Modelo Dinámico de Simulación de Proyectos de Software con XP

Autor: Tamara Gisel Kasiak[1]

Director: Diego Alberto Godoy[1]

[1] Universidad Gastón Dachary, Centro de Investigación en Tecnología de la Información y Comunicaciones, Departamento de Ingeniería y Ciencias de la Producción.
{tamarakasiak,diegodoy}@dachary.edu.ar

Resumen. Administrar Proyectos de Software siguiendo Programación Extrema (XP) implica implementar, de forma conjunta y al extremo, prácticas ya conocidas en el ámbito del Desarrollo de Software, lo que torna a esta actividad aún más compleja. Para tratar esta complejidad, se ha construido un Modelo Dinámico de Simulación (siguiendo los lineamientos de la Dinámica de Sistemas), que agrupa las variables involucradas en un proyecto llevado a cabo con XP, que permite analizar el efecto de la implementación conjunta de las prácticas de dicha metodología y ayuda en la gestión de este tipo de proyectos. Este modelo se ha validado con datos de tres proyectos reales. Además, se han diseñado y ejecutado una serie de experimentos sobre el mismo y se ha realizado el Análisis de Sensibilidad de sus variables más importantes. El modelo construido sirve como ayuda a administradores de proyectos novatos, permitiéndoles estimar de antemano las consecuencias de sus decisiones.

Palabras Claves: Administración de Proyectos de Software, Programación Extrema, Dinámica de Sistemas.

1 Introducción

En el ámbito de la Ingeniería del Software la evolución de las Metodologías de Desarrollo de Software ha llevado a la aparición de las denominadas Metodologías Ágiles, las cuales están destinadas a romper con la rigidez de las Tradicionales, caracterizadas por la extensa documentación del proceso de desarrollo y por la inflexibilidad ante los cambios.

Una de las Metodologías Ágiles más importante y reconocida es la Programación Extrema (XP) [1], para aquellos proyectos de software donde el cambio en los requerimientos es la norma. Por tal motivo reúne un conjunto de prácticas sencillas ya conocidas, pero que en este caso son llevadas a cabo conjuntamente y en forma extrema. Debido a esto, la gestión de proyectos de desarrollo de software se torna algo impredecible y compleja y es difícil anticipar por sus administradores, los efectos que tiene sobre la marcha del mismo, la aplicación de las diferentes prácticas de XP.

No obstante, si bien existen actualmente diversas herramientas destinadas a ayudar a los administradores de proyectos en la estimación y toma de decisiones, como ser los Modelos Dinámicos de Simulación, que permiten evaluar diferentes alternativas de decisión en la gestión, sin intervenir en el desarrollo real del proyecto, la mayoría de ellos han sido desarrollados para proyectos que se llevan a cabo con Metodologías Tradicionales, como ser el modelo de Abdel- Hamid y Madnick [2] y el Modelo Dinámico Reducido [3]. Por lo tanto, es necesario continuar desarrollando modelos similares al que se presentará en este trabajo, que permitan estimar y tomar decisiones en proyectos de software llevados a cabo con Metodologías Ágiles.

2 Trabajos Relacionados

El primer Modelo Dinámico para Proyectos de Desarrollo de Software que surgió fue el Modelo de Abdel-Hamid y Madnick, el cual data de los años 90 y no tiene en cuenta las fases de Definición de Requisitos, Operación y Mantenimiento, por lo que se centra en el Diseño, Codificación, Revisión, Corrección y Pruebas, considerando que los requisitos del proyecto se mantienen estables a lo largo del ciclo de vida. Este modelo está dividido en cuatro subsistemas complejos que son los siguientes: Gestión de Recursos Humanos, Producción de Software, Control y Planificación.

Por otro lado, existen otros modelos que buscan ampliar el modelo mencionado anteriormente, como ser el Modelo SEPS [4], el Modelo de Draper Laboratory [5], el Modelo de Chichacly [6], el Modelo Aranda/Fiddaman/Oliva [7] y el Modelo Multi-proyecto [8]. Además de estos, otro de los más conocidos es el Modelo Dinámico Reducido el cual constituye una reducción del Modelo de Abdel - Hamid y Madnick teniendo en cuenta los “cinco números” en los que se basa la gestión de proyectos según [9], y que son los siguientes: Una medida de la cantidad producida, una medida del tiempo necesario para completar el proyecto, una medida del coste de producción, una indicación de la calidad del producto y una medida de la productividad media de los técnicos.

Por otra parte, más allá de estos modelos destinados a simular proyectos de software desarrollados con Metodologías Tradicionales, se han venido implementando otras técnicas de estimación de esfuerzo para proyectos ágiles, las cuales se debaten entre la utilización de medidas absolutas (horas, días, días ideales, puntos, dinero) o bien escalas de unidades (Escala de Cohn, Tallas de Camiseta). Sin embargo, en proyectos ágiles generalmente las estimaciones son realizadas únicamente por el personal experto el cual se basa en la experiencia pasada. En este sentido, los modelos dinámicos de simulación podrían ser utilizados como herramientas que permitan anticipar y comparar el comportamiento de proyectos de software ágiles, ante determinadas circunstancias inducidas por los administradores de proyectos.

3 Objetivos del Modelo Propuesto

En este trabajo se planteó la construcción de un “Modelo de Simulación Dinámico de Gestión de Proyectos de Desarrollo de Software que utilizan XP”, con el objetivo de

analizar el efecto que tiene el uso de las prácticas de XP en la gestión de proyectos de desarrollo de software y estimar el comportamiento de los mismos.

De esta forma, el modelo permite a los administradores de proyectos, evaluar el impacto de sus decisiones de gestión a lo largo del tiempo, como así también posibilita comparar los resultados de dichas decisiones, ofreciendo datos suficientes para elegir la mejor opción a ser aplicada en el sistema real.

Esto es posible debido a que el modelo habilita a los administradores a realizar cambios en variables críticas del proyecto, como ser recursos, tiempo, tareas a desarrollar, etc., y observar las repercusiones de dichos cambios en el resto del proyecto, y lo más importante, sin comprometer la ejecución real del mismo.

Como se mencionó, el modelo refleja el efecto del uso de prácticas XP en proyectos de desarrollo de software y para ello, su estructura permite simular el desarrollo de una versión o entrega de un proyecto XP a la vez. Es decir, como XP presenta un estilo de desarrollo iterativo e incremental, dentro de un proyecto se negocian con el cliente varias entregas o versiones, por lo que el modelo, está destinado a simular las versiones de a una por vez.

