**Universidad Tecnológica de Tecámac**

**Empresa:** Blue sky

**Proyecto:** Pet Sitting

**Integrantes:**

* Alonso Pérez Antonio
* Jiménez Rodríguez Lizet
* Larios Soto Kay

**Grupo:** 7ITI2

**Fecha de entrega:** 04-Marzo-2019

Índice

[1. Introducción 3](#_Toc2531686)

[2. Análisis de requerimientos 4](#_Toc2531687)

[2.1 Proceso actual 5](#_Toc2531688)

[2.2 Historias de Usuario 6](#_Toc2531689)

[3. Definición de los requerimientos 7](#_Toc2531690)

[3.1 Requerimientos funcionales 8](#_Toc2531691)

[3.2 Requerimientos no funcionales 9](#_Toc2531692)

[4. Modelos del sistema 10](#_Toc2531693)

[4.1 Estructura de sistema 11](#_Toc2531694)

[4.2 Comportamiento del sistema 12](#_Toc2531695)

[5. Propuesta de desarrollo 13](#_Toc2531696)

[6. Plan de pruebas 14](#_Toc2531697)

[6.1 Planeación 15](#_Toc2531698)

[6.2 Planificación 16](#_Toc2531699)

[7. Prototipo del sistema 17](#_Toc2531700)

# Introducción

En la actualidad las mascotas son parte fundamental en la vida de las personas, tanto, que consideran a sus mascotas como parte de su familia, lamentablemente las personas no pueden cuidar de una mascota como se debe por falta de tiempo o por su profesión, hay casos en que las familias salen de vacaciones y los dueños dejan a su mascota solos o son encargados a una persona ajena lo que ocasiona desconfianza a los dueños, por estos motivos se ha decidido crear un dispensador de alimento para la mascota, el cual llene el plato de comida del animal de forma automática con información precargada y además definir la información sobre el tipo de mascota a la que va dirigido. La definición del tipo de mascota y los tiempos en los que se debe dispensar el alimento serán guardados por el usuario mediante una aplicación de escritorio, cabe mencionar que esto solo se realizara una sola vez por el usuario.

Para la elaboración del proyecto es importante hacer el levantamiento de los requerimientos del proyecto porque nos permite analizar las necesidades del usuario como también ayuda a definir los requisitos que debe llevar este proyecto. Se definieron los dos tipos de requerimientos: funcionales y no funcionales.

Así mismo se realizó los diferentes diagramas, modelos y comportamientos, además de la estructura del sistema representada con los diferentes modelos UML evaluados mediante pruebas.

# Análisis de requerimientos

En la ingeniería de software, un análisis de requerimientos es una tarea que cubre el hueco entre la definición del software a nivel sistema y el diseño del mismo. Tanto el desarrollador como el cliente tienen un papel activo, pues juntos definen en detalle los requisitos del sistema a desarrollar y los pasos a seguir.

En otras palabras, el análisis de requisitos abarca aquellas tareas que se dirigen a determinar las necesidades o condiciones que se deben cumplir para un producto o proyecto nuevo o modificado, teniendo en cuenta los requisitos posiblemente conflictivos de los distintos interesados, analizando, documentando, validando y administrando software o requisitos del sistema.

El proceso de análisis de requerimientos refina, modela, especifica y verifica las solicitudes de usuario, y con ello genera documentos base para la ejecución de los procesos siguientes.

Características:

* Es un estudio profundo de una necesidad tecnológica que tiene una empresa, organización o negocio.
* Especifica las características operacionales que tendrá el software a desarrollar.
* Se realiza a través de entrevistas, observación, indagación y demás técnicas específicas.
* Describe el plan del proyecto a seguir.
* Es fundamental entregar el proyecto dentro del tiempo y presupuesto acordados y de los objetivos de negocio.

El análisis de requisitos es crítico para el éxito o el fracaso de un proyecto de sistemas o software.

La calidad de los requisitos se puede mejorar a través de estos y otros métodos como: visualización, uso consistente de plantillas, documentación de dependencias.

