Clase 07. ¿CÓMO SE PREPARA PARA HACER LAS PRUEBAS?

# Analizar documentación

Introducción

El primer paso para realizar cualquier tipo de testing es hacer las pruebas estáticas. Ahí es cuando comenzamos...

## Tipos de documentos

Existen muchos tipos de documentación para trabajar. Aquí veremos los más importantes y reconoceremos que algunos de estos documentos son enteramente nuestra responsabilidad y otros nos son entregados por otras áreas que los crean.

Los documentos a trabajar son:

* Plan maestro de pruebas
* Plan de pruebas
* Casos de uso
* Casos de prueba
* Criterios de aceptación
* Manuales
* Requerimientos funcionales
* Datos y precondiciones
* Historias de Usuario
* Otros

### Plan maestro de pruebas

El Plan Maestro de Pruebas (o Master Test Plan) es escrito generalmente por el gerente del equipo de QA que establece todos los puntos de control, límites, herramientas y cualquier otro punto importante para el equipo de QA entero.

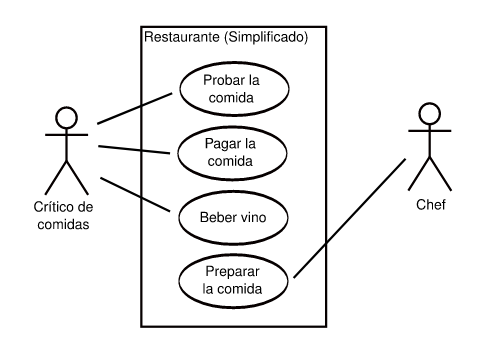
Este documento es lo primero que deberíamos leer al entrar a un equipo nuevo.

### Plan de pruebas

El plan de pruebas toma algunos conceptos del plan maestro, pero en lugar de aplicar el macro, apunta a lo micro de cada proyecto.

Desarrolla límites, puntos de control, seguimientos del proyecto, y otras tareas más.

### Casos de uso

Un diagrama de caso de uso es una descripción de las actividades que puede realizar cada ente que participa del sistema, delimitando de forma simple las tareas que puede realizar y cómo se relacionan entre ellas para con los usuarios.

### Casos de prueba

Un caso de prueba es la herramienta que los testers usaremos para ir guiando nuestro paso a paso al momento de probar.

Son escritos por y para los testers.

Nos apoyamos en otros documentos para escribirlos.

### Criterios de aceptación

Los criterios de aceptación son una práctica en desarrollo que sirve para explicar, antes de la implementación, qué es lo mínimo que se espera del desarrollo.

Esto es escrito por los Product Owners y por lo general se escribe en lenguaje Gerkin (Dado, Cuándo, Entonces)

Dado un usuario.

Cuándo ingresar usuario y contraseña en el login.

Entonces podrá loguearse.

### Criterios de aceptación

Algunos ejemplos de criterios de aceptación:

* Un usuario no puede enviar un formulario sin completar todos los campos obligatorios.
* La información del formulario se almacena en la base de datos de participantes.
* El pago puede efectuarse con tarjeta de crédito.
* Se envía un mensaje de “Registración recibida” luego de recibir la información del formulario.
* Un usuario no puede registrarse con la misma información de otro usuario.

### Manuales

Cuando no hablamos de desarrollo de software específicamente (aunque a veces si exista) está la posibilidad de que se otorgue un manual como guía de trabajo para la realización de pruebas.

Es importante comprender que estos manuales también pueden venir con fallas de usabilidad o falta de descripción.

### Requerimientos funcionales

Son muy amplios pero para poder acotarlos los definiremos con descripciones generales de partes del sistema. Un requerimiento funcional puede ir desde los datos que hay que ingresar, como también un flujo de relación de un sistema.

Estos son escritos por desarrolladores o ingenieros de software, dependiendo el caso.

#### Ejemplo de requerimientos funcionales

* El usuario podrá crear su usuario y contraseña desde una sección llamada “Información de Usuario”. Al crearlo, el sistema deberá notificar con un mensaje que el registro fue exitoso.
* En la sección de Registro/ Información de Contacto el campo teléfono solo deberá admitir caracteres numéricos.
* En la sección de vuelos el usuario podrá elegir entre las siguientes opciones de clases de servicios: Economy Class, Business Class y First Class. Solo podrá elegir una.

