Tecnológico Nacional de México

Instituto Tecnológico de Apizaco

Desarrollo de una Aplicación Didáctica para el Diseño y Evaluación de Mezclas de Concreto

Tesis presentada por Alfredo Xochitemol Cruz para obtener el grado de Maestro en Sistemas Computacionales

Director de Tesis:
M.C. María Guadalupe Medina Barrera

Co-Director de Tesis:
M.C. José Juan Hernández Mora

APIZACO, TLAXCALA. AGOSTO 2018

INSTITUTO TECNOLOGICO DE APIZACO

SEP
SECRETARÍA DE EDUCACIÓN PÚBLICA

TECNOLÓGICO NACIONAL DE MÉXICO

Instituto Tecnológico de Apizaco

Apizaco, Tlax., 01 de Agosto de 2018

ASUNTO: **Aprobación del trabajo de Tesis de Maestría.**

DR. JOSÉ FEDERICO CASCO VÁSQUEZ
JEFE DE LA DIVISIÓN DE ESTUDIOS DE POSGRADO E INVESTIGACIÓN.
P R E S E N T E.

Por este medio se le informa a usted, que los integrantes de la **Comisión Revisora** para el trabajo de tesis de maestría que presenta el **ING. ALFREDO XOCHITEMOL CRUZ**, con número de control **M10370762,** candidato al grado de **Maestro en Sistemas Computacionales** y egresado del **Instituto Tecnológico de Apizaco,** cuyo tema es "**DESARROLLO DE UNA APLICACIÓN DIDÁCTICA PARA EL DISEÑO Y EVALUACIÓN DE MEZCLAS DE CONCRETO",** fue:

A P R O B A D O

Lo anterior, al valorar el trabajo profesional presentado por el candidato y constatar que las observaciones que con anterioridad se le marcaron así como correcciones sugeridas para su mejora ya han sido realizadas.

Por lo que se avala se continúe con los trámites pertinentes para su titulación.

Sin otro particular por el momento, le envió un cordial saludo.

LA COMISIÓN REVISORA

M.C. MARÍA GUADALUPE MEDINA BARRERA

M.C. JOSÉ JUAN HERNÁNDEZ MORA

M.D.S. HIGINIO NAVA BAUTISTA

M.C. JUAN RAMOS RAMOS

C. p.- Interesado.

SEP
SECRETARÍA DE EDUCACIÓN PÚBLICA

TECNOLÓGICO NACIONAL DE MÉXICO

Instituto Tecnológico de Apizaco

Apizaco, Tlax., 07 de Agosto de 2018

No. de Oficio: DEPI/281/18

ASUNTO: **Se Autoriza Impresión de Tesis de Grado.**

ING. ALFREDO XOCHITEMOL CRUZ
CANDIDATO AL GRADO DE MAESTRO
EN SISTEMAS COMPUTACIONALES
No. de Control: **M10370762,**
P R E S E N T E.

Por este medio me permito informar a usted, que por aprobación de la Comisión Revisora asignada para valorar el trabajo, mediante la Opción: **I Tesis de Grado por Proyecto de Investigación**, de la **Maestría en Sistemas Computacionales**, que presenta con el tema: "**DESARROLLO DE UNA APLICACIÓN DIDÁCTICA PARA EL DISEÑO Y EVALUACIÓN DE MEZCLAS DE CONCRETO"** y conforme a lo establecido en el Procedimiento para la Obtención del Grado de Maestría en el Instituto Tecnológico, la División de Estudios de Posgrado e Investigación a mi cargo le emite la:

AUTORIZACIÓN DE IMPRESIÓN

Debiendo entregar un ejemplar del mismo debidamente encuadernado y seis copias en CD en formato PDF, para presentar su Acto de Recepción Profesional a la brevedad.

Sin otro particular por el momento, le envío un cordial saludo.

A T E N T A M E N T E
EXCELENCIA EN EDUCACIÓN TECNOLÓGICA®
PENSAR PARA SERVIR, SERVIR PARA TRIUNFAR®

DR. JOSÉ FEDERICO CASCO VÁSQUEZ
JEFE DE LA DIVISIÓN DE ESTUDIOS DE POSGRADO E INVESTIGACIÓN.

SECRETARÍA DE EDUCACIÓN PÚBLICA
TECNOLÓGICO NACIONAL DE MÉXICO
INSTITUTO TECNOLÓGICO DE APIZACO
DIVISIÓN DE ESTUDIO DE POSGRADO E INVESTIGACIÓN

JFCV/M*H.*mebr.

C.p. Expediente.

Agradecimientos

En primer lugar, quiero agradecer a las instituciones que hicieron posible el desarrollo de esta tesis, al Consejo Nacional de Ciencia y Tecnología (CONACYT) por el apoyo económico otorgado y al Instituto Tecnológico Nacional de México/Instituto Tecnológico de Apizaco por otorgarme la oportunidad para realizar la Maestría en Sistemas Computacionales en este instituto, el apoyo otorgado por ambas instituciones fue esencial para la culminación de este proyecto.

Agradezco a mi directora de tesis, la M. C. María Guadalupe Medina Barrera, por la orientación brindada y al Ing. Miguel Ángel Daza Merino por el asesoramiento, apoyo y conocimiento otorgado durante el desarrollo de esta tesis.

Finalmente, agradezco a mi familia por el apoyo incondicional que siempre me ha dado a lo largo de mi vida.

Resumen

El diseño de mezclas de concreto es un proceso mediante el cual se determina la proporción óptima de los materiales, necesarios para la elaboración del concreto, en base a las características requeridas para la obra. La calidad del concreto resultante depende tanto de la calidad de los materiales como de la correcta combinación de estos, por lo que realizar un correcto proporcionamiento de los materiales es esencial para lograr un concreto de calidad.

En la presente tesis se describen las principales herramientas y métodos utilizados durante el desarrollo de una aplicación didáctica para el diseño y evaluación de mezclas de concreto. De igual forma, se presentan un amplio panorama de las problemáticas presentes en el diseño de mezclas de concreto y las soluciones actuales. Se describe, también, la metodología de desarrollo utilizada, así como el desarrollo de la misma. Por último, se presentan las pruebas de usabilidad y aceptación realizadas a la aplicación, así como los resultados obtenidos. Además, debido a que dicha aplicación muestra paso a paso el desarrollo de los métodos y ofrece una breve explicación sobre cada paso, puede ser utilizada para la enseñanza y aprendizaje de dichos métodos. Los métodos que abarca la aplicación son el método ACI, el método Walker, el método de Bolomey y el método de Füller, cuatro de los métodos más populares utilizados en la actualidad.

Abstract

The design of concrete mixtures is a process by which the optimum proportion of the materials, necessary for the manufacture of the concrete, is determined based on the characteristics required for the work. The quality of the resulting concrete depends both on the quality of the materials and the correct combination of these, so that a proper proportioning of the materials is essential to achieve quality concrete.

In this thesis the main tools and methods used during the development of a didactic application for the design and evaluation of concrete mixtures are described. Similarly, a broad overview of the problems present in the design of concrete mixtures and current solutions are presented. It also describes the development methodology used, as well as the development of it. Finally, the usability and acceptance tests made to the application are presented, as well as the results obtained. In addition, because this application shows step by step the development of the methods and offers a brief explanation of each step, it can be used for the teaching and learning of these methods. The methods covered by the application are the ACI method, the Walker method, the Bolomey method and the Füller method, four of the most popular methods used today.

Índice general

Índice de figuras

Índice de tablas

Capítulo 1

Introducción y Estado del Arte

1.1. Introducción

El concreto es un material para construcción ampliamente utilizado debido a su resistencia, durabilidad y maleabilidad. Se compone básicamente de la mezcla de dos componentes: los agregados y la pasta. Los agregados se dividen en agregado fino (arena) y agregado grueso (grava), mientras que la pasta se compone de agua, cemento y otros aditivos. La calidad del concreto depende tanto de la calidad de los componentes, como de la correcta combinación o mezcla de estos (Kosmatka *et al.*, 2004), por esta razón, es importante lograr un óptimo proporcionamiento de materiales.

El objetivo al diseñar una mezcla de concreto es determinar el proporcionamiento de los materiales que genere la combinación más económica y práctica para producir un concreto que satisfaga los requisitos de la obra. El proporcionamiento de los materiales se logra mediante métodos de diseño de mezclas. Generalmente, estos métodos se basan en funciones fundamentales como la relación agua-cemento, la demanda del agua y la teoría del óptimo proporcionamiento de los agregados, de las cuales se determinan mezclas con las propiedades requeridas (Sobolev, 2003). El proceso de diseño de mezclas de concreto incluye el establecimiento de características específicas y la elección de proporciones de materiales disponibles (Kosmatka *et al.*, 2004).

Existe una gran variedad de métodos para el diseño o proporcionamiento de mez-

clas de concreto. Según las propiedades y características de los materiales, los factores ambientales y la finalidad del concreto, pueden aplicarse o no diferentes métodos. Entre los principales métodos se encuentran: método ACI (*American Concrete Institute*), método Füller, método de módulo de fineza de la combinación de agregados, método Bolomey, método Walker, método Faury entre otros.

La aplicación de estos métodos implica largos procedimientos debido a la gran cantidad de operaciones que conlleva cada uno de estos, por lo que llevar a cabo de forma manual estos métodos implica un extenso e inclusive tedioso procedimiento. Además, es probable que ocurran errores humanos, lo que provoca que se reduzca la fiabilidad de los datos. Debido a lo anterior es ampliamente recomendable el uso de herramientas que agilicen el desarrollo de dichos métodos.

Actualmente existen varios programas de cómputo diseñados para el proporcionamiento de mezclas de concreto, pero la mayoría solo trabaja con uno o dos métodos de diseño o no son completamente enfocados al diseño de mezclas de concreto. Además, de que son muy rígidos en cuanto a su funcionamiento y no permiten modificar los valores obtenidos en cada paso del método, lo cual implica que en ocasiones el usuario no obtenga el resultado deseado.

La enseñanza y aprendizaje de estos métodos presenta contratiempos debido a que para comprobar los resultados de ejercicios hechos es necesario volver a realizar toda la práctica, ya que los programas actuales no muestran el paso a paso de los métodos y no es posible verificar rápidamente dónde se cometió un error si es que lo hubiera.

1.2. Descripción del problema

El diseño o proporcionamiento de mezclas de concreto es un proceso mediante el cual se determina la cantidad óptima de los materiales necesarios para la elaboración del concreto. El proporcionamiento de los materiales se logra mediante métodos de diseño de mezclas como el Método ACI, Walker, Füller y Bolomey. Sin embargo, a menudo este tipo de métodos son extensos e incluyen una gran cantidad de operaciones matemáticas, por lo que resulta difícil elaborar más de uno para una misma práctica. Además, si este tipo de métodos se realiza de forma manual se está propen-

so a errores, consumen mucho tiempo y disminuye la fiabilidad de los resultados. Si no se obtienen los resultados deseados es posible que se necesiten modificar ciertos valores en el procedimiento del método, lo que implicaría repetir el procedimiento, consumiendo así más tiempo del necesario.

En el ámbito didáctico, resulta difícil tanto enseñar y como aprender los distintos métodos de diseños de mezclas de concreto, si no se cuenta con una herramienta para verificar los resultados de forma rápida y confiable.

1.3. Pregunta de investigación

¿Es factible desarrollar una aplicación didáctica que facilite el cálculo de los principales métodos de diseño de mezclas utilizando metodologías de desarrollo ágil?

1.4. Objetivo general

Diseñar y desarrollar de una aplicación didáctica que permita aplicar y comparar diferentes métodos para el diseño de mezclas de concretos.

1.5. Objetivos específicos

1. Determinar los requerimientos funcionales y no funcionales de la aplicación.
2. Definir e investigar los métodos de diseño que incluirá la aplicación.
3. Determinar los valores de entrada y de salida para cada método de diseño de mezclas de concreto.
4. Crear un algoritmo que permita codificar cada método a un lenguaje de programación.
5. Diseñar y desarrollar la base de datos de la aplicación.
6. Diseñar y desarrollar una interfaz intuitiva para la aplicación.
7. Codificar los métodos de diseño seleccionados.

8. Generar un módulo de comparación que permita al usuario realizar el análisis de los datos, incluyendo gráficas y tablas para representar la información.

9. Permitir exportar los resultados a formato PDF.

10. Crear un manual técnico y un manual de usuario.

11. realizar pruebas de usabilidad y aceptación a la aplicación desarrollada.

1.6. Justificación

Es considerado factible desarrollar una aplicación para el diseño de mezclas de concreto, debido a que, al ser modelos sistemáticos, es decir que se rigen por un conjunto ordenado de normas y procedimientos, pueden ser implementados dentro de una aplicación sin afectar su funcionalidad ni resultados. Además, resulta mucho más eficiente en comparación con el desarrollo manual de los distintos métodos de diseño de mezclas, por consecuencia el uno de una aplicación generaría un considerable ahorro de tiempo y aumentaría la fiabilidad de los resultados.

Actualmente ya existen diversos programas de cómputo diseñados para el proporcionamiento de mezclas de concreto, pero la mayoría maneja sólo uno o dos métodos de diseño o no son completamente enfocados al diseño de mezclas. Además, no muestran el procedimiento realizado y por consecuencia no permiten modificar variables en un paso específico, lo que resulta que el usuario pudiera no obtener el resultado deseado.

En el contexto didáctico, desarrollar dicha aplicación, ayudará a maestros a enseñar y a los alumnos a aprender sobre los métodos de diseño de mezclas de concreto de forma más eficiente, ya que permitirá comprobar los resultados obtenidos en las prácticas de clase con los resultados obtenidos en la aplicación de forma más rápida, además de mostrar paso a paso el procedimiento del método.

También permitirá generar informes, tablas y gráficas que representen los resultados obtenidos. Así como permitir al usuario comparar e interpretar los resultados obtenidos, mediante un módulo de comparación, para analizar qué método se adapta mejor a los requerimientos necesarios.

1.7. Alcances

Los puntos que cubrirá la aplicación son:

- Aplicará diferentes métodos para el diseño de mezclas de concreto. Deberán de seleccionarse e implementarse los métodos más representativos entre todos los disponibles. Los métodos seleccionados son: método ACI, método Walker, método Füller y método Bolomey.
- La aplicación se diseñará como una aplicación de escritorio. Es decir que se encontrará instalada dentro del ordenador del usuario y no será necesario el uso de Internet.
- Se incluirán diversos materiales de ayuda para facilitar la comprensión de los métodos de diseño de mezclas, tales como manuales, tablas y normas.
- Se diseñará una interfaz intuitiva y de fácil aprendizaje que permita visualizar los resultados obtenidos a través de los distintos métodos, con el fin de realizar el análisis y la evaluación de los resultados.
- Se incluirán tablas y gráficas, para visualizar de forma rápida y precisa los resultados obtenidos por cada método de diseño.
- Se creará un instalador, para facilitar la implementación de la aplicación en el equipo del usuario.

1.8. Limitaciones

A continuación, se presentan aquellas características las cuales no abarcará la aplicación.

- Sólo se incluirán los métodos de diseño de mezclas de concreto mencionados anteriormente.
- La aplicación no incluirá diferentes tipos de usuarios, pues todos los usuarios tendrán acceso a las mismas funciones.

- La aplicación se ejecutará de manera local y no tendrá que conectarse a Internet.
- Solo exportará los informes, tablas y gráficas a formato PDF.
- Únicamente se desarrollará una aplicación de escritorio, por lo que no se creará una versión web ni una versión para dispositivos móviles.
- El módulo de comparación solamente mostrará el resultado obtenido de cada método y el usuario se encargará de realizar el análisis correspondiente.
- El instalador será creado únicamente para el sistema operativo Windows, aunque al ser desarrollada en java la aplicación podría funcionar en otros sistemas operativos como Linux y MacOS.

1.9. Organización de la tesis

La presente tesis está conformada por seis capítulos. En el Capítulo I se describe el problema, se establece la pregunta de investigación, los objetivos generales, los objetivos específicos, la justificación, los alcances y las limitaciones del proyecto, además, presenta el estado del arte correspondiente al proyecto. El Capítulo II presenta el marco teórico del proyecto: modelos de desarrollo de software, metodologías de desarrollo ágil, las herramientas de desarrollo utilizadas e introduce al diseño de mezclas de concreto. En el Capítulo III se presenta la propuesta metodológica utilizada y en el Capítulo IV se describe el desarrollo y aplicación de ésta. El Capítulo V muestra los resultados de las pruebas realizadas y por último, el capítulo VI presenta las conclusiones y trabajos futuros.

1.10. Estado del Arte

En los últimos años ha crecido el interés por diseñar mezclas de concreto de un modo más eficiente y rápido, esto debido a que el desarrollo de los métodos de diseño tradicionales suele ser bastante extenso. Una manera de agilizar de forma eficiente el diseño de mezclas es con el desarrollo de programas de cómputo. Sin embargo,

este área aún no ha sido ampliamente explotada, ya que existen en el mercado pocos programas de cómputo enfocados al diseño de mezclas. El presente estado del arte revisa algunos ejemplos de la bibliografía disponible relevante para el desarrollo de esta tesis. En la primera sección se presenta un panorama general de los programas de cómputo actuales para el diseño de mezclas, en la segunda sección se presenta información referente a la ingeniería de software actual y por último, en la tercera sección se presenta bibliografía relacionada al diseño de mezclas de concreto.

1.10.1. Programas de cómputo actuales

Actualmente existen diversos programas de cómputo creados para el diseño de mezclas de concreto, pero la mayoría solo trabaja con uno o dos métodos o no son completamente enfocados al diseño de mezclas de concreto. A continuación, se presenta una breve descripción de los principales programas analizados.

Dimezco 2000 es un programa que permite diseñar mezclas de agregados, concretos y morteros, además permite realizar evaluaciones estadísticas de los resultados (Dimezco 2000, 2013). Actualmente se encuentra en su versión 8.0.0 publicada en julio de 2017 y está disponible de forma gratuita. En cuanto el diseño de mezclas de concreto este programa permite trabajar bajo los métodos ACI 211.1, Walker y Método Vitervo O'Reilly, pero no muestra los pasos que se realizaron para obtener los resultados

Bar-Dos es un programa, desarrollado en 2012 por Francisco Javier Bardisa Mollá, miembro del ICITECH (Instituto de Ciencia y Tecnología del Hormigón) en España. El software se encuentra de forma gratuita y realiza el diseño de mezclas bajo el método ACI 211.1, aunque no muestra los pasos realizados ni muestra gráficas representativas de los resultados.

Conmixer v1.0 es otro programa enfocado al diseño de mezclas de concretos, utiliza tres métodos IS (en inglés *Indian Standard*, es decir, Estándar Indú), ACI y DOE (en inglés *British Department of the Environment Method*, es decir, Método Británico del Departamento de Medioambiente). *Conmixer* se encuentra forma gratuita y no necesita instalación (Rojas, 2008).

DM-CONCRET es un programa desarrollado en 2014 por Danilo Saavedra de la Universidad Tecnológica de los Andes de Perú. Realiza el diseño de mezclas mediante

el método ACI 211.1 y para su funcionamiento es necesario tener instalado Microsoft Office. También permite exportar los resultados a formato de Excel, aunque algunas de sus características no vienen incluidas en la versión gratuita. En la tabla 1.1, se puede observar una comparativa de las características de los programas analizados.

Tabla 1.1: Comparativa de programas de cómputo para el diseño de mezclas de concreto.

Software	Gratuito	Versión completa	Métodos	Notas
Dimezco 2000	Sí	No	ACI 211, Walker y Vitervo O'reilly.	Última versión lanzada en julio de 2017.
DM-CONCRET	Sí	No	ACI 211.	Necesita Microsoft Office.
Bar-Dos	Sí	Sí	No hay método específico.	Versión para estudiantes con gran cantidad de herramientas de dosificación.
Comixer	Sí	Sí	IS, ACI y DOE.	Enfocado a normas europeas.

1.10.2. Ingeniería de software

La ingeniería del software según B. Boehm (1976) es "la aplicación práctica del conocimiento científico en el diseño y construcción de programas de computadora y la documentación asociada requerida para desarrollar, operar y mantenerlos". Los lineamientos de la ingeniería de software cambian constantemente por lo que es importante mantenerse al día con las más recientes propuestas. A continuación, se presentan ejemplos de la bibliografía relevante acerca de este tema.

1. Singhto y Dentawattana (2016) presentaron el artículo llamado "*An Experience in Blending the Traditional and Agile Methodologies to Assist in a Small Software Development Project*" en donde se ofrece un marco de referencia e información relevante para la correcta integración y combinación de metodologías tradicionales con metodologías ágiles. A continuación, se presentan las ideas principales del artículo.

   Las metodologías ágiles son enfoques de desarrollo rápido que apuntan hacia la rapidez y eficiencia. *Scrum* es la metodología de desarrollo rápido más utilizada, como evidencia Porrawatpreyakorn (citado por Singhto y Dentawattana, 2016)

presentó los resultados de una encuesta realizada en línea acerca del uso de las metodologías ágiles durante octubre de 2012 y marzo de 2013. Los resultados fueron que *Scrum* es la metodología más popular con 75 %, seguida de TDD (*Test-driven Development* o Desarrollo guiado por pruebas), XP y un híbrido entre *Scrum* y XP.

Las metodologías tradicionales son un proceso de desarrollo secuencial en las cuales cada etapa debe ser completamente terminada antes de avanzar a la siguiente. Las principales metodologías tradicionales son: el modelo en cascada, el modelo incremental, el modelo en V y el modelo en espiral.

Debido a la importancia de elegir la metodología correcta para un proyecto de desarrollo de software, en el artículo también se presenta un *checklist* desarrollado por Singhto y Dentawattana para seleccionar la metodología apropiada. Incluye preguntas relacionadas a los requerimientos, a los usuarios, al proyecto, al equipo de desarrollo y a la documentación requerida.

Por último, se muestra la experiencia propia de los autores al combinar una metodología tradicional con una metodología ágil, en la cual decidieron combinar la metodología *Scrum* con el modelo en cascada. Se combinaron las metodologías y se rediseñaron sus funciones para adaptarlas al proyecto.

II. Trivedi y Sharma (2013) presentaron el artículo llamado "*A Comparative Study between Iterative Waterfall and Incremental Software Development Life Cycle Model for Optimizing the Resources Using Computer Simulation*" en donde se muestra una comparación entre el modelo en cascada y el modelo incremental para la optimización de recursos. Las ideas principales del artículo se presentan a continuación.

   En ocasiones los gerentes del proyecto no saben medir cuantos empleados son suficientes para una fase en particular hasta el resultado. Por lo tanto, el costo, los excesos presupuestados y la calidad son difíciles de lograr. Se utilizo la herramienta de simulación Simphony.NET para determinar el recurso optimizado para un modelo de proceso de software en particular. El modelo en cascada es más fácil de simular porque sus fases se ejecutan unas tras otras, pero en el modelo incremental las fases dependen del gerente del proyecto.

Se simuló el ciclo de vida del modelo incremental y modelo en cascada. Estos modelos fueron ejecutados cinco veces durante 1825 milisegundos con cien proyectos entrantes utilizando la función de distribución matemática y el entorno Simphony.NET con el objetivo de comparar ambos modelos para comparar recursos y determinar mediante simulación el tamaño del equipo óptimo.

III. Orjuela Duarte y Rojas (2008) presentaron el artículo "*Las Metodologías de Desarrollo Ágil como una Oportunidad para la Ingeniería del Software Educativo*". En este artículo se divide en dos partes principales la primera de ellas presenta un marco teórico acerca de las metodologías de desarrollo ágil más utilizadas, como: XP, CRYSTAL y SCRUM. La segunda parte se enfoca en el análisis de las características de dichas metodologías y su aplicación al contexto del desarrollo de software educativo.

Anteriormente las aplicaciones de software educativo surgían de esfuerzos de docentes con algunos conocimientos en informática o de ingenieros con algún interés en el sector educativo, por lo que las metodologías ocupadas no eran siempre las mejores. Este artículo propone el estudio de las metodologías ágiles para generar metodologías menos pesadas. Las metodologías propuestas son:

a) *Programación extrema (Extreme Programming, XP).* XP es una metodología ágil centrada en potenciar las relaciones interpersonales como clave para el éxito en desarrollo de software, promoviendo el trabajo en equipo, preocupándose por el aprendizaje de los programadores, y propiciando un buen clima de trabajo.

b) *Crystal.* Se trata de un conjunto de metodologías para el desarrollo de software caracterizadas por estar centradas en las personas que componen el equipo y la reducción al máximo del número de artefactos producidos. El desarrollo de software se considera un juego cooperativo de invención y comunicación, limitado por los recursos a utilizar.

c) *SCRUM.* Es un modelo de referencia que define un conjunto de prácticas y roles, y que puede tomarse como punto de partida para definir el proceso de desarrollo que se ejecutará durante un proyecto.

iv. Estela Gottberg de Noguera, Gustavo Noguera Altuve y María Alejandra Noguera Gottberg (2011) presentaron el artículo "*Propuesta pedagógica: Una metodología de desarrollo de software para la enseñanza universitaria*". El objetivo de este trabajo es describir una metodología para desarrollar software educativo. Se presentan de manera detallada los elementos asociados con la metodología de desarrollo de software. La metodología propuesta en este artículo es el resultado de la mezcla de dos metodologías para adaptarlas a necesidades reales. Las etapas de las metodologías planteada son:

Fase 1. Análisis de necesidades y factibilidad. En esta fase se realiza un estudio que contempla el análisis de todos los elementos que influyen en el diseño del software educativo, los cuales están referidos al análisis del público, del ambiente, del contenido y del sistema.

Fase 2. Diseño instructivo (guion educativo, diseño funcional). Esta fase consiste en organizar toda la estructura del contenido educativo, la cual está formada por la meta educativa, los objetivos de aprendizaje, las decisiones acerca de los contenidos a desarrollar en el software (expuesta en la fase de análisis) y el prototipo en papel.

Fase 3. Desarrollo. Considerando la estructura de las pantallas que conforman el software educativo, se procede a una serie de formas para mostrar el funcionamiento general del mismo, conforme a las especificaciones de la fase de diseño.

v. Chaparro Lemus y Gómez Estupiñan (2012) presentaron el artículo "*Una visión del desarrollo de software utilizando modelos*" donde mencionan que "desarrollar software es una tarea complicada sobre todo cuando se trata de software demasiado complicado. Esta tarea exige un alto compromiso del equipo de desarrollo, recursos costosos, especialistas altamente cualificados, procesos y métodos cada vez más formales". Con el propósito de agilizar este proceso, surgió un movimiento centrado en el uso de modelos en diferentes niveles de abstracción. Las principales propuestas en este sentido son la Arquitectura Dirigida por Modelos (MDA, por sus siglas en inglés) y el Desarrollo de Software Dirigido por Modelos (MDSD, por sus siglas en inglés).

Aunque MDA y MDSD son metodologías diferentes comparten el mismo propósito: la utilización de modelos para desarrollar software. Mientras que MDSD se ocupa la utilización de los modelos para el desarrollo de software, MDA se enfoca en definir los estándares para desarrollar software haciendo uso intensivo modelos. Se pudiera decir que MDA es la estandarización de MDSD. Este artículo presenta un marco de referencia que ayuda a la implementación de ambas metodologías.

VI. Yagüe, Garbajosa, Díaz y González (2016) presentaron el artículo "*An exploratory study in communication in Agile Global Software Development*". A continuación, se describen las características principales del artículo.

El desarrollo de software global (GSD, por sus siglas en inglés: Global Software Development) cada vez está ganando más importancia. El desarrollo de software multisitio presenta muchos obstáculos tales como el uso de diferentes usos horarios, diferentes culturas e infraestructura de tecnologías de la información, lo que provoca retrasos en el momento de enviar comunicación, lo cual es realmente problemático. Este tipo de comunicación se vuelve incluso más problemática cuando se utiliza el modelo de Desarrollo Ágil de Software Global (AGSD, por sus siglas en inglés: *Agile Global Software Development*), en el cual la comunicación juega un papel primordial.

Este artículo representa los resultados de una investigación exploratoria enfocada al impacto de la comunicación en la infraestructura AGSD. El artículo propone un tipo de infraestructura que soporte la infraestructura AGSD. Además, analiza las ventajas y diferentes entre la infraestructura AGSD y GSD. La infraestructura provista en este estudio propone un buen punto de partida para crear un equipo de trabajo donde los miembros del equipo puedan comunicarse fluidamente en entornos globales y distribuidos.

VII. Mitre Hernández, Ortega Matínez y Lemus Olalde (2014) presentaron el artículo "*Estimación y Control de Costos en Métodos Ágiles para Desarrollo de Software: un Caso de Estudio*". A continuación, se presentan las ideas principales del artículo.

El desarrollo de software utilizando métodos ágiles está en crecimiento debido

a la productividad asociada a estas metodologías, además de la flexibilidad demostrada para equipos pequeños. Sin embargo, estas metodologías cuentan con debilidades claras de estimación y gestión de costos de desarrollo, al igual que los administradores de proyectos no cuentan con las suficientes evidencias para la comprobación del gasto del presupuesto en un proyecto debido a la poca documentación generada y por la falta de seguimiento en el gasto de los recursos.

En este artículo se presenta un método de medición para la estimación, control y gestión de costos y esfuerzos en equipos de desarrollo de software ágil con la finalidad de resolver los problemas encontrados en la literatura: una escasa gestión y monitoreo de costos, poca evidencia acerca de la medición de los costos de un proyecto para los administradores y falta de estimación de costos en métodos ágiles basada en procesos repetibles.

Este trabajo ofrece tres tipos de gráficas para apoyar los problemas mencionados: la gráfica TSPCA (total de cambios en el alcance) para el control de cambios realizados en el proyecto y no sobrepasar el tamaño del proyecto de software. La gráfica CPI/SPI (índice de desempeño del costo/índice del desempeño del calendario) para el control de costo por SP (*Story Points*) y calendario, y la gráfica *Burndown* que permite representar la replanificación de cada iteración, además de la técnica de estimación de esfuerzo con datos históricos de SPs.

VIII. Tomas, Escalona y Mejias (2013) presentaron el artículo "*Open source tools for measuring the internal Quality of Java Software products. A survey*". Este artículo ofrece una recopilación de métricas e indicadores para evaluar objetivamente los diferentes productos resultantes durante el ciclo de vida de un proyecto de software. La cual es un área de investigación que abarca muchos aspectos diferentes, además de ser muy demandada por empresas y equipos de desarrollo de software.

Uno de los métodos más utilizados por los equipos de desarrollo para medir la calidad interna es el análisis estático del código fuente. Este artículo trabaja en esta línea y presenta un estudio de las avanzadas herramientas de software de código abierto que automatizan la recopilación de la información, en particular para los desarrollos en Java. Estas herramientas han sido comparadas de acuerdo

con ciertos criterios definidos en este estudio.

Dieciséis herramientas se analizan a lo largo de las secciones del artículo. La mayoría de ellas automatizan el cálculo de las métricas de calidad interna (adquisición de datos), siendo los colores de código, la complejidad y el tamaño del código los más comunes. *Sonar* y *Squale* son capaces de reunir datos para todas las categorías de métricas, mientras que las otras herramientas están más especializadas en un conjunto limitado de métricas. Existen 3 herramientas completas (*Xradar*, *Sonar* y *Squale*), que realizan la adquisición, análisis y presentación de datos. Estas herramientas son relativamente nuevas y se basan en herramientas más maduras para la adquisición de métricas.

IX. García Pacheco y García Matías (2008) presentaron el artículo "*A Methodology Based on Effective Practices to Develop Educational Software*". A continuación, se presentan las ideas principales del artículo.

El software educativo es uno de los pilares de los sistemas de enseñanza-aprendizaje a distancia que es utilizado como una herramienta para las generaciones futuras de estudiantes.

Las actividades para el desarrollo de software educativo son complejas porque el proceso está enfocado en la experiencia del desarrollador, como los aspectos técnicos de la Ingeniería de Software y la adquisición e implementación del conocimiento pedagógico. Dicho artículo propone la introducción de las *prácticas efectivas* en una metodología alternativa para desarrollar software educativo. La identificación de prácticas efectivas está enfocada a asegurar que el desarrollo del proceso sea conducido con eficacia y orientado a la supervisión pedagógica del proyecto. Las prácticas efectivas que se proponen proporcionan las bases de una metodología alternativa para desarrollar software educativo con el rigor necesario para desarrollar software comercial, esto permite obtener un proceso que se puede repetir con altos niveles de éxito en el área de la instrumentación electrónica, específicamente.

La metodología presentada en el artículo agrupa las practicas efectivas en tres conjuntos.

1. Gestión del proyecto. Como su nombre lo indica cubre las actividades de gestión del proyecto relacionadas a la planificación, gestión, monitoreo y control.
2. Gestión pedagógica. Este grupo cubre las actividades pedagógicas relacionadas a los requerimientos del proyecto.
3. Gestión de la información. Este grupo cubre las actividades relacionas al manejo de los datos. Se encarga de determinar los documentos y las plantillas estándar para controlar el proceso de desarrollo.

x. Cervantes Ojeda y Gómez Fuentes (2012) presentaron el artículo "*Taxonomía de los modelos y metodologías de desarrollo de software más utilizados*". A continuación, se presentan las principales ideas del artículo.

   A través de una recopilación y análisis de los principales modelos de desarrollo de software existentes, el artículo propone una taxonomía que se integra con el fin de facilitar la elección de un modelo apropiado para cada proyecto en particular. Una buena elección de modelo (correctamente aplicado) ahorra tiempo y mejora la calidad de los sistemas que se producen. Sin embargo, la amplia variedad de modelos y metodologías en el mundo del desarrollo de software dificulta esta elección. La taxonomía propuesta presenta un panorama general de los modelos y metodologías más aceptados y los agrupa en categorías. Se discuten las características más representativas de cada una de estas categorías.

   Se propone la clasificación de los modelos y metodologías más citados en la literatura en cinco clases abstractas: Cascada, Evolutivos, Minimización de Desarrollos, Híbridos y Ágiles. Se divide en Modelos Tradicionales (también llamados pesados), que son los que promueven la disciplina por medio de la planificación y la comunicación escrita, y las Metodologías Ágiles, que dan prioridad a la interacción entre los individuos y a la comunicación con el cliente. Las modelos de desarrollo mostrados son:

   - En cascada: pura, con fases solapadas, con subproyectos y con reducción de riesgos.
   - Evolutivos: espiral, entrega por etapas o incremental, entrega evolutiva o iterativo, diseño por planificación y cascada en V.

- Minimización de Desarrollos: componentes reutilizables y diseño por herramientas.
- Híbridos: Proceso Unificado Racional.
- Metodologías Ágiles: Programación Extrema, SCRUM, desarrollo dirigido por pruebas, desarrollo dirigido por características Agile, Lean y Cristal.

XI. Mohammed, Niazi, Alshayeb y Mahmood (2017) presentaron el artículo "*Exploring software security approaches in software development lifecycle: systematic mapping study*" (en español, exploración de enfoques de seguridad en el ciclo de vida del desarrollo de software). A continuación, se describen las ideas principales del artículo.

Existe un mayor uso de enfoques basados en la seguridad para apoyar las actividades de desarrollo de software, tales como requisitos, diseño e implementación. El objetivo de este artículo es identificar los enfoques existentes de seguridad de software utilizados en el ciclo de vida del desarrollo de software (SDLC, por sus siglas en inglés: *Software Development LifeCycle*). Con el fin de cumplir con el objetivo de la seguridad, se llevó a cabo un estudio sistemático de mapeo para identificar los estudios primarios sobre el uso de técnicas de seguridad de software en SDLC.

Después de analizar los estudios seleccionados, se identificaron 52 enfoques de seguridad y se clasificaron en cinco categorías principales:

1. Modelado de requisitos seguros.
2. Identificación de vulnerabilidad, adaptación y mitigación.
3. Proceso enfocado en la seguridad de software.
4. Perfiles de modelado seguro.
5. Modelado seguro no basado en UML.

Los resultados muestran que los enfoques más utilizados son el análisis estático y el análisis dinámico que proporcionan controles de seguridad en la fase de codificación. Además, los resultados muestran que muchos estudios en esta revisión consideraron los controles de seguridad alrededor de la etapa de codificación del desarrollo de software.

1.10.3. Diseño de mezclas de concreto

Con el fin de mejorar las propiedades del concreto resultante al diseñar mezclas de concreto, constantemente están surgiendo nuevos métodos para optimizar la dosificación de mezclas. Por lo tanto, es importante mantenerse informado de las nuevas propuestas de diseño de mezclas de concreto. A continuación, se presentan ejemplos de bibliografía relevante respecto a este tema.

1. Rojas (2012) presenta una recopilación de diversos programas de computadora para del diseño de mezclas de concreto. A continuación, se presenta un breve resumen de los programas mostrados.

    a) Dimezco 2000. El sistema Dimezco 2000 es un programa para diseñar mezclas de agregados, concretos, morteros y realizar evaluaciones estadísticas de los resultados de ensayo de resistencia en compresión del concreto. Es un sistema diseñado para ser utilizado en la industria de la construcción. Utiliza dos métodos para evaluar las mezclas de concreto: el método ACI 211.1 y el método ACI 211.4. La versión 8.0.0 fue lanzada en julio de 2017.

    b) DM–CONCRET v1.0.0. Este programa para ingeniería civil simplifica los cálculos en el diseño de mezcla por el método de la ACI y tiene la propiedad de modificar todos los datos, aunque algunas funcionalidades no vienen incluidas en la versión de prueba. Para que pueda funcionar el programa correctamente es necesario que tenga instalado Microsoft Office, en caso contrario no instalara o no exportara. Permite exportar a formato Excel.

    c) Programa de Dosificación de Hormigones BAR-DOS (Versión para estudiantes). Es un programa diseñado para estudiantes, se encuentra de forma gratuita y en español. Este programa permite medir la resistencia, consistencia, la relación agua-cemento y la proporción de áridos. Cuenta con una interfaz bastante sencilla para el uso de los estudiantes.

    d) Conmixer v1.0. Es un programa en inglés, que no necesita instalarse, ya que puede ejecutarse directamente. La ventaja de este programa es que contiene tres métodos para la evaluación de concretos: IS, ACI y DOE. Está enfocado principalmente a la dosificación de mezclas utilizando normas europeas.

II. Kosmatka, Kerkhoff, Panarese y Tanesi (2004) publicaron el libro "*Diseño y Control de Mezclas de Concreto*" este libro ofrece un panorama general acerca del diseño y control de las mezclas de concreto. Contiene temas como; los fundamentos del concreto, es decir los componentes, características y diferentes estados de este, así como los tipos de cementos que existen. Además, ofrece una amplia descripción de las características de los componentes que intervienen en la mezcla de concreto, tales como: el agua de mezcla, los agregados, aditivos para concreto, fibras y aire incluido en el concreto.

   El capítulo 9 del libro, es acerca del diseño y proporcionamiento de mezclas de concreto normal (sin aditivos). En este capítulo se habla acerca del procedimiento en general para el diseño de mezclas de concreto y de algunos métodos en específico. Los métodos incluidos en el libro son el Método del Volumen, el Método del Volumen Absoluto y el método de la PCA (*Portland Cement Association*) de Relación Agua-Cemento.

III. Choquechambi Mamani, Custisaca Bellido y Quisque Galindo (2013) alumnos de la Universidad Peruana Unión Facultad Ingeniería y Arquitectura presentaron la investigación denominada "*Comparación de 4 métodos para el diseño de mezclas*". Esta investigación tiene la finalidad de hacer una evaluación comparativa en cuanto a resistencia y costo de cuatro métodos de diseño de mezcla de concreto. Los métodos evaluados son:

   - Módulo de Fineza de la Combinación de los Agregados.
   - Método de Füller.
   - Método de Walker.
   - Método ACI.

   La finalidad del estudio es poder determinar cuál método resulta ser más resistente y económicamente viable a razón de un metro cúbico de concreto. Con los resultados de los cuatro métodos aplicando diferentes tipos de pruebas y ensayos y tras siete días de observación con base en el promedio de resistencia se llegó a la conclusión de que el Método de Füller es el más eficiente. Sin embargo, es

también este método el que ocupa mayor cantidad de cemento, por lo que la elección final sigue quedando en función de los requerimientos de la obra.

IV. Han, K. Wang, X. Wang y Monteiro (2016) presentaron el artículo "$2D$ *image analysis method for evaluating coarse aggregate characteristic and distribution in concrete*". Este artículo trata acerca del uso de imágenes 2D como un método de análisis para la evaluación de las características y distribución del agregado grueso en mezclas de concretos. Esto se realiza mediante el procesamiento de fotografías donde se analiza si la mezcla de concreto es correcta.

Una mezcla de concreto bien realizada debe contener el agregado grueso completamente cubierto por la pasta, es decir la combinación entre el agua y cemento. Si el agregado grueso se conecta entre sí, se considera que no es un concreto de calidad.

El uso de este método determina si el tamaño, la forma y la orientación del agregado grueso o grava son los correctos. Además de la relación existente entre ambos. El método de análisis por imágenes 2D puede ser utilizado para evaluar la calidad del concreto, optimizar la proporción de la mezcla y evaluar métodos basados en la información obtenida del análisis de las imágenes.

V. Miller, Monteiro, Ostertag y Horvath (2016) presentaron el artículo llamado *Comparison indices for design and proportioning of concrete mixtures taking environmental impacts into account*, en español: "Índices comparativos para el diseño y proporción de mezclas de concreto tomando en cuenta los impactos ambientales". Este artículo trata sobre la creciente preocupación por las emisiones de gases de invernadero y sobre los nuevos métodos de mezclas de concreto que existen para reducir tales efectos. Ya que una de las principales fuentes de los gases que causan tales efectos es la producción industrial del cemento para su uso en concretos. Las evaluaciones de estos métodos por lo general no toman en cuenta el impacto que tiene en el ambiente el uso del cemento y sus propiedades. Cuando ambas se consideran, rara vez se evalúan características diferentes a las de la resistencia del concreto.

En este artículo, se presentan métodos para evaluar los impactos ambientales con propiedades mecánicas simultáneamente para el diseño de concreto no ar-

mado. Los índices se aplicaron para comparar varios diseños de concreto con diferentes componentes, relaciones de mezcla y con o sin materiales cementosos suplementarios (MCS).

Se llegó a la conclusión de que el potencial de calentamiento global producido por el concreto no es una función de la resistencia a la compresión del mismo. Además, se recomienda el uso de materiales de desecho en la elaboración del concreto, con el fin de reducir la contaminación generada.

1.10.4. Conclusiones

Después de realizar una investigación sobre metodologías de desarrollo de software, se puede observar que la mayoría de las metodologías actuales están enfocadas al desarrollo ágil, como resultado cada vez son menos utilizados los modelos tradicionales como: el modelo en cascada, el modelo en V, el modelo en espiral, el incremental, etc. Sin embargo, es común encontrar modelos híbridos que mezclan metodologías agiles (como *Extreme Programming*, *Crystal* y *Scrum*) con metodologías tradicionales.

En la práctica es muy importante elegir la correcta metodología de desarrollo, ya que en gran medida de esto depende el éxito del proyecto. La metodología de desarrollo debe seleccionarse en base a las características propias del proyecto, determinando cual es la que mejor se adapta a las necesidades.

En cuanto al diseño y proporcionamiento de mezclas de concreto, actualmente existe una gran cantidad de métodos para el diseño mezclas de concreto, los cuales contemplan el uso de diferentes elementos y condiciones de trabajo. Cada vez surgen más técnicas para el diseño de mezclas de concreto. Entre los principales métodos de diseño se encuentran:

- Método ACI
- Método Walker
- Método Füller
- Método Bolomey

Estos métodos se basan en un procedimiento general para el diseño de mezclas, con algunas variantes de un método a otro. Existen métodos que pueden ser aplicados solo a pequeñas construcciones, otros que pueden ser aplicados a grandes construcciones como puentes o edificios y otros que pueden ser aplicados a ambos y más tipos de construcciones. Por lo general, realizar este tipo de métodos es una tarea larga, y cansada debido a los largos procedimientos, por lo que el uso de una aplicación resulta más eficiente.

