Optimización multiobjetivo de la toma de decisiones en gestión de proyectos software basada en simulación

D. Rodríguez[1], M. Ruiz[2], J.C. Riquelme[3], R. Harrison[4]

[1]Universidad de Alcalá, 28871 Alcalá de Henares, Madrid
daniel.rodriguezg@uah.es
[2]Universidad de Cádiz, 11002 Cádiz
mercedes.ruiz@uca.es
[3]Universidad de Sevilla, Av. Reina Mercedes, s/n. 41012, Sevilla
riquelme@us.es
[4]School of Technology, Oxford Brookes University, Wheatley Campus, Oxford OX33 1HX, UK
rachel.harrison@brookes.ac.uk

Abstract. La simulación se ha utilizado con frecuencia en los últimos años como herramienta de ayuda a la optimización de la toma de decisiones en la gestión de proyectos software. Sin embargo, las herramientas actuales que permiten construir y simular los modelos solamente incluyen módulos que permiten la optimización de un único objetivo. Esto no parece suficiente para un ámbito como el de la gestión de proyectos software en el que frecuentemente hay que tomar decisiones que optimicen determinados resultados que, a menudo, entran en conflicto. En este trabajo se presenta un enfoque que consiste en la aplicación de técnicas de optimización multiobjetivo a los resultados obtenidos por el modelo de simulación, con el objeto de permitir a los directores de proyectos software contar con una herramienta que les permita optimizar de manera más efectiva su proceso de toma de decisiones. Para ilustrar la propuesta, se presenta su aplicación en el ámbito de la selección de los valores correspondientes al tamaño del equipo de desarrollo y a la estimación temporal del proyecto de manera que se optimicen, al mismo tiempo, los indicadores de tiempo, coste y productividad en el proyecto.

Keywords. Sistemas Dinámicos, Multiobjetivo, NSGA-II, Gestión de proyectos.

1 Introducción

Los gestores de proyecto se enfrentan a múltiples y contradictorias decisiones durante el desarrollo de un proyecto. Entre las decisiones que tienen que tomar están no sólo el tamaño medio del equipo, pero también el tamaño inicial, que es una de las variables más influyentes en la productividad y, finalmente, en el costo y tiempo requerido para llevar a cabo el proyecto.

El tamaño del equipo ha atraído mucha atención en los últimos años. Los equipos grandes se han considerado ineficaces, mientras que los equipos pequeños son percibidos como mejores. Brooks [2] ya afirmó en 1975 que la asignación de más desarrolladores a un proyecto atrasado lo retrasará aún más, debido a la curva de aprendizaje e incremento de la comunicación. Además, el tamaño del equipo no se mantiene estable durante todo el ciclo de vida del proyecto. Al contrario, son necesarias además de políticas de planificación y gestión para determinar el tamaño del equipo inicial, las políticas para añadir o quitar personal del equipo inicial a lo largo del ciclo de vida del proyecto.

Por lo tanto, los gestores de proyecto necesitan métodos fiables para decidir acerca de los efectos de sus decisiones respecto a la tasa de cambio en los equipos de desarrollo. Desde la aplicación de Sistemas Dinámicos (SD) para el modelado de proyectos de software de Abdel-Hamid y Madnick [1], los SD de simulación se ha aplicado a muchos aspectos de desarrollo de software y de gestión. Tales simulaciones permiten a los gestores de proyectos generar y ejecutar modelos que les ayudan a entender mejor las distintas estrategias de candidatos y decisiones llevadas a lo largo del proyecto. Sin embargo, una evaluación simple de los resultados de la simulación de un modelo a menudo no es suficiente para determinar las mejores decisiones del proyecto. Por lo general, es necesario un estudio más exploratorio para determinar la combinación más adecuada de las decisiones que conducen los mejores resultados del proyecto. La optimización en simulación se puede definir como el proceso de hallar los mejores valores de algunas variables de decisión para un sistema [8]. En la actualidad, la funcionalidad optimización se encuentra como parte de los paquetes de simulación, siendo el enfoque metaheurístico entre los métodos más utilizados, es decir, técnicas metaheurísticas definidas como la familia de algoritmos aproximados (estocásticos) de búsqueda iterativa en el espacio de soluciones.

