UNIVERSIDAD DE LIMA
SCIENTIA ET PRAXIS

Ingeniería de Software 2

Análisis

Hernan Quintana Cruz

hquintan@ulima.edu.pe

Requerimientos

Requerimientos funcionales

●Declaraciones de los servicios que el sistema proveerá a los usuarios.

●Nos indica **¿Qué?** Es lo que va a hacer el sistema.

Requerimientos No Funcionales

●No se refieren a las funcione específicas que entregará el sistema.

●Nos permiten definir propiedades y características que se deben cumplir para el correcto funcionamiento del sistema.

SCRUM/XP

Product Backlog

Listado de User Stories o Historias de Usuario.

Para estandarizar el llamado de las User Stories, utilizaremos el formato **Connextra**:

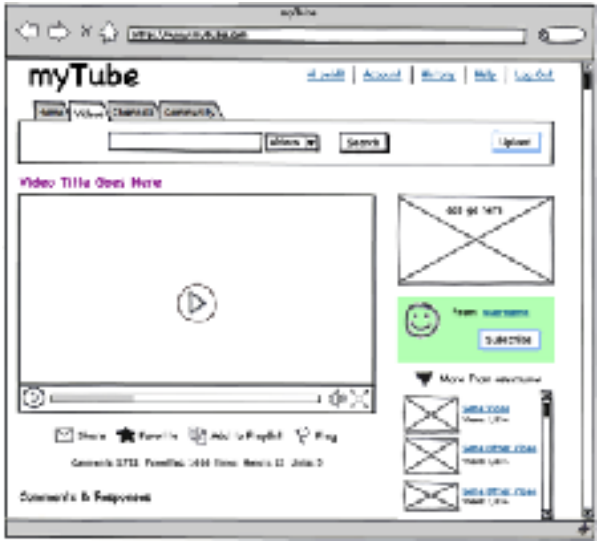
Como <rol>, deseo <objetivo/deseo> ya que <beneficio>.

- Como profesor, deseo poder visualizar las notas de mis evaluaciones, ya que de esa manera puedo hacer seguimiento al rendimiento de mis alumnos.
- Como alumno, deseo poder subir archivos de actividades en clase, ya que de esa manera el profesor pueda revisarlas y darnos feedback.
- Como profesor, deseo poder calificar y registrar feedback de alguna actividad de los alumnos, ya que de esa manera ellos pueden ver en que se equivocaron.

Especificación de User Stories

Debemos hacerlo de dos maneras:

Mockups o Wireframes



BDD

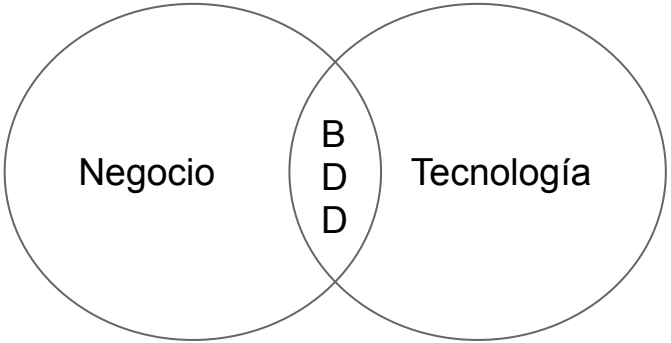
Fuente: https://www.vikingcodeschool.com/web-design-basics/what-are-mockups

BDD

Behavioural Driven Development (Desarrollo guiado por el comportamiento).

Forma estándar (pero no formal) de especificar (detallar) las historias de usuario.

Nos permitirá la generación de test automáticos de validación e inclusive de verificación.



Lenguaje Gherkin

Sintaxis

Feature	• Nombre del User Story (en formato Connextra)
Scenario	• Conjunto de steps que definen una interacción con el sistema. • Definen casos de prueba del Feature.

Steps

Given	• Pone el sistema en un estado conocido (pre). • No se debe de poner interacciones de usuario.
When	• Describe la acción o evento que el usuario realiza. • Es el trigger para la transición de estado del sistema.
Then	• Describe el resultado a llegar dado la transición (evento). • El resultado debe ser algún tipo de output: reporte, pantalla, mensaje.
And, But	• Sirven para concatenar múltiples Given-When-Then

Ejemplo Gherkin

Feature: Como profesor, deseo poder visualizar las notas de mis evaluaciones, ya que de esa manera puedo hacer seguimiento al rendimiento de mis alumnos.

