *Operations Research Vol. 7 pag*, 646-670.

Moder, J. C. (1983). *Project Management with CPM, PERT and Precedence Diagramming.* New York: Van Nostrand Reinchold.

Peixoto, A. & Formoso, C. (1998). EVALUATING BUILDING SYSTEMS BASED ON PRODUCTION PROCESS MANAGEMENT AND LEAN CONSTRUCTION CONCEPTS. *Proceedings IGLC; Evaluating Building Systems Based on Production Process Management and Lean Construction Concepts*.

Ponz Tienda, J. L. (2010). *Ph. D. Thesis: GRCPSP Robusto basado en Producción para proyectos de edificación y Construcción.* Valencia: Universidad Politécnica de Valencia.

Ponz Tienda, J. L. (2009). *Project management con Redes PERT.* Editorial de la Universidad Politécnica de Valencia. 278 pp.

Roy, B. (1962). Graphes et ordonnancements. *Revue Francaise de Recherche Operatinelle*, 323-333.

Roy, B. (1959). Théorie des graphes: Contribution de la théorie des graphes á l'étude de certains problémes linéries. *Comptes rendus des Séances de L'Acadeémie des Sciences, sénce du Avril*, 2437-2449.

Tulio, T., Urgo, M. & Alferi, A. (2008). Project Scheduling with Precedence Relations: An Aplication to Production Planning. *9th Biennal ASME Conference on Engineering System Design and Analysis. Esda 2008.* Haifa, Israel.

Tzortzopoulos, P. & Formoso, C. T. (1999). CONSIDERATIONS ON APPLICATION OF LEAN CONSTRUCTION PRINCIPLES TO DESIGN MANAGEMENT. *IGLC; Considerations on Application of Lean Construction Principles to Design Management*, *7*, 335-344.

Valls, V. & Lino, P. (2001). Criticality analysis in Activity-on-Node Networks with Minimal Time Lags. *Annals of Operations research 102*, 17-37.

Valls, V., Martí, R. & Lino, P. (1996). A Heuristic Algorithm for Project Scheduling with Splitting Allowed. *Journal of Heuristics, 2*, 87-104.

Wiest, J. (1981). Precedence Diagramming Method: Some unusual Characteristics And their Implications For Projects Managers. *Journal of Operations Management. Vol. 1 No. 3*.

Zaderenko, S. G. (1968). *Sistemas de Programación por camino crítico: PERT-CPM-MAN SCHEDULING-RAMPS y otros métodos de elaboración y control de programas.* Buenos Aires. Argentina: 001.424 Z16. Librería Mitre. 197 páginas.