Sin embargo, mientras que los enfoques ya aplicadas en la simulación de paquetes suelen ofrecer resultados fiables cuando se centra en encontrar la solución óptima para un determinado objetivo, por lo general no proporcionan la funcionalidad para la optimización multiobjetivo. En este trabajo se describe un enfoque que consiste en utilizar una técnica de optimización multiobjetivo con el fin de ayudar a los gestores de proyectos software a definir los mejores valores para el tamaño del equipo inicial y las estimaciones de tiempo, costo y productividad de un proyecto dado. Las fases llevadas a cabo en este trabajo son los siguientes. En primer lugar, hemos desarrollado un modelo de simulación SD basado en la literatura y trabajos anteriores. En segundo lugar, hemos generado una base de datos con todas las combinaciones posibles de las entradas de la simulación. Aunque, en teoría, el algoritmo multiobjetivo debe llamar a la herramienta de simulación tantas veces como sea necesario, mientras que converga a los valores óptimos, esto no es posible por problemas de licencia. Por lo tanto, en el proceso de generar la base de datos con resultados de entrada y salida, probablemente ejecutando los modelos de simulación muchas veces más de lo realmente necesario.

2 Modelo de simulación para la gestión de proyectos software

El modelo de simulación empleado se ha creado siguiendo la metodología de Law [6]. Esta sección describe este modelo de acuerdo con la propuesta realizada por Kellner para la descripción de modelos de simulación [5].

2.1 Propósito y ámbito del modelo

El propósito de un modelo de simulación se define como aquellas preguntas que el modelo debe ser capaz de ayudar a responder. En nuestro caso, el propósito del modelo es ayudar a analizar el efecto que la incertidumbre en la estimación inicial del plazo y en el tamaño del equipo inicial del proyecto tiene sobre los indicadores claves del éxito del proyecto, tales como el tiempo necesario para el desarrollo, el coste y la productividad. Otra cuestión importante es determinar el ámbito del modelo de simulación, ya que éste debe tener la extensión suficiente para responder las cuestiones del propósito del modelo. En este caso, el ámbito del modelo corresponde con el de un proyecto de desarrollo de software de tamaño medio e integrado por un único equipo.

A continuación, se resumen las variables de entrada y salida más importantes del modelo.

2.2 Variables de salida

Las variables de salida son los elementos de información necesarios para poder responder a las preguntas claves especificadas en el propósito del modelo. Para este estudio, necesitaremos las siguientes variables de salida:

- *FinProyecto* (*Time*): Tiempo en el que finaliza el proyecto.
- *Coste* (*Cost*): El coste final del proyecto.
- *Productividad* (*Prod*): La productividad media alcanzada por el equipo durante el proyecto. Se calcula mediante la relación existente entre el tamaño del proyecto medido en puntos de función (*Function Points -FP-*) [7] y el tiempo requerido para finalizar el proyecto.

Otras variables de salida que son útiles para completar el análisis durante la simulación son:

- Fracción Completada (*Fraction Complete*): El porcentaje del proyecto completado en cualquier momento dado de la simulación.
- Personal Efectivo (*Effective Workforce*): La tasa de trabajo efectiva desarrollada por el equipo.

2.3 Parámetros de entrada

Los parámetros de entrada que se incluyen en el modelo dependen de cuáles son las variables de salida del mismo y la abstracción del proceso realizada. Para simular un proyecto de desarrollo de software se necesitan diferentes parámetros de entrada que permitan adaptar el modelo a las características tanto del proyecto como de la organización que se va a simular.

