Modelos Automatizables de Estimación muy Temprana del Tiempo y Esfuerzo de Desarrollo de Software de Gestión

Pedro F. Salvetto
Universidad ORT Uruguay, Laboratorio de Investigación de Sistemas de Información,
Montevideo, Uruguay, Cuareim 1451, 11100
salvetto@ort.edu.uy
Juan C. Nogueira
Universidad ORT Uruguay, Laboratorio de Investigación de Sistemas de Información,
Montevideo, Uruguay, Cuareim 1451, 11100
nogueira@ort.edu.uy
Javier Segovia
Universidad Politécnica de Madrid
Departamento de Lenguajes Sistemas Informáticos e Ingeniería de Software
Campus de Montegancedo
28660 Boadilla del Monte (Madrid)
Facultad de Informática
fsegovia@fi.upm.es

Abstract

Contrary to most industrial production processes, software production processes generate intangible products and require intensive communication and coordination, which contributes to increase the risks and to complicate estimation. In spite of many years of research and development, formal and structured estimation (independent of expert judgment) of the time and effort required to develop a Management Information System (MIS) project remains an open problem The most extended estimation techniques at present time are supported by the premise - not so realistic - of stability of
requirements, and require human experts. Current models of estimation are based on metrics available in the early design phase. In this work we define an early MIS complexity metric and introduce formal models, ready to be automated, for estimating time and effort of MIS development projects. These models use as input parameters the development team efficiency, the requirements volatility, the speed of development and the complexity of the system to be developed. The complexity is measured automatically from the user's data views of the system,
with independence of the technology used. These models are applicable continuously during the project, from the very beginning at the requirements engineering phase up to later stages, and do not deny but assume the inevitable changes in requirements, supporting their management.

Keywords: software complexity metrics, early time and effort estimation, data and knowledge oriented development, software measure automation, empirical models

Resumen

A diferencia de los procesos de producción industrial, los procesos de producción de software generan productos intangibles y requieren comunicación y coordinación intensivas lo que contribuye a aumentar los riesgos y dificultar la estimación.
A pesar de largos años de investigación y desarrollo el problema de la estimación formal y estructurada (independiente del juicio experto) del tiempo y esfuerzo requeridos para desarrollar un Sistema de Gestión (SG) permanece abierto. Las técnicas de estimación más extendidas actualmente se apoyan en la premisa - poco realista - de estabilidad de requerimientos, y requieren expertos humanos. Los modelos de estimación actuales, se basan en métricas disponibles recién en la fase de diseño temprano del sistema.
En este trabajo se define una métrica temprana de complejidad de un SG y se presentan modelos formales (automatizables) de estimación del tiempo y esfuerzo de desarrollo de sistemas de información. Estos modelos emplean como parámetros de entrada la eficiencia del grupo de desarrollo, la volatilidad de los requerimientos, la velocidad de desarrollo y la complejidad del sistema a desarrollar. La complejidad es medida **automáticamente a partir de las vistas de datos de usuario del sistema con independencia de la tecnología** a **utilizar.** Estos modelos son aplicables continua y muy tempranamente desde la etapa de ingeniería de requerimientos y no desconocen los **inevitables** cambios en los requerimientos, sino que los asumen y apoyan su gestión.
Palabras clave: métricas de complejidad del software, estimación temprana de tiempo y esfuerzo, desarrollo orientado a datos y conocimiento, automatización de la medición del software, modelos empíricos

1 INTRODUCCIÓN

1.1 Descripción del problema

A diferencia de los procesos de producción industrial, los procesos de producción de software generan productos intangibles y requieren comunicación y coordinación intensivas lo que contribuye a aumentar los riesgos y dificultar la estimación.

A pesar de la existencia de modelos de estimación como COCOMO [5], COCOMO II [4] , su variante en desarrollo COSYSMO [25] y SLIM [21] entre otros, es muy común que los presupuestos y los tiempos de desarrollo sean subestimados [3][4][7][20].Estos modelos se caracterizan por incluir numerosos parámetros que deben ser estimados mediante juicio experto y parecen funcionar mucho mejor en manos de sus creadores, de expertos altamente entrenados en su uso y/o de quienes disponen de bases de datos con información histórica para calibrarlos. Por requerir juicio experto es posible que, observando un mismo escenario, distintos evaluadores obtengan estimaciones significativamente diferentes. En contraste con los avances en herramientas CASE, la gerencia de proyectos de software no ha avanzado sustancialmente. Para contribuir a la disciplina es necesario investigar el problema de la gerencia con un enfoque formal [19].

Algunas de las características del desarrollo de software que contribuyen a explicar la dificultad para construir modelos de estimación son a) el desarrollo de software requiere trabajo intelectual e interacción humana; b) a pesar del nivel repetible del CMM propuesto por el SEI, los procesos de desarrollo de software repetibles son raros; c) los proyectos de desarrollo de software pueden ser replicados, pero no repetidos; d) la tecnología cambia con tanta rapidez que es difícil, e incluso propenso a errores, incorporar la experiencia pasada ya que podemos estar evaluando una situación a partir de premisas no aplicables a ella.

A pesar de largos años de investigación y desarrollo el problema de la estimación formal y estructurada (independiente del juicio experto) del tiempo y esfuerzo requeridos para desarrollar un sistema de gestión permanece abierto. Las técnicas de estimación más extendidas actualmente se apoyan en la premisa - poco realista - de estabilidad de requerimientos, y requieren expertos humanos. Los modelos de estimación actuales, se basan en métricas disponibles recién en la fase de diseño temprano del sistema. Los gerentes deben elegir entre tomar decisiones tempranas bajo alta incertidumbre, o mitigarla pagando con tiempo y encareciendo el proyecto [19].

Putnam y Myers [20] definen 5 métricas medulares para el éxito de la gestión de proyectos de software a) cantidad de funcionalidad, b) productividad, c) tiempo, d) esfuerzo y e) confiabilidad. Las primeras cuatro corresponden al proceso de desarrollo y la quinta a la explotación del sistema y se encuentra fuera del alcance de este trabajo.

Nuestra investigación apunta a desarrollar modelos formales (automatizables) de estimación del tiempo y esfuerzo de desarrollo de sistemas de información, basados en bases de datos relacionales, con procesos evolutivos y ágiles y generación automática de código a partir de especificaciones formales[1].

Estos modelos emplean como parámetros de entrada la eficiencia del grupo de desarrollo, la volatilidad de los requerimientos, la rapidez de ejecución del projecto y la complejidad del sistema a desarrollar **medida con independencia de la tecnología** a utilizar.

En este trabajo se presenta un análisis causal de los factores relacionados con la complejidad cognitiva de un SG y el tiempo y esfuerzo requeridos para su desarrollo. Sobre la base de este análisis se discuten métricas de complejidad cognitiva para SG que consideran las contribuciones que a la misma realizan los datos y los procesos, independientemente de la tecnología usada para su desarrollo y se presentan modelos de estimación de tiempo y esfuerzo de desarrollo en base a elementos disponibles en etapas muy tempranas del proceso. Las métricas usadas son calculadas a partir de información obtenida automáticamente desde las especificaciones del sistema a construir. Para ello se usó una herramienta de recolección automática de métricas que fue explicada en un trabajo previo[2] [22].

2 TRABAJOS PREVIOS

Es bien conocido el trabajo de Albrecht [1] sobre puntos funcionales y todas sus evoluciones y variantes hasta llegar al Standard ISO/IEC 19761:2003 [9].

Grompone [14] propuso una herramienta para apoyar el cálculo de los puntos funcionales y, a partir de los puntos funcionales del sistema a construir, una curva empírica para estimación de esfuerzo de proyectos desarrollados con la herramienta de especificación formal usada para desarrollar los proyectos que hemos relevado.

