

**Universidad Abierta Interamericana**

**Facultad de Tecnología Informática**

**Ingeniería en Sistemas Informáticos**

**Seminario de Aplicación Profesional**

**Nombre del proyecto: “Pickupmeal”**

**Integrantes:**

Córdoba, Pablo – Legajo: 31233 – Sede: Lagos

Dominguez, Jacobo – Legajo: 31241 – Sede: Lagos

**Año:**

2015

**Docentes:**

Ing. Audogio, Pablo

Ing. Bracamonte, Lucas

Contenido

[Inicio del proyecto 6](#_Toc437969507)

[Project Charter 6](#_Toc437969508)

[Scope Statement 12](#_Toc437969509)

[Registro de Stakeholders 14](#_Toc437969510)

[Minutas de reunión – Inicio del Proyecto 15](#_Toc437969511)

[Analisis de requerimientos 18](#_Toc437969512)

[Minutas de reunión – Extracción de requerimientos 18](#_Toc437969513)

[Listado de requerimientos 22](#_Toc437969514)

[Requerimientos Funcionales 22](#_Toc437969515)

[Requerimientos No Funcionales 22](#_Toc437969516)

[Prototipos de interfaz 23](#_Toc437969517)

[Definición de Casos de Uso 26](#_Toc437969518)

[Diagrama de casos de uso 26](#_Toc437969519)

[Especificación de casos de uso 27](#_Toc437969520)

[Modelo de dominio 34](#_Toc437969521)

[Revisión preliminar 35](#_Toc437969522)

[Matriz de calidad 35](#_Toc437969523)

[Análisis de diseño preliminar 36](#_Toc437969524)

[Arquitectura técnica 36](#_Toc437969525)

[Tecnología a aplicar: 36](#_Toc437969526)

[Análisis de robustez 37](#_Toc437969527)

[Modelo de Dominio Actualizado 42](#_Toc437969528)

[Diseño Detallado 43](#_Toc437969529)

[Diagramas de Secuencia 43](#_Toc437969530)

[Diagramas de Clases 47](#_Toc437969531)

[Models 48](#_Toc437969532)

[Dao 49](#_Toc437969533)

[Services 50](#_Toc437969534)

[Controllers 51](#_Toc437969535)

[Dto 52](#_Toc437969536)

[Análisis de requerimientos 53](#_Toc437969537)

[Minutas de reunión – Extracción de requerimientos 53](#_Toc437969538)

[Listado de requerimientos 57](#_Toc437969539)

[Requerimientos Funcionales 57](#_Toc437969540)

[Requerimientos No Funcionales 57](#_Toc437969541)

[Prototipos de Interfaz 58](#_Toc437969542)

[Definición de Casos de Uso 60](#_Toc437969543)

[Diagrama de casos de uso 60](#_Toc437969544)

[Especificación de casos de uso 61](#_Toc437969545)

[Modelo de dominio 65](#_Toc437969546)

[Revisión Preliminar 66](#_Toc437969547)

[Matriz de calidad 66](#_Toc437969548)

[Analisis de Diseño Preliminar 67](#_Toc437969549)

[Arquitectura técnica 67](#_Toc437969550)

[Tecnología a aplicar: 67](#_Toc437969551)

[Análisis de robustez 68](#_Toc437969552)

[Modelo de Dominio Actualizado 69](#_Toc437969553)

[Diseño Detallado 70](#_Toc437969554)

[Diagramas de Secuencia 70](#_Toc437969555)

[Diagramas de Clases 72](#_Toc437969556)

[Models 73](#_Toc437969557)

[Dao 74](#_Toc437969558)

[Services 75](#_Toc437969559)

[Controllers 76](#_Toc437969560)

[Dto 77](#_Toc437969561)

[Anexo I: Procesos de Desarrollo de Software 78](#_Toc437969562)

[Introducción 78](#_Toc437969563)

[Modelo de Cascada 79](#_Toc437969564)

[Modelo Incremental 80](#_Toc437969565)

[Modelo DRA 81](#_Toc437969566)

[Modelo de Prototipos 82](#_Toc437969567)

[Modelo en Espiral 83](#_Toc437969568)

[Modelo Concurrente 84](#_Toc437969569)

[Modelo de desarrollo basado en componentes 85](#_Toc437969570)

[Proceso Unificado: 86](#_Toc437969571)

[Desarrollo ágil 91](#_Toc437969572)

[ICONIX 92](#_Toc437969573)

[Conclusión: 93](#_Toc437969574)

[Anexo II: Gestión de Riesgos 94](#_Toc437969575)

[Introducción: 94](#_Toc437969576)

[Lista de priorización de riesgos 96](#_Toc437969577)

[Planificación de riesgos 98](#_Toc437969578)

[Valoración de riesgos (factores) 99](#_Toc437969579)

[Conclusión: 99](#_Toc437969580)

[Anexo III: Gestión de calidad 100](#_Toc437969581)

[Introducción: 100](#_Toc437969582)

[Modelo CMMI: 100](#_Toc437969583)

[Modelo ISO 102](#_Toc437969584)

[Conclusión 102](#_Toc437969585)

[Anexo IV: Gestión de la configuración del Software 104](#_Toc437969586)

[Introducción: 104](#_Toc437969587)

[Proceso de configuración de software 104](#_Toc437969588)

[Identificación de objetos en la configuración de software 105](#_Toc437969589)

[Control de versión 106](#_Toc437969590)

[Control de cambio 106](#_Toc437969591)

[Pedido de modificación de configuración de software (Usuario) 108](#_Toc437969592)

[Notificación del pedido (Área de sistema) 109](#_Toc437969593)

[Auditoría: 109](#_Toc437969594)

[Informe de estado: 110](#_Toc437969595)

[Anexo V: Políticas de Backup 111](#_Toc437969596)

[Introducción: 111](#_Toc437969597)

[¿Qué es un backup? 111](#_Toc437969598)

[Política de backup. 111](#_Toc437969599)

[Ventajas de un backup 112](#_Toc437969600)

[Desventajas de un backup 112](#_Toc437969601)

[Tipos de backup 112](#_Toc437969602)

[Aplicación: 113](#_Toc437969603)

[Anexo VI: Patrones de Diseño 116](#_Toc437969604)

[Introducción 116](#_Toc437969605)

[Patrones J2EE 118](#_Toc437969606)

[Catálogo de patrones J2EE 118](#_Toc437969607)

[Capa de presentación 118](#_Toc437969608)

[Capa de Negocios 119](#_Toc437969609)

[Capa de Integración 120](#_Toc437969610)

[Aplicación 121](#_Toc437969611)

[Intercepting Filter – Capa de presentación 121](#_Toc437969612)

[Data Transfer Object – Capa de negocios 124](#_Toc437969613)

[Data Access Object – Capa de integración 126](#_Toc437969614)

[Conclusión 130](#_Toc437969615)

[Anexo VII: Seguridad 132](#_Toc437969616)

[Autenticación con JSON Web Tokens 132](#_Toc437969617)

[Anexo VIII: Manual de Usuario 134](#_Toc437969618)

[Introducción: 134](#_Toc437969619)

[**Conclusión** 134](#_Toc437969620)

# Inicio del proyecto

## Project Charter

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **CONTROL DE VERSIONES** | | | | | |
| Versión | Hecha por | Revisada por | Aprobada por | Fecha | Motivo |
| 1.0 | Cordoba Pablo  Dominguez Jacobo | Pablo Audoglio | ---- | 20/04/2015 | Lanzar el inicio del proyecto. |

|  |  |
| --- | --- |
| **NOMBRE DEL PROYECTO** | **SIGLAS DEL PROYECTO** |
| **pickupmeal.com** | **PUM** |
| **DESCRIPCIÓN DEL PROYECTO:** | |
| Se trata de un proyecto que abarca tanto la mediación entre Clientes y Negocios como la logística del negocio para la entrega de los productos solicitados, teniendo a disposición vehículos (motos) y conductores propios. Dicha interacción se realiza mediante una plataforma Web en la que el usuario Cliente puede informarse acerca de las distintas ofertas gastronómicas ofrecidas por los usuarios de tipo Negocio. El sitio, no solo tiene un rol informativo, también sirve de medio interactivo para que el Cliente realice pedidos a los distintos negocios de manera intuitiva, sencilla y clara, pudiendo saber en todo momento en qué estado se encuentra su pedido, ya sea que el Negocio lo ha confirmado para su realización, como el momento en el que sale a la calle. Luego de hacer el pedido, el cuál llega automáticamente al usuario Negocio que corresponda, el mismo puede confirmarlo y comenzar a prepararlo. Cabe destacar que el pedido llega también a la oficina de la empresa creadora del sitio, la cual comienza a trabajar la mejor logística de viajes de los conductores de las motos para lograr viajes eficaces con el menor daño posible al medioambiente. Una vez que el pedido esté listo, los conductores van a buscarlo al negocio y luego lo llevan a destino. | |

|  |
| --- |
| **DEFINICIÓN DE REQUISITOS DEL PROYECTO:** |
| * Cumplimentar las exigencias de contenido solicitado para la aprobación de las cátedras. * Profundizar en los aspectos relacionados con la parte tecnológica y de negocio del proyecto. * Satisfacer en tiempo y forma la solicitud de Entregables del proyecto. * Establecer límites claros respecto a la funcionalidad del producto. * Lograr una calidad de codificación de manera que se pueda lograr una buena escalabilidad. * Lograr una buena comunicación y coordinación entre los miembros del grupo de trabajo. |

|  |  |  |
| --- | --- | --- |
| **OBJETIVOS DEL PROYECTO:** | | |
| ***CONCEPTO*** | ***OBJETIVOS*** | ***CRITERIO DE ÉXITO*** |
| ***1. ALCANCE*** | Definir, idear e implementar un producto integral que satisfaga las necesidades de los usuarios y de los clientes asociados a la solución de la mejor manera, teniendo en cuenta restricciones de distintos aspectos como tiempo, complejidad, herramientas, y conocimientos de los alumnos. | El usuario queda satisfecho con el producto desarrollado y el mismo cumple las expectativas por parte de los profesores de la cátedra. |
| ***2. TIEMPO*** | Realizar el producto propuesto dentro de un rango de tiempo no mayor a la finalización de las asignaturas SAP y TFI | Cumplir con los hitos del proyecto en tiempo y forma. |
| ***3. COSTO*** | El objetivo principal de proyecto es finalizarlo con un fin académico, no obstante, en un futuro inmediato se buscará obtener réditos económicos mediante la implementación de proyecto | Las estimaciones no escapan de los hechos de la realidad.  Sobrepasar los capitales invertidos y establecerse en el mercado |

|  |
| --- |
| **FINALIDAD DEL PROYECTO:** |
| El proyecto tiene como fin último poder incorporar los conocimientos adecuados para un posterior desarrollo en el ámbito laboral, implicando la aprobación de las cátedras nombradas. Por otra parte, se busca establecerse en el mercado regional a partir de diversas estrategias, y así obtener más recursos y crear a partir de esto un círculo virtuoso. |

|  |  |  |
| --- | --- | --- |
| **DESIGNACIÓN DEL PROJECT MANAGER DEL PROYECTO** | | |
| ***NOMBRE*** | Dominguez Jacobo | ***NIVELES DE AUTORIDAD*** |
| ***REPORTA A*** | Audoglio Pablo  Bracamonte Lucas | Audoglio Pablo  Bracamonte Lucas |
| ***SUPERVISA A*** | Cordoba Pablo |

|  |  |
| --- | --- |
| **CRONOGRAMA DE HITOS DEL PROYECTO (Primera Iteración)** | |
| ***HITO O EVENTO SIGNIFICATIVO*** | ***FECHA PROGRAMADA*** |
| Inicio del Proyecto. | 07/04/2015 |
| Reunión I | 20/04/2015 |
| Alcance | 23/04/2015 |
| Project Charter | 23/04/2015 |
| Listado de requerimientos | 28/04/2015 |
| Prototipos GUI | 28/04/2015 |
| Diagrama de caso de uso | 29/04/2015 |
| Descripción de casos de uso | 29/04/2015 |
| Diagramas de Robustez | 29/04/2015 |
| Revisión | 05/05/2015 |
| Entregable: primer parcial de la cátedra SAP | 19/05/2015 |

|  |  |
| --- | --- |
| **ORGANIZACIONES O GRUPOS ORGANIZACIONALES QUE INTERVIENEN EN EL PROYECTO** | |
| ***ORGANIZACIÓN O GRUPO ORGANIZACIONAL*** | ***ROL QUE DESEMPEÑA*** |
| Profesores: Audoglio,Pablo Bracamonte,Lucas | Guía en el desarrollo y seguimiento en las etapas de la creación del software. |
| Universidad Abierta Interamericana | Provee un espacio físico y herramientas que permiten el correcto desarrollo del proyecto. |
| Equipo de trabajo: Pablo Cordoba, Jacobo Dominguez | Análisis, Documentación, Estimación, Diseño, Desarrollo, Testing, Implementación. |

|  |
| --- |
| **PRINCIPALES AMENAZAS DEL PROYECTO** |
| * Poca experiencia en las herramientas utilizadas para el desarrollo. * Conocimientos básicos de programación y manejo de base de datos. * Fracaso en la aceptación por parte de los usuarios finales y/o clientes * Conocimientos básicos en aspectos de negocio * Alcance sobrepasa las expectativas de desarrollo * Falla en la estimación que pueda perjudicar el cumplimiento de los hitos del proyecto. |

|  |
| --- |
| **PRINCIPALES OPORTUNIDADES DEL PROYECTO** |
| * Posibilidades de futuros desarrollos. * Establecerse como empresas referentes en la ciudad. * Posibilidades de generar ingresos mediante publicidades * Inserción en el mercado laboral. * Ganancia de experiencia en el desarrollo de sistemas. |

|  |
| --- |
| **ASUNCIONES** |
| * El proyecto no admite bajo ningún aspecto requerimientos cuya funcionalidad pongan en duda la legalidad del Software. * Los entregables se tratarán de entregar en tiempo y forma * En cuanto a los aspectos legales se hará una tercerización |

|  |
| --- |
| **EXCLUSIONES DEL PROYECTO:** |
| * No se tendrá en cuenta el servicio de gestión de proveedores para la parte de los locales gastronómicos * No se tendrán en cuenta momentáneamente la inclusión de aplicaciones móviles * El servicio se limitará a la ciudad de San Nicolás de los Arroyos. * La plataforma no será multilenguaje |

|  |  |
| --- | --- |
| **RESTRICCIONES DEL PROYECTO:** | |
| ***INTERNOS A LA ORGANIZACIÓN*** | ***AMBIENTALES O EXTERNOS A LA ORGANIZACIÓN*** |
| Tiempo limitado para la aplicación de todas las etapas de la metodología de desarrollo elegida. | Distancia con respecto al cliente lo que dificulta las reuniones frecuentes con el mismo. |
| Poco conocimiento de todas las herramientas para el desarrollo del proyecto. | Ciertos aspectos del dominio del negocio no pueden modelarse a la perfección en un Software. |
| Poco conocimiento del modelo de negocio a tratar. |  |

|  |  |
| --- | --- |
| **SUPUESTOS DEL PROYECTO:** | |
| ***INTERNOS A LA ORGANIZACIÓN*** | ***AMBIENTALES O EXTERNOS A LA ORGANIZACIÓN*** |
| Se supone que los integrantes del equipo poseen conocimientos básicos para encarar la realización del proyecto. | Se asume que los docentes serán una verdadera guía para el desarrollo del proyecto. |
| Se supone que no habrá crisis dentro del equipo que provoque caos en el proyecto. | Se asume que el cliente tiene buena predisposición a la hora de agendar reuniones. |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **PRESUPUESTO PRELIMINAR DEL PROYECTO:** | | | | | |
| ***CONCEPTO*** | | | | | ***MONTO ($)*** |
| ***1. PERSONAL*** | | Analista y programador: Dominguez Jacobo. 4 meses de trabajo | | | $16.800 |
| Analista y programador: Cordoba Pablo. 4 meses de trabajo | | | $16.800 |
| ***2. MATERIALES*** | | Elementos de conectividad (Routers, UPS, Estabiliazadores, Cableado, etc)  Bienes de uso (Impresora Laser Monocromático, Muebles, Iluminación, uniformes, Generador, etc)  Licencias (MySql,JEE) | | | $19.498 |
| ***3. MAQUINAS*** | | 2 Pc (Amd Sempron Hd500 4gb Kit Tec/mouse/Dvd ) + Monitores LCD 19''  3 Motos Baja cilindrada tipo Vespa | | | $57.914 |
| ***4. OTROS COSTOS*** | | Diseño Corporativo y promoción  Aranceles Legales | | | $14.000 |
| ***TOTAL LÍNEA BASE*** | | | | | **$125.012** |
| ***5. RESERVA DE CONTINGENCIA*** | | 5% extra resguardo a posible contratación de personal extra para el desarrollo del proyecto | | | $6.251 |
| ***6. RESERVA DE GESTIÓN*** | | 3 % extra por riesgos desconocidos | | | $3.750 |
| ***TOTAL PRESUPUESTO*** | | | | | **$135.013** |
| **SPONSOR QUE AUTORIZA EL PROYECTO** | | | | | |
| ***NOMBRE*** | ***EMPRESA*** | | ***CARGO*** | ***FECHA*** | |
| Pablo Audoglio | UAI | | Docente |  | |
| Bracamonte, Lucas | UAI | | Docente |  | |

## Scope Statement

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **CONTROL DE VERSIONES** | | | | | |
| Versión | Hecha por | Revisada por | Aprobada por | Fecha | Motivo |
| 1.0 | Cordoba,Pablo; Dominguez,Jacobo | Audoglio,Pablo  Bracamonte, Lucas | Audoglio,Pablo  Bracamonte, Lucas | 23/04/2015 | Determinar los límites del  producto |

|  |  |
| --- | --- |
| **NOMBRE DEL PROYECTO** | **SIGLAS DEL PROYECTO** |
| **pickupmeal.com** | PUM |

|  |  |
| --- | --- |
| **DESCRIPCIÓN DEL ALCANCE DEL PRODUCTO** | |
| ***REQUISITOS QUE INCLUYE*** | ***REQUISITOS QUE EXCLUYE*** |
| Faceta Usuario: incluye el registro de usuarios nuevos, perfiles, información de restaurants y menús con distintos filtros, sistema de puntajes para promociones, gestión de pedidos, trackeo de pedidos, ofertas relacionadas. | Chat en tiempo real, trackeo en tiempo real con gps. |
| Faceta Cliente: incluye la gestión del menú, histórico de platos vendidos, manejo de estado de los pedidos, pedidos en tiempo real. | Gestión de registro de Cliente en la plataforma. |
| Faceta Empresa: pedidos en tiempo real, manejo de logística, manejo de estado, gestión económica referida a los pedidos, gestión de Clientes. Módulo de Seguridad. | Gestión de proveedores, trackeo en tiempo real con gps, manejo administrativo de empleados. |

|  |  |
| --- | --- |
| **CRITERIOS DE ACEPTACIÓN DEL PRODUCTO** | |
| ***CONCEPTOS*** | ***CRITERIOS DE ACEPTACIÓN*** |
| Aspecto general del sistema | Interfaz sencilla y optimizada para la Web, con un diseño al estilo Flat Design con logotipos representativos de la empresa. |
| Performance | Tiempo de respuesta óptimo. |
| Interfaz de usuario | Sencilla, intuitiva y colorida para lograr una interacción amigable y responsive (adaptada al dispositivo). |
| Arquitectura | El proyecto se ejecute bajo los siguientes navegadores web: Chrome, Mozilla, Internet Explorer. |
| Seguridad | Seguridad para que las facetas del negocio no se entrecrucen y dentro de la referida a la empresa, que tengan distintos niveles de jerarquía. |

|  |  |
| --- | --- |
| **ENTREGABLES DEL PROYECTO** | |
| ***FASE DEL PROYECTO*** | ***PRODUCTOS ENTREGABLES*** |
| Ingeniería de Requerimientos | Minutas de reunión, Lista de requerimientos, prototipos de interfaz de usuario. |
| Análisis de Requisitos | Modelo de dominio, diagrama de casos de uso, matriz de calidad, especificación de casos de uso. |
| Análisis de diseño preliminar | Diagramas de robustez, arquitectura técnica, modelo de dominio actualizado. |
| Diseño detallado | Diagrama de secuencia, diagrama de clases. |
| Implementación | Codificación, tests. |

## Registro de Stakeholders

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **CONTROL DE VERSIONES** | | | | | |
| Versión | Hecha por | Revisada por | Aprobada por | Fecha | Motivo |
| 1.0 | Cordoba,Pablo  Dominguez,Jacobo | Bracamonte, Lucas | Audoglio, Pablo | 19/05/2015 | Identificar las personas que se verán afectadas positiva y negativamente por el producto |

|  |  |
| --- | --- |
| NOMBRE DEL PROYECTO | SIGLAS DEL PROYECTO |
| Pickupmeal | **PUM** |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| IDENTIFICACION | | | | | EVALUACION | | | CLASIFICACION | |
| Nombre | Organización | | Localización | Rol en el Proyecto | Expectativas Principales | Influencia Potencial | Fase de mayor interés | Interno/  Externo | Apoyo/  Neutral/  Opositor |
| Negocios gastronómicos | |  | San Nicolás | Cliente | Mejora en los procesos de negocio. | Adhesión al e-business propuesto | Implementación | Externo | Apoyo |
| Usuarios en general |  | | San Nicolás | Cliente | Agilidad y comodidad a la hora de realizar pedido | Mediante los pedidos realizados en la plataforma | Implementación | Externo | Neutro |
| Cadetes |  | | San Nicolás | Empleado | Posibilidad potencial de empleo | Tiempos de entrega | Implementación | Interno | Neutro |
| Administrador | Pickupmeal | | San Nicolás | Empleado | Gestionar recursos y llevar control del negocio | Toma de decisión en la entrega de pedido | Análisis e implementación | Interno | Apoyo |
| Docentes de cátedra | **UAI** | | Rosario | Corrector, guía y apoyo | Obtener un correcto análisis e implementación | Corrección de errores, y consejos en cuanto a los procesos de desarrollo de software | Todas las etapas por las que pasa el proyecto | Interno | Apoyo |

## Minutas de reunión – Inicio del Proyecto

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **MINUTA DE REUNION** | | | | | | |
| **Fecha:** | 20/04/2015 | **Hora Inicio:** | 17:00 | | **Hora Fin:** | 18:40 |
| **Lugar:** | La reunión se realizó en las instalaciones de la empresa creadora de la plataforma web. | | | | | |
| **Participantes** | | | | | | |
| **Cliente** | | | | **Empresa** | | |
| ------------------------------------------------ | | | | 1. Córdoba Pablo 2. Dominguez Jacobo | | |
| **Objetivos de la Reunión** | | | | | | |
| 1. Identificar las necesidades más importantes para los clientes y usuarios. 2. Conocer el motivo por el cual se desea crear la aplicación. 3. Definir un alcance preliminar del proyecto. 4. Definir los beneficios económicos principales. 5. Identificar puntos débiles en el negocio tanto como los aspectos que se van a dejar de lado. 6. Definir los aspectos fundamentales de la aplicación 7. Completar las planillas de PMI para asentar la documentación organizativa del proyecto. | | | | | | |
| **Temas Principales Tratados** | | | | | | |
| 1. Definición de las funciones y aspectos principales del negocio. 2. Aspectos principales de las reglas de negocio. 3. Determinación del alcance del proyecto 4. Integración o no de los proveedores de servicios a los restaurants. 5. Cumplimentación de las plantillas de PMI (Project Charter, Alcance del Proyecto). 6. Nombre comercial de la aplicación. No se llegó a ninguno. | | | | | | |
| **Temas Secundarios Tratados** | | | | | | |
| 1. Planificación de la documentación y calendario a cuatro semanas. 2. Creación de repositorio online para almacenar la documentación entre los desarrolladores. 3. Establecimiento de formatos para la documentación. | | | | | | |
| **Próximos Pasos** | | | | | | |
| 1. Cumplimentación de las plantillas restantes PMI. 2. Continuar con la definición de alcance y funcionalidades. 3. Definición de requerimientos para las primeras iteraciones 4. Prototipos de Interfaces.   Se estipula agenda una próxima reunión dentro del término de dos días. | | | | | | |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **MINUTA DE REUNION** | | | | | | |
| **Fecha:** | 23/04/2015 | **Hora Inicio:** | 17:00 | | **Hora Fin:** | 20:30 |
| **Lugar:** | La reunión se realizó en las instalaciones de la empresa creadora de la plataforma web. | | | | | |
| **Participantes** | | | | | | |
| **Cliente** | | | | **Empresa** | | |
| ------------------------------------------------ | | | | 1. Córdoba Pablo 2. Dominguez Jacobo | | |
| **Objetivos de la Reunión** | | | | | | |
| 1. Cumplimentación de las plantillas restantes PMI. 2. Continuar con la definición de alcance y funcionalidades. 3. Definición de requerimientos para las primeras iteraciones 4. Prototipos de Interfaces. | | | | | | |
| **Temas Principales Tratados** | | | | | | |
| 1. Definición de alcance del proyecto. 2. Cumplimentación de las plantillas PMI para el lanzamiento del proyecto. 3. Definición de primeros requerimientos para las primeras iteraciones. 4. Determinación del nombre de la aplicación. 5. Investigación de otras plataformas similares en busca de oportunidades. | | | | | | |
| **Temas Secundarios Tratados** | | | | | | |
| 1. Diseño de logotipo institucional. | | | | | | |
| **Próximos Pasos** | | | | | | |
| 1. Prototipos de interfaces. 2. Lista de requerimientos. 3. Diagramas de casos de uso.   Se estipula agenda una próxima reunión dentro del término de cinco días. | | | | | | |

