



# Ingenería y Calidad de Software

## SCM - Herramientas de SCM

Implementar una estructura de repo propuesta para el Contenido de la Catedra

- Estructuras de directorios/carpetas: jerarquia de directorios
- Identificar los ICs
  - Nombre del Item
  - Regla de nombrado: unica para el item  
opcional: numero, fecha, cadena de texto
  - extension:
- Definicion de los posibles ICs q se pueden identificar/generar a lo largo de la cursada
- Definir, a mivel de equipo, un criterio para la creacion de la Linea Base:  
escribir en un parrafo diciendo que la linea base se crea en "tal momento"  
o cuando " se "cumplan ciertas condiciones"
- Cantidad de Items considerables q cumplan con las reglas de nombrados

## Linea base

nombre unívoco, no se establece con fecha y hora, se establece cuando se cumplen ciertas condiciones

"Se establecerá la Linea de Base del proyecto cuando se realice la devolución (entrega post-corrección) de los primeros 3 Trabajos Prácticos - ya sea evaluable o no evaluable- denido que estos tienen la aprobación (nota) del cliente (profesores)"

## User Stories

Técnica para reconocer requerimientos ágiles de **usuario** (alineado con algún requerimiento de negocio → el req de usr es medio para cumplir un req de neg). Agil pq esta técnica cumple principios del Manifiesto Agil

Rol a → Producto Owner: persona no técnica a cargo de la US (quien sabe sobre, no quien la escribe). Debe hacer foco en las Necesidad de Usr que van a satisfacer las Necesidad de Neg, ya que se supone que conoce el negocio y será el eslabón entre el Usr/Neg y el Equipo de desarrollo

Técnica para reconocer requerimientos ágiles. Busca traducir una necesidad del usuario (funcionalidad de sistema) de negocio. Interpretarla a nivel técnico pero describirla con el lenguaje del usuario.

"el enfoque de requerimientos ágiles debe estar adherido al Man Ag - cumplir con sus valores y principios"

principio 1: Nuestra mayor prioridad es satisfacer al cliente.

principio 6: El método más eficiente de comunicar información es conversaciones cara a cara.

## Las 3 C de las user stories

- Conversación: (lo más importante) reunión con el usuario para de allí extraer la verdadera necesidad que posteriormente será una funcionalidad
- Card (o "Tarjeta")
  - Frase verbal: frase corta que identifica qué permite saber de qué se está hablando (

- Descripcion: Quien(rol a cardo), Que(valor de negocio), Para que: Como [rol] quiero [que] para [valor de negocio]  
nota: valor de negocio es uno por user stories (no puede ser compuesto (2), habra q separar)
- Criterios de aceptacion: se describe una parte de la conversación ya que ahí se ondiciona por las cuales el cliente aceptara la US
- Pruebas de usuario: luego de los casos de prueba, pequeña experiencia del usuario con el sw donde este acepta o no, esa parte de funcionalidad implementada descripta en la US
- Confirmación:
  - Pruebas de aceptacion: conjunto de escenarios posibles a los que se somete el sw implementado, si el P.O. acepta, posteriormente pasara a produccion

### Caso de prueba

Examinar la calidad del sw, detallando paso por paso ,con ejemplos, de lo que haria el cliente en una situación

## Criterio de Ready

Conjunto de condiciones que se deben cumplir para que la US esta lista para Implementar, es decir, para que el equipo tecnico la desarrolle

INVEST: un criterio aplicable

- Independiente: en un momento de desarrollo dado, tengo independencia entreUserStories
- Negociable: que la US posee cierta flexibilidad a cambios propuestos por parte del equipo de desarrollo con el Product Owner, este es el que finalmente aprueba o no.
- Valuable: "o que da valor": que las US ademas de su fucionalidad, tiene que darle valor al negocio
- Estimable: la posibilidad de definir tamaño (de lo que pide la US) y duracion (que llevaria desarrollarla). Si no se puede, candidata a dividir en Us mas pequeñas

- Small: completar la totalidad de la US en una iteración de trabajo definida (2 o 4 semanas)
- Testable: que lo desarrollado en base a la US sea fácil de aplicar test y que permita la calidad del software

## Empaquetado u organización de US

- Temas
  - Epicas
    - User Story 1: dos US pueden ser parte de una epica
    - User Story 2:

## Estimaciones

### Según el enfoque

- Tradicional:
  - Tamaño: (líneas de código → módulos → paquetes):
  - Esfuerzo: Horas personas lineales horas efectivas de trabajo de
  - Calendario
  - Costo
- Ágil
  - Tamaño: Cantidad de características del Producto → US

### Cómo se realiza

El equipo de desarrollo acuerda una US canónica (patrón). Gralmente la US más pequeña / "fácil". Story Point: peso que se le asigna a la US por compararla con la US Canónica

Escalas:

- Escala decimal: del 1 al 10.