---

[1] Nuestro criterio respecto de la formalidad de la especificación no está restringido al rigor matemático de los lenguajes de especificación formal. Consideramos que si la especificación contiene información suficiente para generar automáticamente la aplicación en cualquier lenguaje y plataforma (siempre que esté disponible el generador correspondiente) y esta especificación es publicada de forma que pueda automatizarse su procesamiento, entonces es una especificación formal. Seguimos el pensamiento de Briand [8] en el sentido de que en el estado del arte actual de la ingeniería de software, apegarse a un formalismo exagerado es cerrar las puertas a la adquisición de nuevos conocimientos.

[2] Esta herramienta fue desarrollada por grupos de estudiantes que realizaban su trabajo final de grado en el Laboratorio de Investigación en Sistemas de Información de la Universidad ORT Uruguay con el apoyo del Departamento de Informática del Ministerio de Transporte y Obras Públicas de Uruguay.

Chalar, Dávila y Berriel [12] encontraron una notable disparidad de criterios para evaluar la complejidad y dificultad de construcción de los distintos objetos que componen un sistema en una encuesta realizada entre expertos y propusieron investigar unidades que fueran útiles como patrón de medida de tamaño de aplicaciones.
Sin embargo todas estas propuestas presentan la misma debilidad: **no son automatizables y dependen del juicio experto**. Putnam [21], Boehm[4, 5] y en general todos los autores que han desarrollado y publicado modelos de estimación de tiempo y esfuerzo de desarrollo, encontraron que existe una relación exponencial entre las variables de entrada y el esfuerzo o tiempo [14].
Salvetto y Nogueira [22] presentaron una herramienta de recolección automática de métricas a partir de especificaciones formales que fue usada para la recolección de datos para este trabajo y definieron la métrica KBIS (Knowledge Base Intelectual Size).
Shao y Wang [24], propusieron una interesante métrica de complejidad cognitiva del software, basada en las estructuras de control y su anidamiento. Latorres y Salvetto [17] desarrollaron en el Ministerio de Transporte y Obras Públicas de Uruguay un IDE de ciclo completo que permite tomar automáticamente y en tiempo real las métricas del Personal Software Process (PSP) [15,16] y relacionar los trabajos con requerimientos posibilitando la medición directa de la volatilidad de los mismos y la cuantificación de su influencia en el esfuerzo y tiempo de desarrollo. No obstante, todavía no existe una muestra importante de proyectos desarrollados con esta herramienta por lo que en este trabajo la volatilidad de los requerimientos ha sido estimada por los gerentes de los proyectos y no medida directamente Bennet [2] publicó un modelo de estimación temprana del esfuerzo de desarrollo de sistemas siguiendo la metodología del DoD tomando métricas desde diagramas idef0 e idef1x. En su trabajo, destaca que no pudo encontrar ninguna correlación lineal entre las variables de entrada y tuvo que recurrir a normalizarlas mediante logaritmos. Esto es coincidente con lo encontrado por nosotros durante el análisis exploratorio de datos y la mayoría de los modelos de estimación que muestran relaciones funcionales exponenciales.

3 NUESTRAS HIPÓTESIS DE TRABAJO

1) Si sólo se desarrolla funcionalidad no redundante y de valor para el negocio o para algún usuario**, la complejidad esencial del sistema a desarrollar queda determinada por la integración de las visiones de datos de los usuarios del sistema y esta puede medirse sobre la base del esquema de bases de datos relacionales generado a partir de la integración de las visiones de datos de los usuarios.**
2) El esquema de base de datos relacional así obtenido contiene información suficiente como para ser usada como entrada de modelos de estimación de tiempo y esfuerzo de desarrollo.
3) Aunque no comprendamos en profundidad los complejos procesos internos e interacciones que ocurren durante el proceso de desarrollo, podemos observarlo y construir modelos que estimen aceptablemente su resultado global. En particular, si el desarrollo se realiza usando herramientas que automaticen la generación del código, la integración de módulos y apoyen el trabajo de grupos reducidos con usuarios integrados a los mismos usando una metodología estándar, se estará reduciendo las fuentes de variabilidad y será posible encontrar modelos útiles para la estimación temprana.

4 CARACTERÍSTICAS DE LA MUESTRA

Los proyectos fueron desarrollados con metodologías ágiles, ciclo de vida evolutivo y herramientas de especificación formal. Se trabajó con equipos de desarrollo reducidos (2 a 5 personas) con usuarios integrados a los mismos. Todos los sistemas fueron desarrollados a partir de las vistas de datos de los usuarios y sus solicitudes. Se desarrollaron utilizando tecnología de bases de datos relacionales. Los sistemas estudiados son Sistemas de Información del tipo data-strong [13] donde los algoritmos de alta complejidad o no existen o son excepcionales En todos los casos, las métricas de entrada de nuestros modelos fueron tomadas en forma automática, con la herramienta de que disponemos, a partir de las especificaciones.
Las únicas excepciones fueron la volatilidad de los requerimientos y el tiempo y esfuerzo requeridos para el desarrollo del proyecto que fueron estimados por los gerentes de proyecto. En relación a estas dos últimas variables, se dispuso de 8 proyectos, muy bien medidos ya que existían registros detallados de horas de trabajo. La duración y esfuerzo de los proyectos restantes fueron estimadas por los gerentes en el transcurso de entrevistas en las que se completaron formularios describiendo las características de los mismos. Puede consultarse los datos relevados en la tabla 4.

La herramienta de especificación formal usada para el desarrollo tiene la capacidad de automatizar[3] el trabajo con los usuarios en el ámbito de sus vistas de datos no normalizadas de la realidad, la integración de estas vistas de datos de usuario, la generación del esquema relacional que representa la integración de las vistas de datos de usuario, la determinación del impacto de un cambio en el esquema relacional, la generación de los programas necesarios para migrar los datos cuando se produce un cambio en las vistas de datos de usuario que impacta en el esquema relacional, la generación del código en diversos lenguajes y plataformas a partir de la especificación (independizándonos por tanto de la tecnología), la integración de varias especificaciones en una sola, detectando las posibles inconsistencias (que obviamente tendrán que levantar los desarrolladores), la publicación de la

[3] La herramienta utilizada en esta investigación fue GeneXus de Artech, www.artech.com

especificación posibilitando la automatización de la obtención de métricas y el manejo de un nombre único para cada atributo.
Se analizaron proyectos desarrollados con una misma metodología de trabajo estable y equipos de desarrollo de similares características, por lo que **la eficiencia es constante dentro de la muestra**.

5 LA METODOLOGÍA Y LA HERRAMIENTA DE ESPECIFICACIÓN

La herramienta de especificación genera el diseño de la base de datos relacional, el código para la creación de la base de datos y los programas aplicativos requeridos por el sistema de información en construcción a partir de su especificación formal[4]. También es posible exportar la especificación por medio de un archivo XML y acceder a los detalles de la especificación en forma tabular.
La herramienta almacena información en una base de conocimiento (KB). En este artículo, siguiendo la terminología estándar de Genexus, KB (Knowledge Base) refiere a un conjunto de objetos que contiene la especificación del sistema en un lenguaje propietario. Estos datos son de gran utilidad para capturar métricas automáticamente.
La metodología asociada a la herramienta se apoya en dos premisas. Primero en una organización mediana o grande, **nadie tiene una visión global de los datos y los procesos**. Por tanto, se requiere integrar las visiones de diferentes usuarios. Segundo, **los requerimientos y las estructuras de bases de datos sufren cambios continuos e inevitables a lo largo del tiempo.**
La especificación permite modelar el sistema, generar la base de datos en tercera forma normal y el código para la aplicación en el lenguaje y plataforma de ejecución deseados. Cuando se produce un cambio en los requerimientos, se ajusta la especificación y la herramienta genera el nuevo esquema de base de datos, reorganiza la base de datos y genera el código requerido.
La especificación usa diferentes tipos de objetos: vistas de datos de usuarios, procedimientos, reportes, paneles de trabajo, y menús.
La construcción de estos objetos se basa en atributos, reglas, eventos, subrutinas y programas externos.
Las Vistas de datos de Usuario modelan las entidades del mundo real según las perciben los usuarios finales. La herramienta diseña y genera la base de datos a partir de estos objetos de especificación. Este es un factor muy importante para nuestra investigación, ya que al generarse el esquema relacional automáticamente a partir de las vistas de usuario y siguiendo un modelo matemático, podemos medir la complejidad esencial directamente a partir de él sin que se produzcan desviaciones atribuibles a estilos de trabajo en el diseño de bases de datos por parte de los desarrolladores o a la forma como los usuarios ven el mundo
Los **Procedimientos** modelan procesos por lotes para gestión de las bases de datos.
Los **Reportes** permiten la recuperación de información.
Los **Paneles de trabajo** representan consultas interactivas a la base de datos. Finalmente, los **menús** organizan los objetos anteriores. Puede encontrarse mayor información sobre esta herramienta y nuestra línea de investigación principal en [22] .