Un problema que se presenta es que existen pocas aplicaciones que trabajan bajo estos métodos de diseños de mezclas. En su mayoría solo trabajan con uno o dos métodos, además de que no muestran paso a paso su desarrollo. A pesar de que la mayoría de estas aplicaciones son gratuitas, se encuentra información muy vaga sobre ellas y la mayaría son bastante antiguas.

En el contexto didáctico, surgen diversos problemas al enseñar estos métodos a los alumnos, ya que por su complejidad no hay una forma rápida de comprobar los resultados. Ninguna de las aplicaciones investigadas muestra el procedimiento del método paso a paso, por lo que no son de mucha utilidad para que el alumno pueda verificar en que paso pudiera haber cometió un error. Debido a que tampoco muestran tablas comparativas de los resultados de los métodos, no son de gran utilidad para que el usuario pueda analizar cuál de los métodos es el más indicado para la obra.

El propósito de la aplicación propuesta es el de cubrir los problemas que se han encontrado tras analizar el Estado del Arte. Se propone una aplicación que permita desarrollar los principales métodos de diseño de mezclas de concreto, que permita mostrar paso a paso el desarrollo de los métodos, para que así el usuario pueda verificar los resultados obtenidos con los de sus sus prácticas y por último, cubrirá la necesidad del usuario de comparar los datos obtenidos por los diferentes métodos, mediante un módulo comparativo que incluirá tablas y gráficas comparativas.

Capítulo 2

Marco Teórico

En este capítulo se presenta una revisión documental de los principales modelos y metodologías de desarrollo de software, de las herramientas de desarrollo utilizadas, así como de los fundamentos básicos del diseño de mezclas, su desarrollo uso y aplicación. De igual forma, se realiza una breve descripción de los métodos incluidos en la aplicación propuesta.

2.1. Modelos de desarrollo de software

Los modelos de desarrollo también conocidos como metodologías pesadas o metodologías tradicionales son aquellas que se basan en una serie de fases secuenciales tales como: análisis, diseño, implementación y pruebas (Ben-Zahia y Jaluta, 2014). Por lo general estos modelos se caracterizan por ser poco flexibles y muy rígidos en cuanto a la documentación que se debe generar en cada etapa. Por lo que su implementación se vuelve una tarea muy ardua, en especial para equipos pequeños. A continuación, se describen brevemente algunos de los modelos de desarrollos más conocidos.

2.1.1. Modelo en cascada

La versión original del modelo en cascada fue presentada por Royce en 1970, aunque son más conocidos los refinamientos realizados por Boehm en 1981, Sommerville

en 1985 y por Sigwarten 1900 (Cataldi *et al.*, 1999). A pesar de que este modelo fue presentado hace mucho tiempo es todavía utilizado en la actualidad.

El modelo se consiste en una serie de fases secuenciales. Es decir que cada fase debe ser completada antes de avanzar a la siguiente y que no se puede avanzar sin que la fase previa esté realizada. Una vez que se ha dado por completada una fase, no se permite regresar a dicha fase. Las fases que componen el modelo en cascada son: requisitos, diseño, implementación, verificación y mantenimiento. En la figura 2.1, puede verse el proceso del modelo en cascada.

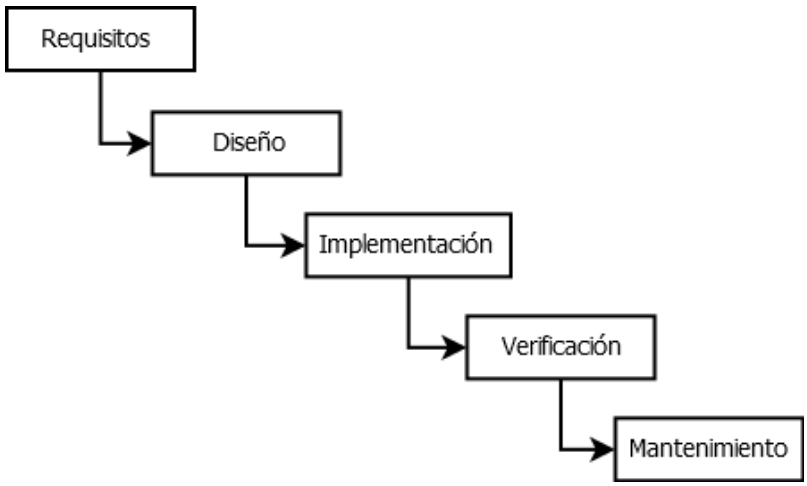


Figura 2.1: Modelo en cascada. Fuente: elaboración propia.

2.1.2. Modelo en espiral

El modelo en espiral fue propuesto por Barry Boehm en el año de 1986. Está basado en el modelo en cascada y en el de prototipos. Las fases del modelo en espiral son similares a las del modelo en cascada, pero con la diferencia de que el modelo en espiral se compone de ciclos o iteraciones. Cada ciclo representa una versión o prototipo del software, la cual es presentada al cliente hasta que éste se encuentra totalmente satisfecho con el producto. El modelo en espiral es especialmente útil cuando los usuarios no están completamente seguros de sus necesidades al inicio del proyecto. En la figura 2.2, puede verse el proceso del modelo en espiral.

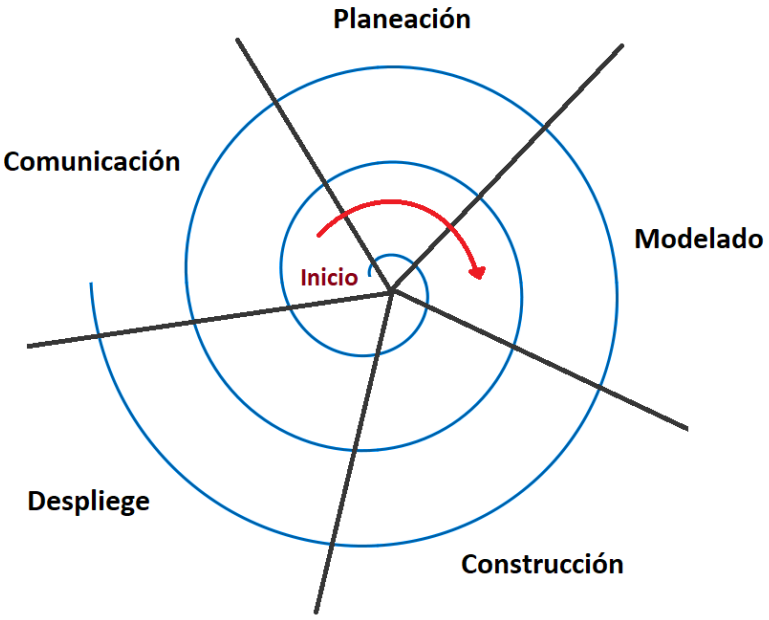


Figura 2.2: Modelo en espiral. Fuente: elaboración propia.

2.1.3. Modelo incremental

El modelo incremental fue introducido por Lehman en el año de 1984, para solucionar los problemas y defectos del modelo en cascada. La implementación del modelo incremental sigue el mismo orden que el de cascada, pero corrige la problemática de la linealidad del modelo en cascada (Cataldi *et al.*, 1999). El modelo se compone de iteraciones o ciclos, donde al final de cada iteración se van agregando nuevas funcionalidades que permiten ir refinando el software. En la figura 2.3, puede verse el proceso del modelo en espiral.

2.2. Metodologías de desarrollo ágil

Con el surgimiento de las metodologías ágiles se pretendía ofrecer una alternativa a los procesos de desarrollo tradicionales, caracterizados por ser rígidos y dirigidos por la documentación que se genera en cada una de las actividades (Orjuela Duarte y Rojas, 2008). En esencia, agilidad significa responder a los cambios de forma rápida y eficiente durante el desarrollo del software (Qureshi, 2012). A continuación, se

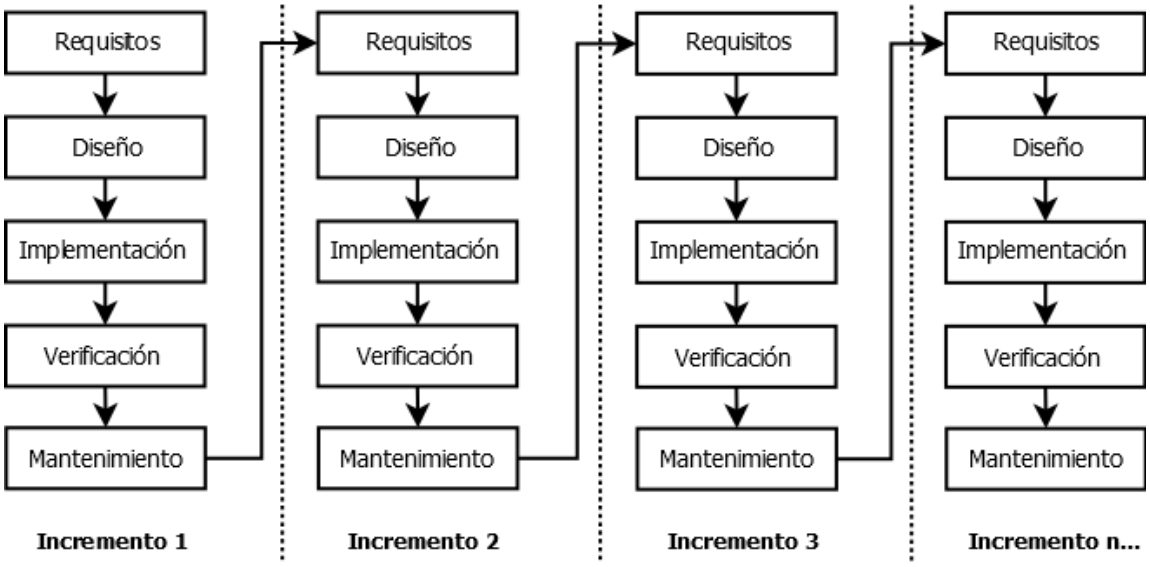


Figura 2.3: Modelo incremental. Fuente: elaboración propia.

describen brevemente algunas de las principales metodologías de desarrollo ágil.

2.2.1. Extreme Programming

La metodología *Extreme Programming* (XP) es la metodología de desarrollo ágil más ampliamente utilizada. Al igual que el resto de las metodologías ágiles XP comparte los valores del *Manifiesto Ágil*, pero va más allá al especificar un conjunto de prácticas simples (Lindstrom y Jeffries, 2004). Es una metodología centrada en potenciar las relaciones interpersonales como clave para el éxito en desarrollo de software, promoviendo el trabajo en equipo, preocupándose por el aprendizaje de los programadores y propiciando un buen clima de trabajo (Orjuela Duarte y Rojas, 2008). XP es una disciplina de desarrollo de software basada en valores de simplicidad, comunicación, retroalimentación y coraje (Lindstrom y Jeffries, 2004). Las características principales de XP pueden clasificarse en: historias de usuario, roles, proceso y prácticas (Beck, 1999). En la figura 2.4, puede verse el proceso de XP.

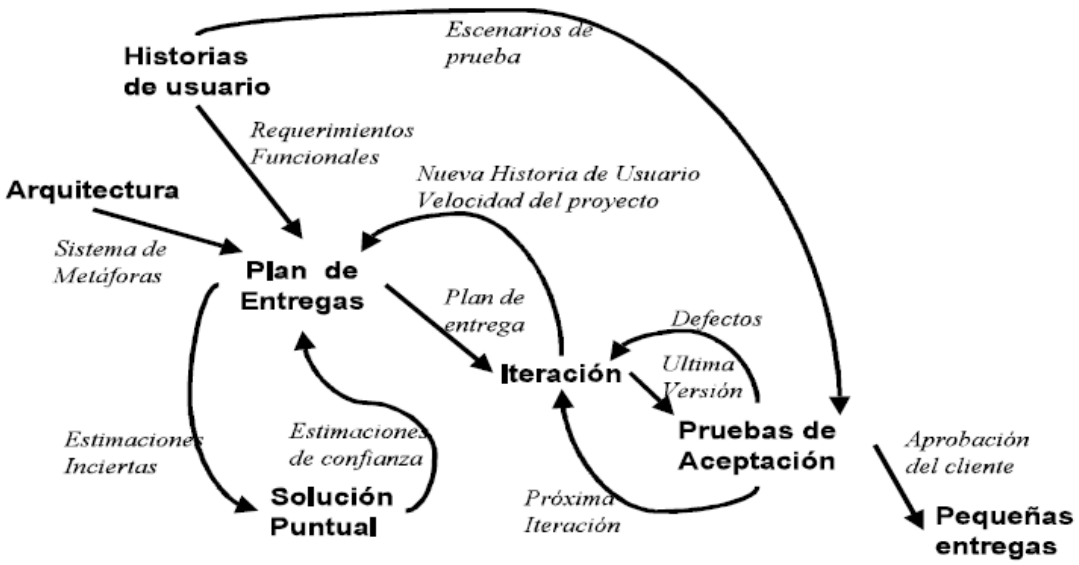


Figura 2.4: Proceso XP (Orjuela Duarte y Rojas, 2008).

2.2.2. Crystal

Crystal se compone de un conjunto de metodologías para el desarrollo de software caracterizadas por estar centradas en las personas que componen el equipo de trabajo y la reducción al máximo del número de artefactos producidos (Orjuela Duarte y Rojas, 2008). Hay cuatro variantes de la metodología Crystal cada una aplica diferentes políticas identificada por un color y se clasifican dependiendo del número de integrantes del equipo: Crystal Clear para equipos de 8 o menos, Amarillo para equipos de 8 a 20, Naranja para equipos de 20 a 50 y Rojo para equipos de 50 o más.

Las metodologías Crystal cumplen con las siguientes características: entregas frecuentes, mejora reflexiva, comunicación osmótica, seguridad personal, enfoque, fácil acceso a usuarios expertos y un entorno técnico con pruebas automatizadas.

2.2.3. SCRUM

Define un marco para la gestión de proyectos, que se ha utilizado con éxito durante los últimos años. Está especialmente indicada para proyectos con un rápido cambio de requisitos. Sus principales características se pueden resumir en dos (Orjuela Duarte y Rojas, 2008):

1. El desarrollo de software se realiza mediante iteraciones, denominadas *Sprint*, con una duración de 30 días. El resultado de cada *Sprint* es un incremento ejecutable que se muestra al cliente.

2. La otra característica importante son las reuniones a lo largo del proyecto, entre ellas destaca la reunión diaria de 15 minutos del equipo de desarrollo para coordinación e integración.

2.3. Herramientas de desarrollo

2.3.1. Java

A Java es un lenguaje de programación multiplataforma orientado a objetos que ofrece una arquitectura versátil, manejo de multihilos, es seguro y con un alto desempeño (Perea *et al.*, 2016). La principal característica de Java es que se trata de un lenguaje independiente de la plataforma, es decir, cualquier programa que sea creado en este lenguaje podrá ser ejecutado en todo tipo computadora y en distintos sistemas operativos como Windows, Solaris, Linux y MacOS. Esto gracias a la máquina virtual de Java (en inglés, *Java Virtual Machine*, JVM) que se encarga de interpretar y ejecutar el código de cualquier programa compilado en Java. Las principales características de java son:

- Orientado a objetos
- Distribuido y dinámico
- Robusto
- Seguro
- Multitarea, y
- Portable

El diseño de Java, su robustez, el respaldo de la industria y su fácil portabilidad han hecho de Java uno de los lenguajes con un mayor crecimiento y amplitud de uso en distintos ámbitos de la industria de la informática.

2.3.2. MySQL

MySQL es un sistema gestor de base de datos de código abierto (en inglés, *Database Management System*, DBMS) que trabaja bajo Windows y muchas versiones de UNIX. Puede ser distribuido de forma libre bajo una Licencia Pública General (en inglés, *General Public License*, GPL) siempre que distribuidor ponga a distribución el código fuente del programa junto con las versiones binarias (Harrington, 2003).

MySQL es la plataforma de base de datos de fuente abierta más confiable actualmente en uso. Muchos de los sitios web más populares y con mayor tráfico en el mundo se basan en MySQL debido a si ubicuidad en plataformas heterogéneas y pilas de aplicaciones, sumado a su conocido rendimiento, fiabilidad y facilidad de uso (Satoto *et al.*, 2016).

2.3.3. GUI Design Studio

GUI Design Studio es una herramienta desarrollada por *Caretta Software* para la creación de prototipos y diseño de interfaces de usuario. Es una aplicación del tipo *drag and drop*, es decir que permite seleccionar y arrastrar elementos sin la necesidad de escribir código, permite desarrollar prototipos de aplicaciones web, de escritorio, móviles y embebidas. Además, cuenta con una versión de prueba con todas sus funcionalidades durante 30 días.

2.4. Diseño de mezclas de concreto

2.4.1. Fundamentos del concreto

El concreto u hormigón es un material para la construcción ampliamente utilizado, debido a su resistencia, durabilidad y maleabilidad. El concreto es básicamente la mezcla de dos componentes: los agregados y la pasta. La pasta se compone de cemento y agua, es la encargada de unir los agregados, creando una masa similar a una roca (Kosmatka *et al.*, 2004). A su vez, los agregados se dividen en dos grupos: el agregado fino (arena) y el agregado grueso (grava). Adicionalmente, el concreto contiene un pequeño volumen de aire atrapado, y puede contener también aire intencionalmente incorporado mediante el empleo de aditivos.

Igualmente, en la mezcla de concreto también se utilizan con frecuencia otros aditivos para propósitos tales como acelerar o retardar el fraguado y el endurecimiento inicial, mejorar la trabajabilidad, reducir la cantidad de agua necesaria, incrementar la resistencia o modificar otras propiedades del concreto (Rivva, 1996).

En la figura 2.5, se puede observar un ejemplo de las proporciones utilizadas la dosificación del concreto.

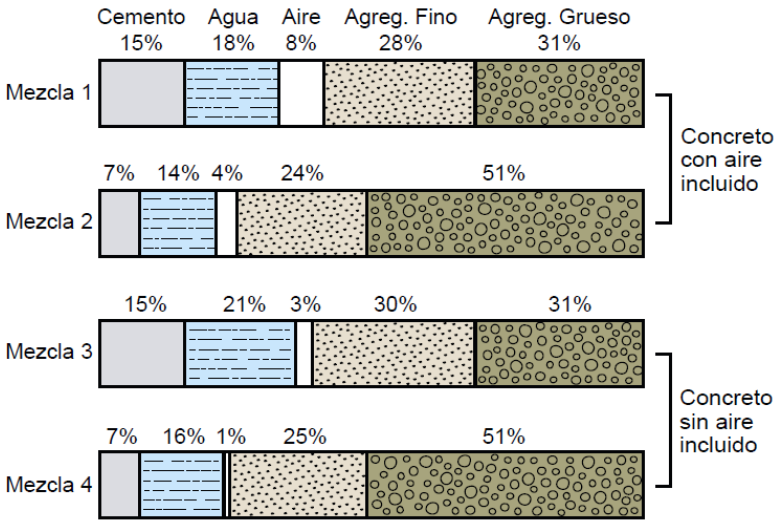


Figura 2.5: Proporciones utilizadas en el diseño de concreto (Kosmatka *et al.*, 2004).

2.4.2. Proporcionamiento de mezclas de concreto

La selección de las proporciones de los materiales que integran un metro cúbico de concreto, es conocida comúnmente como diseño o proporcionamiento de mezclas de concreto. Sin embargo, el diseño de mezclas de concreto también puede referirse únicamente al proceso mediante el cual se determinan las características deseadas en el concreto resultante y al proporcionamiento de mezclas de concreto únicamente como el procedimiento para obtener la cantidad de materiales necesarios para la elaboración del concreto. Algunos autores no hacen diferencia entre el proporcionamiento y el diseño de mezclas de concreto. Por lo tanto, se refieren indistintamente al proporcionamiento con el nombre de diseño de mezclas de concreto y viceversa.

Durante el proceso de diseño de la mezcla se seleccionan los ingredientes más

adecuados y la combinación más conveniente y económica de los mismos, con la finalidad de obtener un producto que en el estado no endurecido tenga la trabajabilidad y consistencia adecuadas; y que endurecido cumpla con los requisitos establecidos previamente por el diseñador o indicados en los planos y/o las especificaciones de obra (Rivva, 1996).

A medida que aumentan o disminuyen los diferentes componentes de la mezcla, el concreto resultante cambia de características. Por ejemplo:

- Si se aumenta la cantidad de cemento de la mezcla también aumenta la resistencia y durabilidad del concreto.
- Si se aumenta la cantidad de agua de la mezcla se obtendrá un concreto menos resistente. Es recomendable usar la menor cantidad de agua posible.
- Demasiado agregado fino da como resultado una mezcla pegajosa y el agregado se va al fondo de la mezcla
- Demasiado agregado fino da como resultado una mezcla áspera y pedregrosa.

La calidad del concreto resultante depende tanto de la calidad de los materiales, como de la correcta combinación de estos, por lo que realizar una correcta dosificación de los materiales es esencial para lograr un concreto de calidad. Un concreto adecuadamente proporcionado debe presentar las siguientes cualidades (Kosmatka *et al.*, 2004):

- Trabajabilidad aceptable del concreto fresco.
- Durabilidad, resistencia y apariencia uniforme de concreto endurecido.
- Economía.

Antes de que se puedan determinar las proporciones de la mezcla, se seleccionan sus características considerándose el uso que se proponer dar al concreto, las condiciones de exposición, tamaño, la forma de los elementos, etcétera.

2.4.3. Información necesaria

Para realizar el diseño de mezclas de concreto es necesario conocer, además de las propiedades que se requieren, de la finalidad del concreto y las propiedades ambientales a las que estará expuesto, información básica sobre las propiedades de los materiales que integran el concreto (Rivva, 1996). Las propiedades de los materiales disponibles es obtenida principalmente mediante pruebas de laboratorio. A continuación, se presentan la información más útil para un adecuado diseño de mezclas.

Cemento

Las principales características requeridas del cemento son:

- Tipo y marca del cemento disponible.
- Peso específico del cemento.
- Superficie específica del cemento.

Agua

Las principales características requeridas del agua son:

- Tipo de agua.
- Análisis clínico del agua.
- Efecto del agua sobre el fraguado.

Agregados

En el caso de los agregados, tanto para el fino como para el grueso es importante conocer:

- Perfil y textura del agregado.
- Análisis granulométrico.
- Peso específico de masa.

- Peso unitario y compacto.
- Porcentaje de absorción y de contenido de humedad.
- Perfil del agregado.
- Tamaño Máximo del Nominal.
- Módulo de fineza.

No todos los métodos hacen uso necesariamente de todas las características aquí presentadas.

2.4.4. Métodos de diseño de mezclas

Los métodos de diseño de mezclas son procedimientos que ayudan a determinar la dosificación más económica de los materiales disponibles para producir un concreto que cumpla con los requisitos establecidos.

Estos métodos de diseños se realizan mediante una serie de pasos y con tablas de referencias para cada paso, pero no se desarrollan de forma automática, ya que los datos de las tablas son sólo de referencia, por lo que es necesario seleccionar los datos con los cuales deseamos trabajar, muchas veces basado en la experiencia.

El primer paso para la dosificación de los elementos del concreto es determinar las características básicas de los ingredientes disponibles. Muchas de estas características se obtienen mediante pruebas de laboratorio realizadas a los materiales, la información básica que se debe obtener es: granulometría, peso específico de masa, porcentaje de absorción y contenido de humedad, tamaño máximo nominal, módulo de fineza, peso seco compactado, entre otros.

Por lo general los métodos de diseño de mezclas incluyen los siguientes pasos, aunque no precisamente en el mismo orden:

1. Selección del tipo de cemento.
2. Selección de los agregados y aditivos.
3. Selección del contenido de aire.

4. Estimación del contenido de agua.
5. Determinación de la relación agua-cemento.
6. Determinación del contenido de cemento.
7. Determinación del volumen del agregado grueso.
8. Determinación del volumen del agregado fino.
9. Ajustes en factor de la humedad.
10. Determinar la cantidad de los materiales por m^3 de concreto.

Cada vez que se requiera modificar alguno de los ingredientes es necesario volver a repetir todo el proceso de diseño. Existen varios métodos de diseño de mezclas, pero todos proporcionan resultados aproximados, estos resultados deben ser verificados mediante pruebas de laboratorio y de campo.

2.4.5. Métodos utilizados

- ACI. El comité 211 del ACI (*American Concrete Institute*) ha desarrollado un procedimiento de diseño de mezclas bastante simple el cual, basándose en algunas tablas proporcionadas por el mismo comité permite obtener los diferentes valores de diseño que integran una unidad cubica de concreto (Rivva, 1996).
- Walker. El denominado Método Walker se desarrolla debido a la preocupación del profesor norteamericano Staton Walker en relación con el hecho de que, sea cual fuera la resistencia de diseño del concreto y por tanto su relación agua-cemento, contenido de cemento y características del agregado fino, la cantidad de agregado grueso era la misma, ello cuando se aplicaba el procedimiento de diseño desarrollado por el Comité 211 del ACI (Rivva, 1996). Con el uso de este método de Walker se resuelve esta preocupación.
- Füller. Füller y Thompson desarrollaron una curva granulométrica continua para la composición óptima de los agregados. La combinación de esta curva con la granulometría de los agregados fino y grueso, permite obtener el volumen

absoluto de los agregados. Se recomienda utilizar este método cuando (Bolívar, 1987).

- La cantidad de cemento por m^3 de hormigón es mayor a 300 kgf.
- La estructura no está fuertemente armada.
- Los agregados son preferiblemente de forma redondeada.

- Bolomey. El procedimiento de diseño es muy similar al método Füller y en lo único que se diferencia es en la curva propuesta para combinar los agregados, ya que toma en cuenta el cemento. Bolomey propuso su propia curva granulométrica continua de agregado con el cemento como variable (Bolívar, 1987).

Capítulo 3

Propuesta metodológica

3.1. Introducción

La mayoría de las metodologías tradicionales de desarrollo de software están dirigidas a grandes equipos de desarrollo y a largos periodos de tiempo. Esto implica que, al aplicarse dichas metodologías a equipos pequeños, se presenten problemas principalmente de sobrecarga de trabajo, lo que también deriva en el incumplimiento de los plazos establecidos. Esto ocurre porque muchas de estas las metodologías incluyen una gran cantidad de documentación y requisitos, por lo que cumplir con todos estos requerimientos se convierte en una tarea extenuante para el equipo desarrollador, principalmente cuando se trata de equipos pequeños.

El principal objetivo de esta documentación es que otros miembros del equipo puedan adaptarse de forma más rápida a las diferentes etapas del proyecto. Sin embargo, al referirse a equipos pequeños de desarrollo en donde, debido a su reducido tamaño, todos los miembros están en constante comunicación, dicha documentación resulta en un gran esfuerzo que podría evitarse. La propuesta metodológica que se presenta en este capítulo trata de reducir al mínimo la documentación empleada y de enfocarse principalmente a la codificación del proyecto.

3.2. Descripción de la metodología propuesta

En este capítulo se presenta una metodología de desarrollo ágil que permite a equipos reducidos trabajar con proyectos de desarrollo de software enfocándose principalmente a la codificación y reduciendo al mínimo la documentación empleada. Esta metodología se basa en el modelo de desarrollo incremental y en la metodología de desarrollo *Extreme Programming* (XP), así se logra mayor agilidad en el proceso de desarrollo, iteraciones cortas en tiempo y una rápida adaptación a los cambios de requisitos hechos por el cliente.

De la metodología XP se retoman aspectos que permiten lograr una retroalimentación continua entre el usuario y el equipo de desarrollo. Mientras que el modelo de desarrollo incremental se retoman conceptos para poder crear y agregar módulos al proyecto como se vayan desarrollando.

3.3. Características del proyecto

No siempre se ha de aplicar la misma metodología para todo tipo de proyectos, sino que es conveniente analizar las necesidades para determinar cuál es la más apropiada (Alfonso *et al.*, 2011). Al seleccionar la metodología adecuada se debe asegurar de que las características del proyecto se adapten a las de la metodología a utilizar. A continuación, se describen algunas características con las que debe cumplir el proyecto, para que esta metodología obtenga mejores resultados:

- Equipo de desarrollo pequeño. Esta metodología está enfocada a equipos pequeños de desarrollo. Se considera pequeño a equipos de 1 a 5 miembros.

- Plazo de tiempo reducido. Cuando el proyecto tiene un tiempo de entrega reducido, se debe elegir una metodología que permita garantizar que el proyecto se termine en tiempo y forma sin dejar de lado la calidad del proyecto.

- Constante interacción con el usuario. Al permitir mayor comunicación del usuario con el equipo desarrollador se agiliza el proceso de desarrollo. Se considera como interacción a todas las reuniones que tiene el usuario o cliente con el equipo desarrollador.

- Requisitos volátiles. Cuando el usuario estará en contacto frecuente con la aplicación, los requisitos de la aplicación también cambiarán de forma constante. Esta metodología ayuda a adaptarse de forma rápida a los cambios de requisitos.

- Tamaño del proyecto. Para obtener mejores resultados al aplicar esta metodología se recomienda que el tamaño del proyecto vaya de pequeño a mediano. Si se aplica esta metodología a grandes proyectos es probable que el equipo se vea envuelto en una tarea demasiado extenuante.

3.4. Características de la metodología propuesta

Las características de la metodología se dividen en: historias de usuario, bitácoras, roles y procesos. Estas características fueron retomadas de la metodología XP y del modelo incremental, por considerarse ampliamente aplicables a equipos pequeños. A continuación, se describe cada tipo de características y sus componentes.

3.4.1. Historias de usuario

Las historias de usuario hacen referencia a los datos obtenidos tras aplicar técnicas utilizadas para la obtención de los requisitos, sin importar que estos sean funcionales o no funcionales. Cada historia de usuario debe ser lo suficientemente entendible y lo suficientemente delimitada para que su implementación no tome demasiado tiempo.

Por otra parte, cada historia de usuario identificada debe ir redactada en un documento, para el desarrollo de esta metodología se sugiere que dicho documento contenga los siguientes elementos: fecha, tipo de actividad (nueva, corrección, mejora), prueba funcional, número de historia, prioridad técnica y del cliente, referencia a otra historia previa, riesgo, estimación técnica, descripción, notas y una lista de seguimiento con la fecha, estado, cosas por terminar y comentarios (Orjuela Duarte y Rojas, 2008).

Sumado a estos elementos pueden agregarse otros como: nombre de la historia de usuario, dependencias relacionadas (es decir, aquellas historias de usuario que deben de completarse antes de realizarse), responsables de la actividad y número de

iteración. Sin embargo, estos elementos pueden variar conforme sea determinado por el equipo desarrollador. En la figura 3.1 se presenta un ejemplo de formato para una historia de usuario.

HISTORIA DE USUARIO			
Nombre: Desarrollar el prototipo de la interfaz de la aplicación		10-abr-16	**Núm.:** 5
Tipo de historia: ☑ Nueva	☐ Corrección	☐ Mejora	
Prioridad técnica: Media	**Prioridad de usuario:** Media	**Estimación (días):** 15	
Historia previa núm.: 4	**Dependencias:** 2 y 3	**Riesgo:** Bajo	
Descripción: Desarrollar un prototipo navegable de la interfaz de la aplicación.			
Notas: El prototipado de la aplicación se desarrollará a través de los bocetos diseñados en la historia anterior.			
Pruebas de funcionalidad: Las pruebas de funcionalidad se llevarán a cabo mediante los casos de uso identificados.			

Figura 3.1: Ejemplo del formato propuesto para una Historia de Usuario.

Las técnicas de obtención de requisitos son variadas y no se limita al uso exclusivo de unas cuantas. Dependiendo del tipo de información que se desea obtener es como se decide con qué técnica trabajar. Algunas técnicas de obtención de requisitos son: entrevistas, cuestionarios y reuniones informales. Siendo siempre el equipo de desarrollo el que decida con cuales trabajar. Tras finalizar la aplicación de las técnicas seleccionadas se debe redactar un documento por cada historia de usuario identificada, además de una bitácora de proyecto para llevar el registro de la aplicación de cada técnica.

3.4.2. Bitácoras

Las bitácoras de proyecto son documentos utilizados para llevar un registro de todos los puntos tratados, acuerdos a los que se llegaron y cambios en los requisitos

identificados durante las reuniones con el cliente. Las bitácoras pueden incluir los siguientes elementos: número de bitácora, fecha, tipo de técnica aplicada, puntos tratados, acuerdos a los que se llegaron y firma de los participantes. En la figura 3.2, se puede observar un ejemplo de bitácora con la estructura propuesta.

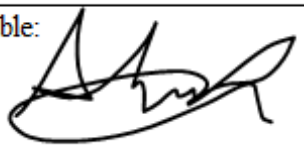
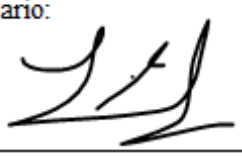
BITÁCORA			
Núm. de bitácora: 1	**Fecha:** 11-may-16	**Tipo de técnica:**	Entrevista
Puntos a tratar: 1.- Necesidad de la aplicación. 2.- Requisitos de la aplicación. 3.- Definir los métodos de diseño de mezclas a utilizar. 4.- Tipos de usuarios. 5.- Documentación necesaria.			
Acuerdos: 1.- La aplicación debe calcular en automático los valores, mediante los valores de 2.- La aplicación debe calcular rápidamente y de forma fiable los resultados. 3.- Se debe mostrar el cálculo de los métodos paso a paso, para que el usuario pueda modificar un valor sin importar la parte del método donde se encuentre. 4.- Los Métodos de diseño de mezclas que se utilizarán son: ACI 211, Norma ASTM, Bolomey, Walker y Füller. 5.- Se incluirá una tabla comparativa de los 5 métodos utilizados. De donde el experto decidirá cuál es el mejor.			
Firmas			
Responsable: 		Usuario: 	

Figura 3.2: Ejemplo del formato propuesto para una bitácora.

Una buena práctica es elaborar también bitácoras de desarrollo. El objetivo de dichas bitácoras es llevar un registro de todos los cambios y avances realizados al proyecto. Las bitácoras de desarrollo pueden incluir los siguientes campos: número de bitácora, fecha de realización, cambios realizados, responsables, firma de los responsables y observaciones. Una vez más será el equipo de desarrollo quien decida que elementos incluir en las bitácoras con las que trabajaran.

3.4.3. Roles

A continuación, se recomiendan algunas roles que deberán ser asignados entre los miembros del equipo. Pueden agregarse o eliminarse roles y funciones según lo considere necesario el equipo de desarrollo.

- Analista. Como su nombre lo indica es quien analiza el problema del cliente y establece una solución que asegure que se cubran todas las necesidades que se presenten. El analista se reúne con el cliente y especifica los requisitos funcionales del proyecto.

- Arquitecto de software. Se encarga de traducir las soluciones hechas por el analista en algoritmos, diagramas y bocetos. El objetivo de esto es permitir al desarrollador codificar los módulos del proyecto de una manera más rápida y sencilla.

- Desarrollador. Es el encargado de escribir el código que permite dar solución a las ideas plasmadas por el arquitecto de software.

- Diseñador. Es quien diseña la interfaz de usuario de la aplicación. El diseño se lleva a cabo mediante prototipos y bocetos. El desarrollador es el encargado de implementar estos bocetos, por esta razón, es importante que tanto el diseñador como el desarrollador trabajen juntos. De igual forma se debe trabajar en conjunto con el usuario, pues es este quien da el visto bueno a la interfaz de la aplicación.

- Encargado de pruebas (*tester*). Es el encargado de realizar las pruebas al sistema para verificar que se cumplan los requisitos del sistema, además verifica que no haya errores en el sistema. Es altamente recomendable que el encargado de las pruebas sea una persona diferente al o los desarrolladores, debido a que puede ver el sistema desde una diferente perspectiva, lo que le permitirá pensar en escenarios en los que probablemente el desarrollador no pensó.

- Usuario. Es quién utilizará el sistema, por lo que, determina los requisitos funcionales del sistema. Además, de que realiza en conjunto al tester las pruebas y da la aprobación de que una historia de usuario ha sido completada con éxito.

Es importante realizar una correcta designación de roles, pues de esto depende en gran medida el éxito del proyecto. Cada rol deberá de ser asignado a la persona que cumpla de mejor forma con las cualidades necesarias para el rol, teniendo en cuenta su conocimiento, experiencia y potencial. Debido al tamaño del equipo es probable que haya personas a las que se les deba asignar más de un rol o inclusive algunos miembros compartan un mismo rol debido a su complejidad. Los roles y funciones deben ser asignados por todos los miembros del equipo, por lo que todos deben estar de acuerdo y consientes de las responsabilidades que implica cada rol.

3.4.4. Proceso

Se conoce como proceso, al ciclo de desarrollo de la metodología propuesta. A continuación, se describe cada paso de dicha metodología.

1. Selección de técnica. El ciclo de desarrollo comienza con la selección de una o varias técnicas de obtención de datos: entrevistas, cuestionarios, etc. Es el analista quien determina qué técnica debe aplicarse en base al tipo de información que se desea obtener. También es el analista quien ejecuta la técnica de obtención de información.

2. Reuniones con el cliente. En las reuniones con el usuario es donde se ejecutan las técnicas de obtención de información.

3. Bitácoras. El analista elabora las bitácoras para llevar un registro de reuniones con cliente.

4. Historias de usuario. Tras las reuniones con el cliente el analista determina las historias de usuario. Sirven para documentar las modificaciones o identificar nuevos requisitos de la aplicación.

5. Obtención de requisitos. De las historias de usuario se identifican los nuevos requisitos solicitados por el cliente.

6. Priorización de requisitos. El analista junto con el cliente, determinan cuál de los requisitos pendientes es más importante y priorizan los requisitos según su nivel de necesidad.

7. Estimación de esfuerzos. El arquitecto de software y el desarrollador estiman el tiempo y esfuerzo necesarios para cubrir cada uno de los requisitos pendientes.

8. Selección de requisitos. En base a la estimación de esfuerzos y a la prioridad de los requisitos, el analista determina que requisitos pueden ser cumplidos y presentados antes de la próxima reunión con el cliente. Una vez determinados, el arquitecto y el diseñador de software plasman una solución a dichos requisitos.

9. Codificación. El desarrollador codifica la solución plasmada por el arquitecto de software. La codificación de los requisitos se realiza en módulos independientes a la aplicación principal. De esta forma no se afecta la funcionalidad de la aplicación mientras se desarrollan.

10. Incremento. El *tester* aplica las pruebas necesarias a los módulos por integrar. Una vez completadas las pruebas y corregidos los errores, el desarrollador se encarga de integrar los nuevos módulos, mediante un incremento a la aplicación principal. Entonces se dice que se ha completado una iteración del ciclo de desarrollo.

El ciclo de desarrollo debe de repetirse tantas veces sean necesarias hasta que se cumplan con todos los requisitos funcionales y no funcionales del proyecto y hasta que el cliente esté satisfecho con la aplicación. En la figura 3.3 se puede observar cómo es que funciona el desarrollo de la metodología propuesta.

3.5. Caso particular

Para desarrollo de la aplicación correspondiente al proyecto de esta tesis, se aplicó la metodología de desarrollo presentada a lo largo de este capítulo, sin embargo, para poder aplicarla fue necesario adaptar dicha metodología a las características únicas del proyecto. A continuación, se describen las principales características de proyecto y su adaptación a la metodología utilizada.

- El equipo de desarrollo será únicamente de una persona, quien se encargará de llevar a cabo la todas de las funciones de los roles propuestos, tales como

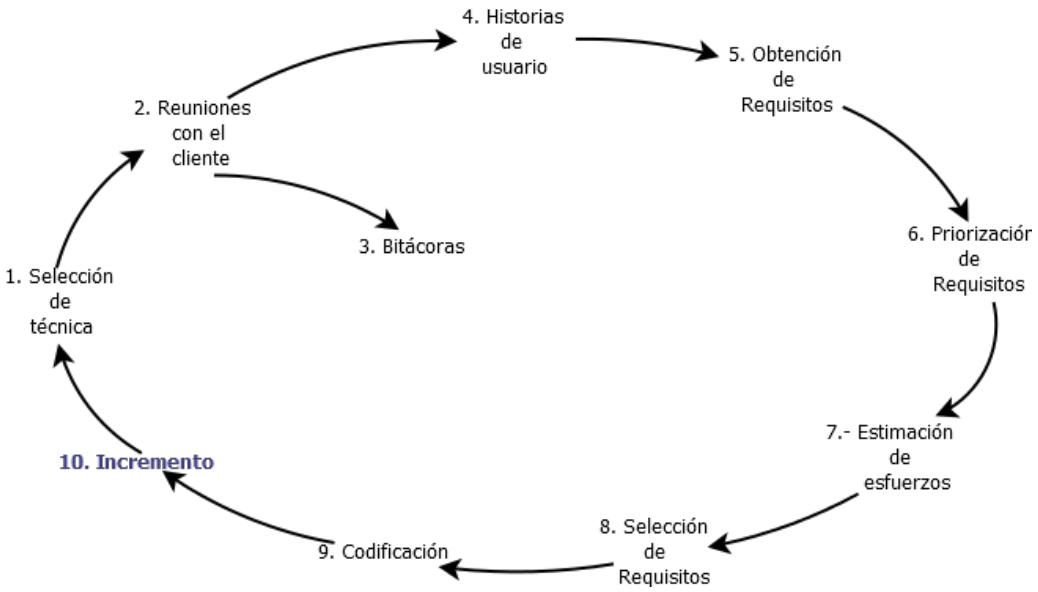


Figura 3.3: Proceso de la metodología propuesta.

analista, arquitecto de software, diseñador, desarrollador y *tester*. Sin embargo es posible que algunas funciones sean omitidas según se considere necesario.

- El cliente estará en constante interacción con el usuario y será este quien determine y apruebe los requisitos del sistema. Además trabajará en conjunto con el desarrollador para aplicar las pruebas necesarias a la aplicación.

- Se tiene planeado realizar siete iteracciones del ciclo de desarrollo de la metodología propuesta. La primera de estas para el desarrollo de la interfaz, las cuatro siguientes para el desarrollo de cada método, la penúltima para el desarrollo del módulo comparativo y la última para el desarrollo de herramientas y detalles de funcionamiento. Cada una de estas iteracciones tendrá una duracción con una estimación inicial de un mes aproximadamente.

- Se llevará a cabo un registro de bitácoras de reunión en las que se plasmarán los acuerdos a los que se llegaron, los cambios a los requisitos y la aprobación por parte del cliente de los módulos presentados. Las bitácoras serán firmadas por el desarrollador y por el cliente.

Una de las principales ventajas de tener un único miembro en el equipo de desa-

rrollo es que se ahorra una gran cantidad de documentación, presente en otras metodologías, necesaria para que el resto del equipo esté informado de los principales cambios y el avance del proyecto. Por otra parte, al tratarse se una sola personas existe una alta probabilidad de que se presente una sobre carga de trabajo, lo que podría generar que se presenten retrasos en los plazos establecidos.

3.6. Conclusiones

Esta metodología presenta un marco de referencia que sirve de guía para que pequeños equipos de desarrollo puedan trabajar de manera eficaz ante proyectos de software, esto mediante pequeñas iteraciones. Para que esta metodología obtenga mejores resultados es preferible que el proyecto cumpla con los siguientes requisitos: pequeño equipo de desarrollo, plazo de tiempo reducido, constante interacción con el usuario, requisitos volátiles y que el tamaño del proyecto vaya de pequeño a mediano.

En esta metodología se propone la implementación de roles, funciones, actividades, bitácoras e historias de usuario que ayudan al equipo a gestionar el proyecto y encaminarlo hacia el éxito. Esta metodología, al igual que las metodologías de desarrollo ágil hace énfasis en la codificación y deja de lado la documentación exhaustiva, común en las metodologías tradicionales. Además, de que permite que el equipo pueda adaptarse rápidamente a los cambios en los requisitos generados por el cliente.

Capítulo 4

Desarrollo de la metodología

4.1. Introducción

En este capítulo se muestra como se aplicó la metodología de desarrollo de software propuesta en el capítulo anterior. Se compone de un total de siete iteraciones o incrementos, los cuales se mencionan a continuación:

1. Desarrollo del prototipo de la interfaz de la aplicación.
2. Desarrollo del método ACI.
3. Desarrollo del método Walker.
4. Desarrollo del método Füller.
5. Desarrollo del método Bolomey.
6. Desarrollo del módulo comparativo.
7. Desarrollo de complementos.