- Exponencial:  $2^n$
- Fibonacci: 0, 1, 1, 2, 3, 5, 8

Asignación de peso segun:

- Complejidad: inherent
- Esfuerzo: despendiendo de la capacidad individual y grupal del equipo del desarrollo
- Incertidumbre: que tan informado y cuanto experiencia con respecto al conocimiento del dominio,

## PRÁCTICO 1 - REQUERIMIENTOS ÁGILES – User Stories

### Conversación

Identificación de los

Identificación Técnica	Conversión a lenguaje del Usr
Consultar gastos	Ver planilla de gastos
Registrar gasto	Guardar gasto
Reg un tipo de gasto	Reg un tipo de gasto
Registrar usuario	Registrar usuario
Iniciar sesión	Iniciar sesión
Cerrar sesion	Cerrar sesion
Filtrar gastos	Filtrar gastos
Modificar gasto? abmc?	Modificar gasto
Eliminar gasto? abmc?	Eliminar gasto
Registrar responsable de gasto?	Agregar responsable de gasto

## Gestión de Productos

## Tipos de Productos de Sw

- **Software enlatado (o estándar, empaquetado):**
  - Se desarrolla para un mercado masivo.
  - Ejemplo: Microsoft Office, navegadores web, sistemas operativos.
- **Software a medida (o dedicado, personalizado):**
  - Se crea específicamente para un cliente o empresa.
  - Ejemplo: un sistema de gestión de stock hecho para una pyme.

## Explicando MVPs, MVFs, MMFs con el Dinosaurio Lean/Agile

- Uvp: Disparador/ Hipotesis de necesidad y funcionalid/es que satisfaceran esas necesidades en el mercado
- Mvp: producto minimo viable → contiene un conjunto de funcionalidades
  - Mvf: los mismo que mvp pero de una sola funcionalidad
  - Mvp[vesion]: versiones de mvp q pueden ser incorporadas al mvp inicial por variaciones o feedback de mercado
- MMP: salgo al mercado con el producto que posee ese conjunto de funcionalidades (Mvp comecializable)
- MMF: salgo a mercado con una sola funcionalidad (feature) (Mvf comercializable)
- MMR: realese: version nueva que se pone en produccionx

## PRÁCTICO 2 - REQUER. ÁGILES – User Stories y Estimaciones

### Conversación

Identificación de los

Nº	Indentificación Técnica	Conversión a lenguaje del Usr
1	Consultar mapa interactivo	Ver mapa interactivo

2	Registrar usuario	Registrar usuario
3	Iniciar sesion	Iniciar sesion
4	Cerrar sesion	Cerrar sesion
5	Consultar horarios de alimentacoin	Ver horarios de alimentacoin
6	Registrar inscripcion actividad	Inscribirse a actividad
7	Registrar compra de entrada	Comprar entrada con efectivo
8	Filtrar actividades	Ver actividades

Nota!: siempre verificar el criterio INVEST

- Comprar entrada con efectivo y Comprar entrada con tarjeta: son criterios de aceptacion
- Los ABMC son candidatas a analizar para ser tomadas como Canonicas
- Analizar "pocresos automaticos" que son desencadenados por acciones del usr

## Planificación de un proyecto de Software

Recordando las 4 P, pero haciendo enfasis en el proyecto:

P de Proyecto:

Caracteristicas

- Resultado unico
- Fecha de inicio y fin
- Elaboracion
- Tareas interrelacionadas

# ¿QUÉ IMPLICA LA PLANIFICACIÓN DE PROYECTOS DE SOFTWARE?

- Definición del Alcance del Proyecto
- Definición de Proceso y Ciclo de Vida
- Estimación
- Gestión de Riesgos
- Asignación de Recursos
- Programación de Proyectos
- Definición de Controles
- Definición de Métricas

