

Herramienta para apoyar la estimación en el desarrollo de aplicaciones web

Osmar Gabriel Pedrozo Petrazzini¹, Cristina Greiner²

¹Estudiante, Licenciatura en Sistemas de Información, Universidad Nacional del Nordeste
Facultad de Ciencias Exactas y Naturales y Agrimensura, 9 de julio 1449 (3400) Corrientes
gabriel.pedrozopetrazzini@gmail.com

²Profesora, Directora Trabajo Final de Aplicación, Licenciatura en Sistemas de Información,
Universidad Nacional del Nordeste
Facultad de Ciencias Exactas y Naturales y Agrimensura, 9 de julio 1449 (3400) Corrientes
cgreiner@exa.unne.edu.ar

Abstract. Una tarea importante en la gestión de proyectos de software consiste en la estimación del esfuerzo, costo y duración que demandará el mismo. La falta de precisión en la estimación puede afectar el cumplimiento de los plazos comprometidos, la calidad del producto entregado, el costo final del desarrollo y la competitividad de las empresas. En este trabajo se estudiaron cuatro metodologías específicas para la estimación de proyectos web y se realizó una aplicación que automatiza, simplifica y guía la aplicación de las mismas, que fue validada con datos experimentales. Los resultados indican que, a mayor especificidad en cuanto al tipo de proyecto y la adecuada configuración de parámetros que identifican el entorno de la aplicación, el ajuste de la estimación es mayor. Se espera que esta herramienta contribuya con la calidad del software permitiendo mejorar la estimación de proyectos web, facilitando los cálculos, realizando un seguimiento de las estimaciones y aportando información histórica como feedback para estimaciones posteriores.

Keywords: Ingeniería web, Estimación de esfuerzo, puntos de función

1. Introducción

En la actualidad, el crecimiento de la complejidad y el número de funciones que el software ofrece, hace que la estimación del esfuerzo sea una actividad importante para el desarrollo de aplicaciones web. Las aplicaciones web constituyen una clase especial de aplicaciones de software que se construyen de acuerdo con ciertas tecnologías y estándares. La Ingeniería Web tiene como objetivo la aplicación de enfoques de la ingeniería de software al desarrollo y evolución de las aplicaciones Web [1].

La cantidad de dominios de aplicación y de usuarios a lo largo del tiempo crece cada vez con mayor aceleración. Como resultado de esto ha crecido la cantidad de personas que desarrollan aplicaciones web. El desarrollo de este tipo aplicaciones ha demostrado ser una tarea compleja: parafraseando el célebre “software crisis” el término “Web Crisis” comenzó a utilizarse en 2001 [2].

Para entender la estimación de recursos para proyectos web, según [3] se han realizado estudios previos que dieron como resultado el desarrollo de modelos que usan como entrada

factores tales como el tamaño de una aplicación web, y conductores de costo (por ej., herramientas, calidad de los desarrolladores, tamaño del equipo) y proveen una estimación de esfuerzo como salida.

En este contexto, la herramienta que se propone permitirá mejorar la estimación de proyectos web, facilitando los cálculos, realizando un seguimiento de las estimaciones y aportando información histórica como feedback para estimaciones posteriores, mejorando la precisión de las mismas.

1.1. Objetivo

Contribuir a la mejora en la calidad del software mediante una herramienta que facilite la aplicación de metodologías de estimación para aplicaciones web, que permita además generar información histórica para mejorar la precisión de estimaciones futuras en el contexto específico de uso.

2. Estimación de software

Según McConnell [4] “una estimación es una predicción de cuánto tiempo durará o costará un proyecto”, constituye la base para la planificación de los proyectos. El desarrollo del software requiere de la estimación para controlar y administrar los recursos que se necesitan utilizar antes y durante el proyecto. Son numerosas las variables, relacionadas con los recursos humanos, el contexto y las políticas que intervienen en su proceso que pueden afectar los resultados finales.

Estimar adecuadamente el tiempo, costo y recursos requeridos para realizar un proyecto de desarrollo o mantenimiento de software es un reto que enfrentan las empresas de software. Las pequeñas y medianas empresas que inician proyectos de mejora de sus procesos, se enfrentan con el desafío de seleccionar, por una parte, el método de estimación más adecuado a su contexto y, por otra, adoptar mejores prácticas que permitan ir consolidando una base histórica de estimación, con el propósito de disminuir cada vez más la desviación entre los valores estimados y reales [5].