4 Construcción del Modelo

Para la construcción del modelo se utilizó el software VenSim PLE 5.4c (Versión Académica) [10] y se siguieron las etapas de la Metodología de Dinámica de Sistemas [11], la cual consta de tres fases.

De esta forma, en la Fase de Conceptualización se ha construido el Diagrama Causal el cual representa, a través de variables e interrelaciones entre las mismas, la estructura del sistema modelado.

Luego en la segunda fase, denominada Fase de Formulación se construyó el Diagrama de Forrester el cual constituye una traducción de las variables y relaciones del Diagrama Causal a ecuaciones matemáticas, posibles de ser programadas y simuladas. Este Diagrama de Forrester fue además dividido en Subsistemas Conservativos de acuerdo con [12], para facilitar su estudio y análisis.

Finalmente, en la Fase de Evaluación se han realizado las corridas experimentales y de validación del modelo, utilizando para ello casos reales y escenarios que representan situaciones frecuentes en proyectos XP. Estas corridas tuvieron como objetivo analizar las diferentes decisiones y escenarios que son posibles evaluar a través de la utilización del modelo.

A continuación se presentan los subsistemas conservativos más importantes, correspondientes al Diagrama de Forrester del modelo.

4.1 Subsistema Recursos Humanos

Este subsistema, tal como se puede ver en la Figura 1, contiene la estructura que representa los procesos de planificación de los programadores a contratar, de contratación, de adquisición de experiencia de los programadores novatos y de abandono del proyecto por parte de los programadores.

Estos procesos tienen lugar de acuerdo con la “Diferencia en el número de programadores deseados” en el proyecto, que es generada cuando el usuario del modelo actualiza la variable “Pares de Programadores Deseados”.

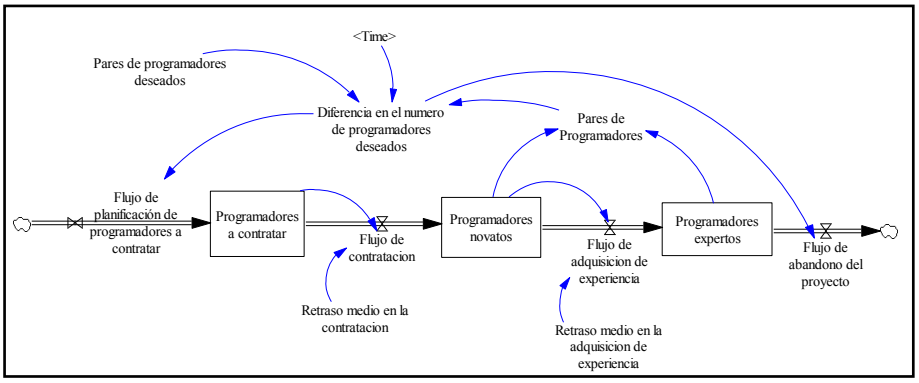

Fig. 1. Subsistema Recursos Humanos

4.2 Subsistema Factor de Carga

El subsistema Factor de Carga de la Figura 2 representa la estructura por medio de la cual se actualiza el Factor de Carga del equipo, para cada iteración, cuando varía la cantidad de programadores novatos y/o expertos, o sus correspondientes factores de carga.

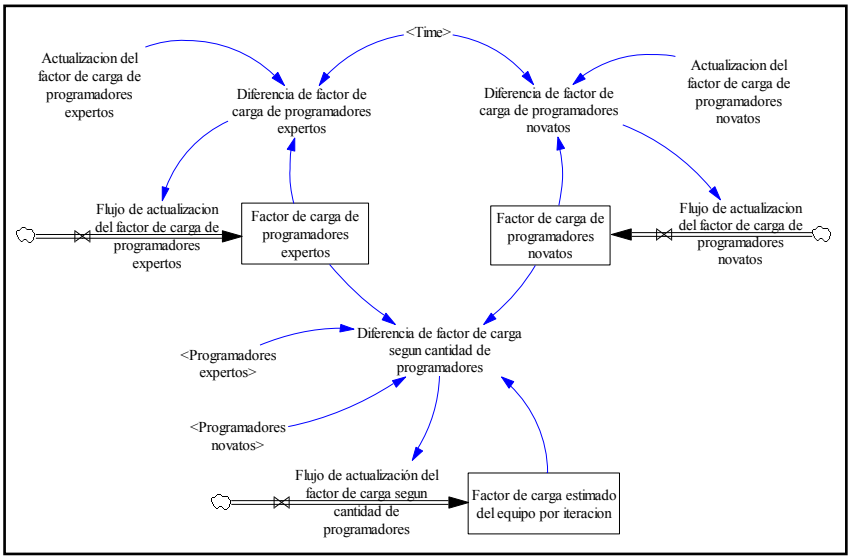

Fig. 2. Subsistema Factor de Carga

4.3 Subsistema de Desarrollo de Pruebas de Unidad

El subsistema de Desarrollo de Pruebas de Unidad de la Figura 3, representa los procesos de planificación y desarrollo de Pruebas de Unidad por iteración.

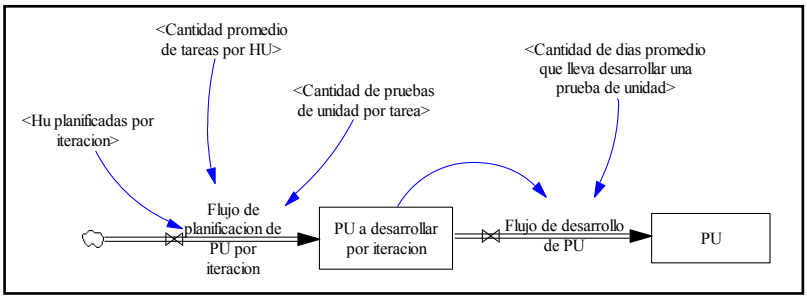

Fig. 3. Subsistema de Desarrollo de Pruebas de Unidad

4.4 Subsistema Presión en el Plazo

El subsistema Presión en el Plazo que puede ser visto en la Figura 4, representa la diferencia existente entre el "Tiempo real de entrega estimado por iteración" y el "Tiempo de entrega pactado" con el cliente, para cada iteración. Esta diferencia activa el proceso de actualización de la "Presión en el Plazo".