En el primer incremento se desarrolló un prototipo de interfaz en base a los requerimientos solicitados por el cliente. Este prototipo fue desarrollado en el software *GUI Design Studio* con el cual se obtuvo una interfaz navegable, la cual fue presentada al cliente, misma que fue aprobada por el cliente para su desarrollo.

Los incrementos del segundo al quinto estuvieron enfocados al desarrollo de los métodos de diseño de mezclas ACI, Walker, Füller y Bolomey respectivamente. En los cuales se codificó paso a paso el procedimiento de cada método, con el fin de que el usuario pudiera observar en pantalla todo el procedimiento por cada método.

En el sexto incremento se desarrolló un módulo comparativo, el cual retoma los resultados obtenidos por cada uno de los métodos y los presenta en una sola pantalla mediante tablas y gráficas. Esto con el objetivo de permitir al usuario evaluar cuál de los métodos calculados

Por último, el séptimo incremento es en el cual se desarrollaron los complementos de la aplicación, tales como: gestión de proyectos (abrir, crear y guardar proyectos) y la inclusión de herramientas útiles en el diseño de mezclas concreto (análisis granulométrico de los agregados, cálculo de la desviación estándar, cálculo de interpolación lineal, etc.).

4.2. Análisis de requerimientos

El análisis de requerimientos de software es una disciplina propia de la ingeniería de software. El objetivo de esta disciplina es poder obtener los requerimientos que definirán un software y que luego será diseñado, programado, probado y posteriormente utilizado por los usuarios finales (Noél *et al.*, 2016). La obtención de requerimientos se realizó mediante entrevistas con el cliente y con usuarios potenciales.

A través de las entrevistas realizadas se pudo definir el problema a resolver, así como los principales requisitos funcionales y no funcionales de la aplicación. En la tabla 4.1, se pueden observar los principales requisitos obtenidos mediante las primeras entrevistas realizadas. En el Anexo A puede observarse la entrevista realizada al cliente.

En base al análisis realizado y en función de los requerimientos obtenidos, se desarrolló un diagrama de bloques del funcionamiento de la aplicación. En dicho diagrama puede observarse el procedimiento general que se debe llevar a cabo para realizar las funciones principales dentro de la aplicación. En la figura 4.1 puede observarse el diagrama de funcionamiento de la aplicación.

Tabla 4.1: Principales requerimientos funcionales y no funcionales de la aplicación.

Número	Descripción	Tipo
1	La aplicación deberá calcular el resultado de las operaciones matemáticas de forma automática y ofrecer valores de sugerencia durante el desarrollo de los métodos de diseño.	Funcional
2	Debe mostrar el desarrollo de los métodos paso a paso, así como las operaciones realizadas.	Funcional
3	Se incluirán en la aplicación los métodos de diseño ACI, Walker, Füller y Bolomey.	Funcional
4	Los resultados individuales de cada método deben mostrarse tanto en peso seco como corregido en factor de la humedad.	Funcional
5	Los resultados se representaran de dos formas: una para pruebas de laboratorio y otra para valores de obra. Los valores de laboratorio se representaran mediante medidas como kilos, litros y gramos. Mientras que los valores para obra incluirán medidas como bultos, cubetas y botes.	Funcional
6	Incluir un módulo comparativo, en el cual se mostrarán tablas y gráficas de los resultados obtenidos en todos los métodos.	Funcional
7	Debe exportar los resultados de cada método y el módulo comparativo a formato PDF.	Funcional
8	La aplicación deberá funcionar bajo el sistema operativo Windows.	Funcional
9	Debe permitir abrir, crear y guardar proyectos.	Funcional
10	Incluir tablas y gráficas para facilitar la lectura de los resultados finales.	No funcional
11	Incluir herramientas matemáticas para el apoyo de cálculos comunes en el diseño de mezclas.	No funcional
12	Incluir la posibilidad de agregar notas en cada una de las pantallas de la aplicación.	No funcional
13	No será necesario incluir diferentes tipos de usuarios, ya que se desea que todos los usuarios tengan acceso a las mismas funciones.	No funcional
14	Se incluirá un manual de usuario de la aplicación.	No funcional

4.3. Iteración 1: Desarrollo del prototipo de interfaz

El objetivo de esta iteración es el construir el prototipo de interfaz bajo el cual, una vez aprobado, se trabajó en las demás iteraciones. El diseño de la interfaz de

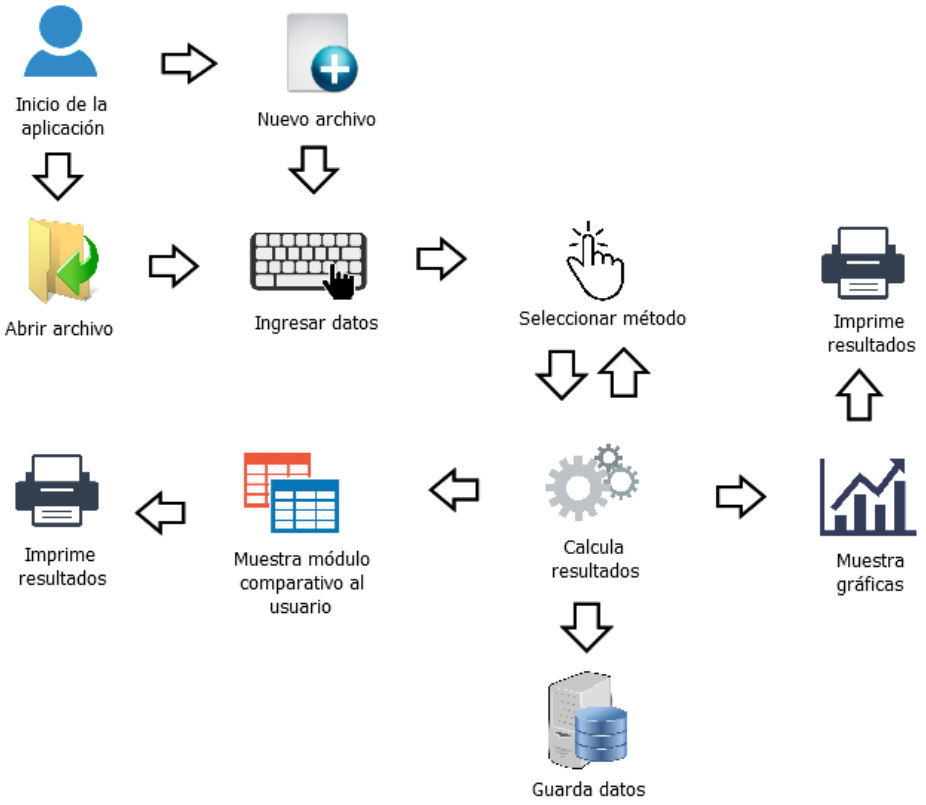


Figura 4.1: Diagrama de bloques del funcionamiento de la aplicación. Una vez iniciada la aplicación el usuario tiene dos opciones; abrir un proyecto existente o iniciar desde uno nuevo. Después el usuario deberá ingresar los datos generales y seleccionar uno de los cuatro métodos de diseño de mezclas disponibles, una vez completado el método el usuario podrá visualizar e imprimir gráficas y tablas con los resultados obtenidos. Los datos se guardan automáticamente cada vez que el usuario realiza un cambio.

aplicación quedó bajo el criterio del desarrollador, por lo cual se presentó una propuesta de interfaz al cliente y fue este quien decidió qué cambios realizar o en caso contrario dió el visto bueno para su uso en la aplicación.

Selección de la técnica

Para el definir los requerimientos de interfaz del proyecto "Desarrollo de una aplicación didáctica para el diseño y evaluación de mezclas de concreto"se realizó una entrevista al cliente, la cual puede verse en el anexo A. En base a los requerimientos

obtenidos en dicha entrevista, se realizó un prototipo de interfaz. Sin embargo, el diseño del prototipo quedo a criterio del desarrollador, siendo el cliente quien dé su aprobación.

Reunión con el cliente

La reunión con el cliente para esta iteración se realizó el día 28 de febrero de 2017. En dicha reunión se aplicó la entrevista para el análisis de requerimientos mencionada en la sección anterior. A partir de su aplicación se generaron bitácoras e historias de usuario las cuales se muestran en las secciones siguientes.

Bitácoras

Tras la primera reunión que se sostuvo con el cliente, se obtuvo una primera bitácora correspondiente a dicha reunión. En ella se pueden observar los puntos tratados, así como los acuerdos a los que se llegaron en la reunión. En la figura 4.2 puede observarse la bitácora obtenida.

Historias de usuario

Después de aplicar la entrevista con el usuario, además de la bitácora correspondiente, también se generó una historia de usuario. Dicha historia de usuario se compone de una actividad principal denominada "Desarrollar el prototipo de la interfaz de la aplicación", la cual a su vez se divide en subtareas y actividades que componen la actividad principal. Asimismo, incluye notas que hay de tomar en cuenta para el desarrollo de las actividades e información acerca de como se desarrollarían las pruebas de funcionalidad pertinentes. La historia de usuario correspondiente a la primera iteración puede observarse en la figura 4.3.

Obtención de requisitos

Una vez obtenida la historia de usuario principal, se procede a identificar los requisitos necesarios para que dicha historia de usuario pueda ser completada correctamente. Deben ser contemplados todos los requisitos que se consideren importantes, los cuales pueden incluir tanto requisitos funcionales como no funcionales. No

BITÁCORA

Número de bitácora: 1	**Fecha:** 28-feb-2017	**Tipo de técnica:** Entrevista

Puntos a tratar:

1. Necesidad de la aplicación.
2. Requisitos de la aplicación.
3. Definir los métodos de diseño de mezclas de concreto a utilizar.
4. Definir los tipos de usuarios.
5. Documentación necesaria.

Acuerdos:

1. La aplicación debe calcular en automático los valores, mediante los valores de entrada obtenidos
2. La aplicación debe calcular rápidamente y de forma fiable los resultados.
3. Se debe mostrar el cálculo de los métodos paso a paso, para que el usuario pueda modificar un valor sin importar la parte del método donde se encuentre.
4. Los valores se deben ser calculados de forma automática, pero el usuario decidirá con qué valores trabajar.
5. Los Métodos de diseño de mezclas que se definidos por el momento son: ACI 211, Bolomey, Walker y Füller.
6. Se mostrarán dos tablas para mostrar los resultados individuales de cada método. Cada una con dos nomenclaturas distintas, pero con los mismos valores. Una para laboratorio y otra para obra.
7. Se incluirá una tabla comparativa de los 5 métodos utilizados. De donde el experto decidirá cuál es el mejor.
8. Los resultados deben poder ser exportados a formato PDF (formato para imprimir).
9. No será necesario definir privilegios de usuarios, pues se desea que todos los usuarios tengan acceso a las mismas funciones.
10. Se desea que la aplicación trabaje bajo la plataforma de Windows.
11. La aplicación se basará en el diseño de mezclas para concreto normal. Se dejará a un lado el uso para concreto con aditivos, pues existen demasiados e incluirlos todos dificultaría el desarrollo de la aplicación.

Firmas

Responsable:	Cliente:
Alfredo Xochitemol Cruz	Ing. Miguel Ángel Daza Merino

Figura 4.2: Bitácora correspondiente a la primera reunión.

necesariamente todos los requisitos serán incluidos en el próximo incremento, por lo que antes deben priorizarse los requisitos y seleccionar solo aquellos que se consideren más importantes y que puedan ser aplicados antes de la siguiente iteración. Los requisitos obtenidos de la historia de usuario correspondiente a la primera iteración pueden observarse en la tabla 4.2.

HISTORIA DE USUARIO			
Nombre: Desarrollar el prototipo de la interfaz de la aplicación		28-feb-17	**Núm.:** 1
Tipo de historia: [X] Nueva	[ ] Corrección	[ ] Mejora	
Prioridad técnica: Alta	**Prioridad de usuario:** Alta	**Estimación (días):** 15	
Historia previa núm.:	**Dependencias:**	**Riesgo:** Medio	
Descripción: • Desarrollar un prototipo navegable de la interfaz de la aplicación. • Incluir un apartado por cada método que muestre su desarrollo paso a paso. • Debe incluir un módulo que permita comparar los resultados obtenidos por los cuatro métodos de mezclas de concretos.			
Notas: • Se usarán los mismos datos de entrada para todos los métodos. • Se incluirá un apartado para agregar notas en cada pantalla.			
Pruebas de funcionalidad: Se realizará un test de usabilidad, en el cual se determinará qué aspectos son necesarios corregir dentro de la aplicación.			

Figura 4.3: Historia de usuario correspondiente a la primera reunión.

Priorización de requisitos

El siguiente paso después haber identificado los requisitos es la priorización de estos. El método de priorización utilizado es el método MoSCoW que clasifica cada requisto con un valor: *Must Have* (debe tener), *Should Have* (debería tener), *Could Have* (podría tener) y *Won't Have* (no tendrá). Cada valor es asignado a cada requisito según sea considerado por el desarrollador. La priorización de requisitos realizada mediante el método MoSCow puede observarse en la tabla 4.3.

Estimación de esfuerzos

La estimación de esfuerzos es una parte muy importante dentro de la metodología utilizada, debido a que si no se realiza correctamente podrían no alcanzarse los objetivos planeados. La estimación de esfuerzos se realizó en base a experiencias previas y toma en cuenta factores como el tamaño, la complejidad, la cantidad de líneas de código, el aprendizaje de métodos y el software utilizado. La estimación de esfuerzos realizada puede observarse en la tabla 4.3.

Tabla 4.2: Requisitos correspondientes a la primera historia de usuario.

Número	Requisito
1	Deben incluirse las opciones para crear, abrir y guardar proyectos.
2	La pantalla principal deberá mostrar el progreso de cada método.
3	Incluir la opción para agregar notas.
4	Incluir acceso rápido a cada método.
5	Mostrar un ejemplo de la exportación a PDF.
6	Incluir barras de desplazamiento si la información no alcanza a ser mostrada en pantalla.
7	El módulo de comparativo debe mostrar los resultados obtenidos en cada método mediante cifra y mediante su representación gráfica.
8	Debe incluir una pantalla para modificar los datos generales del proyecto.
9	Mostrar los pasos del método ACI.
10	Mostrar los pasos del método Walker.
11	Mostrar los pasos del método Füller.
12	Mostrar los pasos del método Bolomey.
13	Incluir una barra de menú con acceso rápido a las funciones principales de la aplicación.

Selección de requisitos

En base a la priorización de requisitos y a la estimación de esfuerzos, se seleccionan aquellos requisitos que se deberán cumplir y que además puedan lograrse antes de la próxima reunión con el cliente. Se estimó que la próxima reunión sería dentro de 15 días a partir de la última reunión. En esta ocasión se seleccionaron todos los requisitos de prioridad *Must have* y *Should have*, dejando de lado los requisitos de prioridad *Could have* y *Won't Have* para próximas iteraciones. Debido a que el desarrollo de los métodos es muy similar de un método a otro, sólo se creará el prototipo de procedimiento para el método ACI. El total de las horas estimadas antes de la selección de requisitos es de 139, sin embargo, al realizarse la selección de requisitos el total de horas estimadas se redujo a 57, cantidad que permite trabajar y completar los requisitos seleccionados cómodamente antes de la próxima reunión, considerando que se trabajó un total de 6 horas diarias. La selección de requisitos

realizada puede observarse en la tabla 4.3.

Tabla 4.3: Priorización, estimación de esfuerzos y selección de requisitos correspondientes a la primera iteración. Los requisitos en color verde son aquellos que sí se incluyen en esta iteración, mientras que los de no tienen color son aquellos que quedaron descartados para este incremento.

Número	Requisito	MoSCoW	Estimado
1	Deben incluirse las opciones para crear, abrir y guardar proyectos.	Must Have	4
2	La pantalla principal deberá mostrar el progreso de cada método.	Should Have	6
3	Incluir la opción para agregar notas.	Must Have	4
4	Incluir acceso rápido a cada método.	Must Have	3
5	Mostrar un ejemplo de la exportación a PDF.	Could Have	6
6	Incluir barras de desplazamiento si la información no alcanza a ser mostrada en pantalla.	Must Have	4
7	El módulo de comparativo debe mostrar los resultados obtenidos en cada método mediante cifra y mediante su representación gráfica.	Should Have	6
8	Debe incluir una pantalla para modificar los datos generales del proyecto.	Could Have	4
9	Mostrar los pasos del método ACI.	Must Have	24
10	Mostrar los pasos del método Walker.	Could Have	24
11	Mostrar los pasos del método Füller.	Won't Have	24
12	Mostrar los pasos del método Bolomey.	Won't Have	24
13	Incluir una barra de menú con acceso rápido a las funciones principales de la aplicación.	Must Have	6
	Horas totales		139
	Total de horas descartadas		82
	Total de horas estimadas		57

Codificación/Desarrollo

La etapa de desarrollo de la interfaz se desarrollo en tres fases: en la primera se llevaron a cabo los primeros bocetos a mano de la aplicación, en la segunda fase se desarrolló en un prototipo navegable de la aplicación y por último, se realizó la

codificación de la aplicación en el lenguaje de programación Java. Algunos de los principales bocetos a lápiz de la aplicación pueden observarse en las figuras 4.4, 4.5 y 4.6.

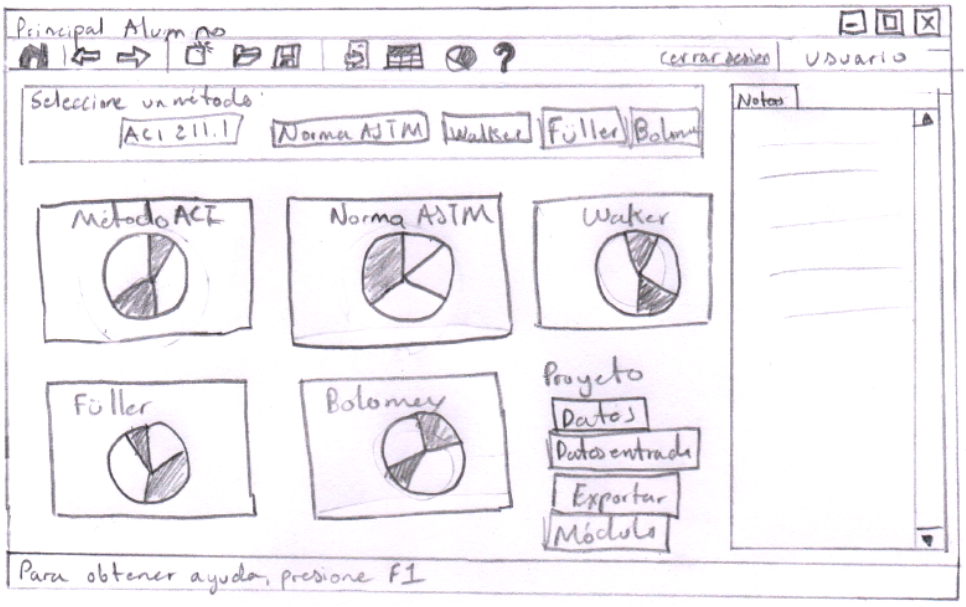


Figura 4.4: Boceto para la pantalla inicial de la aplicación.

Una vez desarrollados y aprobados por el cliente los primeros bocetos de la aplicación se precedió a rediseñar los bocetos, pero ahora con el una de una herramienta de cómputo para el desarrollo de prototipos. La herramienta utilizada fue *Balsamiq Mockups*.

Balsamiq Mockups maneja representaciones de todos los elementos utilizados para la construcción de una aplicación o de una página web, como pantallas de navegadores, títulos, menús, imágenes, videos, etc. Es una herramienta que puede ser usada tanto por clientes como por desarrolladores, con el objetivo de acordar aspectos importantes del alcance de una solución con una mínima inversión de tiempo dedicada. Los clientes pueden hacer uso de *Balsamiq Mockups* sin tener ningún tipo de conocimiento técnico especial. Gracias a ello, pueden comunicar de una manera más eficiente sus ideas y necesidades al grupo de trabajo que realizará las implementaciones. En la figura 4.7 puede observarse un ejemplo del prototipado realizado mediante la herramienta *Balsamiq Mockups*.

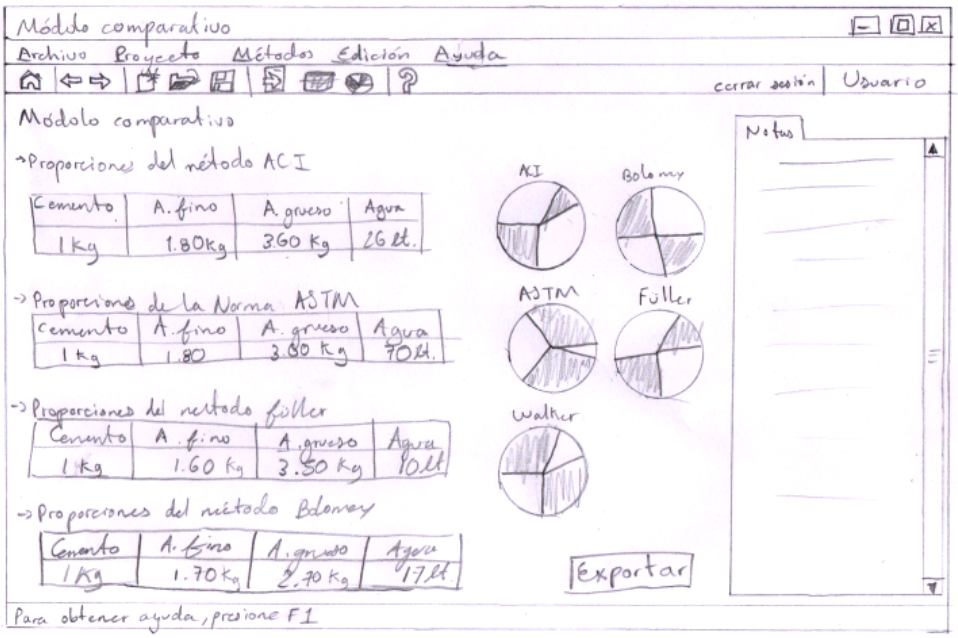


Figura 4.5: Boceto del módulo comparativo.

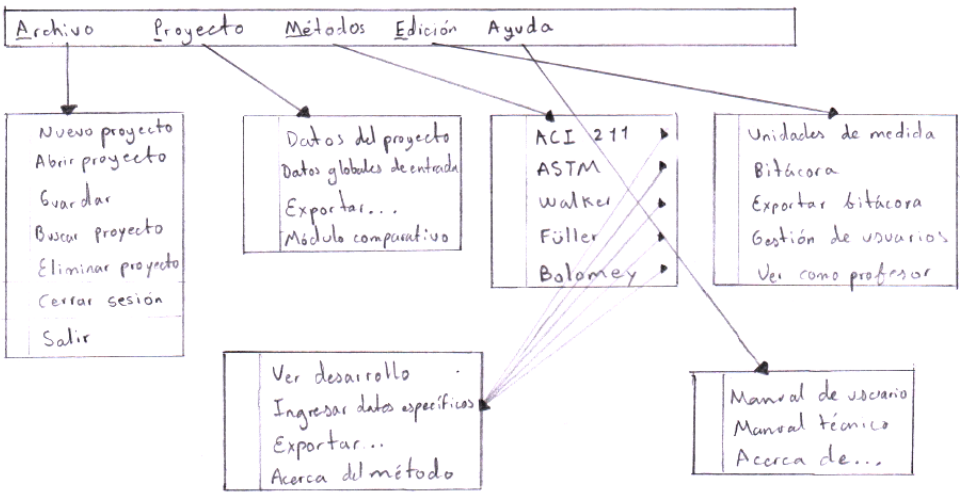


Figura 4.6: Boceto del menú de la aplicación.

El prototipo navegable se desarrolló con el software *GUI Desing Studio* en su versión de prueba. Mediante el uso de esta herramienta se logró obtener un prototipo que permitió al cliente navegar a través de las diferentes pantallas, esto gracias a que

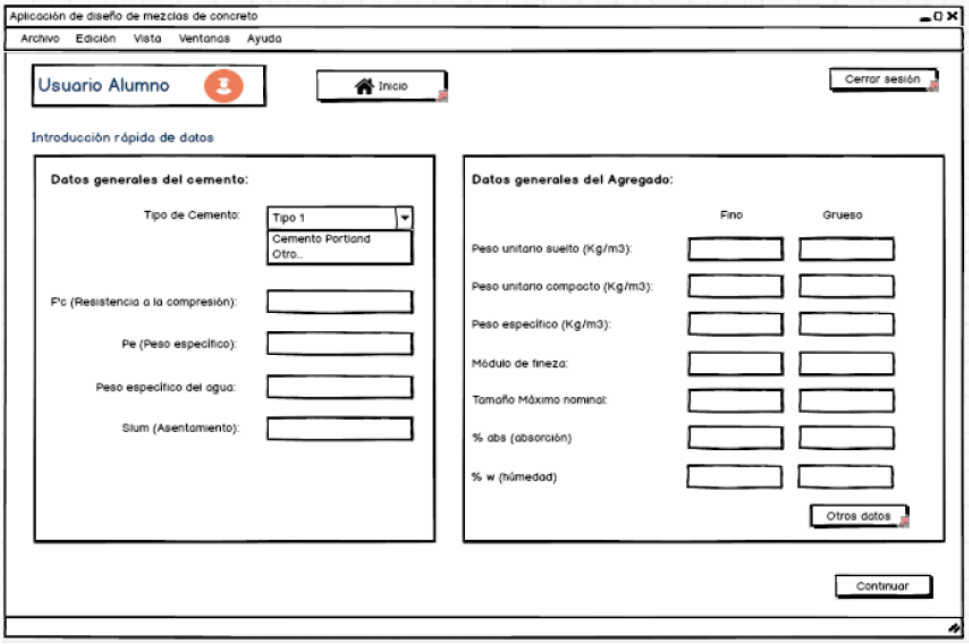


Figura 4.7: Ejemplo de boceto realizado en Balsamiq Mockups.

la herramienta permite conectar varias pantallas entre sí para crear un *storyboard* y ejecutarlos desde el simulador de la aplicación. En la tabla 4.4 se listan todas las pantallas incluidas en el prototipado presentado al cliente. El prototipo final cuenta con un diseño basado en las ventanas del sistema operativo Windows, lo que le da un aspecto más profesional y acerca más al cliente a lo que sería la versión final. En las figuras 4.8 y 4.9 pueden observarse algunas pantallas pertenecientes al prototipo que fue presentado al cliente.

4.4. Iteración 2: Desarrollo del método ACI

La segunda iteración está enfocada al desarrollo del método ACI. En esta iteración se codificaron todas las pantallas pertenecientes a dicho método, además de la pantalla principal de la aplicación, la cual incluye la entrada de datos en común para su uso en todos lo métodos. La interfaz de usuario que utiliza este método y los siguientes se basa en el prototipo presentado y aprobado por el cliente durante la reunión perteneciente a la segunda iteración.

Tabla 4.4: Pantallas incluidas en el prototipo.

Número	Pantalla
1	Pantalla inicial de la aplicación.
2	Abrir proyecto.
3	Crear proyecto.
4	Buscar proyecto.
5	Entrada de datos globales.
6	Desarrollo del método ACI.
7	Resultados del método ACI.
8	Módulo comparativo.
9	Exportar resultados a PDF.
10	Gestión de medidas.
11	Modificar datos generales del proyecto.
12	Menú de la aplicación.

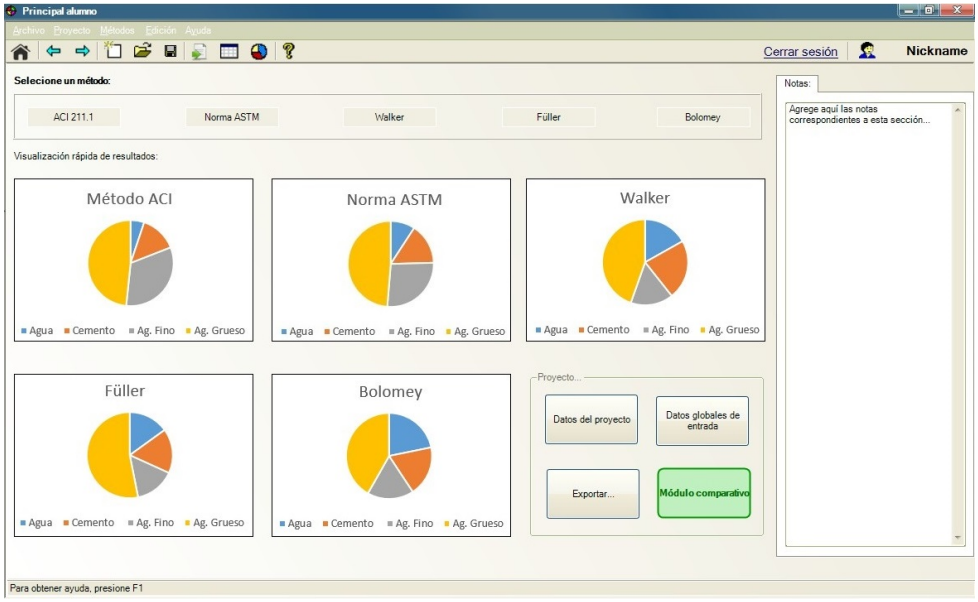


Figura 4.8: Prototipo navegable de la pantalla inicial de la aplicación.

Selección de la técnica

La técnica seleccionada, para la próxima reunión con el cliente, es la de una reunión informal. Debido a que el principal objetivo de la reunión es el de presentar

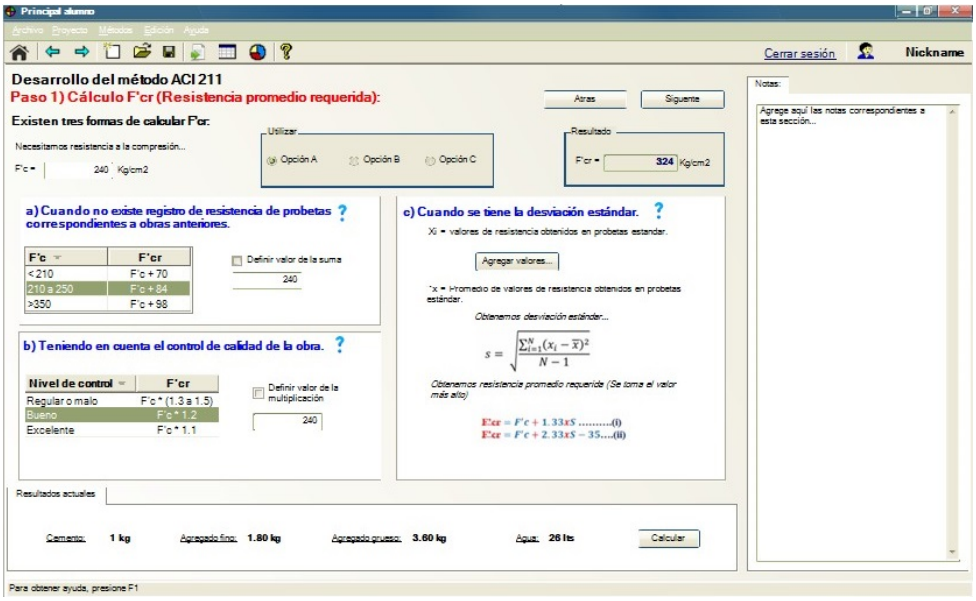


Figura 4.9: Prototipo navegable de la pantalla inicial de la aplicación.

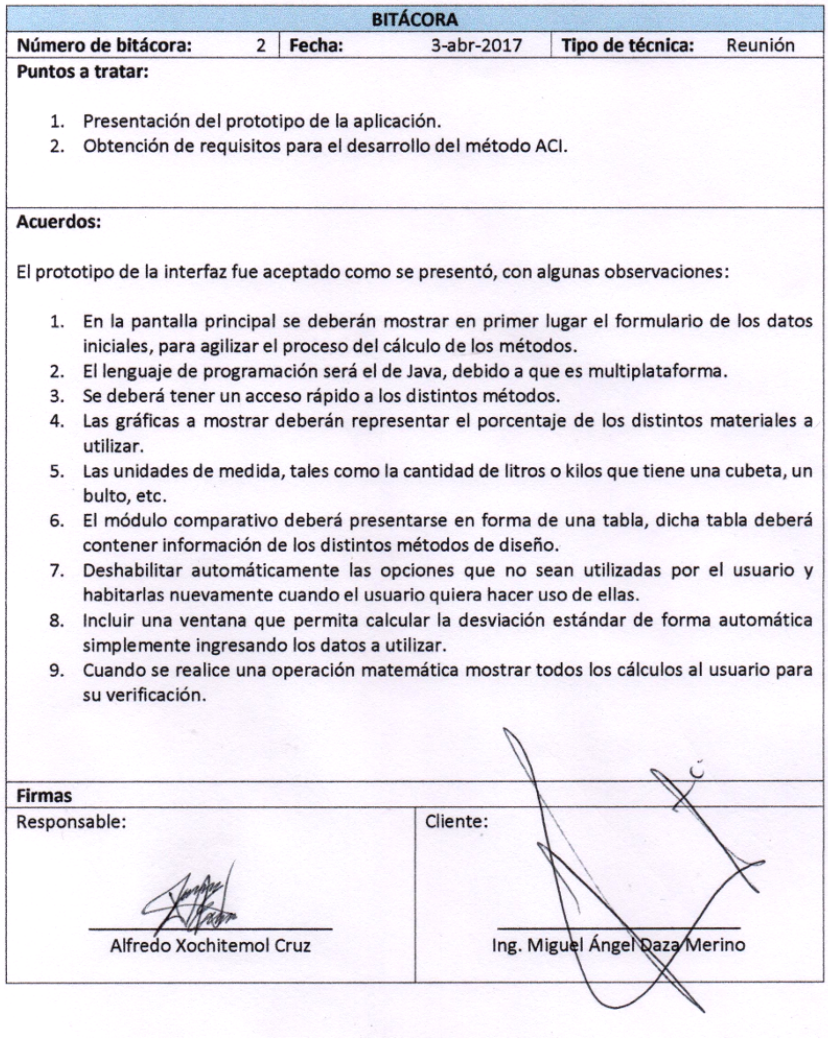
el prototipo de la interfaz y no el de la obtención de información, una reunión informal se considera adecuada.

Reunión con el cliente

Durante está reunión con el cliente se presentó el prototipo de interfaz desarrollado en la iteración anterior, mismo que fue aprobado por el cliente para su aplicación. Además, se obtuvieron los principales requisitos para el desarrollo del primer método de la aplicación, el método ACI.

Bitácoras

Una vez efectuada la reunión con el cliente, correspondiente a la segunda iteración, se generó una bitácora y una historia de usuario principal. La bitácora puede observarse en la figura 4.10, donde se plasman los principales puntos tratados y los acuerdos a los que se llegaron durante dicha reunión.



BITÁCORA

Número de bitácora: 2	**Fecha:** 3-abr-2017	**Tipo de técnica:** Reunión

Puntos a tratar:

1. Presentación del prototipo de la aplicación.
2. Obtención de requisitos para el desarrollo del método ACI.

Acuerdos:

El prototipo de la interfaz fue aceptado como se presentó, con algunas observaciones:

1. En la pantalla principal se deberán mostrar en primer lugar el formulario de los datos iniciales, para agilizar el proceso del cálculo de los métodos.
2. El lenguaje de programación será el de Java, debido a que es multiplataforma.
3. Se deberá tener un acceso rápido a los distintos métodos.
4. Las gráficas a mostrar deberán representar el porcentaje de los distintos materiales a utilizar.
5. Las unidades de medida, tales como la cantidad de litros o kilos que tiene una cubeta, un bulto, etc.
6. El módulo comparativo deberá presentarse en forma de una tabla, dicha tabla deberá contener información de los distintos métodos de diseño.
7. Deshabilitar automáticamente las opciones que no sean utilizadas por el usuario y habitarlas nuevamente cuando el usuario quiera hacer uso de ellas.
8. Incluir una ventana que permita calcular la desviación estándar de forma automática simplemente ingresando los datos a utilizar.
9. Cuando se realice una operación matemática mostrar todos los cálculos al usuario para su verificación.

Firmas

Responsable:	Cliente:
Alfredo Xochitemol Cruz	Ing. Miguel Ángel Daza Merino

Figura 4.10: Bitácora correspondiente a la segunda reunión.

Historia de usuario

El objetivo principal de la historia de usuario generada en esta iteración es el del desarrollo del método ACI. En dicha historia de usuario se describe ampliamente los requerimientos solicitados por el cliente. Además, presenta información útil acerca del desarrollo del método y su implementación. La historia de usuario generada puede observarse en la figura 4.11.

HISTORIA DE USUARIO			
Nombre: Desarrollo del método ACI		03-abr-17	**Núm.:** 2
Tipo de historia:	[X] Nueva	[ ] Corrección	[ ] Mejora
Prioridad técnica: Alta	**Prioridad de usuario:** Alta		**Estimación (días):** 30
Historia previa núm.: 1	**Dependencias:**		**Riesgo:** Medio
Descripción: Desarrollar el procedimiento completo del método ACI. Se deberán mostrar todos los pasos del método y el usuario debe ser capaz de observar todo el procedimiento y las operaciones realizadas, además de poder modificar las variables que considere necesarias en cualquier parte del método. Antes de codificar este método será necesario desarrollar la pantalla principal que incluirá los datos generales necesarios para todos lo métodos.			
Notas: • Antes del codificar el método será necesario crear el apartado de datos generales. • Este método incluirá la corrección en factor de húmedad. • Incluir gráficas y tablas para facilitar la lectura de los resultados.			
Pruebas de funcionalidad: Se realizará un test de usabilidad en el cual se determinará qué aspectos son necesarios corregir dentro de la aplicación.			

Figura 4.11: Historia de usuario correspondiente a la segunda reunión.

Obtención de requisitos

Una vez de generada la historia de usuario principal, se identificaron los requisitos necesarios para su cumplimento. Estos requisitos se extrajeron a través de la información generada tras la reunión efectuada con el cliente. En la tabla 4.5 se pueden observar tanto los requisitos funcionales y no funcionales correspondientes a la segunda historia de usuario.

Priorización de requisitos

La priorización de requisitos, como ya se ha mencionado, se realizó mendiante el método MoSCoW, el cual clasifica a cada valor con uno de cuatro valores posibles: *Must Have* (debe tener), *Should Have* (debería tener), *Could Have* (podría tener) y *Won't Have* (no tendrá). La prioridad asignada a los requisitos mediante este método puede observarse en la tabla 4.6.

Tabla 4.5: Requisitos correspondientes a la segunda historia de usuario.

Número	Requisito
1	Incluir una pantalla para el ingreso de datos generales.
2	Mostrar el desarrollo completo del método.
3	Incluir una sección para comentarios en cada pantalla.
4	Seleccionar automáticamente los valores recomendados.
5	Si el paso lo necesita, incluir distintas tablas de referencia.
6	Se debe poder navegar libremente entre los pasos del método.
7	Mostrar en pantalla en avance del método.
8	En cada paso el usuario deberá poder cambiar el valor calculado por uno propio.
9	Mostrar los resultados finales en forma de gráficas y tablas.
10	Exportar los resultados finales a formato PDF.
11	Incluir una bitácora de los cambios realizados al proyecto.
12	Exportar la bitácora a formato PDF.

Estimación de esfuerzos

La estimación de esfuerzos para cada requisito fue asignada en base a la experiencia previa, considera factores tales como el tamaño, la complejidad, la cantidad de líneas de código, el aprendizaje del método y el software a utilizar. La estimación de esfuerzos asignada en horas para cada requisito puede observarse en la tabla 4.6.

Selección de requisitos

La selección de requisitos se elaboró en base a la priorización y estimación realizada anteriormente. Se seleccionaron aquellos requisitos que se consideró que pudieran completarse sin contratiempos antes del plazo estimado de 30 días, correspondientes a la historia de usuario de esta iteración, considerando que se trabajó un total de 6 horas diarias, lo que da un total de 180 horas de trabajo estimado. En esta ocasión se incluyeron todos los requisitos de prioridad *Must Have*, *Should Have* y *Could Have*, dejando únicamente fuera los requisitos de prioridad *Won't Have*. El total de las horas estimadas por todos los requisitos fue de 194. Sin embargo, después de realizarse la selección de requisitos se redujo a 164 horas, cantidad que permite completar

cómodamente los requisitos seleccionados antes de la próxima reunión con el cliente. La selección de requisitos puede observarse en la tabla 4.6.

Tabla 4.6: Priorización, estimación de esfuerzos y selección de requisitos correspondientes a la segunda iteración. Los requisitos en color verde son aquellos que sí se incluyen en esta iteración, mientras que los de no tienen color son aquellos que quedaron descartados para este incremento.

Número	Requisito	MoSCoW	Estimado
1	Incluir una pantalla para el ingreso de datos generales.	Must Have	15
2	Mostrar el desarrollo completo del método.	Must Have	40
3	Incluir una sección para comentarios en cada pantalla.	Should Have	15
4	Seleccionar automáticamente los valores recomendados.	Must Have	12
5	Si el paso lo necesita, incluir distintas tablas de referencia.	Should Have	15
6	Se debe poder navegar libremente entre los pasos del método.	Must Have	15
7	Mostrar en pantalla en avance del método.	Could Have	10
8	En cada paso el usuario deberá poder cambiar el valor calculado por uno propio.	Must Have	12
9	Mostrar los resultados finales en forma de gráficas y tablas.	Must Have	10
10	Exportar los resultados finales a formato PDF.	Must Have	20
11	Incluir una bitácora de los cambios realizados al proyecto.	Won't Have	20
12	Exportar la bitácora a formato PDF.	Won't Have	10
	Horas totales		194
	Total de horas descartadas		30
	Total de horas estimadas		164

Codificación

El proyecto, como ya se mencionó anteriormente, fue desarrollado en el lenguaje de programación Java y el entorno de desarrollo utilizado fue *NetBeans IDE* en su versión 8.1. *NetBeans* es un producto libre y sin restricciones de uso, principalmente

enfocado al lenguaje de programación Java, sin embargo, cuenta con una gran cantidad de módulos que permiten extender su funcionalidad. La interfaz codificada se basó en el prototipo de interfaz aprobado por el cliente.

En primer lugar, se desarrolló la pantalla inicial (véase figura 4.12), la cual incluye campos para introducir de forma rápida los datos en común necesarios para el cálculo todos los métodos. Además, cuenta con acceso rápido al cálculo de todos los métodos y a las funciones principales de la aplicación. En segundo lugar, se codificaron los pasos correspondientes al método ACI, para el desarrolló de este método se identificaron un total de 13 pasos, los cuales fueron divididos en siete pantallas. En la figura 4.13 puede observarse un ejemplo de las pantallas desarrolladas para el método ACI. Por último, se implementó la exportación a formato PDF, dicho documento incluye los datos iniciales del proyecto, los principales valores obtenidos durante el método y los resultados finales expresados mediante tablas y gráficas.