## Proceso actual

BLUE SKY es una empresa orientada al desarrollo de software, en estos momentos la empresa está elaborando el proyecto “Pet Sitting”, una aplicación de escritorio en el cual su objetivo es realizar un dispensador de alimento automático para las mascotas

La empresa cuenta con tres personas que realizan las siguientes actividades:

* Líder/Documentador
* Realizar la planeación.
* Levantar los requerimientos.
* Documentar el proceso del proyecto.
  + Diseñar el dispensador de alimento.
* Programador/ Diseñador
  + Desarrollo de la base de datos.
  + Desarrollo de los procesos de la aplicación.
  + Diseño de las interfaces de la aplicación.
* Tester
  + Informe de prueba de la aplicación.
  + Informe de prueba del dispensador.
  + Conexión del dispensador con la aplicación.

Los actores que se involucran durante la ejecución del proyecto y además realizan las siguientes actividades:

* Dueño de la mascota
  + Llenar formulario con la información de la mascota.
* Sistema
  + Llenar el plato de comida del animal.
  + Avisar cuando se agota el alimento.

## Historias de Usuario

# Definición de los requerimientos

La definición de requerimientos es el **conjunto de técnicas y procedimientos que nos permiten conocer los elementos necesarios para definir un proyecto de software.** Es una tarea de ingeniería del software que permite especificar las características operacionales del software, indicar la interfaz del software con otros elementos del sistema y establecer las restricciones que debe cumplir el software.

**Clasificación de los requerimientos:**

* Requerimientos funcionales: qué debe hacer el sistema o software.
* Requerimientos no funcionales: cómo debe funcionar el sistema o software (no su implementación), por ej. calidad, rendimiento, facilidad de uso, etc.

**Características que deberían cumplir los requerimientos**:

* Actual: el requerimiento no debe volverse obsoleto con el paso del tiempo.
* Cohesión: el requerimiento debe dirigirse a solo una única cosa.
* Completo: el requerimiento debe estar completamente declarado en un único lugar, sin información faltante.
* Consistente: el requerimiento no debe contradecir ningún otro requerimiento y debe ser completamente consistente con toda la documentación.
* Correcto/necesario: el requerimiento debe cumplir con la necesidad declarada por los interesados en el sistema/software.
* No ambiguo: el requerimiento debe estar concisamente declarado. Debe expresar hechos objetivos, no opiniones subjetivas. Debe poder poder ser interpretado de una única manera.
* Verificable/demostrable: La implementación del requerimiento debe poder ser resuelta en alguno de estos cuatro métodos: inspección, análisis, demostración o prueba.

## Requerimientos funcionales

|  |  |
| --- | --- |
| # | Requerimientos funcionales |
| 1 | Registrar mascota. |
| 2 | Modificar datos de la mascota. |
| 3 | Eliminar datos de la mascota. |
| 4 | Mostrar datos de la mascota. |
| 5 | Envía datos al dispositivo. |
| 6 | Envía notificación para actualizar datos de la mascota. |
| 7 | Envía notificación de aviso cuando haya poca cantidad de alimento. |
| 8 | Dispensar la cantidad de alimento a soltar. |
| 9 | Llenar el plato de comida de la mascota. |
| 10 | Abrir puerta del alimento. |
| 11 | Cerrar puerta del alimento. |
| 12 | Detectar insuficiencia de alimento del dispensador. |

Los requerimientos funcionales son declaraciones de los servicios que proveerá el sistema, de la manera en que éste reaccionará a entradas particulares. En algunos casos, los requerimientos funcionales de los sistemas también declaran explícitamente lo que el sistema no debe hacer.

## Requerimientos no funcionales

# Modelos del sistema

## Estructura de sistema

## Comportamiento del sistema

# Propuesta de desarrollo

# Plan de pruebas

Es el proceso de prueba básico que comprende la planificación y el control de pruebas, análisis y diseño de pruebas, implementación y ejecución de pruebas, evaluación de criterios de salida y elaboración de informes, y actividades de cierre de pruebas.