Si la "Presión en el Plazo" es positiva, indica un retraso en la entrega de la iteración, si es negativa en cambio, indica que la entrega de la iteración se estima con un adelanto.

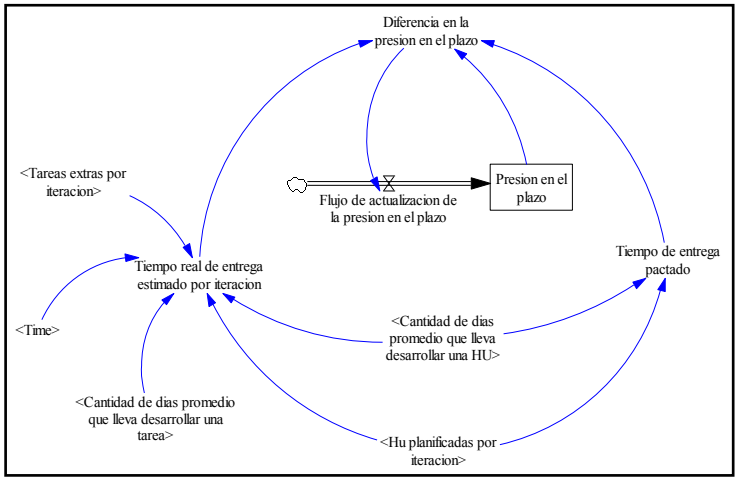

Fig. 4. Subsistema Presión en el Plazo

4.5 Subsistema de Planificación

En la Figura 5 se puede observar el Subsistema de Planificación, en el cual una de las variables más importantes es el Tamaño de la versión en Historias de Usuario (HU), la cual puede ser modificada por el usuario a lo largo del desarrollo, por medio de la variable "Actualización de la cantidad de HU a desarrollar en la versión", lo cual activa el proceso de actualización del tamaño de la versión.

Otra variable importante es la "Cantidad de HU por desarrollar", que almacena en cada momento del desarrollo de la versión, cuántas HU quedan por desarrollar. Así, cuando se planifica una iteración esta variable disminuye en la cantidad de HU que fueron planificadas para dicha iteración, a través del "Flujo de planificación de HU".

Por otro lado, este subsistema representa también los procesos de re-planificación de HU con errores de aceptación y de HU no acabadas.

Cabe mencionar que la planificación de iteraciones se lleva a cabo mediante la variable "Planificación de HU por iteración", que almacena el momento y la cantidad de HU que son planificadas. Además, esta variable también puede ser actualizada por el usuario del modelo, en cualquier momento del desarrollo de la versión.

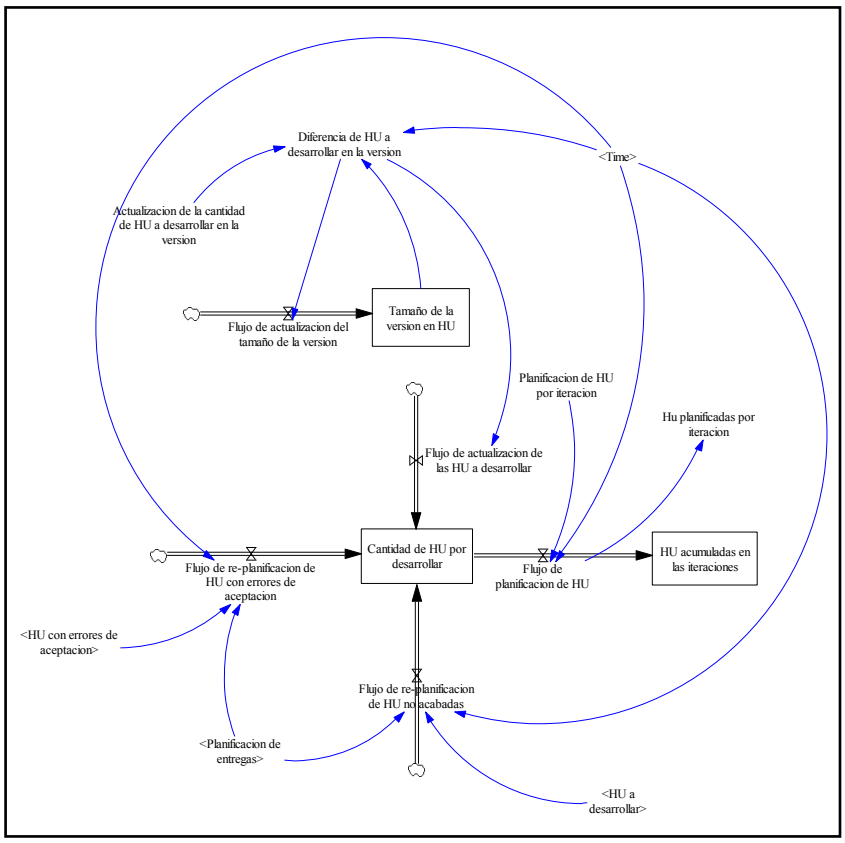

Fig. 5. Subsistema de Planificación

4.6 Subsistema de Desarrollo de Tareas

En la Figura 6 se puede observar el Subsistema de Desarrollo de Tareas, el cual representa todas las etapas por las que pasan cada una de las tareas en que se dividen las HU, comenzando con la planificación de dichas tareas para cada iteración, la cual se basa en la cantidad de "HU planificadas por iteración" y en la "Cantidad promedio de tareas por HU".

De esta forma, una vez planificadas las tareas, se encuentran inicialmente en el nivel "Tareas a desarrollar", y luego van pasando por todos los niveles hasta llegar al nivel "Tareas en Producción", el cual representa la cantidad de tareas desarrolladas en cada iteración.

