

Una Experiencia con la Estimación del Tamaño del Software

Marcela Varas
mvaras@inf.udec.cl

1. Introducción.

Este artículo presenta una experiencia en la estimación del tamaño de un proyecto software sobre la base de una especificación de requisitos considerada buena (no ambigua y bastante completa).

La experiencia se desarrolló dentro de la asignatura Gestión de Proyectos de Ingeniería de Software, un electivo de quinto año de la carrera de Ingeniería Informática en la Universidad de Concepción. Esta experiencia consistió en un simulacro de llamado a licitación para el desarrollo de un producto software para una organización ficticia.

Una de las tareas que los alumnos debieron ejecutar para responder a este llamado, fue determinar el tamaño del producto, estimar costos y plazos. La mayoría de ellos escogió la métrica puntos de función por sobre otras (puntos de característica, líneas de código) por las ventajas que esta métrica tiene.

Todos los alumnos manejaban la técnica de conteo de puntos de función, y utilizaron el mismo método (guía) para realizar el conteo. Todos los alumnos debieron entregar el detalle de las tareas desarrolladas como anexo a la propuesta, por lo que se pudo verificar la correcta aplicación del método para el caso de la estimación del tamaño del software a desarrollar.

Este artículo presenta una breve visión de la influencia de la determinación del tamaño en la planificación de un proyecto, además de una revisión a la importancia de las estimaciones en etapas tempranas del desarrollo. A continuación, se presentan los resultados obtenidos en la experiencia descrita anteriormente y una hipótesis explicativa de los resultados. Finalmente, las conclusiones y referencias bibliográficas.

2. El Tamaño y la Planificación.

Dentro de la gestión de un proyecto de desarrollo de software, la planificación es una actividad de gran importancia. Esta actividad involucra el establecimiento de objetivos y metas para un proyecto, y las estrategias, políticas y procedimientos para alcanzarlos.

Todo proyecto de ingeniería de software debe partir con un buen plan, pero lamentablemente, la planificación es una tarea nada de trivial. Uno de los aspectos que dificultan la labor de administradores y jefes de proyecto en torno a la planificación es la difícil tarea de realizar una estimación de costos y plazos realista.

El manejador de costo principal para un proyecto de desarrollo de software es sin duda el tamaño del producto. La medida del tamaño debe ser tal que esté en relación directa con el esfuerzo de desarrollo, por lo que las métricas de tamaño tratan de considerar todos los aspectos que influyen en el costo, como tecnología, tipos de recursos y complejidad.

En torno a esta problemática, se han propuesto diversas soluciones para disminuir la incertidumbre en la estimación del tamaño. Entre otras, se encuentra el uso de registros históricos de medidas del tamaño de la aplicación y esfuerzo requerido, el uso de distintos sistemas que aporten inteligencia al análisis de esta información, la estimación utilizando modelos matemáticos de la literatura, etc. (una propuesta y más antecedentes acerca de métricas de tamaño se puede encontrar en [Varas95](#)), pero

todas estas técnicas requieren que el tamaño del producto sea conocido.

3. Estimación Temprana del Tamaño del Software.

Para realizar la planificación de un proyecto software, es necesario poseer una estimación certera del esfuerzo necesario para el desarrollo lo más temprano posible, idealmente, con sólo la etapa de especificación de requisitos cubierta. De más está decir que sin la especificación de requisitos cualquier estimación será en la más completa incertidumbre, pues cómo estimar el costo de realizar algo que desconocemos?

A esta altura del desarrollo del software, difícilmente se puede realizar una estimación certera de la cantidad de líneas de código que tendrá la aplicación, ya que en este nivel no tiene por qué estar decidida la herramienta de desarrollo y la cantidad de líneas de código que serán necesarias para implementar una función del software dependen fuertemente del programador y la herramienta de desarrollo.