# Analisis de requerimientos

## Minutas de reunión – Extracción de requerimientos

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **MINUTA DE REUNION** | | | | | |
| **Fecha:** | 29/04/2015 | **Hora Inicio:** | 15:30 | **Hora Fin:** | 18:30 |
| **Lugar:** | La reunión se realizó en las instalaciones de la empresa creadora de la plataforma web. | | | | |
| **Participantes** | | | | | |
| **Cliente** | | | **Empresa** | | |
| ------------------------------------------------ | | | 1. Córdoba Pablo 2. Dominguez Jacobo | | |
| **Objetivos de la Reunión** | | | | | |
| 1. Cumplimentar los prototipos de interfaces. 2. Cumplimentar la lista de requerimientos de las primeras iteraciones. 3. Comienzo de desarrollo del modelo de dominio. 4. Compra del dominio .com para el proyecto. | | | | | |
| **Temas Principales Tratados** | | | | | |
| 1. Diseño de prototipos de interfaz de usuario referentes a los primeros casos de uso. 2. Avance con la lista de requerimientos. 3. Avance en la confección del modelo de dominio. 4. Compra efectiva del dominio para el proyecto (pickupmeal.com). | | | | | |
| **Temas Secundarios Tratados** | | | | | |
| 1. Primeras definiciones de las tecnologías y frameworks a utilizar para el desarrollo de la aplicación. | | | | | |
| **Próximos Pasos** | | | | | |
| 1. Avanzar prototipos de interfaces. 2. Avanzar la lista de requerimientos. 3. Diagrama de casos de uso.   Se estipula agenda una próxima reunión dentro del término de seis días. | | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **MINUTA DE REUNION** | | | | | |
| **Fecha:** | 29/04/2015 | **Hora Inicio:** | 15:30 | **Hora Fin:** | 18:00 |
| **Lugar:** | La reunión se realizó en las instalaciones de la empresa creadora de la plataforma web. | | | | |
| **Participantes** | | | | | |
| **Cliente** | | | **Empresa** | | |
| ------------------------------------------------ | | | 1. Córdoba Pablo 2. Dominguez Jacobo | | |
| **Objetivos de la Reunión** | | | | | |
| 1. Cumplimentar los prototipos de interfaces. 2. Cumplimentar la lista de requerimientos de las primeras iteraciones. 3. Comienzo de desarrollo del modelo de dominio. 4. Confección del diagrama de casos de uso | | | | | |
| **Temas Principales Tratados** | | | | | |
| 1. Finalización de los diseños de interfaz. 2. Avance con la lista de requerimientos. 3. Avance en la confección del modelo de dominio. 4. Diagrama de casos de uso. | | | | | |
| **Temas Secundarios Tratados** | | | | | |
| 1. Definiciones definitivas de las tecnologías a implementar. | | | | | |
| **Próximos Pasos** | | | | | |
| 1. Avanzar con el diagrama de casos de uso. 2. Comenzar con las descripciones de los casos de uso. 3. Avanzar con el modelo de dominio. 4. Comenzar con los diagramas de robustez.   Se estipula agenda una próxima reunión dentro del término de dos semanas. | | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **MINUTA DE REUNION** | | | | | |
| **Fecha:** | 13/05/2015 | **Hora Inicio:** | 15:30 | **Hora Fin:** | 19:00 |
| **Lugar:** | La reunión se realizó en las instalaciones de la empresa creadora de la plataforma web. | | | | |
| **Participantes** | | | | | |
| **Cliente** | | | **Empresa** | | |
| ------------------------------------------------ | | | 1.Córdoba Pablo  2.Dominguez Jacobo | | |
| **Objetivos de la Reunión** | | | | | |
| 1. Revisión de requisitos. 2. Revisión de prototipos de interfaz. 3. Revisión de casos de uso. 4. Revisión de modelo de dominio. | | | | | |
| **Temas Principales Tratados** | | | | | |
| 1. Revisión y nueva redacción de los requisitos funcionales y no funcionales. 2. Rediseño de algunos aspectos de los prototipos de interfaz. 3. Rediseño de los casos de uso. 4. Rediseño del modelo de dominio. | | | | | |
| **Temas Secundarios Tratados** | | | | | |
| ---------------------------------------- | | | | | |
| **Próximos Pasos** | | | | | |
| 1. Terminar redacción de casos de uso. 2. Comenzar diagramas mediante software. 3. Comenzar diagramas de robustez.   Se estipula agenda una próxima reunión dentro del término de dos semanas. | | | | | |

Se adjunta el cuestionario realizado para la extracción de requisitos.

Cuestionario

* ¿Cuál es la idea principal para su negocio?

La idea principal de la empresa es crear una plataforma web que sirva de intermediario entre los negocios y los usuarios de manera que estos puedan realizar pedidos. Además se desea ofrecer un servicio de entrega para estos productos.

* En particular ¿Cuál cree usted que es el principal objetivo de negocio?

El foco principal del negocio es gestionar los pedidos realizados por los usuarios de manera que el producto llegue de la mejor manera a destino.

* En pocas palabras, ¿De qué manera pretende que los usuarios realicen los pedidos?

El objetivo, es que se presente una página con los negocios adheridos al e-business de manera que el usuario pueda elegir mediante sus preferencias, de acuerdo a distintos filtros, de donde solicitar el pedido.

* ¿Qué criterio de filtrado debería incluir la plataforma para la elección del Negocio?

Los negocios de deberían filtra tanto por su nombre como por la categoría de los platos que este contenga.

* Una vez que el usuarioescogeel local, ¿De qué manera pretende mostrar la información de los productos ofrecidos?

La solución óptima sería mostrar los platos agrupados por categoría, dentro de cada uno de ellos mostrar la información básica como por ejemplo el nombre, precio, tiempo de cocción, etc. Además ofrecer la posibilidad al usuario de realizar diferentes búsquedas de acuerdo a sus preferencias.

* Con respecto al Usuario ¿Cómo gestiona el pedido?

El Usuario puede agregar, modificar o eliminar platos. Además una vez efectuada esta elección, se podrá guardar el pedido para realizarlo en un futuro.

* ¿Es pertinente la forma en la que se muestra el detalle del pedido que se va realizando?

Preferentemente, lo ideal sería visualizar el detalle en la misma página en la que aparecen los productos mientras se van gestionando.

## Listado de requerimientos

### Requerimientos Funcionales

|  |  |
| --- | --- |
| Código | Descripción |
| RF01 | Se requiere que la solución se implemente a travez de una arquitectura Web, con posibilidad en un futuro de migrar a aplicación movil o de otro tipo. |
| RF02 | El Usuario puede elegir entre los diferentes negocios adheridos a la plataforma |
| RF03 | El *Usuario* puede realizar pedidos al negocio elegido |
| RF04 | El *Usuario* puede agregar, eliminar y modificar platos dentro del pedido en gestión |
| RF05 | El *Usuario* puede guardar un pedido para efectivizarlo en un futuro |

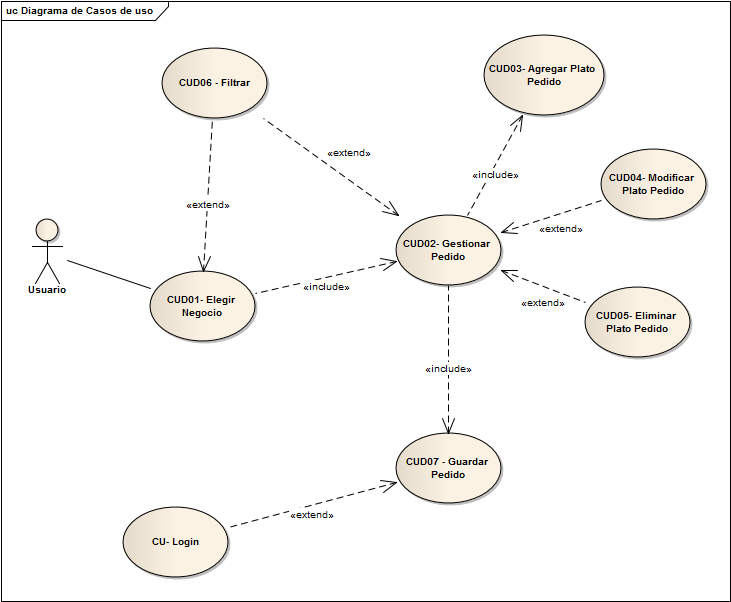
### Requerimientos No Funcionales

|  |  |
| --- | --- |
| Código | Descripción |
| RNF01 | Se pretende que el diseño del sitio web sea de tipo Responsive, es decir, que se adapte a las distintos tamaños de los dispositivos |
| RNF02 | El *usuario* tiene la posibilidad de filtrar los *negocios* de acuerdo a las siguientes preferencias: Categorías, KeyWords relacionadas al nombre del local |
| RNF03 | El menú del negocio debe mostrar los platos según sus categorías. |
| RNF04 | Una vez en el menú del Negocio el usuario puede filtrar los platos por: Categoría y Keywords |
| RNF05 | Se muestra dentro de la misma página el detalle del pedido a medida que el *usuario* lo va gestionando |

## Prototipos de interfaz

## Definición de Casos de Uso

### Diagrama de casos de uso



### Especificación de casos de uso

|  |  |  |
| --- | --- | --- |
|  | **Caso de Uso** | **Fecha: 19/05/2015** |
| **Código** | CUD01 | |
| **Nombre** | Elegir Negocio | |
| **Referencias** | RF01, RF02, RNF01, RNF02. | |
| **Autor** | Cordoba Pablo, Domínguez Jacobo | |
| **Revisores** | Ing Pablo Audoglio, Ing. Lucas Bracamonte | |
| **Versión** | 1.0 | |
| **Estado** | Validado | |
| **Descripción** | El actor podrá elegir un *Negocio,* dentro de un listado de adheridos al e-business con la posibilidad de filtrar de acuerdo a las diferentes categorías o al nombre del Local. | |
| **Actores** | Usuario | |
| **Pre-condición** | El actor debe ingresar dentro de la barra de navegación a la opción “Home” | |
| **Puntos de Extensión** | Paso 2: CUD06 – Filtrar | |
| **Curso básico** | | **Curso Alternativo** |
| 1. El sistema muestra la página “elegirNegocio” con un listado central que cuenta con los *Negocios* adheridos, con algunos datos como el nombre, horarios, dirección y un logotipo distintivo. Además, orientado sobre la izquierda de la pantalla, cuenta con filtros donde el actor podrá filtrar por keywords y categoría de los platos. Finalmente, del lado derecho de la pantalla, se presentan los últimos 5 pedidos del *Usuario* en el caso que esté logueado. | | * 1. El sistema no puede cargar la página debido a errores de diversas fuentes.   2. El sistema muestra un mensaje de error pertinente.   3. El actor recarga o cierra la página. |
| 1. Opcionalmente, el actor puede realizar una búsqueda para filtrar los resultados. CUD06 - Filtrar | |  |
| 1. El actor elige uno de los *Negocios*. | |  |
| 1. El sistema redirige al actor a la página “gestionarPedido”, invocando al CUD02 Gestionar Pedido. | | * 1. El sistema no puede hacer la redirección   2. El sistema muestra una página con un mensaje de error pertinente.   3. El actor recarga o cierra la página. |
| **Pos-condición** | El actor ha elegido un negocio. | |

|  |  |  |
| --- | --- | --- |
|  | **Caso de Uso** | **Fecha: 19/05/2015** |
| **Código** | CUD02 | |
| **Nombre** | Gestionar Pedido | |
| **Referencias** | RF01, RF03, RF04, RF05, RNF01, RNF03, RNF04, RNF05 | |
| **Autor** | Cordoba Pablo, Domínguez Jacobo | |
| **Revisores** | Ing Pablo Audoglio, Ing. Lucas Bracamonte | |
| **Versión** | 1.0 | |
| **Estado** | Validado | |
| **Descripción** | El actor podrá gestionar un pedido, agregando, modificando o quitando ítems, los cuales se presentan ordenados por categorías. | |
| **Actores** | Usuario | |
| **Pre-condición** | El actor debe elegir un negocio al cual desea realizarle un pedido | |
| **Puntos de Extensión** | Paso 2 CUD06 – Filtrar ; Paso 4.a CUD04- Modificar Plato Pedido; Paso 4.b CUD05- Eliminar Plato pedido | |
| **Curso básico** | | **Curso Alternativo** |
| 1. El sistema muestra la página “gestionarPedido” con un listado central de los platos que contiene el *Negocio,* ordenados por categorías. Orientado a la izquierda de la pantalla se muestra un input para realizar búsqueda de platos por keywords, debajo se presentan una serie de categorías para refinar la preferencia del actor. Del lado derecho de la pantalla, se presenta un pequeño panel con la información básica del negocio, como el nombre, la dirección y el teléfono. Debajo se presentará de forma dinámica el detalle de pedido en trámite y por último un botón para guardarlo. | | * 1. El sistema no puede cargar la página debido a errores de distintas fuentes.   2. El sistema muestra un mensaje con el error pertinente.   3. El actor recarga o cierra la página. |
| 1. El sistema verifica si se accedió a la página con un pedido existente y en el caso de que así sea, carga la información en el panel con el detalle de pedido en trámite | | * 1. El sistema no puede cargar la información del pedido.   2. El sistema informa al cliente con un mensaje.   3. El actor recarga o cierra la página. |
| 1. Opcionalmente, el actor puede realizar una búsqueda para filtrar los resultados. CUD06 - Filtrar | |  |
| 1. El actor hace click en el botón (+) para agregar un plato | |  |
| 1. El sistema invoca al CUD03- Agregar Plato Pedido | |  |
| 1. El actor podrá realizar las siguientes acciones dentro del panel del detalle de pedido 2. Modificar plato 3. Eliminar plato | |  |
| 1. El actor hace click en el botón guardar pedido | |  |
| 1. El sistema invoca al CUD07 – Guardar Pedido | |  |
| **Pos-condición** | El actor ha gestionado un pedido. | |

|  |  |  |
| --- | --- | --- |
|  | **Caso de Uso** | **Fecha: 19/05/2015** |
| **Código** | CUD03 | |
| **Nombre** | Agregar Plato Pedido | |
| **Referencias** | RF01, RF04, RNF01, RNF05 | |
| **Autor** | Cordoba Pablo, Domínguez Jacobo | |
| **Revisores** | Ing Pablo Audoglio, Ing. Lucas Bracamonte | |
| **Versión** | 1.0 | |
| **Estado** | Validado | |
| **Descripción** | El actor podrá elegir agregar un plato al pedido en trámite | |
| **Actores** | Usuario | |
| **Pre-condición** | El actor debe estar situado en la página gestionar pedidos. | |
| **Condición** | El actor debe hacer click sobre el ícono (+) dentro del ítem que desea agregar. | |
| **Curso básico** | | **Curso Alternativo** |
| 1. El sistema muestra el formulario modal “modalAgregarPlato” con el nombre del ítem escogido, un input para que el actor ingrese una *aclaración* acerca de la elaboración del plato, un combo para seleccionar la cantidad requerida, un campo que calcula el subtotal y por último un botón para aceptar la elección | | * 1. Debido a errores internos, no se puede mostrar el modal.   2. El sistema muestra un mensaje de error.   3. El actor acepta el mensaje y reintenta o cierra la página. |
| 1. El actor ingresa valores a los diferentes campos, la aclaración permite nulos y en el caso de la cantidad existe una mínima establecida por el *negocio* | |  |
| 1. El actor hace click en el botón Agregar | | * 1. El actor hace click en cerrar   2. El sistema envía al actor a la página realizar pedido, sin realizar modificaciones en el detalle. |
| 1. El sistema verifica que el detalle que se está agregando sea único, en el caso que no lo sea, se verifica la aclaración y se modifica la cantidad agregada | |  |
| 1. El sistema retorna a la página realizar pedido, actualizando el panel de detalle de pedido | |  |
| **Pos-condición** | El actor ha agregado un plato | |

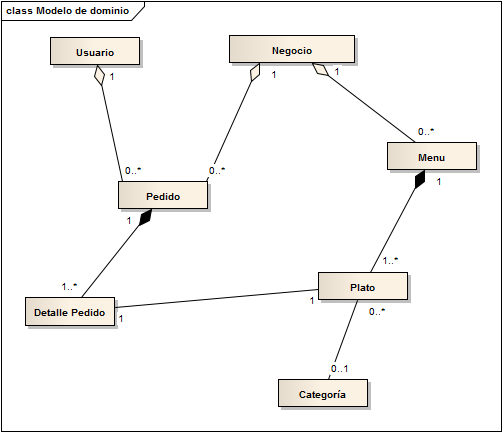
|  |  |  |
| --- | --- | --- |
|  | **Caso de Uso** | **Fecha: 19/05/2015** |
| **Código** | CUD04 | |
| **Nombre** | Modificar Plato Pedido | |
| **Referencias** | RF01, RF04, RNF01, RNF05 | |
| **Autor** | Cordoba Pablo, Domínguez Jacobo | |
| **Revisor** | Ing Pablo Audoglio | |
| **Versión** | 1.0 | |
| **Estado** | Validado | |
| **Descripción** | El actor podrá modificar la cantidad de un plato del pedido en trámite | |
| **Actores** | Usuario | |
| **Pre-condición** | El actor debe estar situado en la página realizar pedido, con un detalle que posea por lo menos un ítem | |
| **Condición** | Punto de extensión CUD02- Gestionar pedido. El actor modifica los valores de la cantidad del plato en el detalle. | |
| **Curso básico** | | **Curso Alternativo** |
| 1. El actor modifica la cantidad del plato dentro del detalle del pedido | |  |
| 1. El sistema actualiza los valores del subtotal perteneciente al ítem y el costo total del pedido | | * 1. El sistema no puede realizar la modificación   2. El sistema muestra una página con un mensaje de error.   3. El actor recarga o cierra la página. |
| **Pos-condición** | El actor ha modificado un plato | |

|  |  |  |
| --- | --- | --- |
|  | **Caso de Uso** | **Fecha: 19/05/2015** |
| **Código** | CUD05 | |
| **Nombre** | Eliminar Plato Pedido | |
| **Referencias** | RF01, RF04, RNF01, RNF05 | |
| **Autor** | Cordoba Pablo, Domínguez Jacobo | |
| **Revisores** | Ing Pablo Audoglio, Ing. Lucas Bracamonte | |
| **Versión** | 1.0 | |
| **Estado** | Validado | |
| **Descripción** | El actor podrá eliminar un plato del pedido en trámite | |
| **Actores** | Usuario | |
| **Pre-condición** | El actor debe estar situado en la página realizar pedido, con un detalle que posea por lo menos un plato. | |
| **Condición** | Punto de extensión CUD02- Gestionar Pedido. Paso 2.b. El actor elimina uno a varios ítems del detalle de pedido | |
| **Curso basico** | | **Curso Alternativo** |
| 1. El sistema cuenta con dos opciones para borrar ítems del detalle del pedido: 2. De a un ítem, haciendo click sobre el ícono (x) 3. Todos los ítems asociados al detalle mediante el botón (pulgar abajo) | |  |
| 1. El sistema muestra una mensaje, para confirmar la baja | | * 1. El sistema no puede mostrar el mensaje de alerta por un error   2. El sistema muestra en un popup el error ocurrido   3. El actor acepta el mensaje y reintenta o cierra la página. |
| 1. El actor acepta la alerta disparada por el sistema | | * 1. El actor no acepta la alerta disparada por el sistema.   2. El sistema no realiza la baja del detalle |
| 1. El sistema actualiza la lista de platos en el pedido, como también el total del mismo. | | * 1. El sistema no puede realizar la baja   2. El sistema muestra una página con un mensaje de error.   3. El actor recarga o cierra la página. |
| **Pos-condición** | El actor ha eliminado uno o todos los platos | |

|  |  |  |
| --- | --- | --- |
|  | **Caso de Uso** | **Fecha: 19/05/2015** |
| **Código** | CUD06 | |
| **Nombre** | Filtrar | |
| **Referencias** | RF01, RNF01, RNF02, RNF04 | |
| **Autor** | Cordoba Pablo, Domínguez Jacobo | |
| **Revisores** | Ing Pablo Audoglio, Ing. Lucas Bracamonte | |
| **Versión** | 1.0 | |
| **Estado** | Validado | |
| **Descripción** | El actor podrá eliminar un plato del pedido en trámite | |
| **Actores** | Usuario | |
| **Pre-condición** | El actor debe estar situado en la página realizar pedido, con un detalle que posea por lo menos un plato. | |
| **Condición** | Punto de extensión: Paso 2 CUD-01 Elegir negocio, Paso 2 CUD-02 Gestionar pedido. El actor realiza acciones sobre el panel de búsqueda | |
| **Curso básico** | | **Curso Alternativo** |
| 1. El actor ingresa valores en el campo del filtrado, o selecciona una de las categorías que muestra el sistema asociadas a ciertos criterios de disponibilidad. | |  |
| 1. El sistema realiza la consulta y devuelve de manera asincrónica una lista relacionada a las preferencias del actor. | | 2.1 El sistema no puede realizar la consulta por diversos motivos.  2.2 El sistema muestra una página con un mensaje de error pertinente.  2.3 El actor recarga o cierra la página. |
| **Pos-condición** | El actor ha filtrado según diferentes criterios. | |

|  |  |  |
| --- | --- | --- |
|  | **Caso de Uso** | **Fecha: 19/05/2015** |
| **Código** | CUD07 | |
| **Nombre** | Guardar Pedido | |
| **Referencias** | RF01, RF05, RNF01 | |
| **Autor** | Cordoba Pablo, Dominguez Jacobo | |
| **Revisores** | Ing. Pablo Audoglio, Ing. Lucas Bracamonte | |
| **Versión** | 1.0 | |
| **Estado** | A validar | |
| **Descripción** | El actor podrá guardar un pedido en el sistema. | |
| **Actores** | Usuario | |
| **Pre-condición** | El actor debe haber agregado al menos un plato al pedido. | |
| **Puntos de Extensión** | Paso 4 validar login | |
| **Condición** | El actor hace click en el botón “Guardar”. | |
| **Curso básico** | | **Curso Alternativo** |
| 1. El sistema verifica que exista al menos un plato en el pedido. | | 1.1 El sistema verifica que no hay ningún pedido, notifica al usuario. |
| 2. El sistema abre un formulario modal y le consulta al actor si realmente quiere guardar el pedido. | |  |
| 1. El actor hace click en el botón “Aceptar” del modal. | | * 1. El actor hace click en el botón cancelar.   2. El sistema retorna al actor a la página “gestionarPedido” |
| 1. El sistema valida que exista un usuario activo en sesión, en caso contrario lo redirige al login | |  |
| 3. El sistema persiste el pedido en la base de datos. | | 3.1 El sistema no puede persistir el pedido. 3.2 El sistema notifica al usuario acerca del error y le propone que lo intente de nuevo más tarde. |
| 4. El sistema envía un mensaje al usuario notificándolo de que se ha concretado el guardado del pedido. | |  |
| 5. El sistema redirige al actor a la página "elegirNegocio." | |  |
| **Pos-condición** | El actor ha guardado un pedido | |