5.1 Publicación de la especificación

La herramienta publica los detalles de la especificación en dos sabores: textual en XML y tabular por medio de un OLEDB provider. El primero apunta a exportar la KB y el segundo permite a los programadores acceder a los metadatos y extender y adecuar la herramienta a sus necesidades. El OLEDB provider expone los artefactos de especificación y sus relaciones. Esta facilidad fue un factor decisivo en la elección del tipo de herramienta.

5.2 Acceso a las fuentes de datos sobre la complejidad del sistema

Hay tres fuentes de medición de la complejidad, todas ellas derivadas de la especificación del sistema sin intervención humana y por lo tanto no sesgadas y adecuadas para ser recolectadas automáticamente e incorporadas a un ambiente integrado de desarrollo: a) artefactos de especificación y sus relaciones b) especificaciones textuales del sistema c) código fuente generado.
Estas fuentes de información se encuentran disponibles en todas las etapas del ciclo de vida. Obviamente la generación de código tiene algunas limitaciones en etapas tempranas.
Las dos primeras dependen solamente de la especificación, en cambio la tercera es dependiente del lenguaje generado.
Tratando de aprovechar la facilidad de acceso a los artefactos de especificación y sus relaciones proporcionada por el OLEDB provider, las métricas medulares de nuestra investigación fueron calculadas a partir de la exposición tabular de los artefactos de especificación y sus relaciones.

---

[4] Puede obtenerse más información sobre GeneXus en http://genexus.com. Trabaja con especificaciones formales en el sentido que a partir de ellas puede generar la aplicación en cualquier lenguaje y plataforma si se dispone del generador correspondiente.

6 DETERMINACIÓN DE MÉTRICAS CANDIDATAS.

6.1 Preguntas de investigación de este artículo

En este artículo apuntamos a responder las siguientes preguntas de investigación:

- ¿Cuáles son las métricas tempranas que permiten predecir tiempo y esfuerzo de desarrollo?
- ¿Es posible considerar una métrica de complejidad esencial de un sistema de información independiente de la tecnología usada para su implantación?
- ¿Se puede generalizar esta métrica a sistemas desarrollados con otras herramientas, especialmente cuando estas no tienen la capacidad de generación automática del esquema a partir de las vistas de datos de usuario?

6.2 Características de las métricas buscadas

El uso de especificaciones formales es vital porque: a) nos permite usar las mismas hipótesis que [19]; b) posibilita una recolección de métricas no ambigua y automática; c) permite la medida temprana de la complejidad basada en los requerimientos; y d) posibilita una medición post mortem objetiva y automática del tamaño de un proyecto midiendo la complejidad total de la especificación final.

Antes de seleccionar las métricas, establecimos las características que debían tener que incluyen aspectos prácticos, precisión, disponibilidad, factibilidad de automatización, resistencia del personal y significación para el usuario.

A continuación presentamos un resumen de las principales características encontradas:

Disponibilidad temprana: necesitamos métricas disponibles en estados muy tempranos del proceso de desarrollo.

Tardías: también necesitamos métricas tardías porque usamos métricas post mortem para encontrar correlaciones con métricas tempranas. Los modelos post-mortem no se presentan en este artículo por limitaciones de espacio. Pueden encontrarse en [23].

Repetibles: necesitamos que diferentes observadores arriben al mismo resultado.

Recolectables automáticamente: la recolección de métricas manuales es muy pesada, consume tiempo, está sujeta a desviaciones y es inexacta.

Fácil de calcular: queremos métricas calculables con algoritmos simples.

No invasivas: la recolección de métricas no debe interferir con el proceso de desarrollo.

Claro significado: los usuarios deben entender el significado de la métrica.

Simplicidad: preferimos métricas con el menor número de parámetros siguiendo el principio de la navaja de Occam.

6.3 Independencia de la Tecnología

Buscamos una medida de complejidad de una base de conocimiento, que debe ser independiente de la tecnología usada. Esta independencia es consecuencia inmediata del hecho de que, a partir de la especificación formal, es posible generar la aplicación en diferentes plataformas y lenguajes.

6.4 Complejidad

Necesitamos medidas objetivas, repetibles y automáticas para prescindir de las interpretaciones de los expertos que pueden ser sesgadas [19][21].

A partir de las vistas de datos de usuario de un sistema queda determinado el esquema de bases de datos relacionales que representa la información esencial contenida en ellas, con independencia de cuales sean las vistas de datos de usuario y sus posibles intersecciones. Este esquema es generado automáticamente por la herramienta usada para el desarrollo del sistema y el relevamiento.

6.5 Métricas de complejidad derivadas de la especificación

Deseamos una métrica que esté relacionada con el esfuerzo intelectual y cognitivo requerido para construir el producto. Los modelos post-mortem no se presentan en este artículo por limitaciones de espacio [23].

Necesitamos medir la complejidad cognitiva del producto. La métrica debe depender directamente de los requerimientos funcionales en lugar de los detalles de implementación que son resueltos por el generador de código. Las distintas visiones del sistema que tienen los usuarios finales son, en cierto sentido, derivadas de detalles de implementación originados en la estructura organizacional. Por este motivo, no las tomamos en cuenta y nos basamos en el esquema automáticamente generado a partir de ellas. Por esa razón usamos algunos de los artefactos provistos en la especificación formal independientes del generador y de las visiones de los usuarios que representan problemas esenciales en el desarrollo de un sistema de información.

6.6 Métricas de complejidad a partir de los objetos de la especificación

Las **Visiones de Usuario (UV)** son los principales contribuyentes a la complejidad y aparecen durante **etapas muy tempranas de la fase de análisis de requerimientos.** Representan las visiones de diferentes usuarios a partir de las cuales la herramienta genera la base de datos. Durante algún tiempo pensamos que a partir de ellas se podría construir modelos como los que estamos buscando. Un análisis más profundo nos mostró que las UV son portadoras

de la complejidad esencial del sistema pero que esta debe ser medida indirectamente desde el esquema relacional generado a partir de ellas ya que el mismo sistema puede ser visto de diferente forma por distintos usuarios.

6.6.1 Métricas de complejidad con un enfoque cognitivo

La especificación del sistema representa su complejidad esencial, ya que a partir de ella es posible generar la aplicación en distintos lenguajes y plataformas.

Los sistemas que estamos estudiando son del tipo data-strong según DeMarco [13]. De Marco propone que estos sistemas sean estimados en base a los datos.

Existen dos fuentes principales de complejidad cognitiva los datos (Data Structure´s Cognitive Complexity) y las funciones y procesos (Process Structure´s Cognitive Complexity).

Los datos contribuyen a la complejidad mediante su estructura. Calero, Piattini, et al [11] definen métricas para evaluación de complejidad de bases de datos relacionales que clasifican en (1) **table oriented metrics** Number of unique Attributes (NA) y Referential Degree (RD) y (2) **schema oriented metrics**, Depth Referencial Tree (DRT) y Normality Ratio. El RD es el número de claves foráneas en el esquema. Nosotros agregamos el número de tablas (NT).

