**David C. Cortés, Kevin J. Carranza, Ricardo A. Marino**

**No. de Equipo Trabajo: {*4*}**

# [[1]](#footnote-1) INTRODUCCIÓN

En esta sección se describe de manera general en qué consiste este documento y su contenido

# DESCRIPCIÓN DEL PROBLEMA A RESOLVER

En esta sección se debe describir de manera general el problema que se propone resolver en el proyecto de clase. Se debe especificar el propósito u objetivo general.

# usuarios DEL PRODUCTO DE SOFTWARE

En esta sección se deben mencionar las características y clasificación de los usuarios (perfiles/roles) que utilizarán el producto. La clasificación puede ser en función a la frecuencia de uso, grupo de funcionalidades utilizadas, privilegios de acceso y seguridad, nivel de experiencia y otros parámetros.

.

# REQUERIMIENTOS FUNCIONALES DEL SOFTWARE

Los requerimientos funcionales de un sistema, son aquellos que describen cualquier actividad que este deba realizar, en otras palabras, el comportamiento o función particular de un sistema o software cuando se cumplen ciertas condiciones.

Entre los posibles requerimientos funcionales de un sistema, se incluyen:

* Descripciones de los datos a ser ingresados en el sistema.
* Descripciones de las operaciones a ser realizadas en cada pantalla que se presenta.
* Descripción de los flujos de trabajo realizados por el sistema.
* Descripción de los reportes del sistema y otras salidas.
* Definición de quiénes pueden ingresar datos en el sistema.

Título del Proyecto a Realizar

De esta manera, se deben describir las interacciones que tendrán los usuarios con el software.

Cada funcionalidad se debe especificar así:

## Nombre de la funcionalidad

En el título de la funcionalidad, se recomienda utilizar nombres muy descriptivos para cada funcionalidad. No limitarse a nombrarlas “Funcionalidad 1”, en cambio usar por ejemplo: “Autorización de pedido de compra”.

## Descripción: Descripción breve de la funcionalidad.

* *Acciones iniciadoras y comportamiento esperado*:

Secuencia de acciones del usuario y respuestas esperadas del programa para esta funcionalidad.

*Requerimientos funcionales*:

Lista detallada de los requerimientos funcionales asociados a esta funcionalidad.

Para cada requerimiento funcional se establece cómo debe mostrarse el software y cuáles comportamientos debe desempeñar para que el usuario pueda realizar la función que necesita.

Es recomendable prever y describir cómo debe responder el software ante condiciones de error y entradas de datos inválidas.

Las funcionalidades mínimas sobre los datos que se manejen deben prever operaciones de:

* Creación
* Actualización
* Eliminación
* Consulta total de los datos
* Búsqueda parcial de datos
* Ordenamiento
* Almacenamiento

Aunque en otros cursos se estudian estrategias de organización y almacenamiento, en este curso el almacenamiento se requiere principalmente para facilitar las pruebas del prototipo de software. También, para facilitar su implementación, se deja abierta la opción a que se apoyen en el uso de sistemas manejadores de bases de datos, o se haga almacenamiento por archivo, de objetos u otra estrategia que les convenga, siempre que se garantice la implementación y uso de las estructuras de datos vistas en clase en memoria.

IMPORTANTE: En cada una de las entregas para reportar el avance en el desarrollo del proyecto, se especificarán las funcionalidades mínimas y las estructuras de datos mínimas requeridas que se deben implementar. También, se debe presentar un análisis (especialmente, comparativo) breve de la eficiencia de las estructuras de datos usadas.

NOTA: En el siguiente enlace web (URL) puede encontrar una explicación de cómo diferenciar Requisitos Funcionales de los No Funcionales:

<https://reqtest.com/requirements-blog/functional-vs-non-functional-requirements/>

# DESCRIPCIÓN DE LA INTERFAZ DE USUARIO PRELIMINAR

Se debe incluir una descripción general de la interfaz de usuario del software. Si se propone el desarrollo de una interfaz de usuario gráfica, se espera que se usen *Mock Ups* (fotomontajes o figuras) que le permitan al futuro usuario visualizar cómo quedarán los diseños.