## Objetivo del proyecto

- Claro: sin ambigüedades y que todos los involucrados en el proyecto entiendan lo mismo por "objetivo"
- Alcanzable: que con los recursos que se poseen es posible realizar lo que se plantea

## Alcance del proyecto

"es el trabajo, solo el trabajo que es necesario para cumplir con el objetivo"

"no son las cosas que hace el producto (reqs), es el listado de cosas que se deben realizar (estilo paso a paso) para cumplir uno de

## Definir el proceso y el proceso de vida del proyecto

Dependiendo del ciclo del vida de proceso (dependiendo del tipo que sea: Empírico o Definido) que se elija, el proyecto adapta ese proceso, definido teóricamente, estableciendo cuánto de qué actividad se realizará, y en el caso que sea necesario, la omisión de alguna.

Las actividades, que producto de la adaptación no se llevan a cabo, es bajo el concepto de "esta actividad No Aplica", es decir, bajo alguna justificación; no por libre albedrio

## Definir el equipo de trabajo del proyecto

El proceso define los roles y responsabilidades, en el proyecto se instancia. "el proceso se define en termino de roles"

ej: el product owner: es el responsable de...

En el proyecto se define qué **persona** tendrá qué rol. En el proyecto se le asigna a una persona el/los roles

Una persona puede tener más de un rol, por lo tanto, más de una responsabilidad

ej: el product owner será Juan Jose W

## Estimación (enfoque TRADICIONAL)

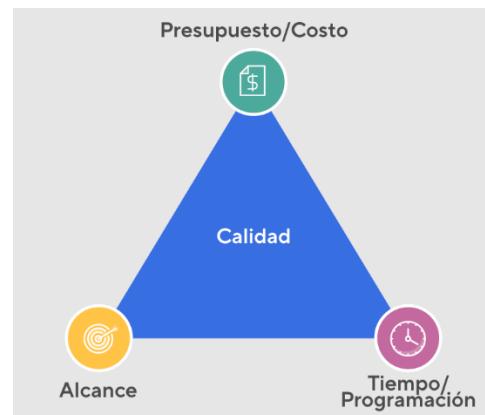
Predicción. Un supuesto que siempre tiene una probabilidad asociada

- Tamaño del Producto → asociado a Obj, Alcances
  - Unidad de medida de estimación: Casos de Uso x Complejidad
    - ej.: n cantidad de CU simples; m cantidad de CU de mediana complejidad, k cantidad...
- Esfuerzo del Proyecto
  - Unidad de medida de estimación: Horas personas lineales
    - Cantidad de horas netas/efectivas que se dedican a la construcción/desarrollo de los CU
    - ej.: n cantidad de horas para un Junior + m cantidad de horas para un Senior
- Tiempo y calendario del Proyecto
  - Análisis de actividades secuenciales y en paralelo
  - Contemplar las horas no productivas en la jornada laboral
- Costo
  - Ultimo ítem ya que todo lo anterior condiciona la monetización
  - costo base:
    - cantidad de horas definidas en el esfuerzo
    - precio hora del personal: puede ser horas ingeniero planas o por rol
  - ganancia: fuera del proyecto (objetivos comerciales)

- Recusos Criticos

licencias de sw, hardware especifico, otros

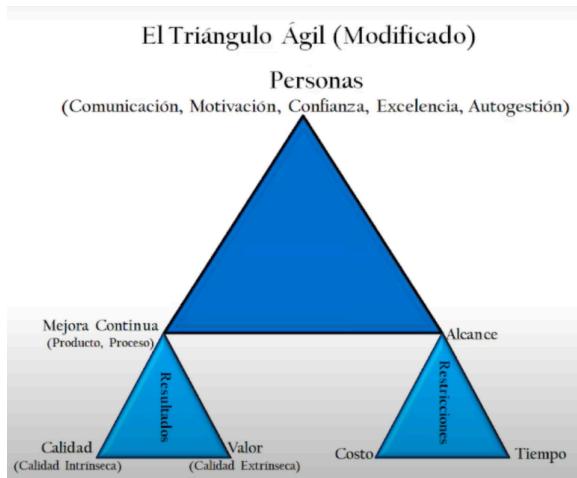
### Triangulo de Hierro (enfoque Tradicional)