Existen diversos métodos para estimar el esfuerzo de desarrollo, que son clasificados en dos categorías: la primera comprende a los métodos paramétricos, son aquellos en los cuales el proceso de cuantificación del resultado está basado en un proceso mecánico, por ejemplo la aplicación de una fórmula derivada de los datos históricos. La segunda categoría, comprende a los métodos heurísticos, son aquellos en los cuales la cuantificación del resultado se produce a partir del juicio y/o la experiencia de un experto [6]. En un trabajo previo se elaboró un cuadro para resumir los principales métodos reportados en la literatura [7].

2.1. Estimación basada en puntos de función

Los Puntos de Función (PF) constituyen una métrica para establecer el tamaño y complejidad de los sistemas informáticos basada en la cantidad de funcionalidad requerida y

entregada a los usuarios. Miden el tamaño lógico o funcional de los proyectos o aplicaciones de software basado en los requerimientos funcionales del usuario.

El método de Puntos de Función fue publicado por primera vez en el año 1979 por Allan J. Albrecht [8] y se obtienen utilizando una relación empírica basada en medidas cuantitativas del dominio de información del software y valoraciones subjetivas de su complejidad. Pretende medir la funcionalidad entregada al usuario independientemente de la tecnología utilizada para la construcción y explotación del software, y también ser útil en cualquiera de las fases de vida del software, desde el diseño inicial hasta la explotación y mantenimiento.

A continuación, se describen los métodos de estimación seleccionados en función de sus requerimientos de información que los hacen apropiados para contextos de desarrollo de pequeña o mediana envergadura.

Use Case Points (UCP)

La metodología UCP se basa en la utilización de casos de uso como dato de entrada para calcular el esfuerzo -en horas-hombre (hh)- necesario para el desarrollo de un proyecto de software [11].

El método utiliza cuatro variables principales:

- Actores (Ac): se evalúa su característica intrínseca y se determina la forma en que interactúan con el sistema a desarrollar, asignándole un peso: a los simples se asigna 1, a los medios 2, y a los complejos 3.
- Casos de uso (CU): Se clasifican en: Simple (peso = 5) si tiene 3 o menos transacciones; Medio (peso = 10) si posee de 4 a 7 transacciones; y Complejo (peso = 15) si posee más de 7 transacciones.
- Factores técnicos de complejidad: Influencias técnicas que puedan afectar el proceso de desarrollo. Existen trece, cada uno ponderado en función de su impacto relativo en una escala de 0 a 5, donde 0 es irrelevante, 3 es medio, y 5 es esencial.
- Factores del entorno del proyecto: indican la influencia del grupo humano involucrado en el proyecto. Se utiliza un total de 8 factores, con una escala de 0 a 5 para ponderarlos, siendo 0 irrelevante, 1 un fuerte impacto negativo, 3 es promedio y 5 con un fuerte impacto positivo.

Los pasos a seguir para conseguir la estimación son los siguientes [12]:

- Revisar los aspectos clave de los requerimientos para calcular un recuento de Puntos Caso de Uso sin ajustar (1)

$$UUCP = \sum_{i=1}^n Ac_i P_i + \sum_{i=1}^m CU_i P_i \quad (1)$$

UUCP = puntos de casos de uso sin ajustar

Ac= Actores

CU = Casos de uso

P = peso asignado

- Evaluar los factores técnicos (TCF) y de entorno (ECF) para crear los factores de ajuste.
- Ajustar los UUCP para obtener los Puntos Caso de Uso ajustados (UCP) (2), que posteriormente se transformarán en una estimación de esfuerzo (hh).

$$UCP = UUCP * TCF * ECF \quad (2)$$

Se obtiene una aproximación del tamaño en puntos de casos de uso. Luego se calcula el esfuerzo considerando el Factor de Productividad (FP), el cual indica la cantidad de horas-hombre necesarias para completar un punto de caso de uso (3). El método establece un valor de 20 FP si no se cuenta con datos históricos en la empresa. Conseguido el esfuerzo se calcula la duración del proyecto en días (4).

$$\text{Esfuerzo} = UCP * FP \quad (3)$$

$$\text{Duración} = \text{Esfuerzo} / \text{Horas por día} \quad (4)$$

Method Based On Web Object (WEBMO)

WEBMO es un modelo de estimación orientado a aplicaciones web basado en COCOMO II, propuesto por Reifer [11].