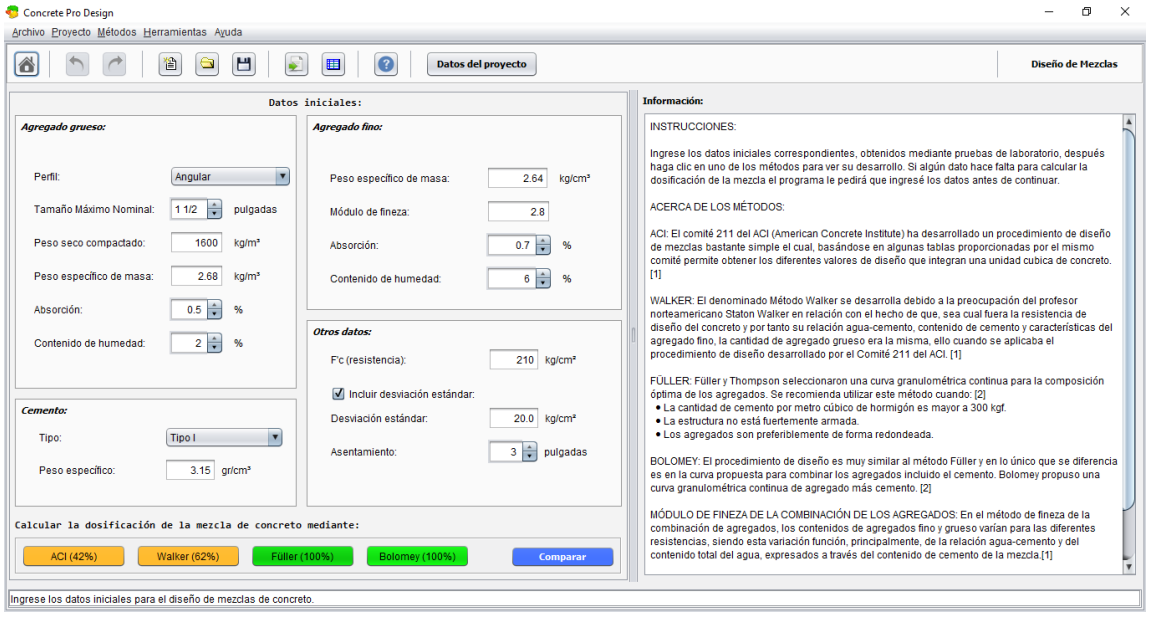


Figura 4.12: Pantalla inicial de la aplicación.

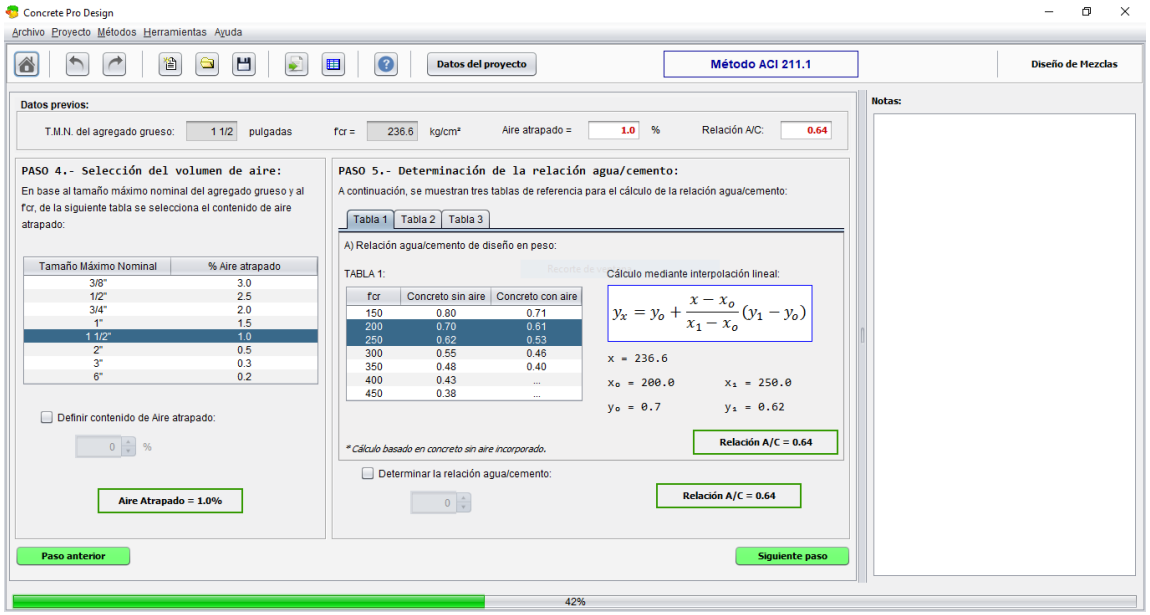


Figura 4.13: Ejemplo de desarrollo del método ACI.

4.5. Iteración 3: Desarrollo del método Walker

La iteración número tres está enfocada al desarrollo del método Walker. Para esta iteración se estimó que el tiempo de desarrollo sería de 25 días, ya que su procedimiento es muy parecido al del Método ACI, por lo que se pueden reutilizar elementos ya desarrollados para el método ACI e incluirlos dentro el método Walker, sin embargo, habrá componentes que será necesario desarrollar completamente desde cero. Durante esta iteración se códificaron las pantallas pertenecientes a dicho método, en este caso fue necesario realizar quince pasos, lo cuales fueron divididos en ocho pantallas distintas.

Selección de técnica

Los objetivos de la próxima reunión son el de presentar al cliente el método ACI que fue desarrollado en la iteración anterior y el de obtener los requisitos necesarios para el desarrollo del método Walker. Debido a que el método Walker cuenta con un procedimiento muy similar al del método ACI, el principal objetivo de la próxima reunión no es el de obtener información, sino el de presentar el método desarrollado

en la iteración anterior y obtener observaciones. Debido a esto, una reunión informal se considera adecuada para la esta iteración.

Reunión con el cliente

El principal tema de esta reunión fue el del presentar al cliente el método ACI, el cual fue desarrollado en la iteración anterior. El cliente interactuó con la aplicación y aprobó el módulo desarrollado correspondiente al primer método, salvo con algunas pequeñas observaciones. Las observaciones realizadas pueden observarse en bitácora generada durante esta reunión (véase figura 4.14).

BITÁCORA

Número de bitácora: 3	**Fecha:** 2-may-2017	**Tipo de técnica:** Reunión

Puntos a tratar:

1. Presentación del primer módulo de la aplicación correspondiente al método ACI.
2. Recopilar notas y observaciones por parte del cliente acerca del módulo presentado.
3. Obtención de los principales requisitos para el desarrollo del método Walker.

Acuerdos:

El módulo correspondiente al método ACI fue presentado al cliente con algunas observaciones:

1. Se deberá incluir dos gráficas para representar los resultados, una para las proporciones en peso seco y otra para las proporciones corregidas en factor de la humedad.
2. Al realizar la exportación a formato PDF también incluir ambas gráficas en el reporte generado.
3. Mostrar claramente el título del método.
4. Se aprobó el diseño del módulo presentado y no se realizaron observaciones.
5. No dejar avanzar en el método si no se ha calculado un valor válido.
6. Cambiar el nombre a ciertos campos.

En cuanto a los requerimientos para el desarrollo el método Walker son los mismos que para el método ACI, salvo algunos nuevos requerimientos que pudieron identificarse:

1. Realizar automáticamente las operaciones necesarias con los valores previos. En caso de que los valores salgan de las tablas recomendadas activar automáticamente la casilla con un valor propio.
2. Mostrar de diferentes colores los valores en peso seco y corregidos en factor de humedad.

Firmas

Responsable:	Cliente:
Alfredo Xochitemol Cruz	Ing. Miguel Ángel Daza Merino

Figura 4.14: Bitácora correspondiente a la tercera reunión.

Una tarea igual de importante para esta reunión fue la de definir los requisitos necesarios para el desarrollo del siguiente módulo, el cual corresponde al desarrollo del método Walker. Durante la reunión se estableció que el método ACI y Walker son muy similares entre sí, esto debido al que el método Walker se basa en el método ACI, pero con la implementación de una tabla que toma en cuenta características propias de los agregados que el método ACI no.

Bitácoras

Una vez efectuada la reunión con el cliente correspondiente a esta iteración, se generó una bitácora perteneciente a dicha reunión. En la bitácora se plasman las observaciones realizadas por parte del cliente al módulo del método ACI, presentado durante esta reunión. Además, describe los requisitos para el desarrollo del método Walker, así como sus principales diferencias con el método ACI. La bitácora correspondiente a esta iteración puede observarse en la figura 4.14.

Historias de usuario

En esta iteración se identificaron dos historias de usuarios principales. La primera de ellas está enfocada a corregir aquellas observaciones realizadas por el cliente al módulo del método ACI. La segunda historia de usuario está dirigida al desarrollo del siguiente módulo, el cual corresponde al método Walker. En cada historia de usuario se presenta información útil acerca de la historia de usuario y su implementación. Las historias de usuario generadas durante esta iteración pueden observarse en las figuras 4.15 y 4.16.

Obtención de requisitos

Los requisitos generados en esta iteración corresponden a los obtenidos a partir de las historias de usuario número 3 y 4 (figura 4.15 y 4.16). En el caso de la historia de usuario número 3, los requisitos fueron obtenidos de las observaciones hechas por el cliente al módulo ACI, que fue desarrollado en la iteración anterior y presentado durante esta reunión. Por otra parte, los requisitos relacionados a la historia de usuario número 4 corresponden al desarrollo del método Walker. En la tabla 4.7 pueden observarse lo requisitos pertenecientes a las historias de usuario 3 y 4.

HISTORIA DE USUARIO					
Nombre:	Corregir observaciones del método ACI		02-may-17	**Núm.:**	3
Tipo de historia:	☐ Nueva	☒ Corrección	☐ Mejora		
Prioridad técnica:	Alta	**Prioridad de usuario:**	Alta	**Estimación (días):**	5
Historia previa núm.:	2	**Dependencias:**	2	**Riesgo:**	Medio
Descripción: Se realizarán las correciones del módulo correspondientes al método ACI: • Se deberán agregar una gráfica de pastel para representar el valor los resultados también en peso seco y no únicamente corregido en factor de la humedad. • En la importación a PDF exportar ambas gráficas. • Cada gráfica debe poder ser expandible para observarla mejor.					
Notas: Crear un botón para que el usuario pueda elegir que gráfica desea visualizar, por defecto seleccionar la gráfica corregida en factor de la humedad.					
Pruebas de funcionalidad: Se realizará un test de usabilidad en el cual se determinará qué aspectos son necesarios corregir dentro de la aplicación.					

Figura 4.15: Primera historia de usuario correspondiente a la tercera iteración.

HISTORIA DE USUARIO					
Nombre:	Desarrollo del método Walker		02-may-17	**Núm.:**	4
Tipo de historia:	☒ Nueva	☐ Corrección	☐ Mejora		
Prioridad técnica:	Alta	**Prioridad de usuario:**	Alta	**Estimación (días):**	20
Historia previa núm.:	3	**Dependencias:**	2	**Riesgo:**	Medio
Descripción: • Codificar los pasos del desarrollo del método Walker. Se deben mostrar todos los pasos y el procedimiento. • Al igual que el método ACI, debe de mostrar las gráficas de resultados para peso seco y corregido en factor de humedad. • Hará uso de los mismos datos iniciales que para el método ACI. • Aplicar los mismos requerimientos que para el método anterior.					
Notas: Debido a que el método Walker se basa en el método ACI, se retomaran pasos ya codificados para el método ACI y se adaptarán al método Walker. De esta forma se reducirá el tiempo de desarrollo y la estimación inicial para esta historia de usuario.					
Pruebas de funcionalidad: Se realizará un test de usabilidad en el cual se determinará qué aspectos son necesarios corregir dentro de la aplicación.					

Figura 4.16: Segunda historia de usuario correspondiente a la tercera iteración.

Tabla 4.7: Requisitos correspondientes a la tercera y cuarta historia de usuario.

Número	Requisito
Requisitos de la historia de usuario número tres	
1	Incluir una gráfica para representar las proporciones en peso seco.
2	Aumentar los datos incluidos en el documento PDF.
3	Verificar y modificar el nombre de algunos campos.
Requisitos de la historia de usuario número cuatro	
4	Mostrar el desarrollo completo del método.
5	Incluir una sección para comentarios en cada pantalla.
6	Seleccionar automáticamente los valores recomendados.
7	Si el paso lo necesita, incluir distintas tablas de referencia.
8	Se debe poder navegar libremente entre los pasos del método.
9	Mostrar en pantalla en avance del método.
10	En cada paso el usuario deberá poder cambiar el valor calculado por uno propio.
11	Mostrar los resultados finales en forma de gráficas y tablas.
12	Exportar los resultados finales a formato PDF.
13	Incluir una bitácora de los cambios realizados al proyecto.
14	Exportar la bitácora a formato PDF.

Priorización de requisitos

La priorización de requisitos, como ya se ha mencionado, se realizó mendiante el método MoSCoW, el cual clasifica a cada valor con uno de cuatro valores posibles: *Must Have* (debe tener), *Should Have* (debería tener), *Could Have* (podría tener) y *Won't Have* (no tendrá). La prioridad asignada a los requisitos mediante este método puede observarse en la tabla 4.8.

Estimación de esfuerzos

La estimación de esfuerzos para cada requisito fue asignada en base a la experiencia previa, se consideraron factores como el tamaño, la complejidad, la cantidad de lineas de código, el aprendizaje del método Walker, entre otros. Cabe destacar que esta ocasión al tratarse de un método que tiene mucha similitud con el método anterior se reduce considerablemente la estimación en tiempo para el proceso de

desarrollo, sin importar que el procedimiento de este método sea ligeramente más extenso. La estimación de esfuerzos asignada en horas puede observarse en la tabla 4.8.

Selección de requisitos

La selección de los requisitos se elaboró en base a la priorización de requisitos y estimación de esfuerzos realizada anteriormente. Se seleccionaron aquellos requisitos que se consideró que pudieran completarse sin contratiempos antes del plazo estimado de 25 días, correspondientes a la suma de las historias de usuario 3 y 4, con un tiempo estimado de 5 y 20 días respectivamente. Se considera que se trabajó un total de 6 horas diarias, lo que da un total de 150 horas de trabajo estimado. En esta ocasión se incluyeron todos los requisitos de prioridad *Must Have*, *Should Have* y *Could Have*, dejando únicamente fuera los requisitos de prioridad *Won't Have*. El total de las horas estimadas por todos los requisitos fue de 168. Sin embargo, después de realizarse la selección de requisitos se redujo a 138 horas, cantidad que permite completar cómodamente los requisitos seleccionados antes de la próxima reunión con el cliente. La selección de requisitos puede observarse en la tabla 4.8.

Codificación

Como resultado de la fase de codificación, se dio solución a las necesidades planteadas en las historias de usuario 3 y 4. En primer lugar, para la historia número 3 se realizaron las modificaciones y mejoras solicitadas por el cliente al módulo anterior, el cual corresponde al método ACI, siendo la modificación de más importancia la adición de una gráfica para representar la proporción de materiales en peso seco y no sólo corregidos en factor de la humedad dentro del método ACI, dicha modificación también fue implementada dentro del método Walker. En segundo lugar, para la historia de usuario número 4 se desarrollo el método Walker, que incluye un total de 15 pasos divididos en 8 pantallas diferentes. Un ejemplo de una pantalla desarrollada para el método Walker puede observarse en la figura 4.17.

Tabla 4.8: Priorización, estimación de esfuerzos y selección de requisitos correspondientes a la tercera iteración. Los requisitos en color verde son aquellos que sí se incluyen en esta iteración, mientras que los de no tienen color son aquellos que quedaron descartados para este incremento.

Número	Requisito	MoSCoW	Estimado
Requisitos de la historia de usuario número tres			
1	Incluir una gráfica para representar las proporciones en peso seco.	Must Have	15
2	Aumentar los datos incluidos en el documento PDF.	Must Have	5
3	Verificar y modificar el nombre de algunos campos.	Must Have	5
Requisitos de la historia de usuario número cuatro			
4	Mostrar el desarrollo completo del método.	Must Have	30
5	Incluir una sección para comentarios en cada pantalla.	Should Have	10
6	Seleccionar automáticamente los valores recomendados.	Must Have	10
7	Si el paso lo necesita, incluir distintas tablas de referencia.	Should Have	12
8	Se debe poder navegar libremente entre los pasos del método.	Must Have	13
9	Mostrar en pantalla en avance del método.	Could Have	5
10	En cada paso el usuario deberá poder cambiar el valor calculado por uno propio.	Must Have	10
11	Mostrar los resultados finales en forma de gráficas y tablas.	Must Have	8
12	Exportar los resultados finales a formato PDF.	Must Have	15
13	Incluir una bitácora de los cambios realizados al proyecto.	Won't Have	20
14	Exportar la bitácora a formato PDF.	Won't Have	10
	Horas totales		168
	Total de horas descartadas		30
	Total de horas estimadas		138

4.6. Iteración 4: Desarrollo del método Füller

La cuarta iteración está enfocada al desarrollo del método Füller. En esta iteración se realizó una estimación inicial de 30 días, durante los cuales se codificó el desarrollo del método. El método Füller es muy diferente a los métodos ya desa-

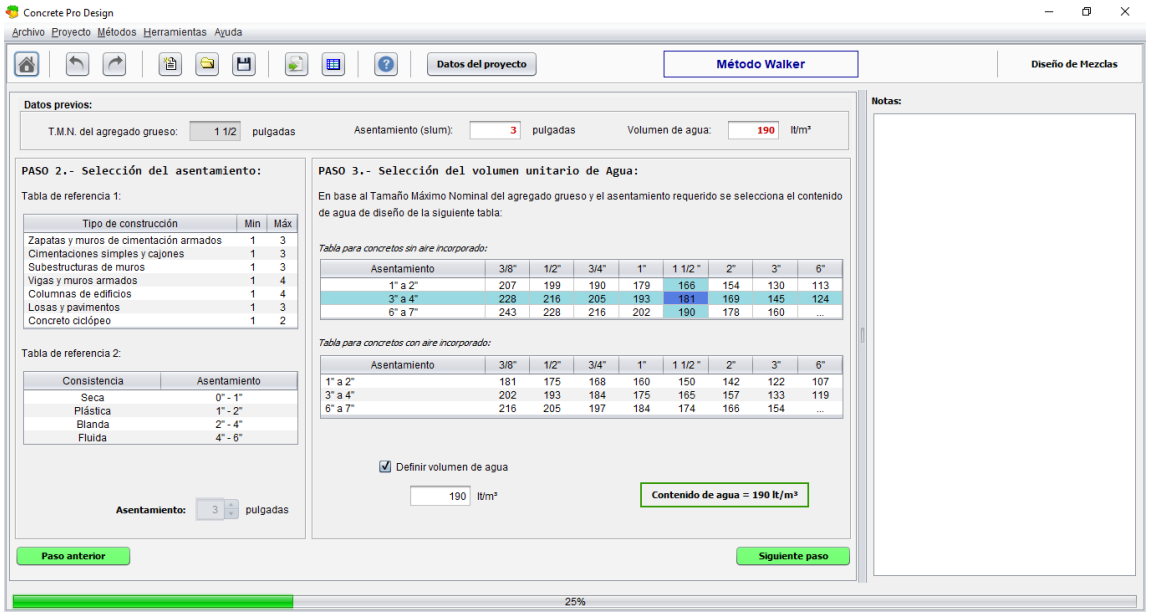


Figura 4.17: Ejemplo de desarrollo del método Walker.

rrollados en las iteraciones anteriores (métodos ACI y método Walker), por lo que muy pocos elementos fueron reutilizados y en su mayoría fueron desarrollados completamente desde cero. El desarrollo completo de este método consta de 11 pasos, los cuales fueron divididos en seis pantallas diferentes. Además, fueron realizadas las correcciones y observaciones hechas por el cliente al método Walker, el cual fue presentado durante la reunión correspondiente a esta iteración.

Selección de técnica

Los objetivos principales para la próxima reunión con el cliente son los de presentar el método Walker, para obtener las correcciones necesarias y el de obtener los requisitos necesarios para el desarrollo del método Füller. Debido a que el procedimiento del método Füller y sus requerimientos serían explicados por el cliente, una entrevista informal se considera adecuada para esta interación, donde el cliente pueda expresar sus opiniones libremente sin ningún esquema que riga la entrevista.

Reunión con el cliente

Durante esta reunión se presentó el módulo desarrollado en la iteración anterior, es decir, el método Walker. El cliente pudo interactuar nuevamente con la aplicación y más específicamente con el método Walker, a lo cual, aprobó dicho método salvo con algunas pequeñas observaciones a corregir durante esta iteración. De igual forma, verificó y aprobó las correcciones hechas al módulo ACI, con lo que el método ACI queda concluido.

En esta reunión también se definieron los requisitos necesarios para codificar el método correspondiente a esta iteración, es decir, el método Füller. Se estableció que el método Füller es muy diferente a los métodos desarrollados anteriormente, por lo que sólo podrán reutilizarse algunos componentes codificados anteriormente y la mayoría tendrán que ser desarrollados desde cero.

Bitácoras

Una vez efectuada la reunión con el cliente, se produjo la bitácora correspondiente a dicha reunión. En la bitácora se plasman los eventos más relevantes que tomaron lugar durante la reunión. En ella pueden observarse los puntos tratados, los acuerdos a los que se llegaron, las observaciones realizadas al método Walker, los principales requisitos para el desarrollo del método Füller, así como algunas notas importantes. La bitácora correspondiente a esta reunión puede observarse en la figura 4.18.

Historias de usuario

En esta iteración se identicaron dos historias de usuario principales. La primera de ellas es la de corregir las observaciones realizadas por el cliente al método Walker. La segunda historia de usuario está enfoca al desarrollo del módulo correspondiente al método Füller. En cada historia de usuario se presenta información útil acerca de la historia de usuario y su implementación. Las dos historias de usuario generadas durante esta iteración pueden observarse en las figuras 4.19 y 4.20.

BITÁCORA

Número de bitácora: 4	**Fecha:** 5-jun-2017	**Tipo de técnica:** Reunión

Puntos a tratar:

1. Verificación de las observaciones realizadas al primer módulo.
2. Presentación del segundo módulo de la aplicación correspondiente al método Walker.
3. Recopilar notas y observaciones por parte del cliente acerca del segundo módulo.
4. Obtención de los principales requisitos para el desarrollo del método Füller.

Acuerdos:

Se presentaron y aprobaron las correcciones solicitadas al método ACI, con lo cual dicho método queda concluido.

El módulo correspondiente al método Walker también fue presentado al cliente con algunas observaciones.

1. Se aprobó el diseño del módulo presentado y no se realizaron observaciones en cuanto a diseño.
2. No permitir avanzar al siguiente paso si no se ha calculado un valor válido.
3. Mostrar un aviso si el usuario está ingresando un valor inválido.

A continuación, se enlistan los principales requerimientos para el método Füller:

1. El método Füller no aplicará la corrección en factor de la humedad.
2. Para el método Füller se tomará en cuenta el perfil del agregado grueso.
3. Si en alguno de los pasos los valores no pueden ser definidos porque los datos se encuentran fuera de las tablas, el usuario podrá definir un valor propio.
4. Contará con una barra de desplazamiento para seleccionar los valores de *K* en el cálculo de la relación A/C.
5. Mostrará la gráfica de Füller antes del análisis granulométrico.
6. La malla de referencia para método Füller será definida por el usuario.
7. Al final del Método solo mostrará la gráfica en peso seco.

Firmas

Responsable:	Cliente:
Alfredo Xochitemol Cruz	Ing. Miguel Ángel Daza Merino

Figura 4.18: Bitácora correspondiente a la cuarta reunión.

Obtención de requisitos

Los requisitos generados en esta iteración corresponden a los obtenidos a partir de las historias de usuario número 5 y 6 (figura 4.19 y 4.20). En el caso de la historia de usuario número 5, los requisitos fueron obtenidos de las observaciones hechas por el cliente al módulo Walker, que fue desarrollado en la iteración anterior y presentado durante esta reunión. Por otra parte, los requisitos relacionados a la historia de usuario número 6 corresponden al desarrollo del método Füller. En la

HISTORIA DE USUARIO					
Nombre:	Corregir observaciones del método Walker.		05-jun-17	**Núm.:**	5
Tipo de historia:	☐ Nueva	☒ Corrección		☐ Mejora	
Prioridad técnica:	Alta	**Prioridad de usuario:**	Alta	**Estimación (días):**	5
Historia previa núm.:	4	**Dependencias:**	4	**Riesgo:**	Medio

Descripción:
Se realizarán las correciones del módulo correspondientes al método Walker:
• No permitir avanzar al siguiente paso si no se ha calculado un valor válido.
• Mostrar un aviso si el usuario está ingresando un valor inválido.
• Al igual que el método ACI, mostrar gráficas tanto para peso seco como corregido en factor de la humedad.

Notas:
En cuanto al diseño del módulo no se persentaron observaciones, por lo que no será necesario realizar mejoras de aspecto visual.

Pruebas de funcionalidad:
Se realizará un test de usabilidad en el cual se determinará qué aspectos son necesarios corregir dentro de la aplicación.

Figura 4.19: Primera historia de usuario correspondiente a la cuarta iteración.

HISTORIA DE USUARIO					
Nombre:	Desarrollo del método Füller		05-jun-17	**Núm.:**	6
Tipo de historia:	☒ Nueva	☐ Corrección		☐ Mejora	
Prioridad técnica:	Alta	**Prioridad de usuario:**	Alta	**Estimación (días):**	25
Historia previa núm.:	5	**Dependencias:**	4	**Riesgo:**	Medio

Descripción:
• Mostrar todos los pasos correspondientes al desarrollo del método.
• Mostrar en pantalla tanto la curva de Füller, como el análisis granulométrico.
• Incluir una barra de desplazamiento para seleccionar los valores permitidos de K en el cálculo de la relación A/C.
• La malla de referencia para el método Füller será definida por el usuario.

Notas:
• El método Füller no aplicará la corrección en factor de la húmedad, sino que únicamente lo hará en peso seco.
• El método Füller se calcula mediante el uso de la gráfica de Füller y del análisis granulométrico.

Pruebas de funcionalidad:
Se realizará un test de usabilidad en el cual se determinará qué aspectos son necesarios corregir dentro de la aplicación.

Figura 4.20: Segunda historia de usuario correspondiente a la cuarta iteración.

tabla 4.9 pueden observarse lo requisitos pertenecientes a las historias de usuario 5 y 6.

Tabla 4.9: Requisitos correspondientes a la quinta y sexta historia de usuario.

Número	Requisito
Requisitos de la historia de usuario número cuarto	
1	No permitir avanzar al siguiente paso si no se ha calculado un valor válido.
2	Mostrar una advertencia si el usuario está ingresando un valor inválido.
3	Mostrar una gráfica para las proporciones en peso seco.
Requisitos de la historia de usuario número seis	
4	Mostrar el desarrollo completo del método.
5	Incluir una barra de desplazamiento para seleccionar los valores permitidos de K en el cálculo de la relación A/C según la parábola de Füller.
6	Mostrar gráficamente la parábola de Füller.
7	Desarrollar una herramienta para realizar el análisis granulométrico.
8	Exportar a PDF la gráfica correspondiente al análisis granulométrico y la parábola de Füller.
9	Calcular automáticamente la malla referencia óptima para el método Füller.
10	Mostrar en pantalla en avance del método.
11	En cada paso el usuario deberá poder cambiar el valor calculado por uno propio.
12	Mostrar los resultados finales en forma de gráficas y tablas.
13	Exportar los resultados finales a formato PDF.
14	Incluir una bitácora de los cambios realizados al proyecto.
15	Exportar la bitácora a formato PDF.

Priorización de requisitos

La priorización de requisitos, como ya se ha mencionado, se realizó mendiante el método MoSCoW, el cual clasifica a cada valor con uno de cuatro valores posibles: *Must Have* (debe tener), *Should Have* (debería tener), *Could Have* (podría tener) y

Won't Have (no tendrá). La prioridad asignada a los requisitos mediante este método puede observarse en la tabla 4.10.

Estimación de esfuerzos

La estimación de esfuerzos para cada requisito fue asignada en base a la experiencia previa, se consideraron factores como el tamaño, la complejidad, la cantidad de lineas de código, el aprendizaje del método Füller, entre otros. Debido a que este método es muy diferente a los métodos desarrollados anteriormente, el tiempo de desarrollo estimado se aumenta considerablemente para algunas tareas. La estimación de esfuerzos asignada en horas puede observarse en la tabla 4.10.

Selección de requisitos

La selección de los requisitos se elaboró en base a la priorización de requisitos y estimación de esfuerzos realizada anteriormente. Se seleccionaron aquellos requisitos que se consideró que pudieran completarse sin contratiempos antes del plazo estimado de 30 días, correspondientes a la suma de las historias de usuario 5 y 6, con un tiempo estimado de 5 y 25 días respectivamente. Se considera que se trabajó un total de 6 horas diarias durante los 30 días, lo que da un total de 180 horas de trabajo estimado.

En esta ocasión se incluyeron todos los requisitos de prioridad *Must Have*, *Should Have* y *Could Have*, dejando únicamente fuera los requisitos de prioridad *Won't Have*. El total de las horas estimadas por todos los requisitos fue de 191, cantidad que no puede ser cubierta con el plazo estimado de 30 días. Sin embargo, después de realizarse la selección de requisitos se descartaron 45 horas, por lo que se redujo la cantidad de horas estimadas a 146 horas, cantidad que permite cubrir cómodamente los requisitos seleccionados antes de la próxima reunión con el cliente. La selección de requisitos puede observarse en la tabla 4.10.

Codificación

Como resultado de la fase de codificación, se dio solución a las necesidades planteadas en las historias de usuario 5 y 6. En primer lugar, para la historia número 5 se realizaron las modificaciones y mejoras solicitadas por el cliente al método Walker.

Tabla 4.10: Priorización, estimación de esfuerzos y selección de requisitos correspondientes a la cuarta iteración. Los requisitos en color verde son aquellos que sí se incluyen en esta iteración, mientras que los de no tienen color son aquellos que quedaron descartados para este incremento.

Número	Requisito	MoSCoW	Estimado
Requisitos de la historia de usuario número cuarto			
1	No permitir avanzar al siguiente paso si no se ha calculado un valor válido.	Must Have	10
2	Mostrar una advertencia si el usuario está ingresando un valor inválido.	Must Have	5
3	Mostrar una gráfica para las proporciones en peso seco.	Must Have	10
Requisitos de la historia de usuario número seis			
4	Mostrar el desarrollo completo del método.	Must Have	40
5	Incluir una barra de desplazamiento para seleccionar los valores permitidos de K en el cálculo de la relación A/C según la parábola de Füller.	Could Have	8
6	Mostrar gráficamente la parábola de Füller.	Should Have	15
7	Desarrollar una herramienta para realizar el análisis granulométrico.	Should Have	20
8	Exportar a PDF la gráfica correspondiente al análisis granulométrico y la parábola de Füller.	Won't Have	8
9	Calcular automáticamente la malla referencia óptima para el método Füller.	Won't Have	7
10	Mostrar en pantalla en avance del método.	Could Have	5
11	En cada paso el usuario deberá poder cambiar el valor calculado por uno propio.	Must Have	10
12	Mostrar los resultados finales en forma de gráficas y tablas.	Must Have	8
13	Exportar los resultados finales a formato PDF.	Must Have	15
14	Incluir una bitácora de los cambios realizados al proyecto.	Won't Have	20
15	Exportar la bitácora a formato PDF.	Won't Have	10
	Horas totales		191
	Total de horas descartadas		45
	Total de horas estimadas		146

Debido a que el método Walker guarda gran similitud con el método ACI y a que

el módulo del método ACI ya ha sido completamente aprobado por el cliente, solo una pequeña cantidad de modificaciones fueron necesarias. En segundo lugar, para la historia de usuario número 6 se desarrollo el método Füller, que incluye un total de 11 pasos divididos en 6 pantallas diferentes. Un ejemplo de una pantalla desarrollada para el método Walker puede observarse en la figura 4.21.

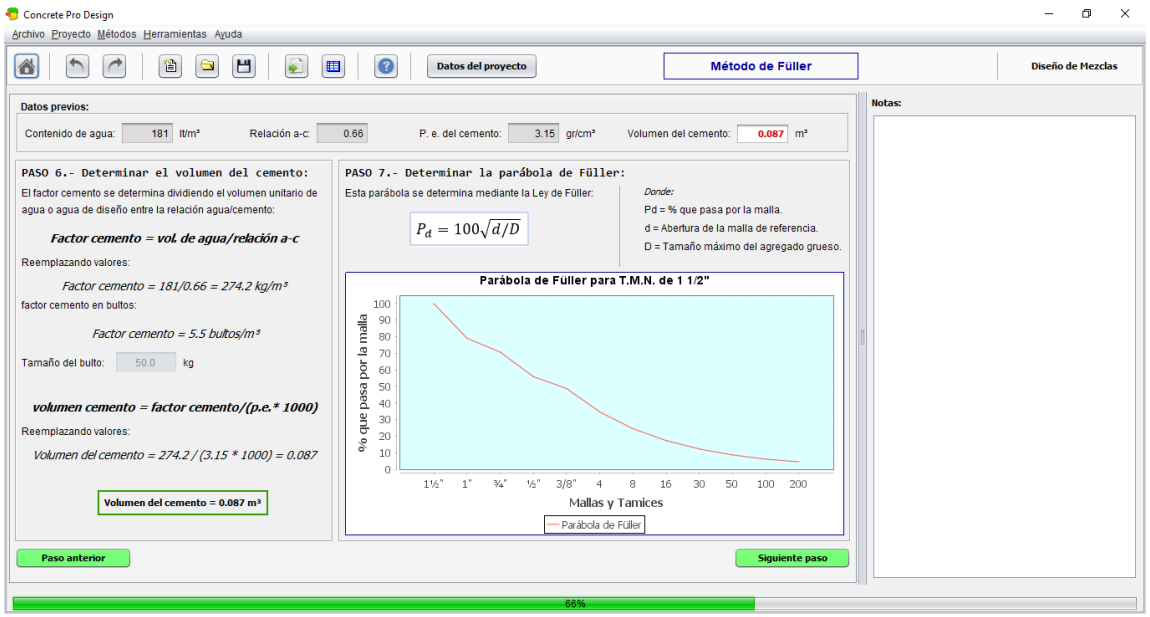


Figura 4.21: Ejemplo de desarrollo del método Füller.

4.7. Iteración 5: Desarrollo del método Bolomey

La quinta iteración está en enfocada al desarrollo del último de los cuatro métodos, es decir, el método Bolomey. El tiempo estimado para esta iteración es de 20 días, esto debido a que su procedimiento es muy parecido al del método Füller, el cual ya ha sido desarrollado, por lo que se pueden reutilizar elementos ya desarrollados e incluirlos en el método Bolomey, reduciendo así el tiempo de desarrollo. Sin embargo, hay elementos que necesitaron ser desarrollados desde cero. Durante esta iteración se códificó el Método Bolomey el cuál consta de 11 pasos los cuales fueron divididos en 6 pantallas diferentes.

Selección de técnica

Las tareas principales para la próxima reunión con el cliente son las de presentar el módulo correspondiente al método Füller y el de obtener los requisitos necesarios para el desarrollo del método Bolomey. Debido a que el procedimiento del método Bolomey es muy similar al del método Füller, el principal objetivo para la próxima reunión no es el de obtener información, sino el de presentar el método Füller y obtener observaciones por parte del cliente. Debido a todo esto, una reunión informal se considera adecuada para esta iteración.

Reunión con el cliente

En esta ocasión, el objetivo principal de la reunión fue el de presentar al cliente el módulo perteneciente al método Füller, el cual fue desarrollado durante la iteración anterior. El cliente interactuó con dicho método y lo aprobó sin correcciones mayores. El segundo de los objetivos fue el de obtener los requisitos necesarios para el desarrollo del método Bolomey. Durante la reunión se estableció que el método Bolomey se basa en el método Füller, por lo que su desarrollo es muy parecido. La principal diferencia entre ambos es que el método Bolomey no utiliza la fórmula para calcular la parábola de Füller (véase 4.1), sino que utiliza una versión modificada que incluye las características propias de cemento del cemento (véase 4.2).

Parábola de Füller:

$$P_d = 100\sqrt{\frac{d}{D}} \qquad (4.1)$$

Donde:
P_d = porcentaje que pasa por la malla.
d = apertura de la malla de referencia (mm).
D = Tamaño Máximo Nominal (TMN) del agregado grueso (mm).

Parábola de Bolomey:

$$Y = A + (100 - A)\sqrt{\frac{d}{D}} \quad (4.2)$$

Donde:
Y = porcentaje que pasa por la malla.
d = apertura de la malla de referencia (mm).
D = Tamaño Máximo Nominal del agregado grueso (mm).
A = Coeficiente que depende de la forma del agregado y la consistencia del cemento.

Bitácoras

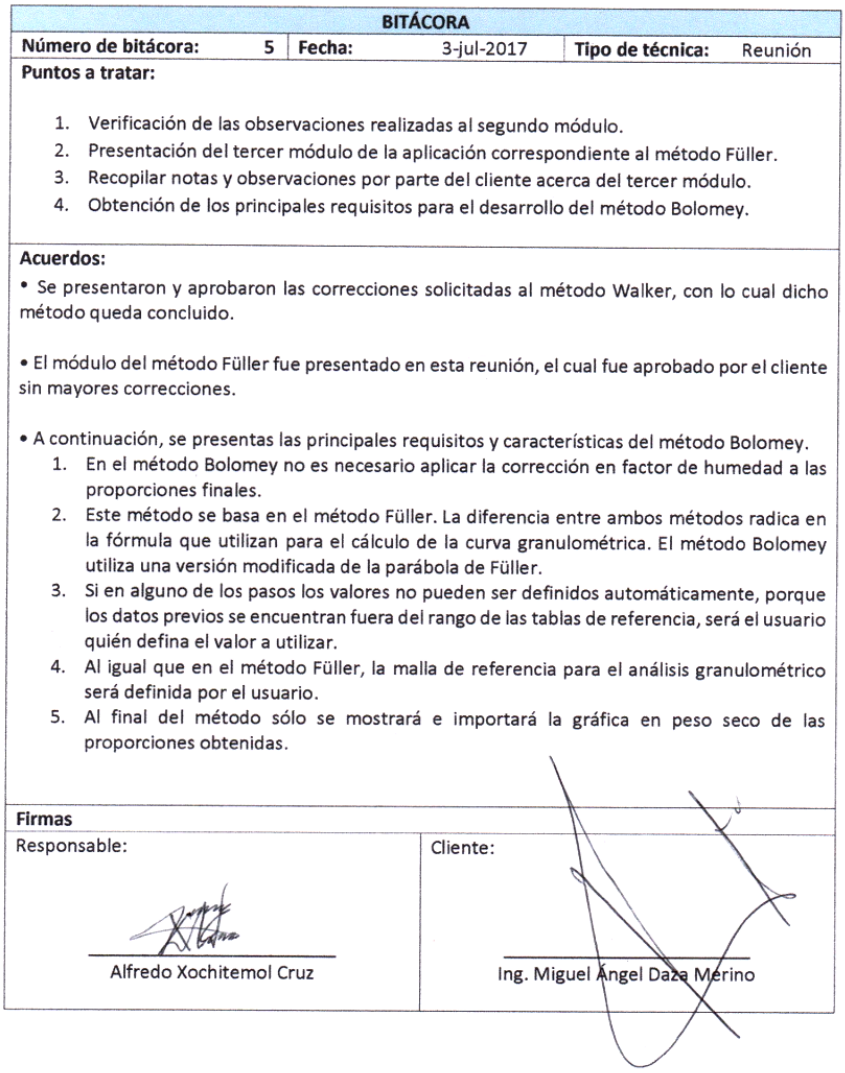
Una vez realizada la reunión con el cliente, se elaboró una bitácora para llevar a cabo el registro de los principales eventos ocurridos durante dicha reunión. En la bitácora se registraron las principales observaciones que realizó el cliente con respecto al método que se presentó, es decir, el método Füller. Además, describe los requisitos, mencionados por el cliente, necesarios para el desarrollo del método Bolomey, así como las principales diferencias entre este método y el Füller. La bitácora correspondiente a esta iteración puede observarse en la figura 4.22.

Historias de usuario

La finalidad de la historia de usuario generada en esta iteración es la del desarrollo del último método, es decir, el método Bolomey. En dicha historia de usuario se describen los requisitos solicitados por el cliente. Además presenta información útil acerca del desarrollo del método y su implementación. La historia de usuario generada puede observarse en la figura 4.23.

Obtención de requisitos

Los requisitos para esta iteración corresponden al conjunto de requisitos generados a partir de la historia de usuario número 7 (véase figura 4.23) y a los correspondientes a las correcciones de módulos anteriores. Los requisitos relacionados a la historia de usuario número 7 corresponden a aquellos que son necesarios para el desarrollo del método Bolomey, mientras que los de corrección corresponden a las



BITÁCORA

Número de bitácora: 5	**Fecha:** 3-jul-2017	**Tipo de técnica:** Reunión

Puntos a tratar:

1. Verificación de las observaciones realizadas al segundo módulo.
2. Presentación del tercer módulo de la aplicación correspondiente al método Füller.
3. Recopilar notas y observaciones por parte del cliente acerca del tercer módulo.
4. Obtención de los principales requisitos para el desarrollo del método Bolomey.

Acuerdos:

• Se presentaron y aprobaron las correcciones solicitadas al método Walker, con lo cual dicho método queda concluido.

• El módulo del método Füller fue presentado en esta reunión, el cual fue aprobado por el cliente sin mayores correcciones.

• A continuación, se presentas las principales requisitos y características del método Bolomey.

1. En el método Bolomey no es necesario aplicar la corrección en factor de humedad a las proporciones finales.
2. Este método se basa en el método Füller. La diferencia entre ambos métodos radica en la fórmula que utilizan para el cálculo de la curva granulométrica. El método Bolomey utiliza una versión modificada de la parábola de Füller.
3. Si en alguno de los pasos los valores no pueden ser definidos automáticamente, porque los datos previos se encuentran fuera del rango de las tablas de referencia, será el usuario quién defina el valor a utilizar.
4. Al igual que en el método Füller, la malla de referencia para el análisis granulométrico será definida por el usuario.
5. Al final del método sólo se mostrará e importará la gráfica en peso seco de las proporciones obtenidas.

Firmas

Responsable:	Cliente:
Alfredo Xochitemol Cruz	Ing. Miguel Ángel Daza Merino

Figura 4.22: Bitácora correspondiente a la quinta reunión.

observaciones hechas por el cliente al módulo anterior y que deben de corregirse durante esta iteración. En la tabla 4.11 pueden observarse los requisitos pertenecientes a esta iteración.

Priorización de requisitos

La priorización de requisitos, como ya se ha mencionado, se realizó mendiante el método MoSCoW, el cual clasifica a cada valor con uno de cuatro valores posibles:

HISTORIA DE USUARIO			
Nombre: Desarrollo del método Bolomey		03-jun-17	**Núm.:** 7
Tipo de historia:	[X] Nueva	[ ] Corrección	[ ] Mejora
Prioridad técnica: Alta	**Prioridad de usuario:** Alta		**Estimación (días):** 20
Historia previa núm.: 6	**Dependencias:** 6		**Riesgo:** Medio
Descripción: • Mostrar todos los pasos correspondientes al desarrollo del método. • Mostrar en pantalla tanto la curva de Bolomey, como el análisis granulométrico. • Incluir una barra de desplazamiento para seleccionar los valores permitidos de K en el cálculo de la relación A/C. • La malla de referencia para el método Bolomey será definida por el usuario.			
Notas: • El método Bolomey no aplicará la corrección en factor de la húmedad, sino que únicamente lo hará en peso seco. • La principal diferencia entre el método Bolomey y el Füller es que el Bolomey utiliza una versión modificada de la fórmula para la parábola de Füller.			
Pruebas de funcionalidad: Se realizará un test de usabilidad en el cual se determinará qué aspectos son necesarios corregir dentro de la aplicación.			

Figura 4.23: Historia de usuario correspondiente a la quinta iteración.

Must Have (debe tener), *Should Have* (debería tener), *Could Have* (podría tener) y *Won't Have* (no tendrá). La prioridad asignada a los requisitos mediante este método puede observarse en la tabla 4.12.

Estimación de esfuerzos

La estimación de esfuerzos para cada requisito fue asignada en base a la experiencia previa, se consideraron factores como el tamaño, la complejidad, la cantidad de lineas de código, el aprendizaje del método de Bolomey, entre otros. Debido a que este método es bastante parecido al método Füller, el cual fue ya desarrollado anteriormente, el tiempo de desarrollo estimado se reduce considerablemente para algunas tareas, ya que distintos elementos pueden ser reutilizados y adaptados al método Bolomey sin gran complejidad. La estimación de esfuerzos asignada en horas puede observarse en la tabla 4.12.