En nuestro caso, el modelo construido contiene los parámetros de entrada que describen las características del proyecto que se va a desarrollar tales como las estimaciones iniciales de tamaño y tiempo, el nivel de calidad deseado en el producto, el tamaño inicial del equipo, su composición y el tamaño máximo permitido, el salario en función de los niveles de experiencia, etc. Además, el modelo también dispone de un conjunto de parámetros de entrada que permiten adaptar el modelo a las características de la organización que desarrolla el proyecto, tales como: los retrasos en la contratación o las reasignaciones, el tiempo medio de realización de trabajo extra, el efecto de la fatiga en la productividad y en la calidad, etc.

Al objeto de proporcionar una explicación clara del modelo, solamente se describirán aquí los parámetros de entrada que nos permiten modelar la toma de decisiones relacionada con el propósito del modelo, es decir, el tamaño inicial del equipo de desarrollo y su composición en relación con los niveles de experiencia, junto con las estimaciones de tamaño y temporales del proyecto:

- Personal Nuevo Inicial ($NoviceWf$): El número inicial de personas sin experiencia asignadas al proyecto.
- Personal Experto Inicial ($ExpWf$): El número inicial de personal experto asignado inicialmente al proyecto.
- Tamaño Proyecto ($Size$): La estimación del tamaño del proyecto medido en puntos de función.
- Tiempo Estimado ($SchldTime$): La estimación del plazo estimado para la finalización del proyecto.

2.4 Abstracción del proceso

Cuando se desarrolla un modelo de simulación es necesario identificar los elementos claves del proceso que se va a simular, sus interrelaciones y su comportamiento, para incluirlos en el modelo. Es necesario recoger en el modelo aquellos elementos que son relevantes para el propósito del modelo y que se consideran que afectan a los valores que toman las variables de salida. El modelo desarrollado está estructurado en tres subsistemas:

- *Desarrollo*: Este subsistema modela el proceso de desarrollo de software excluyendo las actividades de especificación de los requisitos, y los procesos de operación y mantenimiento.
- *Gestión del equipo*: Este subsistema modela las actividades de contratación, formación, asimilación y transferencia de los recursos humanos. Incluye el comportamiento recogido en la ley de Brooks para modelar las sobrecargas por formación y comunicación derivadas del tamaño del equipo.

– *Control y planificación*: Este subsistema proporciona la estimación inicial del proyecto y modela cómo y bajo qué circunstancias estas estimaciones serán revisadas a lo largo del ciclo de vida del proyecto.

Bajo el enfoque de simulación de la Dinámica de Sistemas, cualquier sistema independientemente de su complejidad, está formado por una red de lazos de realimentación cuyas interacciones determinan el comportamiento dinámico del sistema simulado. Por lo tanto, una gran parte del trabajo de construcción de modelos de simulación bajo este enfoque consiste en determinar y representar estos lazos de realimentación, que junto con las estructuras de flujos y niveles, los retrasos temporales y las no linealidades determinan la dinámica del sistema. Para el propósito de este estudio, el modelo de simulación construido consiste en una red de 77 lazos de realimentación interrelacionados y 89 ecuaciones.

2.5 Experimentos de sensibilidad

Utilizando el modelo descrito, se diseña un escenario para la simulación y análisis de la sensibilidad de las principales variables de salida ante la variación de los principales parámetros de entrada.

Supongamos que el tamaño del proyecto se ha estimado en 500 puntos de función y que, a partir de los datos históricos de la organización el tiempo requerido para desarrollar un punto de función son dos días. Por tanto, el tiempo estimado para desarrollar el proyecto será de aproximadamente 50 meses. Supongamos además que para este proyecto en particular se dan una serie de circunstancias en relación con la estimación temporal que implican que el director del proyecto posea cierta incertidumbre en relación con las estimaciones temporales y el tamaño inicial del equipo de proyecto.

En este contexto, un experimento de sensibilidad puede ayudar al director del proyecto a visualizar el efecto que la infra- o sobreestimación del plazo del proyecto, además del tamaño inicial del proyecto y su composición en relación con el personal experto y sin experiencia pueden tener sobre los resultados del proyecto.