### Datos y precondiciones

Está relacionado con requisitos funcionales, pero en muchos casos los datos son una documentación importante para realizar un trabajo como usuarios y contraseñas o saber qué datos pueden ingresar o no.

Las precondiciones son un complemento ya que en algunas situaciones, las pruebas pueden estar ligadas a faltantes o algún otro dato que nos puede ser de utilidad al momento de probar.

### Historias de usuario

Una historia de usuario es un pedido informal de una función de software escrita desde el punto de vista del usuario final.  
Este tipo de documentos es escrito por los Product Owners.

Como cliente

Quiero ingresar con usuario y contraseña

Para poder loguearme

### Otras pruebas estáticas

Existen muchos tipos de documentación que pueden surgir de acuerdo al proyecto que trabajemos. Los mismos se encuentran dentro de los “requisitos funcionales” ya que son distintos diagramas de metodologías funcionales.

Lo importante es siempre solicitar la mayor cantidad de documentación posible, previo a comenzar con nuestras pruebas.

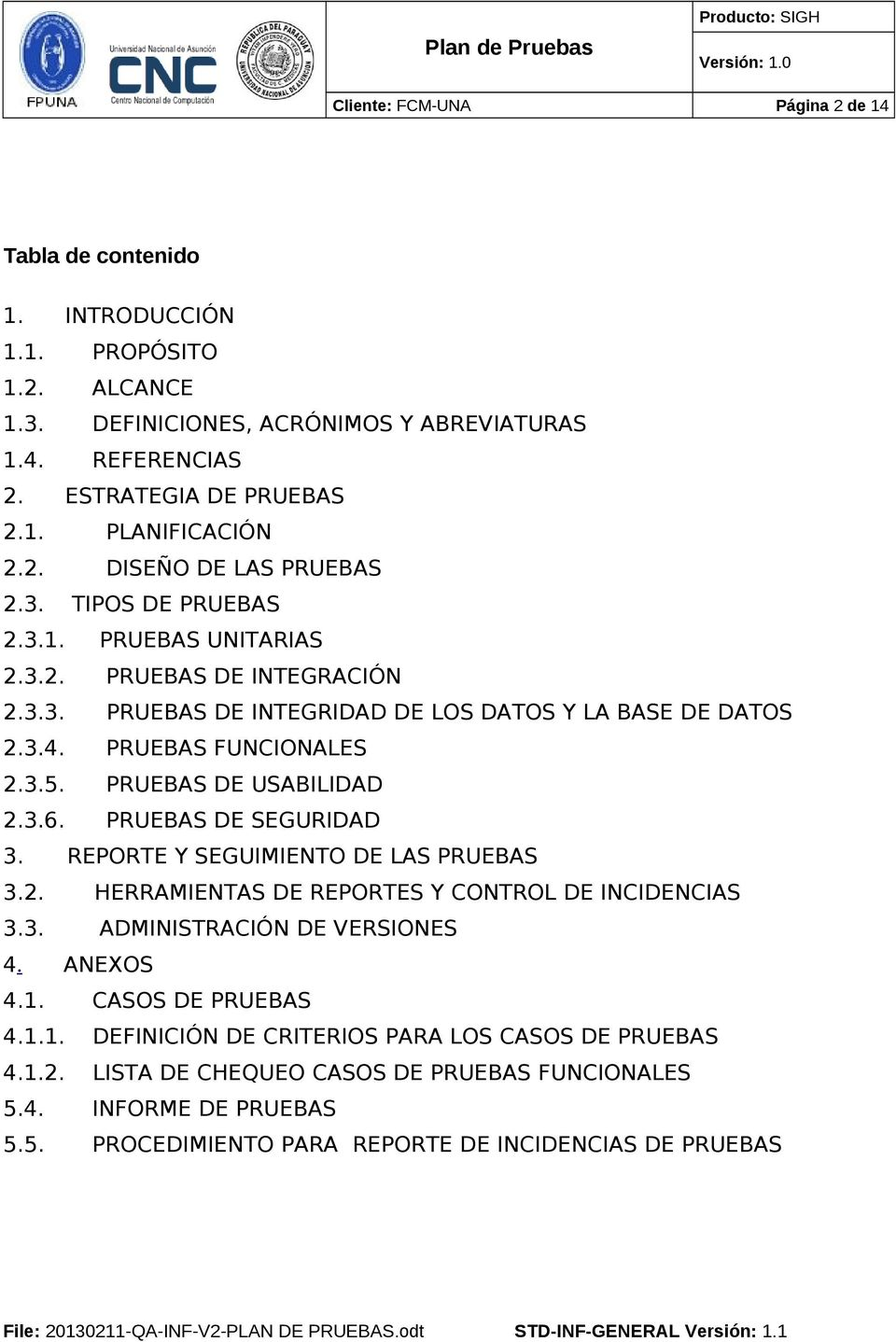
## Armar un plan de pruebas

### Plan de pruebas

Describe las actividades de prueba para proyectos de desarrollo y mantenimiento.