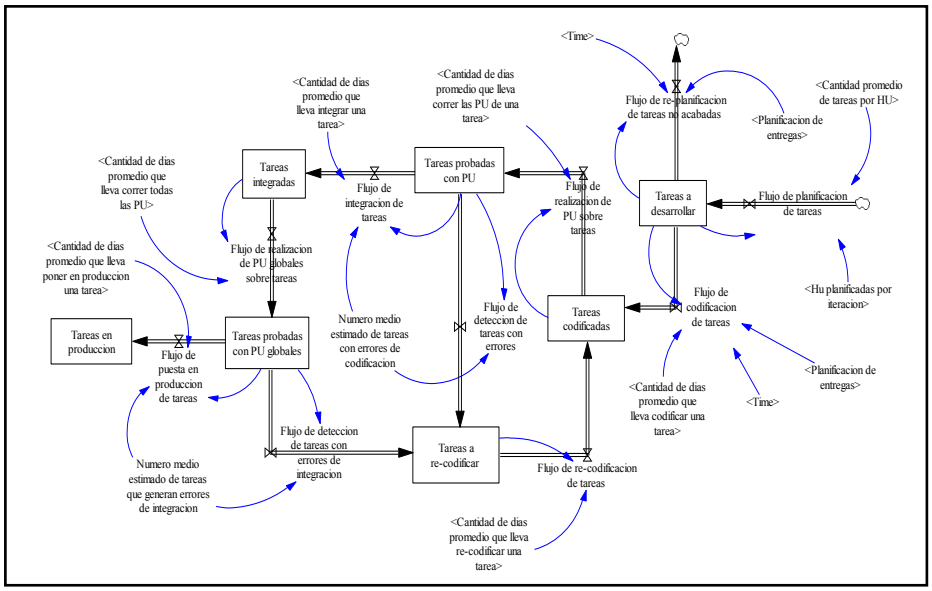

Fig. 6. Subsistema de Desarrollo de Tareas. (Fuente [13])

Por otro lado, además de los subsistemas presentados anteriormente, el modelo consta de otros subsistemas, que son los siguientes: Subsistema de Desarrollo de Pruebas de Aceptación, Subsistema de Desarrollo de Historias de Usuario, Subsistema Horas de Trabajo por Día, Subsistema de Actualización de los Días Ideales de Ingeniería por HU y por Tarea y finalmente, un subsistema que agrupa las Variables Auxiliares y Constantes del Modelo.

En la Tabla 1 del Anexo se detallan las ecuaciones de las variables más relevantes de cada subsistema presentado anteriormente.

5 Validación del Modelo

Para la validación del modelo se utilizaron datos de tres Proyectos de Software reales realizados con XP.

El primero de ellos fue un Proyecto de Desarrollo de un Sistema de Gestión de una Empresa de Confecciones, cuyos detalles se pueden ver en [14].

El segundo caso de validación fue el Sistema "ONess, un Proyecto Open Source para el Negocio Textil Mayorista" [15].

Finalmente, la tercera validación se realizó con un "Caso Práctico de Aplicación de la Metodología XP al Desarrollo de Software para un Mini mercado". Los detalles de este caso se pueden encontrar en [16].

A continuación se presentarán algunos de los resultados obtenidos con el segundo y tercer caso de validación.

5.1 Sistemas ONess.

El Sistema ONess, correspondiente al segundo caso de validación, consistió en el desarrollo de una versión, dividida en tres iteraciones. Fueron 16 las historias de usuarios (HU) planificadas para ser desarrolladas en dicha versión, por un equipo de 7 programadores. No obstante, al iniciar el desarrollo el cliente decide quitar 4 de las 16 HU planificadas, restando solamente 12.

El detalle de las iteraciones de este proyecto se puede observar en la Tabla 1.

Tabla 1. Detalle de las iteraciones del proyecto presentado en [15]

Iteración	Historias de Usuario (HU)	Tareas en Promedio por HU	Duración Real en días	Duración Estimada en días
1	2	8	21	7,5
2	3	12	18	9,5
3	7	18	8	19,7
TOTAL	**12**	**38**	**47**	**36,7**

De acuerdo con los datos presentados en la Tabla 1, en la Figura 7 se puede observar la planificación de las HU para cada una de las iteraciones. Esta planificación es representada en el modelo por el "Flujo de planificación de HU".

Cabe mencionar además que la planificación de iteraciones es un aspecto que puede ser controlado por el usuario del modelo durante la ejecución del mismo, a través de la variable "Planificación de HU por iteración". Esta variable permite indicar en qué momento del desarrollo de la versión se planifican las iteraciones y qué cantidad de HU se planifican para cada una de ellas.

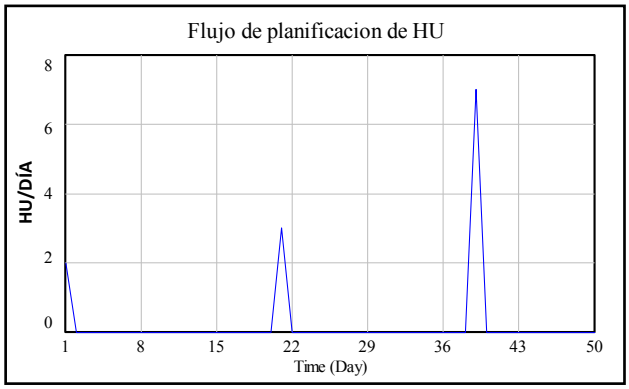

Fig. 7. Planificación de HU para cada iteración

Otra de las variables importantes del modelo es la "Cantidad de HU por desarrollar", la cual fue inicializada al comenzar el proyecto con 16 HU, luego debido a lo mencionado anteriormente, este valor desciende a 12, cuando el cliente elimina 4 HU del ámbito de la versión, al iniciar el desarrollo del proyecto. De aquí en más, este nivel se reduce cada vez que se planifica una iteración (Figura 7), disminuyendo de acuerdo con la cantidad de HU planificadas. Esto puede ser comprendido mejor observando el gráfico "Cantidad de HU por desarrollar" de la Figura 8.

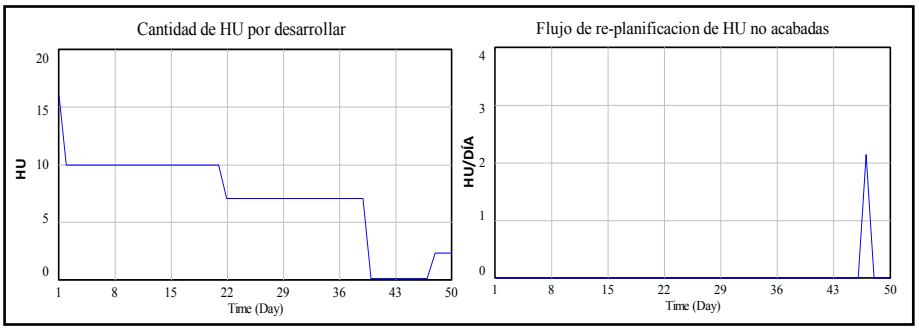

Fig. 8. Cantidad de HU por desarrollar y Re-planificación de HU no acabadas

De esta forma, y como puede verse en la Figura 8, la "Cantidad de HU por desarrollar" desciende a 10 el día 2 del proyecto luego de haberse planificado la primera iteración con 2 HU, el primer día del desarrollo del proyecto.