La tabla 1 recoge los valores de los parámetros de control del experimento de sensibilidad. En este experimento, el modelo de simulación se ejecuta para obtener una base de datos que contiene los valores asociados a todas las posibles combinaciones de los valores de los parámetros de control del experimento. Considerando este experimento de sensibilidad, suponemos que el tamaño mínimo del equipo de desarrollo es de dos personas expertas. El número de personas expertas puede variar de dos a diez personas. En relación con el número inicial de personas sin experiencia en el equipo, los valores varían en un rango de 0 a 10. Estas restricciones llevan a contemplar equipos cuyo tamaño inicial no lleva a superar como máximo las veinte personas. En relación con las estimaciones temporales, el experimento contempla variaciones en un rango de 45 a 80 meses.

La figura 1 muestra la sensibilidad de la variable *FracciónCompletada*. Si esta variable de salida alcanza el valor 1, significa que el proyecto está terminado ya

Table 1. Parámetros de control para el experimento de sensibilidad

Parametro de entrada	*Rango*	*Salto*
Initial Novice Workforce	[0-10]	1
Initial Experienced Workforce	[2-10]	1
Scheduled Completion Time	[45-80]	5

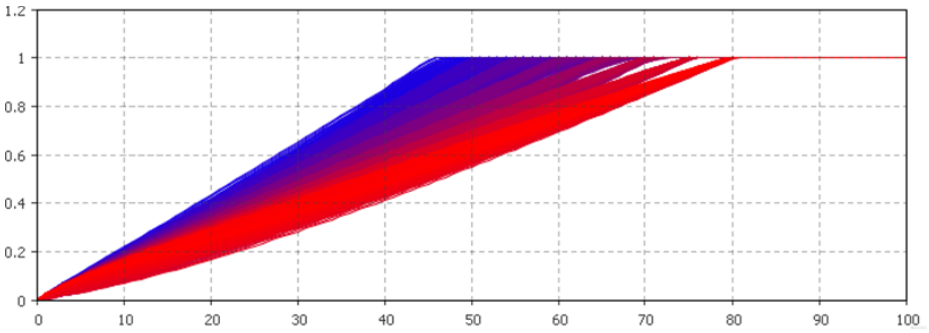

Fig. 1. Sensibilidad de la variable *FractionComplete*

que el 100% de las tareas previstas han sido desarrolladas. Según el experimento realizado, el plazo final del proyecto se sitúa entre los 44 y los 81 meses.

La figura 2 muestra la sensibilidad de la variable de salida del personal efectivo *EffectiveWorkforce*. Esta variable representa la tasa de trabajo efectiva que el equipo de desarrollo es capaz de alcanzar en cualquier momento de la simulación. Su valor se obtiene a partir de calcular la productividad real de cada equipo concreto teniendo en cuenta las sobrecargas por comunicación y formación.

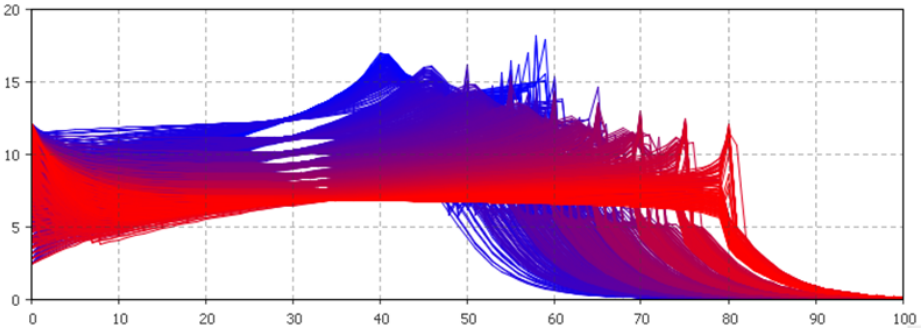

Fig. 2. Sensitivity of the output variable *EffectiveWorkforce*

La figura 3 muestra la sensibilidad de la variable de salida Coste. Esta variable muestra la evolución temporal del coste de cada proyecto simulado. Como es

previsible, cuanto mayor es el tamaño del equipo, mayores serán los costes del proyecto.