Scenario: Visualizar mis evaluaciones realizadas
Given Haberse logueado en el sistema
And Haber seleccionado la opción de evaluaciones del menú
When el profesor selecciona un curso de los que tiene registrado
Then Se ve un listado de evaluaciones.
Scenario: No se tienen evaluaciones realizadas
Given Haberse logueado en el sistema
And Haber seleccionado la opción de evaluaciones del menú
And no se encuentran evaluaciones en la base de datos
When el profesor selecciona un curso de los que tiene registrado
Then Se ve un mensaje que dice que “No existen evaluaciones creadas”

Actividad 1: Especificación Gherkin

Realizar un listado de 3 user stories del sistema blackboard.

De estas 3 historias, especificar 1 con lenguaje Gherkin. Debe ser lo más específico posible.

Sprint Backlog

Durante las reuniones se realiza lo siguiente:

- Se categorizan las user stories: Feature, Bug, Trabajo.
- Se listan las tareas a realizar en cada user story
- Se estiman cada user story.
- Se planifica que user stories van a ir en los sprints a realizar (tomando en cuenta las estimaciones y la **velocidad)**.

Estimación

Plan & Document

Basada en modelos estadísticos (COCOMO II, Puntos de Función, Puntos de Aplicación).

Análisis de funcionalidades antes de la implementación. Esto podría tomar bastante tiempo y esfuerzo.

Ágil

Estimación en equipo. Compromiso de cada miembro del equipo.

Basada en puntos que miden la **complejidad** y no los tiempos (aunque haya una proporcionalidad directa).

Una herramienta a utilizar es el planning poker (https://www.mountaingoatsoftware.com/tools/planning-poker)

Velocidad

Puntos / semana que el **equipo** puede realizar.

Nos permite delimitar las capacidades del equipo para realizar una cantidad de user stories en un sprint.

Ejemplo:

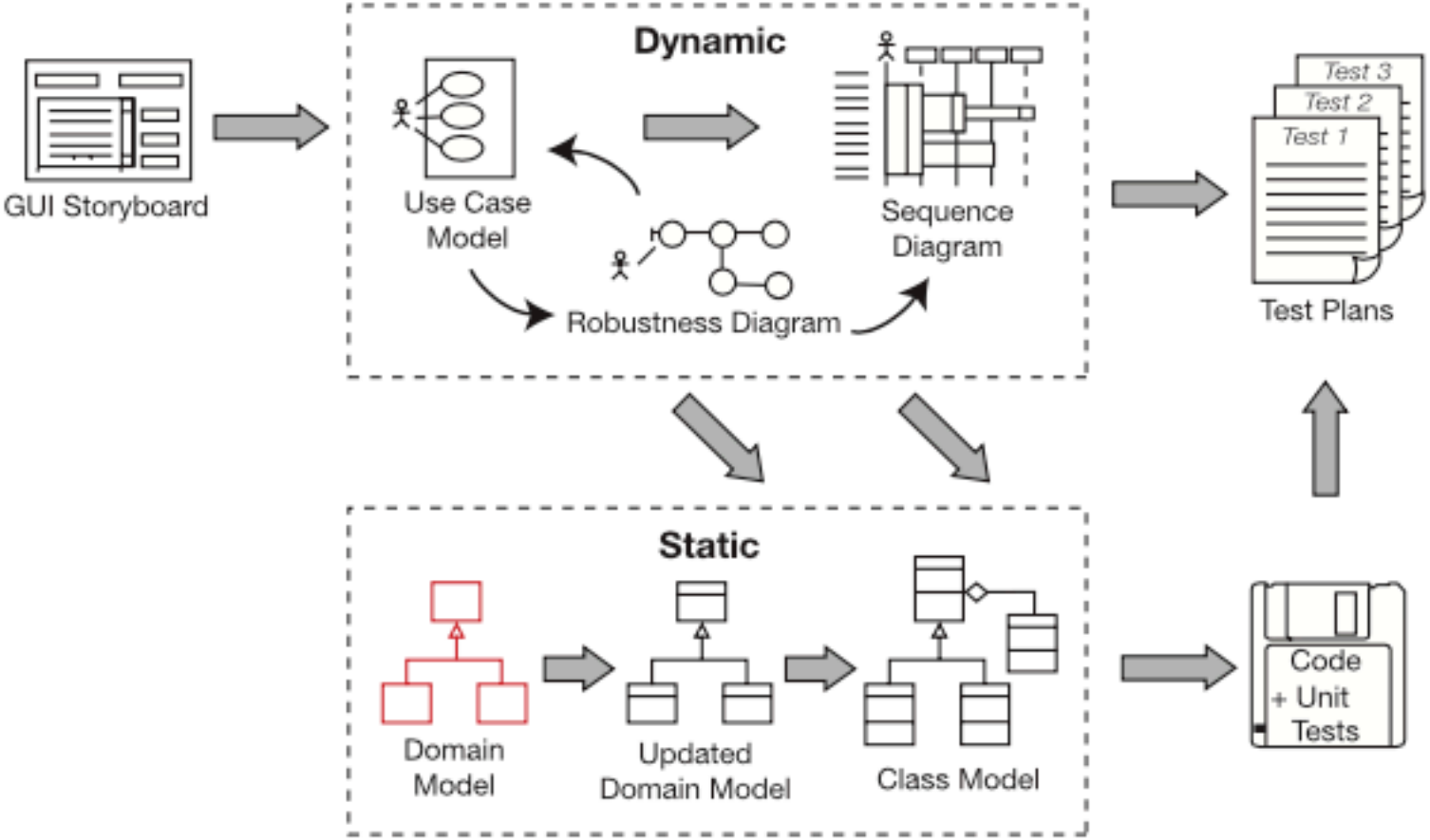
a) Siendo la velocidad del equipo de 10 y teniendo estas estimaciones, ¿cuánto tiempo me demoraría en implementar el proyecto?

US1	5 puntos
US2	8 puntos
US3	3 puntos
US4	5 puntos
US5	5 puntos
US6	3 puntos
US7	8 puntos
US8	30 puntos
US9	3 puntos

b) Priorizar qué user stories se realizarán en cada sprint (especificar la cantidad de sprints).