Esta situación se repite para las otras dos iteraciones de la versión. No obstante, una situación particular que contempla el modelo y que se observa en esta figura, es que al final de la tercera iteración la "Cantidad de HU por desarrollar" aumenta.

Este comportamiento se debe a que, según los datos reales obtenidos del proyecto en estudio, la tercera iteración no llega a completarse por falta de tiempo, por lo tanto, las HU no acabadas deben ser re-planificadas para iteraciones posteriores.

Esta re-planificación ocurre a través del "Flujo de re-planificación de HU no acabadas", que incrementa el nivel de "Cantidad de HU por desarrollar" el día 47, como puede verse en la Figura 8.

Finalmente, luego de pasar por todas las etapas del desarrollo, las HU llegan a la última etapa representada por la variable "HU aceptadas por el cliente". Ésta variable es un nivel que representa la acumulación de HU entregadas al cliente, aceptadas por el mismo y por lo tanto finalizadas. Este nivel puede observarse en la Figura 9.

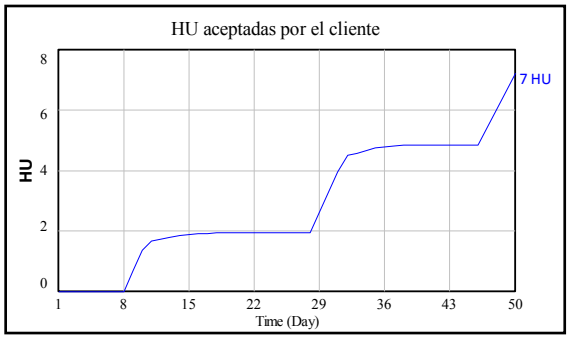

Fig. 9. HU aceptadas por el cliente.

En la Figura 9 se puede observar también que tal como se ha planificado en el proyecto real, la versión es entregada aproximadamente en el día 47, con tres iteraciones realizadas y 7 HU completadas. Las HU no acabadas en la tercera iteración, fueron re-planificadas, como ya se ha explicado anteriormente.

5.2 Sistema para Mini mercado

Como ya se ha mencionado, para la tercera validación del modelo se utilizaron los datos de un caso práctico de aplicación de la metodología XP al desarrollo de software para un mini mercado. Este proyecto fue desarrollado por un equipo de 2 programadores y consistió en el desarrollo de 21 HU divididas en 4 iteraciones. El detalle de dichas iteraciones puede ser observado en la Tabla 2.

Tabla 2. Detalle de las iteraciones del proyecto presentado en [16]

Iteración	Historias de Usuario (HU)	Duración Real en días	Duración Estimada en días
1	5	14	14
2	6	14	14
3	4	14	14
4	6	7	14
TOTAL	**21**	**49**	**56**

De acuerdo con los datos de la Tabla 2 se puede observar que la cuarta iteración tuvo una duración real menor a la estimada, esto se debió a que al comienzo de la misma (día 42) el equipo de desarrollo aumentó el tiempo destinado al desarrollo en "Horas de trabajo por día" de 4 para 8, con lo cual la "Cantidad de días promedio que lleva desarrollar una HU" en el proyecto disminuyó ocasionando que el desarrollo de las HU de la cuarta iteración sea más rápido. Estas situaciones se pueden observar en la Figura 10 presentada a continuación.

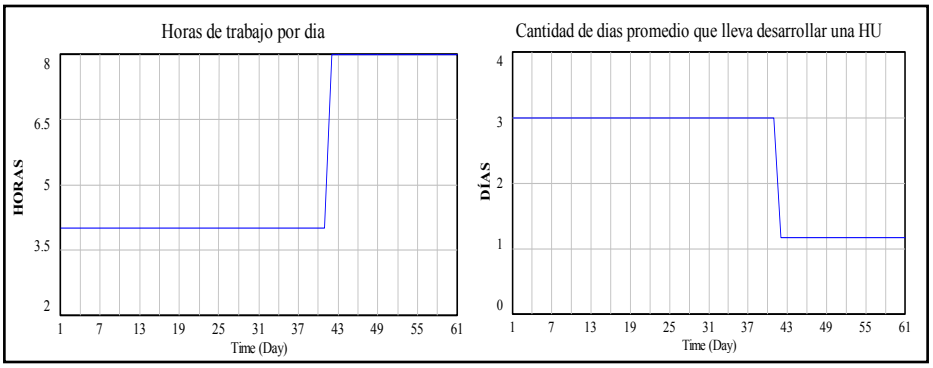

Fig. 10. Horas de Trabajo por Día y Cantidad de Días que lleva desarrollar una HU

Por otro lado, al igual que para el caso presentado en la sección anterior en la Figura 11 se puede ver el "Flujo de planificación de HU" para todo el proyecto y la "Cantidad de HU por desarrollar", la cual al comienzo del proyecto es de 21 HU disminuyendo hasta 0 al final del mismo.

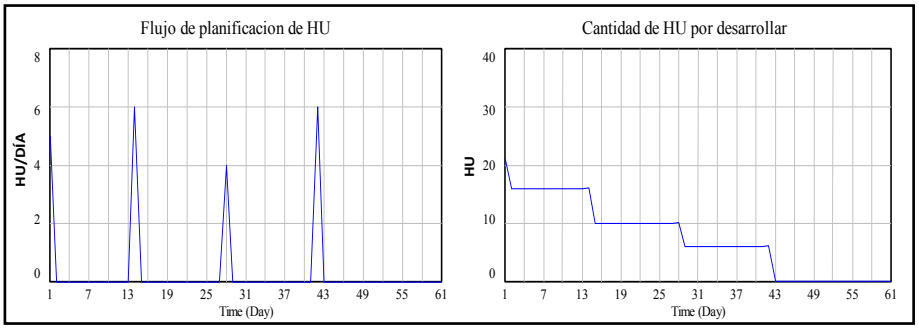

Fig. 11. Planificación de HU y Cantidad de HU por desarrollar

De esta forma, de acuerdo con la Figura 11, se puede observar que el día 1 del proyecto se planifica la primera iteración con 5 HU, por lo tanto la "Cantidad de HU por desarrollar" disminuye de 21 a 16 HU. De igual manera, el día 14 correspondiente al final de la primera iteración se planifican las 6 HU de la segunda iteración, disminuyendo la "Cantidad de HU por desarrollar" de 16 para 10 HU. El mismo comportamiento se aplica a las dos siguientes iteraciones.