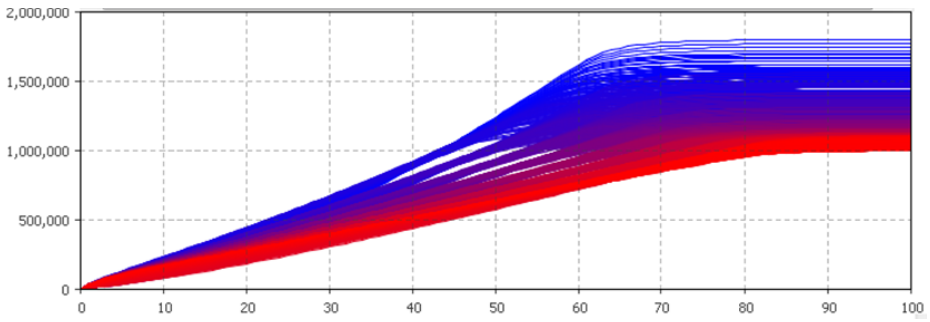

Fig. 3. Sensibilidad de la variable *CumulativeCost*

Entre las muchas decisiones de gestión que necesita adoptar un director de proyecto, las que guardan relación con la composición del equipo de proyecto son las que tienen una mayor repercusión en la productividad [10]. Esta afirmación conlleva la necesidad de encontrar la evidencia empírica sobre las relaciones existentes entre los atributos del proyecto, la productividad y los niveles de experiencia de los miembros del equipo de proyecto que pueden conducir a optimizar las decisiones de gestión. Más concretamente, y en relación con el tamaño el equipo de proyecto, se acepta generalmente que el tiempo empleado en la comunicación entre los miembros del equipo se incrementa conforme crece el tamaño del equipo. Por tanto, el tamaño del equipo afecta directamente a las decisiones relacionadas con la estimación temporal, que es uno de los factores directamente relacionados con el éxito de los proyectos [11]. Además, el tamaño del equipo es importante cuando se toman decisiones sobre la estructura del equipo y el posible particionamiento de proyectos en subproyectos. Si fuera posible encontrar un tamaño óptimo de equipo, entonces la descomposición de proyectos trozos más pequeños se convierte en una práctica clave de gestión con implicaciones directas es la toma de decisiones relacionada con la gestión de equipos distribuidos.

2.6 Optimización de la simulación

Una vez que la sensibilidad de las variables de salida del modelo se ha determinado, el siguiente paso consiste en utilizar el modelo para determinar qué combinación de los valores de los parámetros de entrada optimiza los valores de los indicadores claves del proyecto. Las herramientas actuales de simulación proporcionan utilidades de optimización de los modelos de simulación pero sólo utilizan la evaluación de una función de fitness para guiar el proceso de búsqueda del óptimo. Por tanto, esto significa que todos las variables de salida a estudiar deben ser agregadas en una única función objetivo o función de fitness que luego

se utiliza en técnicas clásicas de optimización, tales como el enfriamiento simulado y la búsqueda dispersa.

Este enfoque trae problemas en relación con cómo normalizamos, priorizamos y elegimos los pesos de los diferentes objetivos de la función de fitness global. En la gestión de proyectos software es además frecuente que objetivos que entran en conflicto interaccionen entre ellos de manera no lineal. Por tanto, encontrar una función de fitness adecuada se convierte en un punto crítico en este enfoque dado que el conjunto de soluciones producidas es altamente dependiente de la función seleccionada y de los pesos elegidos.

En este apartado, utilizamos el módulo de optimización que incorpora Anylogic para optimizar los resultados de la simulación. Seguidamente se muestran los resultados obtenidos tanto en optimización simple como multiobjetivo.

1. *Optimización simple.* En la optimización simple la herramienta encuentra el valor de cada uno de los parámetros de entrada que permite maximizar o minimizar una variable de salida determinada.
   La tabla 2 muestra los valores de los parámetros de entrada que optimizan cada variable de salida de acuerdo con los diferentes experimentos de optimización llevados a cabo.