El DRT es la longitud del mayor camino referencial en el esquema. Los ciclos sólo se consideran una vez. La proporción de normalidad es el número de tablas en tercera forma normal dividido por el número total de tablas en el esquema. Esta métrica no es útil para nosotros ya que el esquema es generado automáticamente en tercera forma normal por lo que su valor siempre es 1. Calero, Piattini el al [11] estudian las propiedades formales que cumplen estas métricas y plantean que se debe seguir investigando su validez empírica. En este trabajo, mediante regresión lineal de las variables antes mencionadas normalizadas con transformación logarítmica, se estudian estas dependencias. Shao y Wang [24] proponen una métrica de complejidad basada en pesos cognitivos, útil para medir la contribución de la PSCC a la KBCC. Sin embargo, esta métrica se basa en las estructuras de control, elementos que se encuentran disponibles en etapas finales del proceso de desarrollo, por lo que serán consideradas con la finalidad de construir modelos post mortem para investigar la contribución de los elementos tardíos a la complejidad y poder identificar (en trabajos posteriores) un patrón de peso de una base de conocimiento, que permita gestionar los hitos y contratos de un proyecto a partir de la funcionalidad entregada y el avance en el desarrollo del proyecto.

7 MODELOS DE PREDICCIÓN DEL TIEMPO Y ESFUERZO DE DESARROLLO

Si bien, nuestro relevamiento es sobre proyectos post-mortem[5], deseamos estudiar la posibilidad de construir modelos predictores basados en elementos presentes en etapas muy tempranas del proceso. Por este motivo, no se considerarán todas las métricas disponibles. Identificamos dos fuentes principales de complejidad, una proveniente de los procesos (Process´Structure Cognitive Complexity) y otra de los datos (Data Structure´Cognitve Complexity). El PSCC no es tomado en cuenta en este trabajo porque durante el análisis exploratorio de los datos se constató que su contribución a la explicación del esfuerzo y el tiempo es poco significativa y además, se basa en métricas post-mortem. Como el esquema de datos deriva automáticamente de las vistas de datos de los usuarios y además no existen algoritmos de alta complejidad ni se han realizado trabajos que no tengan valor para el usuario, el DSCC es un elemento muy significativo para medir la complejidad esencial del sistema y explicar el esfuerzo y tiempo invertidos en su desarrollo. La volatilidad de los requerimientos, tiene una importancia muy grande porque si es muy alta puede poner el proyecto fuera de control, pero además, como el relevamiento realizado fue sobre proyectos post-mortem, contribuye a la explicación del trabajo total realizado, del cual no quedan evidencias en la KB final. En este relevamiento, la volatilidad de los requerimientos (RV) fue estimada por los gerentes de los proyectos. El esfuerzo lo medimos en meses hombre considerando los desarrolladores y los usuarios integrados al equipo de desarrollo. No se tomaron en cuenta tareas de apoyo realizadas por los usuarios como obtención de datos de prueba, datos faltantes en sistemas anteriores, etc. El tiempo fue medido en meses. Dos factores que influyen sobre el tiempo y esfuerzo son la rapidez de ejecución del proyecto y la volatilidad de los requerimientos. La rapidez de ejecución fue evaluada como esfuerzo por tiempo. Como se desea usar métricas presentes tempranamente en el ciclo de vida se utilizó esfuerzo medio del 10% inicial de ejecución del proyecto (IME). La volatilidad de los requerimientos (RV) se estimó como un valor porcentual de acuerdo a la fórmula de Nogueira [19]

$$RV=\frac{DR+BR}{NR}*100$$

Donde DR= número de requerimientos descartados (dead), BR=número de requerimientos nacidos (born) y NR = número total de requerimientos. Como ninguno de los proyectos disponía de este registro, durante las entrevistas se solicitó a los gerentes de los proyectos que la estimaran en <10% baja, 10-20% media y mayor al 20% alta, asignándose los valores 10, 20 y 30 respectivamente

Durante el análisis exploratorio de los datos no se encontraron correlaciones lineales importantes entre las variables estudiadas. Sólo se pudieron encontrar correlaciones lineales entre los logaritmos de dichas variables, lo que es coincidente con lo encontrado por otros autores [2],[4],[5],[14],[21].

---

[5] En la tabla 4 se presentan los datos relevados.

7.1 Construcción validación y Evaluación de los modelos

Los modelos se construyeron mediante regresión lineal sobre el 70% de la muestra elegido aleatoriamente y se validaron estudiando su valor predictivo para el total de la muestra. Se evaluó el desempeño estadístico de los modelos mediante el coeficiente de determinación (R^2), la precisión y consistencia se midió mediante el valor promedio del error relativo (MRE) y la capacidad de predicción a un determinado nivel k (PRED(k)) definidas por Conte et al [10].

El error relativo (RE) , el MRE (medium relative error) y PRED(K) se calculan

$$RE = \frac{\text{Valor Observado-Valor Estimado}}{\text{Valor Observado}} \qquad MRE = \frac{\sum |RE|}{N} \qquad \text{PRED(K)} = \frac{\text{Número de Predicciones con } |RE|<K}{\text{Número total de observaciones}}$$

En todos los casos las variables dependientes e independientes se normalizaron tomando logaritmos, pero los errores se estudiaron luego de deshacer el cambio de variable, es decir que se estudió el verdadero valor predictivo de los modelos respecto de las variables que se desea estimar y no de su logaritmo que podría ser mucho mayor. Luego los resultados se evaluaron de acuerdo a la siguiente tabla tomada de [2].

Calificación	Consistencia	Precisión
Excelente	**MRE ≤ 0.20**	**PRED(20) ≥ 0.80**
Bueno	**MRE ≤0.25**	**PRED(25) ≥0.75**
Aceptable	**MRE ≤0.30**	**PRED(30) ≥0.70**
Pobre	**MRE > 0.30**	**PRED(30) < 0.70**

Tabla 1 Escalas de calidad de modelos de estimación[2]

7.2 Modelos

7.2.1 Variables Independientes

Se construyeron **modelos tempranos** en los cuales las variables independientes fueron DRT, RD, NA, NT, IME y RV. A partir de esos modelos se definió una nueva variable independiente que es una función de DRT, RD, NA y NT que representa el peso de la complejidad cognitiva de la estructura de datos que denominamos DCXW. Luego construimos modelos a partir de esta métrica e IME y RV para investigar nuestra hipótesis de que **la estructura de los datos contiene información suficiente para definir la complejidad esencial del sistema a construir**.

7.2.2 Modelos de estimación muy temprana del tiempo

Aplicando regresión lineal sobre los logaritmos de DRT, RD, NA, NT, IME y RV, se construyó un modelo predictor del logaritmo de TIME y se llegó a la siguiente expresión:

$$\text{Ln(TIME)} = \alpha \text{Ln(DRT)} + \beta \text{Ln(RD)} + \chi \text{Ln(NA)} + \delta \text{Ln(NT)} + \gamma \text{Ln(IME)} + \lambda \text{Ln(RV)} - 4{,}252472984$$

Deshaciendo el cambio de variable

$$\text{TIME} = 0{,}014 \text{DRT}^{\alpha} \text{RD}^{\beta} \text{NA}^{\chi} \text{NT}^{\delta} \text{IME}^{\gamma} \text{RV}^{\lambda} \qquad \text{Very Early Time Estimation Model 1 (VETEM1)}$$

Donde

α = -2,41, β=- 0,33, χ = 1,82, δ=-0,14, γ=- 0,23 y λ= 0,32

Los factores DRT, RD, NA y NT representan la complejidad esencial del sistema, mientras que los restantes el efecto de la velocidad de ejecución del proyecto, la volatilidad de los requerimientos y la constante factores ambientales y del equipo de desarrollo.

Este modelo tiene R^2= 0,9698, MRE=0,0729, PRED(5)=0,5, PRED(10)=0,8, PRED(15)=0,9, PRED(20)=0,95 y PRED(25)=1 lo que muestra que predice con un error menor al 25% en todos los casos observados y tiene calificación excelente según los criterios de la tabla 1. Teniendo en cuenta la precisión con que han sido estimados los datos de entrada y a los efectos prácticos consideraremos los valores de los coeficientes de regresión redondeados a dos dígitos decimales.