La unidad de medida de tamaño es el “Web Object”, y abarca además funciones específicas de las aplicaciones web, tales como: Links, Multimedia, Scripts, etc. [12].

Para calcular el esfuerzo y la duración se utilizan las formulas (5) y (6):

$$\text{Esfuerzo} = A \left(\prod cdi \right) (\text{tamaño})^{P1} \quad (5)$$

$$\text{Duracion} = B(\text{esfuerzo})^{P2} \quad (6)$$

A, B son constantes multiplicativas de esfuerzo.

P1, P2 exponentes que se determinan en función del tipo de proyecto y del tamaño, que se detallan en la tabla 1.

Cdi: Manejadores de costos.

Tamaño: líneas de código fuente, expresado en miles. Se obtiene multiplicando el total de Web Objects por una constante asociada al lenguaje de programación.

Tabla 1. Constantes de acuerdo al tipo de proyecto

Parámetros / Exponentes	A	B	P1	P2
Basado en el comercio electrónico	2,3	2	1,05	*
Aplicación financiera	2,7	2,2	1,05	*
Aplicación de B2B (negocio a negocio)	2	1,5	1	*
Aplicación basada en utilitarios informativos	2,1	2	1	*

(*) 0,5 para menos de 300 puntos de objeto o 0,32 para más 300.

Para calcular el producto de los manejadores de costos (cdi) se seleccionan los valores (que se detallan en [12]), asociados a distintos aspectos vinculados con el proyecto:

CPLX: Complejidad y confiabilidad del producto

PDIF: Dificultad de la plataforma

PERS: Capacidades del personal

PREX: Experiencia del personal

FCIL: Comodidades

SCED: Restricciones de horarios

RUSE: Grado de reutilización planeado

TEAM: Trabajo en equipo

PEFF: Eficiencia del proceso

Los Web Objects se calculan a partir de los siguiente predictores:

ILF: Archivos lógicos internos.

EIF: Archivos externos

EI: Entradas externas

EO: Salidas externas

EQ: Consultas externas

MmF: Archivos multimedia

WBB: Constructor de bloques web

Scripts: número de macros, contenedores, etc

Links: número de referencias

En cada caso, se asigna un peso (bajo, medio, alto) según corresponda.

Chilean Web Application Development Effort Estimation (CWADEE)

El CWADEE es un método para la estimación de esfuerzo en proyectos de software web, enfocado en apoyar al estimador experto para obtener una estimación más precisa en escenarios inmaduros. El método está diseñado para ser utilizado con poca información histórica de la empresa, para proyectos web pequeños y medianos, el tipo de proyecto usual en empresas tipo pyme.

Introduce una métrica denominada Data Web Points (DWP) que mide el tamaño de forma indirecta. Para ello parte del modelo de datos del sistema a desarrollar que se obtiene de las especificaciones del cliente en las primeras entrevistas.

El método realiza una clasificación de las entidades y relaciones del modelo de datos asignándoles un tipo de patrón. Los patrones para las entidades son Regular, Dependiente, y Relación, considerando relaciones 1-N y 1-1. Luego, realiza un conteo de la cantidad de entidades y relaciones para cada tipo de patrón y le asigna un peso determinado por el estimador experto. A continuación se suman las cantidades de patrones por su peso para obtener los DWP.

Para calcular el esfuerzo, hace uso de la fórmula (7):

$$E = UC * CD * (DWP(1 + X^*))P \quad (7)$$

E es el esfuerzo en horas-hombre

UC es el costo de usuario

CD es el producto de todos los directores de costo (cost driver)

DWP cantidad de Data Web Points que representan el tamaño de la aplicación

X* es el coeficiente de representatividad de los DWP

P es una constante. Se define como 1,05 para proyectos con DWP < 300 y de 1,02 para proyectos con DWP > 300.