Por otra parte, tal como se ha mencionado para el caso de validación anterior, una vez planificadas las HU son desarrolladas pasando por diversas etapas hasta alcanzar la última de ellas representada en el modelo por la variable "HU aceptadas por el cliente", la cual puede ser observada en la Figura 12.

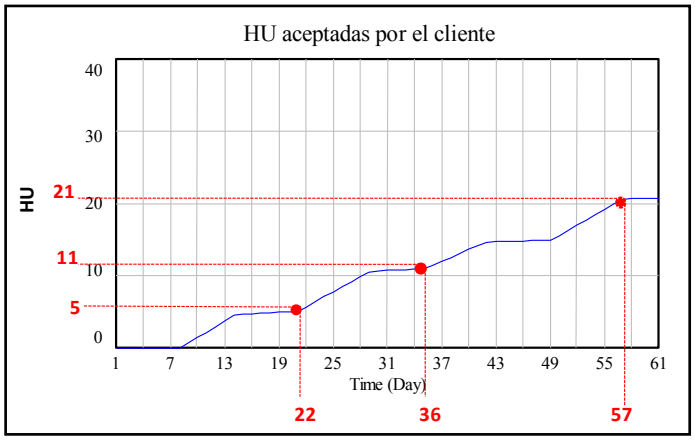

Fig. 12. HU Aceptadas por el cliente

Cabe mencionar que el modelo presenta un retraso de 8 días hasta que es aceptada la primera HU, debido a que son justamente 8 las etapas por las que pasan las HU durante su desarrollo normal. Por lo tanto se puede ver (Figura 12) que a los 14 + 8 días es entregada la primera iteración, con 5 HU terminadas. Del mismo modo, a los 28 + 8 días es entregada la segunda iteración con 6 HU más listas. Lo mismo ocurre para la tercera y cuarta iteración, haciendo un total de 21 HU finalizadas a los 49 + 8 días, correspondientes al final del proyecto.

6 Experimentos Realizados

Al igual que en la validación, el modelo ha sido simulado bajo diversas condiciones que representan situaciones típicas en proyectos XP. En este caso se presenta uno de los experimentos realizados, el cual se refiere al desarrollo de una versión de un proyecto XP, la cual engloba 15 HU a ser completadas en 45 días. Esta versión fue dividida en 3 iteraciones de 5 HU cada una. Así también, del desarrollo de la misma participaron 6 pares de programadores (4 novatos y 8 expertos).

En este escenario se analizó la incorporación de 3 HU más al final de la primera iteración, el abandono del proyecto de 2 programadores al comienzo de la segunda iteración y la re-planificación de HU no acabadas al final de primera y segunda iteración.

De esta forma, en la Figura 13 se puede observar cómo aumenta el tamaño de la versión, de 15 para 18 HU, al actualizarse la cantidad de HU al final de la primera iteración (día 14).

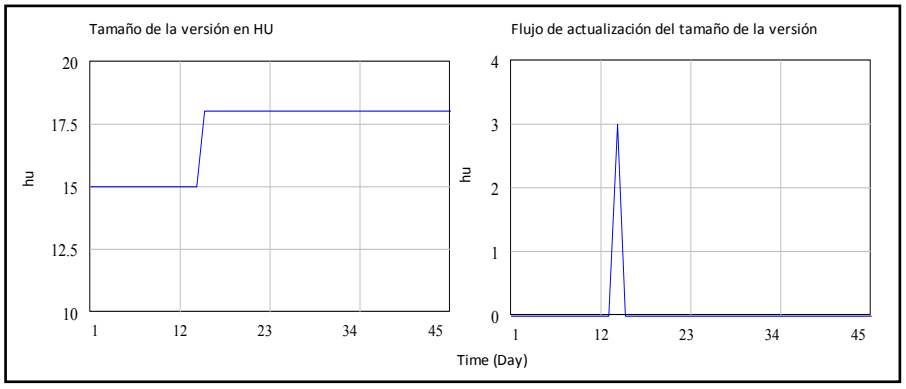

Fig. 13. Actualización del Tamaño de la Versión (Fuente [13])

Por otro lado, en la Figura 14 se puede observar cómo la cantidad de "Pares de Programadores" disminuye el día 15 (comienzo de la segunda iteración), cuando 2 programadores abandonan el equipo de desarrollo.

Debido a esta situación, el crecimiento en la cantidad de "Programadores Expertos" (que aumenta a medida que los novatos adquieren experiencia) se ve interrumpido el día 15, como se puede apreciar también en la Figura 14.

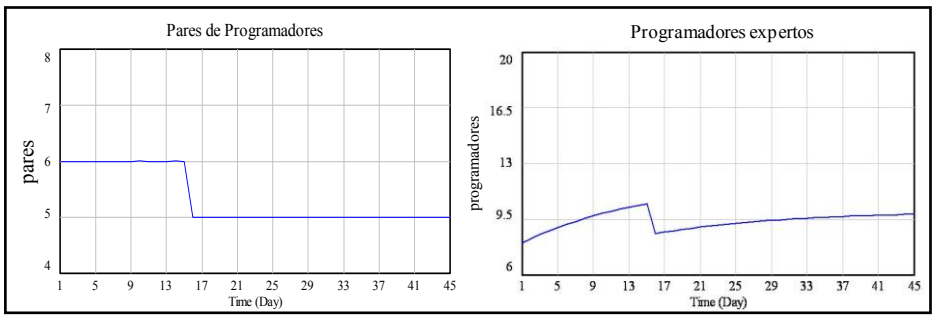

Fig. 14. Pares de Programadores y Pares de Programadores Expertos (Fuente [13])

Finalmente, en la Figura 15 se refleja (a través del "Flujo de re-planificación de HU no acabadas"), la replanificación de HU no acabadas al final de las 2 primeras iteraciones. La replanificación en la iteración 1 corresponde al hecho de que, como se observa en la Figura 14, la cantidad de programadores expertos es baja al comienzo de la versión, y por lo tanto, la cantidad de horas de trabajo diarias disponibles para desarrollo es también baja en este etapa.

La replanificación en la iteración 2 corresponde, en cambio, al incremento en la cantidad de HU planificadas para dicha iteración (Figura 13) y al abandono del proyecto por parte de 2 programadores tal como se mostró en la Figura 14.

