



Buenas prácticas de software



Unidad 3. Mejores prácticas para la ingeniería de diseño.



Buenas prácticas de software



3.1 Mejores prácticas para el modelado de software.



Buenas prácticas de software



Consideraciones

Cuando se habla de la metodología para el desarrollo de software indiscutiblemente implica etapas diversas, las cuales se desarrollan secuencialmente o paralelamente siempre y cuando dependan de la estrategia con la que se defina.

Pero sin dudarlas dichas etapas deben llegar a la construcción del software.



Buenas prácticas de software



Análisis

Inicialmente se debe identificar lo que se quiere hacer, de acuerdo a como se solicite y la forma de como se va a desarrollar, de igual forma se asocia a la creación de un modelo de los requisitos levantados previamente.

Diseño – desarrollo

Una vez que se tiene el análisis podemos extraer las estructuras de datos, así como el diseño físico y el diseño lógico a lo que se le llama interfaz de usuario.



Buenas prácticas de software



3.1.1 Esquematización del modelo

El autor Sommerville en su libro conceptualiza el termino del modelo de un proceso de desarrollo de software como *“Una representación simplificada de un proceso de software, representada desde una perspectiva específica. Por su naturaleza los modelos son simplificados, por lo tanto, un modelo de procesos del software es una abstracción de un proceso real ([Sommerville, 2005](#)).*



Buenas prácticas de software



El modelado del software, es una técnica que permite identificar la estructura del software a desarrollar.

La elaboración del modelo ayuda a "identificar y visualizar" cada una de las partes y al sistema completo.

Se crean modelos de alto nivel, que son utilizados para la comunicación con el cliente y para concretar los requerimientos que serán desarrollados.



Buenas prácticas de software



El modelado es vital en todo tipo de proyectos,

El modelado permite identificar la arquitectura del software.

Para el modelado del software se utiliza el lenguaje UML.

UML se usa como lenguaje gráfico para modelar el software y permite: Definir, especificar, construir y documentar los procesos o funciones del software.

UML es visual, y modela varios elementos del software que ayudan a interpretar de mejor manera los requerimientos y funciones del software.

Cada diagrama sirve para modelar diferentes partes o puntos de vista de un sistema de software.



Buenas prácticas de software



3.1.2 Técnicas y lenguajes de modelado.

Sommerville define modelo de proceso de software como *“Una representación simplificada de un proceso de software, representada desde una perspectiva específica. Por su naturaleza los modelos son simplificados, por lo tanto, un modelo de procesos del software es una abstracción de un proceso real ([Sommerville, 2005](#)).”*



Buenas prácticas de software



3.1.2 Técnicas y lenguajes de modelado.

La herramienta Lucidchart permite de manera fácil y ágil realizar dibujos de los diagramas UML.

Se puede obtener una cuenta la cual es gratuita y se puede usar para realizar:

Diagrama de clases.

Diagrama de estados.

Diagrama de actividades.

Diagrama secuencial.

Diagrama de componentes.

Diagrama de casos de uso.

Diagrama de despliegue.



Buenas prácticas de software



Beneficios del modelado de software.

1. Eficiente la productividad del equipo.
2. Minimiza la cantidad de errores en el código.
3. Facilita la comprensión de las funciones.
4. Ayuda a la descomposición y modulado.
5. Ayuda al desarrollo y a su mantenimiento.
6. Ayuda a la reusabilidad.



Buenas prácticas de software



3.1.3 Nomenclatura para modelado diagramas UML.

UML forma parte del modelo orientado a objetos.

Ayuda a la visualización de la estructura del software o sistema.

Es una herramienta práctica que utilizan los desarrolladores para el modelado lógico de un sistema o software.



Buenas prácticas de software



Elementos de la metodología Orientada a Objetos:

Objeto

Son los atributos de las estructuras de datos.

Clase

Así se les conoce a los tipos de datos, así como al almacenamiento de los mismos.

Diagramas de casos de uso.

Permiten modelar cada una de las funciones del sistema (componentes del software).