El costo de usuario (UC) es una función de los tipos de usuario que interactúan con el sistema. El método define tres tipos de usuarios por defecto: el Administrador, el Actualizador y el Consultor, pero es posible definir otros en función del sistema. Para cada usuario se determina el porcentaje de uso de funcionalidades del sistema o fracción de ámbito (I) y el grado de reutilización de las funcionalidades del sistema (R), que puede venir de la implementación de funcionalidad para otros usuarios del sistema, o de otros sistemas. Teniendo esto en cuenta, se utiliza la siguiente fórmula para determinar el costo de usuario (8):

$$UC = \sum I_i(1 - R_i) \quad (8)$$

Los directores de costo son tomados de WEBMO, salvo uno denominado CLIEN, que se refiere al tipo de Cliente (Conocimiento tecnológico que tiene el cliente). Los valores fueron ajustados luego del estudio de 22 proyectos al momento de definir CWADEE, por lo que presentan diferentes valores del WEBMO original.

Por último, el coeficiente de representatividad de los DWP indica la cantidad de esfuerzo adicional que se requiere para implementar funcionalidades que no se pueden deducir a partir del modelo de datos, tales como, imágenes, navegación, etc. Para determinar su valor se debe recurrir a la información histórica de la empresa sobre proyectos similares, específicamente el esfuerzo medido en horas-hombre. Los pasos son:

- En primer lugar, se determina el valor real de DWP (DWP*) de un proyecto despejando de (9):

$$DWP^* = \sqrt[p]{\frac{E}{UC * CD}} \quad (9)$$

- Luego se obtiene X, el porcentaje del tamaño del sistema que no puede inferirse del modelo de datos, como se indica en (10):

$$X = \frac{DWP}{DWP^*} \quad (10)$$

- Si se tienen más proyectos similares se realiza un promedio de todos ellos para obtener X*, de lo contrario, se considera X*=X.

Raw Estimation Based on Standard Components (RESC)

Este método estima proyectos web y utiliza datos históricos. Explora las similitudes en cuanto a funcionalidad que tienen los componentes presentes en las páginas web considerando el tiempo, esfuerzo y el costo invertido en el desarrollo.

Los componentes considerados son: Mantenimiento de datos, Menú/Navegación, Consultas, Informes por impresora/ pantalla, Procesos en background, Autenticación de datos, Tablas, Notificaciones.

Luego, realiza una evaluación de estos componentes en Puntos de Función para establecer una relación entre la cantidad de Puntos de Función y el esfuerzo, tiempo y costo que implicó la realización del proyecto.

A partir de esta relación, RESC determina qué cantidad de esfuerzo, tiempo y costo es invertido en cada Punto de Función.

Teniendo estos valores como datos históricos, al momento de estimar un proyecto similar, se realiza una estimación a través del juicio de experto de la cantidad de componentes con los que contará el producto, estimando para cada tipo de componente un valor mínimo, máximo y estimado.

A partir de los valores estimados se realiza un ajuste de los mismos considerando su ponderación (11):

$$\text{Estimación Ajustada} = (4 * Est + Max + Min)/6 \quad (11)$$

Una vez obtenida la estimación ajustada se calcula la cantidad de puntos de función que tendrá el proyecto (12):

$$PF = \sum_{i=1}^n P_i A_i \quad (12)$$

P = peso en PF estándar

A_i = Estimación Ajustada de cada componente estándar.

Por analogía con el proyecto web con mayor similitud al estimado, se calcula el tiempo, esfuerzo y costo que demandará este proyecto (13), (14) y (15):

$$Esfuerzo = TPF * DH \quad (13)$$

$$Costo = TPF * CPH \quad (14)$$

$$Tiempo(en\ meses) = Esfuerzo / HD / DM \quad (15)$$

TPF = Total de puntos de función calculados para el proyecto estimado

DH = dedicación por punto de función en el proyecto histórico CPH = Costo por punto de función en el proyecto histórico.

HD = Horas al día dedicadas al proyecto.

DM = Días al mes dedicado al proyecto.

Debido a la gran cantidad de cálculos que requiere cada metodología y la falta de herramientas que den soporte a las mismas, resulta interesante contar con una aplicación que simplifique, automatice y sea atractiva para el usuario para incentivar el estudio y uso de metodologías de estimación en proyectos web.

3. Metodología

Para cumplir con los objetivos del trabajo, en primer lugar se aplicaron las 4 metodologías de estimación de software en un caso de estudio, con el objetivo de evaluar el comportamiento y la precisión de la estimación de cada método. En base a los resultados obtenidos, una segunda etapa consistió en el desarrollo de una herramienta para automatizar la aplicación de las técnicas de estimación analizadas. La validación de la herramienta se realizó con los mismos datos experimentales del caso de estudio.