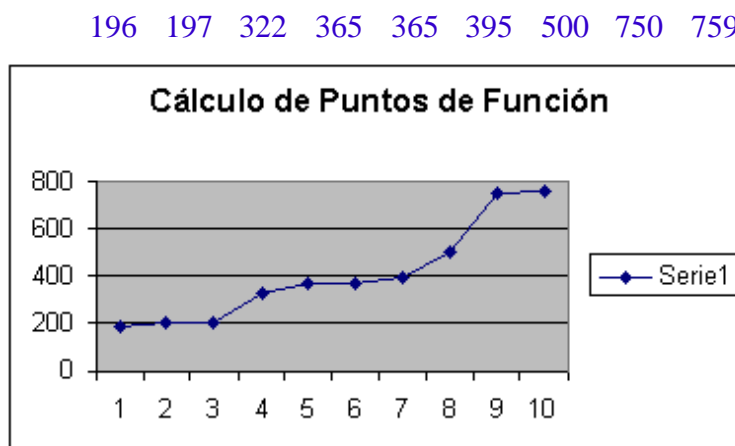
Por otra parte, los puntos de función pueden ser estimados con mayor certeza a partir de una buena especificación de requisitos (aunque no con toda la certeza deseada).

Lamentablemente, cuando los proyectos se plantean sobre la base de especificaciones vagas o inexistentes, es muy poco lo que se puede hacer.

4. Resultados de una Experiencia en la Determinación del Tamaño de un Software.

Para presentar una respuesta a un llamado a licitación, 13 alumnos de quinto año debieron determinar una buena estimación del tamaño del producto a desarrollar, eligiendo 10 de ellos la métrica puntos de función.

Los resultados obtenidos se muestran la tabla y gráficos siguientes. En ellos se muestra la medida en puntos de función obtenida por cada alumno, en un rango de 181 a 759.



La desviación estándar de estos datos es de aproximadamente 211 PF (Puntos de Función) sobre el promedio, lo que es sin duda muy alto. Esta gran diferencia condujo a estimaciones de esfuerzo y plazos drásticamente distintas. Esto influyó directamente en la cantidad de personal que cada alumno asignó al proyecto y a la organización del mismo (provocando impacto también en la función de staffing), además de la influencia directa en los costos y plazos. Durante el desarrollo del proyecto, las

diferencias en estas estimaciones provocarían grandes diferencias también en la dirección, por lo que estaría efectivamente afectando al proyecto en su conjunto, y por lo tanto al éxito o fracaso del mismo.

5. Explicando los Resultados.

Algo interesante e inesperado resultó este cálculo dispar entre candidatos a ingenieros civiles informáticos sobre la base de una misma especificación de requisitos y un mismo método de cálculo de puntos de función.

La asignación de complejidad que el método exige para cada ítem (entrada, salida, consulta, archivo lógico, archivo de interfaz), es un tanto subjetiva y normalmente difiere de un desarrollador a otro, pero esto no explicaría una desviación de tal magnitud (211PF).

La causa de estas diferencias de tamaño estimado está en el entendimiento de qué constituye efectivamente una salida externa, una entrada externa, un archivo lógico interno y una consulta.

La especificación del software que fue objeto de medición, está organizada por tipo de información. Los reportes y consultas están claramente especificados, mediante un diseño de formularios, pero no así las entradas. Sólo se especifican los datos que se deben ingresar.

De lo anterior, se puede deducir que el número de salidas y consultas estaba más bien claro, y que la causa de las grandes diferencias estuvo en las entradas y archivos lógicos internos.

Para el caso de las entradas, dado que no se especificaba cuál conjunto de datos sería ingresado simultáneamente, algunos alumnos los agruparon según su criterio, obteniendo un número de entradas cercano al número de consultas y salidas, pero otros consideraron que cada dato a ingresar constituía una entrada, por lo que el número de entradas aumentó en gran medida. A pesar de que la complejidad de las entradas que involucran a más de un dato es mayor que la de una entrada de un solo dato, esto no alcanzó a corregir la diferencia entre las medidas.

Para el caso de los archivos lógicos internos, algunos alumnos agruparon grandes conjuntos de información en un archivo lógico interno, mientras que otros modelaron la información en un esquema entidad relación, obteniendo una aproximación mucho más cercana a la cantidad de tablas que efectivamente implementarían. Evidentemente, el número de archivos lógicos internos fue mucho mayor en el segundo caso.

Otro factor de dificultad para realizar el cálculo del tamaño del software fue la existencia del requisito "se debe posibilitar la capacidad de realizar consultas no estructuradas". Este requisito fue tratado de maneras disímiles: fue dejado fuera, considerado al equivalente de 15 o más consultas complejas, o se decidió que se utilizaría una interfaz a alguna herramienta que proveyera dicha funcionalidad. Este punto también provocó diferencias.

