Descripción: utemUNIVERSIDAD TECNOLÓGICA METROPOLITANA

FACULTAD DE INGENIERÍA

DEPARTAMENTO DE INFORMÁTICA



Tarea 2:

Ingeniería de Software

Integrantes : Javier Carrasco

Rosa González Soto

Hernán Jara Leiva

Asignatura : Ingeniería de Software

Profesor : Luis Herrera Henríquez

Sección : 2

Fecha : Lunes, 8 de Septiembre del 2014

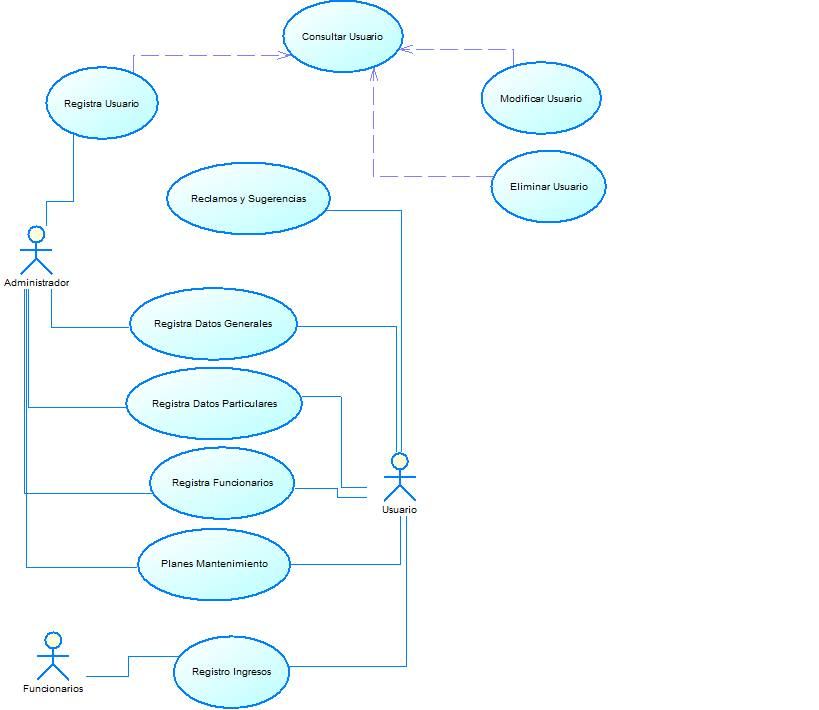
**Casos de Uso Proyecto Gestión Edificios**

Un caso de uso es una descripción de los pasos o las actividades que deberán realizarse para llevar a cabo algún proceso. En el contexto de ingeniería del software, un caso de uso es una secuencia de interacciones que se desarrollarán entre un sistema y sus actores en respuesta a un evento que inicia un actor principal sobre el propio sistema. Los diagramas de casos de uso sirven para especificar la comunicación y el comportamiento de un sistema mediante su interacción con los usuarios y/u otros sistemas.

Notación de caso de uso:



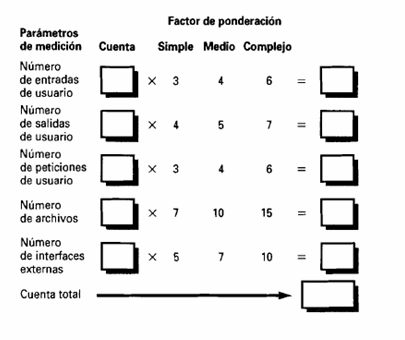
El modelo de caso de uso del proyecto sería:



**Punto de función**

La medida de punto de función se diseñó originalmente para aplicarse a aplicaciones de sistemas de información de gestión. Para acomodar estas aplicaciones, se enfatizó la dimensión de datos (los valores de dominios de información) para la exclusión de dimensiones (control) funcionales y de comportamiento.

Los puntos de función se calculan completando la siguiente tabla:



|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Componente** | **Bajo** | **Medio** | **Alto** | **Total** |
| **EI** | 0\*3 | 8\*4 | 0\*6 | 32 |
| **EO** | 0\*4 | 0\*5 | 0\*7 | 0 |
| **EQ** | 0\*3 | 7\*4 | 0\*6 | 28 |
| **ILF** | 0\*7 | 1\*10 | 0\*15 | 10 |
| **EIF** | 0\*5 | 0\*7 | 0\*10 | 0 |
|  |  |  |  | 70 |

* Entradas: Existen 8 entradas de complejidad media para el sub-sistema Gestión Administrador. No existen entradas de complejidades baja o alta.   
    
  Salidas: No existen una salida de datos hacia el exterior de la aplicación.
* Consultas: Existen 7 entradas de complejidad media para el subsistema Gestión Usuario. No existen entradas de complejidades baja o alta.
* Fichero Lógicos Internos: Existe un almacén de complejidad media.
* Fichero Externos: No se utilizaron almacenes externos de datos.