### LA RESTRICCIÓN TRIPLE



### Modificación hacia un Enfoque Agil



## Proceso Unificado de Desarrollo

con ciclo de vida iterativo

- En el enfoque Tradicional

De alcance fijo: se trabaja el tiempo que sea necesario hasta que se obtenga esa funcionalidad de sistema requerida

- En el enfoque Agil:

De duracion fija: se desarrolla todo lo relacionado a la funcionalidad en un tiempo fijo

## Calendarizacion

Diferida

anticipada

## Riesgos

## Seguimiento

# PRÁCTICO 3 - REQUER. ÁGILES – User Stories y Estimaciones

Listado de Requerimientos

Nº	Conversión a lenguaje del Usr	Identificación Técnica
1	Comprar ropa	
2	Vender Ropa	
3	Comprar ropa	
4	Filtrar ropa	
5		
6		
7		
8		

## Scrum

Framework para gestión ágil de proyectos para poder generar de manera adaptativa soluciones a complejas

No dice cómo construir software

Adhiere a todas las prácticas que dicta el Manifiesto Ágil (4 Valores; 12 Principios)

Como sigue el Manifiesto Ágil es aplicado a los Procesos Empírico, por consecuencia, también es guiado sus 3 pilares:

- Transparencia (hacer que la información sea accesible y comprensible),
- Inspección (evaluar periódicamente el progreso y los artefactos) y
- Adaptación (ajustar el plan o el producto basándose en los hallazgos de la inspección).

Categorías de responsabilidad (anteriormente llamadas roles)

Responsables de crear los productos de trabajo en el proyecto

- Scrum Master
- Product Owner
- Desarrolladores

## Principio N° 4 del MA: Tecnico y no tecnicos trabajando en conjunto

1. P.O.: No tecnico → eslabon entre los usuarios/clientes y el equipo de desarrollo. Si no existe, se inventa uno, este es llamado Proxie (asume el rol del P.O.). Un P.O. por producto
2. S.M.: Tecnico: Lider de servicio: eslabon entre el P.O. y el equipo de desarrollo
  - a. que las reglas del **Scrum** se cumplan
  - b. organizar reuniones
  - c. gestionar y disponibilizar los recursos necesarios para
  - d. encargado del Cómo se

## TimeBox/Eventos/Ceremonias

### Planning

### Daily

Sprint: es una iteracion de duracion fija. Al final, genera un incremento del producto. Contenedor de todas las actividad. Duracion maxima de **un mes**

### TimeBox:

### Product Backlog

La cantidad que caracteristicas minimas necesarias que permita realizar la primera iteracion completa, osea, el primer Sprint.

PBI (product backlog items): de forma generica ya que Scrum recomienda que estos items sean User Stories. Sin embargo, puede usarse otras cosas dif a las US

#### Elementos que puede contener el PB

Lista priorizada de caractiriticas que debe tener.

- US (User Stories): requerimiento de usuario que sigue un req de negocio
  - Epica: "una US grande"
  - Tema:

- Deuda Técnica: un proyecto con deuda técnica funciona. Ej.: harcodado, nivel malo de acoplamiento y cohesión
- Defectos

### Definition of Done

En el contexto ideal de Scrum con equipos Autogestionados, Motivados, etc. donde una persona podría realizar una US completa

To	Doing	Done
US1 (sp:3)		
US1 (sp:5)		

Si la US se desagrega en Tareas, al completarlas se logra que esa US cumpla con DoD

### Estimación:

Complejidad con respecto a pagos y conexión

### Nivel de incertidumbre:

A nivel negocio

A nivel técnico

### Esfuerzo:

Complejidad:

Justificación:

## Canonica

Se puede estimar la canonica con Sp de 1, esta es:

- US Patrón: US parámetro para la estimación de las otras US en base a la comparación a la US a estimar con esta ya estimada de Sp1
- Al asignar una Sp a la US: se tiene q justificar el Por qué, para uno (capacidad técnica, experiencia previa), la US canonica tiene 1 Sp o la US estimada tiene la Sp q obtuvo

- Se puede asignar

## Producto

alcance del producto

que se le va a entregar al cliente, eso hay q definirlo

en el enfq tradicional → ESR (def objetivos, def alcances y **listaba requerimientos**)

en el enfq agils → tomaba un puñado de reuques y los trabaja en un proyecto  
donde no hay necesario

## Listado de caracteristicas

## Proyecto

“el proyecto de desarollar el producto de autogestion”

cumple con un objetivo que depende...