## Modelo de dominio



## Revisión preliminar

### Matriz de calidad

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **Requerimientos funcionales** | | | | |
| **Caso de uso** | RF01 | RF02 | RF03 | RF04 | RF05 |
| **CUD01** | X | X |  |  |  |
| **CUD02** | X |  | X | X | X |
| **CUD03** | X |  |  | X |  |
| **CUD04** | X |  |  | X |  |
| **CUD05** | X |  |  | X |  |
| **CUD06** | X |  |  |  |  |
| **CUD07** | X |  |  |  |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **Requerimientos no funcionales** | | | | |
| **Caso de uso** | RNF01 | RNF02 | RNF03 | RNF04 | RNF05 |
| **CUD01** | X | X |  |  |  |
| **CUD02** | X |  | X | X | X |
| **CUD03** | X |  |  |  | X |
| **CUD04** | X |  |  |  | X |
| **CUD05** | X |  |  |  | X |
| **CUD06** |  | X |  | X |  |
| **CUD07** | X |  |  |  |  |

# Análisis de diseño preliminar

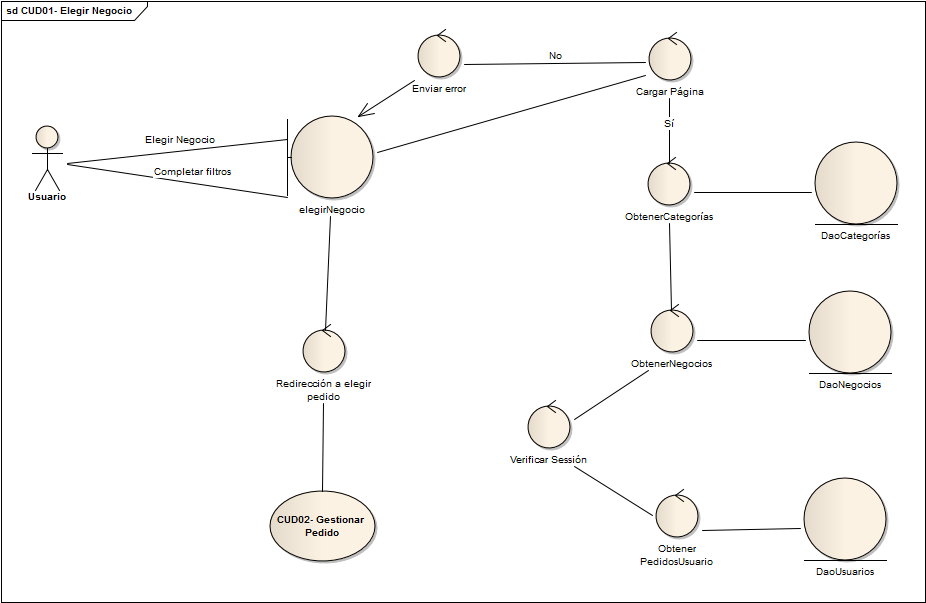
## Arquitectura técnica

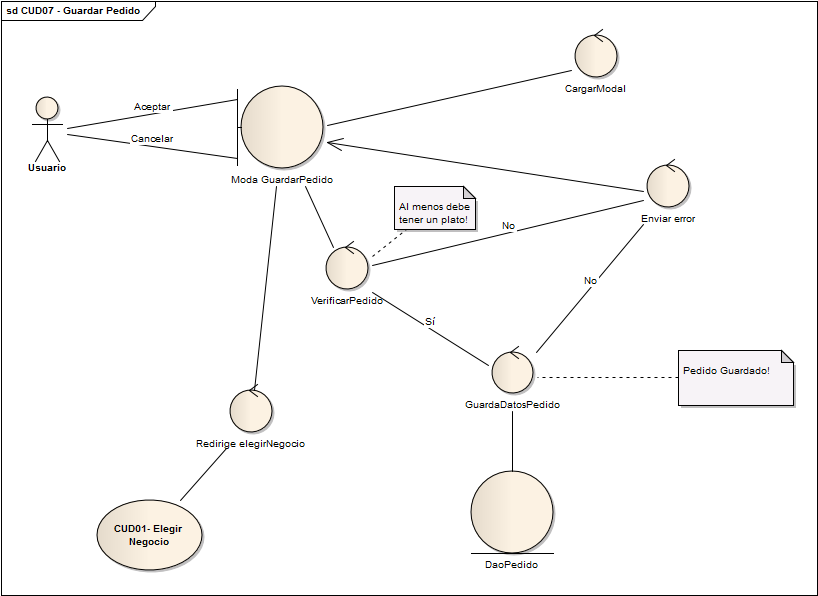
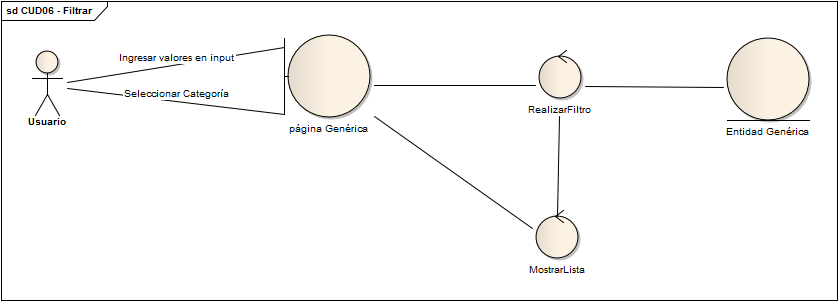
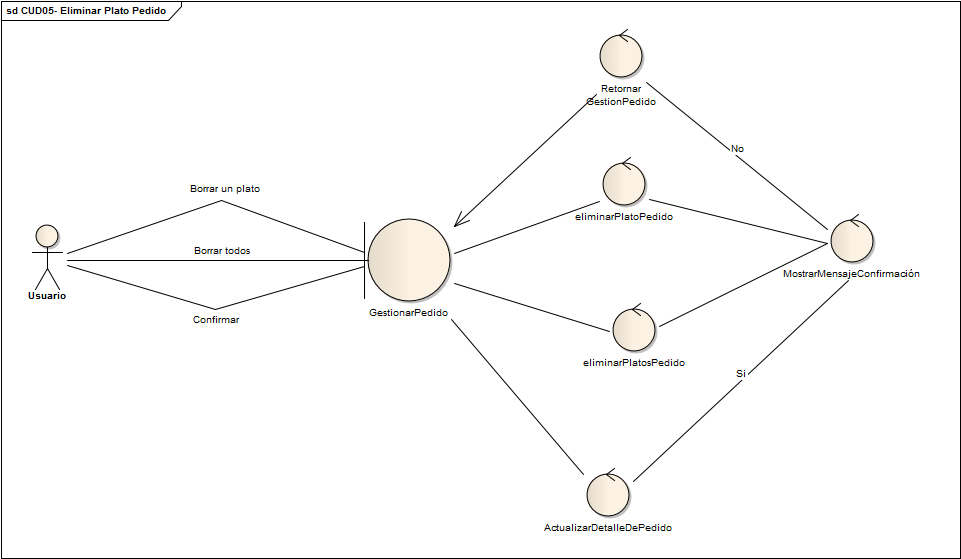
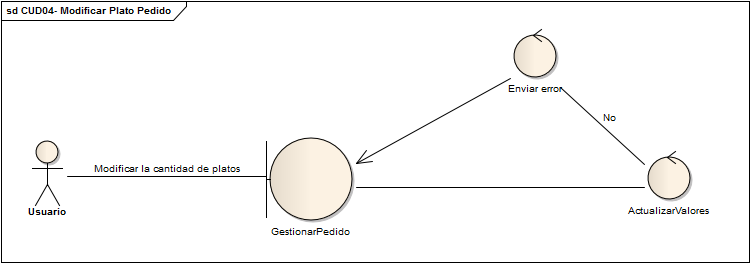
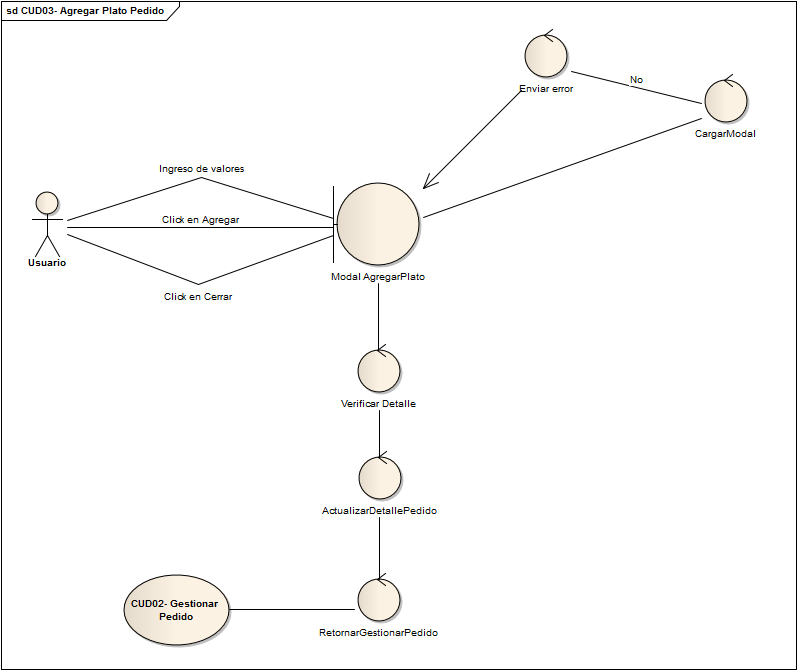
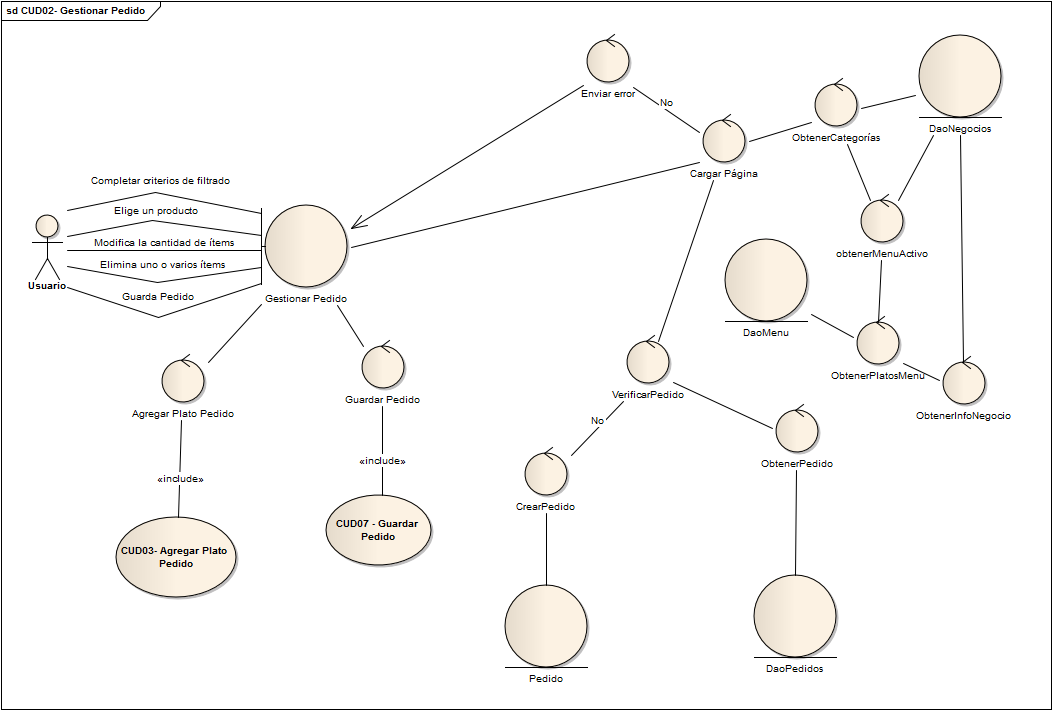
La arquitectura sería la de una aplicación web desarrollada en capas, haciendo uso de frameworks correspondientes para dicho uso y para la persistencia de datos (ORM). En cuanto a la base de datos, se utilizará una de tipo relacional.

### Tecnología a aplicar:

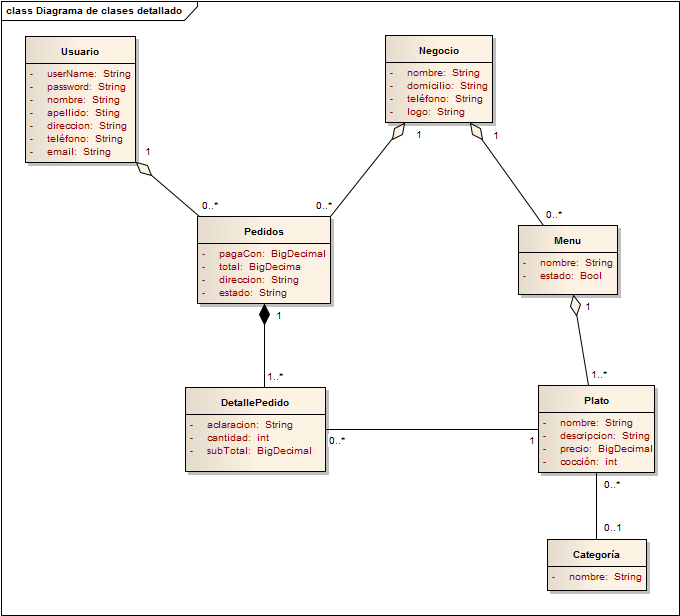
* **Base de datos:** 
  + Motor: MySql Standard Edition 5.6.
  + ORM: Hibernate Framework 4.
  + IDE: MySql Workbench 6.1.
* **Desarrollo de la aplicación:**
  + Lenguajes Backend: Java J2EE, XML, JSTL.
  + Lenguajes Frontend: HTML 5, CSS3, JavaScript.
  + Framework: Spring MVC 3. AngularJs
  + IDE: Intellij IDEA. Atom, Brackets.
* **Diseño Visual:**
  + Adobe Illustrator CS6
  + Adobe Photoshop CS6
* **Servidores:**
  + Apache Tomcat 8.
* **Gestion y documentación del proyecto:**
  + Maven 3.2.3
  + Git y repositorio remoto en Github.
  + Enterprise Arquitect.
  + Suit ofimática Microsoft Office.

## Análisis de robustez



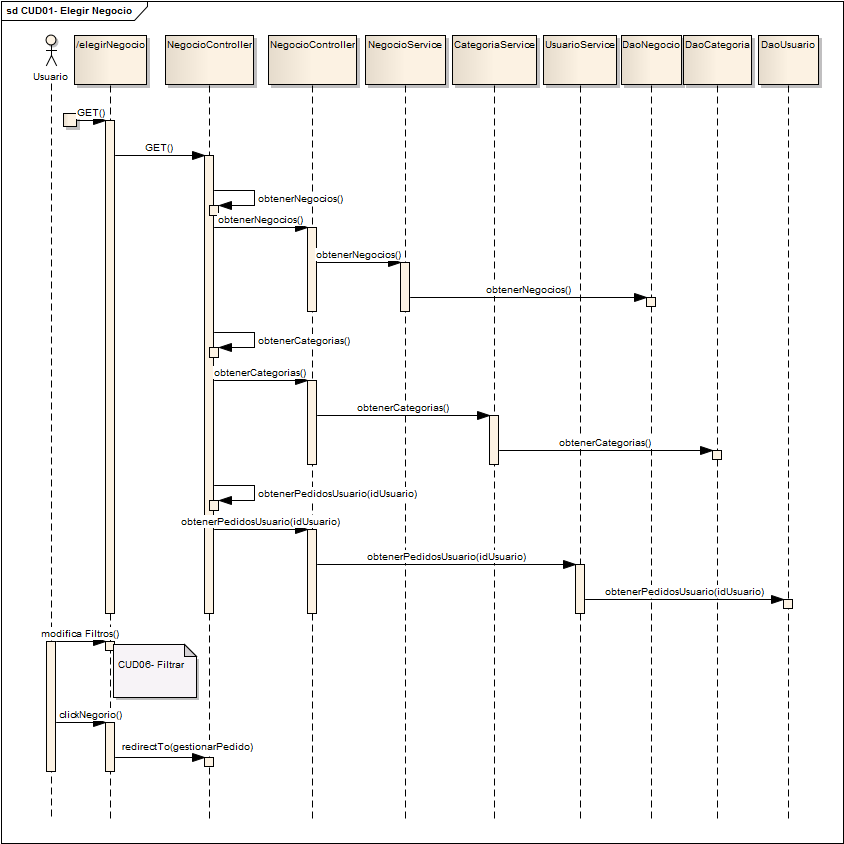


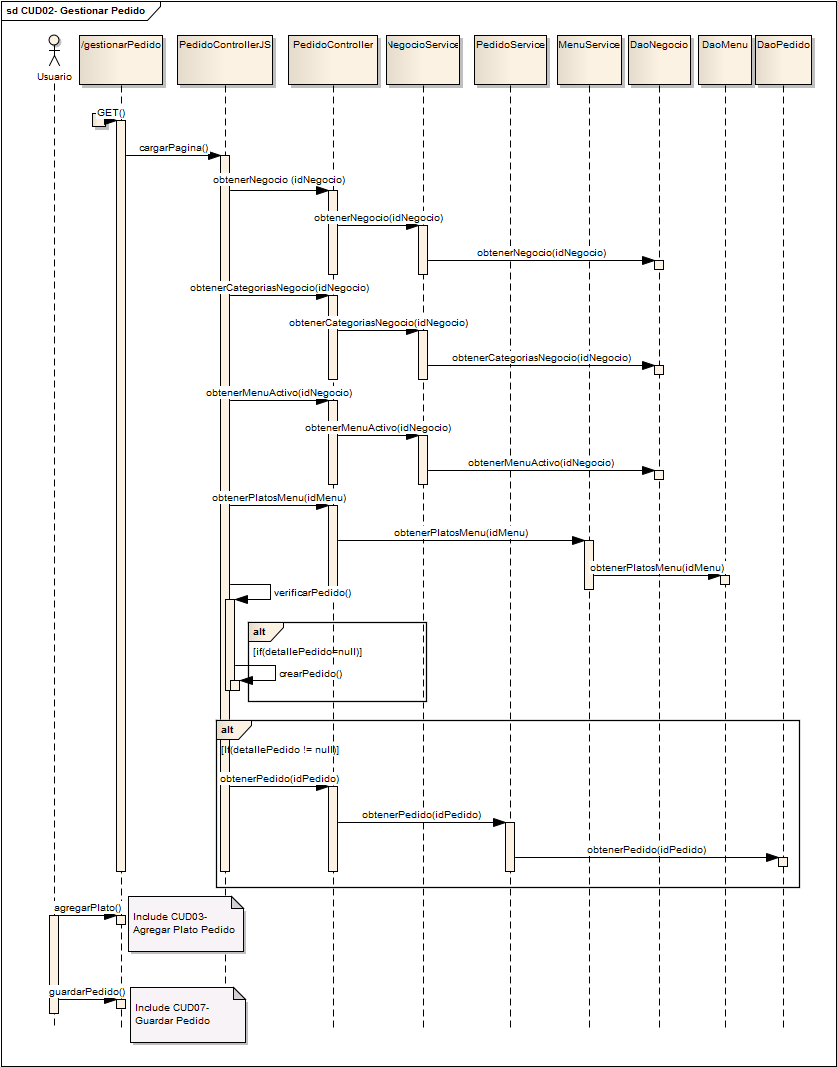
## Modelo de Dominio Actualizado

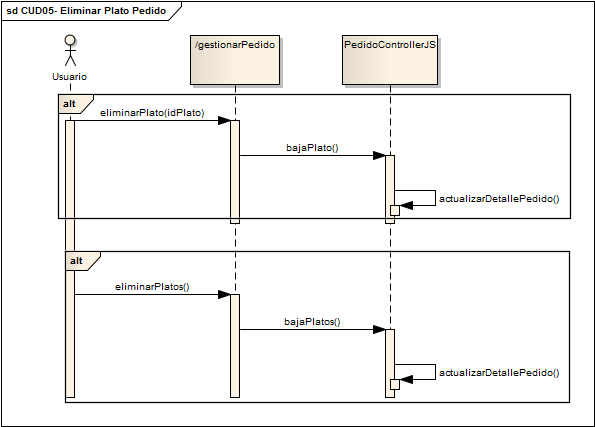


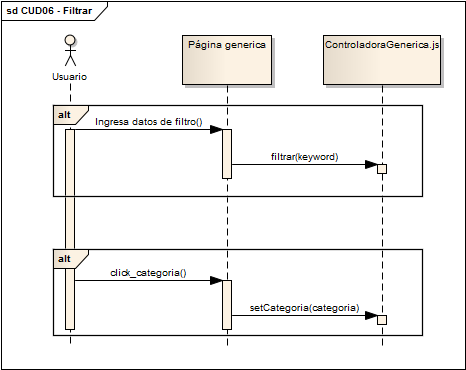
# Diseño Detallado

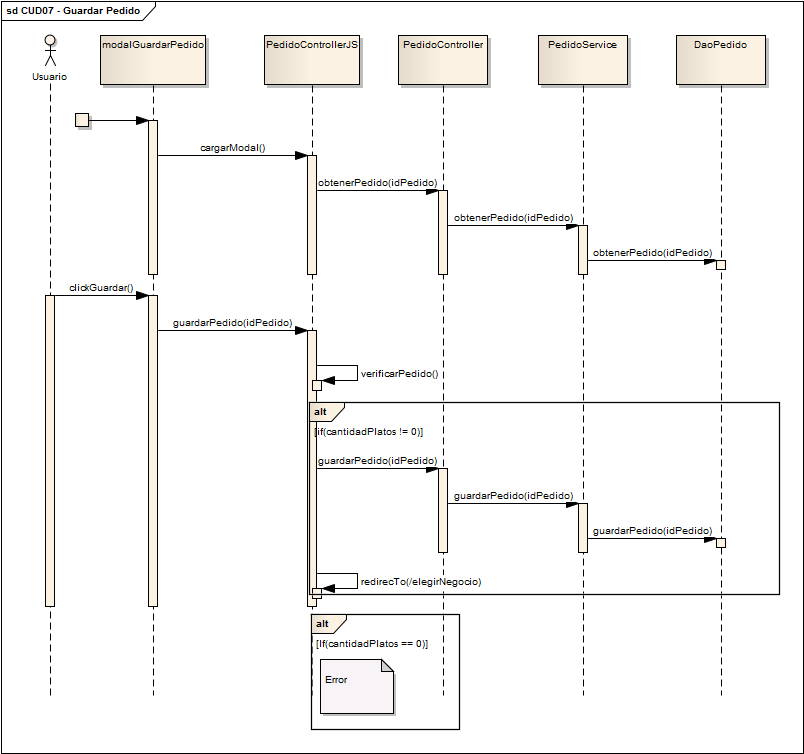
## Diagramas de Secuencia





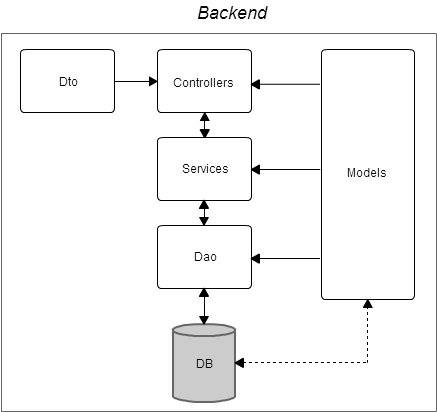




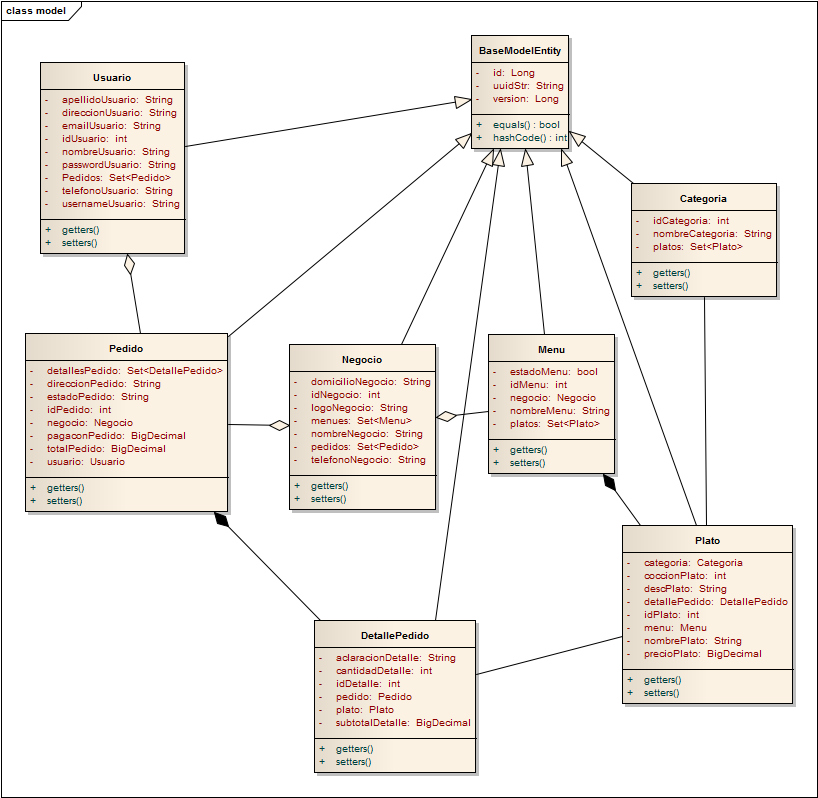


## Diagramas de Clases

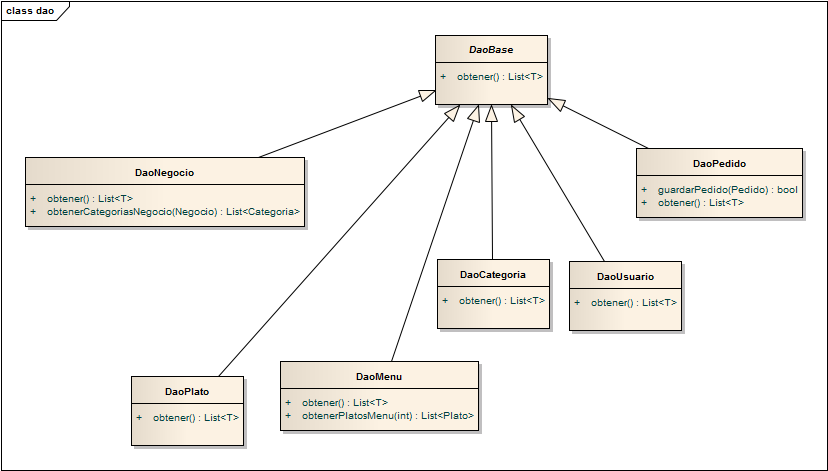
Para mayor compresión se han dividido todas las clases de software a utilizar basándose en las distintas capas que conforman a la arquitectura de la aplicación. La misma puede ser comprendida en base al siguiente esquema:



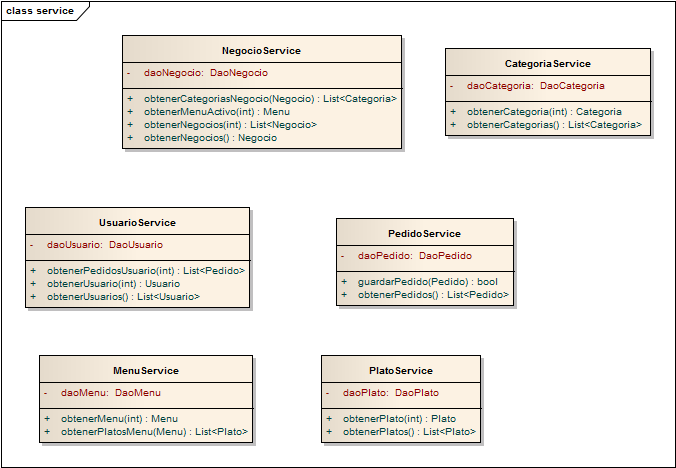
### Models



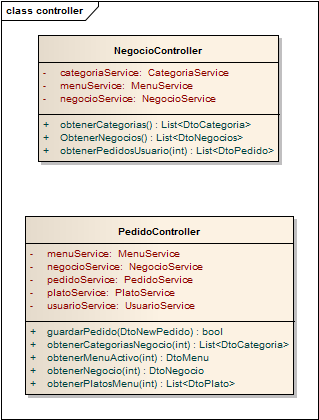
### Dao



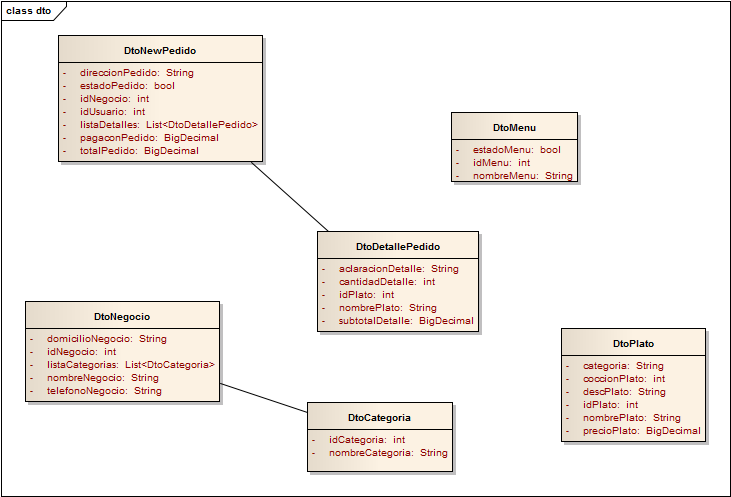
### Services



### Controllers



### Dto



# Análisis de requerimientos

## Minutas de reunión – Extracción de requerimientos

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **MINUTA DE REUNION** | | | | | |
| **Fecha:** | 24/08/2015 | **Hora Inicio:** | 15:30 | **Hora Fin:** | 17:40 |
| **Lugar:** | La reunión se desarrolló vía videoconferencia. | | | | |
| **Participantes** | | | | | |
| **Cliente** | | | **Empresa** | | |
| ------------------------------------------------ | | | 1. Córdoba Pablo 2. Dominguez Jacobo | | |
| **Objetivos de la Reunión** | | | | | |
| 1. Redacción de requisitos. 2. Creación de prototipos de interfaz. 3. Creación de diagrama de casos de uso. 4. Creación de modelo de dominio. | | | | | |
| **Temas Principales Tratados** | | | | | |
| 1. Redacción de los requisitos funcionales y no funcionales. 2. Diseño de los prototipos de interfaz. 3. Diseño del diagrama de casos de uso. 4. Diseño del modelo de dominio. | | | | | |
| **Temas Secundarios Tratados** | | | | | |
| 1. Estructura de la seguridad del sitio. | | | | | |
| **Próximos Pasos** | | | | | |
| 1. Redacción de descripción de casos de uso. 2. Comenzar diagramas de robustez   Se estipula agenda una próxima reunión dentro del término de dos días. | | | | | |

*Se adjunta cuestionario del cuál se desprenden los nuevos requisitos.*

**Cuestionario**

* Una vez que el usuario gestiona su pedido, ¿Cómo prosigue la operación?

El usuario tiene que tener la posibilidad de efectivizar el envío de su pedido, con el fin de cerrar la operación

* ¿Qué restricciones se le presentan al usuario a la hora de efectivizar el pedido?

Primeramente, el usuario que realiza un pedido en el e-commerce, tiene que tener una cuenta en la página de la cual necesitamos la dirección, el nombre, un e-mail y teléfono como datos esenciales. Otra manera puede ser que se registren mediante los usuarios de las redes sociales como Facebook, twitter entre otras.

* En el caso que el usuario quiera enviar su pedido a una dirección que no sea la posee en su cuenta actual, ¿Cómo le gustaría manejarlo?

El usuario debe poder ingresar una nueva dirección temporal para el pedido.

* Una vez que se termina la operación de envío, ¿Cuál le parece el próximo paso?

Es de suma importancia indicarle al usuario una estimación del tiempo en que tardará el pedido a llegar a sus manos. El paso a seguir sería informarle al usuario los datos del pedido mediante un correo electrónico, de forma que se afiance la confianza.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **MINUTA DE REUNION** | | | | | |
| **Fecha:** | 10/09/2015 | **Hora Inicio:** | 15:30 | **Hora Fin:** | 19:00 |
| **Lugar:** | La reunión se realizó en las instalaciones de la empresa creadora de la plataforma web. | | | | |
| **Participantes** | | | | | |
| **Cliente** | | | **Empresa** | | |
| ------------------------------------------------ | | | 1. Córdoba Pablo 2. Dominguez Jacobo | | |
| **Objetivos de la Reunión** | | | | | |
| 1. Pulida de los requisitos. 2. Creación de prototipos de interfaz vía software. 3. Modificación de los casos de uso. 4. Revisión y actualización del modelo de dominio. | | | | | |
| **Temas Principales Tratados** | | | | | |
| 1. Revisión y pulida de los requisitos. 2. Creación de prototipos de interfaz vía software. 3. Modificación de los casos de uso. 4. Actualización del modelo de dominio. | | | | | |
| **Temas Secundarios Tratados** | | | | | |
| ---------------------------------------- | | | | | |
| **Próximos Pasos** | | | | | |
| 1. Comenzar los diagramas de Robustez.   Se estipula agenda una próxima reunión dentro del término de dos semanas. | | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **MINUTA DE REUNION** | | | | | |
| **Fecha:** | 24/09/2015 | **Hora Inicio:** | 19:30 | **Hora Fin:** | 21:30 |
| **Lugar:** | Videoconferencia. | | | | |
| **Participantes** | | | | | |
| **Cliente** | | | **Empresa** | | |
| ------------------------------------------------ | | | 1. Córdoba Pablo 2. Dominguez Jacobo | | |
| **Objetivos de la Reunión** | | | | | |
| 1. Creación de diagramas de robustez. | | | | | |
| **Temas Principales Tratados** | | | | | |
| 1. Creación de diagramas de robustez. 2. Revisión general de coherencia en la documentación. | | | | | |
| **Temas Secundarios Tratados** | | | | | |
| ---------------------------------------- | | | | | |
| **Próximos Pasos** | | | | | |
| * + - 1. Comenzar la etapa de diseño preliminar.   Se estipula agenda una próxima reunión dentro del término de 3 semanas. | | | | | |

## Listado de requerimientos

### Requerimientos Funcionales

|  |  |
| --- | --- |
| Código | Descripción |
| RF06 | Se requiere que el usuario confirme el cierre del pedido |
| RF07 | El usuario tiene que estar logueado para efectivizar el pedido |
| RF08 | Se requiere que el usuario pueda modificar la dirección en la cuál se enviará el pedido |
| RF09 | Se requiere notificación al usuario del pedido realizado |

### Requerimientos No Funcionales

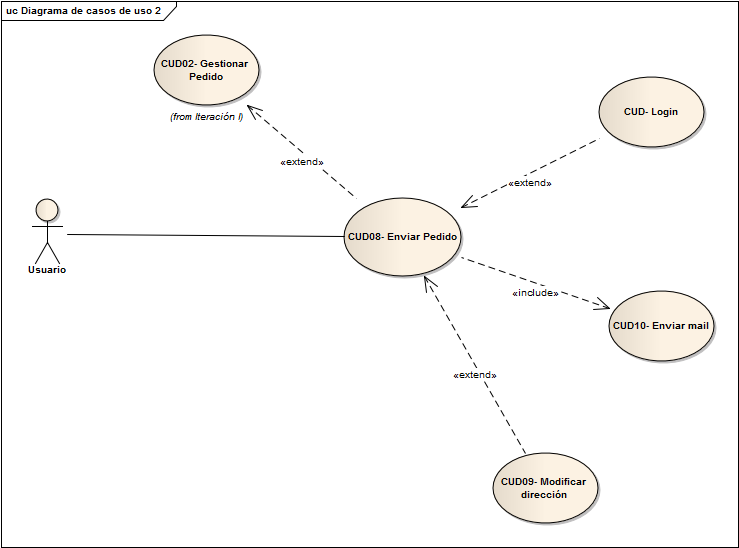
|  |  |
| --- | --- |
| Código | Descripción |
| RNF06 | El usuario puede loguearse usando servicios como Facebook, Twitter, Github |
| RNF07 | Se debe informar un tiempo estimado de entrega |
| RNF08 | Se pretende que se informe al usuario tanto por medio de e-mail como notificaciones push |

## Prototipos de Interfaz

Aca hay que imprimir del otro documento!

## Definición de Casos de Uso

### Diagrama de casos de uso

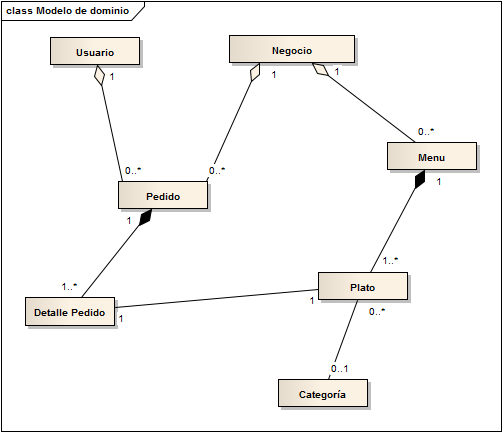


### Especificación de casos de uso

|  |  |  |
| --- | --- | --- |
|  | **Caso de Uso** | **Fecha: 10/09/2015** |
| **Código** | CUD08 | |
| **Nombre** | Enviar Pedido | |
| **Referencias** | RF06, RF07, RF08, RF09 | |
| **Autor** | Cordoba Pablo, Dominguez Jacobo | |
| **Revisores** | Ing. Pablo Audoglio, Ing. Lucas Bracamonte | |
| **Versión** | 1.0 | |
| **Estado** | A validar | |
| **Descripción** | El actor podrá enviar solicitud de envió para su pedido | |
| **Actores** | Usuario | |
| **Pre-condición** | El actor debe haber agregado al menos un plato al pedido y haber hecho click en el botón confirmar pedido | |
| **Puntos de Extensión** | Paso Alternativo 1.1 Login, Paso 4.a CUD10-Editar dirección | |
| **Condición** | El actor hace click en el botón “Enviar”. | |
| **Curso básico** | | **Curso Alternativo** |
| 1. El sistema verifica si existe un usuario logueado. | | 1.1 El sistema redirige al Login. |
| 2. El sistema carga la información de pedido en el panel de detalle y el perfil del usuario (Nombre, dirección) | |  |
| 3. El sistema presenta la página “Enviar pedido”, mostrando los datos mencionados en el paso anterior incluyendo un botón que permite editar la dirección, un campo para ingresar el valor a abonar y un campo de texto donde se actualiza el vuelto. | |  |
| 4. El actor ingresa los campos requeridos (actualizando el label de vuelto automáticamente) y hace click en el botón enviar para confirmar su solicitud | | 4.a El actor puede modificar la dirección de envío haciendo click en el botón editar. Punto de extensión CUD10- Editar dirección.  4.b El actor hace click en el botón volver para retornar con su elección del pedido |
| 6 El sistema verifica el correcto ingreso de datos | | 6.1 El sistema detecta error en el ingreso de datos y muestra una alerta del tipo informativa |
| 7 El sistema muestra un formulario modal preguntando si el usuario está seguro de solicitar el pedido | |  |
| 8 El actor acepta el formulario modal | | 8.1 El actor cancela el formulario modal, y por lo tanto la solicitud |
| 9 El sistema guarda el pedido en la base de datos | |  |
| 10 El sistema muestra que el pedido ha sido procesado | | 10.1 El sistema muestra una alerta, que no pudo procesar el pedido |
| 11.Se incluye el CUD10- Enviar Mail | |  |
| **Pos-condición** | | El actor ha enviado una solicitud de pedido. |

|  |  |  |  |
| --- | --- | --- | --- |
|  | | | |
|  | **Caso de Uso** | | **Fecha: 10/09/2015** |
| **Código** | CUD09 | | |
| **Nombre** | Editar dirección | | |
| **Referencias** | RF07, RF08 | | |
| **Autor** | Cordoba Pablo, Dominguez Jacobo | | |
| **Revisores** | Ing. Pablo Audoglio, Ing. Lucas Bracamonte | | |
| **Versión** | 1.0 | | |
| **Estado** | A validar | | |
| **Descripción** | El actor podrá modificar la dirección de solicitud de pedido. | | |
| **Actores** | Usuario | | |
| **Pre-condición** | El actor debe estar logueado | | |
| **Puntos de Extensión** |  | | |
| **Condición** | El actor hace click en el botón “Editar dirección”. | | |
| **Curso básico** | | | **Curso Alternativo** |
| 1. El sistema muestra un formulario modal, el cual cuenta con: un campo con la dirección actual del usuario (editable) y un mapa informativo señalizándola | | | 1.1 El sistema no puede recuperar la dirección actual del usuario y/o ubicar los datos en el mapa |
| 2 El actor edita el campo de dirección | | | 2.1 El actor hace click en el botón cancelar |
| 3 El sistema interpreta la dirección indicada y la señaliza en el mapa | | | 3.1 El sistema no puede indicar la dirección indicada en el mapa |
| 4 El actor hace click en el botón aceptar para confirmar la modificación | | | 4.1 El actor cancela el formulario modal |
| 5. El sistema guarda los nuevos valores indicados por el actor | | | 5.1 El sistema no puede persistir los datos proporcionados por el usuario |
| 6 El sistema muestra un mensaje informando el suceso de la operación | | | 6.1 El sistema muestra un mensaje de error al actor |
| **Pos-condición** | El actor ha modificado una dirección | | |
|  | **Caso de Uso** | **Fecha: 10/09/2015** | |
| **Código** | CUD10 | | |
| **Nombre** | Enviar Mail | | |
| **Referencias** | RF09 | | |
| **Autor** | Cordoba Pablo, Dominguez Jacobo | | |
| **Revisores** | Ing. Pablo Audoglio, Ing. Lucas Bracamonte | | |
| **Versión** | 1.0 | | |
| **Estado** | A validar | | |
| **Descripción** | El actor enviará un mail de confirmación al usuario | | |
| **Actor** | El sistema | | |
| **Pre-condición** | El usuario debe haber confirmado todos los pasos del envío de pedido | | |
| **Puntos de Extensión** |  | | |
| **Condición** | El usuario acepta el formulario modal de confirmación | | |
| **Curso básico** | | **Curso Alternativo** | |
| 1. El sistema recupera los datos del usuario para enviar el correo. | | 1.1 El sistema no puede recuperar el correo electrónico del usuario | |
| 2. A partir del evento que dispara el caso de uso, el sistema tomas los datos relevantes y los dispone en un template para enviárselo al usuario. | |  | |
| 3. El sistema envía el mail creado al usuario | | 3.1 El sistema tiene problemas en la entrega del mail  3.2 El sistema informa al usuario sobre el inconveniente | |
| **Pos-condición** | El actor ha enviado un mail informativo. | | |
|  |  | | |

## Modelo de dominio



## Revisión Preliminar

### Matriz de calidad

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Requisitos Funcionales** | | | |
| **Caso de uso** | RF06 | RF07 | RF08 | RF09 | |
| **CUD08** | X | X | X | X | |
| **CUD09** |  | X | X |  | |
| **CUD10** |  |  |  | X | |

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Requisitos No Funcionales** | | |
| **Caso de uso** | RNF06 | RNF07 | RNF08 | |
| **CUD08** | X |  |  | |
| **CUD09** |  | X | X | |
| **CUD10** |  | X | X | |

# Analisis de Diseño Preliminar

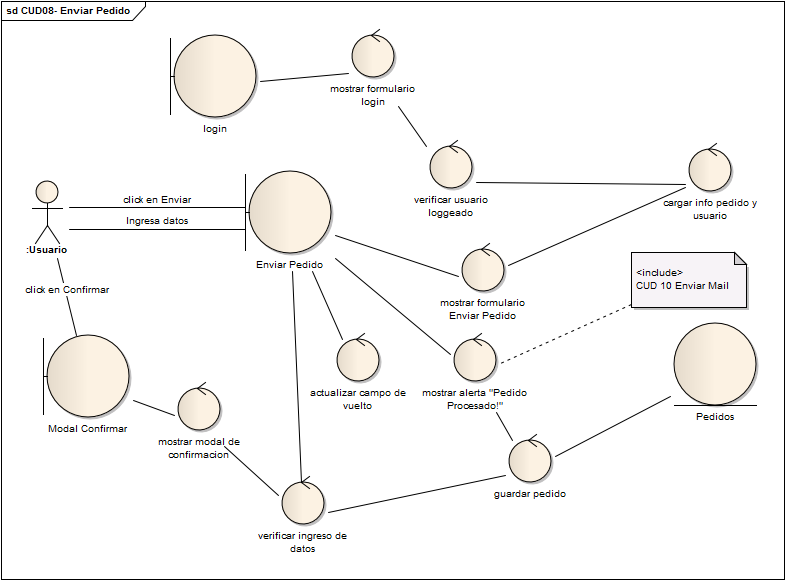
## Arquitectura técnica

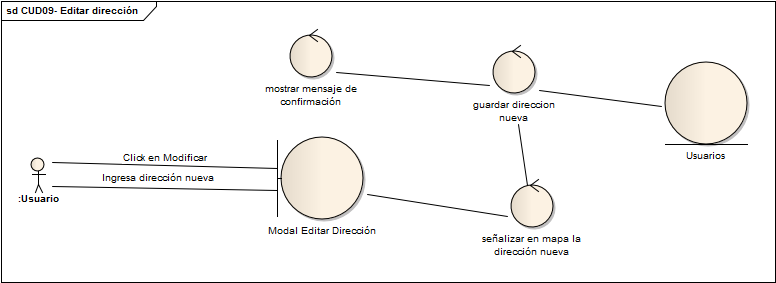
La arquitectura sería la de una aplicación web desarrollada en capas, haciendo uso de frameworks correspondientes para dicho uso y para la persistencia de datos (ORM). En cuanto a la base de datos, se utilizará una de tipo relacional.

### Tecnología a aplicar:

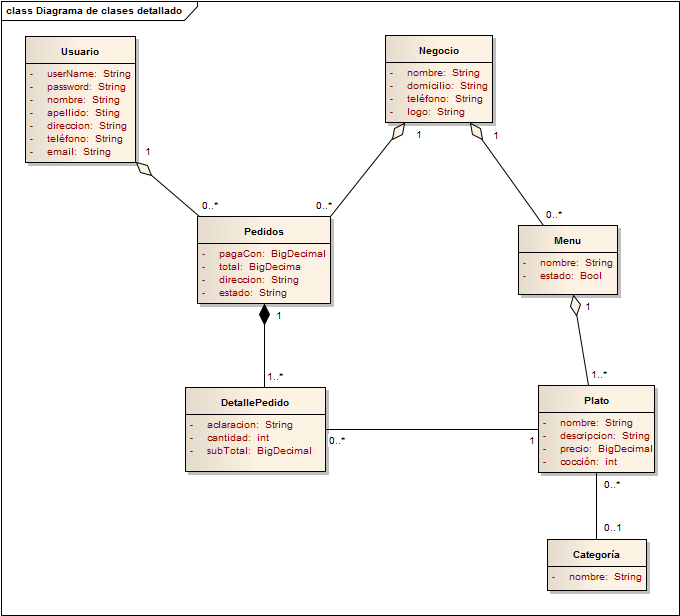
* **Base de datos:** 
  + Motor: MySql Standard Edition 5.6.
  + ORM: Hibernate Framework 4.
  + IDE: MySql Workbench 6.1.
* **Desarrollo de la aplicación:**
  + Lenguajes Backend: Java J2EE, XML, JSTL.
  + Lenguajes Frontend: HTML 5, CSS3, JavaScript.
  + Framework: Spring MVC 3. AngularJs
  + IDE: Intellij IDEA. Atom, Brackets.
* **Diseño Visual:**
  + Adobe Illustrator CS6
  + Adobe Photoshop CS6
* **Servidores:**
  + Apache Tomcat 8.
* **Gestion y documentación del proyecto:**
  + Maven 3.2.3
  + Git y repositorio remoto en Github.
  + Enterprise Arquitect.
  + Suite ofimática Microsoft Office.