Tabla 4.11: Requisitos correspondientes a la quinta iteración

Número	Requisito
1	Mostrar el desarrollo completo del método.
2	Incluir una barra de desplazamiento para seleccionar los valores permitidos de K en el cálculo de la relación A/C.
3	Mostrar gráficamente la parábola de Bolomey.
4	Incluir una herramienta para realizar el análisis granulométrico.
5	Exportar a PDF la gráfica correspondiente al análisis granulométrico y la parábola de Bolomey.
6	Calcular automáticamente la malla referencia óptima para el método de Bolomey.
7	Mostrar en pantalla en avance del método.
8	En cada paso el usuario deberá poder cambiar el valor calculado por uno propio.
9	Mostrar los resultados finales en forma de gráficas y tablas.
10	Exportar los resultados finales a formato PDF.
11	Incluir una bitácora de los cambios realizados al proyecto.
12	Exportar la bitácora a formato PDF.

Selección de requisitos

La selección de los requisitos se elaboró en base a la priorización y estimación de esfuerzos realizada anteriormente. Se seleccionaron aquellos requisitos que se consideró que pudieran completarse sin contratiempos antes del plazo estimado de 20 días, correspondientes a la historia de usuario número 7 y que conforma esta iteración. Se considera que se trabajó un total de 6 horas diarias durante los 20 días, lo que da un total de 120 horas de trabajo estimado.

En esta ocasión se incluyeron todos los requisitos de prioridad *Must Have*, *Should Have* y *Could Have*, dejando únicamente fuera los requisitos de prioridad *Won't Have*. El total de las horas estimadas por todos los requisitos fue de 149, cantidad que no puede ser cubierta con el plazo estimado de 20 días. Sin embargo, después de realizarse la selección de requisitos se descartaron 45 horas, por lo que se redujo la cantidad de horas estimadas a un total de 104 horas, cantidad que permite cubrir cómodamente los requisitos seleccionados antes de la próxima reunión con el cliente.

La selección de requisitos puede observarse en la tabla 4.12.

Tabla 4.12: Priorización, estimación de esfuerzos y selección de requisitos correspondientes a la quinta iteración. Los requisitos en color verde son aquellos que sí se incluyen en esta iteración, mientras que los de no tienen color son aquellos que quedaron descartados para este incremento.

Número	Requisito	MoSCoW	Estimado
1	Mostrar el desarrollo completo del método.	Must Have	30
2	Incluir una barra de desplazamiento para seleccionar los valores permitidos de K en el cálculo de la relación A/C.	Must Have	9
3	Mostrar gráficamente la parábola de Bolomey.	Could Have	14
4	Incluir una herramienta para realizar el análisis granulométrico.	Should Have	18
5	Exportar a PDF la gráfica correspondiente al análisis granulométrico y la parábola de Bolomey.	Won't Have	8
6	Calcular automáticamente la malla referencia óptima para el método de Bolomey.	Won't Have	7
7	Mostrar en pantalla en avance del método.	Could Have	5
8	En cada paso el usuario deberá poder cambiar el valor calculado por uno propio.	Must Have	9
9	Mostrar los resultados finales en forma de gráficas y tablas.	Must Have	6
10	Exportar los resultados finales a formato PDF.	Must Have	13
11	Incluir una bitácora de los cambios realizados al proyecto.	Won't Have	20
12	Exportar la bitácora a formato PDF.	Won't Have	10
	Horas totales		149
	Total de horas descartadas		45
	Total de horas estimadas		104

Codificación

Como resultado de la fase de codificación, se desarrolló el módulo correspondiente al último de los cuatro métodos, es decir, el método de Bolomey. Para el desarrollo de este método se identificaron y codificaron once pasos, los cuales fueron divididos en seis pantallas diferentes. Este método comparte muchas de las características con el

método Füller, por lo que la codificación se redujo considerablemente, esto debido a muchos elementos, que ya se habían sido desarrollado anteriormente para el método Füller, pudieron ser reutilizados y adaptados con facilidad al método de Bolomey. En la figura 4.24 puede observarse un ejemplo de una de las pantallas desarrolladas para el método de Bolomey.

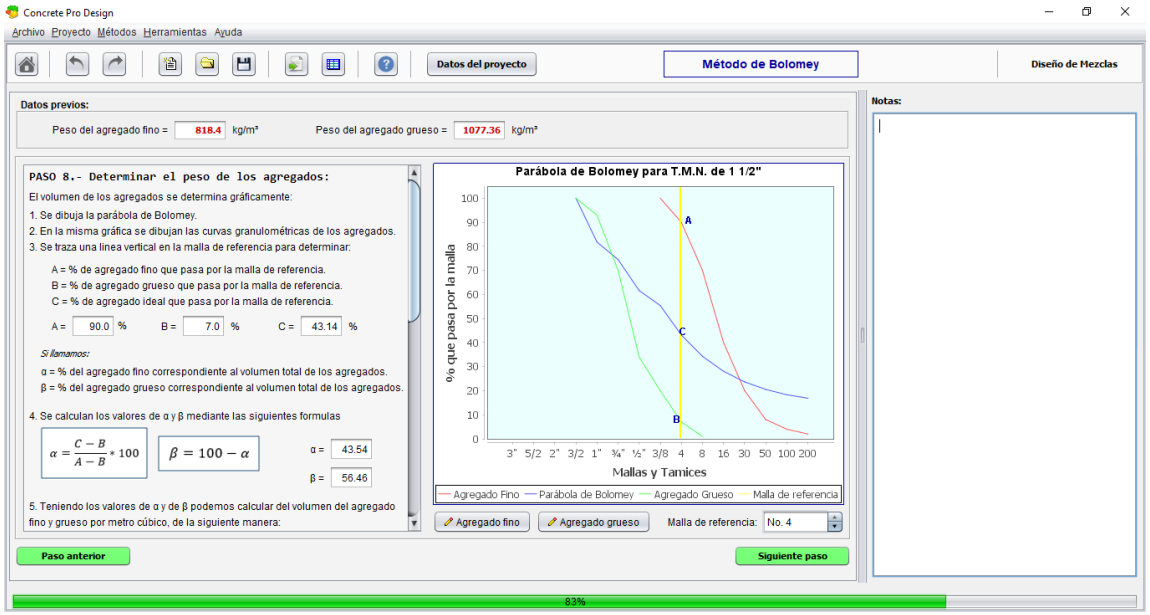


Figura 4.24: Ejemplo de desarrollo del método Bolomey.

4.8. Iteración 6: Desarrollo del módulo comparativo

La sexta iteración esta enfocada al desarrollo del Módulo comparativo. El módulo comparativo es un componente dentro de la misma aplicación que permite comparar en una sola pantalla los principales resultados y proporciones obtenidas en cada uno de los métodos. El tiempo estimado para esta iteración fue de 15 días, tiempo durante el cual se desarrolló una sola pantalla con tablas y gráficas para representar la información, además es posible exportar toda la información presente el módulo comparativo a formato PDF.

Selección de técnica

La técnica utilizada para la próxima reunión con el cliente es la de una reunión informal, ya que en dicha reunión el principal objetivo es que el cliente explique los principales requisitos con los que debe cumplir el módulo comparativo, además, se presentará al usuario el módulo correspondiente al método Bolomey, de lo que se espera obtener una retroalimentación por parte del cliente. Debido a esto, es necesario que el cliente pueda expresar libremente sus opiniones sin un formato que guíe la reunión, por lo tanto una reunión informal se considera adecuada.

Reunión con el cliente

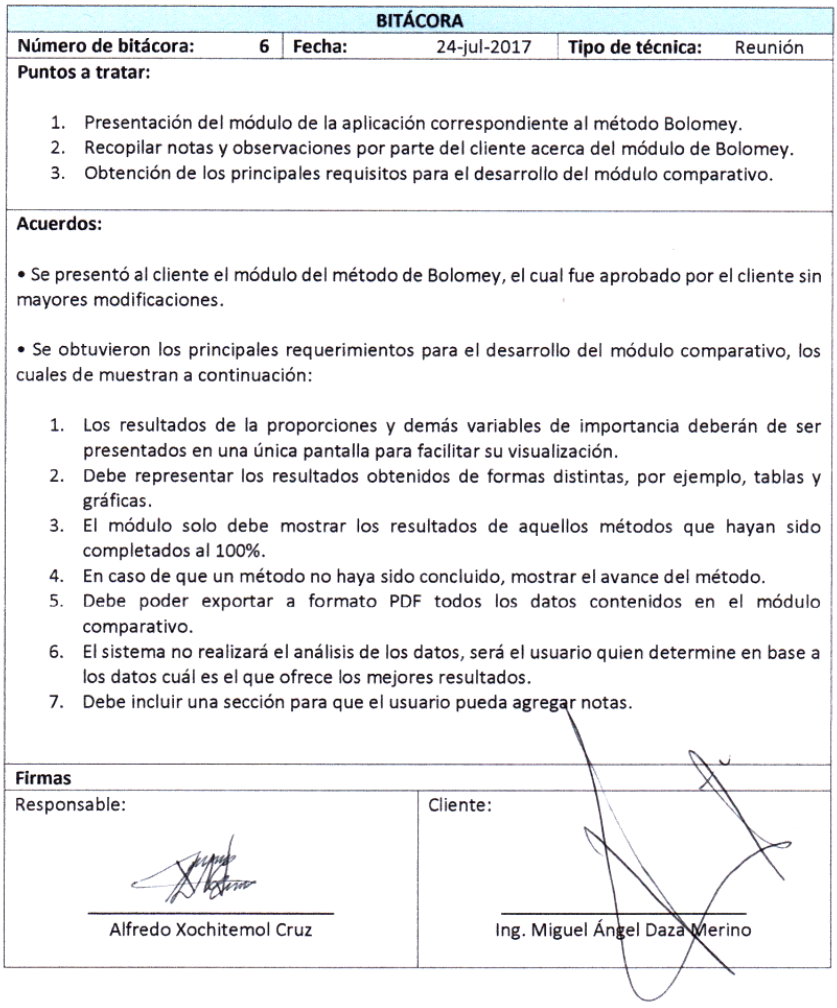
Los principales objetivos de la reunión, como ya se mencionó anteriormente, son los de presentar el módulo de Bolomey al cliente y el de obtener los requisitos para el desarrollo del módulo comparativo. Durante la primera parte de la reunión el cliente interactuó con el módulo de Bolomey y debido a que este módulo comparte gran parte de sus características con el método de Füller, el cual ya había sido presentado, el cliente lo aprobó sin más correcciones. Durante la segunda parte de la reunión el cliente estableció los requisitos principales para el desarrollo del módulo comparativo, el cual fue desarrollado durante esta iteración.

Bitácoras

Una vez efectuada la reunión con el cliente, se laboró una bitácora para llevar a cabo un registro de los principales eventos ocurridos durante la reunión. En la bitácora se registraron las observaciones que realizó el cliente con respecto al módulo de Bolomey así como los principales requisitos, mencionados por el cliente, para el desarrollo del módulo comparativo. Además, en dicha bitácora se describen los puntos que se trataron y los acuerdos a los que se llegaron durante la reunión. La bitácora correspondiente a esta iteración puede observarse en la figura 4.25.

Historias de usuario

En está iteración se generó una única historia de usuario, cuyo objetivo principal es el del desarrollo del módulo comparativo. En dicha historia de usuario se plasman



BITÁCORA

Número de bitácora: 6	**Fecha:** 24-jul-2017	**Tipo de técnica:** Reunión

Puntos a tratar:

1. Presentación del módulo de la aplicación correspondiente al método Bolomey.
2. Recopilar notas y observaciones por parte del cliente acerca del módulo de Bolomey.
3. Obtención de los principales requisitos para el desarrollo del módulo comparativo.

Acuerdos:

• Se presentó al cliente el módulo del método de Bolomey, el cual fue aprobado por el cliente sin mayores modificaciones.

• Se obtuvieron los principales requerimientos para el desarrollo del módulo comparativo, los cuales de muestran a continuación:

1. Los resultados de la proporciones y demás variables de importancia deberán de ser presentados en una única pantalla para facilitar su visualización.
2. Debe representar los resultados obtenidos de formas distintas, por ejemplo, tablas y gráficas.
3. El módulo solo debe mostrar los resultados de aquellos métodos que hayan sido completados al 100%.
4. En caso de que un método no haya sido concluido, mostrar el avance del método.
5. Debe poder exportar a formato PDF todos los datos contenidos en el módulo comparativo.
6. El sistema no realizará el análisis de los datos, será el usuario quien determine en base a los datos cuál es el que ofrece los mejores resultados.
7. Debe incluir una sección para que el usuario pueda agregar notas.

Firmas

Responsable:	Cliente:
Alfredo Xochitemol Cruz	Ing. Miguel Ángel Daza Merino

Figura 4.25: Bitácora correspondiente a la sexta reunión.

los principales detalles de la misma, como su nombre, fecha, número de historia, tipo, prioridad, tiempo estimado, entre otras. Además, muestra las principales tareas para el desarrollo del módulo comparativo e información de importancia acerca del método y su implementación. La historia de usuario número 8, generada durante esta iteración, puede observarse junto a todos sus detalles en la figura 4.26.

HISTORIA DE USUARIO			
Nombre: Desarrollo del método Bolomey		03-jun-17	**Núm.:** 7
Tipo de historia:	[X] Nueva	[ ] Corrección	[ ] Mejora
Prioridad técnica: Alta	**Prioridad de usuario:** Alta		**Estimación (días):** 20
Historia previa núm.: 6	**Dependencias:** 6		**Riesgo:** Medio
Descripción: • Mostrar todos los pasos correspondientes al desarrollo del método. • Mostrar en pantalla tanto la curva de Bolomey, como el análisis granulométrico. • Incluir una barra de desplazamiento para seleccionar los valores permitidos de K en el cálculo de la relación A/C. • La malla de referencia para el método Bolomey será definida por el usuario.			
Notas: • El método Bolomey no aplicará la corrección en factor de la húmedad, sino que únicamente lo hará en peso seco. • La principal diferencia entre el método Bolomey y el Füller es que el Bolomey utiliza una versión modificada de la fórmula para la parábola de Füller.			
Pruebas de funcionalidad: Se realizará un test de usabilidad en el cual se determinará qué aspectos son necesarios corregir dentro de la aplicación.			

Figura 4.26: Historia de usuario número 8, correspondiente a la sexta iteración.

Obtención de requisitos

Los requisitos para el desarrollo de este módulo corresponden a los obtenidos a partir de la historia de usuario número 8, la cual fue generada durante esta iteración (véase figura 4.26) y a las observaciones realizadas por parte del cliente al módulo anterior. Los requisitos de la historia de la historia de usuario número 8 corresponden a aquellos que son necesarios para el desarrollo del módulo comparativo y su correcta implementación, mientras que los de corrección corresponden a las observaciones hechas por el cliente al módulo anterior y que deben corregirse durante esta iteración. En la tabla 4.13 pueden observarse los requisitos pertenecientes a esta iteración.

Priorización de requisitos

La priorización de requisitos, como ya se ha mencionado, se realizó mendiante el método MoSCoW, el cual clasifica a cada valor con uno de cuatro valores posibles: *Must Have* (debe tener), *Should Have* (debería tener), *Could Have* (podría tener) y *Won't Have* (no tendrá). La prioridad asignada a los requisitos mediante este método puede observarse en la tercera columna de la tabla 4.14.

Tabla 4.13: Requisitos correspondientes a la sexta iteración.

Número	Requisito
1	Realizar las correcciones de las observaciones realizadas por el cliente al módulo Bolomey.
2	El módulo comparativo deberá ser mostrado en una sola pantalla, con el fin de facilitar su visualización.
3	El módulo debe representar las proporciones y demás datos de importancia mediante tablas y gráficas.
4	Sólo deben de mostrarse los resultados de aquellos métodos que han sido completados al cien porciento.
5	En caso de que un método no haya sido completado, mostrar únicamente el porcentaje de avance del método.
6	Se deben poder exportar todos los datos del módulo comparativo a formato PDF.
7	Realizar una bitácora que incluya los últimos cambios realizados a los métodos.
8	Incluir el análisis granulométrico de los agregados en el módulo comparativo.
9	Incluir una sección para agregar notas.
10	Se debe poder expandir todas las gráficas.
11	Sugerir automáticamente el método que ofrece los mejores resultados.

Estimación de esfuerzos

La estimación de esfuerzos para cada requisito fue asignada en base a la experiencia previa, se consideraron factores como el tamaño, la complejidad, la cantidad de lineas de código, entre otros. El tamaño del módulo comparativo se considera pequeño en comparación a los módulos desarrollados anteriormente, por lo tanto el tiempo de desarrollo estimado total es bastante reducido. La estimación de esfuerzos asignada a cada requisito puede observarse en la cuarta columna de la tabla 4.14.

Selección de requisitos

La selección de los requisitos se elaboró en base a la priorización y estimación de esfuerzos realizada anteriormente. Se seleccionaron aquellos requisitos que se consi-

deró que pudieran completarse sin contratiempos antes del plazo estimado de 15 días, correspondientes a la historia de usuario número 8 y que conforma esta iteración. Se considera que se trabajó un total de 6 horas diarias durante los 15 días, lo que da un total de 90 horas de trabajo estimado.

En esta ocasión se incluyeron todos los requisitos de prioridad *Must Have* y *Should Have*, dejando fuera los requisitos de prioridad *Could Have* y *Won't Have*. El total de las horas estimadas por todos los requisitos fue de 100, cantidad que no puede ser cubierta con el plazo estimado de 15 días. Sin embargo, después de realizarse la selección de requisitos se descartaron 33 horas, por lo que se redujo la cantidad de horas estimadas a un total de 67 horas, esta cantidad permite cubrir cómodamente los requisitos seleccionados antes de la próxima reunión con el cliente. La selección de requisitos puede observarse en la tabla 4.14.

Codificación

Durante la fase de codificación se desarrollo el módulo comparativo, dicho módulo está conformado por una sola pantalla que contiene los resultados de los métodos calculados. La información es representada de tres formas diferentes, la primera de ellas es mediante una tabla que muestra, además de las proporciones obtenidas, las principales variables calculadas a lo largo del método. La segunda forma es mediante gráficas de pastel, una por cada método calculado y que representan las proporciones en peso seco. La tercera forma es mediante una gráfica de barras que incluye las proporciones en peso seco de todos los métodos calculados. Además, se incluyó la exportación de los datos del módulo comparativo a formato PDF y la posibilidad de agregar notas o comentarios a un costado de la pantalla. En la figura 4.27 se puede observar el módulo comparativo desarrollado.

4.9. Iteración 7: Desarrollo de complementos

La séptima y última iteración está enfocada al desarrollo de los complementos y herramientas de la aplicación. Se refiere a los elementos necesarios para ofrecer una mejor experiencia de trabajo al usuario y que no se encuentran dentro de los módulos principales de la aplicación, correspondientes a los métodos de diseño de

Tabla 4.14: Priorización, estimación de esfuerzos y selección de requisitos correspondientes a la sexta iteración. Los requisitos en color verde son aquellos que sí se incluyen en esta iteración, mientras que los de no tienen color son aquellos que quedaron descartados para este incremento.

Número	Requisito	MoSCoW	Estimado
1	Realizar las correcciones de las observaciones realizadas por el cliente al módulo Bolomey.	Must Have	10
2	El módulo comparativo deberá ser mostrado en una sola pantalla, con el fin de facilitar su visualización.	Should Have	10
3	El módulo debe representar las proporciones y demás datos de importancia mediante tablas y gráficas.	Must Have	15
4	Sólo deben de mostrarse los resultados de aquellos métodos que han sido completados al cien porciento.	Must Have	7
5	En caso de que un método no haya sido completado, mostrar únicamente el porcentaje de avance del método.	Should Have	7
6	Se deben poder exportar todos los datos del módulo comparativo a formato PDF.	Must Have	10
7	Realizar una bitácora que incluya los últimos cambios realizados a los métodos.	Could Have	10
8	Incluir el análisis granulométrico de los agregados en el módulo comparativo.	Could Have	8
9	Incluir una sección para agregar notas.	Should Have	8
10	Se debe poder expandir todas las gráficas.	Could Have	5
11	Sugerir automáticamente el método que ofrece los mejores resultados.	Won't Have	10
	Horas totales		100
	Total de horas descartadas		33
	Total de horas estimadas		67

mezclas. Entre los principales complementos a desarrollar se encuentran la gestión de proyectos y la inclusión de herramientas matemáticas útiles en cálculo de mezclas de concreto. El tiempo estimado para está iteración fue de 15 días, tiempo durante el cuál se desarrollaron los complementos y se aplicaron las fases del ciclo de desarrollo que se describen en las secciones siguientes.

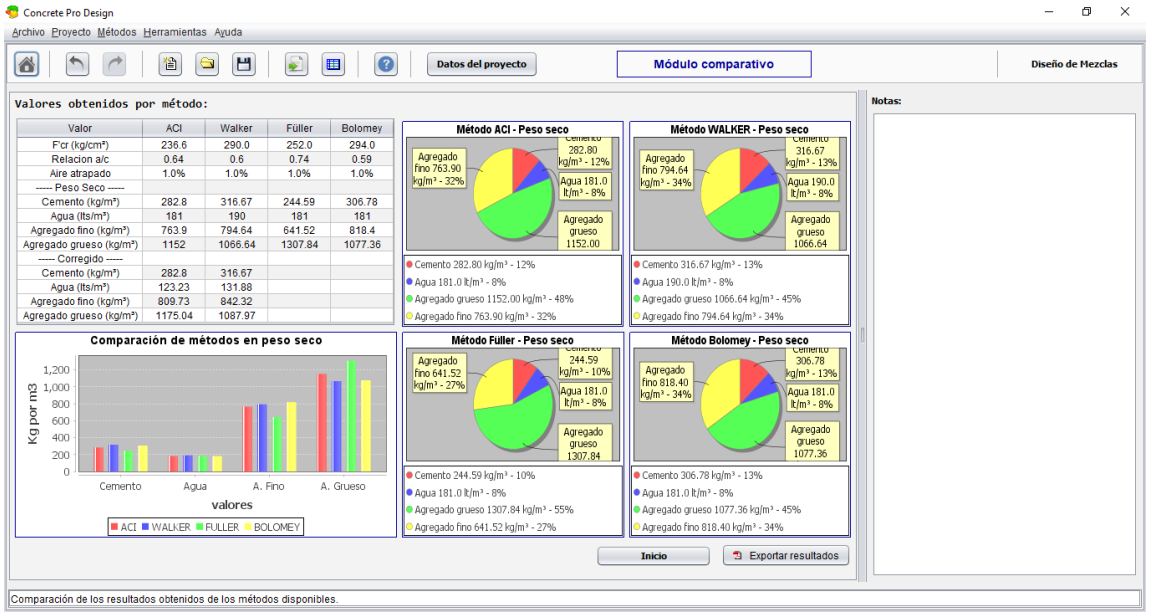


Figura 4.27: Módulo comparativo.

Selección de técnica

Los objetivos principales para la próxima reunión son los de presentar al cliente el módulo comparativo y el de obtener los requisitos principales para el desarrollo de los complementos de la aplicación. Debido a esto, se considera necesario que el cliente pueda expresar libremente sus opiniones sin un formato que guíe la reunión, por lo tanto una reunión informal se considera adecuada para esta iteración.

Reunión con el cliente

Las principales tareas para la reunión con el cliente, como ya se mencionó anteriormente, son las de presentar al cliente el módulo comparativo y el de obtener los principales requisitos para el desarrollo de los complementos de la aplicación. En la primera parte de la reunión el cliente pudo interactuar con el módulo comparativo, el cual aprobó sin mayores observaciones. Durante la segunda parte de la reunión, el cliente estableció cuales son los principales complementos que debe incluir la aplicación, así como los requisitos necesarios para su implementación.

Bitácoras

Una vez efectuada la reunión con el cliente se elaboró la bitácora correspondiente, con el fin de llevar a cabo el registro de los principales eventos ocurridos durante la reunión. En la bitácora se registraron las principales observaciones realizadas por el cliente al módulo comparativo, el cual fue presentado en la primera parte de la reunión. Además, se describen los principales requisitos, mencionados por el cliente durante, necesarios para el desarrollo de los complementos de la aplicación. En la bitácora también se pueden mencionan los puntos que se trataron y los acuerdos a los que se llegaron durante la reunión. En la figura 4.28 se puede observarse la bitácora correspondiente a esta iteración.

BITÁCORA

Número de bitácora: 7	**Fecha:** 14-ago-2017	**Tipo de técnica:** Reunión

Puntos a tratar:

1. Presentación del módulo comparativo.
2. Recopilar observaciones por parte del cliente acerca del módulo comparativo.
3. Obtención de requisitos para el desarrollo de los complementos de la aplicación.

Acuerdos:

Se verificaron que se hayan cubierto las correcciones hechas al módulo comparativo. Una vez verificados todos los requisitos el cliente aprobó el módulo comparativo. Hasta este punto ya también se han aprobado y verificado también todos los métodos de diseño de mezclas.

Se definieron cuales serían los complementos que se incluirían la aplicación, la cuales son:

1. Menú de la aplicación.
2. Crear nuevos proyectos.
3. Guardar proyectos.
4. Abrir proyectos existentes.
5. Gestionar los datos del proyecto.
6. Incluir una herramienta para el análisis granulométrico.
7. Crear una herramienta para el cálculo de la desviación estándar.
8. Gestión de las unidades de medida de la aplicación.
9. Incluir herramienta para el cálculo de la interpolación lineal.

Firmas

Responsable:	Usuario:
Alfredo Xochitemol Cruz	Ing. Miguel Ángel Daza Merino

Figura 4.28: Bitácora correspondiente a la séptima reunión.

Historias de usuario

Durante está iteración se generó la última de las historias de usuario, la cual lleva por nombre *Desarrollo de los complementos de la aplicación*. Esta historia de usuario se divide a su vez en pequeñas tareas o actividades que en conjunto permiten completar la historia de usuario principal. Entre las tareas que contiene la historia de usuario principal destacan las de gestionar proyectos, desarrollar herramientas matemáticas y la inclusión de un menú principal dentro de la aplicación. Además, en dicha historia de usuario se plasman los detalles de la misma, como su nombre, fecha, número de historia, tipo, prioridad, tiempo estimado, entre otras. La historia de usuario generada durante esta iteración puede observarse en la figura 4.29.

HISTORIA DE USUARIO					
Nombre:	Desarrollo de los complementos de la aplicación.		14-ago-17	**Núm.:**	9
Tipo de historia:	[X] Nueva	[ ] Corrección		[ ] Mejora	
Prioridad técnica:	Alta	**Prioridad de usuario:**	Alta	**Estimación (días):**	15
Historia previa núm.:	8	**Dependencias:**	8	**Riesgo:**	Medio
Descripción: • Se deben de crear herramientas para gestionar los prouectos es decir: a) abrir proyectos b) crear proyectos c) guardar proyectos. • Gestionar los datos del proyecto. • Incluir una herramienta para el análisis granulométrico. • Crear herramientas mátematicas como el cálculo de la desviación estándar y la interpolación lineal. • Gestionar las unidades de medida que se utilizan en la aplicación.					
Notas: • Se debe crear un menú principal de la aplicación, desde el cual puedan accederse a rápidamente a todas las funciones de la aplicación.					
Pruebas de funcionalidad: Se realizará un test de usabilidad en el cual se determinará qué aspectos son necesarios corregir dentro de la aplicación.					

Figura 4.29: Historia de usuario número 9, correspondiente a la séptima iteración.

Obtención de requisitos

Los requisitos para esta iteración fueron obtenidos a partir de la historia de usuario número 9, correspondientes a esta iteración (véase figura 4.29) y de las observa-

ciones realizadas por el cliente al módulo comparativo desarrollado en la iteración anterior. Los requisitos de la historia de usuario número 9 corresponden a aquellos complementos que el cliente desea que se incluyan en los complementos de la aplicación, así como los requisitos necesarios para su desarrollo. Mientras que, los requisitos de corrección corresponden a las observaciones realizadas por el cliente al módulo anterior. En la tabla 4.15 pueden observarse los requisitos pertenecientes a esta iteración.

Tabla 4.15: Requisitos correspondientes a la séptima iteración.

Número	Requisito
1	Realizar las correcciones a las observaciones realizadas por el cliente al módulo comparativo.
2	Implementar la gestión de proyectos, es decir abrir, crear y guardar proyectos.
3	Gestionar los datos del proyecto.
4	Incluir una herramienta para el análisis granulométrico.
5	Crear una herramienta para el cálculo de la desviación estándar.
6	Gestionar las unidades de medida de la aplicación.
7	Crear una herramienta para el cálculo de la interpolación lineal.
8	Crear un menú principal de la aplicación con acceso a todas las funciones de la aplicación.
9	Crear una bitácora de todos los cambios aplicados al proyecto.
10	Exportar la bitácora a PDF.

Priorización de requisitos

La priorización de requisitos, como ya se ha mencionado, se realizó mendiante el método MoSCoW, el cual clasifica a cada valor con uno de cuatro valores posibles: *Must Have* (debe tener), *Should Have* (debería tener), *Could Have* (podría tener) y *Won't Have* (no tendrá). La prioridad asignada a los requisitos mediante este método puede observarse en la tercera columna de la tabla 4.16.

Estimación de esfuerzos

La estimación de esfuerzos para cada requisito fue asignada en base a la experiencia previa, se consideraron factores como el tamaño, la complejidad, la cantidad de lineas de código, entre otros. El tamaño del módulo de complementos se considera pequeño en comparación a los módulos desarrollados anteriormente, por lo tanto el tiempo de desarrollo estimado total es bastante reducido en comparación. La estimación de esfuerzos asignada a cada requisito puede observarse en la cuarta columna de la tabla 4.16.

Selección de requisitos

La selección de los requisitos se elaboró en base a la priorización y estimación de esfuerzos realizada anteriormente. Se seleccionaron aquellos requisitos que se consideró que pudieran completarse sin contratiempos antes del plazo estimado de 15 días, correspondientes a la historia de usuario número 9 y que conforma esta iteración. Se considera que se trabajó un total de 6 horas diarias durante los 15 días, lo que da un total de 90 horas de trabajo estimado.

En esta ocasión se incluyeron todos los requisitos de prioridad *Must Have*, *Should Have* y *Could Have*, dejando únicamente fuera los requisitos de prioridad *Won't Have*. El total de las horas estimadas por todos los requisitos fue de 114, cantidad que no puede ser cubierta con el plazo estimado de 15 días. Sin embargo, después de realizarse la selección de requisitos se descartaron 33 horas, por lo que se redujo la cantidad de horas estimadas a un total de 81 horas, esta cantidad permite cubrir cómodamente los requisitos seleccionados antes de la próxima reunión con el cliente. La selección de requisitos puede observarse en la tabla 4.16.

Codificación

En la fase de codificación se desarrollaron los complementos y herramientas que dan solución a las necesidades del cliente y que se plasman en la historia de usuario número 9. Los complementos se refieren a todos aquellos elementos necesarios para ofrecer una mejor experiencia de trabajo al usuario y que no se encuentran dentro de los módulos principales de la aplicación, es decir, los módulos de métodos de diseño

Tabla 4.16: Priorización, estimación de esfuerzos y selección de requisitos correspondientes a la séptima iteración. Los requisitos en color verde son aquellos que sí se incluyen en esta iteración, mientras que los de no tienen color son aquellos que quedaron descartados para este incremento.

Número	Requisito	MoSCoW	Estimado
1	Realizar las correcciones a las observaciones realizadas por el cliente al módulo comparativo.	Must Have	10
2	Implementar la gestión de proyectos, es decir abrir, crear y guardar proyectos.	Must Have	15
3	Gestionar los datos del proyecto.	Must Have	8
4	Incluir una herramienta para el análisis granulométrico.	Should Have	8
5	Crear una herramienta para el cálculo de la desviación estándar.	Should Have	10
6	Gestionar las unidades de medida de la aplicación.	Could Have	8
7	Crear una herramienta para el cálculo de la interpolación lineal.	Should Have	10
8	Crear un menú principal de la aplicación con acceso a todas las funciones de la aplicación.	Must Have	12
9	Crear una bitácora de todos los cambios aplicados al proyecto.	Won't Have	20
10	Exportar la bitácora a PDF.	Won't Have	13
	Horas totales		114
	Total de horas descartadas		33
	Total de horas estimadas		81

de mezclas. Los complementos desarrollados durante la fase codificación se enlistan a continuación:

- Gestión de proyectos, es decir, abrir, crear y guardar proyecto.
- Gestión de los datos del proyecto.
- Herramienta para el análisis granulométrico.
- Herramienta para el análisis de la desviación estándar.
- Gestión de las unidades de medida utilizadas en la aplicación.

- Herramienta para el cálculo de la interpolación lineal.
- Menú principal de la aplicación.

Un ejemplo de los complementos desarrollados puede verse en la figura 4.30, donde se observa la herramienta para el cálculo de la desviación estándar.

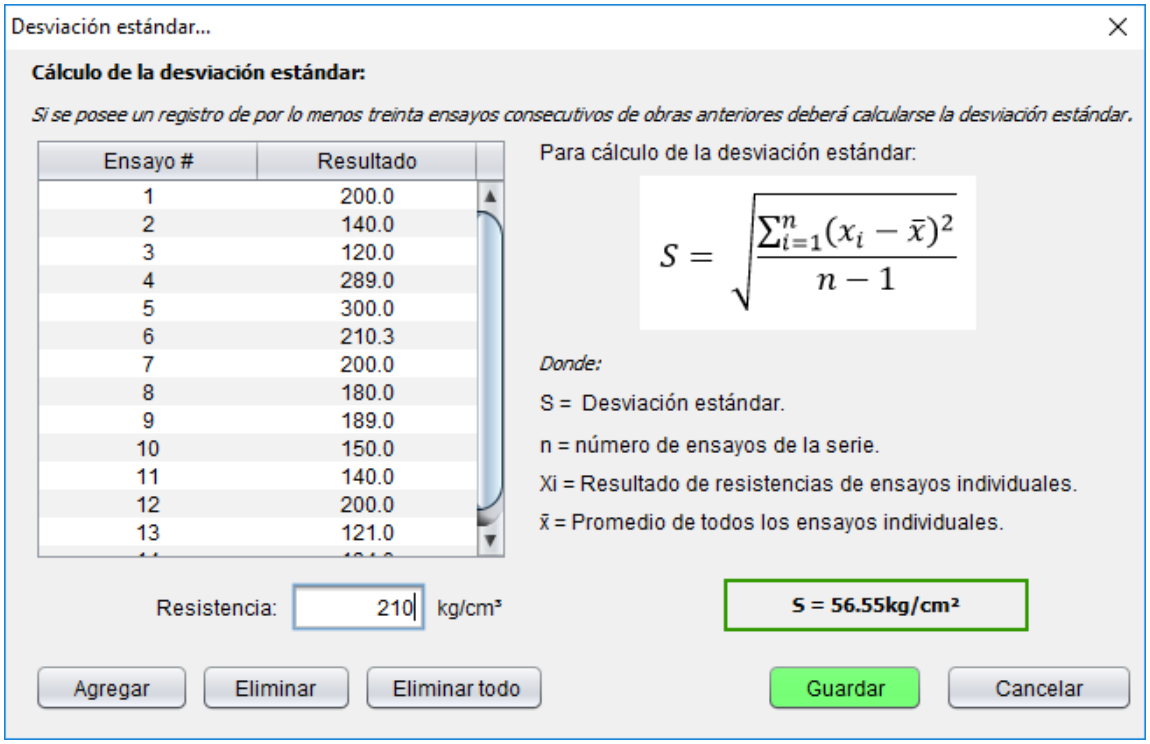


Figura 4.30: Herramienta para el cálculo de la desviación estándar.

4.10. Conclusiones

Una vez concluidas las siete iteraciones, se volvió a mostrar al cliente la aplicación. Una vez que el cliente interactuó con la aplicación terminada, éste la aprobó y se mostró satisfecho con la misma, por lo cual se dió por terminado el ciclo de desarrollo de la aplicación. Sin embargo, dependiendo de los resultados que se obtengan de las pruebas de usabilidad, es probable que sea necesario realizar correcciones.

La aplicación desarrollada permite calcular rápidamente el proporcionamiento de mezclas de concreto mediante los métodos de diseño ACI, Walker, Füller y Bolomey. A diferencia de los programas actuales, mostrados en la tabla 1.1, esta aplicación permite mostrar el desarrollo paso a paso de cada uno de los métodos. Además, ofrece una breve explicación en cada paso, por lo que puede ser utilizada para la enseñanza y aprendizaje de estos, así como para verificar resultados obtenidos en prácticas previas. Una ventaja más sobre las demás aplicaciones es que permite modificar directamente las variables que interfieren en cada paso, ofreciendo así mayor flexibilidad al proceso de cada método. También incluye un módulo comparativo que muestra mediante gráficas y tablas los resultados obtenidos en cada uno de los métodos calculados, de esa manera el usuario puede evaluar qué método le ofrece mejores resultados.

Capítulo 5

Pruebas y resultados

5.1. Introducción

Cualquier pieza de software completo, desarrollado o adquirido, puede verse como un sistema que debe probarse, ya sea para decidir acerca de su aceptación, para analizar defectos globales o para estudiar aspectos específicos de su comportamiento, tales como seguridad o rendimiento. A éste tipo de pruebas donde se estudia el producto completo se les llama *Pruebas de Sistema* (Fernández Peña, 2003).

En este capítulo se muestran las pruebas realizadas a la aplicación, así como los resultados obtenidos de las mismas. El objetivo de las pruebas es el verificar la calidad del sistema y el de identificar qué aspectos del sistema necesitan mejorías. La pruebas de usabilidad, contenido, navegación, diseño y aceptación se realizaron mediante un *test* que se aplicó al cliente y a usuarios finales.

5.2. Aspectos de la calidad evaluados

Pressman (2002) define la calidad de software como "la concordancia con los requisitos funcionales y de rendimiento explícitamente establecidos, con los estándares de desarrollo explícitamente documentados y con las características implícitas que se espera de todo software desarrollado profesionalmente". Calidad en el desarrollo de software es asegurar el mínimo de sorpresas posibles durante todas las etapas del proceso, por eso es recomendable la utilización de Estándares o modelos de calidad

(Contanzo, 2014). Para la selección de los criterios a evaluar se tomaron en cuenta los siguientes modelos de calidad de software:

- Modelo Mc Call. Este modelo fue creado por Jim Mc Call en 1977. Establece 3 perspectivas para el análisis de la calidad de software, define 11 factores y 23 criterios relacionados a estos. Las métricas que propone son preguntas que ponderan numéricamente un determinado atributo del producto de software. Después de obtener los valores para todas las métricas de un criterio específico, el promedio de todas ellas es el valor para ese criterio (Cavano y McCall, 1978).

- Modelo FURPS. Este modelo fue desarrollado por Hewlett-Packard en el año 1987. En el se desarrollan un conjunto de factores de calidad de software, bajo el acrónimo de *FURPS*: funcionalidad (*Functionality*), usabilidad (*Usability*), confiabilidad (*Reliability*), desempeño (*Performance*) y capacidad de soporte (*Supportability*) (Olsina, 2007).

- Modelo BOHEM. Este modelo propone una jerarquía de niveles, en forma de un árbol con tres ramas principales, que permiten que el software sea de utilidad: Portabilidad, Facilidad de Uso y Facilidad de Mantenimiento. Se estructura en tres niveles: Aplicaciones primarias, Construcciones Intermedias (factores) y Construcciones Primitivas, y finalmente las Métricas que determinan los valores para los criterios (construcciones primitivas) (Boehm, 1981).

- ISO/IEC 9126. El Estándar internacional (ISO), aplicable a todo tipo de software, está basado en un modelo jerárquico con tres niveles: Características, Subcaracterísticas y Métricas. En el primer nivel tiene seis características principales: Funcionalidad, Fiabilidad, Eficiencia, Facilidad de Mantenimiento, Portabilidad y Facilidad de Uso. Estas características (factores) están compuestas a su vez por 27 subcaracterísticas (subfactores) relacionadas con la calidad externa, y 21 subcaracterísticas relacionadas con la calidad interna (ISO/IEC, 2001).

5.3. Test de usabilidad y aceptación

El cuestionario o *test* de usabilidad y aceptación se aplicó a dos perfiles diferentes de usuario: *docente y alumno*. Ambos perfiles de usuario se encuentran estrictamente relacionados con los principales propósitos de la aplicación, la enseñanza y el aprendizaje. En total, el cuestionario se aplicó a seis usuarios diferentes, entre los cuales se encuentran dos docentes y cuatro alumnos.

El cuestionario tiene como objetivo medir cinco diferentes aspectos de la calidad: usabilidad, contenido, navegación, diseño y aceptación. El cuestionario está enfocado a nuevos usuarios, es decir, usuarios que nunca antes han interactuado con la aplicación sin importar el perfil al que pertenezcan.

En la primera parte del *test* el usuario tendrá unos minutos para familiarizarse con la aplicación y posteriormente se le solicitará que realice tareas específicas dentro de la aplicación. Una vez que el usuario ha realizado las tareas, procede a contestar el cuestionario, en un principio se le pide que se conteste unas preguntas que ayudan a definir cuál es el perfil del usuario y qué tan familiarizado se encuentra con el uso de la computadora y aplicaciones similares. Las preguntas siguientes se encargan de medir los criterios de usabilidad del sistema y deben ser contestadas en base a la escala de *Likert*.

La escala de *Likert*, nombrada así por Rensis Likert, es una escala psicométrica utilizada comúnmente en cuestionarios, donde el usuario debe especificar su nivel de acuerdo o desacuerdo con una declaración. La escala tipo *Likert* mide tanto el grado positivo como neutral y negativo de cada pregunta o declaración. Los posibles valores de la escala utilizada en este cuestionario son: totalmente en desacuerdo, algo en desacuerdo, ni de acuerdo ni desacuerdo, algo de acuerdo y totalmente de acuerdo. El *test* de usabilidad y aceptación aplicado, puede observarse en el anexo B. A continuación, se describen brevemente los aspectos de la calidad evaluados en el *test*.

Usabilidad

La usabilidad es una medida que define el grado de facilidad y rapidez con el cual la aplicación puede ser utilizada por los usuarios para conseguir objetivos específicos.

Las preguntas de usabilidad incluidas en el*test* evalúan qué tan fácil es para los usuarios realizar las tareas por primera vez, la eficiencia, velocidad y el manejo de errores de la aplicación. Las preguntas 1 a 6 del cuestionario se encargan de medir el factor de usabilidad de la aplicación.

Contenido

El objetivo de las preguntas de contenido es evaluar la calidad del contenido e información presentada dentro de la aplicación. Ayudan a determinar si el contenido mostrado es el correcto y si le permite al usuario hacerse una idea concreta de la información que se presenta en pantalla y su significado. Las preguntas 7 a 10 del cuestionario evalúan la calidad del contenido de la aplicación.

Navegación

Las preguntas de navegación permiten evaluar que tan fácil y claro resulta para el usuario poder desplazarse a través de las diferentes pantallas de la aplicación. Además, ayudan a determinar si los elementos de desplazamiento se encuentran ubicados correctamente dentro de la aplicación y si la información presentada permite al usuario identificar en que parte de la aplicación se encuentra en todo momento. Las preguntas 11 a 15 del cuestionario evalúan la navegabilidad de la aplicación.

Diseño

El objetivo de las preguntas de diseño es evaluar si los elementos de la aplicación tienen un diseño cuidado, claro, atractivo visualmente y acorde con la temática. Incluye preguntas que ayudan a determinar si los elementos dentro de la aplicación representan de forma adecuada la información, si los iconos e imágenes se encuentran correctamente relacionadas con el contenido, si los colores utilizados son agradables para el usuario, etcétera. Las preguntas 16 a 21 del cuestionario evalúan el diseño se la aplicación.

Aceptación

Las preguntas de aceptación permiten determinar el grado de satisfacción del cliente referente al tiempo empleado, a los resultados obtenidos, al contenido y herramientas incluidas y en general con el uso de la aplicación. Las preguntas 22 a 27 del cuestionario evalúan la aceptación del usuario hacia la aplicación.