esta relacionado con el objetivo del producto (por eso solo menciona)  
en la culmi

alcance

todo el trabajo y solo el trabajo ncecesario para poder cumplir con el objetivo  
del proyecto

es decir, el conjunto de actividades/tareas

- gestionar de proy
- reques
- diseño
- prueba

el alcance del producto condiciona el alcance del proyecto

mvp: una idea de producto de la cual se plantea una hipotesis (si el mercado (gente/ususrios) pretede usarlo)

prototipo, video, ppt

mmp: producto minimo comercializable: q se puede lanzar al mercado (pago o gratis)

## Testing

### Tradicional

Recordando las etapas del Proceso de Desarrollo de Sw (PUD)

- Requerimientos
- Analisis
- Diseño
- Implementacion
- PRUEBA/TESTING
- Despliegue

Dependiendo del nivel de especificación o detalle (decisiones) que se haya hecho en una Etapa/Ambiente del Desarrollo de Sw, la siguiente tendra mas o menos responsabilidad de detallar (tomar esas decisiones o las faltantes) lo necesario para poder cumplir

### Proceso de Testing

Etapas	Artefacto	Se puede realizar a partir de	Producto de la etapa de Pud
Planificación	Plan de pruebas	Requerimientos (ers)	Requerimientos
Diseño	Casos de Prueba	Requerimientos (ers)	Requerimientos
Ejecución	Reporte de defectos	Release (versión de producto)	Implementación
Cierre			

A partir de que los requerimientos están mediamente definidos, se puede adelantar con el Plan de pruebas y Diseño de Casos de prueba

Luego, a partir de una o la primera versión de producto (salida de la Implementación del PUD), se puede llevar a cabo la Ejecución de los Casos de prueba para poder obtener un Reporte de defectos

## Tipos de testing

- AdHoc
 

no metodológico, no sistemático, no tiene proceso, no se basa en casos de prueba.

Testing al estilo Fuerza bruta o Explorativo
- Metodológico
 

todo lo contrario del adhoc: sigue

## Vinculación de nivel de testing con nivel de proceso de desarrollo tradicional

Etapa del Proc Desarr Trad (PUD)	Niveles de Testing/Prueba Teórico	Rol
Requerimientos		
Análisis		
Diseño		
Implementación	Pruebas Unitarias	Desarrollador
Prueba	Pruebas de Integración Pruebas de	Rol de Tester en el equipo de desarrollo

	Sistema/Version	
Despliegue	Pruebas de Aceptacion de Usuario (UAT)	Usuario del sistema: Product Owner

## Caso de prueba

se prueban/llevan a cabo las pruebas de usuario (posibilidades/caminos posibles de la US)

- Identificacion: nombre de la funcion de prueba: generalmente un nombre largo muy preciso/especifico

Se debe conocer el producto, es decir, debo trabajar a con los Requerimientos para Diseñarlos, y para eso, los reqs deben estar

Descripcion de pasos/acciones de un ESCENARIO: que se deben hacer indicando que variables se seleccionaran y con cual valor deberan completar (condiciones de seteo)

Imp: sino se cumple con lo que dicta el CdP este (falla)

- qué acciones se ejecutaran
- que variable y con cual valor se completa o ingresa
- con lo anterior, que resultado se espera del caso de prueba
- precondiciones o contexto de sistema: que cosas deben estar ya disponibles en el sistemas para que caso de prueba pase

Ejemplo: Caso de Prueba correspondiente al Caso de Uso: Iniciar sesion

paso1: Seleccionar la opcion Iniciar sesion

paso2: ingrese el usuario genavillegas e ingrese la contraseña 1234

Cuando se deja de probar?

existen diferentes estrategias

- Defectos: cero defectos bloqueantes y graves y una cantidad n de los otros defectos.
- tiempo de testeo
-

estas estrategias son plasmadas en la Planificación de Pruebas  
el testing no asegura la calidad, pone en evidencia los defectos de sw.  
la calidad de un sw es una cuestión adicional: construyo sw con calidad:  
actividad transversal a toda y cada una de las actividades del desarrollo del sw,  
sea tradicional o agil

## Verificación y Validación

### Error y Defecto

Error: cosas mal hechas (problemas de código) en el workflow de Implementación en el PUD