## Análisis de robustez



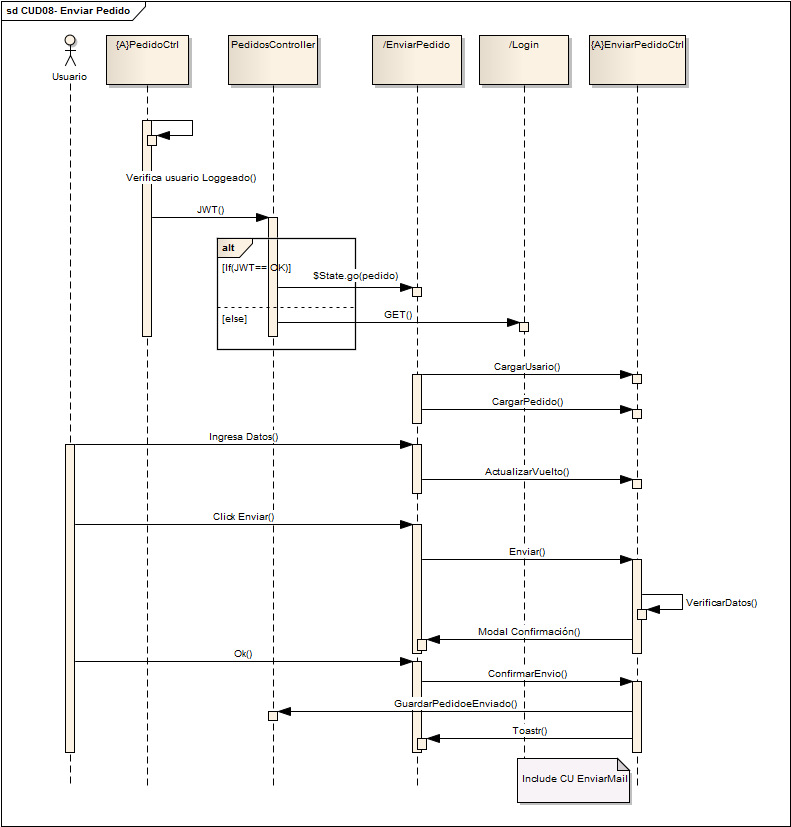


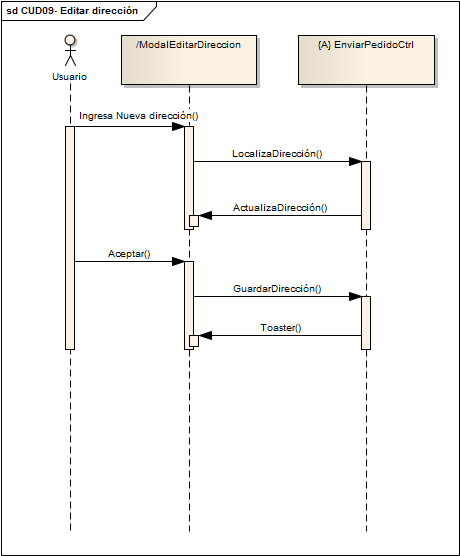
## Modelo de Dominio Actualizado



# Diseño Detallado

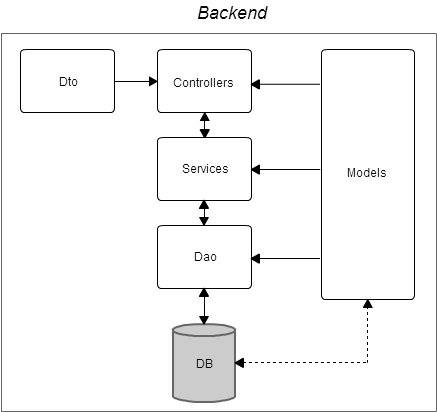
## Diagramas de Secuencia



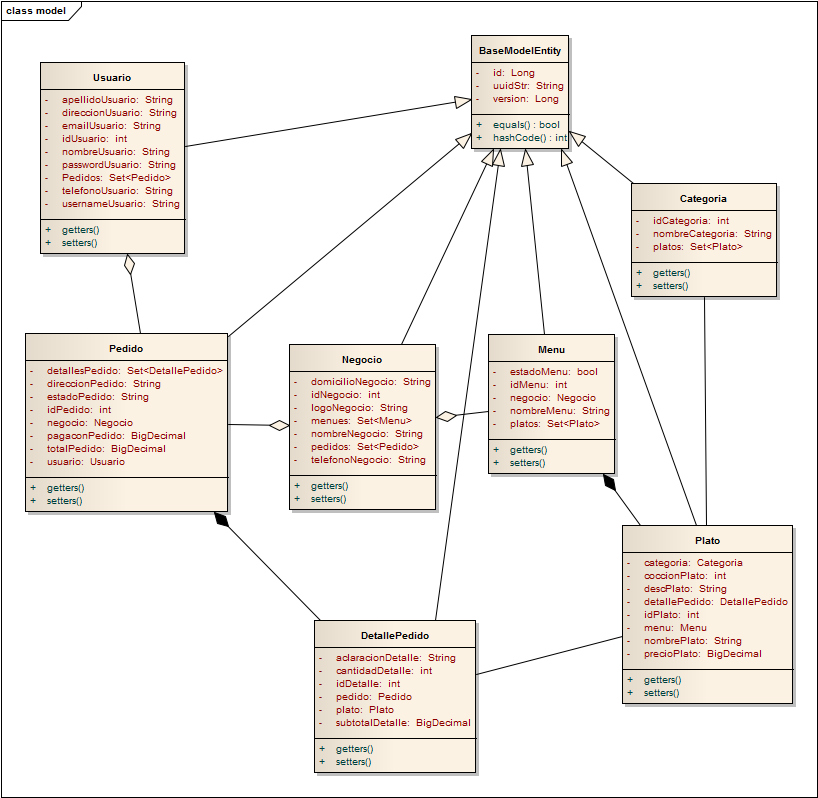


## Diagramas de Clases

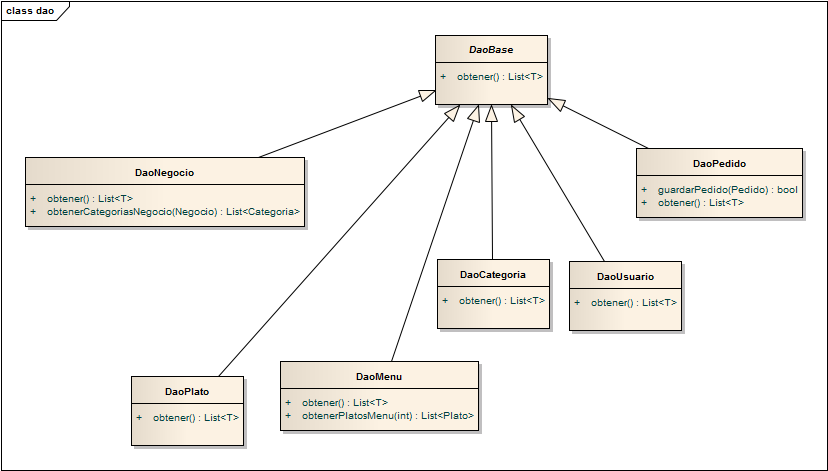
Para mayor compresión se han dividido todas las clases de software a utilizar basándose en las distintas capas que conforman a la arquitectura de la aplicación. La misma puede ser comprendida en base al siguiente esquema:



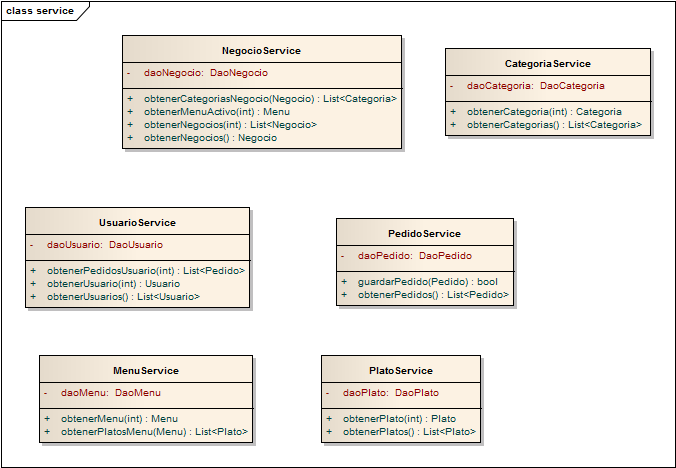
### Models



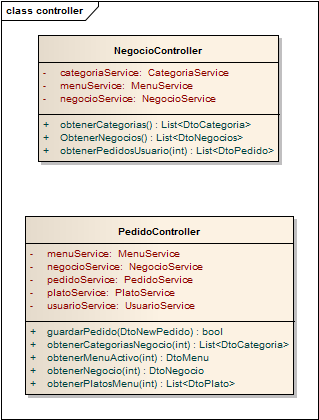
### Dao



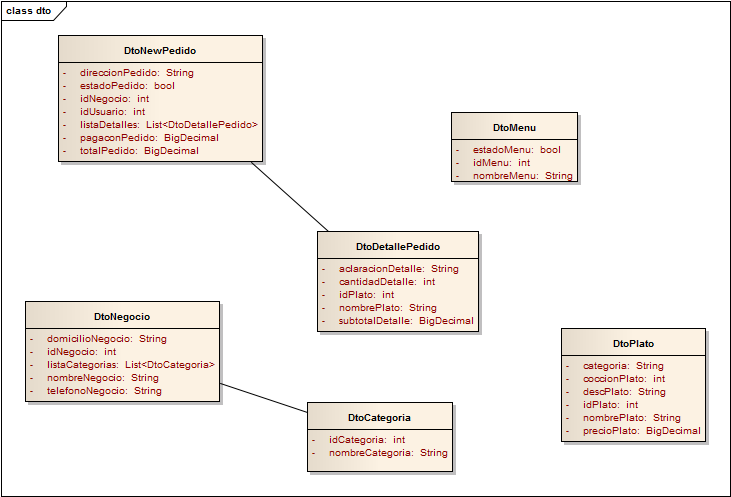
### Services



### Controllers



### Dto



# Anexo I: Procesos de Desarrollo de Software

## Introducción

Los modelos prescriptivos de proceso se propusieron originalmente para ordenar el caos del desarrollo del software. La historia ha indicado que estos modelos convencionales han traído consigo cierta cantidad de estructuras útiles para el trabajo en la ingeniería del software, y han proporcionado un camino a seguir razonablemente efectivo para los equipos de desarrollo. Sin embargo, el trabajo de ingeniería del software y el producto resultante aún permanecen en un estado intermedio entre la *estructuración* y el *borde del caos.* Esta definición permite que el proyecto pueda contener aspectos estructurales organizativos y metódicos y a su vez, otorga cierta libertad a la hora de aplicar la creatividad de los grupos de trabajos al proyecto y al producto. Los encargados de definir estos modelos de desarrollo son los ingenieros de software y sus gerentes los adaptan a sus necesidades y después los siguen.

Tener una guía para la creación de software es importante porque proporciona estabilidad, control y organización a una actividad que si no se controla puede volverse demasiado caótica. Algunas veces los modelos de proceso prescriptivo se han referido como “modelos rigurosos de proceso”, ya que a menudo incluyen las capacidades sugeridas por la IMCM[[1]](#footnote-1). Sin embargo, todos los modelos de proceso se pueden adaptar para usarlos de forma efectiva y en un proyecto de software específico.

El proceso conduce a un equipo de software a través de un conjunto de actividades del marco de trabajo que se organizan en un flujo de proceso, el cual puede ser línea, incremental o evolutivo. La terminología y los detalles de cada modelo de proceso difieren, pero las actividades genéricas del marco de trabajo permanecen razonablemente consistentes.

Desde el punto de vista de un ingeniero de software, los productos de trabajo son los programas, documentos y datos que se producen como consecuencia de las actividades y tareas que define el proceso.

En el presente anexo expondremos los principales modelos prescriptivos usados, con sus ventajas y desventajas.

### Modelo de Cascada

El modelo en cascada, también llamado *ciclo de vida clásico,* sugiere un enfoque sistemático, secuencial al desarrollo de software, que se inicia con la especificación de requerimientos del cliente y que continúa con la planeación, el modelado, la construcción y el despliegue para culminar en el soporte del software terminado. Es lineal y cada etapa empieza sí o sí cuando termina la anterior. Se trata del paradigma más antiguo para la ingeniería del software.



#### Ventajas:

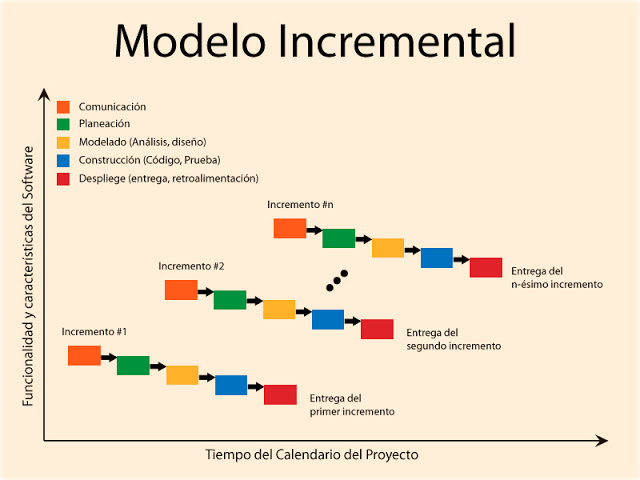
* Es un modelo sencillo y disciplinado.
* Es fácil aprender a utilizarlo y comprender su funcionamiento.
* Eficaz para proyectos pequeños en los cuales TODOS los requerimientos se definen en la primera etapa del proceso.

#### Desventajas:

* Es muy raro que los proyectos reales sigan el flujo secuencial que propone el modelo. Los cambios confunden mientras el equipo de proyecto actúa.
* Con frecuencia es difícil para el cliente establecer todos los requisitos de manera explícita. Se presentan dificultades al intentar lidiar con la incertidumbre inicial en muchos proyectos.
* El cliente debe tener mucha paciencia. Podrá revisar el producto una vez terminado. Un error puede ser muy costoso.

### Modelo Incremental

En este modelo, se entrega una serie de lanzamientos llamados incrementos, que proporcionan en forma progresiva más funcionalidad para los clientes a medida que se entrega cada uno de los incrementos. El primer incremento es un producto esencial que luego es modificado para incluir más funciones.



#### Ventajas:

* Se reduce el tiempo de desarrollo inicial, ya que la funcionalidad a entregar es muy básica.
* Provee un impacto ventajoso frente a los clientes, que es la entrega temprana de partes operativas del software.
* Reduce las desventajas del modelo cascada al ámbito de cada incremento.

#### Desventajas:

* No recomendable para casos de sistemas de tiempo real, de alto nivel de seguridad, de procesamiento distribuido y/o de alto índice de riesgos.
* Requiere de mucha planeación, tanto administrativa como técnica.
* Los errores en los requisitos se detectan muy tarde.

### Modelo DRA

Es un modelo incremental que resalta un ciclo de desarrollo corto. Se deben entender bien los requisitos y limitar el ámbito del proyecto para que el proceso tenga éxito. Cada gran función de un software puede ser encarada por un equipo DRA y después integrarlas y formar un todo.



#### Ventajas:

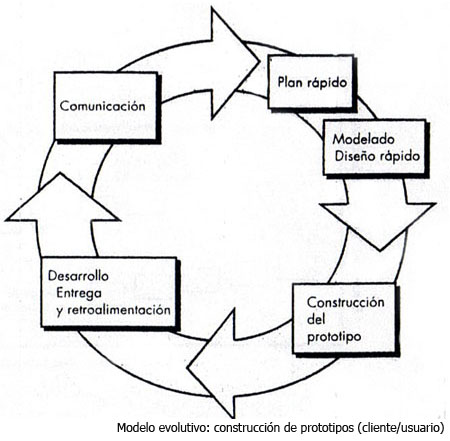
* Es muy rápido. Permite obtener un producto en 90 días.
* Permite trabajar varias personas a la vez.

#### Desventajas:

* Tiene inconvenientes para proyectos grandes, necesita suficientes recursos humanos para crear el número correcto de equipos.
* Si los desarrolladores y clientes no se comprenden con las actividades necesarias para completar el sistema, los proyectos fallarán.
* El DRA sería inapropiado cuando los riesgos técnicos son altos.

### Modelo de Prototipos

Es un modelo que puede implementarse conjuntamente con otros tipos de procesos de desarrollo. Se establecen requerimientos esenciales, se diseña e implementa una versión sencilla del software, se presenta al cliente y se toman en cuenta allí nuevos requisitos o especificaciones.



#### Ventajas:

* Ayuda en la comunicación entre el ingeniero de software y el cliente, aclarando requisitos o creando nuevos.
* Permite tener una idea del software “ideal” para el cliente.

#### Desventajas:

* El cliente cree que la creación de prototipos es efectivamente la creación del software final y no entiende que el mismo solo es un mecanismo para definir y aclarar requisitos.
* A veces el desarrollador aplica lenguajes, algoritmos o técnicas que son apropiadas a nivel prototipo para demostrar al cliente efectividad y rapidez. Luego a la hora de construir el software original, puede pasar que el desarrollador quiera aplicar dichos métodos que no son compatibles para garantizar la calidad del sistema.

### Modelo en Espiral

Las actividades de este modelo se conforman en una espiral, en la que cada bucle o iteración representa un conjunto de actividades. Las actividades no están fijadas a ninguna prioridad, sino que las siguientes se eligen en función del análisis de riesgo, comenzando por el bucle interior.



#### Ventajas:

* El análisis de riesgos se hace de forma explícita y clara, se reducen los mismos en cada iteración.
* Incorpora objetivos y mecanismos de calidad.
* Integra el desarrollo con el mantenimiento, incorporando mejoras y nuevos requerimientos sin romper con la metodología.

#### Desventajas:

* Genera mucho tiempo en el desarrollo del sistema.
* Es un modelo costoso.
* Requiere experiencia en la identificación de riesgos.
* Puede producir inconvenientes con el cliente debido a un contrato.

### Modelo Concurrente

El modelo de desarrollo concurrente, llamado algunas veces ingeniería concurrente, se representa en forma esquemática como una serie de actividades del marco de trabajo, acciones y tareas de la ingeniería del software y sus estados asociados. Todas las actividades existen de manera concurrente, pero se encuentran en diferentes estados. Se definen una serie de eventos que dispararán transiciones de estado a estado para cada una de las actividades, acciones o tareas.



#### Ventajas:

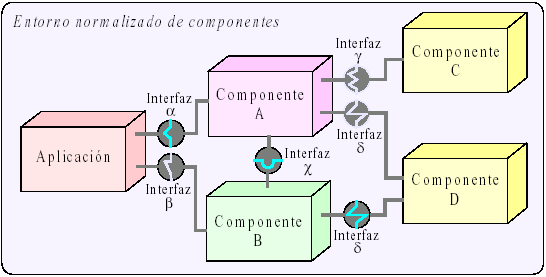
* Excelente para proyectos en los que se conforman grupos de trabajo independientes
* Proporciona una imagen exacta del estado actual del proyecto.

#### Desventajas:

* Si no se dan las condiciones señaladas no es aplicable.
* Si no existen grupos de trabajo no se puede trabajar con este método.

### Modelo de desarrollo basado en componentes

El modelo de desarrollo basado en componentes incorpora muchas de las características del modelo espiral. Es evolutivo por naturaleza y exige un enfoque interactivo para la creación del software. Sin embargo, el modelo de desarrollo basado en componentes configura aplicaciones desde componentes preparados de software (clases). Un componente es una pieza de código preelaborado que encapsula alguna funcionalidad expuesta a través de interfaces estándar. El modelo de desarrollo basado en componentes conduce a la reutilización del software, y la reutilización proporciona beneficios a los ingenieros de software.

****

#### Ventajas:

* Nos lleva a una maximización de código reutilizable.
* Permite que las pruebas sean ejecutadas probando cada uno de los componentes antes de probar el conjunto completo de componentes ensamblados.
* Simplifica el mantenimiento del sistema. Cuando existe un débil acoplamiento entre componentes, el desabollador es libre de actualizar y/o agregar componentes según sea necesario, sin afectar otras partes del sistema.
* Mayor calidad. Dado que un componente puede ser construido y luego mejorado continuamente por un experto u organización, la calidad de una aplicación basada en componentes mejorará con el tiempo.

#### Desventajas:

* Genera mucho tiempo en el desarrollo del sistema
* Modelo costoso
* Requiere mucha experiencia en la identificación de riesgos.
* Genera mucho trabajo adicional.

### Proceso Unificado:

Se trata de un modelo iterativo e incremental dividido en cuatro fases: INICIO, ELABORACIÓN, CONSTRUCCIÓN y TRANSICIÓN. Cada una de estas fases está a su vez, dividida en una serie de iteraciones. Estas ofrecen como resultado un incremento del producto desarrollado que añade funcionalidades al sistema. Estas iteraciones se dividen en disciplinas. Este modelo está dirigido por casos de uso, que permiten capturar requisitos funcionales. También el modelo está centrado en la arquitectura y enfocado en los riesgos.

El Proceso Unificado es un proceso de software genérico que puede ser utilizado para una gran cantidad de tipos de sistemas de software, para diferentes áreas de aplicación, diferentes tipos de organizaciones, diferentes niveles de competencia y diferentes tamaños de proyectos.

Provee un enfoque disciplinado en la asignación de tareas y resposabilidades dentro de una organización de desarrollo. Su meta es asegurar la producción de software de muy alta calidad que satisfaga las necesidades de los usuarios finales, dentro de un calendario y presupuesto predecible.

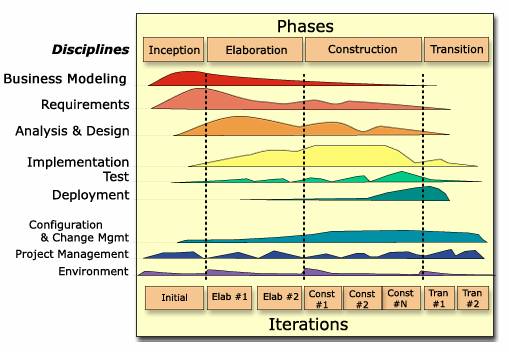
El Proceso Unificado tiene dos dimensiones:

Un eje horizontal que representa el tiempo y muestra los aspectos del ciclo de vida del proceso a lo largo de su desenvolvimiento

Un eje vertical que representa las disciplinas, las cuales agrupan actividades de una manera lógica de acuerdo a su naturaleza.

La primera dimensión representa el aspecto dinámico del proceso conforme se va desarrollando, se expresa en términos de fases, iteraciones e hitos (milestones).

La segunda dimensión representa el aspecto estático del proceso: cómo es descrito en términos de componentes del proceso, disciplinas, actividades, flujos de trabajo, artefactos y roles.



El Proceso Unificado se basa en componentes (component-based), lo que significa que el sistema en construcción está hecho de componentes de software interconectados por medio de interfaces bien definidas (well-defined interfaces).

El Proceso Unificado usa el Lenguaje de Modelado Unificado (UML) en la preparación de todos los planos del sistema. De hecho, UML es una parte integral del Proceso Unificado, fueron desarrollados a la par.

#### El Proceso Unificado es dirigido por casos de uso

Un sistema de software se crea para servir a sus usuarios. Por lo tanto, para construir un sistema exitoso se debe conocer qué es lo que quieren y necesitan los usuarios prospectos.

El término usuario se refiere no solamente a los usuarios humanos, sino a otros sistemas. En este contexto, el término usuario representa algo o alguien que interactúa con el sistema por desarrollar.

Un *caso de uso* es una pieza en la funcionalidad del sistema que le da al usuario un resultado de valor. Los casos de uso capturan los requerimientos funcionales. Todos los casos de uso juntos constituyen el *modelo de casos de uso* el cual describe la funcionalidad completa del sistema. Este modelo reemplaza la tradicional especificación funcional del sistema. Una especificación funcional tradicional se concentra en responder la pregunta: ¿Qué se supone que el sistema debe hacer? La estrategia de casos de uso puede ser definida agregando tres palabras al final de la pregunta: ¿por cada usuario? Estas tres palabras tienen una implicación importante, nos fuerzan a pensar en términos del valor a los usuarios y no solamente en términos de las funciones que sería bueno que tuviera. Sin embargo, los casos de uso no son solamente una herramienta para especificar los requerimientos del sistema, también dirigen su diseño, implementación y pruebas, esto es, dirigen el proceso de desarrollo.