La planificación depende de la política y la estrategia de prueba de la organización, los ciclos de vida de desarrollo y los métodos utilizados, el alcance de la prueba, los objetivos, los riesgos, las restricciones, la criticidad, la capacidad de ser probado y la disponibilidad de los recursos.

### Ejemplo de plan de pruebas



Fuente: [plan de pruebas del sistema de gestión hospitalaria](https://docplayer.es/12255323-Plan-de-pruebas-sistema-de-gestion-hospitalaria-plan-de-pruebas-file-20130211-qa-inf-v2-plan-de-pruebas-odt-std-inf-general-version-1.html)

### Planificación en el tiempo

A medida que el proyecto y la planificación de la prueba evolucionan, se dispone de más información y se pueden incluir más detalles en el plan de prueba.

La planificación de la prueba es una actividad que se realiza de forma continua a lo largo de todo el ciclo de vida del producto.

### Alcance del plan

La planificación puede documentarse en:

* Un plan maestro de prueba.
* Planes de prueba separados para los niveles de prueba, tales como las pruebas de sistema y las pruebas de aceptación.
* Tipos de prueba separados como la prueba de usabilidad y la prueba de rendimiento.

### Actividades de planificación

Al momento de realizar el plan de pruebas, es necesario hacer una lista de actividades que marcan los límites, tiempos y control de todo el sistema de pruebas.

1. Determinar el alcance, los objetivos y los riesgos de la prueba
2. Definir el enfoque general de la prueba
3. Integrar y coordinar las actividades de prueba en las actividades del ciclo de vida del software
4. Tomar decisiones sobre qué se va a probar, las personas y otros recursos necesarios para las actividades de prueba.
5. Establecer un calendario para las actividades de análisis, diseño, implementación, ejecución y evaluación de la prueba, ya sea en fechas particulares o en el contexto de cada iteración.
6. Selección de métricas para monitorizar y controlar la prueba.
7. Elaborar un presupuesto para las actividades de prueba.
8. Determinar el nivel de detalle y la estructura de la documentación de la prueba.

El contenido de los planes de prueba varía y puede extenderse más allá de los temas previamente identificados.

## Armar los casos de pruebas

### Casos de pruebas

Este es momento de entrar en detalle respecto qué son los casos de prueba, cómo se arman y el papel que cumplen, entre otros.

La documentación de los casos de prueba irá variando acorde a la herramienta o plataforma que tengamos para documentarlos.

Trabajaremos con algunos de los términos más globales que podemos tener y los definiremos.

* Set de condiciones o variables bajo las cuales un tester determina si el software funciona correctamente o no.
* Buena definición de casos de prueba nos ayuda a reproducir defectos

### Título y carátula

Por lo general, tendremos una carpeta, una carátula o título que englobe la sección a probar. Esto vendrá acompañado del nombre del proyecto, el tester que armo los casos, si existen modificaciones y sus fechas, responsable de ejecutar los casos.

Cuando se trata de proyecto a clientes externos, se puede agregar el nombre del cliente y alguna referencia a ese proyecto.

### Numeración de los casos

Para mantener un orden, cada caso de prueba va a contener un número que lo identifica y, en paralelo, cada paso dentro del caso, su propia numeración. Esto sirve para poder identificar trazabilidad para errores y conocer los estados.

### Cuerpo de los casos

El cuerpo de los casos de prueba es la parte más importante. Ahí iremos escribiendo el paso a paso de cómo esperamos que se comporte el sistema.

Su descripción será del tipo “acción realizada” y “resultado esperado”. Cuando sabemos que el resultado esperado es incorrecto, ingresamos un bug.

### Otros datos

Los casos pueden ser positivos y negativos, lo cual deberá ser ingresado en algún lugar el dato. También se le agrega una prioridad al caso, como para saber su relevancia dentro del proyecto.