Basándonos en este modelo definimos el peso de la complejidad cognitiva de los datos DCXW

$$\text{DCXW} = \text{DRT}^{-2{,}4}\ \text{RD}^{-0{,}3} \text{NA}^{1{,}8} \text{NT}^{-0{,}15}$$

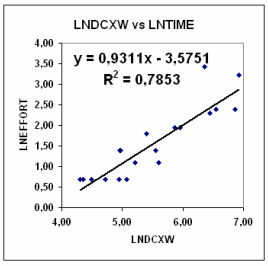
A partir de esta nueva variable se obtuvo el siguiente modelo.

$$\text{TIME} = 0{,}02\ \text{DCXW}^{1{,}08}\ \text{IME}^{-0{,}7} \text{RV}^{0{,}15} \qquad \text{Very Early Time Estimation Model 2 (VETEM2)}$$

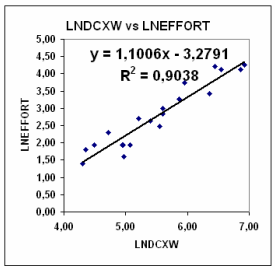
7.2.3 Resumen de modelos de estimación temprana de tiempo de desarrollo

En las Fig 1 y 2 se grafican los valores resultantes de los modelos vs los observados. Se presentan la recta y=x y la recta de tendencia del modelo así como el R^2 de la recta. También se grafican los errores relativos cometidos con la finalidad de apreciar si se trata de modelos optimistas o pesimistas y los valores absolutos de los errores relativos para evaluar los errores relativos máximos que se puede estar cometiendo. Se grafica la frecuencia relativa acumulada que representa la cantidad de observaciones donde se esta cometiendo un error <=k. Esta frecuencia, para el caso de los errores relativos tomados en valor absoluto coincide con PRED(K). En la Fig 3 se grafican los residuos.

En la tabla 2 se resumen los errores en valor absoluto mostrando la distribución de las observaciones que caen en cada rango, se presenta el valor de PRED(K) y se incluyen medidas de posicionamiento y dispersión tanto para los errores relativos como para sus valores absolutos. Se observa que el primer modelo permite realizar predicciones con un error relativo menor al 25% en el peor caso. Puede considerarse que el modelo es pesimista ya que sobreestima con mayor frecuencia de lo que subestima (Fig 3 y 4). En cambio, el segundo modelo si bien tiene una dispersión mayor parece balancear mejor las sub y sobreestimaciones. El primero califica como excelente y el otro estaría en una región intermedia entre excelente y bueno ya que el MRE es adecuado a la categoría de excelente, pero el PRED(20) es 0,75 en lugar del 0,8 requerido por los criterios de la Tabla 1. Si bien el modelo 2 tiene un error porcentual máximo mayor, no debemos olvidar que estos son errores relativos. Observando las gráficas notamos que se comporta mejor en los proyectos que insumieron más tiempo, en cambio el modelo 1 parece más adecuado para proyectos de menor duración. Se identificó una métrica de complejidad de la estructura de los datos. En las siguientes gráficas se muestra la correlación que guarda con tiempo y esfuerzo.



RE Min=2%,Max=70% MRE=30%



RE Min=3%,Max=79% MRE=25%

Se observa que la correlación es entre los logaritmos lo que coincide con lo encontrado por otros autores y por nosotros mismos. Esta correlación es más marcada para el esfuerzo. Esto no resulta sorprendente, ya que el proyecto puede ser ejecutado con diferente rapidez. Los modelos que surgen de la regresión sólo sobre DCXW, tienen los errores y R^2 que se muestran en las gráficas. Esto parece confirmar nuestras hipótesis de trabajo. Los coeficientes de correlación son 0,89 para time y 0,95 para effort.

7.2.4 Modelos de estimación muy temprana del esfuerzo

Se construyeron modelos predictores del esfuerzo de la misma forma que los de tiempo. **Se mantuvo la misma definición para el DCXW que se obtuvo a partir del modelo de tiempo, lo que no contradice la hipótesis de que la complejidad esencial del sistema guarda relación con la de los datos.**

A continuación se presentan los modelos obtenidos.

$$Ln(EFFORT)=\alpha Ln(DRT) + \beta Ln(RD) + \chi Ln(NA) + \delta Ln(NT) + \gamma Ln(IME) + \lambda Ln(RV) - 4{,}252472984$$

Deshaciendo el cambio de variable

$$EFFORT=0{,}007 DRT^{\alpha}\ RD^{\beta} NA^{\chi} NT^{\delta} IME^{\gamma} RV^{\lambda}$$ Very Early Effort Estimation Model 1 (VEEEM1)

Donde

α = -2,12, β=- 0,035, χ = 1,26 δ=0,07,γ=-0,48 y λ= 0,88

También se sugiere por razones prácticas redondear estos exponentes a dos dígitos.

De la misma forma que para el tiempo, se estimo el esfuerzo a partir de la complejidad de la estructura de los datos.

$$EFFORT=0{,}01\ DCXW^{1{,}02}\ IME^{0{,}29}\ RV^{0{,}5}$$ Very Early Effort Estimation Model 2 (VEEEM2)

En las Fig 4-6 y la tabla 3 se presentan las características de estos modelos en forma análoga a como se hizo con los de tiempo.

7.2.5 Resumen de modelos de estimación temprana del esfuerzo de desarrollo

Se presentaron dos modelos que califican como excelente de acuerdo a los criterios de la tabla 1 cuya subestimación es a lo sumo del orden del 20% y la sobreestimación del 30% en los peores casos, con un MRE del orden del 10% y sin una dispersión marcada. En virtud de lo reducido de la muestra a partir de la cual se construyeron estos modelos y los buenos resultados arrojados por ambos no es posible decidir por uno u otro.

8 CONCLUSIONES

Se definió una métrica muy temprana de complejidad cognitiva de la estructura de los datos (DCXW).Se presentaron modelos de estimación muy temprana de tiempo y esfuerzo a partir de métricas recogidas automáticamente desde las vistas de datos de usuario y que no requieren de juicio experto ni intervención humana.