3.1. Aplicación experimental de los métodos de estimación

Para evaluar el comportamiento de los métodos estudiados, se tomó como caso de estudio el trabajo práctico obligatorio realizado por los alumnos de la asignatura Taller de Programación I de la LSI. Esta asignatura, ubicada en el tercer año del plan de estudio, tiene como objetivo profundizar el estudio de herramientas de desarrollo de software orientadas a la plataforma web. Para el experimento los alumnos debían cumplir las siguientes consignas:

- Desarrollar una aplicación orientada al comercio electrónico, respetando los criterios y las funcionalidades requeridas, previamente definidos por el profesor.
- Realizar la especificación de requerimientos utilizando la metodología NDT, que incluye el modelado con diagramas de casos de uso.
- Utilizar las herramientas: Eclipse, HTML, CSS, PHP5 y PhpMyAdmin.
- Realizar la presentación de los resultados del proyecto en 2 fases: la primera incluye la programación de los documentos HTML y de las hojas de estilo, y la segunda, la programación en PHP, diseño y conexión de la base de datos.
- Por cada fase, presentar una minuta, indicando la tarea realizada y la duración correspondiente.

Al finalizar el plazo previsto, se presentaron 13 proyectos. A partir de los datos de las minutas entregadas, se obtuvo un promedio de la duración real del desarrollo de los alumnos. Para la primera fase se estableció una duración promedio de 2 hs por día, durante 14 días. Y para la segunda fase una duración promedio de 4 hs por día, durante 30 días. Considerando la duración medida en meses de 20 días de 5 hs, se obtuvo una duración real de 1,48 meses.

Para definir el contexto, se asume que ningún estudiante tiene experiencia en el desarrollo de aplicaciones web, ni en las herramientas que se enseñan en el cursado de la asignatura.

Teniendo en cuenta las consignas dadas para el desarrollo de la aplicación, las características de las herramientas y el contexto específico, se realizó la estimación de la duración con los siguientes métodos: Puntos de Casos de Uso, CWADEE, WEBMO y RESC, a fin de determinar qué método predice mejor la duración para este tipo de proyecto, en función de la información que cada uno requiere.

Para el procesamiento de los datos se utilizó una planilla de cálculo en la que se introdujeron los valores y se especificaron las fórmulas de cada uno de los métodos.

3.2. Desarrollo de la herramienta

Para el desarrollo del producto software se siguió la metodología en cascada modificada, la cual tiene la posibilidad que haya retroalimentación entre sus etapas.

Se seleccionó esta metodología por ser apropiada al contexto particular de requerimientos estables, ya que el producto fue desarrollado por una sola persona, además de que esta metodología está orientada a la documentación, algo muy valorado en este trabajo final.

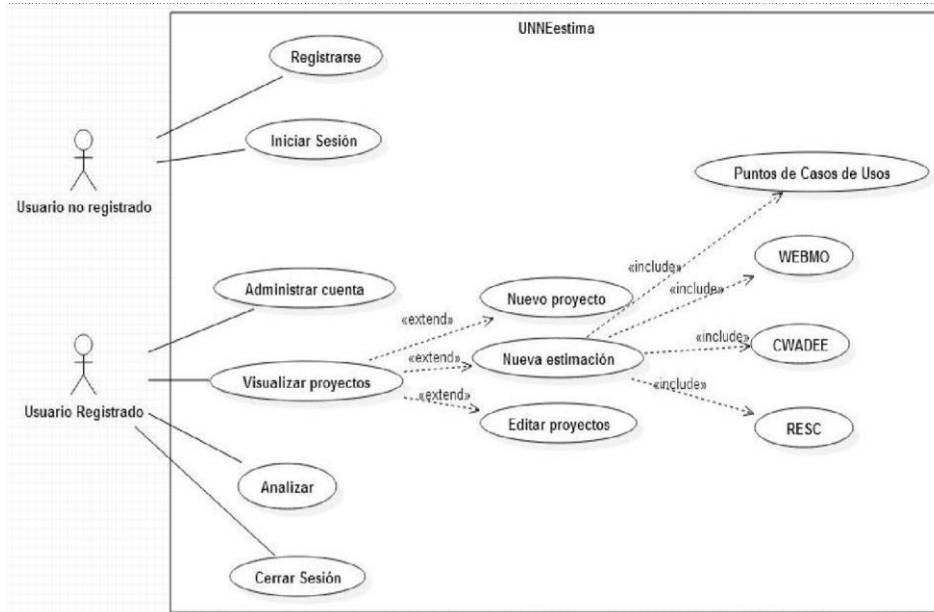


Fig. 1. Diagrama de casos de usos de la aplicación.