Caso de prueba:

* Entrada + acciones + salida
* Salida obtenida== Salida esperada
* Salida obtenida!=Salida esperada

**¿Qué es un plan de pruebas?**

Es un producto formal que define los objetivos de la prueba de un sistema, establece y coordina una estrategia de trabajo, y provee del marco adecuado para elaborar una planificación paso a paso de las actividades de prueba.

**Clasificación**

**Fases de proceso de prueba**

* Planificación de las pruebas
* Diseño de las pruebas
* Implementación de las pruebas
* Evaluación de las pruebas

**Fases de plan de pruebas**

* Pruebas unitarias
* Pruebas de integración
* Pruebas del sistema
* Pruebas de implementación
* Pruebas de aceptación

## Planeación

## Planificación

# Prototipo del sistema

Un prototipo es un modelo (representación, demostración o simulación) fácilmente ampliable y modificable de un sistema planificado, probablemente incluyendo su interfaz y su funcionalidad de entradas y salidas.

Existen varios tipos de prototipos, cada uno de los cuales permite la realización de un tipo determinado de pruebas y con un determinado nivel de realismo. En ingeniería de requisitos, los prototipos más comunes son los siguientes:

* Mock-ups. Se trata de pantallas, típicamente dibujadas a mano en papel, que representan un aspecto concreto del sistema. El soporte que proporcionan a la validación es muy limitado, con la excepción, quizás, de aclarar el interfaz gráfico deseado en casos complejos.
* Storyboards. Son una evolución de los mock-ups, ya que además del interfaz, se muestra la secuencia de acciones, o escenarios, que se deben realizar con el programa. Por ejemplo es habitual que, antes de cerrar un programa, se pregunte si se desea cerrar el fichero de trabajo.
* Maquetas. Una maqueta es una versión simplificada del sistema software deseado. Típicamente, una maqueta representa únicamente el interfaz del sistema y, opcionalmente, las conexiones entre pantallas mediante la utilización de elementos activos como los botones. Si fuera necesaria mayor fidelidad, podrían codificarse partes del sistema, de tal modo que además, del interfaz, el software pudiera ofrecer algunos resultados reales. Ello es lo que se conoce como “prototipo funcional”.

Otros tipos de prototipos:

* Baja Fidelidad: conjunto de dibujos (por ejemplo, una presentación de escenarios) que constituye una maqueta estática, no computarizada no operativa de una interfaz de usuario para un sistema en planificación.
* Alta Fidelidad: conjunto de pantallas que proporcionan un modelo dinámico, computarizado y operativo de un sistema en planificación.
* Exploratorio: prototipo no reutilizable utilizado para clarificar las metas del proyecto, identificar requerimientos, examinar alternativas de diseño o investigar un sistema extenso y complejo.
* Experimental: prototipo utilizado para la validación de especificaciones de sistema
* Operacional: prototipo iterativo que es progresivamente refinado hasta que se convierte en el sistema final.
* Horizontal: prototipo que modela muchas características de un sistema pero con poco detalle. Dicho detalle alcanzará una profundidad determinada, va a resultar especialmente útil en las etapas tempranas de diseño y tiene como objetivo el test del modo de interacción global, al contemplar funciones comunes que el usuario va a utilizar frecuentemente.
* Vertical: prototipo que modela pocas características de un sistema pero con mucho detalle. Va a resultar especialmente útil en etapas más avanzadas del diseño y tiene como objetivo el test de detalles del diseño
* Diagonal: prototipo horizontal hasta un cierto nivel, a partir del cual se puede considerar
* Global: prototipo del sistema completo. Prototipo horizontal expandido que modela una gran cantidad de características y cubre un amplio rango de funcionalidades. Va a resultar muy útil a lo largo de todo el proceso de diseño.
* Local: prototipo de un único componente o característica del sistema de usabilidad crítica. Va a resultar de utilidad en algunas etapas específicas del proceso de diseño.

**¿Que se logra con su desarrollo?**

Modificación del Sistema en Etapas tempranas de su desarrollo, el éxito del uso del prototipo depende de qué tan pronto y con qué frecuencia se reciba la retroalimentación del usuario para hacer cambios y adecuarlos a las necesidades actuales.