Burndown chart

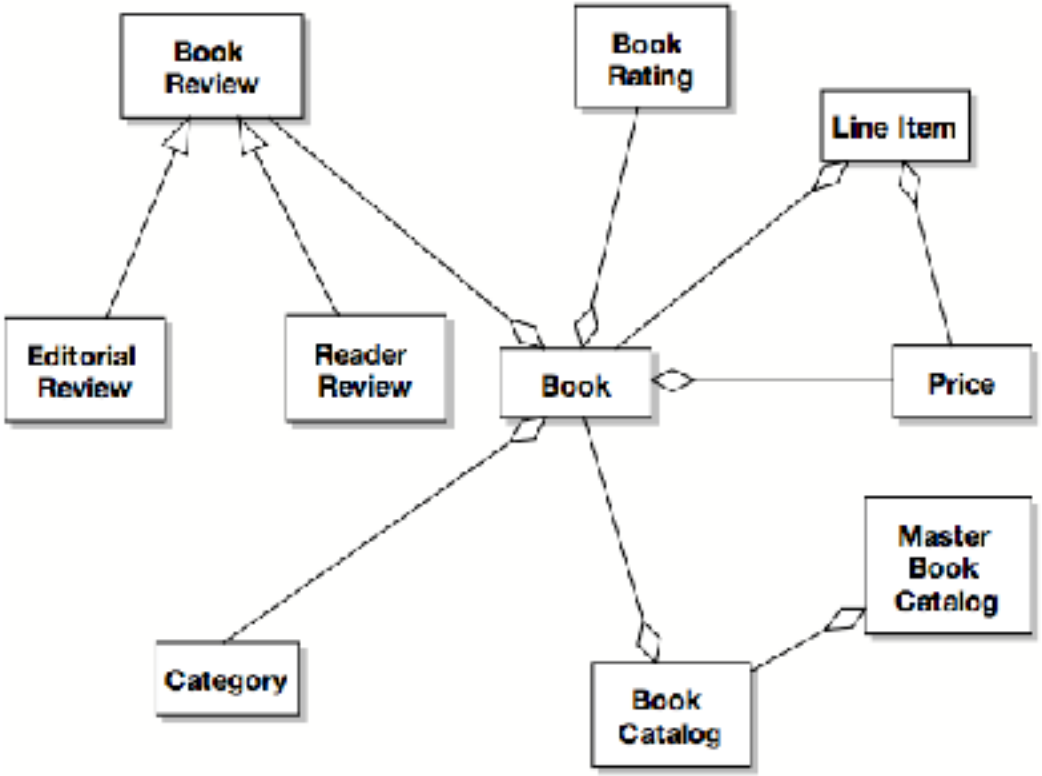
ICONIX


GUI Storyboard
Dynamic
Use Case Model
Sequence Diagram
Robustness Diagram
Test 3
Test 2
Test 1
Test Plans
Static
Domain Model
Updated Domain Model
Class Model
Code + Unit Tests

1. Domain Model (Modelo de dominio)

- Glosario del proyecto.
- Define el alcance y plantea una base sobre la cual se crearán los casos de uso.
- Delimita el “problem space”.
- Nos permite mostrar relaciones entre los objetos pertenecientes al dominio.
    - Agregaciones (has-a)
    - Generalizaciones (is-a)