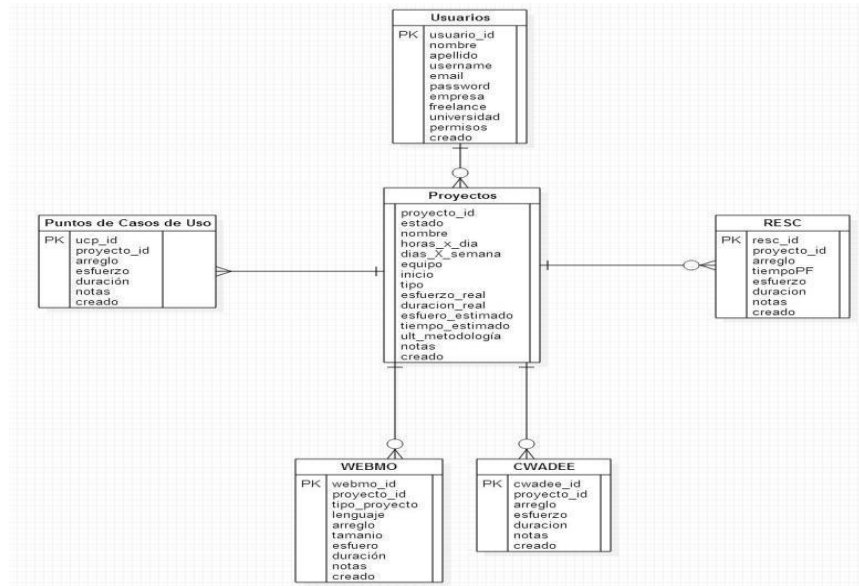


Fig. 2. Modelo de Base de Datos.

4. Descripción y validación de la herramienta

Para la implementación de la herramienta se planteó como arquitectura de software el patrón de diseño Modelo-vista-controlador (MVC) el cual separa los datos y la lógica de negocio de una aplicación de la interfaz de usuario y el módulo encargado de gestionar los eventos y las comunicaciones. Este patrón de arquitectura de software se basa en las ideas de reutilización de código y la separación de conceptos, características que buscan facilitar la tarea de desarrollo de aplicaciones y su posterior mantenimiento.

4.1. Interfaz general.

El sistema es completamente funcional y responsive y cambia su estructura dependiendo del tamaño de la pantalla en la que se está visualizando. Se definieron 3 tamaños diferentes que se ilustran a continuación.

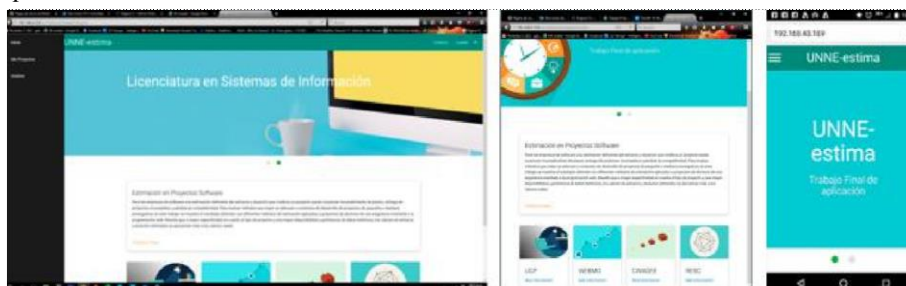


Fig. 3. Vista del sistema en pantallas de distinto tamaño..

4.2. Menús y secciones.

En todas las páginas el sistema cuenta con un menú superior y otro en el lateral izquierdo para dispositivos de pantallas grandes, que se combinan en uno a partir de las pantallas medianas. Estos menús guían la navegación por los distintos componentes del sistema:

- **Página de inicio:** cuenta con una breve reseña acerca de la importancia de la estimación en proyectos software y una breve descripción de las 4 metodologías de estimación que se utilizan en el sistema.
- **Registro de usuario:** datos personales y lugar de trabajo del usuario.
- **Mis proyectos:** es un panel que permite al usuario administrar las altas, bajas y modificaciones a los datos de los proyectos y realizar las correspondientes estimaciones.
- **Proyecto nuevo:** al momento de realizar la estimación de un proyecto, se despliega un menú que permite elegir la metodología a utilizar.