Defecto: Errores sin corregir que pasaron del workflow de Implementación

	Error	Defecto
Dónde ocurren	En la etapa de Implementación del PUD	Se descubren en las etapas Prueba y Despliegue del PUD
Cómo se detectan	Revisiones técnicas (se escribe código y se lo revisa)	Ejecutando los Casos de Prueba
Cómo se solucionan	Re-escribir el código	Informar el defecto, delegar a Desarrollo donde se aplica Re-trabajo
Costo	Medio	Alto
Probleáticas		En pruebas de aceptación <b>fallidas</b> a veces se deben a mala definición de los requerimientos: Sistema construido correctamente pero Sist NO correcto

## Ciclo de prueba y Estrategia de pruebas

### Estrategia de pruebas

- Sin retroceso
- Con retroceso

## Requerimientos - Pruebas y Tipos de pruebas

En que Ambiente del testing	Que se prueba	Tipos de Prueba asociada	En que consiste	Ejemplo
	Funcional	Reqs de operacion normal	Se prueba el sistema en un curso normal	Las pruebas de usuario que (pansa) de la US
	Funcional	Reqs negatica	Se prueba el sistema haciendo cosas q no se podrian hacer	Incribirse a una materia q no puedo por correlativas
	Funcional	Proceso de negocio		
	Funcional	Documentación	Usuario y de Config de sistema	
	No Funcional	Performance		
	No Funcional	Interfaces		
	No Funcional	Escala completa		
	No Funcional	Stress	Probar e limite soporta el sistema, que pasa cuando "se clava" y en cuanto tiempo se "levanta" de nuevo el sistema	
	No Funcional	Seguridad	Buscar vulnerabilidades en el sistema	
	No Funcional	Accesibilidad	Si estan cubiertas todas o la mayoria de	

las  
posibilidades  
de las formas  
de acceder a  
las  
funcionalidades  
del sistema

## Agilismo

cambia el paradigma de trabajo.

se coloca en detalle en el lugar/estapa/ambiente donde corresponda/sea indispensable:

- En la US no se detalla/plasman valores de las variables
- En los Casos de Prueba se plasma "esa decision"/ese nivel de detalle asociado a los variables, valores, contexto de sistema, etc.

incrementa la calidad del software ya que aumenta la probabilidad de que el sw sea el correcto

## TDD - Test Driven Development

técnica de testing agil que resuelve las pruebas unitarias, lo que hace el desarrollador

"desarrollo guiado por testing"

- detección temprana y eficiente de errores
  - scope correcto: programar lo justo y necesario de lo que el cliente quiere
- funciones como caja negra

la salida de tdd: funcionalidades de sistema correctamente escritas

## PRÁCTICO 6 - User Stories y TDD

## **Eco Harmony Park - Inscribirme a Actividades (Par)**

- Uso de la tecnica de US: preguntar al cliente (profes) las dudas sobre la US
- Aplicacion de la tecnica de TDD
  - Desarrollo de la pruebas unitarias: las PU deben estan antes del codigo desarollado
  - Correcto uso del repo: primeros los commits de las las pruebas y luego el codigo
- Documento de estilo de codigo o buenas practicas: PEP8 (python)
- La funcionalidad debe andar/correr
- Interfaz de usuario
- Base de datos
- No se corrige la aplicacion de patrones de codigo
- Simular la conexion con algun externo: Mock o Mockito

## **Casos de prueba**

Recordar la Severidad de los Defectos:

Baja

Media

Alta

Critica

Bloqueante : (no se suelen presentar en el testeting)

Niveles de pruebas

Pruebas de usuario

Casos de pruebas

Regresión: la ejecución de un conjunto de casos de prueba completo

Problema: introducción de errores nuevos

Se dan tamb en la pruebas de integración para verificar q no se haya roto  
nada duerante esta etapa

CI/CD:

Github/Gitlab: tiene plataforma para automatizar los CI/CD. Otros: Jenkins,  
Docker

Utilizar un esquema de branching: permitiría hacer un merge al main  
automatico

Es un evolución en la gestion de configuracion

Consejos al pasar a producción y que falle:

- Enterarse antes que el cliente
- Analizar la posibilidad de solucionarlo, sino
- Hacer Rollback (volver a los cambios anteriores)

## Práctico N° 11 - Casos de Prueba - Detección de Bugs

Ejecutar los Casos de pruebas y Reportar bugs de los Tp de los otros  
compañeros??