Aún y cuando los casos de uso dirigen el proceso, no son elegidos de manera aislada. Son desarrollados a la par con la arquitectura del sistema, esto es, los casos de uso dirigen la arquitectura del sistema y la arquitectura del sistema influencia la elección de los casos de uso. Por lo tanto, al arquitectura del sistema y los casos de uso maduran conforme avanza el ciclo de vida.

#### El Proceso Unificado está centrado en la arquitectura

El papel del arquitecto de sistemas es similar en naturaleza al papel que el arquitecto desempeña en la construcción de edificios. El edificio se mira desde diferentes puntos de vista: estructura, servicios, plomería, electricidad, etc. Esto le permite al constructor ver una radiografía completa antes de empezar a construir. Similarmente, la arquitectura en un sistema de software es descrita como diferentes vistas del sistema que está siendo construido.

El concepto de arquitectura de software involucra los aspectos estáticos y dinámicos más significativos del sistema. La arquitectura surge de las necesidades de la empresa, tal y como las interpretan los usuarios y otros stakeholders, y tal y como están reflejadas en los casos de uso. Sin embargo, también está influenciada por muchos otros factores, tales como la plataforma de software en la que se ejecutará, la disponiblidad de componentes reutilizables, consideraciones de instalación, sistemas legados, requerimientos no funcionales (ej. desempeño, confiabilidad). La arquitectura es la vista del diseño completo con las características más importantes hechas más visibles y dejando los detalles de lado. Ya que lo importante depende en parte del criterio, el cual a su vez viene con la experiencia, el valor de la arquitectura depende del personal asignado a esta tarea. Sin embargo, el proceso ayuda al arquitecto a enfocarse en las metas correctas, tales como claridad (understandability), flexibilidad en los cambios futuros (resilience) y reúso.

¿Cómo se relacionan los casos de uso con la arquitectura? Cada producto tiene función y forma. Uno sólo de los dos no es suficiente. Estas dos fuerzas deben estar balanceadas para obtener un producto exitoso. En este caso función corresponde a los casos de uso y forma a la arquitectura. Existe la necesidad de intercalar entre casos de uso y arquitectura. Es un problema del “huevo y la gallina”. Por una parte, los casos de uso deben, cuando son realizados, acomodarse en la arquitectura. Por otra parte, la arquitectura debe proveer espacio para la realización de todos los casos de uso, hoy y en el futuro. En la realidad, ambos arquitectura y casos de uso deben evolucionar en paralelo.

#### El Proceso Unificado es Iterativo e Incremental

Desarrollar un producto de software comercial es una tarea enorme que puede continuar por varios meses o años. Es práctico dividir el trabajo en pequeños pedazos o mini-proyectos. Cada mini-proyecto es una iteración que finaliza en un incremento. Las iteraciones se refieren a pasos en el flujo de trabajo, los incrementos se refieren a crecimiento en el producto. Para ser más efectivo, las iteraciones deben estar controladas, esto es, deben ser seleccionadas y llevadas a cabo de una manera planeada.

Los desarrolladores basan su selección de qué van a implementar en una iteración en dos factores. Primero, la iteración trata con un grupo de casos de uso que en conjunto extienden la usabilidad del producto. Segundo, la iteración trata con los riesgos más importantes. Las iteraciones sucesivas construyen los artefactos del desarrollo a partir del estado en el que fueron dejados en la iteración anterior.

En cada iteración, los desarrolladores identifican y especifican los casos de uso relevantes, crean el diseño usando la arquitectura como guía, implementan el diseño en componentes y verifican que los componentes satisfacen los casos de uso. Si una iteración cumple sus metas – y usualmente lo hace – el desarrollo continúa con la siguiente iteración. Cuando la iteración no cumple con sus metas, los desarrolladores deben revisar sus decisiones previas y probar un nuevo enfoque.

#### Ventajas:

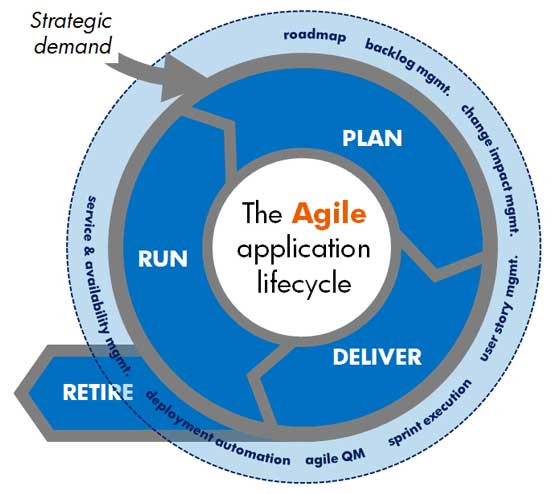
* Identificación y reducción de riesgos.
* Utilización de todo el equipo de desarrollo en todo el proceso.
* Interacción permanente con el cliente.
* Inclusión de nuevos requerimientos.
* Mayor control del proceso.

#### Desventajas:

* Es necesario “aterrizar” los conceptos, lo cual puede resultar un poco difícil para quien no tenga experiencia en el uso de procesos de ingeniería de software.
* Se basa demasiado en la documentación.
* Es necesario que todos los miembros de los equipos de trabajo conozcan UML.

### Desarrollo ágil

Combina una filosofía y un conjunto de directrices de desarrollo; busca la satisfacción del cliente y la entrega temprana de software incremental; equipos de proyecto pequeños y con alta motivación; métodos informales; mínimo de productos de trabajo de la ingeniería de software; simplicidad general del desarrollo. Las directrices de desarrollo resaltan la entrega sobre el análisis y el diseño, y la comunicación activa y continua entre los desarrolladores y los clientes. Rasgos claves entre el equipo de trabajo: competencia, enfoque común, colaboración, habilidad para la toma de decisiones, capacidad de resolución de problemas confusos, confianza y respeto mutuo, organización propia.



#### Ventajas:

* Intenta superar las debilidades advertidas y reales en la ingeniería del software convencional.
* Es un modelo adaptable, tiene mucha importancia la retroalimentación con el cliente para lograr un modelo incremental.

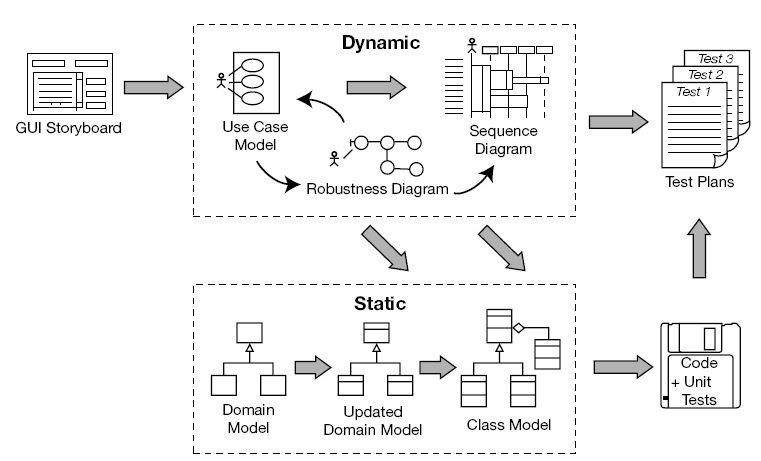
#### Desventajas:

* Es imposible aplicar todos los beneficios a todos los proyectos, productos, personas y situaciones.

### ICONIX

Es una metodología pesada-ligera de Desarrollo de Software que se halla a medio camino entre un RUP (Rational Unified Process) y un XP (eXtreme Programming).

ICONIX deriva directamente del RUP y su fundamento es el hecho de que un 80% de los casos pueden ser resueltos tan solo con un uso del 20% del UML, con lo cual se simplifica muchísimo el proceso sin perder documentación al dejar solo aquello que es necesario. Esto implica un uso dinámico del UML de tal forma que siempre se pueden utilizar otros diagramas además de los ya estipulados si se cree conveniente. ICONIX se guía a través de casos de uso y sigue un ciclo de vida iterativo e incremental. El objetivo es que a partir de los casos de uso se obtenga el sistema final.



La metodología está formada por cuatro fases principales.

#### *Fase 1: Análisis de requisitos*

Dentro de esta fase se realizan las siguientes tareas:

* Modelo del dominio
* Elaboración rápida de prototipos
* Modelo de casos de usos

#### *Fase 2: Análisis y diseño preliminar*

Dentro de esta fase se realizan las siguientes tareas:

* Descripción de los casos de uso
* Diagramas de robustez

#### *Fase 3: Diseño*

* Dentro de esta fase se realiza la siguiente tarea:
* Diagramas de secuencia

#### *Fase 4: Implementación*

* Dentro de esta fase se realiza la siguiente tarea:
* Escribir y generar código

#### Ventajas:

* Incluye un juego mínimo de diagramas de UML y algunas valiosas técnicas que se toman de los casos del uso para codificar rápida y eficazmente.
* El enfoque es flexible y abierto; siempre se puede seleccionar de los otros aspectos del UML para complementar los materiales básicos.
* ICONIX es un modelo pequeño que no descuida el análisis y el diseño. Usa un análisis de robustez que reduce la ambigüedad al describir los casos de uso.

#### Desventajas:

* No puede ser usado para proyectos grandes.
* Necesita información rápida y puntual de los requisitos, el diseño y las estimaciones.
* Se debe conocer los diagramas UML.

## Conclusión:

En este proyecto se utilizó la metodología ICONIX debido a su versatilidad a la hora del modelado ya que no es necesaria tanta documentación redundante, lo que conlleva a un desarrollo e implementación mucho más rápido y acorde a los requisitos expresados por el cliente. Todos estos aspectos son fundamentales dado el corto tiempo que ofrece la cursada de la cátedra. De todas formas, con esta metodología es posible completar un software robusto y de calidad en un tiempo acotado. Otro de los aspectos que favorecen a la elección de este modelo de desarrollo es que es uno de los más usados actualmente por las Software Factories argentinas.

# Anexo II: Gestión de Riesgos

## Introducción:

Para realizar este análisis en primer lugar nos preguntamos lo siguiente:

* ¿Qué riesgos causarían que el proyecto de software salga mal?
* ¿Cómo impactará el proyecto sobre los intereses del cliente?
* ¿Qué personal usa el sistema, está capacitado para hacerlo?
* ¿Qué mantenimiento requerirá el sistema?
* ¿En qué posición se encuentra nuestro cliente frente al mercado?
* ¿Se identificaron correctamente los requisitos por parte del cliente?¿Fueron entendidos por los analistas?
* ¿Los requisitos del proyecto son estables?
* ¿El equipo del proyecto tiene experiencia con el sistema a implementar?

Entre otras preguntas y supuestos que nos ayudan a identificar las dificultades que se nos plantearán.

A partir de la información obtenida de este estudio identificaremos los riesgos potenciales que pueden ocurrir o no en la empresa y así, podremos identificar los mismos, evaluarlos y estimar el impacto que causará para poder crear planes de contingencia en el caso que estos se presenten.

A continuación se presentarán en una tabla los riesgos potenciales para el proyecto.

|  |  |  |
| --- | --- | --- |
| **Tipo de Riesgo** | **Categoría** | **Riesgos Potenciales** |
| Requerimientos | Proyecto/Producto | El cliente no tiene en claro los requerimientos que pretende para el sistema |
| Requerimientos | Proyecto/Producto | Modificaciones de los requerimientos en una etapa temprana del proyecto |
| Personas | Proyecto | Poca experiencia a la hora del análisis y desarrollo |
| Estimación | Proyecto/Producto | Mal cálculo del planeamiento del proyecto en general debido a la falta de experiencia |
| Software | Producto | Fallas en las herramientas de desarrollo |
| Software | Producto | Licencias básicas que no cumplan con los requerimientos de los desarrolladores |
| Legal | Proyecto/Producto | Licencias de las herramientas |
| Personas | Proyecto | Ausentismo de personal, por motivos de salud o personales |
| Personas | Proyecto | Clima laboral insatisfactorio y/o falta de comunicación en el equipo |
| Personas | Proyecto | Diferencia de opiniones y objetivos |
| Tecnológico | Proyecto/Producto | Falta de hardware por parte del cliente |
| Financiero | Negocio | Falta de recursos económicos, recorte de presupuestos |
| Personas/Organizacional | Proyecto | Baja de uno de los integrantes del proyecto |
| Personas | Proyecto | El cliente no desea continuar con el proyecto |
| Negocio | Negocio | Cambio legales que intervengan en la integridad del proyecto |
| Personas | Proyecto | Usuarios finales oponiéndose a la implementación del proyecto |
| Personas | Proyecto | Falta de capacitación de los usuarios |
| Producto | Negocio | Competencia lanza una opción a nuestro sistema |
| Económico | Negocio | Variación de la moneda |
| Software | Producto | Pérdida de datos |
| Requerimientos | Producto | Expectativas muy elevadas por parte del cliente |
| Económico/Requerimientos | Proyecto/Producto | Ajuste de presupuesto por nuevos requerimientos |
| Organizacional | Negocio | Cambios organizacionales del lado del cliente |
| Organizacional | Negocio | Falta de conocimiento del negocio a tratar |
| Organizacional/Económico | Negocio | Cambio del sponsor |

## Lista de priorización de riesgos

|  |  |  |  |
| --- | --- | --- | --- |
| **Nº** | **Riesgo** | **Probabilidad** | **Efectos** |
| 1 | El cliente no desea continuar con el proyecto | 2-Bajo | 1-Catastrófico |
| 2 | Baja de uno de los integrantes del proyecto | 2-Bajo | 1-Catastrófico |
| 3 | Cambio legales que intervengan en la integridad del proyecto | 2-Bajo | 1-Catastrófico |
| 4 | Usuarios finales oponiéndose a la implementación del proyecto | 3-Moderado | 2-Serio |
| 5 | Pérdida de datos | 3-Moderado | 2-Serio |
| 6 | Cambios organizacionales del lado del cliente | 2-Bajo | 2-Serio |
| 7 | Mal cálculo del planeamiento del proyecto en general debido a la falta de experiencia | 3-Moderado | 2-Serio |
| 8 | Clima laboral insatisfactorio y/o falta de comunicación en el equipo | 2-Bajo | 2-Serio |
| 9 | Fallas en las herramientas de desarrollo | 2-Bajo | 2-Serio |
| 10 | Ausentismo de personal, por motivos de salud o personales | 3-Moderado | 3-Tolerable |
| 11 | Cambio del sponsor | 2-Bajo | 3-Tolerable |
| 12 | Falta de recursos económicos, recorte de presupuestos | 3-Moderado | 3-Tolerable |
| 13 | Licencias básicas que no cumplan con los requerimientos de los desarrolladores | 2-Bajo | 3-Tolerable |
| 14 | El cliente no tiene en claro los requerimientos que pretende para el sistema | 3-Moderado | 3-Tolerable |
| 15 | Modificaciones de los requerimientos en una etapa temprana del proyecto | 3-Moderado | 3-Tolerable |
| 16 | Poca experiencia a la hora del análisis y desarrollo | 3-Moderado | 3-Tolerable |
| 17 | Diferencia de opiniones y objetivos | 2-Bajo | 3-Tolerable |
| 18 | Falta de hardware por parte del cliente | 2-Bajo | 4-Insignificante |
| 19 | Expectativas muy elevadas por parte del cliente | 3-Moderado | 3-Tolerable |
| 20 | Variación de la moneda | 4-Alto | 4-Insignificante |
| 21 | Falta de experiencia en el desarrollo y gestión del proyecto. | 3-Moderado | 4-Insignificante |
| 22 | Falta de capacitación de los usuarios. | 3-Moderado | 4-Insignificante |
| 23 | Falta de conocimiento del negocio a tratar | 4-Alto | 4-Insignificante |

## Planificación de riesgos

Se realizan diferentes estrategias de prevención, minimización y planes de contingencia para los riesgos que pueden tener mayor impacto en el proyecto.

|  |  |
| --- | --- |
| **Riesgos** | **Estrategia** |
| El cliente no desea continuar con el proyecto | -Se busca mantener un contacto activo con el cliente de buena forma.  -Constante interés por el proyecto  -Contingencia: Se termina el proyecto para reutilizarlo |
| Baja de uno de los integrantes del proyecto | -Se buscar crear un clima hostil de trabajo ente los miembros del equipo  -Contingencia: Se busca una persona que pueda reemplazar ese espacio o se continua con el proyecto sin ese miembro |
| Pérdida de datos | -Se establece un plan de copia de seguridad para los datos más importantes del negocio |
| Mal cálculo del planeamiento del proyecto en general debido a la falta de experiencia | -Establecer un rango de tiempo mayor al obtenido en las estimaciones para poder cumplir con los límites  -Contingencia: Reconocer la falta de experiencia y recalcular las estimaciones |
| Fallas en las herramientas de desarrollo | -Crear copias de seguridad constantes para poder restaurar correctamente luego de una falla |
| Cambio legales que intervengan en la integridad del proyecto | -Hacer un análisis previo que determine si el proyecto es legalmente viable  -Contingencia: Renegociar con el cliente considerando las modificaciones pertinentes |

## Valoración de riesgos (factores)

|  |  |
| --- | --- |
| **Tipo de riesgo** | **Indicadores** |
| Tecnología | Entrega retrasada del hardware o del soporte del software. |
| Personal | Mala relaciones entre los miembros del equipo, mala disposición a realizar las diferentes actividades |
| Herramientas | Falta de eficiencia de las herramientas utilizadas, rechazo a la utilización de algunas herramientas |
| Requerimientos | Cambios frecuentes en la funcionalidad por parte del cliente |
| Estimación | Fracaso en los tiempos acordados para la entrega del producto |

## Conclusión:

En un mercado tan cambiante y bajo factores tan aleatorios que pueden afectar la estabilidad de nuestro proyecto, es importante siempre tomar recaudos en cuanto a los riesgos, por más leves que parezcan. La seriedad de los mismos puede variar de un día para el otro, es vital que el equipo siempre tenga en cuenta estos aspectos y trate siempre estar un paso adelantado al desastre.

# Anexo III: Gestión de calidad

## Introducción:

Según el IEEE “La calidad del software es el grado con el que un sistema, componente o proceso cumple los requerimientos especificacados y las necesidades o expectativas del cliente o usuario”. La mejora de procesos significa entender los procesos existentes y cambiarlos para mejorar la calidad del producto y/o reducir los costos, el tiempo de desarrollo y reducir el número de defectos en el software entregado.

La utilización de sistemas de calidad otorga un gran número de beneficios a las compañías que apuestan a estas estrategias. No solo reducen sus costos, sino que además incrementan sus ingresos debido al mayor grado de satisfacción por parte de sus clientes, imponiéndose antes otras compañía que no posean un sistema de calidad.

A continuación se detallarán dos modelos importantes en la gestión de la calidad del software.

## Modelo CMMI:

El CMMI (Capability Matury Model Integration) es un modelo de calidad del software que clasifica las empresas en niveles de madurez. Estos niveles sirven para conocer la madurez de los procesos que se realizan para producir software

Este modelo se compone de 5 niveles de madurez, a continuación de detallarán cada uno de ellos

* Inicial o Nivel 1 CMMI. Este es el nivel en donde están todas las empresas que no tienen procesos. Los presupuestos se disparan, no es posible entregar el proyecto en fechas, los empleados si tienen que quedar durante noches y fines de semana para terminar un proyecto. No hay control sobre el estado del proyecto, el desarrollo del proyecto es completamente opaco, no se sabe que pasara con el.
* Nivel 2 CMMI. Quiere decir que el éxito de los resultados obtenidos se pueden repetir. La principal diferencia entre este nivel y el anterior es que el proyecto es gestionado y controlado durante el desarrollo del mismo. El desarrollo no es opaco y se puede saber el estado del proyecto en todo momento.

Los procesos que hay que implantar para alcanzar este nivel son:

* + Gestión de requisitos
  + Planificación de proyectos
  + Seguimiento y control de proyectos
  + Gestión de proveedores
  + Aseguramiento de la calidad
  + Gestión de la configuración
* Nivel 3 CMMI. Alcanzar este nivel significa que la forma de desarrollar proyectos (gestión e ingeniería) esta definida, por definida quiere decir que esta establecida, documentada y que existen métricas (obtención de datos objetivos) para la consecución de objetivos concretos.

Los procesos que hay que implantar para alcanzar este nivel son:

* + Desarrollo de requisitos
  + Solución Técnica
  + Integración del producto
  + Verificación
  + Validación
  + Desarrollo y mejora de los procesos de la organización
  + Definición de los procesos de la organización
  + Planificación de la formación
  + Gestión de riesgos
  + Análisis y resolución de toma de decisiones

La mayoría de las empresas que llegan al nivel 3 paran aquí, ya que es un nivel que proporciona muchos beneficios y no ven la necesidad de ir más allá porque tienen cubiertas la mayoría de sus necesidades.

* Nivel 4 CMMI. Los proyectos usan objetivos medibles para alcanzar las necesidades de los clientes y la organización. Se usan métricas para gestionar la organización.

Los procesos que hay que implantar para alcanzar este nivel son:

* + Gestión cuantitativa de proyectos
  + Mejora de los procesos de la organización
* Nivel 5 CMMI. Los procesos de los proyectos y de la organización están orientados a la mejora de las actividades. Mejoras incrementales e innovadoras de los procesos que mediante métricas son identificadas, evaluadas y puestas en práctica.

Los procesos que hay que implantar para alcanzar este nivel son:

* + Innovación organizacional
  + Análisis y resolución de las causas

Normalmente las empresas que intentan alcanzar los niveles 4 y 5 lo realizan simultáneamente ya que están muy relacionados.

La implantación de un modelo de estas características es un proceso largo y costoso que puede costar varios años de esfuerzo. Aun así el beneficio obtenido para la empresa es mucho mayor que lo invertido.

## Modelo ISO

ISO 9000 es un conjunto de normas sobre calidad y gestión de calidad, establecidas por la Organización Internacional de Normalización (ISO). Se pueden aplicar en cualquier tipo de organización o actividad orientada a la producción de bienes o servicios. Las normas recogen tanto el contenido mínimo como las guías y herramientas específicas de implantación como los métodos de auditoría. El ISO 9000 especifica la manera en que una organización opera sus estándares de calidad, tiempos de entrega y niveles de servicio. Existen más de 20 elementos en los estándares de esta ISO que se relacionan con la manera en que los sistemas operan.

Las ventajas que posee este modelo son:

* Estandarizar las actividades del personal que trabaja dentro de la organización por medio de la documentación.
* Incrementar la satisfacción del cliente al asegurar la calidad de productos y servicios de manera consistente, dada la estandarización de los procedimientos y actividades.
* Medir y monitorear el desempeño de los procesos.
* Incrementar la eficacia y/o eficiencia de la organización en el logro de sus objetivos.
* Mejorar continuamente en los procesos, productos, eficacia, entre otros.
* Reducir las incidencias negativas de producción o prestación de servicios.

## Conclusión

Tanto las normas ISO como la metodología CMMI buscan obtener calidad en el desarrollo del producto, la decisión del método depende mucho del objetivo de la empresa.

En el caso de CMMI se restringe al proceso de producción más específica estableciendo una jerarquía de niveles en cuanto a la calidad del software, en tanto ISO 9000 tiene más en cuenta el ambiente en la producción y la atención hacia el cliente, trata de fijar requisitos mínimos que debe cumplir un sistema de calidad, permitiendo así que la empresa tenga su propio sistema de calidad acorde a sus características y a un costo de implementación más accesible que la metodología CMMI la cual incrementa el precio conjunto a su jerarquía. A continuación se presentará un cuadro con más comparaciones

|  |  |
| --- | --- |
| ISO 9000 | CMMI |
| Estándar | Modelo |
| Amplia dirección, no detallado | Detallado |
| Un conjunto de requerimientos a satisfacer | Pasos progresivos (niveles) |
| Sin guías de implementación | Institucionalización y guía de implementación |
| La norma ISO 9000 es muy general, ya que su concepción fue pensada para abarcar cualquier tipo de empresa | CMMI está enfocada hacia el desarrollo concreto del software |
| ISO 9000 provoca una realidad empresarial en la que el único reto es mantener la certificación | CMMI mueve al sistema de calidad de la organización en una dirección de mejora continua |
| Requiere de auditorías externas periódicas para comprobar su continuidad y adecuación | Aporta a la empresa capacidad de autoevaluación y cierta independencia |
| Realiza un chequeo a la organización en base a la comparación de uno de los estándares predefinidos | Se basa en la evaluación de la capacidad y madurez del proceso |

En nuestro caso no se justifica aplicar algunos de estos métodos, ya que el proyecto tiene un fin principalmente académico y además no se requiere altos niveles de calidad. Sin embargo nosotros tratamos de acercarnos a una buena calidad a través de buenas prácticas, como la gestión de riesgos, estudios de viabilidad, gestión de configuración de software, testing, entre otras tareas desarrolladas en el proyecto.