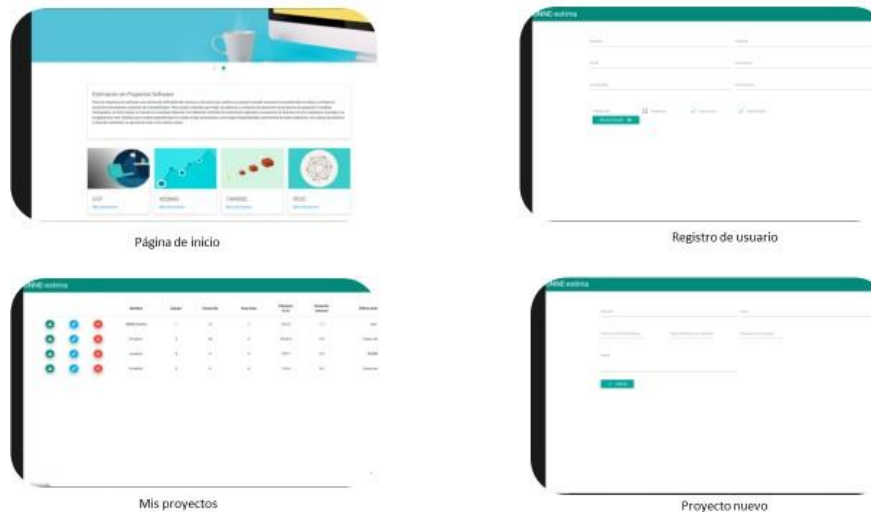


Fig. 4. Secciones de la herramienta.

4.3. Técnicas de estimación.

El sistema permite realizar estimaciones de los proyectos software a partir de las técnicas que se describieron anteriormente. Para ello, una vez seleccionado el Proyecto a estimar el Sistema cuenta con un menú desde el cual permite seleccionar la técnica a utilizar.

- **Casos de Uso:** Cuenta con dos filas de inputs para recibir las cantidades de los distintos actores y casos de usos.
Para evaluar los factores de complejidad y de entorno, se debe seleccionar un valor entre 0 y 5 con el input del tipo range para cada factor.
- **WebMo:** Se debe seleccionar el tipo de proyecto como también el lenguaje entre unos valores precargados y luego ingresar la cantidad de objetos web diferenciándolos entre simples, medios y complejos. Para cada multiplicador de esfuerzo se deberá asignar un peso entre 5 posibles.
- **Cwadee:** Para utilizar esta técnica es necesario contar con un modelo de datos del proyecto a estimar y los datos históricos de algún otro proyecto similar.
- **Resc:** De manera similar a CWADEE, este método requiere contar con los datos históricos de un proyecto similar al que se desea estimar. Se ingresan las cantidades de los componentes especificados de un proyecto similar, y las cantidades que tendrá el proyecto nuevo, ajustado por un mínimo, un máximo y un estimado.



Fig. 5. Técnicas de estimación disponibles en el sistema.

4.4. Análisis de las estimaciones.

El Sistema muestra para cada una de las técnicas utilizadas para un determinado Proyecto una estimación en esfuerzo (h-h) y duración (meses).

Se puede apreciar que se mejora significativamente el ingreso de los datos debido al diseño de la interfaz más amigable, como así también permite el ajuste de los valores de los parámetros con mayor facilidad.









Casos de Uso		Esfuerzo (h-h)	Duración (meses)
		397.5	4.0
WEBMO		Esfuerzo (h-h)	Duración (meses)
		136.9	1.37
CWADEE		Esfuerzo (h-h)	Duración (meses)
		149.7	1.49
RESC		Esfuerzo (h-h)	Duración (meses)
		149.7	1.49

Fig. 6. Ejemplo de resultados de la estimación para un proyecto.

El Sistema también cuenta con una representación gráfica de los resultados que permite visualizar la duración de cada proyecto según las distintas metodologías utilizadas y también la evolución de las estimaciones.