5.4. Resultados obtenidos

El test de usabilidad y aceptación como ya se mencionó se aplicó a seis usuarios diferentes (dos docentes y cuatro alumnos), cuyos resultados son representados a través de seis gráficas de barras que representa los datos obtenidos de cada uno de los aspectos de calidad evaluados. En las secciones siguientes se muestran los resultados obtenidos.

Resultados de usabilidad

Las preguntas utilizadas para medir el criterio de usabilidad, corresponden a las preguntas del número 1 al 6 del *test* aplicado a los usuarios. Estas preguntas pueden observarse dentro del anexo B. Los resultados obtenidos a las seis preguntas de usabilidad se encuentran representados en la figura 5.1, donde puede observase que la mayoría de las respuestas son positivas, por lo que puede concluirse que la usabilidad de la aplicación es buena. Es importante mencionar que de las preguntas correspondientes a la usabilidad, la pregunta 3, que evalúa la velocidad de la aplicación, es la que obtuvo la menor puntuación y se convierte a un aspecto de mayor prioridad en próximas mejoras.

Resultados de contenido

Las preguntas utilizadas para medir el criterio de contenido, corresponden a las preguntas del número 7 al 10 del cuestionario aplicado a los usuarios. Estas preguntas pueden observarse dentro del anexo B. Los resultados obtenidos a las cuatro preguntas de contenido se encuentran representados en la figura 5.2, donde además puede observase que la mayoría de las respuestas son positivas, por lo que puede

Figura 5.1: Resultados de usabilidad.

concluirse que el criterio del contenido de la aplicación es bueno. Cabe mencionar que de los aspectos que se evalúan en las preguntas correspondientes al contenido, la inferencia de contenido es la que obtuvo la menor calificación, lo que la convierte un aspecto de prioridad a mejorar en próximas mejoras.

Resultados de navegación

Las preguntas utilizadas para medir el criterio de navegación, corresponden a las preguntas del número 11 al 15 del cuestionario aplicado a los usuarios. Estas preguntas pueden observarse dentro del anexo B. Los resultados obtenidos a las cinco preguntas de navegación se encuentran representados en la figura 5.3, donde además puede observase que la mayoría de las respuestas son positivas, por lo que puede concluirse que la navegabilidad de la aplicación es buena. Aunque en general se obtuvieron resultados positivos, los aspectos de visibilidad de funciones y de desplazamiento pueden mejorar.

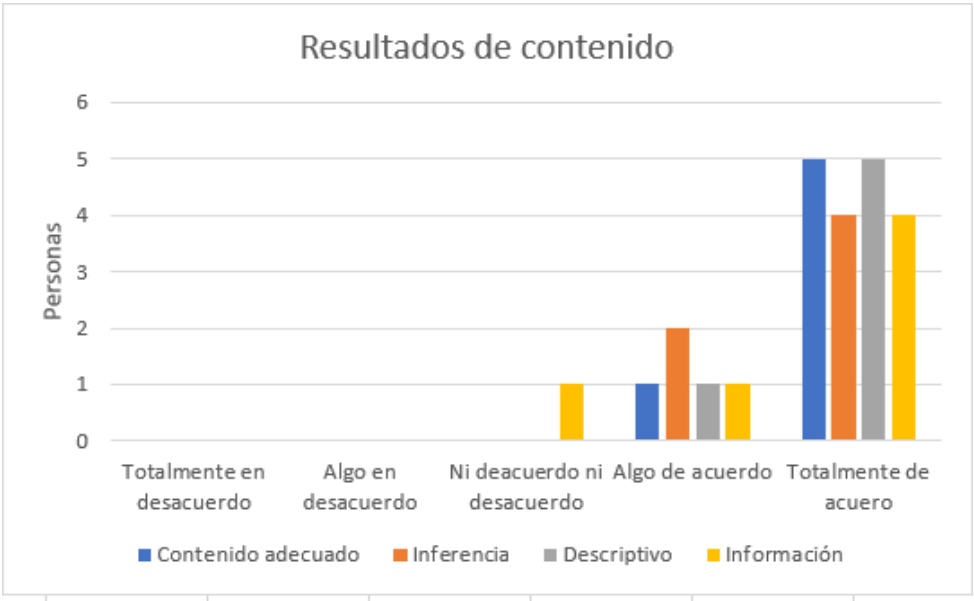


Figura 5.2: Resultados de contenido.

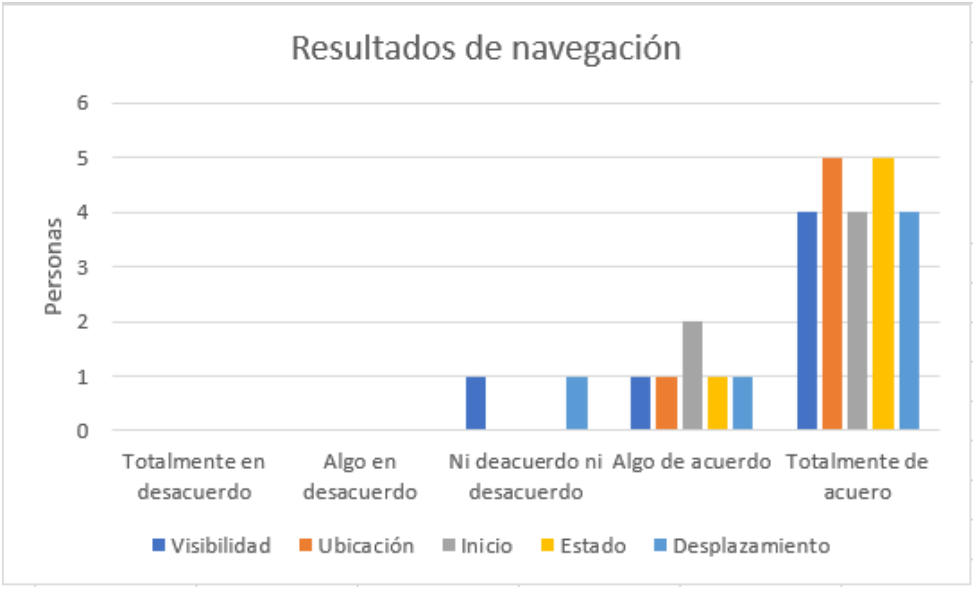


Figura 5.3: Resultados de navegacion.

Resultados de Diseño

Las preguntas utilizadas para medir el criterio de diseño, corresponden a las preguntas del número 16 al 21 del cuestionario aplicado a los usuarios. Estas preguntas

pueden observarse dentro del anexo B. Los resultados obtenidos a las seis preguntas de diseño se encuentran representados en la figura 5.4, donde además puede observase que la mayoría de las respuestas son positivas, por lo que puede concluirse que diseño de la aplicación satisface al cliente. Cabe mencionar que de los aspectos evaluados en el diseño, el que corresponde a la selección de colores fue evaluado con la puntuación más alta posible, sin embargo los aspectos de cantidad de imágenes y tamaño de los gráficos fueron los que obtuvieron la calificación más baja y necesitan mejorar.

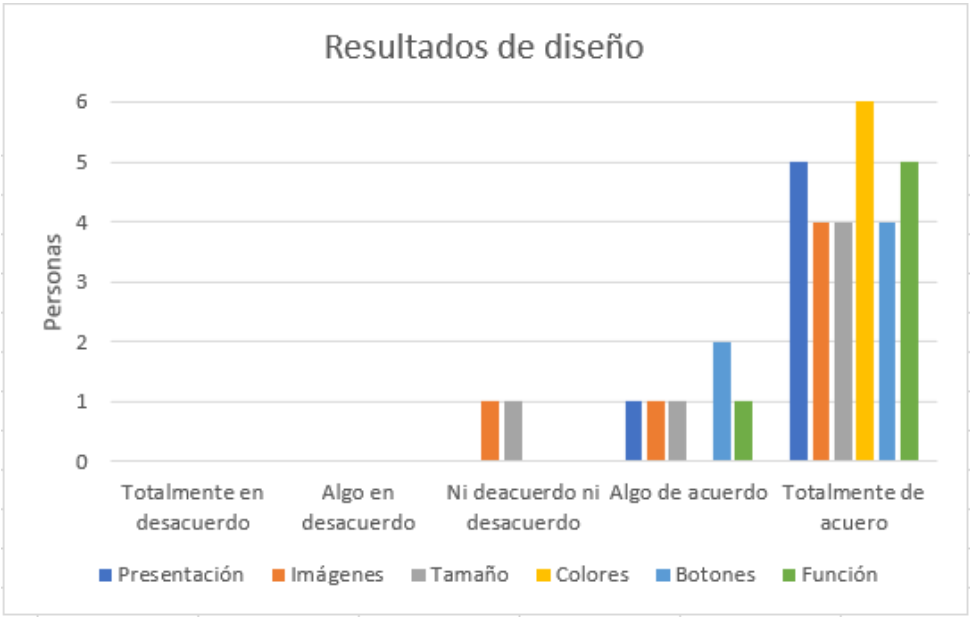


Figura 5.4: Resultados de diseño.

Resultados de Aceptación

El último de los criterios a eveluar es el de aceptación por parte del usuario. Las preguntas utilizadas para medir este criterio, corresponden a las preguntas del número 22 al 27 del cuestionario aplicado a los usuarios. Estas preguntas pueden observarse dentro del anexo B. Los resultados obtenidos a las seis preguntas de aceptación se encuentran representados en la figura 5.5, donde además puede observase que la mayoría de las respuestas son positivas, por lo que puede concluirse que la aceptación hacia la aplicación por parte del usuario es buena. El criterio de acep-

tación de la aplicación fue el mejor calificado de los cinco criterios evaluados, sin embargo, aún existen aspectos como el del tiempo empleado en el desarrollo de los métodos que pueden mejorar aún más.

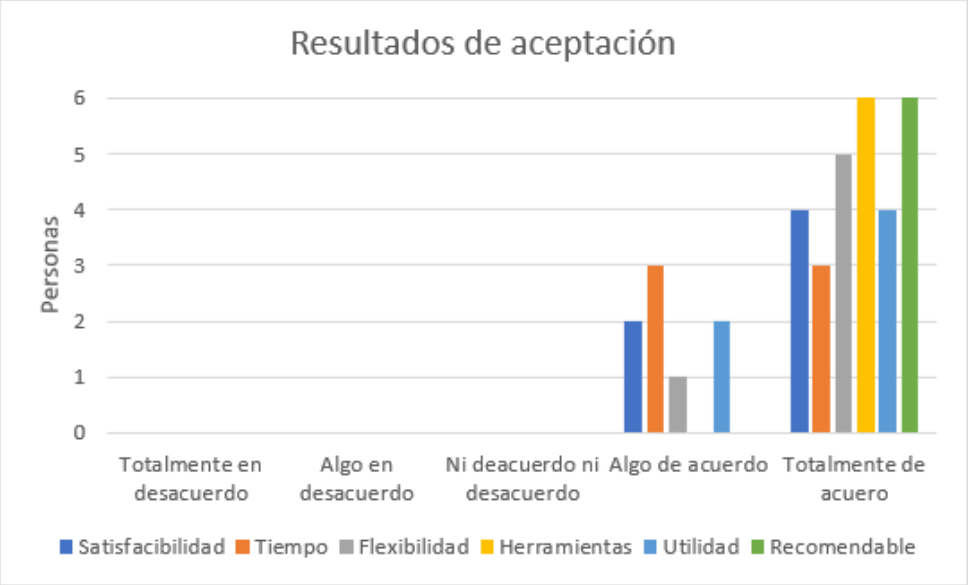


Figura 5.5: Resultados de aceptación.

5.5. Interpretación de los resultados

El primer paso para interpretar los resultados es asignar un puntaje a cada posible respuesta en la escala de *Likert*, en la tabla 5.1 puede observarse el valor asignado a cada posible respuesta de la escala.

El siguiente paso es obtener la puntuación total de los cuestionarios aplicados. Las puntuaciones se obtienen sumando los valores obtenidos en cada respuesta. La mínima puntuación posible por cuestionario es de 27 y la máxima puntuación posible es de 135. Después se aplica una sencilla fórmula para obtener el promedio de respuesta de los participantes: PT/NP, donde PT es la puntuación total obtenida y NP es el número de preguntas. Entonces se obtiene una puntuación del uno al cinco, que se compara con la escala de *Likert*, para su análisis e interpretación. Sin embargo, para obtener la medición de todos los encuestados se suma el promedio

Tabla 5.1: Valor asignado a cada respuesta de la escala de Likert.

Respuesta	Valor asignado
Totalmente en desacuerdo	1
Algo en desacuerdo	2
Ni de acuerdo ni desacuerdo	3
Algo de acuerdo	4
Totalmente de acuerdo	5

de cada uno y se divide entre el número de participantes. En la tabla 5.2, puede observarse los puntajes y promedios obtenidos por cada uno de los participantes.

Tabla 5.2: Resultados de los cuestionarios por participante.

Número	Perfil	Puntaje	Promedio
1	Docente	125	4.6
2	Docente	126	4.6
3	Estudiante	131	4.8
4	Estudiante	123	4.5
5	Estudiante	130	4.8
6	Estudiante	128	4.7
Total		763	4.6

El valor promedio total corresponde a 4.6, el cual al ser comparado con la escala de *Likert* (véase figura 5.6), se puede concluir que, en general, la aplicación desarrollada complace satisfactoriamente los criterios de calidad evaluados.

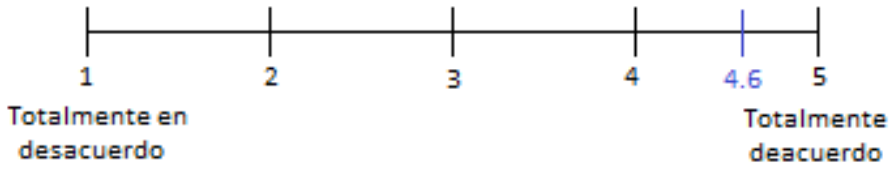


Figura 5.6: Promedio total dentro de la Escala de Likert.

5.6. Conclusiones

Se aplicó el cuestionario usabilidad y aceptación (véase anexo B) a seis usuarios diferentes: cuatro alumnos y dos docentes. Dicho cuestionario se divide en cinco grupos de preguntas, donde cada grupo corresponde a uno de los cinco criterios de calidad evaluados (usabilidad, contenido, navegación, diseño y aceptación), como puede observarse en las figuras 5.1, 5.2, 5.3, 5.4 y 5.5. La mayoría de las respuestas obtenidas son satisfactorias, por lo que puede concluirse que los cinco criterios evaluados cumplen satisfactoriamente las necesidades del usuario.

Asimismo, en la tabla 5.2 puede observarse que el promedio obtenido por cada usuario es satisfactorio, de igual forma, en la figura 5.6 puede verse que el puntaje promedio total es de 4.6, correspondiente a los seis usuarios evaluados, el cual es satisfactorio dentro de la escala de *Likert*. En base a lo anterior, se concluye que la aplicación muestra altos índices de aceptación por parte de los usuarios, sin embargo, aún existen puntos importantes que mejorar en próximas versiones de la aplicación.

Capítulo 6

Conclusiones y trabajos futuros

En este capítulo se presentan las conclusiones obtenidas tras el desarrollo de una aplicación didáctica para diseño y evaluación de mezclas de concreto, así como los trabajos futuros que pueden seguir realizándose en base al trabajo desarrollado durante este proyecto.

6.1. Conclusiones

El objetivo de este trabajo fue el de desarrollar una aplicación que facilitara el cálculo de los principales métodos de diseño mezclas de concreto y que además permitiera aprender acerca de su desarrollo. Esto debido principalmente a que, en la actualidad no existen programas de cómputo para el diseño de mezclas de concreto que permitan el uso de varios métodos, que muestren el desarrollo del método y que además permitan comparar los resultados obtenidos entre los distintos métodos. La comparativa de los principales programas de cómputo evaluados puede observarse en la tabla 1.1.

El desarrollo de la aplicación se realizó mediante la metodología de desarrollo descrita en el capítulo 3, donde se describe ampliamente. Entre sus principales características destacan que se trata de una metodología de desarrollo ágil enfocada a pequeños equipos de desarrollo y que surge de la combinación de distintos atributos del modelo de desarrollo incremental y de la metodología de desarrollo ágil *Extreme Programming* (XP), pero adaptándolos a pequeños equipos de desarrollo. En esta

metodología se propone la implementación de roles, funciones, actividades, bitácoras e historias de usuario. En la figura 3.3 puede observar el proceso de la metodología utilizada.

Para el desarrollo de este proyecto, dicha metodología fue llevada a cabo por una sola persona, quien asumió la mayoría de los roles y funciones descritos en la metodología empleada. El proceso o ciclo de desarrollo de la metodología se realizó a través de siete iteraciones:

1. Desarrollo del prototipo de la interfaz de la aplicación.
2. Desarrollo del método ACI.
3. Desarrollo del método Walker.
4. Desarrollo del método Füller.
5. Desarrollo del método Bolomey.
6. Desarrollo del módulo comparativo.
7. Desarrollo de complementos.

Una vez concluidas las siete iteraciones, se obtuvo una aplicación que satisface los principales requerimientos solicitados por el cliente, los cuales se encuentran descritos en la tabla 4.1. La aplicación desarrollada permite calcular de forma rápida el proporcionamiento de mezclas de concreto mediante los métodos de diseño ACI, Walker, Füller y Bolomey. A diferencia de los programas actuales (véase tabla 1.1) esta aplicación permite mostrar el desarrollo paso a paso de cada uno de los métodos, además, incluye un módulo comparativo que permite comparar los resultados obtenidos en cada método en una sola pantalla, con el fin de que el usuario pueda realizar un análisis de los datos y defina cuál de los métodos calculados ofrece mejores resultados. Una ventaja más sobre otras aplicaciones es que permite modificar directamente las variables las variables que interfieren en cada paso, ofreciendo así mayor flexibilidad al procedimiento de cada método.

Para medir el nivel satisfacción por parte del usuario se aplicó a diversos usuarios un test de usabilidad y aceptación, en cual se evaluaron cinco criterios de la calidad:

usabilidad, contenido, navegación, diseño y aceptación. En base a los resultados obtenidos y como puede observarse en las figuras 5.1, 5.2, 5.3, 5.4 y 5.5 la mayoría de las respuestas son positivas, por lo que puede concluirse que los cinco criterios evaluados cumplen satisfactoriamente las necesidades del usuario. Sin embargo, se identificaron aspectos importantes que mejorar en próximas versiones de la aplicación.

Con base en lo anterior se puede dar una respuesta positiva a la pregunta de investigación presentada al inicio de esta tesis: sí es factible el desarrollo de una aplicación didáctica para el cálculo de los principales métodos de diseño utilizando herramientas y metodologías de desarrollo ágil.

6.2. Trabajos futuros

Como trabajo futuro se propone continuar con los siguientes puntos:

- Incluir más métodos de diseño de mezclas. Existe una gran cantidad de métodos de diseño de mezclas, por lo que podrían incluirse más métodos a la aplicación, así como al módulo comparativo.
- Incluir una bitácora que registre todos los cambios en el proyecto. De esta manera, el usuario podrá comparar como han ido variando los resultados finales conforme se modifican la variables del método.
- Desarrollar una aplicación móvil que de igual forma permita realizar el cálculo de diseño de mezclas bajo los métodos ACI, Walker, Füller y Bolomey. Además de que incluya funciones para comparar los resultado obtenidos, tales como las del módulo comparativo.
- Realizar la evaluación de la aplicación por parte de una concretera o un organismo especializado como Instituto Mexicano del Cemento y del Concreto.

Bibliografía

ISO/IEC (2001). *Software Engineering-Product Quality: Quality model*, volumen 1. ISO/IEC.

Alfonso, P., Mariño, S., y Godoy, M. (2011). Propuesta metodológica para la gestión de proyecto de software ágil basado en la web. zulia, ve. *Revista Multiciencias*, 11(4):397.

Beck, K. (1999). *Extreme Programming Explained: Embrace Change*. Pearson Education.

Ben-Zahia, M. A. y Jaluta, I. (2014). Criteria for selecting software development models. *2014 Global Summit on Computer & Information Technology (GSCIT)*, pp. 1–6.

Boehm, B. W. (1976). Software engineering. *IEEE*, pp. 1226–1241.

Boehm, B. W. (1981). *Software engineering economics*, volumen 197. Prentice-hall Englewood Cliffs (NJ).

Bolívar, O. G. (1987). *Guía práctica para el diseño de mezclas de hormigón*. Universidad Nacional de Colombia sede Medellín. Medellín, Colombia.

Cataldi, Z., F. Lage, R. P., y García Martinez, R. (1999). Ingeniería de software educativo. *Proceedings del V Congreso Internacional de Ingeniería Informática*, pp. 185–199.

Cavano, J. P. y McCall, J. A. (1978). A framework for the measurement of software quality. En *ACM SIGMETRICS Performance Evaluation Review*, volumen 7, pp. 133–139. ACM.

Cervantes Ojeda, J. y Gómez Fuentes, M. C. (2012). Taxonomía de los modelos y metodologías de desarrollo de software más utilizados. *Universidades*, 52:37–47.

Chaparro Lemus, L. O. y Gómez Estupiñan, J. F. (2012). Una visión del desarrollo de software utilizando modelos. *Gerencia Tecnológica Informática*, pp. 69–82.

Choquecchambi Mamani, J. G., Cutisaca Bellido, K. H., y Quispe Galindo, J. C. (2013). Comparación de 4 métodos para el diseño de mezclas. Technical report, Universidad Peruana Unión Facultad Ingeniería y Arquitectura.

Contanzo, M. A. (2014). Comparación de modelos de calidad, factores y métricas en el ámbito de ingeniería de software. pp. 1–36.

Dimezco 2000 (2013). Bienvenido a dimezco. Recuperado de https://dimezco2000.wordpress.com/2013/09/30/bienvenido-a-dimezco-2000/.

Fernández Peña, J. M. (2003). Prueba de software basada en componentes. estado actual. Technical report, Instituto Politécnico Nacional (IPN).

García Pacheco, I. y García Matías, J. (2008). A methodology based on effective practices to develop educational software. *Computación y Sistemas*, 11(4):313–332.

Gottberg de Noguera, A., Noguera Altuve, G., y Gottberg Noguera, M. A. (2011). Propuesta pedagógica: una metodología de desarrollo de software para la enseñanza universitaria. *Universidades*, pp. 49–57.

Han, J., Wang, K., Wang, X., y Monteiro, P. (2016). 2D image analysis method for evaluating coarse aggregate characteristic and distribution in concrete. *Construction and Building Materials*, 127:30–42.

Harrington, J. L. (2003). *SQL Clearly Explained Second Edition*. Morgan Kaufmann Publishers.

Kosmatka, S. H., Kerhoff, B., Panarese, W. C., y Tanesi, J. (2004). *Diseño y Control de Mezclas de Contreto*. Portland Cement Association.

Lindstrom, L. y Jeffries, R. (2004). Extreme programming and agile software development methodoligies. *Information Systems Management*, pp. 41–54.

Miller, S. A., Monteiro, P., y Ostertag, C. P. y Horvath, A. (2016). Comparison indices for design and proportioning of concrete mixtures taking environmental impacts into account. *Cement and Concrete Composites*, 68:131–143.

Mitre Hernández, H. E., Ortega Martínez, E., y Lemus Olalde, C. (2014). Estimación y control de costos en métodos ágiles para desarrollo de software: un caso de estudio. *Ingeniería Investigación y Tecnología*, pp. 403–418.

Mohammed, N., Niazi, N., Alshayeb, M., y Mahmood, S. (2012). Exploring software security approaches in software development lifecycle: A systematic mapping study. *Computer Standards & Interfaces*, 50:107–115.

Noél, R., Munoz, R., Becerra, C., y Villarroel, R. (2016). Developing competencies for software requirements analysis through project based learning. En *Computer Science Society (SCCC), 2016 35th International Conference of the Chilean*, pp. 1–7. IEEE.

Olsina, L. (2007). Ingeniería web; marco de medición y evaluación de calidad. *Departamento de informática. Universidad Nacional de San Luis-La Rioja–Catamarca*.

Orjuela Duarte, A. y Rojas, M. (2008). Las metodologías de desarrollo Ágil como una oportunidad para la ingeniería del software educativo. *Revista Avances en Sistemas e Informática*, pp. 159–171.

Perea, R., Fernández, I., Arroyo, M., Rodríguez, J. A., Camacho, E., y Montesinos, P. (2016). Multiplatform application for precision irrigation scheduling in strawberries. *Agricultural Water Management*, pp. 194–201.

Pressman, R. (2002). *Ingeniería del Software. Un enfoque práctico*. McGraw-Hill.

Qureshi, M. R. J. (2012). Agile software development methodology for medium and large projects. *IET Digital Library*, 6:358–363.

Rivva, E. (1996). *Diseño de mezclas*. Hozlo S.CR.L.

Rojas, J. J. (2008). Programas para el diseño de mezclas de concreto. Recuperado de https://civilgeeks.com/2011/03/12/programa-para-diseno-de-mezclas-de-concreto/.

Satoto, K. I., Isnanto, R. R., Kridalukmana, R., y Martono, K. T. (2016). Optimizing mysql database system on information systems research, publications and community service. *Information Technology, Computer, and Electrical Engineering (ICITACEE)*, pp. 1–5.

Singhto, W. y Denwattana, N. (2016). An experience in blending the traditional and agile methodologies to assist in a small software development project. *13th International Joint Conference on Computer Science and Software Engineering (JCSSE)*, pp. 1–5.

Sobolev, K. (2003). The development of a new method for the proportioning of high-performance concrete mixtures. *Cement & Concrete Composites*, 23:901–907.

Tomas, P., Escalona, M. J., y Mejias, M. (2013). Open source tool for measuring the internal quality of java software products. a survey. *Computer Standards & Interfaces*, 36:244–255.

Trivedi, P. y Sharma, A. (2003). A comparative study between iterative waterfall and incremental software development life cycle model for optimizing the resources using computer simulation. *Information Management in the Knowledge Economy (IMKE)*, pp. 188–194.

Yagüe, A., Diaz, J., y González, E. (2016). An exploratory study in communication in agile global software development. *Computer Standards & Interfaces*, 23:184–197.

Anexo A

Entrevista para el análisis de requerimientos

Entrevista para el análisis de requerimientos

PREGUNTA	RESPUESTA	OBSERVACIONES
Preguntas referentes al problema		
1. ¿Qué problemas surgen al no contar con una aplicación, para el diseño de mezclas de concreto?	a. Se vuelve lento b. Errores de cálculo c. Valores de interpretación	Se requiere de un aplicación que calcule en automático los valores.
1.1 ¿Por qué existe este problema?	a) Lento. Porque se tarda mucho tiempo en calcular los valores finales.	La aplicación debe calcular estos valores de forma rápida.
	b) Errores de cálculo. Al realizarse los cálculos de forma manual se está propenso a errores humanos.	Los cálculos deben mostrarse en su totalidad, para decidir en cual se requiere un cambio.
	c) Valores de interpretación. Los métodos contienen datos que quedan a criterio del diseñador, por lo que puede haber errores.	Los valores que sean de interpretación deben ser calculados como una receta. Dando la opción de cambiar los valores.
1.2 ¿Cómo lo resuelve ahora?	a) Lento. No hay forma de resolver este problema. Solamente con se puede agilizar con la práctica continua.	
	b) Errores de cálculo. Actualmente no existe una forma rápida para solucionar e identificar los errores de cálculo.	Por lo regular se revisa paso a paso todo el método, hasta encontrar el error.
	c) Valores de interpretación. Hay pasos en el diseño de mezclas, en los que el diseñador debe elegir cuál es el valor más apropiado.	Existen metodologías para determinar este valor en base a los demás valores.
1.3 ¿Cómo le gustaría que se resolviera?	a) Lento. Mediante una aplicación en la que solo se ingresen los valores de entrada y arroje resultados.	
	b) Errores de cálculo. La aplicación debe realizar los cálculos en automático para aumentar la fiabilidad de los resultados.	Se debe permitir el cambio en los valores durante cualquier paso del método.
	c) Valores de interpretación. La aplicación debe interpretar en	El usuario debe ser capaz de modificar los

	automático estos valores y acercarlos lo más posible a la mejor interpretación.	valores interpretados automáticamente.
Preguntas referentes a los métodos		
2. ¿Cuántos y cuáles modelos de diseño de mezclas se utilizarán?	Los métodos que se utilizaran son: • ACI 211 • Norma ASTM • Bolomey • Walker • Füller	
Métodos de diseño propuestos: ACI 211, ASTM, Módulo de fineza, Walker y Füller.	De los métodos propuestos solo hubo un cambio: el método de finura por el método Bolomey.	
Preguntas referentes a la representación de la información		
3. ¿Cómo desea que se muestre la información (reporte, graficas, tablas, etc.)?	Se desea que la información se presente en formas de tablas. Pero utilizando dos nomenclaturas una para laboratorio y otra para obra.	
4. ¿Cómo desea que se compare la información entre varios métodos(reporte, graficas, tablas, etc.)?	De momento sólo se utilizarán dos tablas. Ambas con la misma información, pero representadas de formas distintas, la de laboratorio utilizará medidas como litros, kilos, etc. Mientras que la de obra utilizara medidas como bultos, cubetas y botes.	También se mostrará una tabla comparativa de los métodos utilizados.
Referente a las funciones de la aplicación		
5. ¿Qué elementos y funciones considera necesarias para la aplicación?	De momento no hay más funciones que se consideren necesarias.	Probablemente, durante el transcurso del proyecto irán surgiendo nuevas ideas.
6. ¿Cómo desea que se guarden los resultados (documentos[PDF, EXCEL], imágenes, etc.)?	Los resultados se deben poder exportarse a PDF (formato para imprimir) y a EXCEL (para ir guardado los resultados obtenidos).	
Referente a los usuarios de la aplicación		

7. ¿Cuántos usuarios serán necesarios para la aplicación (Profesor, alumno, experto, etc.)?	No será necesario el uso de múltiples usuarios. Debido a que se desea que todos tengan acceso a todas las funciones por igual	
8. En caso de múltiples usuarios ¿A qué funciones tendrá acceso cada usuario?	No será necesario el uso de múltiples usuarios.	
9. ¿Los usuarios tienen experiencia con aplicaciones informáticas de este tipo?¿Qué otras aplicaciones?	Se han utilizado diferentes programas, pero ninguno ha cumplido con las expectativas esperadas.	Ya sea porque tienen muchas funciones o inclusive muy pocas.
Requerimiento de la interfaz e implementación		
10. ¿Sobre qué sistema operativo deberá operar la aplicación?	El usuario prefiere el uso de la aplicación bajo el sistema operativo Windows.	Al desarrollarse en Java será multiplataforma.
11. ¿Se ha intentado con anterioridad implementar una aplicación parecida?	Solamente las aplicaciones encontradas en internet, pero ninguna que se haya intentado desarrollar a la medida de las necesidades.	
12. ¿Hay algún requerimiento sobre la interfaz que considere necesarios (colores, ventanas, imágenes)?	Diseño quedará a criterio del desarrollador de la aplicación.	Conforme se vaya presentando la aplicación, probablemente surjan recomendaciones
13. ¿Cuáles son los requerimientos de seguridad necesarios?	Al no ser necesario el uso de usuarios, no será necesario el uso de medidas de seguridad y acceso a funciones.	
14. ¿Qué tipo de documentación necesita?	Se requerirá: • Manual de usuario	
Observaciones generales		

- La aplicación se basará en el diseño de mezclas para concreto normal. Se dejará a un lado el uso para concreto con aditivos, pues existen demasiados e incluirlos todos dificultaría el desarrollo de la aplicación.

- Hay muchos elementos que pueden afectar los condiciones del concreto, desde la calidad de los elementos hasta las condiciones del concreto. Por esta razón, no siempre se pueden obtener los mismos resultados en las pruebas de laboratorio, aunque se utilice siempre la misma fórmula.

- Se realizarán las pruebas de laboratorio al concreto elaborado con las proporciones obtenidas. Si no se obtienen los resultados esperados se deberá regresar a la aplicación y cambiar los valores que se consideren necesarios.

Anexo B

Test de usabilidad y aceptación

TEST DE USABILIDAD Y ACEPTACIÓN

Herramienta de evaluación para una aplicación enfocada al diseño y evaluación de mezclas de concreto

Tipo de usuario: **Nuevo**

Instrucciones: Este test deberá de aplicarse a usuarios nuevos, es decir, usuarios que no hayan utilizado anteriormente la aplicación. Cada participante tendrá unos minutos para familiarizarse con la aplicación y después deberá realizar las siguientes tareas:

1. Introducir o modificar los datos iniciales del proyecto actual.
2. Calcula el diseño de mezclas mediante el método ACI o Walker.
3. Calcula el diseño de mezclas mediante el método Füller o Bolomey.
4. Exportar los resultados de algún método calculado a formato PDF.
5. Visualizar el módulo comparativo.
6. Exportar el módulo comparativo a formato PDF.
7. Modificar los datos generales del proyecto.
8. Utilizar alguna de las herramientas matemáticas que se encuentran en la barra de menús de la aplicación.
9. Guardar un proyecto.
10. Abrir un proyecto guardado.
11. Cerrar la aplicación.

Una vez realizada estas tareas, responda el cuestionario a continuación. Para cada pregunta seleccione el valor en la escala que considere adecuado. Al final del cuestionario se encuentra una sección para comentarios, donde podrá anotar las observaciones que considere de utilidad.

Perfil del usuario

¿A qué se dedica?

¿Cuántas veces a la semana utiliza la computadora?

¿Con qué propósito utiliza la computadora? ¿Recreación, trabajo, investigación?

¿Ha recibido algún tipo de capacitación para usar la computadora?

¿Ha utilizado anteriormente algún software para el cálculo de mezclas de concreto?

¿Cuántas veces a utilizado está aplicación?

Usabilidad

1. ¿Le resultó fácil llevar a cabo las tareas solicitadas?

Totalmente en desacuerdo	Algo en desacuerdo	Ni de acuerdo ni desacuerdo	Algo de acuerdo	Totalmente de acuerdo
1	2	3	4	5

2. ¿Considera que resulta fácil el uso de la aplicación?

Totalmente en desacuerdo	Algo en desacuerdo	Ni de acuerdo ni desacuerdo	Algo de acuerdo	Totalmente de acuerdo
1	2	3	4	5

3. ¿Fue rápida la velocidad con la que se desempeña la aplicación?

Totalmente en desacuerdo	Algo en desacuerdo	Ni de acuerdo ni desacuerdo	Algo de acuerdo	Totalmente de acuerdo
1	2	3	4	5

4. ¿No hubo necesidad de solicitar ayuda para utilizar la aplicación?

Totalmente en desacuerdo	Algo en desacuerdo	Ni de acuerdo ni desacuerdo	Algo de acuerdo	Totalmente de acuerdo
1	2	3	4	5

5. ¿Supo en todo momento qué hacer a continuación?

Totalmente en desacuerdo	Algo en desacuerdo	Ni de acuerdo ni desacuerdo	Algo de acuerdo	Totalmente de acuerdo
1	2	3	4	5

6. ¿El sistema estuvo libre de errores durante el tiempo que lo utilizó?

Totalmente en desacuerdo	Algo en desacuerdo	Ni de acuerdo ni desacuerdo	Algo de acuerdo	Totalmente de acuerdo
1	2	3	4	5

Contenido

7. En general, ¿Considera que el contenido mostrado es el adecuado?

Totalmente en desacuerdo	Algo en desacuerdo	Ni de acuerdo ni desacuerdo	Algo de acuerdo	Totalmente de acuerdo
1	2	3	4	5

8. ¿Se puede distinguir fácilmente, sólo con la mirada cuál es la información más importante?

Totalmente en desacuerdo	Algo en desacuerdo	Ni de acuerdo ni desacuerdo	Algo de acuerdo	Totalmente de acuerdo
1	2	3	4	5

9. ¿La información que se muestra es lo suficientemente descriptiva?

Totalmente en desacuerdo	Algo en desacuerdo	Ni de acuerdo ni desacuerdo	Algo de acuerdo	Totalmente de acuerdo
1	2	3	4	5

10. ¿Los títulos del menú y los nombres de su contenido le ayudaron a inferir que información se mostraría al acceder a ellos?

Totalmente en desacuerdo	Algo en desacuerdo	Ni de acuerdo ni desacuerdo	Algo de acuerdo	Totalmente de acuerdo
1	2	3	4	5

Navegación

11. ¿Se pueden ver las funciones principales mientras se navega en el resto de las pantallas?

Totalmente en desacuerdo	Algo en desacuerdo	Ni de acuerdo ni desacuerdo	Algo de acuerdo	Totalmente de acuerdo
1	2	3	4	5

12. ¿Existen en pantalla elementos suficientes que le permitan saber en qué parte de la aplicación se encuentra?

Totalmente en desacuerdo	Algo en desacuerdo	Ni de acuerdo ni desacuerdo	Algo de acuerdo	Totalmente de acuerdo
1	2	3	4	5

13. ¿Considera que es fácil volver a la pantalla de inicio sin importar la pantalla en la que se encuentre?

Totalmente en desacuerdo	Algo en desacuerdo	Ni de acuerdo ni desacuerdo	Algo de acuerdo	Totalmente de acuerdo
1	2	3	4	5

14. ¿Existen elementos que le ayudan a determinar dentro de qué métodos ya ha estado?

Totalmente en desacuerdo	Algo en desacuerdo	Ni de acuerdo ni desacuerdo	Algo de acuerdo	Totalmente de acuerdo
1	2	3	4	5

15. Si minimizó la ventana principal de la aplicación, ¿Considera que las barras de desplazamiento ayudan a visualizar de forma correcta el contenido de la ventana?

Totalmente en desacuerdo	Algo en desacuerdo	Ni de acuerdo ni desacuerdo	Algo de acuerdo	Totalmente de acuerdo
1	2	3	4	5

Diseño

16. ¿Le pareció adecuada la forma en la que se presentan los elementos dentro de las pantallas de la aplicación?

Totalmente en desacuerdo	Algo en desacuerdo	Ni de acuerdo ni desacuerdo	Algo de acuerdo	Totalmente de acuerdo
1	2	3	4	5

17. ¿Considera que las imágenes utilizadas son suficientes? ¿Representan de forma adecuada la información?

Totalmente en desacuerdo	Algo en desacuerdo	Ni de acuerdo ni desacuerdo	Algo de acuerdo	Totalmente de acuerdo
1	2	3	4	5

18. ¿Las gráficas de pastel dentro del módulo comparativo y los métodos se muestran de un tamaño adecuado?

Totalmente en desacuerdo	Algo en desacuerdo	Ni de acuerdo ni desacuerdo	Algo de acuerdo	Totalmente de acuerdo
1	2	3	4	5

19. ¿Cree que los colores utilizados en la aplicación son adecuados? ¿Son agradables a la vista?

Totalmente en desacuerdo	Algo en desacuerdo	Ni de acuerdo ni desacuerdo	Algo de acuerdo	Totalmente de acuerdo
1	2	3	4	5

20. ¿Considera que los botones utilizados son del tamaño correcto?

Totalmente en desacuerdo	Algo en desacuerdo	Ni de acuerdo ni desacuerdo	Algo de acuerdo	Totalmente de acuerdo
1	2	3	4	5

21. ¿Los iconos e imágenes utilizadas le ayudaron a identificar más rápidamente la función de los botones?

Totalmente en desacuerdo	Algo en desacuerdo	Ni de acuerdo ni desacuerdo	Algo de acuerdo	Totalmente de acuerdo
1	2	3	4	5

Aceptación

22. En general, ¿Está satisfecho con la aplicación?

Totalmente en desacuerdo	Algo en desacuerdo	Ni de acuerdo ni desacuerdo	Algo de acuerdo	Totalmente de acuerdo
1	2	3	4	5

23. ¿Está conforme con el tiempo empleado? ¿Cree que el uso de la aplicación reduce el tiempo del cálculo del diseño de mezclas?

Totalmente en desacuerdo	Algo en desacuerdo	Ni de acuerdo ni desacuerdo	Algo de acuerdo	Totalmente de acuerdo
1	2	3	4	5

24. ¿Cree que los resultados obtenidos son flexibles en cuanto a su modificación?

Totalmente en desacuerdo	Algo en desacuerdo	Ni de acuerdo ni desacuerdo	Algo de acuerdo	Totalmente de acuerdo
1	2	3	4	5

25. ¿Considera que las herramientas incluidas son de utilidad?

Totalmente en desacuerdo	Algo en desacuerdo	Ni de acuerdo ni desacuerdo	Algo de acuerdo	Totalmente de acuerdo
1	2	3	4	5

26. ¿Piensa que fueron incluidas la mayoría de herramientas matemáticas útiles en el cálculo de mezclas de concreto?

Totalmente en desacuerdo	Algo en desacuerdo	Ni de acuerdo ni desacuerdo	Algo de acuerdo	Totalmente de acuerdo
1	2	3	4	5

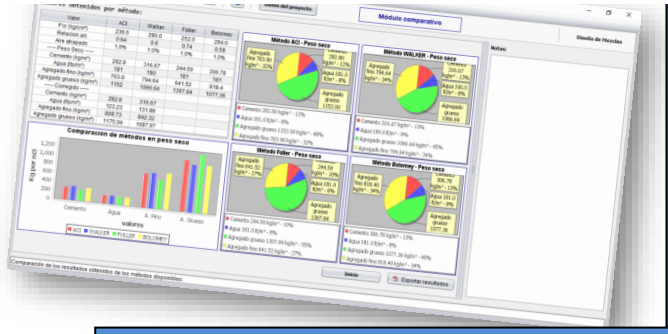
27. ¿Recomendaría el uso de la aplicación a otras personas?

Totalmente en desacuerdo	Algo en desacuerdo	Ni de acuerdo ni desacuerdo	Algo de acuerdo	Totalmente de acuerdo
1	2	3	4	5

Comentarios o sugerencias de mejora:

Anexo C

Manual de usuario



Concrete Pro Design

Manual de usuario

Alfredo Xochitemol Cruz

Tecnológico Nacional de México

Febrero 2018

Índice general

Índice de figuras

Capítulo 1

Instalación

1.1. Descripción de la aplicación

El diseño o proporcionamiento de mezclas de concreto es un proceso mediante el cual se determina la cantidad óptima de los materiales necesarios para la elaboración del concreto. La calidad del concreto resultante depende tanto de la calidad de los materiales como de la correcta combinación de estos, por lo que realizar un correcto proporcionamiento de los materiales es esencial para lograr un concreto de calidad.

El diseño de la mezcla para concreto normal y sin aditivos puede realizarse mediante métodos ampliamente conocidos como lo son el método del comité ACI, el método de Walker, el método de Füller y el método de Bolomey. Sin embargo, estos métodos son considerados extensos e incluyen una gran cantidad de operaciones matemáticas.

Esta aplicación permite realizar el diseño de mezclas de concreto mediante los métodos ACI, Walker, Bolomey y Füller. Además, permite comparar los resultados obtenidos en cada método para que el usuario pueda evaluar cuál es el que ofrece mejores resultados. Con esta aplicación se espera aumentar la fiabilidad de los cálculos, verificar resultados y ayudar con el aprendizaje de dichos métodos.

1.2. Instalación

1.2.1. Requisitos de instalación

- Sistema operativo: Windows 7 (32 y 64 bits), Windows Vista (32 bits y 64 bits), Windows XP (32 bits con SP2 o superior), Windows 8 (32 y 64 bits), Windows 8.1 (32 y 64 bits), Windows 10 (32 y 64 bits).
- Memoria RAM: 256 MB.
- Espacio libre en el disco duro: 240 MB.
- versión de Java mínima: 1.7.

1.2.2. Proceso de instalación

Para instalar la aplicación en el sistema operativo Windows, simplemente se debe de ejecutar el archivo de instalación con el nombre de *Setup_ConcreteProDesign_v1.0.exe* (véase figura 1.1).

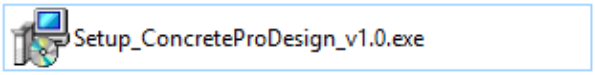


Figura 1.1: Archivo de instalación.