Otro factor importante, que se da también en los casos reales (esta experiencia fue académica), fue la presión a la que estaban sujetos los participantes. Debían entregar una propuesta de calidad dentro de los plazos, o su castigo sería equivalente a no ganarse el proyecto: una baja calificación, o peor aún, reprobación de la asignatura. Por otro lado, al tener el conocimiento de que el ganador del proyecto no tendría que desarrollarlo, podían arriesgarse a hacer ofrecimientos interesantes y convenientes, pues no ponían en juego su prestigio. Este es un riesgo que normalmente no debieran correr las empresas proveedoras de servicios de desarrollo de software serias, las que debieran hacer estimaciones más bien conservadoras.

6. Conclusiones

Para poder planificar, la especificación de requisitos debe existir. Esto significa que, como en otras áreas de la ingeniería tradicional, los proyectos de ingeniería de software debieran comenzar con una etapa de análisis del problema (o ingeniería de sistemas), para poder definir bien cuál es la solución software a considerar.

Luego de tener especificada la solución, se puede realizar el plan del desarrollo, utilizando una métrica portable como lo es puntos de función.

Para realizar el cálculo de los puntos de función debe existir un consenso dentro de la organización para el tratamiento de los requisitos que no se ajustan al método, además de realizar un pre diseño para que este tamaño sea una buena estimación del tamaño real de la aplicación.

Por otra parte, se deben manejar aspectos como la re estimación del tamaño cada vez que se va teniendo mayor conocimiento del producto, esto es, una vez que se va avanzando dentro del desarrollo.

Además, esta experiencia demuestra que el método de cálculo por puntos de función es subjetivo y puede arrojar resultados diversos dependiendo de quién ejecute el método.

7. Bibliografía y Referencias.

- [Albrecht83] Allan J. Albrecht and John E. Gaffney, "Software Function, Lines of Code, and Development Effort Prediction: A Software Science Validation", IEEE Transactions on Software Engineering, vol SE-9, No 6, November 1983.
- [Matson94] Jack E. Matson, Bruce E. Barrett, and Joseph M. Mellichamp, "Software Development Cost Estimation using Function Points", IEEE Transactions on Software Engineering, vol 20, no 4, April 1994.
- [Boehm88] Barry W. Boehm and Philip N. Papaccio, "Understanding and Controlling Software Costs", IEEE Transactions on Software Engineering, vol. 14, no 10, October 1988.
- [Low90] Graham C. Low and D. Ross Jeffery, "Function Points in the Estimation and Evaluation of the Software Process", IEEE Transactions on Software Engineering, vol 16, no 1, January 1990.
- [Symons88] Charles R. Symons, "Function point Analysis: Difficulties and Improvements", IEEE Transactions on Software Engineering, vol 14, no1, January 1988.
- [Jeffery93] Jeffery, G. C. Low, and M. Barnes, "A Comparison of Function Point Counting Techniques", Transactions on Software Engineering, vol 19, no 5, May 1993.
- [Fenton94] Norman Fenton, "Software measurement: A Necessary Scientific basis", IEEE Transactions on Software Engineering, vol 20, no 3, March 1994.
- [Mukho92] Tridas Mukhopadhyay and Sunder Kekre, "Software effort Models for Early Estimation of Process Control Applications", Transactions on Software Engineering, vol 18, no 10, October 1992.
- [Boehm81] W. Boehm, "Software Engineering Economics", Englewood Cliffs, NJ: Prentice-Hall, 1981.
- [Pressman94] Roger S. Pressman, "Ingeniería del Software, Un Enfoque Practico", tercera Edición, McGraw-Hill editores, 1994.
- [Sommer92] Ian Sommerville, "Software Engineering", Fourth Edition, Addison-

Wesley Publishing Company, 1992.

[Jones86] Caper Jones, "Programming Productivity", McGraw-Hill, 1986

[Varas95] Marcela Varas C, "Modelo de Gestión de Proyectos Software: Estimación del Esfuerzo de Desarrollo", Encuentro Chileno de Computación, Arica 1995.