**Extreme Programming(XP)**

**Manifiesto ágil:**

• Al individuo y las interacciones del equipo de desarrollo sobre el proceso y las herramientas. La gente es el principal factor de éxito de un proyecto software. Es más importante construir un buen equipo que construir el entorno. Muchas veces se comete el error de construir primero el entorno y esperar que el equipo se adapte automáticamente. Es mejor crear el equipo y que éste configure su propio entorno de desarrollo en base a sus necesidades.

• Desarrollar software que funciona más que conseguir una buena documentación. La regla a seguir es “no producir documentos a menos que sean necesarios de forma inmediata para tomar un decisión importante”. Estos documentos deben ser cortos y centrarse en lo fundamental.

• La colaboración con el cliente más que la negociación de un contrato. Se propone que exista una interacción constante entre el cliente y el equipo de desarrollo. Esta colaboración entre ambos será la que marque la marcha del proyecto y asegure su éxito.

• Responder a los cambios más que seguir estrictamente un plan. La habilidad de responder a los cambios que puedan surgir a los largo del proyecto (cambios en los requisitos, en la tecnología, en el equipo, etc.) determina también el éxito o fracaso del mismo. Por lo tanto, la planificación no debe ser estricta sino flexible y abierta.

Los valores anteriores inspiran los doce principios del manifiesto. Son características que diferencian un proceso ágil de uno tradicional. Los dos primeros principios son generales y resumen gran parte del espíritu ágil. El resto tienen que ver con el proceso a seguir y con el equipo de desarrollo, en cuanto metas a seguir y organización del mismo. Los principios son:

**I.** La prioridad es satisfacer al cliente mediante tempranas y continuas entregas de software que le aporte un valor.

**II.** Dar la bienvenida a los cambios. Se capturan los cambios para que el cliente tenga una ventaja competitiva.

**III.** Entregar frecuentemente software que funcione desde un par de semanas a un par de meses, con el menor intervalo de tiempo posible entre entregas.

**IV.** La gente del negocio y los desarrolladores deben trabajar juntos a lo largo del proyecto.

**V.** Construir el proyecto en torno a individuos motivados. Darles el entorno y el apoyo que necesitan y confiar en ellos para conseguir finalizar el trabajo.

**VI.** El diálogo cara a cara es el método más eficiente y efectivo para comunicar información dentro de un equipo de desarrollo.

**VII.** El software que funciona es la medida principal de progreso.

**VIII.** Los procesos ágiles promueven un desarrollo sostenible. Los promotores, desarrolladores y usuarios deberían ser capaces de mantener una paz constante.

**IX.** La atención continua a la calidad técnica y al buen diseño mejora la agilidad.

**X.** La simplicidad es esencial.

**XI.** Las mejores arquitecturas, requisitos y diseños surgen de los equipos organizados por sí mismos.

**XII.** En intervalos regulares, el equipo reflexiona respecto a cómo llegar a ser más efectivo, y según esto ajusta su comportamiento.

**Programación Extrema (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 desarrolladores, y propiciando un buen clima de trabajo. XP se basa en realimentación continua entre el cliente y el equipo de desarrollo, comunicación fluida entre todos los participantes, simplicidad en las soluciones implementadas y coraje para enfrentar los cambios. XP se define como especialmente adecuada para proyectos con requisitos imprecisos y muy cambiantes, y donde existe un alto riesgo técnico.

**Roles XP**

* Programador: Hérnan Jara, Javier Carrasco, Rosa González
* Cliente: Sofía Camilla(Profesora)
* Encargado de Pruebas (Tester): Hérnan Jara, Javier Carrasco, Rosa González
* Encargado de Seguimiento (Tracker): Hérnan Jara, Javier Carrasco, Rosa González
* Entrenador(Coach): Hérnan Jara, Javier Carrasco, Rosa González(Roles rotativos mes a mes)
* Consultor: Luis Herrera(Profesor), Fernando Rubilar(Ayudante)
* Gestor(Big Boss): Hérnan Jara, Javier Carrasco, Rosa González(Roles rotativos mes a mes)

**Proceso XP**

1. El cliente define el valor de negocio a implementar.

2. El programador estima el esfuerzo necesario para su implementación.

3. El cliente selecciona qué construir, de acuerdo con sus prioridades y las restricciones de tiempo.

4. El programador construye ese valor de negocio.

5. Vuelve al paso 1.

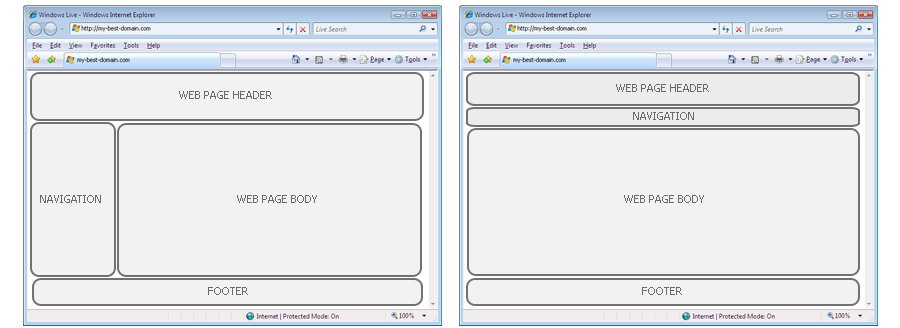
\*Estos pasos se cumplieron la primera semana, y están especificados de manera más detallada en la Carta Gantt.