Fig. 7. Ejemplo de resultados de la estimación para un proyecto.

5. Conclusiones y futuros trabajos

La Ingeniería de Software considera que la estimación de software es un requisito necesario para la gestión de proyectos de software exitosos. Contribuye a gestionar los recursos más eficientemente, proporcionar a los clientes un producto software en los plazos previstos, disminuir los costos y minimizar el retrabajo. Para que la estimación pueda ser incorporada como parte del proceso de desarrollo es necesario contar con herramientas que faciliten esta tarea y que los resultados sean tangibles.

En este trabajo se muestran los resultados de la aplicación experimental de cuatro métodos de estimación en un caso de estudio tomado del ámbito académico, que sirvió para conocer al detalle los cálculos requeridos en cada uno de ellos. Las estimaciones obtenidas se compararon con los valores reales de esfuerzo y duración. De los resultados surge que, cuánto más información de los proyectos se aporta y su relación con proyectos similares anteriores, la estimación se aproxima más a los valores reales.

En cuanto a la utilización de la herramienta, se pudo comprobar que disminuye significativamente el tiempo que demanda el ingreso de los datos, permitiendo realizar distintas combinaciones de valores de los parámetros que se requieren, con una interfaz intuitiva que facilita su uso a los gestores de proyectos o desarrolladores. Permite también el análisis global de las estimaciones realizadas, monitoreando la brecha entre los valores estimados y reales, que permitirán introducir las correcciones o adaptaciones necesarias para lograr la finalización exitosa de los proyectos.

Como trabajo futuro se propone aumentar las funcionalidades de seguimiento y evaluación de las estimaciones, automatizar la captura de datos de proyectos anteriores para cumplir los requerimientos de los métodos RESC y CWADEE, y validar la herramienta en un ámbito real del desarrollo de software, con el propósito de que la herramienta pueda ser transferida a las empresas de software de la región.

6. Referencias

1. Kappel, Gerti; Pröll Birgit; Reich, Siegfried; Retschitzegger, Werner: An Introduction to Web Engineering, en [13], pp. 1–21.
2. Ginige, Athula; Murugesan, San: Web Engineering: An Introduction, IEEE Multimedia 8(1), 2001, pp. 14–18.
3. Azhar, D., Mendes, E., Riddle, P.: A systematic review of web resource estimation. PROMISE '12: Proceedings of the 8th International Conference on Predictive Models in Software Engineering. (2012)
4. McConnell, S.: Software Estimation: Demystifying the Black Art (Developer Best Practices). Microsoft. (2006)
5. Moløkken, K., Jørgensen, M., Tanilkan, S.S., Gallis, H., Lien, A.C. and Hove, S.E.: A survey on software estimation in the Norwegian industry. In Proceedings of the 10th International Symposium on Software Metrics. pp. 208–219. (2004)
6. Nasir, M., Ahmad, F.: An Empirical Study to Investigate Software Estimation Trend in Organizations Targeting CMMI. Proceedings of the 5th IEEE/ACIS International Conference on Computer and Information Science and 1st IEEE/ACIS International Workshop on Component-Based Software Engineering, Software Architecture and Reuse. (2006)
7. Dapozo, G.N., Greiner, C.L., Medina, Y., Ferraro, M., Pedrozo Petrazzini, O.G., Lencina, B.: Métodos de estimación de software. Un análisis desde un enfoque evolutivo. III Jornadas de Investigación en Ingeniería del NEA y países limítrofes. UTN - Facultad Regional Resistencia. ISBN: 978-950-42-0157-1. (2014)
8. Albrecht, A.: Measuring Application Development Productivity. Proc of IBM applications. (1979)
9. Remón, C.A., Thomas, P.: Análisis de Estimación de Esfuerzo aplicando Puntos de Caso de Uso. XVI Congreso Argentino de Ciencias de la Computación. (2010)
10. Garzías, J.: Método de Estimación de Puntos de Caso de Uso. Disponible en: <http://www.233gradosdeti.com/estimacion-puntos-caso-de-uso/>
11. Reifer, D.J.: Web Development: Estimating Quick-to-Market Software. Software, IEEE Computer Society, November/December 2000, pp. 57-64. (2000)
12. Reifer, D.J.: Estimating Web Development Costs: There Are Differences. CROSSTALK The Journal of Defense Software Engineering. (2002)