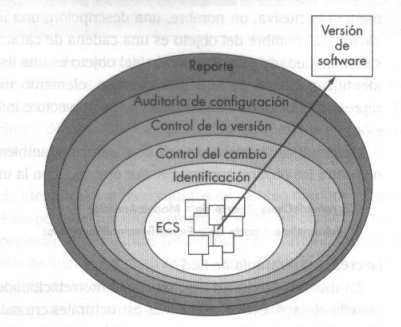
# Anexo IV: Gestión de la configuración del Software

## Introducción:

Realizamos este anexo ya que a la hora del desarrollo de un sistema es inevitable que sucedan diferentes cambios tanto en la empresa como agentes externos a la misma que interfieran en el sistema y que requieran de una modificación en la configuración del mismo. Por otro lado puede que también haya intenciones por parte del cliente de agregar requerimientos al sistema para darle distintas funcionalidades al mismo.

Por este motivo es necesario gestionar con eficacia los cambios en el sistema para controlar las diferentes versiones del producto y auditar en informar los cambios realizados y así tener una sólida gestión del proyecto a lo largo de su vida útil.

## Proceso de configuración de software

****

## Identificación de objetos en la configuración de software

En esta etapa se requiere nombrar cada uno y por separado los elementos de configuración del software (ECS). En el siguiente cuadro se detallaran los mismos

|  |  |  |  |
| --- | --- | --- | --- |
| **ELEMENTO** | **TIPO** | **VERSIÓN** | **LISTA DE RECURSOS** |
| Requerimientos | **Documento** |  | Listado de requerimientos |
| Casos de uso | **Documento** |  | Diagrama de Paquete de Casos de Uso, Diagrama de Casos de Uso. |
| Especificaciones de Caso de uso | **Documento** |  | Descripción de los casos de uso del sistema |
| Interfaces de Usuario | **Programa** |  | Código Fuente de pickupmeal |
| Clases de software | **Programa** |  | Código Fuente de pickupmeal |
| Diagramas de Secuencia | **Documento** |  | Diagramas de Secuencias de los CU. |
| Diagramas de Clases | **Documento** |  | Diagrama de Clases |
| Diagramas de Robustez | **Documento** |  | Diagrama de robustez |
| Modelo de Datos | **Documento**  **Programa** |  | Base de datos, Modelo de datos. |

## Control de versión

Este procedimiento combina diferentes herramientas para gestionar las diferentes versiones que se crean durante el proceso de desarrollo del software.

* **Línea Base**

“Una especificación o producto que se ha revisado formalmente y se está de acurdo con los resultados, y que a partir de ahí sirve como la base para el desarrollo posterior y que puede cambiarse sólo por medio de procedimientos formales de control de cambio”.

A continuación se detallan los sucesos por los cuales se realizarán diferentes cambios en la numeración de las versiones.

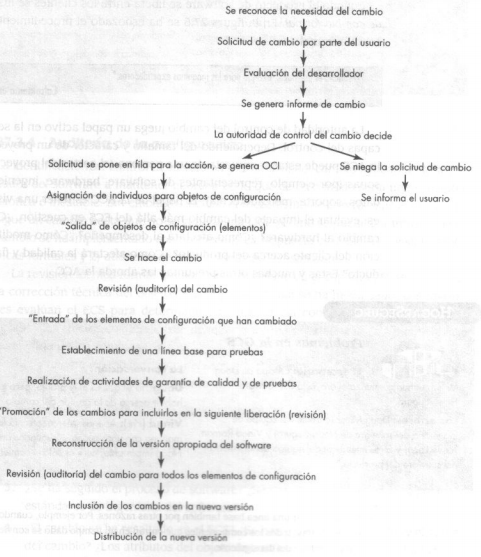
* **VERSION** Se producirá un cambio cada vez que se agreguen funcionalidades que afecten al modelo de datos del programa
* **RELEASE**  Se producirá un cambio cada vez que se modifiquen elementos de la configuración del software que afecten a las controladoras o al modelo de datos.
* **BUILT** Se producirá cada vez que se produzca un cambio en la vista del sistema, interpretándose como cambios en la interfaz del programa y que no afectan ni a las controladoras ni al modelo del sistema.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **VERSIÓN** |  | **RELEASE** |  | **BUILT** |
| XXX | - | XXX | - | XXX |

## Control de cambio

En esta etapa el usuario deberá completar la planilla “Pedido modificación software gestión” y donde deberá completar los datos que correspondan a la sección perteneciente al usuario, luego deberá enviar por e-mail a la dirección destinada .El departamento de sistemas entregará un informe donde decidirá si la modificación será aceptada o denegada, esto lo hará mediante el formulario “Pedido modificación software gestión” perteneciente al departamento de sistema.

En el siguiente esquema extraído de la documentación de la cátedra se ilustra el proceso de control de cambio



## Pedido de modificación de configuración de software (Usuario)

|  |  |  |
| --- | --- | --- |
| **Pedido de Modificación** | | |
| N° Solicitud: | | |
| Fecha: | | |
| Proyecto/Cliente: | | |
| Solicitante del Cambio: | | |
| Versión. Variante: | | |
| **Tipo Cambio** | | |
| Adición | Corrección | Optimización |
|  |  |  |
| **Descripción del Cambio** | | |
|  | | |
| **Prioridad** | | |
| Alta | Media | Baja |
|  |  |  |
| **Estado** | | |
| Aprobado | Rechazado | Aplazado |
|  |  |  |
| Conclusión: | | |
|

## Notificación del pedido (Área de sistema)

|  |  |  |
| --- | --- | --- |
| **Informe del pedido** | | |
| N° Solicitud: | | |
| Fecha Solicitud: | | |
| Proyecto: | | |
| Cliente: | | |
| Revisor del Cambio: | | |
| Fecha de revisión: | | |
| **Áreas Afectadas** | | |
|  | | |
| **Estado** | | |
| Aprobado | Rechazado | Aplazado |
|  |  |  |
| Conclusión: | | |

## Auditoría:

En esta etapa se realiza un control mediante una planilla que contiene una lista de actividades predefinidas.

|  |  |
| --- | --- |
| **Preguntas de Verificación** | **Si/No** |
| ¿Se realizo la orden de Cambio objeto de configuración? |  |
| ¿Se incorporaron modificaciones adicionales? |  |
| ¿Se llevó a cabo RTF para evaluar la corrección técnica? |  |
| ¿Se ha seguido el proceso del software? |  |
| ¿Se han seguido procedimientos de GCS para señalar el cambio, registrarlo y darlo a conocer? |  |
| ¿Se han actualizado adecuadamente todos los ECSs relacionados? |  |
| ¿Se han resaltados los cambios en el ECS? |  |
| ¿Se han especificado las fechas de cambio? |  |
| ¿Reflejan los cambios los atributos del objeto de configuración? |  |

## Informe de estado:

|  |  |  |
| --- | --- | --- |
| **Informe de estado** | | |
| Nº Solicitud: | | Fecha **\_ \_/ \_ \_ / \_ \_** |
| Responsable del Cambio: | | |
| Proyecto |  | |
| Fecha de Cambio |  | |
| Descripción de Cambio |  | |
| Estado |  | |
| Conclusión |  | |

# Anexo V: Políticas de Backup

## Introducción:

### ¿Qué es un backup?

Las copias de seguridad, mejor conocidas o usualmente nombradas como *backups* son una forma de protección de datos tal que permiten la recuperación de la información (ya sean datos o aplicaciones) en el caso de que tenga lugar la pérdida del equipamiento informático (o hardware), ya sea debido a desastres naturales, fallos en los dispositivos de almacenamiento o sistema operativo, espionaje, malversación de información, siniestros, errores de operación del sistema, etc. A fin de evitar que dichos sucesos interfieran en la normal actividad del negocio o empresa donde se encuentre funcionando la solución informática, es necesario establecer una *política*  de backup que sirva como normativa para los resguardos. Para establecer la política es necesario conocer las partes que la componen.

### Política de backup.

Lo primero que hay que especificar es el *alcance* del documento donde se establece la política de backup dentro de la organización. Es decir, especificar a quién está dirigido el escrito.

Luego se deberá realizar un listado de todas las personas que reciben una copia del documento, es importante detallar las responsabilidades de todos los involucrados en el proceso de resguardo y restauración de datos. Resulta pertinente mencionar el valor de los datos resguardados, quizás no en términos monetarios, sino en función de su importancia dentro de la organización. Por ejemplo, especificar aquellos datos o servidores necesarios para que el negocio continúe en funcionamiento ante un evento catastrófico.

Por otro lado se debe detallar los medios utilizados para la realización del backup especificando los tipos, los lugares en donde estos estarán ubicados. y el responsable de esta tarea. Un punto importante en este plan es la metodología utilizada, que detalla el tipo de método a realizar, la frecuencia, etc.

También se agrega un documento de procedimiento de restauración donde se define los pasos a seguir por cualquier persona u oficina de la organización que requiere la restauración de datos. Aquí es recomendable hablar sobre la confidencialidad de los datos resguardados, así como del protocolo para retirar los datos resguardados y la responsabilidad de quien los retira. Este plan se realiza para que no se de una perdida de datos pero en caso que este suceda se especifican las condiciones a darse cuando este problema surge.

### Ventajas de un backup

* Otorga un mayor nivel de seguridad de los datos para el sistema.
* Permite la recuperación de información relevante para el negocio en el caso de que ocurra un desastre o falla de Software/Hardware.
* Otorga mayor robustez y confiabilidad a la solución informática propuesta para el cliente.

### Desventajas de un backup

* Implica un gasto aparte que se suma al presupuesto del sistema. El valor del mismo, varía acorde a la metodología y los soportes de almacenamiento seleccionados para realizar la copia de seguridad.
* Si no se estipula una buena política de backup, puede perderse mucha información importante para el negocio.
* Consume tiempo y recursos que de no estipularse de buena manera, pueden influir en la dinámica del negocio donde se aplicó la solución informática.

### Tipos de backup

#### Backups incrementales

Una operación de backup incremental sólo copia los datos que han variado desde la última operación de backup de cualquier tipo. Se suele utilizar la hora y fecha de modificación estampada en los archivos, comparándola con la hora y fecha del último backup. Las aplicaciones de backup identifican y registran la fecha y hora de realización de las operaciones de backup para identificar los archivos modificados desde esas operaciones. Como un backup incremental sólo copia los datos a partir del último backup de cualquier tipo, se puede ejecutar tantas veces como se desee, pues sólo guarda los cambios más recientes. La ventaja de un backup incremental es que copia una menor cantidad de datos que un backup completo. Por ello, esas operaciones se realizan más deprisa y exigen menos espacio para almacenar el backup.

#### Backups diferenciales

Una operación de backup diferencial es similar a un backup incremental la primera vez que se lleva a cabo, pues copiará todos los datos que hayan cambiado desde el backup anterior. Sin embargo, cada vez que se vuelva a ejecutar, seguirá copiando todos los datos que hayan cambiado desde el anterior completo. Por lo tanto, en las operaciones subsiguientes almacenará más datos que un backup incremental, aunque normalmente muchos menos que un backup completo. Además, la ejecución de los backups diferenciales requiere más espacio y tiempo que la de los backups incrementales, pero menos que la de los backup completos.

### Aplicación:

#### Política de backup

**Alcance:** la descripción de esta política de backup será entregada al líder del departamento de sistemas, el cual es el responsable de la integridad de los datos de ***pickupmeal.com***

**Responsabilidades:** el backup se realizará periódicamente de forma automática, sin embargo, el líder del departamento de sistemas será el encargado de verificar cada cierto tiempo (sin necesidad que ocurra un evento indeseable) que éste se realice de manera correcta, así como también se encargará de la restauración de los datos si es necesario. Los datos son de vital importancia para el funcionamiento de ***pickupmeal.com*** son los siguientes: *Negocios, Usuarios, menues, platos y pedidos*

#### Descripción del resguardo de los datos:

En nuestro sistema se utilizará el backup de tipo completo, que suele llevar más tiempo pero proporciona la seguridad de poseer todos los datos en determinada fecha y la recuperación de los mismos es extremadamente veloz y confiable.

Se utilizará un disco rígido distinto al utilizado para las transacciones habituales del sistema cuando está en pleno funcionamiento, dicha elección se basa en el hecho de que como solo se utilizará para dicho fin, es menos propenso probabilísticamente a errores o fallas. Dicho disco no tendrá sistema operativo, simplemente estará formateado con el tipo de archivos NTFS.

La frecuencia en la que se realizará el backup será la siguiente: todos los días en un momento determinado entre las 15:00 y 17:00 que es el horario cuando menos actividad existe en el local.

Técnicamente el backup se realiza mediante dos scripts en lenguaje MySQL que se ejecutan automáticamente mediante una tarea programada en el sistema operativo (Windows 10 en este caso). Debido a la forma en la que está programada la tarea, el hecho en sí del backup es transparente para el usuario salvo que se le presentará una pantalla de línea de comando propio del Sistema Operativo (consola DOS) en la cual el usuario solo deberá presionar una tecla para confirmar que se ha realizado el backup.

#### Descripción técnica de la solución de backup:

El método más utilizado para crear copias de seguridad de MySQL se basa en el uso del comando mysqldump. Este comando se incluye dentro de las utilidades del propio servidor MySQL, por lo que ya se instaló cuando instalaste MySQL

A continuación se especificarán los scripts y archivos utilizados, incluyendo su código y algunas capturas de su funcionamiento.

Primeramente se describen los scripts utilizados para la realización de la copia de seguridad, están realizados en lenguaje MySQL para ser ejecutados en MySQL, el cual es el motor de base de datos seleccionado para este sistema. Cuando se ejecutan los mismos, producen un archivo .bak que se almacenará en el disco ajeno a donde está instalado el sistema transaccional. El archivo .bak tendrá como nombre: [Nombre\_Base\_Datos] + fecha\_realizacion\_backup . Se almacenará el fichero con la ruta “D:\Backup – pickupmeal”. La generación del tipo de archivos .bak permite una fácil recuperación y traslado de los backups (mediante pendrive, disco extraíble, internet, etc). Cada dos semanas el administrador de base de datos se conectará remotamente a la pc utilizada para la eliminación de los archivos .bak que ya no sean necesarios.

#### Código scripts sql:

*HACER\_BACKUP\_pickupmeal.sql*

c:

cd \Program Files\MySQL\MySQL Server 5.6\bin

mysqldump -u root -p15561000 pumdb > c:\backup\backup.sql

**Restore:** en caso de que sea necesario, el restore de la base de datos se realiza manualmente. El encargado es el líder de departamento de sistema, lo puede hacer tanto personalmente en el local o mediante conexión remota vía TeamViewer o similiar. Se utiliza un archivo . bat generado el cual restorea automaticamenta la base de datos.

#### Conclusión:

En las actividades rutinarias de una empresa que posee un sistema para su administración suelen usarse las computadoras full time, produciendo más desgaste en los componentes de las mismas, aumentando la probabilidad de errores y fallas en ellos. Así mismo, los posibles eventos externos o internos de la empresa pueden afectar las computadoras o sus componentes produciendo la perdida de datos importantes, en muchos casos, activos determinantes en la actividad que realice la organización. Por ello es necesaria siempre una política de backup, sea o no requerida explícitamente por el cliente, por que permite a nuestro software ser confiable, robusto y poderosamente útil para el cliente en el caso de que ocurran siniestros.

# Anexo VI: Patrones de Diseño

## Introducción

Un *patrón de diseño* es una abstracción de una solución en un nivel alto. Los patrones solucionan problemas que existen en muchos niveles de abstracción. Hay patrones que abarcan las distintas etapas del desarrollo; desde el **análisis hasta el diseño** y desde la **arquitectura** hasta la **implementación**.

Muchos diseñadores y arquitectos de *software* han definido el término de ***patrón de diseño*** de varias formas que corresponden al ámbito a la cual se aplican los patrones. Luego, se dividió los patrones en diferentes categorías de acuerdo a su uso.

Los diseñadores de software extendieron la idea de **patrones de diseño** al proceso de desarrollo de software. Debido a las características que proporcionaron los *lenguajes orientados a objetos* (como herencia, abstracción y encapsulamiento) les permitieron relacionar entidades de los lenguajes de programación a entidades del mundo real fácilmente, los diseñadores empezaron a aplicar esas características para crear soluciones comunes y reutilizables para problemas frecuentes que exhibían patrones similares.

Fue por los años 1994, que apareció el libro "*Design Patterns: Elements of Reusable Object Oriented Sofware*" escrito por los ahora famosos **Gang of Four** (GoF, que en español es la pandilla de los cuatro) formada por Erich Gamma, Richard Helm, Ralph Johnson y John Vlissides. Ellos recopilaron y documentaron 23 patrones de diseño aplicados usualmente por expertos diseñadores de software orientado a objetos. Desde luego que ellos no son los inventores ni los únicos involucrados, pero ese fue luego de la publicación de ese libro que empezó a difundirse con más fuerza la idea de patrones de diseño.

El grupo de **GoF** clasificaron los patrones en 3 grandes categorías basadas en su **PROPÓSITO**: creacionales, estructurales y de comportamiento.

* **Creacionales**: Patrones creacionales tratan con las formas de crear instancias de objetos. El objetivo de estos patrones es de abstraer el proceso de instanciación y ocultar los detalles de cómo los objetos son creados o inicializados.
* **Estructurales**: Los patrones estructurales describen como las clases y objetos pueden ser combinados para formar grandes estructuras y proporcionar nuevas funcionalidades. Estos objetos adicionados pueden ser incluso objetos simples u objetos compuestos.
* **Comportamiento**: Los patrones de comportamiento nos ayudan a definir la comunicación e iteración entre los objetos de un sistema. El propósito de este patrón es reducir el acoplamiento entre los objetos.

En el segundo nivel, ellos clasificaron los patrones en 2 ámbitos: Clases y objetos. Es así que, tenemos 6 tipos de patrones:

|  |  |
| --- | --- |
| **Creacionales** | * **Creacional de la Clase** Los patrones creacionales de Clases usan la herencia como un mecanismo para lograr la instanciación de la Clase. Por ejemplo el método Factoría. * **Creacional del objeto** Los patrones creacionales de objetos son más escalables y dinámicos comparados de los patrones creacionales de Clases. Por ejemplo la Factoría abstracta y el patrón Singleton. |
| **Estructurales** | * **Estructural de la Clase** Los patrones estructurales de Clases usan la herencia para proporcionar interfaces más útiles combinando la funcionalidad de múltiples Clases. Por ejemplo el patrón Adaptador (Clase). * **Estructural de Objetos** Los patrones estructurales de objetos crean objetos complejos agregando objetos individuales para construir grandes estructuras. La composición de l patrón estructural del objeto puede ser cambiado en tiempo de ejecución, el cual nos da flexibilidad adicional sobre los patrones estructurales de Clases. Por ejemplo el Adaptador (Objeto), Facade, Bridge, Composite. |
| **Comportamiento** | * **Comportamiento de Clase** Los patrones de comportamiento de Clases usan la herencia para distribuir el comportamiento entre Clases. Por ejemplo Interpreter. * **Comportamiento de Objeto** Los patrones de comportamiento de objetos nos permite analizar los patrones de comunicación entre objetos interconectados, como objetos incluidos en un objeto complejo. Ejemplo Iterator, Observer, Visitor. |

## Patrones J2EE

Con la aparición del **J2EE**, todo un nuevo catálogo de patrones de diseño emergió. Desde que **J2EE** es una arquitectura por si misma que involucra otras arquitecturas, incluyendo servlets, *JavaServer Pages*, *Enterprise JavaBeans*, y más, merece su propio conjunto de patrones específicos para diferentes aplicaciones empresariales.

De acuerdo al libro "*J2EE PATTERNS Best Practices and Design Strategies*", existen 5 capas en la arquitectura J2EE:

* Cliente
* Presentación
* Negocios
* Integración
* Recurso

El libro explica 15 patrones J2EE que están divididos en 3 de las capas: presentación, negocios e integración.

### Catálogo de patrones J2EE

### Capa de presentación

|  |  |
| --- | --- |
| **Decorating Filter / Intercepting Filter** | Un objeto que está entre el cliente y los componentes Web. Este procesa las peticiones y las respuestas. |
| **Front Controller/ Front Component** | Un objeto que acepta todos los requerimientos de un cliente y los direcciona a manejadores apropiados. El patrón Front Controller podría dividir la funcionalidad en 2 diferentes objetos: el *Front Controller* y el *Dispatcher*. En ese caso, El Front Controller acepta todos los requerimientos de un cliente y realiza la autenticación, y el Dispatcher direcciona los requerimientos a manejadores apropiada. |
| **View Helper** | Un objeto helper que encapsula la lógica de acceso a datos en beneficio de los componentes de la presentación. Por ejemplo, los *JavaBeans* pueden ser usados como patrón View Helper para las páginas JSP. |
| **Composite view** | Un objeto vista que está compuesto de otros objetos vista. Por ejemplo, una página *JSP* que incluye otras páginas JSP y *HTML*usando la directiva include o el action include es un patrón Composite View. |
| **Service To Worker** | Es como el patrón de diseño MVC con el Controlador actuando como Front Controller pero con una cosa importante: aquí el Dispatcher (el cual es parte del Front Controller) usa View Helpers a gran escala y ayuda en el manejo de la vista. |
| **Dispatcher View** | Es como el patrón de diseño MVC con el controlador actuando como Front Controller pero con un asunto importante: aquí el Dispatcher (el cual es parte del Front Controller) no usa View Helpers y realiza muy poco trabajo en el manejo de la vista. El manejo de la vista es manejado por los mismos componentes de la Vista. |

### Capa de Negocios

|  |  |
| --- | --- |
| **Business Delegate** | Un objeto que reside en la capa de presentación y en beneficio de los otros componentes de la capa de presentación llama a métodos remotos en los objetos de la capa de negocios. |
| **Value Object/ Data Transfer Object/ Replicate Object** | Un objeto serializable para la transferencia de datos sobre la red. |
| **Session Façade/ Session Entity Façade/ Distributed Façade** | El uso de un bean de sesion como una fachada (facade) para encapsular la complejidad de las interacciones entre los objetos de negocio y participantes en un flujo de trabajo. El Session Façade maneja los objetos de negocio y proporciona un servicio de acceso uniforme a los clientes. |
| **Aggregate Entity** | Un bean entidad que es construido o es agregado a otros beans de entidad. |
| **Value Object Assembler** | Un objeto que reside en la capa de negocios y crea Value Objets cuando es requerido. |
| **Value List Handler/ Page-by-Page Iterator/ Paged List** | Es un objeto que maneja la ejecución de consultas SQL, caché y procesamiento del resultado. Usualmente implementado como beans de sesión. |
| **Service Locator** | Consiste en utilizar un objeto Service Locutor para abstraer toda la utilización JNDI y para ocultar las complejidades de la creación del contexto inicial, de búsqueda de objetos home EJB y recreación de objetos EJB. Varios clientes pueden reutilizar el objeto Service Locutor para reducir la complejidad del código, proporcionando un punto de control. |

### Capa de Integración

|  |  |
| --- | --- |
| **Data Access Object Service** | Consiste en utilizar un objeto de acceso a datos para abstraer y encapsular todos los accesos a la fuente de datos. El DAO maneja la conexión con la fuente de datos para obtener y almacenar datos. |
| **Service Activator** | Se utiliza para recibir peticiones y mensajes asíncronos de los clientes. Cuando se recibe un mensaje, el Service Activator localiza e invoca a los métodos de los componentes de negocio necesarios para cumplir la petición de forma asíncrona. |

## Aplicación

Dada la arquitectura de ***pickupmeal***, se plantean utilizar y describir al menos un patrón por capa, a continuación se describirán los patrones seleccionados.