**Prácticas XP**

* **Planificación**: Se conversó directamente con nuestro cliente (Prof Sofía Camilla) y obtuvieron sus necesidades para la creación del software. En adelante, se planificó un plan de avance de proyecto.
* **Entregas:** Se encuentra en proceso de desarrollo. El primer paso es educación con respecto a lenguaje a utilizar y el modo en que se ejecutará. Semana a semana se entregan avances sobre el proyecto en clases.
* **Metáfora:** El cliente es dueño de un departamento ubicado en la comuna de Santiago centro. Existen varias problemáticas de comunicación en el inmueble entre habitantes y administración, es por esto que existe una necesidad básica es obtener información detallada y pública acerca de gastos comunes, gastos de reparación/mantención, reglamento del edificio, bodegas disponibles para arrendar y/o comprar, estacionamientos disponibles para arrendar y/o comprar, reserva de salas, cartolas históricas de gastos y gráficos anuales de cada departamento.

Debido a esto, plantea la creación de un sistema de información de ingeniería para integrar el trabajo de la administración, la relación entre los trabajadores y los habitantes del edificio.

* **Diseño:** Diseño básico de una página web



* **Pruebas:** Se realizarán por parte de los miembros del equipo, buscando fallos, así como también mostrando avances al cliente.
* **Refactorización:** Se realizará por parte de todo el equipo una vez se superen las distintas fases de diseño.
* **Programación:** Se contará con el equipo en un mismo lugar, para lograr manejo más rápido de consultas e interacción entre códigos.
* **Propiedad colectiva de código:** Para esto se utilizará Github, en el cual se maneja control de versiones, se sabrá qué se modificó y por quién, además de que se podrán agregar comentarios con respecto a dicha modificación.
* **Integración continua:** Nuevamente, Github juega un papel importante en el desarrollo de esta tarea por el uso de control de versiones.
* **40 horas por semana:** Se asignaron días y horas de trabajo en las que el equipo está en contacto y podrá suplir sus necesidades en el tiempo requerido.
* **Cliente in-situ:** Al cliente se le puede ubicar tanto como por correo electrónico como en persona dentro del Departamento de Física en la Universidad Tecnológica Metropolitana.
* **Estándares de programación:**

|  |  |  |
| --- | --- | --- |
| **Tipo de control** | **Prefijo** | **Ejemplo** |
| Panel 3D | Pnl | pnl\_Grupo |
| Botón animado | Ani | ani\_Buzon |
| Casilla de verificación | Chk | chk\_SoloLectura |
| Botón | Btn | btn\_ejemplo |
| LinkButton | Lnk | lnk\_Ejemplo |
| HyperLink | Hyp | hyp\_Ejemplo |
| GroupBox | Grp | grp\_Ejemplo |
| CheckedListBox | Clst | Clst\_Ejemplo |
| DataTimePicker | Dtp | Dtp\_Ejemplo |
| MonthCalendar | Cal | Cal\_Ejemplo |
| PrintDialog | Pd | Pd\_Ejemplo |
| PrintPreviewDialog | Ppd | Ppd\_Ejemplo |
| PrintPreviewControl | Ppc | Ppc-Ejemplo |
| ErrorProvider | Errp | Errp\_Ejemplo |
| PrintDocument | Pdoc | Pdoc\_Ejemplo |
| PageSetupDialog | Psd | Psd\_Ejemplo |
| Diálogo común | Dlg | Dlg\_ArchivoAbrir |
| Comunicaciones | Com | Com\_Fax |
| Control | Ctr | Ctr\_Activo |
| Control de datos | Dat | Dat\_Biblio |
| Cuadro de datos | Dbcmb | Dbcmb\_Lenguaje |
| Formulario | Frm | Frm\_Entrada |
| Gráfico | Grf | Grf\_Ingresos |
| Imagen (Image) | Img | imgIcono |
| Menú | Mnu | mnuArchivoAbrir |
| Vista de árbol | tre | treOrganizacion |
| Barra de herramientas | Tlb | tlbAcciones |
| GridView | Gdv | gdvMostrarDatos |
| DataList | Dlg | dlCargarDatos |
| SqlDataSource | Sds | sdsConexion |
| XmlDataSource | Xds | xdsXmlCarga |
| ReportViewer | Rw | ReportViewer |
| RequeridFieldValidator | Rfv | rfvValidacion |
| RangeValidator | Rv | rvRangoValidacion |
| RegularExpressionValidator | Rev | revValidacionRegular |
| Login | Lg | lgLogin |
| LoginView | lv | lvVistaUsuario |

**Carta Gantt**

La Carta Gantt consiste en elaborar un cronograma donde se fijan tareas, metas y objetivos a cumplir de acuerdo a tiempos establecidos por el propio empresario, quien deberá evaluar y constatar el cumplimiento de ellos en función de lo programado. Sirve para calendarizar tareas. Es esencialmente una gráfica en donde las barras representan cada tarea o actividad. La longitud de cada barra representa la relatividad de la tarea.