Table 2. Parámtros de entrada para la optimización simple

Salida		Parámetros de entrada		
		$NoviceWf$	$ExpWf$	$SchldTime$
$Cost$	\$992K	0	10	80
$SchldTime$	44.75	3	10	40
$Prod$	11.47	3	10	40

   Puede observarse que el tamaño del equipo inicial y la fecha de terminación planificada varían dependiendo del objetivo que se desea conseguir. Realmente, esto no es una situación muy realista en el contexto de la gestión de proyectos de desarrollo de software ya que los directores de proyecto podrían estar interesados en encontrar cuál es la combinación de los parámetros de entrada que conduce a alcanzar la máxima productividad junto con el mínimo coste y el tiempo de desarrollo menor. Por tanto, se hace necesario realizar una optimización multiobjetivo.
2. *Optimización multiobjetivo.* Cuando se utilizan las herramientas de simulación actuales para optimizar los resultados de la simulación, tales como Anylogic^{TM}, es necesario agregar todos los objetivos en una única función de fitness. Esta función de fitness es entonces maximizada o minimizada dependiendo de los requisitos utilizando generalmente búsqueda dispersa.
   La forma más simple de hacer esto es reunir todos los objetivos en una única función de fitness mediante una función lineal. Para el propósito de este estudio se ha asumido que el coste del proyecto es el objetivo principal de

manera que los pesos se han establecido en función de cómo los restantes objetivos se relacionan con el objetivo principal (Ecuación 1).

$$fitness(i) = cummCost(i) + \frac{timeProjEnds(i)}{weightTime(i)} + \frac{prod(i)}{weightProd(i)} \quad (1)$$

Los experimentos de optimización llevados a cabo con la herramienta determinaron que los valores de los parámetros de entrada que optimizan la función de fitness son: 10 Personal Experto ($ExpWf$) y ningún personal novel ($NoviceWf$) y un plazo de ejecución del proyecto ($SchldTime$) de 80 meses.
El módulo de optimización de Anylogic concluye que independientemente de los pesos que se coloquen en la función lineal, un equipo de desarrollo formado por diez personas expertas y una estimación temporal de 80 meses es la mejor configuración posible para maximizar la productividad y minimizar coste y tiempo de desarrollo. Sin embargo, acabamos de ver que utilizando optimización simple, esta combinación de valores en los parámetros de entrada minimiza el coste, pero no minimiza el tiempo ni maximiza la productividad.
En el siguiente apartado aplicamos y analizamos los resultados de emplear técnicas de optimización multiobjetivo en este problema.

3 Optimización multiobjetico con NSGA-II

Como se ha comentado anteriormente, hay un gran número de problemas dentro en la ingeniería del software que se pueden resolver con técnicas metaheurísticas. A su vez, existen múltiples técnicas metaheurísticas disponibles, y los problemas de optimización multi-objetivo que involucran multiples objetivos y contrapuestos. En general, las soluciones proporcionadas por estos algoritmos se definen mediante el frente de Pareto, que puede ser definido formalmente de la siguiente manera. El conjunto de vectores de decisión no dominados constituyen el emph frente de Pareto , es decir, un conjunto de soluciones para los que no se puede mejorar un objetivo sin empeorar al menos uno de los otros objetivos. En este trabajo, como algoritmo multiobjetivo, se usó NSGA-II (*Non-dominated Sorting Genetic Algorithm-II*), desarrollado por Deb *et al.* [3] como una extensión de una propuesta anterior de Srinivas y Deb [9]. El NSGA-II es un algoritmo eficiente, incluso con un gran número de objetivos y tamaño de la población. El algorithmo NSGA-II se describe a continuación en algoritmo 1.

Los algoritmos genéticos multi-objetivo complementan los modelos de simulación, optimizando múltiples parámetros al mismo tiempo sin hacer suposiciones acerca de qué objetivos tiene prioridad. En este trabajo, hemos utilizado JMetal[1] [4], un *frameowork* que implementa una gran variedad de técnicas multiobjetivo, incluyendo NSGA-II.