Otro dato que puede sumarse es el tipo de acción entre la acción y el resultado obtenido.

### Estados del caso

Hace referencia a si se logró completar, algo hizo que falle, este bloqueado o si todavía no se realizó ese paso del caso.

En general se los denomina como:

* Paso
* Bloqueado
* Falló

### Más opciones

Muchos datos son coordinados entre el mismo equipo. A veces, se sugiere agregar en qué dispositivos y sistemas operativos se realizan. Si se realizó algún cambio, es óptimo agregar la fecha de modificación, lo mismo si se encontró alguna observación extra.

### Casos de pruebas negativos

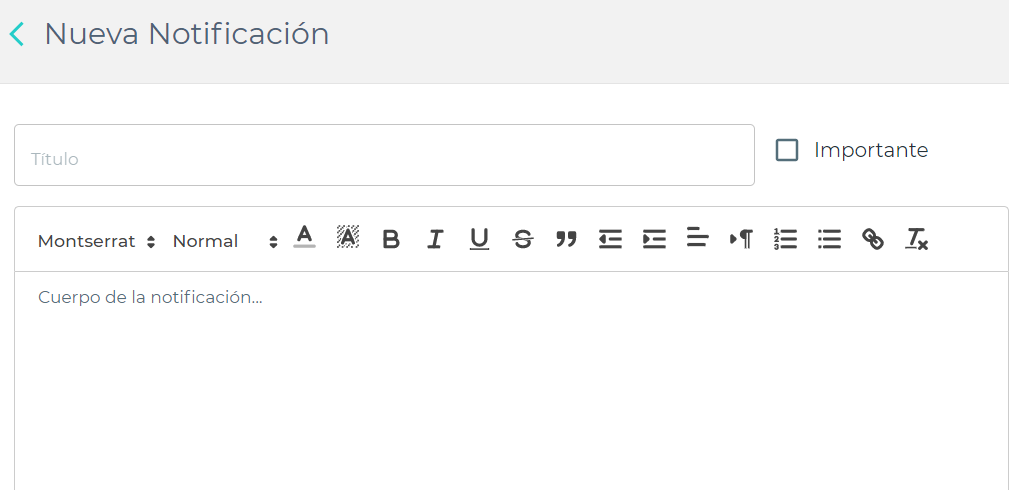
* Casos de prueba que se utilizan para validar a la aplicación contra datos no válidos
* Verifican si los errores se muestran al usuario donde se supone que deben hacerlo

### Casos de pruebas positivos

* Son generalmente la primera forma de prueba que un tester realizaría en una aplicación (camino feliz).
* Es el proceso de ejecutar escenarios de prueba que un usuario final ejecutaría para su uso.
* La prueba positiva implica ejecutar un escenario de prueba con solo datos correctos y válidos.

#### Ejemplos

Dada la siguiente pantalla de “nueva notificación”



**Tenemos los siguientes requerimientos:**

* El cuadro de texto del título es un parámetro obligatorio.
* El cuerpo no es obligatorio.
* El cuadro de título sólo puede tener caracteres a-z y A-Z. No se permiten números, caracteres especiales.
* El título puede tener un máximo de 10 caracteres.