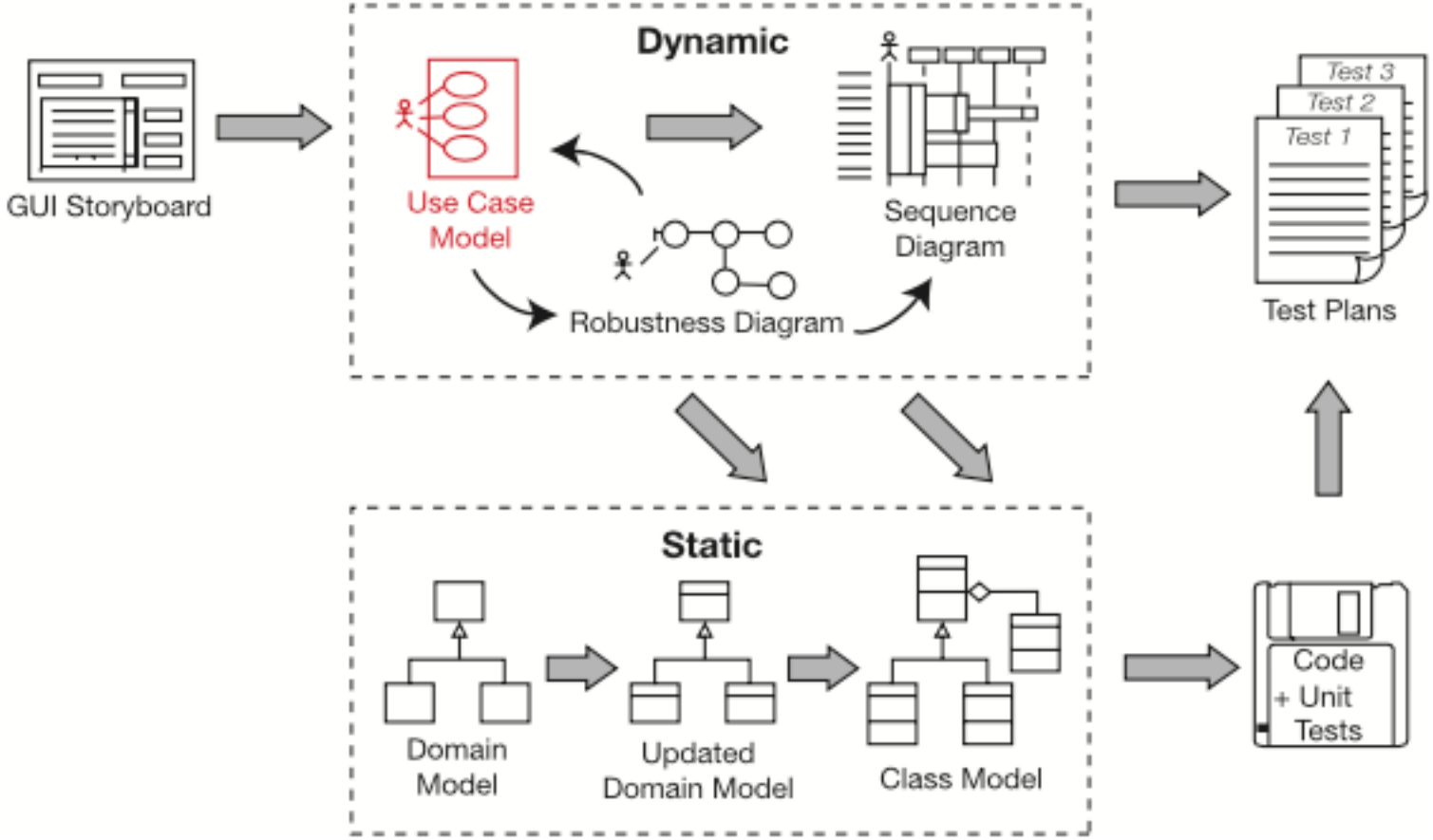
Ejemplo de Domain Model



Actividad 2. Domain Model del Blackboard

Tips

- No multiplicidad (1..* , * , etc).
- No relacionarlo aún con nada que tenga que ver con base de datos.
- Las clases/objetos no tienen que tener métodos.
- No listar clases/objetos que no sean entidades o que tengan otras relaciones distintas a has-a y is-a.


GUI Storyboard
Dynamic
Use Case Model
Sequence Diagram
Robustness Diagram
Test 3
Test 2
Test 1
Test Plans
Static
Domain Model
Updated Domain Model
Class Model
Code + Unit Tests

2. Diagrama de Casos de Uso

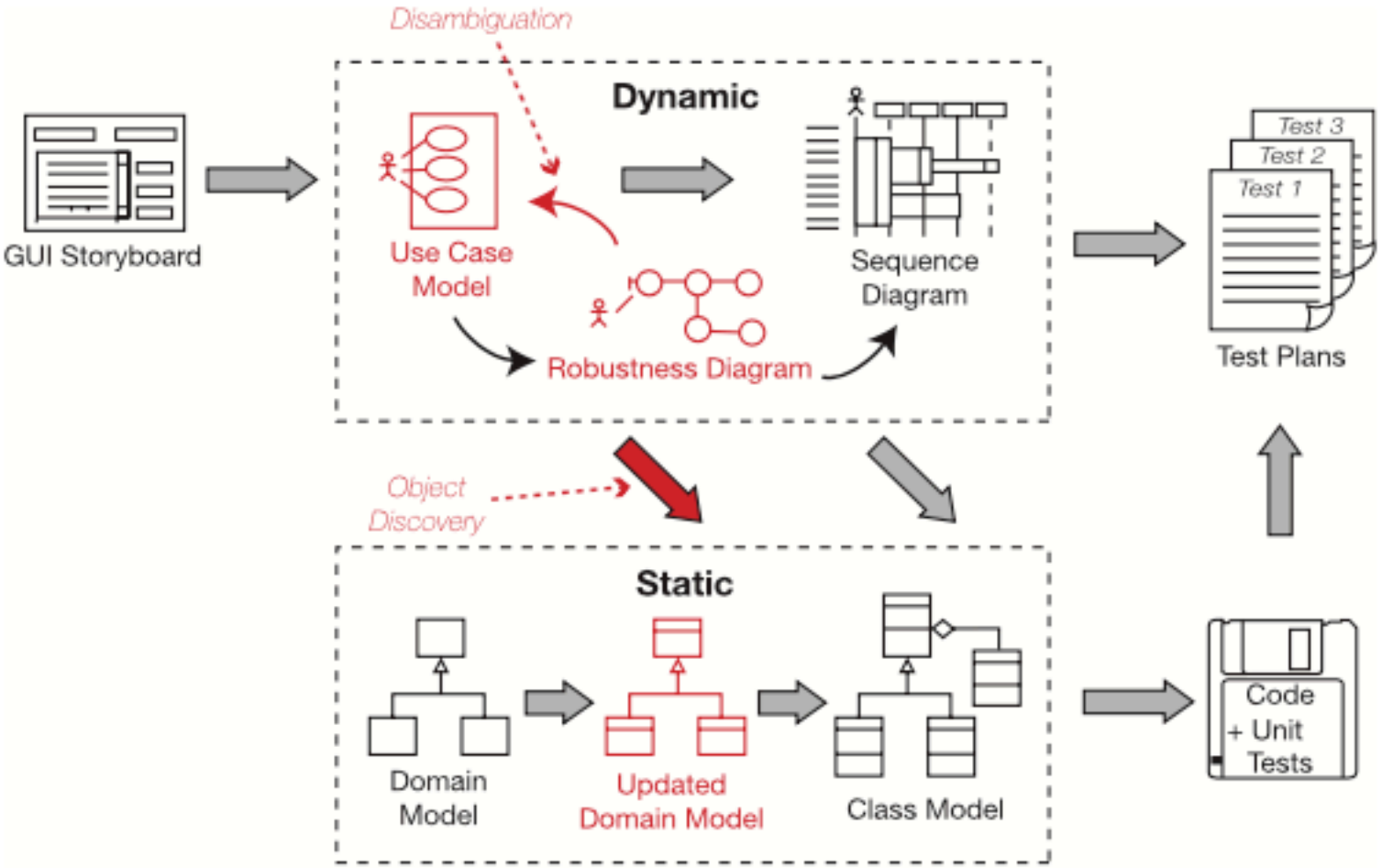
- Necesarios para clarificar y definir los requerimientos funcionales.
- Nos plantea el alcance real del software a implementar.
- Los reqs funcionales no son la única fuente de los casos de uso, también lo son las entrevistas con los usuarios, storyboards, prototipos, etc.