Tres de los cuatro modelos obtenidos califican como excelentes de acuerdo a los criterios de Bennet [2], basados en las métricas de consistencia y precisión de Conte, Dunsmore y Shen [10]. Estos modelos usan métricas propuestas por Calero, Piattini, Polo y Ruiz [11] (DRT, RD, NA) a las que nosotros agregamos el número de tablas NT. Los resultados alcanzados, confirman la hipótesis enunciada hace tanto tiempo por DeMarco [13] de que los sistemas data-strong pueden ser estimados a partir de los datos y la bondad de las métricas de complejidad para bases de datos relacionales propuestas por Calero, Piattini et al [11] y contribuyen a aportar el soporte empírico que los propios autores expresaron en su publicación que era necesario. Los resultados de esta investigación, basados en que el modelo de bases de datos relacionales se apoya en un modelo matemático, ponen de manifiesto en forma explícita la relación que guardan estas variables con la complejidad del sistema a construir y muestran que esta estimación puede realizarse muy tempranamente y sin recurrir a juicio experto. Identificamos una métrica de complejidad cognitiva del esquema relacional a partir de la cual se construyeron modelos de estimación de tiempo y esfuerzo, que incluso se comportaron mejor, para los proyectos más grandes, que los construidos directamente a partir de las variables independientes originales. Se debe tener en cuenta que esta métrica fue definida durante la construcción del modelo de estimación de tiempo y luego usada sin cambios en los modelos de estimación de esfuerzo. Se verificó que existe una fuerte correlación entre esfuerzo y la métrica DCXW. Si se consideran los modelos que surgen de la regresión lineal solamente a partir de DCXW se puede estimar esfuerzo y tiempo con errores relativos promedio del 30% y máximos del 80%. Se intentó agregar elementos que contemplaran el peso de los procesos y las estructuras de control a los modelos post mortem, pero si bien aumentó la precisión de los modelos, no se apreciaron diferencias sustanciales (esto no aparece en el trabajo por razones de espacio, puede encontrarse mayor información en [23]). Esto confirma, o al menos no contradice nuestra hipótesis de que la complejidad esencial de un SG se encuentra en las visiones de usuario y puede ser estudiada a partir del esquema de bases de datos relacional derivado de ellas. Las métricas elegidas son intuitivas (para nosotros, desafortunadamente no para los usuarios) y fueron sugeridas con anterioridad por otros autores. Las métricas usadas fueron obtenidas automáticamente. Todo esto contribuye a aumentar la confianza en los resultados obtenidos. A pesar de que las métricas surgen automáticamente de las vistas de datos de usuario, **no son intuitivas para ellos.** Como, además, el mismo sistema puede ser visto de diferentes formas por distintos usuarios, no pudimos por el momento construir modelos de estimación con métricas obtenidas directamente a partir de las vistas de datos de usuario, sin embargo algunos de estos parámetros como NA y RD pueden ser estimados. Estos modelos son de estimación muy temprana para quien disponga de una herramienta similar a la que se usó para desarrollar los sistemas estudiados. También podría calcularse la métrica DCXW si se dispusiera de una herramienta que, dadas las vistas de datos de usuario, calcule DRT, RD, NA y NT. Naturalmente, luego deberá ser relacionada con tiempo y esfuerzo para cada ambiente de desarrollo, especialmente si no se dispone de una herramienta que nos independice de la tecnología. Los modelos encontrados, pueden ser también usados para estimar el efecto de diferentes velocidades de ejecución del proyecto, volatilidades de los requerimientos y/o el agregado o supresión de vistas de datos de usuario sobre el tiempo y esfuerzo requeridos. Estos modelos pueden ser útiles para elaborar planes de versiones e iteraciones en el desarrollo y discutir con los usuarios las reales posibilidades técnicas de alcanzar determinadas fechas sobre una base objetiva. También pueden ser útiles para gestionar contratos y ajustar precios de acuerdo a las variaciones experimentadas en el esquema de datos, rapidez de ejecución y volatilidad de los requerimientos con respecto a las acordadas inicialmente. La conclusión final parece ser que si normalizamos nuestras prácticas y concentramos nuestro esfuerzo intelectual en los aspectos no automatizables del proceso podemos ser mucho más predecibles y eficientes, confirmando nuestras hipótesis de trabajo.

9 LÍNEAS DE TRABAJO FUTURAS

Desarrollar y publicar un algoritmo que, a partir de las vistas de datos de usuario, nos de las métricas usadas en este trabajo de manera de poder calcular tempranamente estos valores y que cada equipo de desarrollo pueda vincularlos con su experiencia.

Buscar una métrica que permita *pesar* la funcionalidad entregada en una KB y apoye la gestión de contratos y proyectos en base al avance y la funcionalidad entregada.

10 RECONOCIMIENTOS

Los valiosos y constructivos comentarios de los revisores anónimos contribuyeron a mejorar este artículo. Este trabajo no hubiera sido posible sin la colaboración de Karina Santo, José Luís Chalar, Gustavo Carriquiry y Claudia Araujo de Artech Consulting, Enrique Latorres y Jose Luís Subelzú del Departamento de Informática del Ministerio de Transporte y Obras Públicas de Uruguay, Juan Andrés Leiras del Departamento de Informática de Sanidad Policial, Óscar Camargo de la Universidad del Trabajo de Uruguay y la Universidad ORT Uruguay y Gonzalo Pérez y Joaquín González de CONEX Consulting. Fueron fundamentales los aportes de Nicolás Jodal, Karina Santo y José Luis Chalar de Artech Consultores. Merecen un comentario aparte el soporte, aliento, numerosas conversaciones e inteligentes sugerencias de Julio Fernández, Decano de Desarrollo Académico de la Universidad ORT Uruguay. Representaron un invalorable aporte las conversaciones mantenidas con Ernestina Menasalvas tanto en UPM como en ORT. Fueron importantes los acertados y constructivos comentarios de Luis Olsina de la Universidad Nacional de La Pampa durante sus visitas a ORT y en el transcurso de las defensas de los proyectos finales de grado de los

estudiantes que participaron de esta investigación. Estudiantes de grado y postgrado del Laboratorio de Investigación en Sistemas de Información y la Cátedra de Teoría de la Facultad de Ingeniería de la Universidad ORT Uruguay colaboraron con esta investigación. Las visitas a UPM del primer autor son financiadas por el Programa de Desarrollo Tecnológico del BID.

11 REFERENCIAS

1 Albrecht, A. J. Measurement application developments, Proceedings of IBM Applications Development Joint SHARE/GUIDE Symposium, Monterey, CA, pp 83-92,1979

2 Bennett , Warren. Predicting software system development effort Very early in the life-cycle using Idef0 and idef1x models. Phd Dissertation Submitted to the Faculty of Mississippi State University. December 1996.

3 Boehm, B. A Spiral Model of Software Development and Enhancement. Computer. May, 1988.

4 Boehm, B. et al. Software Cost Estimation with COCOMO II. Prentice Hall, 2000

5 Boehm, B. Software Engineering Economics. Prentice Hall, 1981.

6 Boehm, B. Software Risk Management. IEEE Computer Society Press. 1989.

7 Boehm, B., Madachy R., Selby, R. Cost Models for Future Software Life Cycle Processes: COCOMO 2.0. http://sunset.usc.edu/COCOMOII/cocomo.html.

8 Briand Lionel, El Emam Khaled , Morasca Sandro. On the Application of Measurement Theory in Software International Software Engineering Research Network technical report #ISERN-95-04

9 Common Software Internacional Consortium Full Function Point Measurement Manual. (THE COSMIC IMPLEMENTATION GUIDE FOR ISO/IEC 19761: 2003) VERSION 2.2 January 2003

10 Conte, Samuel D., H. E. Dunsmore, and Vincent Y. Shen. 1986. Software engineering metrics and models. Menlo Park, CA: Benjamin/Cummings.

11 Coral Calero, Mario Piattini, Macario Polo, Francisco Ruiz. Grupo ALARCOS, Departamento de Informática,Universidad de Castilla La Mancha. Métricas para la evaluación de Complejidad de Bases de Datos Relacionales. Computación y Sistemas Vol. 3, Nº 4, pp 264-273, 2000, CIC – IPN. ISSN 1405-5546.

12 Dávila Daniel and Chalar Luis, Estimación de Proyectos, Métricas y Herramientas XII Genexus International Meeting.

13 DeMarco,T. Controlling Sofiware Projects. New York: Yourdon, 1982.

14 Grompone Juan. Gestión de Proyectos de Software. La Flor de Itapebí. Olmer S.A. 1996. Montevideo, Uruguay ISBN 9974-592-05-4.

15 Humphrey Watts. Introduction to the Team Software Process. Addison Wesley 1999.

16 Humphrey, Watts. A discipline for Software Engineering. Software Engineering Institute 1995.

17 Latorres, Enrique, Salvetto, Pedro, Larre Borges, Uruguay, Nogueira, Juan C.Una herramienta de apoyo a la gestión del proceso de desarrollo de software. IX CACIC, Octubre 2003.

18 Nogueira, J.C. A Formal Estimation Model for Software Projects. International Conference on Computer Science,
Software Engineering, Information Technology, e-Business, and Applications (CSITeA 02). Foz do Iguacu,
Brasil, 2002.

19 Nogueira, J.C. A Formal Risk Assessment Model for Software Projects. Ph.D. Dissertation. Naval Postgraduate
School, 2000.

20 Putnam, L. and Myers, W. Five Core Metrics: The intelligence behind successful software mangement. Dorset House.2003

21 Putnam, L. and Myers, W. Industrial Strength Software. Effective Management Using Measurement. IEEE Computer Society Press, 1997.

22 Salvetto, P., Nogueira J.C. Size estimation for Management Information Systems Based on Early Metrics. Proceedings CSITeA03. Río de Janeiro, 2003.