[1] `http://jmetal.sourceforge.net/`

Algorithm 1 Algoritmo NSGA-II [3]

$P_0 \leftarrow$ **makeInitalRandomPopulation**()
$Q_0 \leftarrow$ **makeNewPopulation**(P_0)
$R_0 = \emptyset \leftarrow \wedge\ t \leftarrow 0$
while $t \leq max\_generations$ **do**
    $R_t \leftarrow P_t \cup Q_t$
    $\mathcal{F} \leftarrow$ **fastNonDominatedSort**(R_t)
    $P_{t+1} \leftarrow \emptyset \wedge i \leftarrow 1$
    while $|P_{t+1}| + |\mathcal{F}_i| \leq N$ **do**
        crowdingDistance($\mathcal{F}_i$)
        $P_{t+1} \leftarrow P_{t+1} \cup F_i$
        $i \leftarrow i + 1$
    end while
    Sort($\mathcal{F}_i, \prec_n$)
    $_{t+1} \leftarrow P_{t+1} \cup \mathcal{F}_i[1 : (N - |P_{t+1}|)]$
    $Q_{t+1} \leftarrow$ **makeNewPopulation**(P_{t+1}))
    $t \leftarrow t + 1$
end while
return $\mathcal{F}_1$

Teniendo en cuenta el modelo de simulación SD descrito anteriormente y dos objetivos, optimizando *time* y $effort$, se puede observar que hay una gran diferencia entre el costo de terminar el proyecto en la semana 44 o en la semana 45 como se muestra en la Tabla 3 y gráficamente en la figura 4. Esto se debe al hecho de que hay más personal involucrado y después de este punto, se puede observar que el costo se reduce si se aumenta la duración del proyecto con los mismos valores de personal iniciales. Desde el punto de vista de la ingeniería de software es también interesante observar que este *codo* en la figura.

Table 3. Salida del algoritmo NSGA-II para dos objetivos, *Time* and *Cost*

NoviceWf	*ExpWf*	*SchldTime*	*Time*	*Cost* ($(K))
3	10	40	44.75	1,289
2	10	45	44.85	1,287
1	10	45	45.58	1,128
0	10	55	54.90	1,053
0	10	60	59.84	1,035
0	10	65	64.79	1,021
0	10	70	69.72	1,010
0	10	75	74.71	1,001
0	10	80	79.68	992

Se obtiene un poco más de variedad en el número de personas si consideramos tres objetivos, es decir, además de los anteriores la maximización de la productividad. Esto se puede observar tabla 4.