Tips

- Escribirlos en “activo”.
- Utilizar la forma evento -> respuesta.
- Usar storyboards (obligatorio).
- Relacionar un caso de uso con los objetos propuestos en el Domain Model.
- Dividir los casos de uso en paquetes.

Caso de Uso	Login de usuario
Actores	Alumno, Profesor
Precondición	Deben haber ingresado a la página de login de la aplicación.
Flujo básico	El comprador ingresa su nombre de usuario y su password y luego hace click en el botón Login. El sistema verifica si el usuario existe en el sistema y si coincide con su contraseña. El sistema permite ingresar al usuario.
Flujo alterno 1	**Login Incorrecto** El comprador ingresa su nombre de usuario y su password y luego hace click en el botón Login. El sistema verifica si el usuario existe en el sistema y si coincide con su contraseña. El sistema muestra un mensaje “Contraseña y/o usuario incorrecto”.
Poscondición	El usuario ingresa al sistema.

Actividad 3. Diagrama de casos de uso Blackboard.


Disambiguation
Dynamic
GUI Storyboard
Use Case Model
Sequence Diagram
Robustness Diagram
Test 3
Test 2
Test 1
Test Plans
Object Discovery
Static
Domain Model
Updated Domain Model
Class Model
Code + Unit Tests

3. Diagrama de Robustez

- Asegura que el caso de uso sea escrito en el contexto del modelo del dominio.
- “Concretiza” los casos de uso.
- Se encuentra en el límite de lo que se conoce como análisis (¿Qué?) con el diseño (¿Cómo?).

Objetos Interfaz

Objetos Entidad

Controladores

Tips

- Cada caso de uso debe tener su diagrama de robustez.
- Cada pantalla debe tener su objeto interfaz.
- Controladores son funciones lógicas de software.
- Actualizar constantemente los casos de uso y el modelo de dominio.

Actividad 4. Realizar Diagrama de Robustez a CU. Login

Referencias

- Rosenberg, D., & Stephens, M. (2013). Use case driven object modeling with UML: theory and practice. Berkeley, CA: Apress.
- Rubin, K. S. (2013). Essential Scrum: a practical guide to the most popular agile process. Upper Saddle River, NJ: Addison-Wesley.