23 Salvetto, Pedro, Modelos Automatizables de Estimación muy temprana de Tiempo Y Esfuerzo de Desarrollo. XIV encuentro internacional Genexus. Montevideo, Uruguay 16/5/04.Transmitida en vivo por internet. La conferencia y transparencias pueden descargarse de http://www.gxtechnical.com/main/hevviewsession.aspx?8,60,581,19%3a569

24 Shao, J. and Wang Y. A new measure of software complexity based on cognitive weights. Can. J. Elec. Comput. Eng, Vol 28,Nº 2, April 2003.

25 Valverdi, Ricardo, Bohem, Barry and Reifer Donald. COSYSMO: A Constructive Systems Engineering Cost Model Coming of Age. *Submitted for the INCOSE 2003 Symposium, June 29 – July 3 2003, Washington, DC*

Anexo 1 Figuras y Tablas

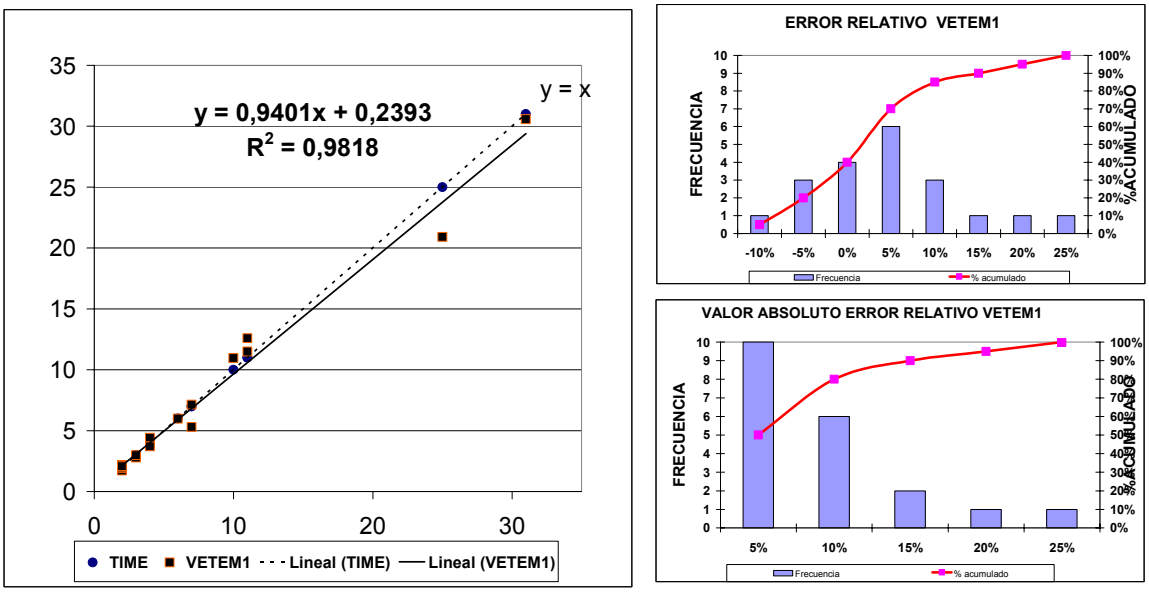


Fig 1 Modelos de Estimación Temprana de Tiempo de Desarrollo (VETEM1)

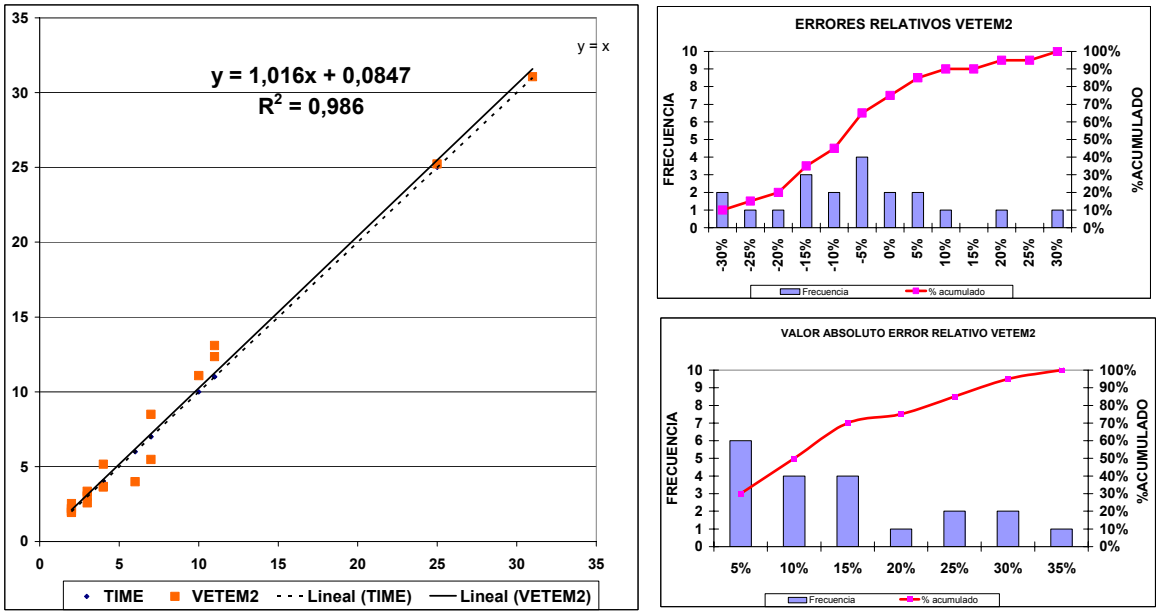


Fig 2 Modelos de Estimación Temprana de tiempo de desarrollo (VETEM2)

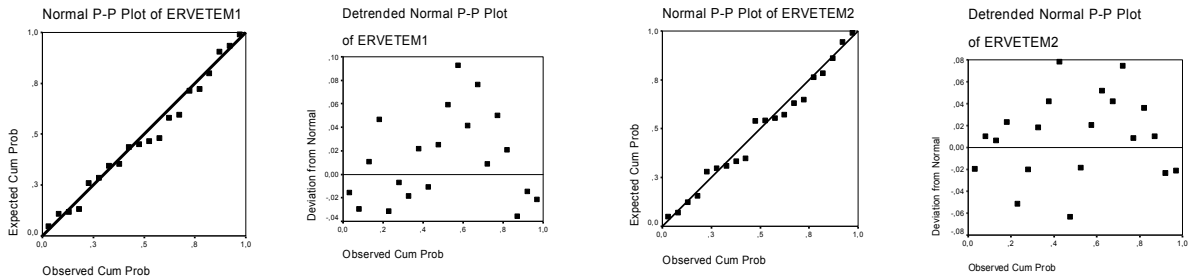


Fig 3 Gráfica de Normalidad de los Residuos de Modelos de Estimación Temprana de Tiempo

Modelos de Estimación Temprana de Tiempo																	
ERRORES RELATIVOS																	
EN VALOR ABSOLUTO																	
	PRED(k)								MIN	MAX	MRE	DESV EST	MIN	MAX	PRO	DESV EST	R2	CAL
	RE	5%	10%	15%	20%	25%	30%	35%										
VETEM1	Frecuencia	10	6	2	1	1			0,3%	24,3%	7,3%	6,4%	-14,4%	24,3%	1,8%	9,6%	0,9698	Ex
	% Acumulado	0,50	0,80	0,90	0,95	1,00												
VETEM2	Frecuencia	6	4	4	1	2	2	1	0,2%	33,6%	12,2%	10,0%	-28,9%	33,6%	-3,0%	15,7%	0,9509	B
	% Acumulado	0,30	0,50	0,70	0,75	0,85	0,95	1,00										