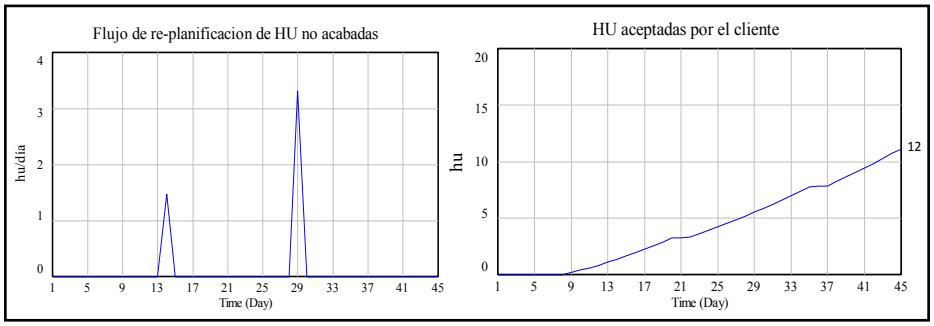

Fig. 15. Replanificación de HU no acabadas y HU aceptadas por el cliente (Fuente [13])

Estas replanificaciones de HU observadas en la Figura 15 ocasionan que la cantidad de HU planificadas para ser entregadas al cliente (18 HU), al final de la versión, no se alcance. Esto se refleja en la Figura 15 a través de las "HU aceptadas por el cliente", donde se puede observar que al final de la versión solamente 12 HU son aceptadas por el cliente.

7 Resultados

En el presente trabajo se ha construido un "Modelo de Simulación Dinámico de Gestión de Proyectos de Desarrollo de Software que utilizan XP", lo cual hace que se diferencie de otros trabajos en los que se modelan proyectos con Metodologías Tradicionales. De esta forma, el modelo puede ser usado como herramienta para analizar el efecto que tiene el uso de prácticas de XP en proyectos de software. En este sentido, es preciso mencionar que las prácticas que quedaron reflejadas en el modelo, de acuerdo Kent Beck en [17], son las siguientes: "El juego de la planificación", "Versiones pequeñas", "Hacer pruebas", "Recodificación", "Programación en parejas", "Integración continua" y "40 horas semanales", que son las que fueron posibles de representar utilizando Dinámica de Sistemas.

Para ofrecer una buena flexibilidad el modelo permite la alteración de valores durante su ejecución, como ser la cantidad de requerimientos del cliente, las fechas de comienzo y entrega de cada iteración, la cantidad de requerimientos a ser desarrollados en cada iteración, la cantidad de programadores, las horas extras agregadas por día, el porcentaje de tiempo destinado a cada etapa del desarrollo, el factor de carga del equipo de desarrollo, entre otras.

La validación del modelo realizada con los proyectos [14], [15] y [16] como casos de "entrenamiento" fue positiva ya que el mismo se comportó de acuerdo con los datos reales. Finalmente, luego de validar el modelo, ejecutarlo bajo diferentes escenarios y realizar el análisis de sensibilidad, se ha llegado a la conclusión de que el mismo cumple con sus objetivos y puede ser utilizado como herramienta para evaluar diferentes decisiones de gestión sobre proyectos de software desarrollados con XP.

8 Trabajos Futuros

Como trabajos futuros se plantea la posibilidad de lograr desarrollar un modelo íntegro que permita simular el desarrollo completo de un Proyecto, es decir, con todas sus versiones a la vez.

Por otro lado, una variante que podría ser incorporada al modelo es la "Planificación por alcance" dentro de la Metodología XP ya que el mismo fue construido basándose en la "Planificación por tiempo".

Además sería interesante la construcción de una jerarquía de Modelos Dinámicos para simular Proyectos llevados a cabo con Metodologías Ágiles. Dicha jerarquía de modelos podría estar inspirada en la presentada en [3] para Metodologías Tradicionales, la cual divide a los modelos en Básicos, Intermedios y Avanzados.

9 Referencias

1. Programación Extrema, http://www.programacionextrema.org
2. Abdel – Hamid T. K., Madnick S. E.; "Software Project Dynamics An Integrate Approach", Editorial Prentice –New Jersey (1991). ISBN: 0-13-822040-9 .
3. Ramos I., Toro M., Ruiz M.; "Modelo Dinámico Reducido", Informe Técnico: LSI-2001-01, Universidad de Sevilla (2001).
4. Lin C. Y.; "Walking on Battlefields: Tools for Strategic software management", American Programmer, Vol. 6, No. 5, pp. 33-40. (1993).
5. Smith B. J., Nguyen N., Vidale R.; "Death of a software manager: How to avoid career suicide through dynamic software process modeling", American Programmer, Vol. 6, No. 5, pp. 10-17. (1993).
6. Chichakly K.; "The Bifocal Vantage Point. Managing software projects from a systems thinking perspective", American Programmer, Vol. 6, No. 5, pp. 18-25. (1993).
7. Aranda R., Fiddaman T., Oliva R.; "Quality MicroWorlds: Modeling the impact of quality initiatives over the software product life cycle", American Programmer, Vol. 6, No. 5, pp.51-61. (1993).
8. Abdel - Hamid T. K.; "A Multiprojject Perspectiv of Single-Project Dynamics", Journal of Systems Software, Vol. 22, No. 3, pp. 151-166. (1993).
9. Putnam L. H., Myers W.; "Executive Briefing. Controlling Software Development". IEEE Computer Society Press (1996). ISBN: 9780818674525
10. Vensim, http://www.vensim.com/index.html
11. Aracil, J.; "Dinámica de Sistemas", Editorial Isdefe, España (1995). ISBN: 8-46 833802-8
12. Torrealdea J.; "Dinámica de Sistemas. Elementos y Estructuras de un Modelo. Construyendo Modelos". Ciencias de la Computación e Inteligencia Artificial. Universidad del País Vasco. País Vasco.
13. Kasiak T., Godoy D.; "Simulación de Proyectos de Software desarrollados con XP: Subsistema de Desarrollo de Tareas" XIV Workshop de Investigadores en Ciencias de la Computación 2012. pp. 572 - 576. ISBN 978-950-766-082-5. Argentina (2012).
14. Universidad Politécnica de Valencia: Departamento de Sistemas Informáticos y Computación, http://users.dsic.upv.es/asignaturas/facultad/lsi/ejemploxp/
15. ONess, http://oness.sourceforge.net/proyecto/html/index.html
16. Echeverry Tobón L. M., Delgado Carmona L. E.; "Caso Práctico de la Metodología Ágil XP al Desarrollo de Software", Universidad Tecnológica de Pereira: Facultad de Ingeniería. Colombia (2007).
17. Beck, K.; "Una Explicación de la Programación Extrema. Aceptar el Cambio", Editorial Addison Wesley, España (2002). ISBN: 8-47-829055-9

Anexo

Ecuaciones del Modelo

En la Tabla 1 que se presenta a continuación se detallan algunas de las ecuaciones de las variables de los subsistemas del modelo presentado en la Sección 4.