Una vez ejecutado el archivo de instalación se deberá de hacer clic en el botón siguiente hasta terminar la intalación. Adicionalmente, se instalarán automáticamente los complementos necesarios para la ejecución de la aplicación, en la figuras 1.2 y 1.3 puede observarse parte del proceso de instalación.

Una vez instalado se creará un acceso directo en el menú de inicio y en el escritorio del sistema operativo.

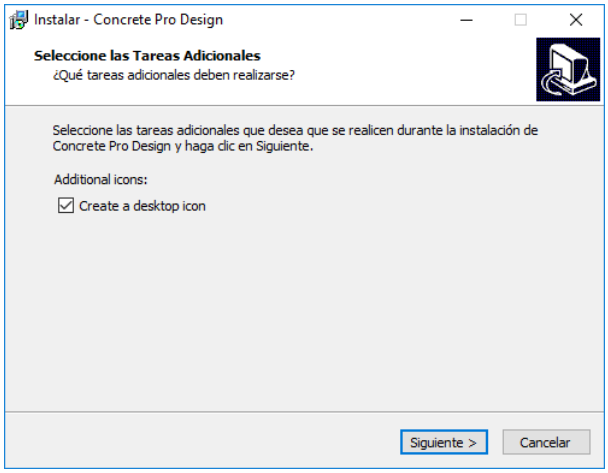


Figura 1.2: Instalación 1.

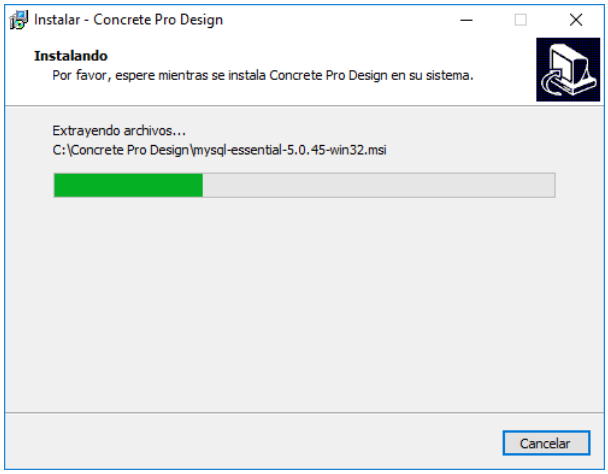


Figura 1.3: Instalación 2.

Capítulo 2

Uso de la aplicación

Se puede iniciar la aplicación de tres formas distintas:

1. Haciendo doble clic sobre el icono con nombre *Concrete Pro Desing* del escritorio del Sistema operativo.

2. Abriendo la aplicación desde el menú de inicio.

3. Ir a la ruta de instalación C:Concrete Pro Design y ejecutar el archivo Concrete Pro Design.exe.

2.1. Ingreso de los datos iniciales

Una vez abierta la aplicación se mostrará la pantalla de datos iniciales del último proyecto en el que se trabajó. Si es la primera vez que se inicia la aplicación los datos iniciales aparecerán vacíos. Estos datos son los que se utilizaran para el cálculo de los métodos de diseño de mezclas por lo que para poder avanzar es necesario ingresar todos los campos. Una vez ingresado todos los campos se puede proceder a calcular un método, seleccionando uno de los cuatro disponibles en la parte inferior de la pantalla. Si hubiera un campo que no se completo correctamente la aplicación mostrará una advertencia y resaltará en color rojo los campos con valores inválidos. En la figura 2.1 puede observarse la pantalla inicial de la aplicación.

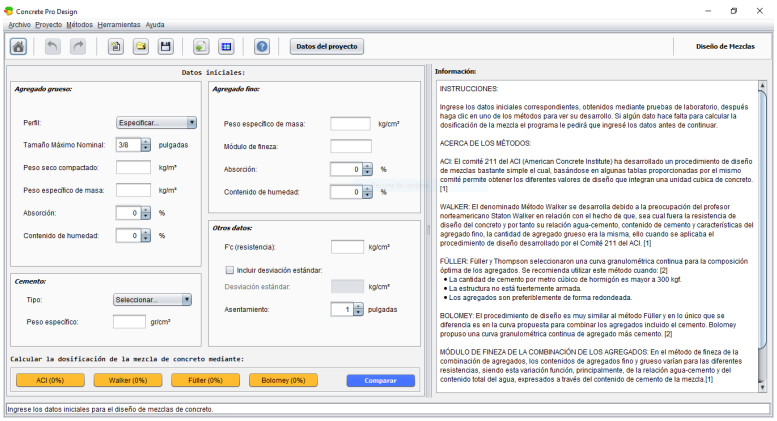


Figura 2.1: Pantalla de ingreso de los datos iniciales.

2.2. Cálculo de los métodos

2.2.1. Método ACI

Para acceder al cálculo del método ACI, es necesario dar clic en el botón *ACI*, dentro de la pantalla de datos iniciales o bien abriendo el método desde el menú principal de la aplicación. Para calcular el método ACI es necesario realizar 13 pasos y navegar a través de 7 pantallas que los contienen. En las pantallas del método puede observarse una barra de color verde en la parte inferior que indica el progreso del método. Cada paso realiza las operaciones en automático, pero en ocasiones es necesario a que el usuario seleccione un valor para poder continuar. En cada una de las pantallas pueden agregarse comentarios en la parte izquierda de la pantalla. En las imágenes 2.2, 2.3 y 2.4 pueden observarse ejemplos de las pantallas del método ACI.

2.2.2. Método Walker

Para acceder al cálculo del método Walker, es necesario dar clic en el botón *Walker*, dentro de la pantalla de datos iniciales o bien abriendo el método desde el

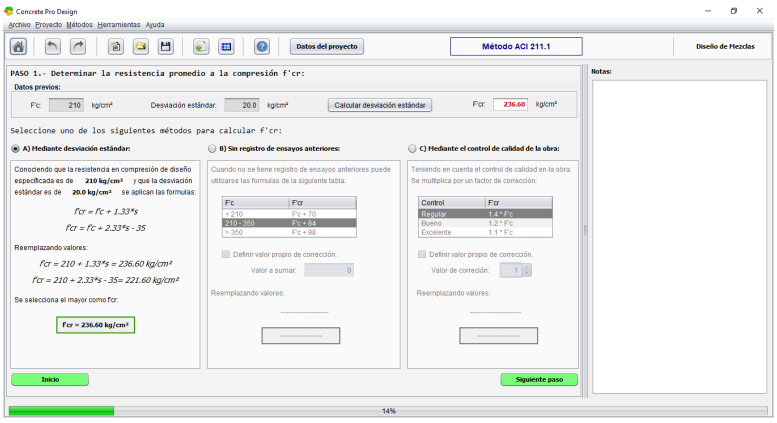


Figura 2.2: Pantalla de ejemplo 1 del método ACI.

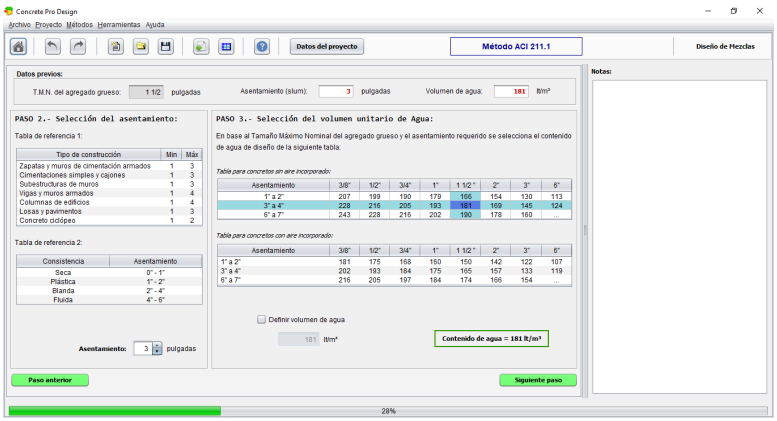


Figura 2.3: Pantalla de ejemplo 2 del método ACI.

menú principal de la aplicación. Para calcular el método Walker es necesario realizar 13 pasos y navegar a través de 8 pantallas que los contienen. En las pantallas del método puede observarse una barra de color verde en la parte inferior que indica el progreso del método. Cada paso realiza las operaciones en automático, pero en ocasiones es necesario a que el usuario seleccione un valor para poder continuar. En

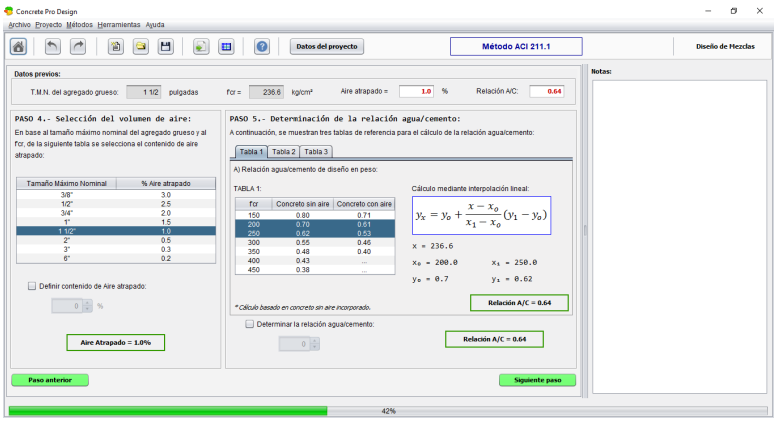


Figura 2.4: Pantalla de ejemplo 3 del método ACI.

cada una de las pantallas pueden agregarse comentarios en la parte izquierda de la pantalla. En las imágenes 2.5, 2.6 y 2.7 pueden observarse algunas de las pantallas pertenecientes al método Walker.

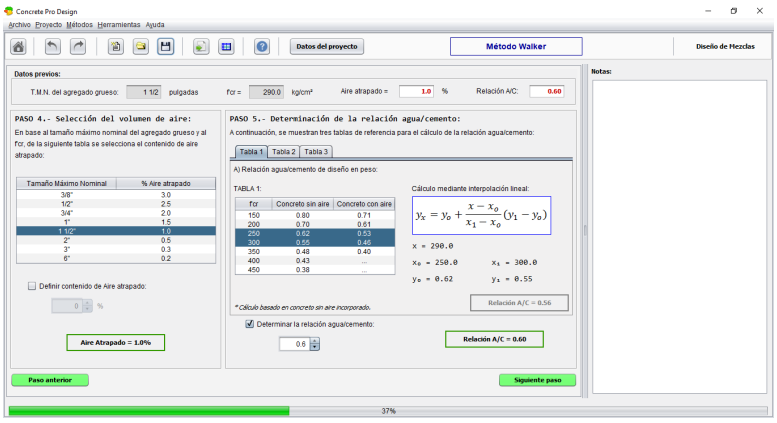


Figura 2.5: Pantalla de ejemplo 1 del método Walker.

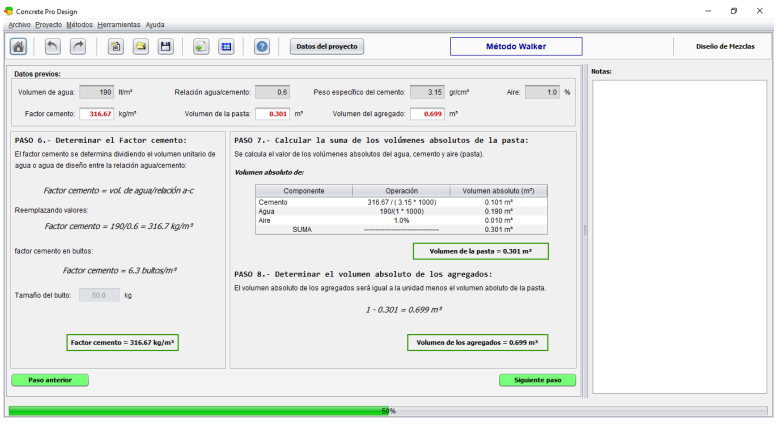


Figura 2.6: Pantalla de ejemplo 2 del método Walker.

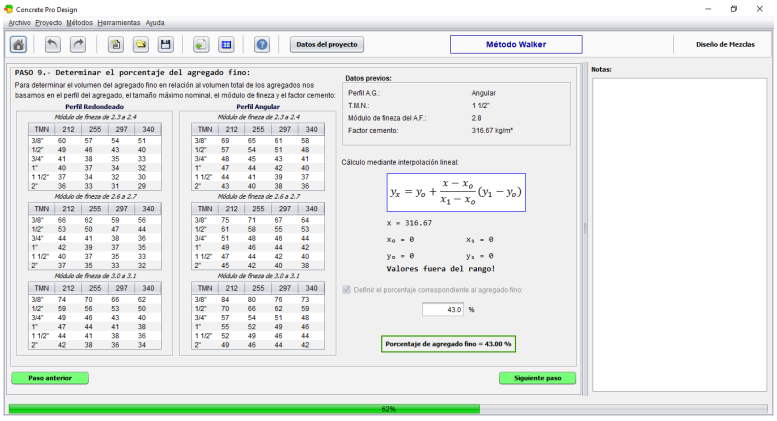


Figura 2.7: Pantalla de ejemplo 3 del método Walker.

2.2.3. Método Füller

Para acceder al cálculo del método Füller, es necesario dar clic en el botón *Füller*, dentro de la pantalla de datos iniciales o bien abriendo el método desde el menú principal de la aplicación. Para calcular el método Füller es necesario realizar 11

pasos y navegar a través de 6 pantallas que los contienen. En las pantallas del método puede observarse una barra de color verde en la parte inferior que indica el progreso del método. Cada paso realiza las operaciones en automático, pero en ocasiones es necesario a que el usuario seleccione un valor para poder continuar. En cada una de las pantallas pueden agregarse comentarios en la parte izquierda de la pantalla. En las imágenes 2.8, 2.9 y 2.10 pueden observarse algunas de las pantallas del método Füller.

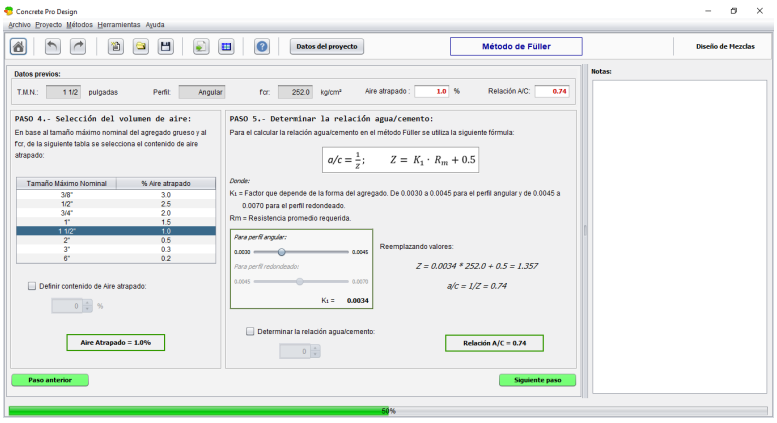


Figura 2.8: Pantalla de ejemplo 1 del método Füller.

2.2.4. Método Bolomey

Para acceder al cálculo del método Bolomey, es necesario dar clic en el botón *Bolomey*, dentro de la pantalla de datos iniciales o bien abriendo el método desde el menú principal de la aplicación. Para calcular el método Bolomey es necesario realizar 11 pasos y navegar a través de 6 pantallas que los contienen. En las pantallas del método puede observarse una barra de color verde en la parte inferior que indica el progreso del método. Cada paso realiza las operaciones en automático, pero en ocasiones es necesario a que el usuario seleccione un valor para poder continuar. En cada una de las pantallas pueden agregarse comentarios en la parte izquierda de la

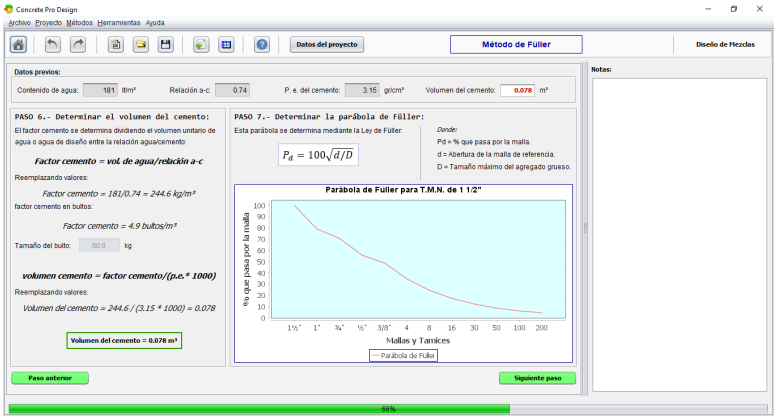


Figura 2.9: Pantalla de ejemplo 2 del método Füller.

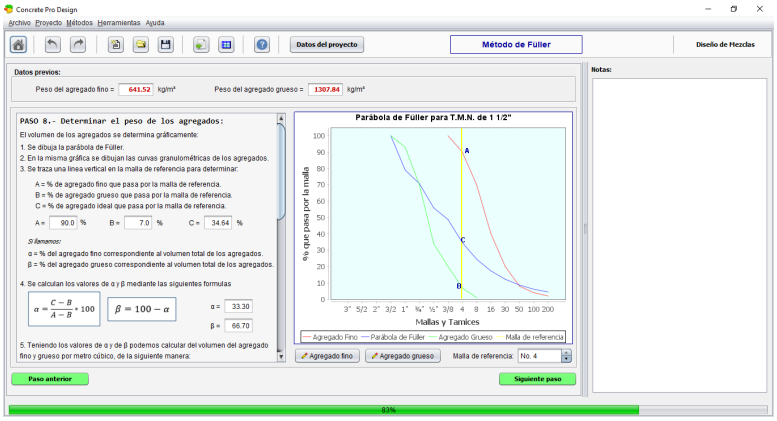


Figura 2.10: Pantalla de ejemplo 3 del método Füller.

pantalla. En las imágenes 25 a 30 pueden observarse todas las pantallas del método Bolomey.

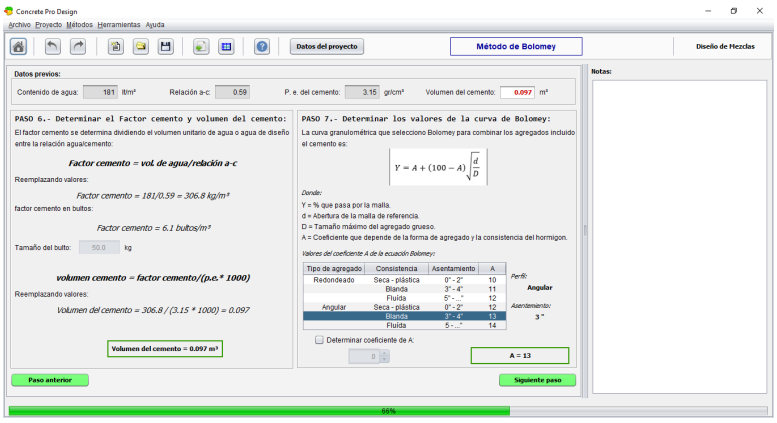


Figura 2.11: Pantalla de ejemplo 1 del método de Bolomey.

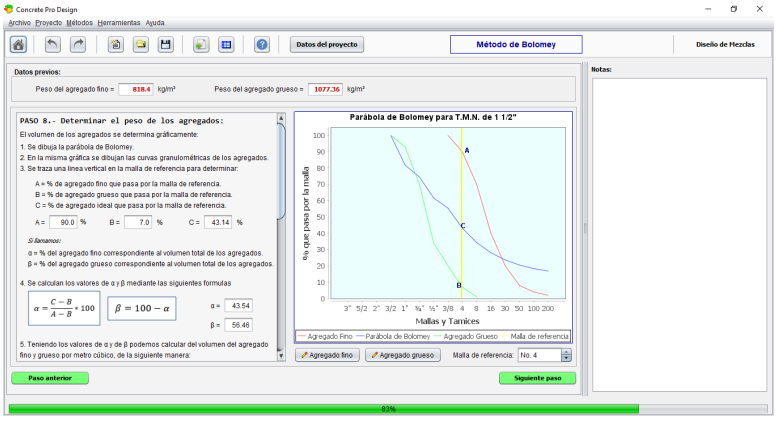


Figura 2.12: Pantalla de ejemplo 2 del método de Bolomey.

2.2.5. Método Módulo comparativo

El módulo comparativo permite comparar los resultados obtenidos en cada uno de los métodos calculados en una sola pantalla. Esto a través de tablas, gráficas de pastel y de barras. Para ingresar al módulo comparativo se debe hacer clic en *comparar* en

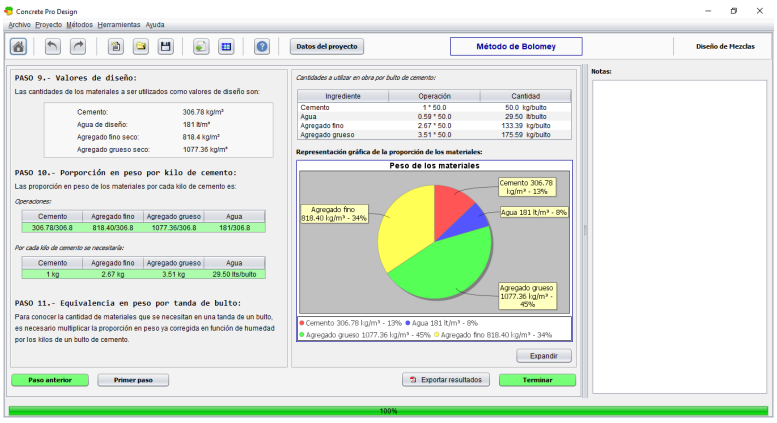


Figura 2.13: Pantalla de ejemplo 3 del método de Bolomey.

la pantalla inicial o bien a través del menú de la aplicación. En esta pantalla también se permite el ingreso de comentarios. En la figura 2.14, puede observarse el módulo comparativo.

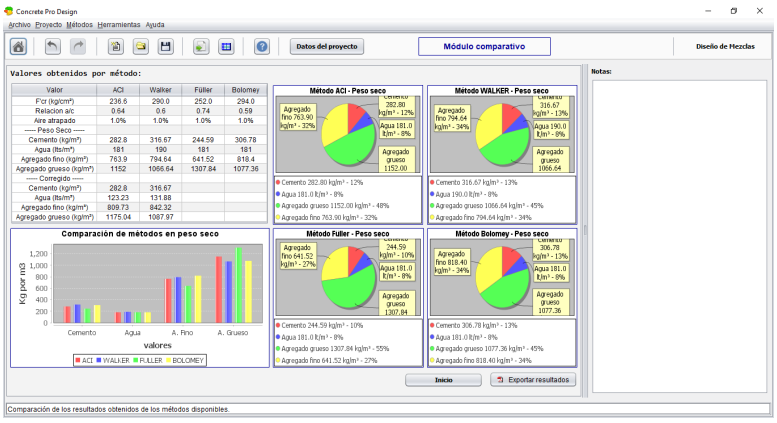


Figura 2.14: Módulo comparativo

2.3. Exportar resultados a formato PDF

Para exportar a formato PDF debemos ir a la pantalla final del método que queremos exportar. Una vez ahí, debemos hacer clic en el botón *Exportar resultados*, después se nos abrirá una pantalla para seleccionar la ubicación y el nombre en el que se guardará el archivo PDF (Véase figura 2.15). Después se abrirá en automático el documento PDF (véase figura 2.16).

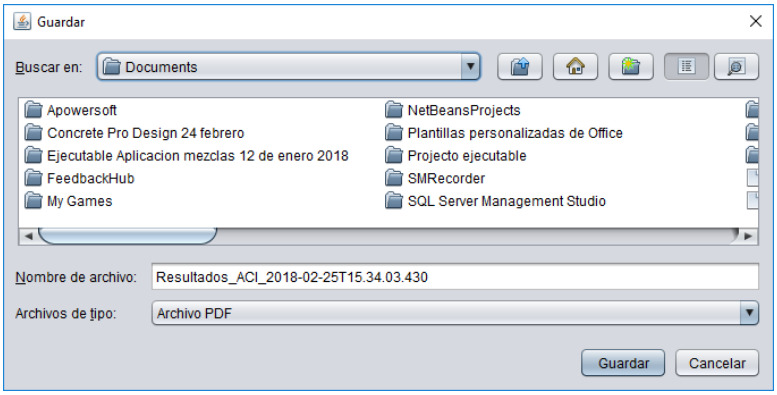


Figura 2.15: Cuadro de diálogo para guardar archivos.

2.4. Crear, abrir y guardar proyectos

Para abrir un proyecto debemos hacer clic sobre el icono *ABRIR* en la barra de tareas o bien hacer clic en *Abrir* desde el menú archivo dentro de la aplicación. Una vez abierto nos abrirá un cuadro para seleccionar archivos, para restaurar archivos correctamente debe ser un archivo con extensión .sql generado por la misma aplicación. Después de cargar el archivo se volverá a cargar la aplicación con los nuevos datos. Un ejemplo de un proyecto válido puede encontrarse en la ruta de instalación bajo el nombre: C:Concrete Pro Design/Proyecto de Ejemplo.sql

Para guardar un proyecto debemos hacer clic sobre el icono *GUARDAR* en la barra de tareas o bien hacer clic en *Guardar* desde el menú archivo dentro de la

aplicación. Una vez seleccionado nos abrirá un cuadro para seleccionar la ubicación del archivo, al dar clic en aceptar generará un archivo sql con el nombre dado el cual contendrá todos los datos del proyecto.

Para crear un proyecto debemos hacer clic sobre el icono *NUEVO* en la barra de tareas o bien hacer clic en *Nuevo proyecto* desde el menú archivo dentro de la aplicación. Una vez seleccionado se cargará nuevamente la aplicación con una plantilla vacía.

2.5. Funciones Matemáticas

La aplicación incluye funciones matemáticas comunes en el cálculo de mezclas de concreto. Para acceder directamente a ellas debemos abrir el menú Herramientas del menú principal. Las funciones que incluye son: Análisis granulométrico del agregado fino y grueso (véase figura 2.17), desviación estándar (véase figura 2.18) y cálculo de interpolación lineal (véase figura 2.19).

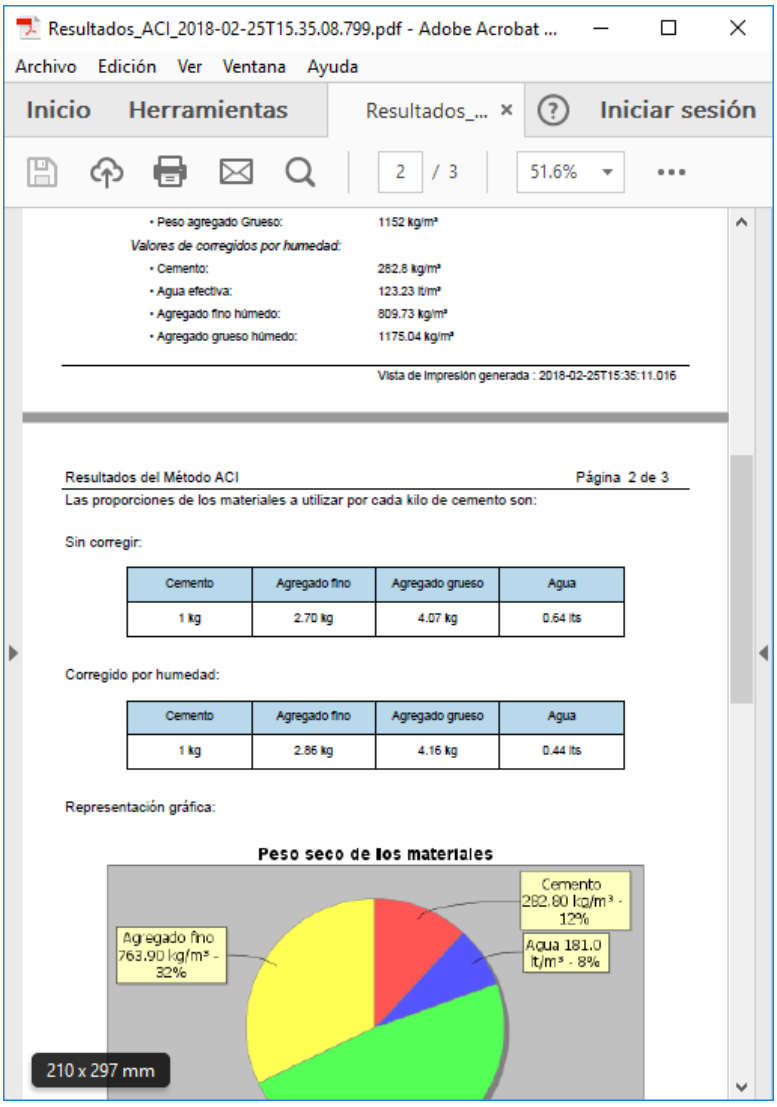


Figura 2.16: Archivo PDF de resultados.

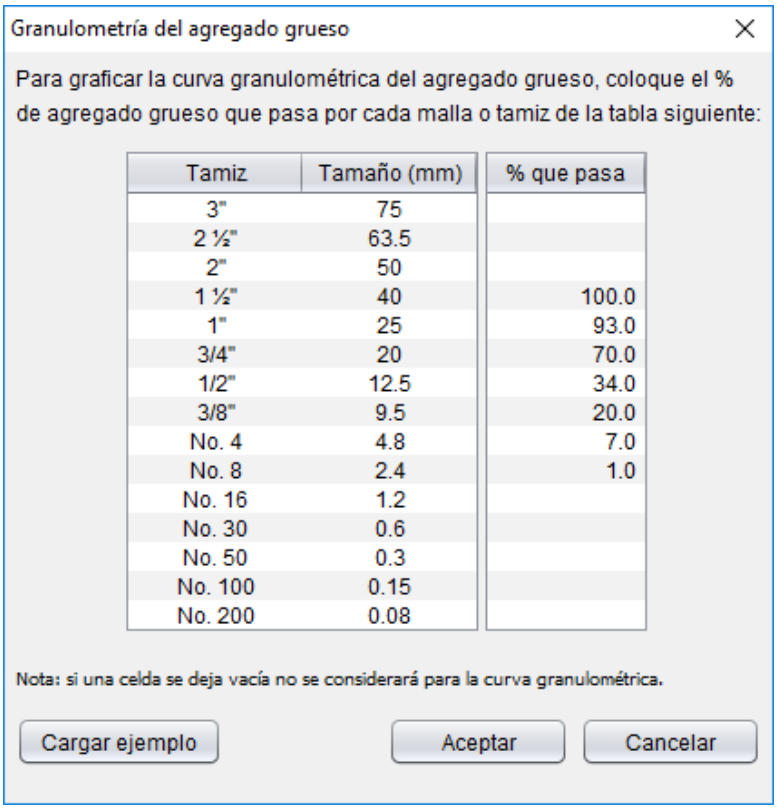


Figura 2.17: Herramienta para el análisis granulométrico.

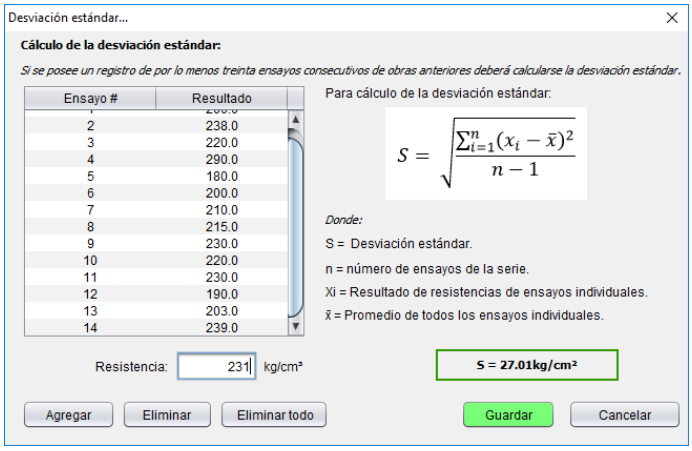


Figura 2.18: Herramienta para el cálculo de la desviación estándar.

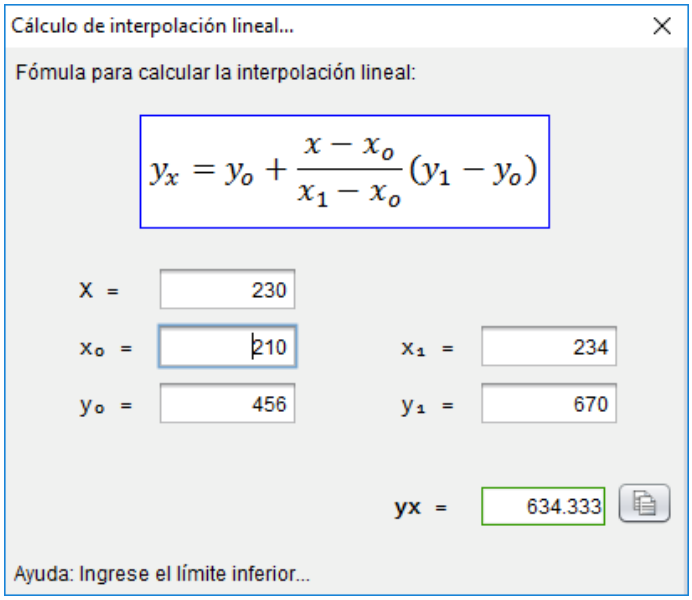


Figura 2.19: Herramienta para el cálculo de la interpolación lineal.

Anexo D

Publicaciones

SEP
SECRETARÍA DE EDUCACIÓN PÚBLICA

TECNOLÓGICO NACIONAL DE MÉXICO

ACADEMIA JOURNALS

CONGRESO INTERNACIONAL DE INVESTIGACIÓN DE ACADEMIA JOURNALS.COM, CELAYA 2017

OTORGAN EL PRESENTE

CERTIFICADO

A

ING. ALFREDO XOCHITEMOL CRUZ
M.C. MARÍA GUADALUPE MEDINA BARRERA
M.C. JUAN JOSÉ HERNÁNDEZ MORA
M.C. HIGINIO NAVA BAUTISTA
M.C. JUAN RAMOS RAMOS

POR SU PARTICIPACIÓN CON LA PONENCIA TITULADA

PROPUESTA METODOLÓGICA DE DESARROLLO ÁGIL PARA SOFTWARE ENFOCADA A EQUIPOS PEQUEÑOS

PUBLICADA EN EL PORTAL DE INTERNET
CELAYA.ACADEMIAJOURNALS.COM
VOLUMEN ONLINE CON ISSN 1946-5351 VOL. 9, NO. 6, 2017
E INDIZACIÓN EN FUENTE ACADÉMICA PLUS (EBSCO) Y
LIBRO DIGITAL EBOOK CON ISBN 978-1-939982-32-2, ONLINE.
LA CUAL FUE PRESENTADA EN EL
TECNOLÓGICO NACIONAL DE MÉXICO EN CELAYA
LOS DÍAS 8, 9 Y 10 DE NOVIEMBRE DE 2017, CELAYA, GUANAJUATO, MÉXICO.

DR. RAFAEL MORAS
EDITOR, ACADEMIAJOURNALS.COM
PROFESOR DE INGENIERÍA INDUSTRIAL Y ADMINISTRATIVA
ST. MARY´S UNIVERSITY, SAN ANTONIO, TX. EEUU

ING. ALEJANDRO ÁLVAREZ BARCENAS
COORDINADOR GENERAL DEL
CONGRESO INTERNACIONAL DE INVESTIGACIÓN
ACADEMIA JOURNALS, CELAYA 2017

N° 2577

Cel1030

Memorias del Congreso Internacional de Investigación Academia Journals Celaya 2017 Celaya, Guanajuato, México Noviembre 8, 9, y 10, 2017

PROPUESTA METODOLÓGICA DE DESARROLLO ÁGIL PARA SOFTWARE ENFOCADA A EQUIPOS PEQUEÑOS

Ing. Alfredo Xochitemol Cruz[1], M.C. María Guadalupe Medina Barrera[2], M.C. Juan José Hernández Mora[3], Mtro. Higinio Nava Bautista[4] y M.C. Juan Ramos Ramos[5]

***Resumen*—La mayoría de las metodologías tradicionales de desarrollo de software están dirigidas a grandes equipos de desarrollo y a largos periodos de tiempo. Cuando se trabaja con un equipo pequeño de desarrollo resulta difícil cumplir con todos procesos requeridos de dichas metodologías, debido a que se convierte en una tarea extenuante para el equipo desarrollador. En este artículo se presenta una metodología de desarrollo ágil que permite a equipos reducidos trabajar con proyectos de desarrollo de software enfocándose principalmente a la codificación, al análisis y al diseño, reduciendo la documentación. Esta metodología se basa en el modelo de desarrollo incremental y en la metodología de desarrollo *Extreme Programming*, así se logra mayor agilidad en el proceso de desarrollo, iteraciones cortas en tiempo y una rápida adaptación y aceptación a los cambios en los requisitos del proyecto.**

***Palabras clave*—Metodología de Desarrollo de Software, Ingeniería de Software, Desarrollo Ágil.**

Introducción

Avison y Fitzgerald definen una Metodología de Desarrollo de Software como una colección de procedimientos, técnicas, herramientas y documentos auxiliares que ayudan a los desarrolladores de software en sus esfuerzos por implementar nuevos sistemas de información[1]. Existen una gran cantidad de modelos y metodologías de desarrollo de software, cada una con diferentes características, ventajas y desventajas. Por lo que seleccionaj6r una adecuada Metodología de Desarrollo de Software (MDS) para nuestro proyecto es el primer gran paso hacia el éxito del mismo. Las MDS son indispensables para crear, o modificar software de calidad que cumpla con los requisitos de los usuarios, ya que, si no se utiliza la metodología apropiada, seguramente no se alcanzará el objetivo[2].

El problema es que la mayoría de estas metodologías se enfocan a grandes proyectos y a grandes equipos de desarrollo. Esto implica que, al aplicarse dichas metodologías a equipos pequeños, se presenten problemas principalmente de sobrecarga de trabajo lo que también deriva en el incumplimiento de los plazos establecidos. Esto ocurre porque muchas de estas metodologías incluyen una gran cantidad de documentación por cada etapa del proyecto. El principal objetivo de esta documentación es que otros miembros del equipo puedan adaptarse de forma más rápida a las diferentes etapas del proyecto. Sin embargo, al referirse a equipos pequeños de desarrollo en donde, debido a su reducido tamaño, todos los miembros están en constante comunicación, dicha documentación resulta un gran esfuerzo que podría evitarse. La propuesta metodológica que se presenta en este artículo, trata de reducir al mínimo la documentación empleada y de enfocarse principalmente a la codificación, análisis y diseño del proyecto.

Es importante mencionar y tener en cuenta el trabajo de Andrés Loboguerrero, Lorena Castañeda y Fernando Arboleda, quienes presentaron un trabajo llamado Metodología Ágil para Equipos Pequeños Usando Plataformas Microsoft[3], donde se presenta una metodología ágil utilizando *Microsoft Solutions Framework for Agile Software Development* (MSF4ASD) con los lineamientos de gestión de proyectos presentados en la Guía del PMBOK.

En la primera parte de este trabajo se presenta un panorama general de los modelos y metodologías de desarrollo actuales. La segunda parte describe la propuesta metodológica de desarrollo ágil enfocada a equipos reducidos, la cual está basada en una combinación de modelos y metodologías adaptándolas a las necesidades únicas que presenta un equipo pequeño de desarrollo. Además, se considera que se tiene una constante interacción con el usuario, para así agilizar el proceso de desarrollo y permitir iteraciones cortas en tiempo.

[1]Ing. Alfredo Xochitemol Cruz, alumno de Maestría en Sistemas Computacionales en el Instituto Tecnológico de Apizaco, Tlaxcala, México alfredoxcruz@gmail.com **(autor corresponsal)**
[2]M.C. María Guadalupe Medina Barrera es parte del núcleo académico de la Maestría en Sistemas Computacionales en el Instituto Tecnológico de Apizaco, Tlaxcala, México lupita_medina@hotmail.com
[3]M.C. Juan José Hernández Mora es parte del núcleo académico de la Maestría en Sistemas Computacionales en el Instituto Tecnológico de Apizaco, Tlaxcala, México jjhmora@itapizaco.edu.mx
[4]El Mtro. Higinio Nava Bautista es colaborador de la Maestría en Sistemas Computacionales en el Instituto Tecnológico de Apizaco, Tlaxcala, México higinionava@hotmail.com
[5]El M.C. Juan Ramos Ramos es colaborador de la Maestría en Sistemas Computacionales en el Instituto Tecnológico de Apizaco, Tlaxcala, México jramos2@hotmail.com

Memorias del Congreso Internacional de Investigación Academia Journals Celaya 2017 Celaya, Guanajuato, México Noviembre 8, 9, y 10, 2017

Modelos de Desarrollo

Los modelos de desarrollo también conocidos como metodologías pesadas o metodologías tradicionales son aquellas que se basan en una serie de fases secuenciales tales como: análisis, diseño, implementación y pruebas[4]. Por lo general estos modelos se caracterizan por ser poco flexibles y muy rígidos en cuanto a la documentación que se debe generar en cada etapa. Por lo que su implementación se vuelve una tarea muy ardua, en especial para equipos pequeños. A continuación, se describen brevemente algunos de los modelos de desarrollos más conocidos.

Modelo en cascada

La versión original del modelo en cascada, fue presentada por Royce en 1970, aunque son más conocidos los refinamientos realizados por Boehm en 1981, Sommerville en 1985 y Sigwart y col. en 1900[5]. A pesar de que este modelo fue presentado hace mucho tiempo es todavía utilizado en la actualidad. El modelo se consiste en una serie de fases secuenciales. Es decir que cada fase debe ser completada antes de avanzar a la siguiente y que no se puede avanzar sin que la fase previa esté realizada. Una vez que se ha dado por completada una fase, no se permite regresar a dicha fase. Las fases que componen el modelo en cascada son: requisitos, diseño, implementación, verificación y mantenimiento.

Modelo en espiral

El modelo en espiral fue propuesto por Barry Boehm en el año de 1986. Está basado en el modelo en cascada y en el de prototipos. Las fases del modelo en espiral son similares a las del modelo en cascada, pero con la diferencia de que el modelo en espiral se compone de ciclos o iteraciones. Donde cada ciclo representa una versión o prototipo del software, la cual es presentada al cliente hasta que éste se encuentra totalmente satisfecho con el producto. El modelo en espiral es especialmente útil cuando los usuarios no están completamente seguros de sus necesidades al inicio del proyecto.

Metodologías de Desarrollo Ágil

Las metodologías de desarrollo ágil surgen como una alternativa a las metodologías tradicionales las cuales se caracterizan por su poca flexibilidad y su rigurosa documentación. Este tipo de metodología se enfoca en la agilidad en el desarrollo de software. En esencia, agilidad significa responder a los cambios de forma rápida y eficiente[6].

Extreme Programming

La metodología *Extreme Programming* (XP) es la metodología de desarrollo ágil más ampliamente utilizada. Al igual que el resto de las metodologías ágiles XP comparte los valores del Manifiesto Ágil, pero va más allá al especificar un conjunto de prácticas simples[7]. Es una metodología centrada en potenciar las relaciones interpersonales como clave para el éxito en desarrollo de software, promoviendo el trabajo en equipo, preocupándose por el aprendizaje de los programadores y propiciando un buen clima de trabajo[8].

XP es una disciplina de desarrollo de software basada en valores de simplicidad, comunicación, retroalimentación y coraje[7]. Las características principales de XP pueden clasificarse en: historias de usuario, roles, procesos y prácticas[9].

Crystal

La metodología ágil se compone de un conjunto de MDS caracterizadas por estar centradas en las personas que componen el equipo de trabajo y la reducción al máximo del número de artefactos producidos[8]. Hay cuatro variantes de la metodología Crystal cada una aplica diferentes políticas identificada por un color y se clasifican dependiendo del número de integrantes del equipo: Crystal Clear para equipos de 8 o menos, Amarillo para equipos de 8 a 20, Naranja para equipos de 20 a 50 y Rojo para equipos de 50 o más.