Tabla 2 Comparación de modelos de estimación temprana del tiempo de desarrollo

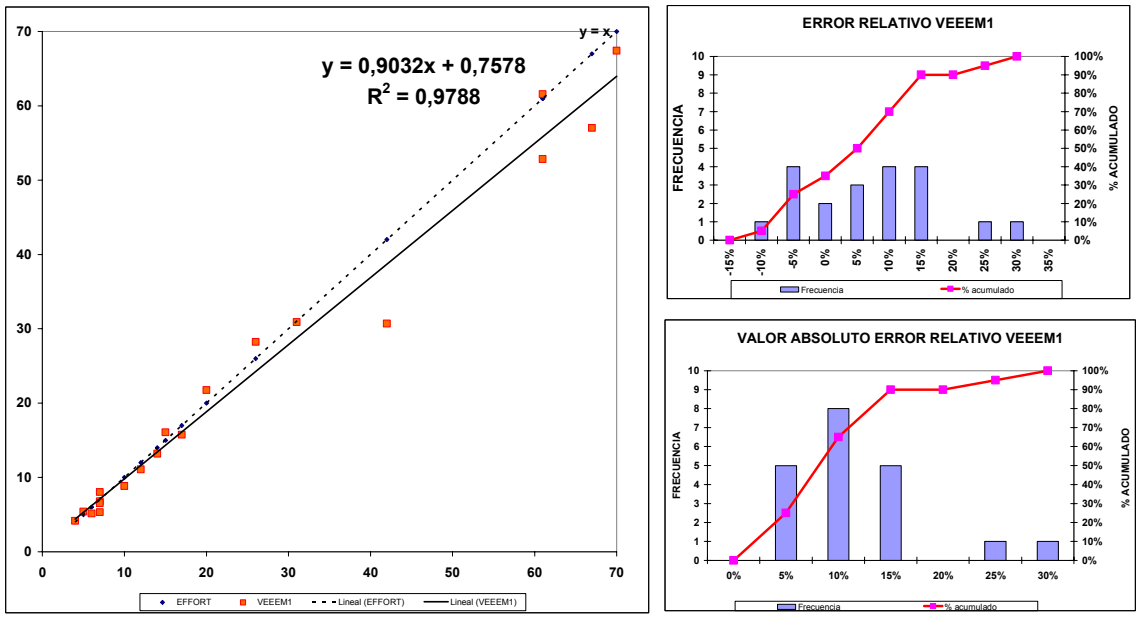


Fig 4 Modelos de Estimación Temprana del Esfuerzo de Desarrollo (VEEEM1)

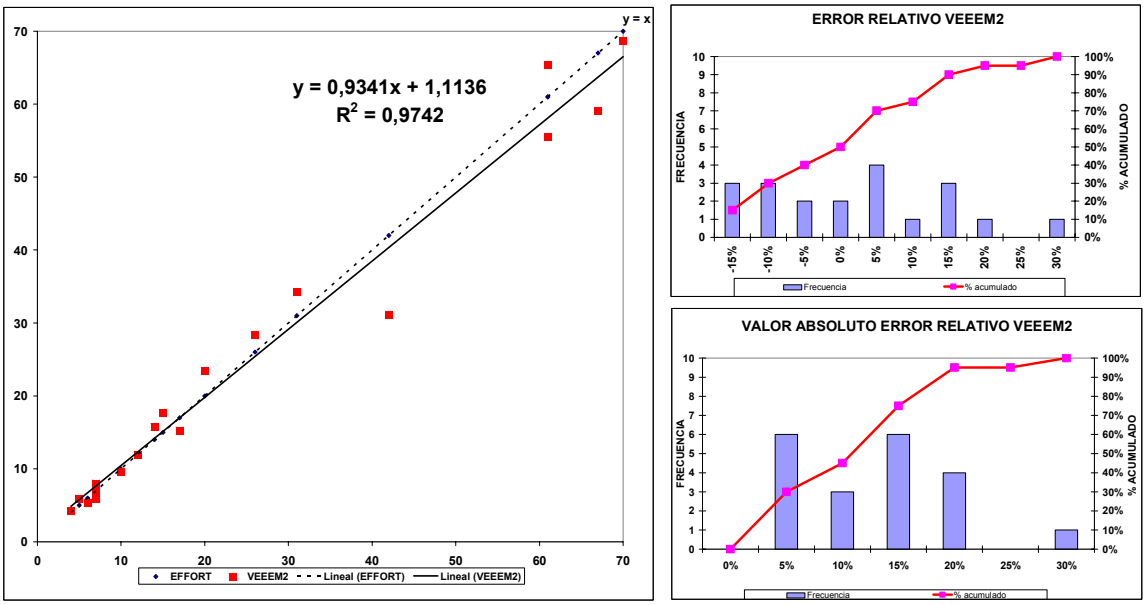


Fig 5 Modelos Tempranos de Estimación del Esfuerzo de desarrollo (VEEEM2)

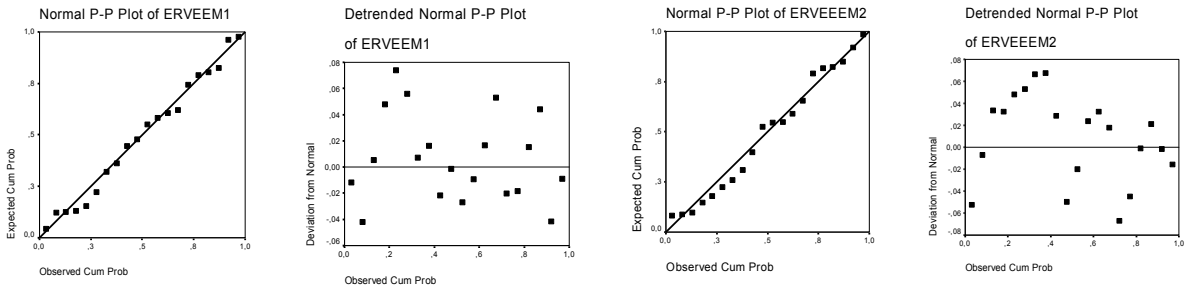


Fig 6 Gráfica de Normalidad de los Residuos de los Modelos de Estimación Temprana de Esfuerzo

Modelos de Estimación Temprana de Esfuerzo																		
ERRORES RELATIVOS																		
EN VALOR ABSOLUTO																		
PRED(k)									MIN	MAX	MRE	DESV EST	MIN	MAX	PRO	DESV EST	R2	CAL
	RE	5%	10%	15%	20%	25%	30%	35%										
VEEEM1	Frecuencia	5	8	5	0	1	1		0,3%	26,9%	9,6%	7,0%	-15,0%	26,9%	4,3%	11,2%	0,9788	Ex
	% Acumulado	0,25	0,65	0,90	0,90	0,95	1,00											
VEEEM2	Frecuencia	6	3	6	4	0	1		0,2%	25,9%	10,1%	7,0%	-18,1%	25,9%	-0,9%	12,4%	0,9742	Ex
	% Acumulado	0,30	0,45	0,75	0,95	0,95	1,00											

Tabla 3 Comparación de modelos de estimación temprana del esfuerzo de desarrollo

num	EFFORT	TIME	IME	RV	DRT	RD	NA	NT	DCXW
1	70	25	3	20	3	114	542	85	1022
2	14	6	4	20	1	4	29	6	222
3	12	4	3	10	3	6	120	9	258
4	26	7	3	30	5	97	535	85	352
5	5	4	2	10	1	3	25	10	145
6	17	3	6	10	8	284	1042	195	270
7	15	3	6	30	5	33	259	24	183
8	42	7	6	20	6	258	842	136	387
9	4	2	2	20	4	13	123	35	73
10	20	3	8	20	6	103	489	40	271
11	7	2	4	10	5	75	329	69	160
12	67	10	5	30	6	100	948	115	627
13	7	2	3	30	5	32	182	27	90
14	7	4	2	10	1	4	23	6	143
15	61	11	5	30	4	111	588	74	693
16	10	2	4	30	4	25	136	14	113
17	61	11	6	10	4	94	666	65	956
18	6	2	2	30	3	13	79	17	77
19	7	2	4	10	3	17	111	17	140
20	31	31	1	30	1	1	44	8	577

Tabla 4 Datos Relevados