Tabla 1. Ecuaciones del Modelo

Nombre	Tipo	Unidad	Valor Inicial	Ecuación
Subsistema Recursos Humanos				
Diferencia en el numero de programadores deseados	Auxiliar	Programadores		IF THEN ELSE(Pares de programadores deseados(Time) <>0, (Pares de programadores deseados(Time)-Pares de Programadores)*2,0)
Pares de Programadores	Auxiliar	Pares		(Programadores expertos + Programadores novatos)/2
Flujo de contratación	Flujo	Programadores por día		IF THEN ELSE(Programadores a contratar<>0, Programadores a contratar/ Retraso medio en la contratacion, 0)
Flujo de adquisición de experiencia	Flujo	Programadores por día		Programadores novatos / Retraso medio en la adquisicion de experiencia
Programadores expertos	Nivel	Programadores	0	Flujo de adquisicion de experiencia - Flujo de abandono del proyecto
Subsistema Factor de Carga				
Diferencia de factor de carga de programadores expertos	Auxiliar	Dmnl		IF THEN ELSE (Actualizacion del factor de carga de programadores expertos (Time) <> 0, Actualizacion del factor de carga de programadores expertos(Time)-Factor de carga de programadores expertos,0)
Diferencia de factor de carga segun cantidad de programadores	Auxiliar	Dmnl		(((Factor de carga de programadores expertos* Programadores expertos)+ (Factor de carga de programadores novatos* Programadores novatos))/(Programadores expertos+Programadores novatos))- Factor de carga estimado del equipo por iteracion
Flujo de actualización del factor de carga segun cantidad de programado-	Flujo	Dmnl		IF THEN ELSE (Diferencia de factor de carga segun cantidad de programadores<>0, Diferencia de factor de carga segun cantidad de programadores,0)

res				
Factor de carga estimado del equipo por iteración	Nivel	Dmnl	2 (Factor de carga por defecto)	Flujo de actualización del factor de carga segun cantidad de programadores
Subsistema de Desarrollo de Pruebas de Unidad				
PU a desarrollar por iteración	Nivel	Pruebas de Unidad	0	Flujo de planificacion de PU por iteracion-Flujo de desarrollo de PU
Flujo de desarrollo de PU	Flujo	Pruebas de Unidad por día		IF THEN ELSE(PU a desarrollar por iteracion>0 :AND: (PU a desarrollar por iteracion-(1/Cantidad de dias promedio que lleva desarrollar una prueba de unidad)>0), 1/Cantidad de dias promedio que lleva desarrollar una prueba de unidad, IF THEN ELSE(PU a desarrollar por iteracion>0 :AND: (PU a desarrollar por iteracion-(1/Cantidad de dias promedio que lleva desarrollar una prueba de unidad)<0), PU a desarrollar por iteracion, 0))
Subsistema Presión en el Plazo				
Diferencia en la presion en el plazo	Auxiliar	Días		IF THEN ELSE(Tiempo de entrega pactado<>0 :AND: Tiempo real de entrega estimado por iteracion<>0,((Tiempo real de entrega estimado por iteracion)-(Tiempo de entrega pactado))-Presion en el plazo,0)
Tiempo de entrega pactado	Auxiliar	Días		IF THEN ELSE(Hu planificadas por iteracion<>0, Hu planificadas por iteracion*Cantidad de dias promedio que lleva desarrollar una HU, 0)
Tiempo real de entrega estimado por iteración	Auxiliar	Días		IF THEN ELSE(Hu planificadas por iteracion<>0,((Hu planificadas por iteracion*Cantidad de dias promedio que lleva desarrollar una HU)+(Tareas extras por iteración(Time) * Cantidad de dias promedio que lleva desarrollar una tarea)), 0)
Subsistema de Planificación				
Diferencia de HU a desarrollar en la version	Auxiliar	Historias de Usuario		IF THEN ELSE(Actualizacion de la cantidad de HU a desarrollar en la version(Time) <>0, Actualizacion de la cantidad de HU a desarrollar en la version(Time)-Tamaño de la version en HU,0)
Tamaño de la version en HU	Nivel	Historias de Usuario	Depende del proyecto que se simule.	Flujo de actualizacion del tamaño de la version
Cantidad de	Nivel	Historias de	Tamaño de	Flujo de actualizacion de las HU a

HU por desarrollar		Usuario	la version en HU	desarrollar+"Flujo de re-planificacion de HU no acabadas"+"Flujo de re-planificacion de HU con errores de aceptacion"-Flujo de planificacion de HU
Flujo de planificacion de HU	Flujo	Historias de Usuario por día		Planificacion de HU por iteracion(Time)
Subsistema de Desarrollo de Tareas				
Flujo de planificacion de tareas	Flujo	Tareas por Día		IF THEN ELSE(Tareas a desarrollar<0.5, Hu planificadas por iteracion*Cantidad promedio de tareas por HU,0)
Tareas a desarrollar	Nivel	Tareas	0	Flujo de planificacion de tareas-Flujo de codificacion de tareas-"Flujo de re-planificacion de tareas no acabadas"
Flujo de codificacion de tareas	Flujo	Tareas por Día		IF THEN ELSE(Planificacion de entregas(Time) =0,IF THEN ELSE(Tareas a desarrollar>0:AND: (Tareas a desarrollar-(1/Cantidad de dias promedio que lleva codificar una tarea)>=0), 1/Cantidad de dias promedio que lleva codificar una tarea, IF THEN ELSE(Tareas a desarrollar>0:AND: (Tareas a desarrollar-(1/Cantidad de dias promedio que lleva codificar una tarea)<0), Tareas a desarrollar, 0)),0)
Tareas en producción	Nivel	Tareas	0	Flujo de puesta en produccion de tareas