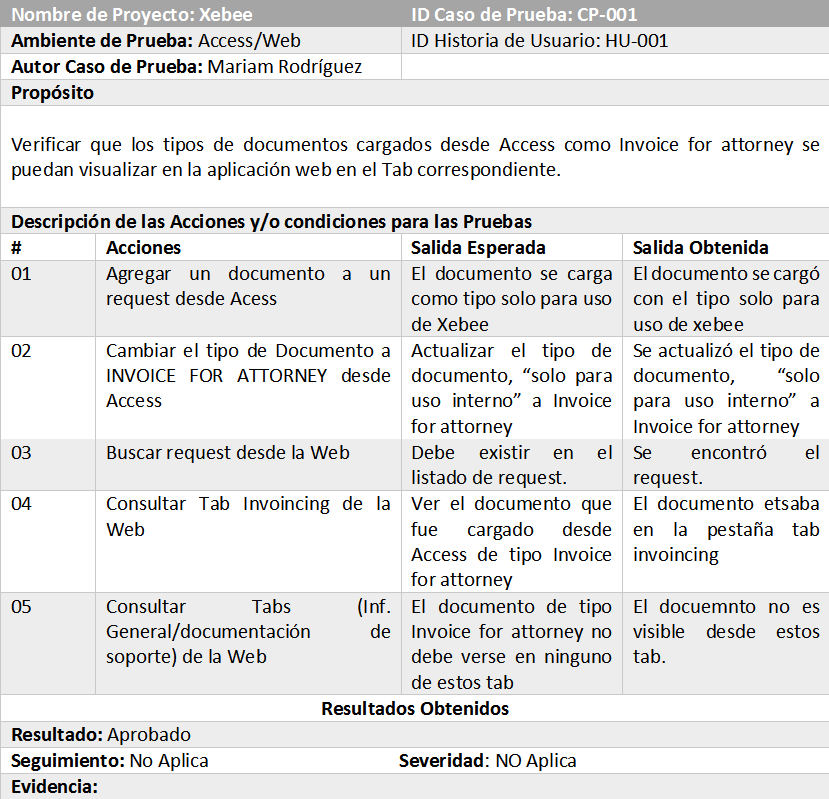
##### Casos de prueba positivos:

* ABCDEFGH (validación de mayúsculas dentro del límite de caracteres)
* Validación de minúsculas abcdefgh dentro del límite de caracteres)
* aabbccddmn (validación de límite de caracteres)
* aDBcefz (mayúsculas combinadas con validación de minúsculas dentro del límite de caracteres)

##### Casos de prueba negativos:

* ABCDEFGHJKIOOOOOKIsns (el título supera los 10 caracteres)
* abcd1234 (el título tiene valores numéricos)
* Sin titulo proporcionado
* snd12-\*/-^\_ (el título que contiene caracteres especiales)

### Ejemplos de casos de prueba



Fuente: [expertosmicrosoftaccess.com](https://expertosmicrosoftaccess.com/blog/2015/08/18/como-hacer-casos-de-prueba-y-asegurar-la-calidad-del-software/)

## Repaso

1. Ingresamos la información principal del caso de prueba: id, nombre, objetivo y tipo.
2. Ingresamos el resultado que esperamos al ejecutar el caso, las precondiciones y datos de entrada que utilizaremos.
3. Escribimos los pasos de ejecución y los resultados obtenidos (opcional).
4. Tipificamos su criticidad (alta, media y baja).
5. Agregamos toda la información extra de cada caso.
6. Probamos y ejecutamos el caso de prueba.
7. Definimos un estado.

# Herramientas

Las herramientas para manager de casos de prueba se conocen como “Test Management Tools” y existen muchas en el mercado.

En general, su uso varía entre el costo, pero gran cantidad de herramientas, o algunas gratuitas y open source pero con límites.

## Test management tools

Destacamos algunas de las herramientas más usadas:

* Testlink
* Zephyr by Smartbear
* TestRail by IDERA
* SpiraTest by Inflectra

## Glosario

**Product Owner:** Dueño de Producto. Rol de Scrum. Responsable de maximizar el valor del producto y el trabajo del Equipo de Desarrollo.

**Gherkin:** es un lenguaje diseñado para resolver un problema muy específico. Este problema se basa en la comunicación entre los perfiles de negocio y los perfiles técnicos a la hora de trabajar bajo un enfoque BDD.

**Requerimientos Funcionales:** descripciones generales de partes de un sistema

**Historias de Usuario/User Story:** las historias de usuario, son pequeñas descripciones de los requerimientos de un cliente.

**Casos de Prueba:** Set de condiciones o variables bajo las cuales un tester determina si el software funciona correctamente o no.

**Casos de Prueba Positivos:** La prueba positiva implica ejecutar un escenario de prueba con solo datos correctos y válidos.

**Casos de Prueba Negativos:** La prueba negativa implica ejecutar un escenario de prueba con solo datos incorrectos, no válidos y nulos.

#### ¿QUIERES SABER MÁS? TE DEJAMOS MATERIAL AMPLIADO DE LA CLASE

* [Cómo Crear un Plan Pruebas de Software](https://www.youtube.com/watch?v=0anZpU5W0Z8&t=402s) | Quality-Stream
* [How to Choose a Test Management Tool](https://www.inflectra.com/Ideas/Whitepaper/How-to-choose-a-Test-Management-Tool.aspx) | inflectra

#DEMOCRATIZANDOLAEDUCACIÓN