NOTA: Existen múltiples herramientas que ayudan en este tipo de tareas. Por ejemplo, podrán encontrar una herramienta útil para esta laboren: https://balsamiq.com/

# Entornos de desarrollo y de operación

En esta sección se debe describe el entorno en el que se desarrollará el software, así como el entorno en el que operará el software en tiempo de ejecución, específicamente el sistema operativo, recursos y el hardware sobre el que operará el software.

# PROTOTIPO DE SOFTWARE INICIAL

Para esta entrega de avance en el desarrollo del proyecto, se debe realizar una primera versión de un prototipo de software funcional de acuerdo con los requisitos que se explican a continuación.

Primero, el software desarrollado se debe registrar en un repositorio de software Github[[2]](#footnote-2). Para facilitar el uso de esta plataforma, se sugiere estudiar el tutorial disponible en:

<https://www.atlassian.com/git/tutorials/learn-git-with-bitbucket-cloud>

Se debe organizar el software en el repositorio de una manera estructurada como se ilustra en la Figura 1:



Figura 1. Estructura del repositorio requerida durante el proyecto.

Tenga en cuenta que por cada entrega se debe generar en el repositorio, una rama diferente.

Adicionalmente, para mantener una versión gráfica de desarrollo del repositorio, se podrá apoyar mediante una herramienta como Sourcetree, disponible en el siguiente URL:

<https://www.sourcetreeapp.com/>

Segundo, en este prototipo se deben implementar por lo menos dos instancias de la estructura de datos tipo **Lista implementada como arreglos**. Para aclarar, aquí Lista se refiere a una estructura de datos lineal, o contenedor de datos en la cual los datos se organizan de una manera lineal (como en una hilera) de tal manera que cada dato puede tener a lo sumo un antecesor y a lo sumo un sucesor.

Tercero, para estas listas implementadas como arreglos se deben soportar por lo menos las siguientes operaciones funcionales:

* Creación de la lista
* Inserción de un solo dato
* Actualización de un solo dato
* Eliminación de un solo datos
* Búsqueda de un solo dato
* Consulta de todos los datos
* Almacenamiento de los datos

Para facilitar su implementación, se deja abierta la opción a que se apoyen en el uso de sistemas manejadores de bases de datos, o se haga almacenamiento por archivo, de objetos u otra estrategia que les convenga, siempre que se garantice la implementación y uso de las estructuras de datos almacenadas en memoria dinámica vistas en clase.

# PRUEBAS DEL PROTOTIPO

Se deben realizar y documentar las pruebas del prototipo para algunos ejemplos (casos) de prueba para las funcionalidades que tomen más tiempo así:

* Escoger entre tres y cinco funcionalidades que sean las de mayor costo computacional en tiempo;
* Para cada funcionalidad se deben realizar pruebas para varios tamaños de datos de prueba (n), por lo menos para los siguientes tamaños:
  + 10 mil datos,
  + 100 mil datos,
  + 500 mil datos,
  + 1 millón de datos y,
  + 10 millones de datos
* Hacer una tabla comparativa de los tiempos que toma realizar las funcionalidades consideradas para los diferentes tamaños de los datos de prueba, recuerde acompañarla de gráficos para ello.

# Análisis de Comparativo

Realice una tabla comparativa entre los tiempos que toma realizar las funcionalidades consideradas para los diferentes tamaños y estructuras implementadas durante pruebas. Grafique el correspondiente análisis comparativo y su respectiva complejidad, recuerde usar la notación Big O para ello.

# roles y actividades

Defina para cada integrante, el rol asignado y las actividades realizadas en la actual entrega.

# DIFICULTADES Y LECCIONES APRENDIDAS

Mencione las dificultades encontradas durante el desarrollo del proyecto. Además, haga alusión a las principales lecciones aprendidas durante el proceso.

1. [↑](#footnote-ref-1)
2. https://github.com/ [↑](#footnote-ref-2)