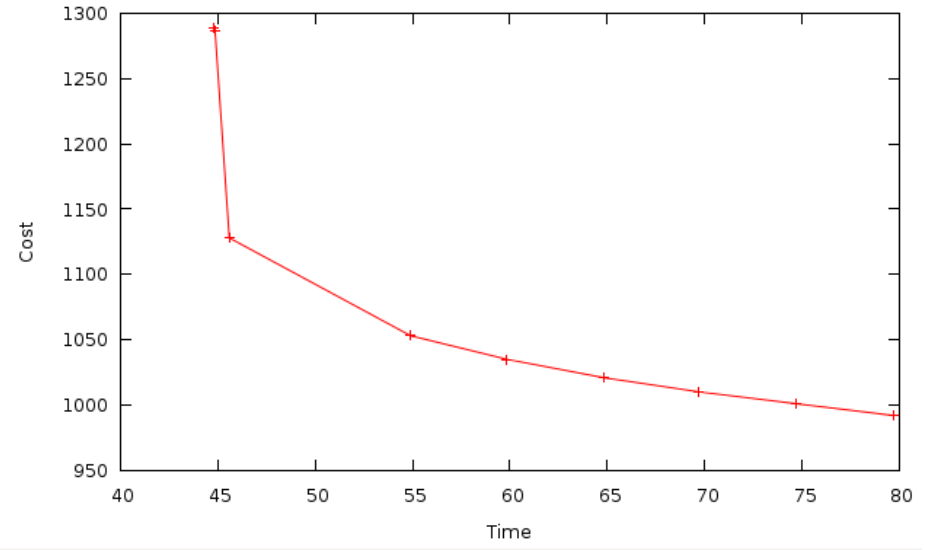

Fig. 4. Gráfica del frente de Pareto para dos objetivos

Table 4. Pareto Front for Three Objectives, *Time*, *Cost* and *Prod*

NoviceWf	*ExpWf*	*SchldTime*	*Time*	*Cost* ($(K))	*Prod*
5	10	35	44.96	1,548	11.12
4	10	40	45.47	1,318	10.99
5	10	45	48.72	1,235	10.26
1	10	50	50.00	1,091	9.99
3	8	60	60.00	1,082	8.33
3	9	65	64.90	1,060	7.70
4	10	70	69.85	1,052	7.15
3	10	75	74.78	1,030	6.68
0	8	80	79.69	993	6.27
0	10	80	79.68	992	6.27

En comparación con los enfoques de un solo objetivo, podemos observar que las soluciones están cerca de los extremos de la gama de soluciones que se encuentran en el frente de Pareto (multiobjetivo). El conjunto de soluciones multiobjetivo puede ayudar a analizar las tendencias del proyecto y explicar las ventajas y desventajas de la aplicación de diferentes políticas.

4 Conclusiones y trabajos futuros

En este trabajo, hemos aplicado una técnica de optimización multiobjetivo a un modelo de simulación de gestión de proyectos usando sistemas dinámicos. Las técnicas de optimización multiobjetivo aplicadas a los proyectos de inginiería del software, ofrecen a los gestores de proyecto un mejor control para la toma de decisiones. Las técnicas multobjective logran mejores resultados en términos de encontrar los parámetros de entrada que maximicen los parámetros de salida tales como el tiempo, coste y productividad que optimizaciones en un sólo objetivo. Esto se debe al hecho de que no hay necesidad de calcular los pesos la hora de combinar los objetivos, y la gama de soluciones en el frente de Pareto también puede ayudar a la comprensión de las diferentes políticas del proyecto.

References

1. Tarek Abdel-Hamid and Stuart E. Madnick. *Software Project Dynamics: An Integrated Approach*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1991.
2. Frederick P. Brooks. *The Mythical Man-Month*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, anniversary ed. edition, 1995.
3. K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. A fast and elitist multiobjective genetic algorithm: Nsga-ii. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197, April 2002.
4. J.J. Durillo, A.J. Nebro, and E. Alba. The jMetal framework for multi-objective optimization: Design and architecture. In *IEEE Congress on Evolutionary Computation (CEC'2010)*, pages 4138–4325, Barcelona, Spain, July 2010.
5. Marc I. Kellner, Raymond J. Madachy, and David M. Raffo. Software rocess simulation modeling: Why? what? how? *Journal of Systems and Software*, 46(2-3):91–105, 1999.
6. Averill M. Law. How to build valid and credible simulation models. In *Proceedings of the 40th Conference on Winter Simulation*, WSC '08, 2008.
7. Christopher J. Lokan. Function points. *Advances in Computers*, 65:297–347, 2005.
8. Sigurdur Ólafsson and Jumi Kim. Simulation optimization: simulation optimization. In *Proceedings of the 34th conference on Winter simulation: exploring new frontiers*, WSC '02, pages 79–84, 2002.
9. N. Srinivas and Kalyanmoy Deb. Muiltiobjective optimization using nondominated sorting in genetic algorithms. *Evolutionary Computation*, 2:221–248, September 1994.
10. Adam Trendowicz and Jörgen Münch. Factors influencing software development productivity – state-of-the-art and industrial experiences. Elsevier, 2009.
11. J.M. Verner, W.M. Evanco, and N. Cerpa. State of the practice: An exploratory analysis of schedule estimation and software project success prediction. *Information and Software Technology*, 49(2):181–193, 2007.