### Intercepting Filter – Capa de presentación

Contexto

El mecanismo de manejo de petición de la capa de presentación recibe muchos tipos diferentes de peticiones, que requieren diversos tipos de procesamiento. Algunas peticiones simplemente reenvían al componente manejador apropiado, mientras que las demás solicitudes deben ser modificadas, auditadas o descomprimidas antes de ser tratadas posteriormente.

Problema

Se requiere pre-procesamiento y post-procesamiento de una solicitud Web cliente y la respuesta.

Cuando una petición entra en una aplicación web, que a menudo tiene que pasar varias pruebas de acceso antes de la etapa de procesamiento principal. Por ejemplo:

* ¿Se ha autenticado el cliente?
* ¿Tiene el cliente una sesión válida?
* ¿Es la dirección IP del cliente de una red de confianza?
* ¿La ruta de solicitud viola alguna restricción?
* ¿Qué codificación usa el cliente para enviar los datos?

Algunos de estos controles son pruebas, las que resultan en una respuesta afirmativa o negativa que determina si el proceso continuará.

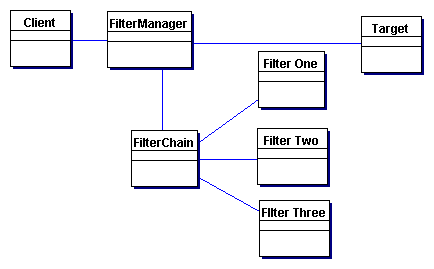
La clave para resolver este problema de una manera flexible y discreto es tener un mecanismo simple para añadir y eliminar componentes de procesamiento, en el que cada componente completa una acción de filtrado específico.

Solución

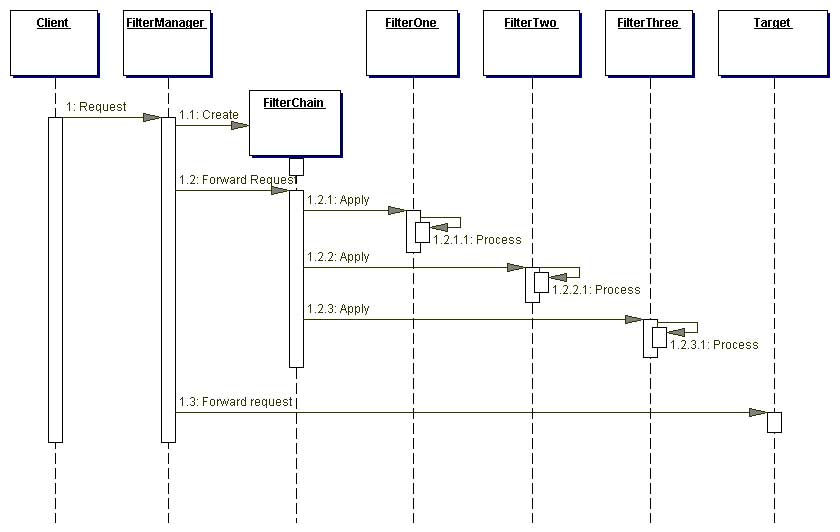
Crear filtros “enchufables” para procesar los servicios comunes de una manera estándar, sin necesidad de cambios en el código de procesamiento de solicitudes de núcleo. Los filtros interceptan las solicitudes entrantes y las respuestas salientes, lo que permite pre-procesamiento y post-procesamiento. Somos capaces de agregar y quitar estos filtros discretamente, sin necesidad de cambios en nuestro código existente.

Podemos, en efecto, decorar nuestro procesamiento principal, con una variedad de servicios comunes, tales como la seguridad, la tala, la depuración, y así sucesivamente. Estos filtros son componentes que son independientes del código principal de la aplicación, y que pueden ser añadidos o eliminados de forma declarativa. Por ejemplo, un archivo de configuración de despliegue puede modificarse para establecer una cadena de filtros. El mismo archivo de configuración podría incluir un mapeo de URLs específicas a esta cadena de filtros. Cuando un cliente solicita un recurso que coincide con este mapeo URL configurado, los filtros de la cadena están cada procesan en orden antes de invocar el recurso de destino solicitado.

Estructura



Participantes y responsabilidades



FilterManager: El FilterManager gestiona el procesamiento de filtros. Crea el FilterChain con los filtros adecuados, en el orden correcto, y se inicia el proceso.

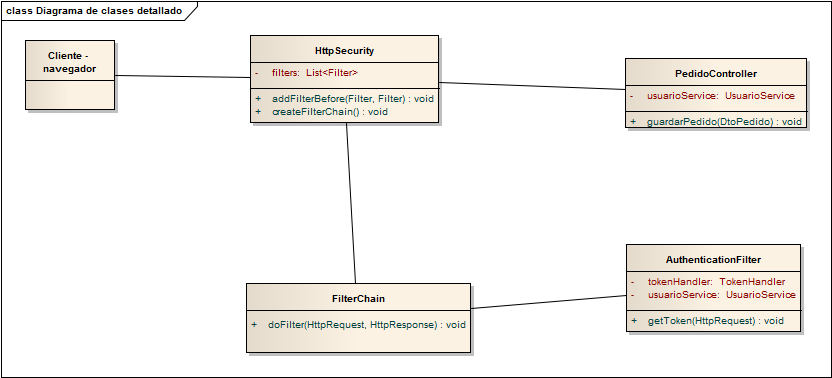
FilterChain: El FilterChain es una colección ordenada de filtros independientes.

FilterOne, FilterTwo, FilterThree: Estos son los filtros individuales que se asignan a un objetivo. El FilterChain coordina su procesamiento.

Target: El Target es el recurso solicitado por el cliente.

#### Aplicación en ***pickupmeal***

En el caso de pickupmeal se utiliza el patrón para verificar la autenticidad de un usuario cuando requiere determinados recursos (seguridad), en este caso el usuario en su request envía (o no) un JSON Web Token. La request es interceptada y se verifica primeramente que exista el token (usuario loggeado) y luego que este token haya sido generado con el secreto que existe en el servidor. Para ello se genera un filtro llamado AuthenticationFilter. Luego la clase FilterChain presente en el framework de J2EE Spring, manipula la cadena de filtros acorde a la lógica de autenticación. Puede dar paso a utilizar el recurso Target solicitado (una API Restful) o devolver un código de error al cliente.



Cliente: el navegador donde surge la request.

HttpSecurity: clase de Spring que interfiere la request y maneja la FilterChain

FilterChain: clase que manipula la cadena de filtros consecutivos en orden.

AuthenticationFilter: el filtro creado para la gestión de tokens y autenticación de los mismos para acceder a los recursos de la API.

PedidoController: clase que manipula los recursos de la API.

### Data Transfer Object – Capa de negocios

Problema

Java 2 Platform, Enterprise Edition (J2EE) aplicaciones implementan componentes de negocio del lado del servidor como beans de sesión y beans de entidad. Algunos métodos expuestos por los componentes de negocio devuelven datos al cliente. A menudo, el cliente invoca métodos Get varias veces de un objeto de negocio hasta que obtiene todos los valores de los atributos.

Los beans de sesión representan los servicios a las empresas y no se comparten entre los usuarios. Los beans de entidad, por otro lado, son multiusuario, objetos transaccionales que representan datos persistentes. Un bean de entidad expone los valores de los atributos proporcionando un método de acceso (también referido como un getter) para cada atributo que desea exponer.

Cada método de llamada realizada al objeto de servicio de negocio, ya sea un bean de entidad o un bean de sesión, es potencialmente remoto. Así, en un (EJB) aplicación Enterprise JavaBeans tales invocaciones remotas usan la capa de red, independientemente de la proximidad del cliente al bean, creando una sobrecarga de la red.

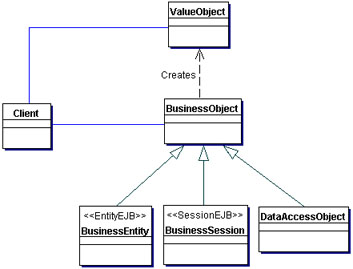
A medida que el uso de estos métodos remotos aumenta, el rendimiento de la aplicación puede degradar de manera significativa. Por lo tanto, el uso de múltiples llamadas para conseguir métodos que devuelven valores de atributos individuales es ineficiente para la obtención de valores de datos de un enterprise bean.

Solución

Utilizar un Transfer Object para encapsular los datos de negocio. Una sola llamada método se utiliza para enviar y recuperar el Transfer Object. Cuando el cliente solicita el recurso, la clase puede construir el Transfer Object, rellenarlo con sus valores de atributos, y pasarlo por valor al cliente.

Cuando una aplicación utiliza un Transfer Object, el cliente realiza una sola invocación de métodos remotos al bean enterprise para solicitar el Transfer Object en lugar de numerosas llamadas de método remoto para obtener valores de atributos individuales.

Estructura



Client: el navegador de donde se realiza la request.

ValueObject : el objeto que se crea para transferir los datos.

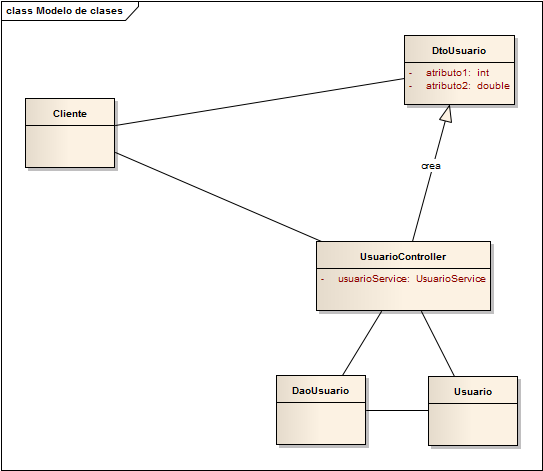
BusinessObject: la clase que manipula la petición y crea el ValueObject.

BusinessEntity, BusinessSession, DataAccesObject: clases de la aplicación de donde provienen los objetos y datos que se manejan a nivel servidor

#### Aplicación en pickupmeal

En el caso de pickupmeal, al usar AngularJS en el frontend y pensando pragmáticamente a futuro en la implementación de aplicaciones móviles, es providencial proveer de archivos JSON al exterior del servidor para que sean consumidos. Debido a la utilización de Hibernate como ORM, los objetos que se obtienen resultan muy complejos (mayormente por las listas y relaciones bidireccionales). Por ello es necesario un objeto más sencillo con los atributos necesarios para la utilización en el frontend. He aquí la utilización del patrón descripto para facilitar la comunicación.

La disposición de clases es la siguiente:



### Data Access Object – Capa de integración

Contexto

El acceso a los datos varía dependiendo de la fuente de los datos. El acceso al almacenamiento persistente, tal como para una base de datos, varía en gran medida dependiendo del tipo de almacenamiento (bases de datos relacionales, bases de datos orientadas a objetos, archivos planos, y así sucesivamente).

Problema

Muchas aplicaciones en el mundo real de Java 2 Platform, Enterprise Edition (J2EE) necesitan utilizar datos persistentes en algún momento. Para muchas aplicaciones, el almacenamiento persistente se implementa con diferentes mecanismos, y no se marcan las diferencias en las API utilizadas para acceder a los diferentes mecanismos de almacenamiento persistentes. Otras aplicaciones pueden necesitar acceder a datos que residen en sistemas separados. Por ejemplo, los datos pueden residir en sistemas mainframe, Lightweight Directory Access Protocol (LDAP) repositorios, y así sucesivamente. Otro ejemplo es cuando a los datos lo proporcionan servicios a través de sistemas externos, como los sistemas de negocio a negocio (B2B) de integración, servicio de oficinas de las tarjetas de crédito, y así sucesivamente.

Las aplicaciones pueden utilizar la API de JDBC para acceder a los datos que residen en un sistema de gestión de bases de datos relacionales (RDBMS). El API JDBC permite el acceso y la manipulación de los datos en el almacenamiento persistente, estándar, como una base de datos relacional. El API JDBC permite a las aplicaciones J2EE utilizar sentencias SQL, que son: el estándar para acceder a las tablas RDBMS. Sin embargo, incluso dentro de un entorno RDBMS, la sintaxis actual y el formato de las sentencias SQL pueden variar dependiendo del producto de base de datos particular.

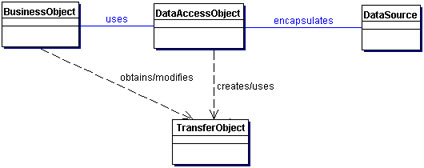
Hay incluso una mayor variación con diferentes tipos de almacenamiento persistente. Mecanismos de acceso, apoyados APIs y características varían entre los diferentes tipos de almacenes persistentes como RDBMS, bases de datos orientadas a objetos, archivos planos, y así sucesivamente. Estas fuentes de datos dispares ofrecen desafíos para la aplicación y potencialmente pueden crear una dependencia directa entre el código de aplicación y el código de acceso a datos.

Solución

Usar un objeto de acceso a datos (DAO) para abstraer y encapsular todo el acceso a la fuente de datos. El DAO gestiona la conexión con la fuente de datos para obtener y almacenar datos.

El DAO implementa el mecanismo de acceso requerido para trabajar con la fuente de datos. La fuente de datos puede ser un almacén persistente como un RDBMS, un servicio externo como un intercambio B2B, un repositorio como una base de datos LDAP, o un servicio de negocio se accede a través de CORBA Internet Inter-ORB Protocol (IIOP) o sockets de bajo nivel. El componente de negocio que se basa en la DAO utiliza la interfaz más simple expuesto por el DAO para sus clientes. El DAO oculta completamente los detalles de implementación de origen de datos de sus clientes. Debido a que la interfaz expuesta por el DAO para clientes no cambia cuando los cambios subyacentes de implementación de origen de datos, este modelo permite la DAO para adaptarse a los diferentes esquemas de almacenamiento sin afectar a sus clientes o componentes de negocio. Esencialmente, la DAO actúa como un adaptador entre el componente y la fuente de datos.

Estructura



Participantes y Responsabilidades



BusinessObject

El BusinessObject representa el cliente datos. Es el objeto que requiere el acceso a la fuente de datos para obtener y almacenar datos.

DataAccessObject

El DataAccessObject es el objeto principal de este patrón. El DataAccessObject abstrae la aplicación de acceso de datos subyacente para el BusinessObject para permitir un acceso transparente a la fuente de datos.

DataSource

Esto representa una implementación de fuente de datos. Una fuente de datos podría ser una base de datos tal como un RDBMS, OODBMS, repositorio XML, el sistema de archivo plano, y así sucesivamente. Una fuente de datos también puede ser otro sistema (legacy / mainframes), servicios (servicio de B2B o buró de tarjeta de crédito), o algún tipo de repositorio (LDAP).

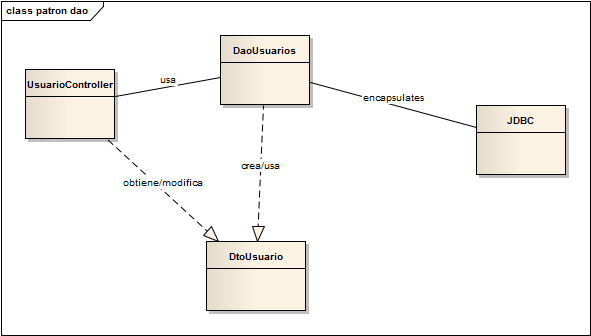
TransferObject

Idem patrón descripto anteriormente.

#### Aplicación en pickupmeal

En el caso de ***pickupmeal***, se tiene una base de datos en MySql la cual se maneja mediante Hibernate. Pensando pragmáticamente y para separar de mejor manera las capas de la aplicación se decide implementar el patrón para aislar el acceso a datos por si en un futuro cambia la fuente de los mismos o se decide implementar otro ORM.

La disposición de las clases es la siguiente:



## Conclusión

Como conclusión del anexo patrones de diseño podemos decir que cuando existe el caso de aplicación resulta realmente útil utilizar estas soluciones, ya que su re utilidad y efectividad está probada en problemas anteriormente planteados, además nos ayuda a mejorar tanto nuestra arquitectura como también nos lleva a realizar una buena práctica ahorrando trabajo innecesario que realizamos la mayoría de las veces.

En el caso del patrón Intercepting Filter, es básicamente una de las venas principales de la seguridad de pickupmeal debido a que por el mismo se verifica la autenticación de los usuarios que quieren acceder a los recursos del servidor.

En el caso del patrón DTO, es muy importante a nivel de traspaso de datos del backend al frontend para simplicidad a la hora de desarrollar y mantener la aplicación.

En el caso del patrón DAO es una buena técnica a futuro para mantener la capa datos encapsulada completamente de las demás partes de la aplicación.

Los patrones de diseño son una excelente práctica que soluciona muchos problemas a la hora de desarrollar.

# Anexo VII: Seguridad

Existen varios sistemas de autenticación en una aplicación web, el sistema más tradicional utilizado hasta el momento funcionaba guardando los datos de la sesión de usuario y lo almacenaba posteriormente esa información en una base de datos como pueden ser Mysql o sql, luego para autenticar al usuario propiamente dicho se realizaba una request a esa base de datos y se comparaba estos valores para verificar si el usuario es quien dice ser. Esta metodología conlleva algunas desventajas propias de la implementación como ser la lentitud, el gasto de recursos entre otras cosas.

Las nuevas tendencias en cuanto al desarrollo web moderno se refiere, es la autenticación por medio de Token, a continuación se detalla el funcionamiento básico de la metodología

El funcionamiento es el siguiente. El usuario se autentica en nuestra aplicación, bien con un par usuario/contraseña, A partir de entonces, cada petición HTTP que haga el usuario va acompañada de un Token en la cabecera. Este Token no es más que una firma cifrada que permite a nuestro API identificar al usuario. Pero este Token no se almacena en el servidor, si no en el lado del cliente (por ejemplo en localStorage o sessionStorage) y el API es el que se encarga de descifrar ese Token y redirigir el flujo de la aplicación en un sentido u otro. Como los tokens son almacenados en el lado del cliente, no hay información de estado y la aplicación se vuelve totalmente escalable. Podemos usar el mismo API para diferentes apliaciones (Web, Mobile, Android, iOS, ...) solo debemos preocuparnos de enviar los datos en formato JSON y generar y descifrar tokens en la autenticación y posteriores peticiones HTTP a través de un middleware.

También nos añade más seguridad. Al no utilizar cookies para almacenar la información del usuario, podemos evitar ataques CSRF (Cross-Site Request Forgery) que manipulen la sesión que se envía al backend. Por supuesto podemos hacer que el token expire después de un tiempo lo que le añade una capa extra de seguridad.

## Autenticación con JSON Web Tokens

En este caso particular se utilizó JWT para autenticar ***pickupmeal.com,***  a continuación se detalla la forma en que se genera este token.

El formato de un JWT está compuesto por 3 string separados por un punto. Donde cada String tiene un significado

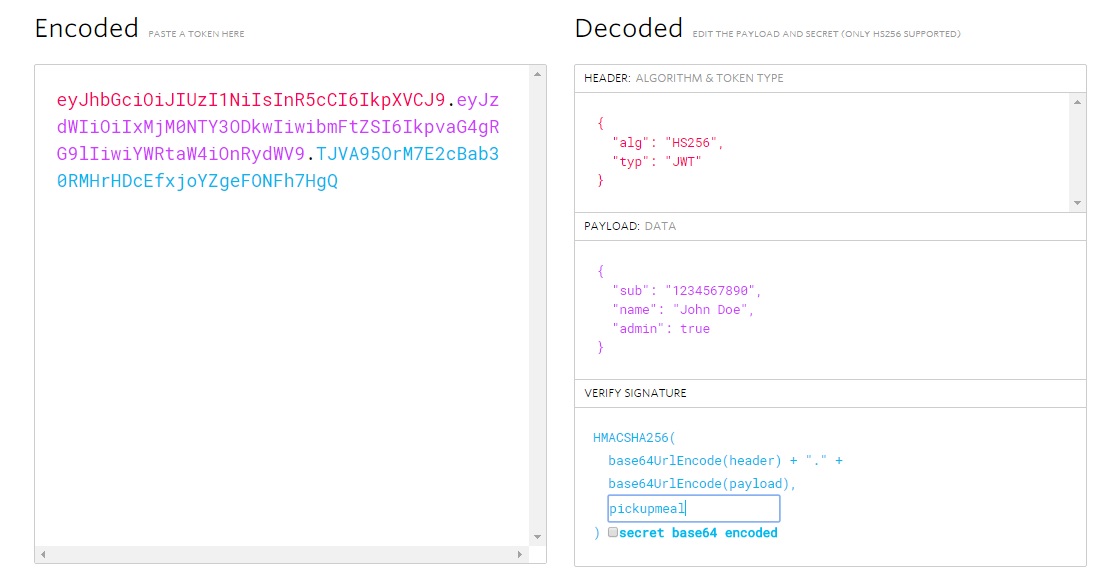
La primera parte del token corresponde al **Header,** esta consta a su vez de otras dos partes, el tipo de token utilizado y el algoritmo por el cual éste es generado.

La segunda parte corresponde al **payload** está compuesto por los llamados JWT Claims donde irán colocados la atributos que definen nuestro token. Exiten varios que puedes consultar aquí, los más comunes a utilizar son:

* sub: Identifica el sujeto del token, por ejemplo un identificador de usuario.
* iat: Identifica la fecha de creación del token, válido para si queremos ponerle una fecha de caducidad. En formato de tiempo UNIX
* exp: Identifica a la fecha de expiración del token. Podemos calcularla a partir del iat. También en formato de tiempo UNIX.

Ademas de estos datos se pueden añadir mas campos, generalmente, en esta parte del token van ubicados los atributos del **User**como el nombre, apellido , dirección entre otras cosas

Como ultima parte del JWT ubicamos la **firma** Está formada por los anteriores componentes (Header y Payload) cifrados en Base64 con una clave secreta (almacenada en nuestro backend). Así sirve de Hash para comprobar la autenticidad del usuario.



# Anexo VIII: Manual de Usuario

## Introducción:

Antes de comenzar con el anexo de manual de usuario, vamos a explicar que es una guía de usuario como más técnicamente se lo llama. Según la definición, éste es un documento de comunicación técnica, que ayuda y da asistencia al usuario para realizar distintas actividades relacionadas a un sistema en particular, también se puede decir, que es una publicación que educa a sus lectores acerca de un tema de forma ordenada y concisa. Se establece que por lo general, este documento está escrito por personal idóneo en el área al cual se desarrolla la guía, utilizando un vocabulario simple para poder llegar a la mayor cantidad de receptores posible. Además en algunos casos el manual está escrito en varios idiomas, esto trae consecuencias en el caso de las traducciones, ya que se puede dar que esta no sea del todo correcta.

Dada su complejidad, todos los productos electrónicos o informáticos suelen contar con su propio manual de usuario. Los artículos más simples (como una pelota o una mesa) no requieren de explicaciones para que los consumidores sepan cómo utilizarlos.

Surge un paradigma frente a los manuales de usuarios, existen las personas que no comienzan a utilizar el producto, hasta aprenderse todas las funciones, pero también están los que no consultan estas guías. Existe un problema para estos últimos ya que hay veces que el objetivo del manual es advertir a los consumidores de las limitaciones de los productos para evitar quejas por fallos que podrían haber sido evitados.

En nuestro caso vamos a realizar la explicación de nuestro software mediante videos demostrativos, dando detalles de cómo se utilizan algunas funciones de pickupmeal. El video será un seguimiento detallado de la forma en que se realizan las operaciones correspondientes a cada módulo.

## **Conclusión**

Finalizando este último anexo, podemos decir primeramente que nos pareció adecuado elegir como explicación el video porque resulta la opción más llevadera e interactiva para el usuario no tiene demasiados conocimientos informáticos. Permitiendo que éste realice un uso adecuado del software durante su vida útil, logrando así sacarle el mayor provecho a la solución informática brindada.

1. IMCM fue desarrollado por un grupo de expertos de la industria, el gobierno y el Software Engineering Institute (SEI) de Carnigie Mellon University. Modelos IMCM guían el desarrollo o mejora de procesos que cumplan con los objetivos empresariales de una organización. Un modelo IMCM también puede ser utilizado como un marco para evaluar la madurez de los procesos de la organización. [↑](#footnote-ref-1)