Las metodologías Crystal cumplen con las siguientes características: entregas frecuentes, mejora reflexiva, comunicación osmótica, seguridad personal, enfoque, fácil acceso a usuarios expertos y un entorno técnico con pruebas automatizadas.

Propuesta Metodológica

A continuación, se describe la propuesta metodológica de desarrollo aplicable a proyectos que cuentan con un equipo pequeño de desarrollo. Esta metodología presenta un marco de trabajo de referencia para el desarrollo del proyecto.

Características del Proyecto

No siempre se ha de aplicar la misma metodología para todo tipo de proyectos, sino que es conveniente analizar las necesidades para determinar cuál es la más apropiada[10]. Al seleccionar la metodología de desarrollo se debe asegurar que las características del proyecto se adapten a la de la metodología a utilizar. A continuación, se describen algunas características con las que debe cumplir el proyecto, para que esta metodología obtenga mejores resultados:

- Equipo de desarrollo pequeño. Esta propuesta metodológica está pensada para equipos de una a cinco personas.

Memorias del Congreso Internacional de Investigación Academia Journals Celaya 2017 Celaya, Guanajuato, México Noviembre 8, 9, y 10, 2017

- Plazo de tiempo reducido. Cuando el proyecto tiene un tiempo de entrega reducido, se debe elegir una metodología que permita garantizar que el proyecto se termine en tiempo y forma sin dejar de lado la calidad del proyecto.
- Constante interacción con el usuario. Al permitir mayor interacción del usuario con el equipo desarrollador, se agiliza el proceso de desarrollo. Se considera como interacción a todas las reuniones que tiene el equipo desarrollador con el usuario.
- Requisitos volátiles. Cuando el usuario estará en contacto frecuente con el proceso de desarrollo, los requisitos de la aplicación también cambiarán de forma constante.
- Tamaño del proyecto. Se recomienda que para poder aplicar esta metodología el tamaño del proyecto vaya de pequeño a mediano, ya que para grandes proyectos es probable que debido al reducido equipo de desarrollo se vuelva una tarea demasiado extenuante.

Descripción de la metodología propuesta

La metodología de desarrollo propuesta se basa en el modelo de desarrollo incremental y en la metodología de desarrollo ágil XP. De esta forma se logra crear una metodología que permite desarrollar una solución a la medida del usuario, permite iteraciones cortas en tiempo, agilidad en el proceso de desarrollo y una rápida adaptación y aceptación a los cambios en los requisitos del proyecto.

De la metodología XP se retoman aspectos que permiten lograr una retroalimentación continua entre el usuario y el equipo de desarrollo. Mientras que el modelo de desarrollo incremental se retoman conceptos para poder crear y agregar módulos al proyecto como se vayan desarrollando.

Características de la metodología propuesta

Las características de la metodología se dividen en: historias de usuarios, bitácoras, roles y procesos. Estas características fueron retomadas de la metodología XP, por considerarse ampliamente aplicables a equipos pequeños. A continuación, se describe cada tipo de características y sus componentes.

Historias de usuario

Las historias de usuario hacen referencia a los datos obtenidos tras aplicar técnicas utilizadas para la obtención de los requisitos, sin importar que estos sean funcionales o no funcionales. Cada historia de usuario debe ser lo suficientemente entendible y lo suficientemente delimitada para que su implementación no tome demasiado tiempo.

Por otra parte, cada historia de usuario identificada debe ir redactada en un documento, para el desarrollo de esta metodología se sugiere que dicho documento contenga los siguientes elementos: fecha, tipo de actividad (nueva, corrección, mejora), prueba funcional, número de historia, prioridad técnica y del cliente, referencia a otra historia previa, riesgo, estimación técnica, descripción, notas y una lista de seguimiento con la fecha, estado, cosas por terminar y comentarios[7].

Sumado a estos elementos pueden agregarse otros como: nombre de la historia de usuario, dependencias relacionadas (es decir, aquellas historias de usuario que deben de completarse antes de realizarse), responsables de la actividad y número de iteración. Sin embargo, estos elementos pueden variar conforme sea determinado por el desarrollador. En la figura 1, se presenta un ejemplo de formato para una historia de usuario.

Las técnicas de obtención de requisitos son variadas y no se limita al uso exclusivo de unas cuantas. Dependiendo del tipo de información que se desea obtener es como se decide con qué técnica trabajar. Algunas técnicas son: entrevistas, cuestionarios y reuniones informales. Siendo siempre el equipo de desarrollo el que decida con cual trabajar. Tras finalizar la aplicación de cada técnica se debe redactar un documento por cada historia de usuario identificada, además de una bitácora de proyecto para llevar el registro de la aplicación de cada técnica.

Bitácoras

Las bitácoras de proyecto son documentos utilizado para llevar un registro de todo los requisitos y cambios. Las bitácoras pueden incluir los siguientes elementos; número de bitácora, fecha, tipo de técnica aplicada, puntos tratados, acuerdos a los que se llegaron y firma de los participantes. En la figura 2, se puede observar un ejemplo de bitácora con la estructura propuesta.

Una buena práctica es elaborar bitácoras de desarrollo. El objetivo de dichas bitácoras es llevar un registro de todos los cambios y avances realizados al proyecto. Las bitácoras de desarrollo pueden incluir los siguientes campos: número de bitácora, fecha de realización, cambio realizado, responsables y firma de los responsables.

Roles

A continuación, se recomiendan algunas funciones y roles que deberán ser asignadas entre los miembros del equipo. Pueden agregarse o eliminarse funciones y roles según lo considere necesario el equipo de desarrollo.

- Analista. Como su nombre lo indica es quien analiza el problema del cliente y establece una solución que asegure que se cubran todas las necesidades del cliente. El analista se reúne con el cliente y especifica los requisitos funcionales del proyecto.

Memorias del Congreso Internacional de Investigación Academia Journals Celaya 2017 © Academia Journals 2017 Celaya, Guanajuato, México Noviembre 8, 9, y 10, 2017

HISTORIA DE USUARIO			
Nombre: Desarrollar el prototipo de la interfaz de la aplicación		10-abr-16	**Núm.:** 5
Tipo de historia: ☑ Nueva ☐ Corrección ☐ Mejora			
Prioridad técnica: Media	**Prioridad de usuario:** Media		**Estimación (días):** 15
Historia previa núm.: 4	**Dependencias:** 2 y 3		**Riesgo:** Bajo
Descripción: Desarrollar un prototipo navegable de la interfaz de la aplicación.			
Notas: El prototipado de la aplicación se desarrollará a través de los bocetos diseñados en la historia anterior.			
Pruebas de funcionalidad: Las pruebas de funcionalidad se llevarán a cabo mediante los casos de uso identificados.			

Figura 1. Ejemplo del formato propuesto para una historia de usuario.

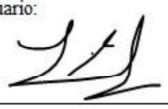
BITÁCORA		
Núm. de bitácora: 1	**Fecha:** 11-may-16	**Tipo de técnica:** Entrevista
Puntos a tratar: 1.- Necesidad de la aplicación. 2.- Requisitos de la aplicación. 3.- Definir los métodos de diseño de mezclas a utilizar. 4.- Tipos de usuarios. 5.- Documentación necesaria.		
Acuerdos: 1.- La aplicación debe calcular en automático los valores, mediante los valores de 2.- La aplicación debe calcular rápidamente y de forma fiable los resultados. 3.- Se debe mostrar el cálculo de los métodos paso a paso, para que el usuario pueda modificar un valor sin importar la parte del método donde se encuentre. 4.- Los Métodos de diseño de mezclas que se utilizarán son: ACI 211, Norma ASTM, Bolomey, Walker y Füller. 5.- Se incluirá una tabla comparativa de los 5 métodos utilizados. De donde el experto decidirá cuál es el mejor.		
Firmas		
Responsable: 	Usuario: 	

Figura 2. Ejemplo del formato propuesto para una bitácora de reuniones.

- Arquitecto de software. Se encarga de traducir las soluciones hechas por el analista en algoritmos, diagramas y bocetos. El objetivo de esto es permitir al desarrollador codificar los módulos del proyecto de una manera más rápida y sencilla.

Memorias del Congreso Internacional de Investigación Academia Journals Celaya 2017 © Academia Journals 2017 Celaya, Guanajuato, México Noviembre 8, 9, y 10, 2017

- Desarrollador. Es el encargado de escribir el código que permite dar solución a las ideas plasmadas por el arquitecto de software.
- Diseñador. Es quien diseña la interfaz de usuario de la aplicación. El diseño se lleva a cabo mediante prototipos y bocetos. El desarrollador es el encargado de implementar estos bocetos, por esta razón, es importante que tanto el diseñador como el desarrollador trabajen juntos. De igual forma se debe trabajar en conjunto con el usuario, pues es este quien da el visto bueno de la interfaz.
- Encargado de pruebas (*tester*). Es el encargado de realizar las pruebas al sistema para verificar que se cumplan los requisitos del sistema, además verifica que no haya errores en el sistema. Es altamente recomendable que el encargado de las pruebas sea una persona diferente al o los desarrolladores, debido a que puede ver el sistema desde una diferente perspectiva lo que le permitirá pensar en escenarios en los que probablemente el desarrollador no pensó.
- El usuario. Es quién usará el sistema, por lo que, determina los requisitos funcionales del sistema. Además, de que realiza en conjunto al *tester* las pruebas y da la aprobación de que una historia de usuario ha sido completada con éxito.

Proceso

Se conoce como proceso, al ciclo de desarrollo de la metodología propuesta. A continuación, se describe cada paso de dicha metodología. En la figura 3 se puede observar el ciclo completo de desarrollo de la metodología.

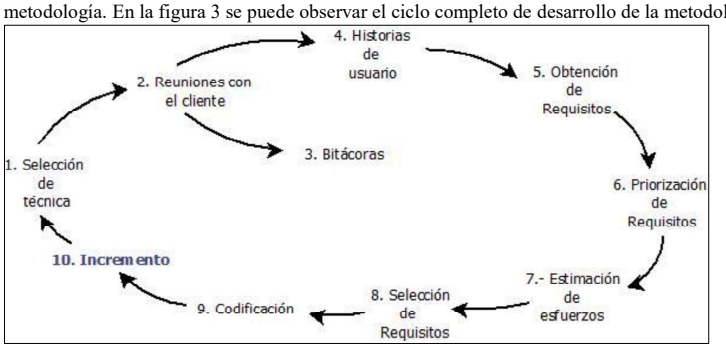


Figura 3. Ciclo de desarrollo de la metodología propuesta. Diagrama de elaboración propia.

1. Selección de técnica. El ciclo de desarrollo comienza con la selección de una o varias técnicas de obtención de datos: entrevistas, cuestionarios, etc. Es el analista quien determina qué técnica debe aplicarse en base al tipo de información que se desea obtener. También es el analista quien ejecuta la técnica de obtención de información.
2. Reuniones con el cliente. En las reuniones con el usuario es donde se ejecutan las técnicas de obtención de datos.
3. Bitácoras. El analista elabora las bitácoras para llevar un registro de reuniones con cliente.
4. Historias de usuario. Tras las reuniones con el cliente el analista determina las historias de usuario. Sirven para documentar las modificaciones o identificar nuevos requisitos de la aplicación.
5. Obtención de requisitos. De las historias de usuario se identifican los nuevos requisitos solicitados por el cliente.
6. Priorización de requisitos. El analista junto con el cliente, determinan cuál de los requisitos pendientes es más importante y priorizan los requisitos según su nivel de necesidad.
7. Estimación de esfuerzos. El arquitecto de software y el desarrollador estiman el tiempo y esfuerzo necesarios para cubrir cada uno de los requisitos pendientes.
8. Selección de requisitos. En base a la estimación de esfuerzos y a la prioridad de los requisitos, el analista determina que requisitos pueden ser cumplidos y presentados antes de la próxima reunión con el cliente. Una vez determinados, el arquitecto y el diseñador de software plasman una solución a dichos requisitos.
9. Codificación. El desarrollador codifica la solución plasmada por el arquitecto de software. La codificación de los requisitos se realiza en módulos independientes a la aplicación principal. De esta forma no se afecta la funcionalidad de la aplicación mientras se desarrollan.
10. Incremento. El tester aplica las pruebas necesarias a los módulos por integrar. Una vez completadas las pruebas y corregidos los errores, el desarrollador se encarga de integrar los nuevos módulos, mediante un

incremento a la aplicación principal. Entonces se dice que se ha completado una iteración del ciclo de desarrollo.

El ciclo de desarrollo debe de repetirse hasta que se cumplan con todos los requisitos funcionales y no funcionales del proyecto.

Conclusiones

El uso de la metodología propuesta permite que un pequeño equipo de desarrollo pueda trabajar con un proyecto de manera rápida mediante pequeñas iteraciones. Esta metodología, al igual que las metodologías de desarrollo ágil hace énfasis en la codificación y deja de lado la documentación exhaustiva como en las metodologías tradicionales. Además de que permite que el equipo pueda adaptarse rápidamente a los cambios en los requisitos generados por el usuario. En esta metodología se presenta una guía que sirve como marco de referencia para que pequeños equipos de desarrollo puedan adaptarse a proyectos donde se está con una constante interacción con el cliente. Se proponen roles, funciones y actividades que ayudan al equipo a gestionar y encaminar el proyecto hacia el éxito.

Referencias

[1]Avison, D. y Fitzgerald, G. "Information Systems Development: Methodologies, Techniques, and Tools". *McGraw-Hill*, 1995.

[2]Rivas, C. I., Corona, V. P., Gutiérrez, J. F. y Hernández, L. "Metodologías Actuales de Desarrollo de Software". *Revista de Tecnología e Innovación*, 2015, 980-986.

[3]Loboguerrero, A.F., Castañeda, L., y Arboleda, H. "Metodología Ágil para equipos pequeños usando plataformas Microsoft". *Revista S&T*, 83-99.

[4]Ben-Zahia, Mohamed A. y Jaluta, Ibrahim. "Criteria for Selecting Software Development Models". *Computer & Information Technology (GSCIT)*, 2014, 1-6.

[5]Cataldi, Z., Lage, F., Pessacq, R. y García Martínez, R. "Ingeniería de software educativo". *Proceedings del V Congreso Internacional de Ingeniería Informática,* 1999, 185-199.

[6]Jameel Qureshi y Rizwan, M. "Agile Software Development Methodology for Medium and Large Projects". *IET Software*, 2012, 358-363.

[7]Orjuela Duarte, A. y Rojas C., M. "Las Metodologías de Desarrollo Ágil como una Oportunidad para la Ingeniería del Software Educativo". *Revista Avances en Sistemas e Informática*, 2008, 156-171.

[8]Awad, M. A. "A comparison between Agile and Traditional Software Development Methodologies". *The University of Western Australia*, 2005.

[9]Lindstrom, L. y, Jeffries, R. "Extreme Programming and Agile Software Development Methodologies. Information Systems Management", 2004, 41-52.

[10]Beck, K. "Extreme Programming Explained. Embrace Change", *Pearson Education*, 1999.

ACADEMIA JOURNALS
OPUS PRO SCIENTIA ET STUDIUM

CEPES Panamericano
CENTRO PANAMERICANO DE ESTUDIOS SUPERIORES

0124

CONGRESO INTERNACIONAL DE INVESTIGACIÓN DE ACADEMIA JOURNALS MORELIA 2018
ACADEMIAJOURNALS.COM

CERTIFICADO

OTORGADO A

ING. ALFREDO XOCHITEMOL CRUZ
M.C. MARÍA GUADALUPE MEDINA BARRERA
M.C. JUAN JOSÉ HERNÁNDEZ MORA
ING. MIGUEL ÁNGEL DAZA MERINO

POR SU ARTÍCULO INTITULADO

DESARROLLO DE UNA APLICACIÓN PARA EL DISEÑO Y EVALUACIÓN DE MEZCLAS DE CONCRETO

EL CUAL FUE PRESENTADO EN EL CONGRESO DESARROLLADO DEL 16 AL 18 DE MAYO DE 2018 EN MORELIA, MICHOACÁN, MÉXICO Y PUBLICADO EN EL PORTAL DE INTERNET *ACADEMIAJOURNALS.COM*, CON ISSN 1946-5351 ONLINE, VOL. 10#3, 2018 Y EN EL LIBRO ELECTRÓNICO ONLINE TITULADO *COMPENDIO DE INVESTIGACIÓN MORELIA 2018* CON ISBN 978-1-939982-36-0

RAFAEL MORAS, PH.D. P.E
EDITOR, ACADEMIA JOURNALS

RECTOR ALDO EMILIO TELLO CARRILLO
CENTRO PANAMERICANO DE ESTUDIOS SUPERIORES

Artículo More140

Instituto de Ciencia, Tecnología e Innovación
Gobierno del Estado de Michoacán
2015 - 2021

Universidad Nova Spania

Memorias del Congreso Internacional
de Investigación Academia Journals
Morelia 2018

Morelia, Michoacán, México
16 al 18 de mayo de 2018

DESARROLLO DE UNA APLICACIÓN PARA EL DISEÑO Y EVALUACIÓN DE MEZCLAS DE CONCRETO

Ing. Alfredo Xochitemol Cruz[1], M.C. María Guadalupe Medina Barrera[2],
M.C. Juan José Hernández Mora[3] e Ing. Miguel Ángel Daza Merino [4]

***Resumen*— El diseño o proporcionamiento de mezclas de concreto es un proceso mediante el cual se determina la cantidad óptima de los materiales necesarios para la elaboración del concreto. El proporcionamiento de los materiales se logra mediante métodos de diseño de mezclas. Sin embargo, a menudo este tipo de métodos son extensos e incluyen una gran cantidad de operaciones matemáticas, por lo que resulta difícil elaborar más de uno para una misma práctica. En este artículo se describen los métodos, herramientas y procedimientos utilizados para el desarrollo de una aplicación de escritorio enfocada al diseño y evaluación de mezclas de concreto. La aplicación desarrollada permite calcular rápidamente el proporcionamiento de mezclas mediante los métodos ACI, Walker, Füller y Bolomey. A diferencia de los programas actuales, esta aplicación permite comparar los resultados obtenidos de cada método dentro de un módulo comparativo con el fin de evaluar cual ofrece mejores resultados.**

***Palabras clave*— Desarrollo de Software, Metodologías Ágiles, Diseño de Mezclas de concreto.**

Introducción

El concreto es un material para construcción ampliamente utilizado, debido a su resistencia, durabilidad y maleabilidad. Se compone básicamente de la mezcla de dos componentes: los agregados y la pasta. Los agregados son comúnmente arena (agregado fino) y grava (agregado grueso), mientras que la pasta se compone de agua, cemento y otros aditivos. La calidad del concreto depende tanto de la calidad de los componentes, como de la correcta combinación o mezcla de estos (Kosmatka, Kerkhoff, Panarese y Tanesi, 2004). El proporcionamiento de los materiales se logra mediante métodos de diseño de mezclas. Generalmente, estos métodos se basan en funciones fundamentales como la relación agua-cemento, la demanda de agua y la teoría del óptimo proporcionamiento de los agregados, de las cuales se determinan mezclas con las propiedades requeridas (Sobolev, 2003).

Llevar a cabo de forma manual los métodos de diseño de mezclas implica un largo e inclusive tedioso procedimiento debido a la gran cantidad de operaciones que conlleva cada uno de estos. Además, es probable que ocurran errores humanos, lo que provoca que se reduzca la fiabilidad de los datos. Este artículo describe el proceso de desarrollo de una aplicación de cómputo para realizar el diseño de mezclas de concreto mediante cuatro métodos distintos.

La aplicación está desarrollada bajo el lenguaje de programación Java. Java es un lenguaje multiplataforma por lo que puede ser ejecutado desde distintos sistemas operativos tales como: Windows, Mac OS y Linux, gracias a la máquina virtual de Java (en inglés *Java Virtual Machine,* JVM). Además, cuenta con conexión a MySQL para almacenar datos y gestionar los proyectos. La aplicación es capaz de realizar cuatro de los métodos de diseño de mezclas más populares, los cuales son: método ACI, método Walker, Método Füller y método Bolomey. Asimismo, incluye un módulo comparativo con el objetivo de que el usuario pueda comparar y evaluar los resultados obtenidos por los distintos métodos de diseño de mezclas, esto mediante tablas y gráficas. Adicionalmente, como cada uno de los métodos está descrito paso a paso también puede ser utilizado como apoyo al aprendizaje de cada uno de los métodos. El resultado individual de cada método, así como el módulo comparativo puede ser exportado a formato PDF.

Soluciones Actuales

Actualmente existen varios programas de cómputo diseñados para el proporcionamiento de mezclas de concreto, pero la mayoría solo trabaja con uno o dos métodos de diseño o no son completamente enfocados al diseño de mezclas de concreto. Además, de que son muy rígidos en cuanto a su funcionamiento y no permiten modificar los

[1]Ing. Alfredo Xochitemol Cruz, alumno de Maestría en Sistemas Computacionales en el Instituto Tecnológico de Apizaco, Tlaxcala, México alfredoxcruz@gmail.com **(autor corresponsal)**
[2]M.C. María Guadalupe Medina Barrera es parte del núcleo académico de la Maestría en Sistemas Computacionales en el Instituto Tecnológico de Apizaco, Tlaxcala, México lupita_medina@hotmail.com
[3]M.C. Juan José Hernández Mora es parte del núcleo académico de la Maestría en Sistemas Computacionales en el Instituto Tecnológico de Apizaco, Tlaxcala, México jjhmora@itapizaco.edu.mx
[4]El Ing. Miguel Ángel Daza Merino es Docente de Ingeniería Civil en el Instituto Tecnológico de Apizaco, Tlaxcala, México.

Memorias del Congreso Internacional
de Investigación Academia Journals
Morelia 2018

Morelia, Michoacán, México
16 al 18 de mayo de 2018

valores obtenidos por cada paso del método, lo cual implica que en ocasiones el usuario no obtenga el resultado deseado. A continuación, se describen brevemente las características de los principales programas encontrados.

Dimezco 2000 es un programa que permite diseñar mezclas de agregados, concretos y morteros, además de realizar evaluaciones estadísticas de los resultados (Dimezco 2000, 2013).

Bar-Dos2 es un programa, desarrollado por Francisco Javier Bardisa Mollá, miembro del ICITECH (Instituto de Ciencia y Tecnología del Hormigón) en España. El software se encuentra de forma gratuita y realiza el diseño de mezclas bajo el método del comité ACI 211.1.

Conmixer v1.0 es otro programa enfocado al diseño de mezclas de concretos, utiliza tres métodos IS (*Indian Standard*, es decir, el Estándar Hindú), ACI y DOE (*British Department of the Environment Method*, es decir, método británico del departamento de medioambiente), se encuentra forma gratuita y no necesita instalación (Rojas, 2011).

DM-CONCRET es un programa desarrollado en 2014 por Danilo Saavedra de la Universidad Tecnológica de los Andes de Perú. Realiza el diseño de mezclas mediante el método ACI 211.1. En el cuadro 1, se puede observar una comparativa de las características de los programas analizados.

Software	Gratis	Versión completa	Métodos	Notas
Dimezco 2000 v8	Sí	No	ACI 211, Walker y Vitervo O'reilly	Última versión lanzada en Julio 2017
DM Concret v1.0	Sí	No	ACI 211	Necesita Microsoft Office
Bar-Dos	Sí	Sí	ACI 211	
Comixer v1.0	Sí	Sí	IS, ACI, DOE	Está enfocado a normas europeas

Cuadro 1. Comparativa de programas de Diseño de Mezclas de Concreto.

En base al análisis realizado se considera viable desarrollar una aplicación que permita trabajar con los cuatro métodos anteriormente mencionados (ACI, Walker, Füller y Bolomey), además de mostrar el procedimiento de cada uno e incluir un módulo comparativo de los datos obtenidos en cada método.

Herramientas de desarrollo

Java

A Java es un lenguaje de programación multiplataforma orientado a objetos que ofrece una arquitectura versátil, manejo de multihilos, es seguro y con un alto desempeño (Perea, Fernández, Arrollo, Rodríguez, Potayo, Montesinos). La tecnología Java se usa para desarrollar aplicaciones para un amplio alcance de entornos, desde dispositivos del consumidor hasta sistemas empresariales heterogéneos (Perry, 2012).

MySQL

MySQL es un sistema gestor de base de datos de código abierto (en inglés, *Database Management System*, DBMS) que trabaja bajo Windows y muchas versiones de UNIX. Puede ser distribuido de forma libre bajo una Licencia Pública General (en inglés, *General Public License*, GPL) siempre que distribuidor ponga a distribución el código fuente del programa junto con las versiones binarias (Harrington, 2003). MySQL es la plataforma de base de datos de fuente abierta más confiable actualmente en uso. Muchos de los sitios web más populares y con mayor tráfico en el mundo se basan en MySQL debido a si ubicuidad en plataformas heterogéneas y pilas de aplicaciones, sumado a su conocido rendimiento, fiabilidad y facilidad de uso (Satoto, Isnanto, Kridalukmana y Martono, 2016).

GUI Design Studio

GUI Design Studio es una herramienta desarrollada por Caretta Software para la creación de prototipos y diseño de interfaces de usuario. Es una aplicación del tipo *drag and drop*, es decir que permite seleccionar y arrastrar elementos sin la necesidad de escribir código, permite desarrollar prototipos de aplicaciones web, de escritorio, móviles y embebidas. Además, cuenta con una versión de prueba con todas sus funcionalidades durante 30 días.

Memorias del Congreso Internacional
de Investigación Academia Journals
Morelia 2018

Morelia, Michoacán, México
16 al 18 de mayo de 2018

Metodología

Descripción de la Metodología

En programación, una metodología corresponde a un conjunto de métodos utilizados para alcanzar los objetivos planteados (Ojeda-Guerra, 2015). No siempre se ha de aplicar la misma metodología para todo tipo de proyectos, sino que es conveniente analizar las necesidades para determinar cuál es la más apropiada (Alfonso, Mariño y Godoy, 2011). La metodología utilizada fue una de diseño propio (Xochitemol, Medina, Hernández, Nava y Ramos, 2017), enfocada al desarrollo ágil en equipos pequeños de desarrollo, dicha metodología combina elementos del modelo de desarrollo incremental y de la metodología de desarrollo ágil XP (*Extreme Programming*) adaptándolos a un equipo pequeño de desarrollo. La aplicación fue desarrollada por solo una persona y se debe de tener en cuenta que el cliente estaba en constante interacción con la aplicación.

El proceso de la metodología utilizada consta de diez pasos, los cuales se describen a continuación (Xochitemol *et al*., 2017):

1. Selección de técnica. El ciclo de desarrollo comienza con la selección de una o varias técnicas de obtención de datos: entrevistas, cuestionarios, etc. Es el analista quien determina qué técnica debe aplicarse en base al tipo de información que se desea obtener. También es el analista quien ejecuta la técnica de obtención de información.
2. Reuniones con el cliente. En las reuniones con el usuario es donde se ejecutan las técnicas de obtención de datos.
3. Bitácoras. El analista elabora las bitácoras para llevar un registro de reuniones con cliente.
4. Historias de usuario. Tras las reuniones con el cliente el analista determina las historias de usuario. Sirven para documentar las modificaciones o identificar nuevos requisitos de la aplicación.
5. Obtención de requisitos. De las historias de usuario se identifican los nuevos requisitos solicitados por el cliente.
6. Priorización de requisitos. El analista junto con el cliente, determinan cuál de los requisitos pendientes es más importante y priorizan los requisitos según su nivel de necesidad.
7. Estimación de esfuerzos. El arquitecto de software y el desarrollador estiman el tiempo y esfuerzo necesarios para cubrir cada uno de los requisitos pendientes.
8. Selección de requisitos. En base a la estimación de esfuerzos y a la prioridad de los requisitos, el analista determina que requisitos pueden ser cumplidos y presentados antes de la próxima reunión con el cliente. Una vez determinados, el arquitecto y el diseñador de software plasman una solución a dichos requisitos.
9. Codificación. El desarrollador codifica la solución plasmada por el arquitecto de software. La codificación de los requisitos se realiza en módulos independientes a la aplicación principal. De esta forma no se afecta la funcionalidad de la aplicación mientras se desarrollan.
10. Incremento. El *tester* aplica las pruebas necesarias a los módulos por integrar. Una vez completadas las pruebas y corregidos los errores, el desarrollador se encarga de integrar los nuevos módulos, mediante un incremento a la aplicación principal. Entonces se dice que se ha completado una iteración del ciclo de desarrollo.

Este ciclo debe realizarse tantas veces se considere necesario y hasta que se hayan cumplido con todos los requisitos solicitados por el cliente. En la figura 1, se muestra el proceso de la metodología utilizada.

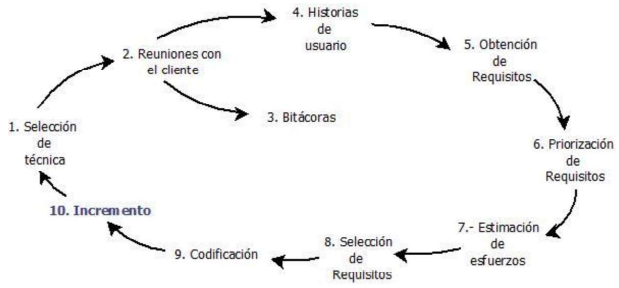


Figura 1. Proceso se la metodología aplicada. (Xochitemol et al., 2017).

Memorias del Congreso Internacional
de Investigación Academia Journals
Morelia 2018

Morelia, Michoacán, México
16 al 18 de mayo de 2018

Análisis de requisitos

El análisis de requisitos de software es una disciplina propia de la ingeniería de software. El objetivo de esta disciplina es poder obtener los requisitos que definirán un software y que luego será diseñado, programado, probado y posteriormente utilizado por los usuarios finales (Noël, Munoz, Becerra y Villarroel, 2016). La obtención de requisitos se realizó mediante entrevistas con el cliente y con usuarios potenciales. Los principales requisitos obtenidos pueden observarse en el cuadro 2.

Número	Descripción	Tipo
1	La aplicación debe calcular el resultado de los métodos de forma automática.	Funcional
2	Debe mostrar el cálculo de los métodos paso a paso.	Funcional
3	Se deben incluir los métodos ACI, Walker, Füller y Bolomey.	Funcional
4	Los resultados individuales de cada método deben mostrarse tanto en peso seco como corregido en factor de la humedad.	Funcional
5	Incluir un módulo comparativo, en el cual mostrará tablas y gráficas de los resultados obtenidos.	Funcional
6	Debe exportar los resultados de cada método a formato PDF.	Funcional
7	Incluir gráficas para facilitar la lectura de los resultados finales.	No funcional
8	Permitir abrir y guardar proyectos.	Funcional
9	Incluir herramientas matemáticas para el apoyo de cálculos comunes en el diseño de mezclas.	No funcional
10	Incluir la posibilidad de agregar notas en cada una de las pantallas de la aplicación.	No funcional

Cuadro 2. Principales requisitos funcionales y no funcionales de la aplicación.

En base al análisis realizado a los requisitos obtenidos, se creó el diagrama de bloques que muestra del funcionamiento de la aplicación, el cual puede observarse en la figura 2.

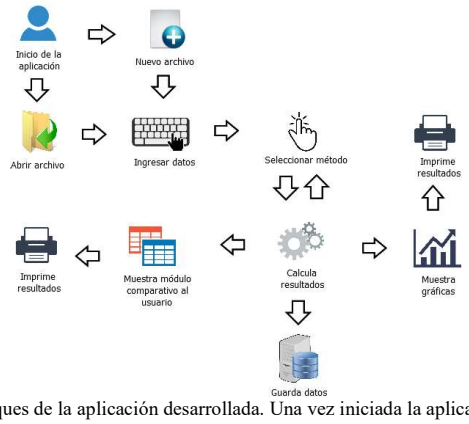


Figura 2. Diagrama de bloques de la aplicación desarrollada. Una vez iniciada la aplicación el usuario tiene dos opciones; abrir un proyecto existente o iniciar desde uno nuevo. Después el usuario deberá ingresar los datos generales y seleccionar uno de los cuatro métodos de diseño de mezclas disponibles, una vez completado el método el usuario podrá visualizar e imprimir gráficas y tablas con los resultados obtenidos. Los datos se guardan automáticamente cada vez que el usuario realiza un cambio.

Prototipado rápido

Memorias del Congreso Internacional
de Investigación Academia Journals
Morelia 2018

Morelia, Michoacán, México
16 al 18 de mayo de 2018

El prototipado rápido de la aplicación se realizó a través de la herramienta GUI Design Studio, dicho prototipo fue presentado y aprobado por el cliente. Un ejemplo de la pantalla inicial del prototipo presentado al cliente puede ver se en la figura 3.

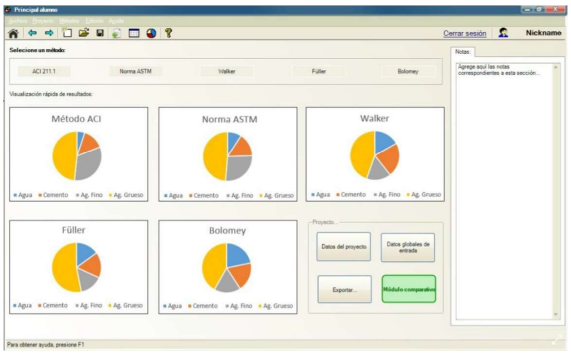


Figura 3. Pantalla principal de la aplicación según el prototipado.

Desarrollo de la aplicación

En base a la aplicación de la metodología de desarrollo que se presentó a lo largo de esta sección, así como en el uso de las herramientas de desarrollo mencionadas en la sección anterior y tras siete iteraciones se obtuvo una aplicación capaz de calcular el proporcionamiento de mezclas de concreto mediante los métodos de diseño de mezclas de concreto ACI, Walker, Bolomey y Füller. Además, cuenta con un módulo que permite comparar los resultados obtenidos en cada uno de los métodos calculados, con el objetivo de que el usuario pueda evaluar cuál de estos obtiene los mejores resultados. La aplicación también incluye otras funcionalidades como herramientas matemáticas comunes en el diseño de mezclas (cálculo de desviación estándar, interpolación lineal, granulometría de agregados, etc.), gestión de proyectos y exportación de resultados a formato PDF, tanto de los resultados individuales por método como los del módulo comparativo.

Resultados

A lo largo del ciclo de desarrollo, el cumplimiento de los requisitos (véase cuadro 2) fue validado por el cliente durante cada iteración, habiendo un total de siete iteraciones. Como resultado de esto, se obtuvo una aplicación que cumple con todos los requisitos solicitados por el cliente, logrando de esta forma su satisfacción. El progreso y las observaciones realizadas durante cada iteración pueden observarse en el cuadro 3. Los requisitos número 1, 4 y 7 necesitaron de varias iteraciones para ser completadas.

Conclusiones

La aplicación desarrollada permite calcular rápidamente el proporcionamiento de mezclas de concreto mediante los métodos de diseño ACI, Walker, Füller y Bolomey. A diferencia de los programas actuales mostrados en el cuadro 1, esta aplicación permite mostrar el desarrollo paso a paso de cada uno de los métodos. Además, ofrece una breve explicación en cada paso, por lo que puede ser utilizada para la enseñanza y aprendizaje de estos, así como para verificar resultados obtenidos en prácticas previas.

Una ventaja más sobre las demás aplicaciones es que permite modificar directamente las variables que interfieren en cada paso, ofreciendo así mayor flexibilidad al proceso de cada método. También incluye un módulo comparativo que muestra mediante gráficas y tablas los resultados obtenidos en cada uno de los métodos calculados, de esa manera el usuario puede evaluar qué método le ofrece mejores resultados. Debido a la practicidad de la aplicación como trabajo futuro se considera factible trasladar la aplicación a dispositivos móviles, así como incluir más métodos de diseños de mezclas.

Memorias del Congreso Internacional
de Investigación Academia Journals
Morelia 2018

Morelia, Michoacán, México
16 al 18 de mayo de 2018

Descripción de la iteración.	Requisito #	Resultado	Observaciones
1.- Desarrollo del prototipo de la interfaz de aplicación.	2	Aprobado	Prototipo aprobado.
	3	Aprobado	
	5	Aprobado	
2.- Desarrollo del método ACI.	1	En progreso	Se aprobó lo relacionado al método ACI.
	4	En progreso	
	7	En progreso	
3.- Desarrollo del método Walker.	1	En progreso	Se aprobó lo relacionado al método Walker.
	4	En progreso	
	7	En progreso	
4.- Desarrollo del método Füller.	1	En progreso	Se aprobó lo relacionado al método Füller.
	4	En progreso	
	7	En progreso	
5.- Desarrollo del método Bolomey.	1	Aprobado	Se aprobó el proceso de todos los métodos.
	4	Aprobado	
	7	Aprobado	
6.- Desarrollo del módulo comparativo.	5	Aprobado	Se aprobó el módulo comparativo.
7.- Desarrollo de complementos.	6	Aprobado	Se aprobaron las herramientas matemáticas, la gestión de proyectos y la exportación a PDF.
	8	Aprobado	
	9	Aprobado	
	10	Aprobado	

Cuadro 3. Validación de requisitos durante las iteraciones.

Referencias

Alfonso, P.L., Mariño, S. y Godoy, M.V. (2011). Propuesta Metodológica para la Gestión de Software Ágil Basado en la Web. *Multiciencias*, 11(4), 395-401.

Dimezco 2000 (2013). Bienvenido(a) Dimezco 2000. Disponible en https://dimezco2000.wordpress.com/2013/09/30/bienvenido-a-dimezco-2000/.

Harrington, J.L. (2003). SQL Clearly Explained Second Edition. EE.UU.: Morgan Kaufmann Publishers.

Kosmatka, S. H., Kerkhoff, B., Panarese, W. C. y Tanesi, Jussara (2004). Diseño y Control de Mezclas de Concreto. Illinois, EE.UU.: Portland Cement Association.

Noël, R., Munoz, R., Becerra, C. y Villarroel, R. (2016). Developing competencies for software requirements analysis through project based learning. *2016 35th International Conference of the Chilean Computer Science Society (SCCC).* 1-7.

Ojeda-Guerra, C.N. (2015). A Simple Software Development Methodology Based on MVP for Android Applications in a Classroom Context. *2015 IEEE International Conference on Computer and Information Technology.* 1429-1434.

Perea, R., Fernández, I., Arroyo, M., Rodríguez, J.A., Camacho, E. y Montesinos, P. (2016). Multiplatform application for precision irrigation scheduling in strawberries. *Agricultural Water Management.* 183, 194-201.

Perry, J.S. (2012). Introducción a la programación Java, parte 1: Conceptos básicos del lenguaje Java. Disponible en https://www.ibm.com/developerworks/ssa/java/tutorials/j-introtojava1/index.html

Rojas, J. J. (2011). Programas para el diseño de mezclas de concreto. Obtenido de civilgeeks.com. Disponible en: https://civilgeeks.com/2011/03/12/programa-para-diseno-de-mezclas-de-concreto/.

Satoto, K.I., Isnanto, R.R., Kridalukmana, R. y Martono, K.T. (2016). Optimizing MySQL database system on information systems research, publications and community service. *Information Technology, Computer, and Electrical Engineering (ICITACEE)*. 1-5.

Sobolev, K. (2003). The development of a new method for the proportioning of high-performance concrete mixtures. *Cement & Concrete Composites*, 23, 901–907.

Xochitemol Cruz, A., Medina Barrrera, M.G., Hernández Mora, J.J., Nava Bautista, H. y Ramos, J. (2017). Propuesta Metodológica de Desarrollo Ágil para Software Enfocada a Equipos Pequeños. *Memorias del Congreso Internacional de Investigación Academia Journals Celaya 2017.* 36, 7199-7204.

Anexo E

Cartas

SEP
SECRETARÍA DE EDUCACIÓN PÚBLICA

TECNOLÓGICO NACIONAL DE MÉXICO
Instituto Tecnológico de Apizaco
Departamento de Ciencias de la Tierra

"Año del Centenario de la Promulgación de la Constitución Política de los Estados Unidos Mexicanos"

Apizaco, Tlax., a **27/Noviembre/2017**
Depto. de Ciencias de la Tierra
Departamento No. 15
Núm. de Oficio: C. T. **1047/17**

Asunto: Constancia de Satisfacción

MTRO. FELIPE PASCUAL ROSARIO AGUIRRE
DIRECTOR DEL I.T. APIZACO
P R E S E N T E

AT´N: DR. JOSÉ FEDERICO CASCO VÁZQUEZ
JEFE DE LA DIVISIÓN DE ESTUDIOS DE POSGRADO E INVESTIGACIÓN

Por medio del presente le envío un cordial saludo y aprovecho para hacerle constar que posterior a la recepción del proyecto **"Desarrollo de una aplicación didáctica para el diseño y evaluación de mezclas de concreto"** realizado por el Ing. Alfredo Xochitemol Cruz, con número de control: M10370762 y dirigido por la M.C. María Guadalupe Medina Barrera, alumno y profesora respectivos de la Maestría en Sistemas Computacionales, de la Institución que usted destacadamente dirige, cubre y satisface las expectativas planteadas al inicio de este proyecto.

Agradeciendo sus atenciones a la presente quedo de usted.

A T E N T A M E N T E
"PENSAR PARA SERVIR, SERVIR PARA TRIUNFAR"

SECRETARÍA DE EDUCACIÓN PÚBLICA
TECNOLÓGICO NACIONAL DE MÉXICO
INSTITUTO TECNOLÓGICO DE APIZACO
DEPARTAMENTO DE CIENCIAS DE LA TIERRA

M. en I. MIGUEL ANGEL TLATZIMATZI FLORES
JEFE DEL DEPTO. DE CIENCIAS DE LA TIERRA

MATF*Izis
c.c.p. División de Estudios de Posgrado e Investigación
c.c.p. Dirección
c.c.p. Archivo

SEP
SECRETARÍA DE EDUCACIÓN PÚBLICA

TECNOLÓGICO NACIONAL DE MÉXICO
Instituto Tecnológico de Apizaco
Departamento de Ciencias de la Tierra

"Año del Centenario de la Promulgación de la Constitución Política de los Estados Unidos Mexicanos"

Apizaco, Tlax., a **27/Noviembre/2017**
Depto. de Ciencias de la Tierra
Departamento No. 15
Núm. de Oficio: C. T. **1046/17**

Asunto: Constancia

MTRO. FELIPE PASCUAL ROSARIO AGUIRRE
DIRECTOR DEL I.T. APIZACO
P R E S E N T E

AT´N: DR. JOSÉ FEDERICO CASCO VÁZQUEZ
JEFE DE LA DIVISIÓN DE ESTUDIOS DE POSGRADO E INVESTIGACIÓN

Por medio del presente le envío un cordial saludo y aprovecho para hacerle constar que el Ing. Alfredo Xochitemol Cruz, alumno de la Maestría en Sistemas Computacionales con número de control: M10370762, de la Institución que usted destacadamente dirige, participo en el proyecto:

"DESARROLLO DE UNA APLICACIÓN DIDÁCTICA PARA EL DISEÑO Y EVALUACIÓN DE MEZCLAS DE CONCRETO"

El proyecto tuvo una duración de seis meses en el periodo comprendido de 01 de Marzo al 31 de Agosto del 2017 y fue realizado en el Instituto Tecnológico de Apizaco.

En virtud de que se ha cubierto satisfactoriamente con los objetivos del proyecto. Tengo bien dar constancia de que dicho trabajo realizado por el Ing. Alfredo Xochitemol Cruz, cubre y satisface las expectativas planteadas al inicio del proyecto.

Agradeciendo sus atenciones a la presente quedo de usted.

A T E N T A M E N T E
"PENSAR PARA SERVIR, SERVIR PARA TRIUNFAR"

M. en I. MIGUEL ANGEL TLATZIMATZI FLORES
JEFE DEL DEPTO. DE CIENCIAS DE LA TIERRA

SECRETARÍA DE EDUCACIÓN PÚBLICA
TECNOLÓGICO NACIONAL DE MÉXICO
INSTITUTO TECNOLÓGICO DE APIZACO
DEPARTAMENTO DE CIENCIAS DE LA TIERRA

MATF*lzis
c.c.p. División de Estudios de Posgrado e Investigación
c.c.p. Dirección
c.c.p. Archivo