El diagrama permite identificar cada una de las funciones así como la interacción de las mismas. Se ve con claridad que hace el sistema.



Buenas prácticas de software



Un caso de uso.

Identifica lo que hace un sistema.

Una elipse por su denotación gráfica se puede decir que es un conjunto de secuencias internas y externas la cual desarrolla el sistema.

Escenarios

Muestras la secuencia de cada una de las funciones que desarrolla el caso de uso.

El actor:

Es aquel que lleva a cabo la modelación de los requisitos funcionales

Representan entidades externas al sistema.

Se representa por medio de un muñequito.

Actor.

Interaccionan con el sistema mediante mensajes cortos que se colocan dentro de los casos de uso (elipse) e indican la acción que realiza el sistema o usuario.



Buenas prácticas de software



Relaciones: Se indican con una línea y denotan la relación de los actores con los casos de uso.

Tipos de relaciones:

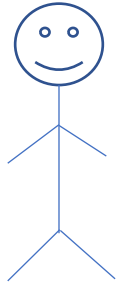
La Conexión, es una línea continua y es la relación sencilla del actor y un caso de uso.

La Inclusión, es una flecha punteada con la palabra incluye .

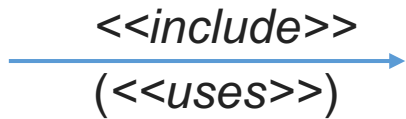
La Extensión, es un comportamiento adicional a un caso de uso y este se relación con el anterior se utiliza la palabra extend

La Generalización, Se utiliza la flecha punteada.

3.1.3 Nomenclatura para el modelado de diagramas UML.



Actor

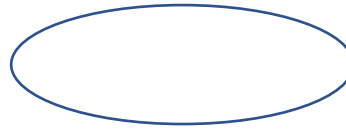


Include

o

Inclusión

Indica la relación entre dos casos de uso, en donde hay dependencia, actividades similares o iguales.



Caso de Uso



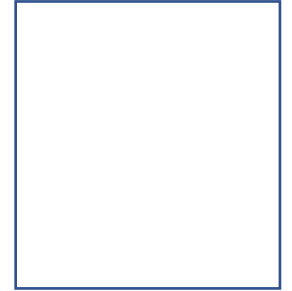
Comunica

Simplemente conecta un actor con un caso de uso, indica la participación del actor con el caso de uso




Extends o Extender

Cuando un caso de uso requiere de otra acción o función para finalizar. La dependencia de los casos de uso



Límite del sistema



Generaliza, la punta de la flecha indica hacia el caso de uso general

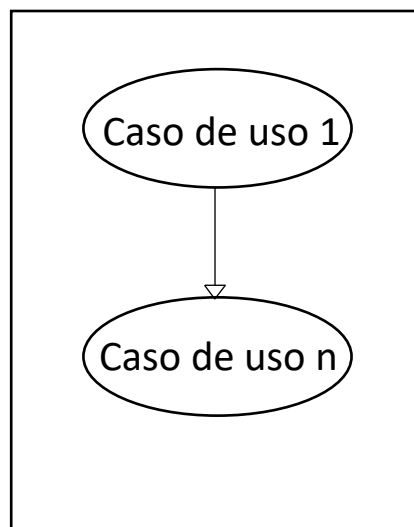


Buenas prácticas de software

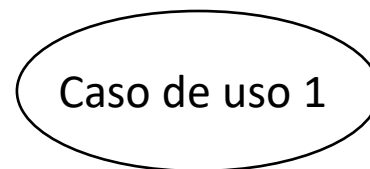


Diagrama de casos de uso

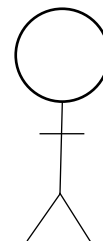
- Sistema
El rectángulo representa los límites del sistema que contiene los *casos de uso*. Los *actores* se ubican fuera de los límites del sistema.



- Casos de Uso
Se representan con óvalos. La etiqueta en el óvalo indica la función del sistema.



- Actores
Los *actores* son los usuarios de un sistema.





Buenas prácticas de software



Relaciones

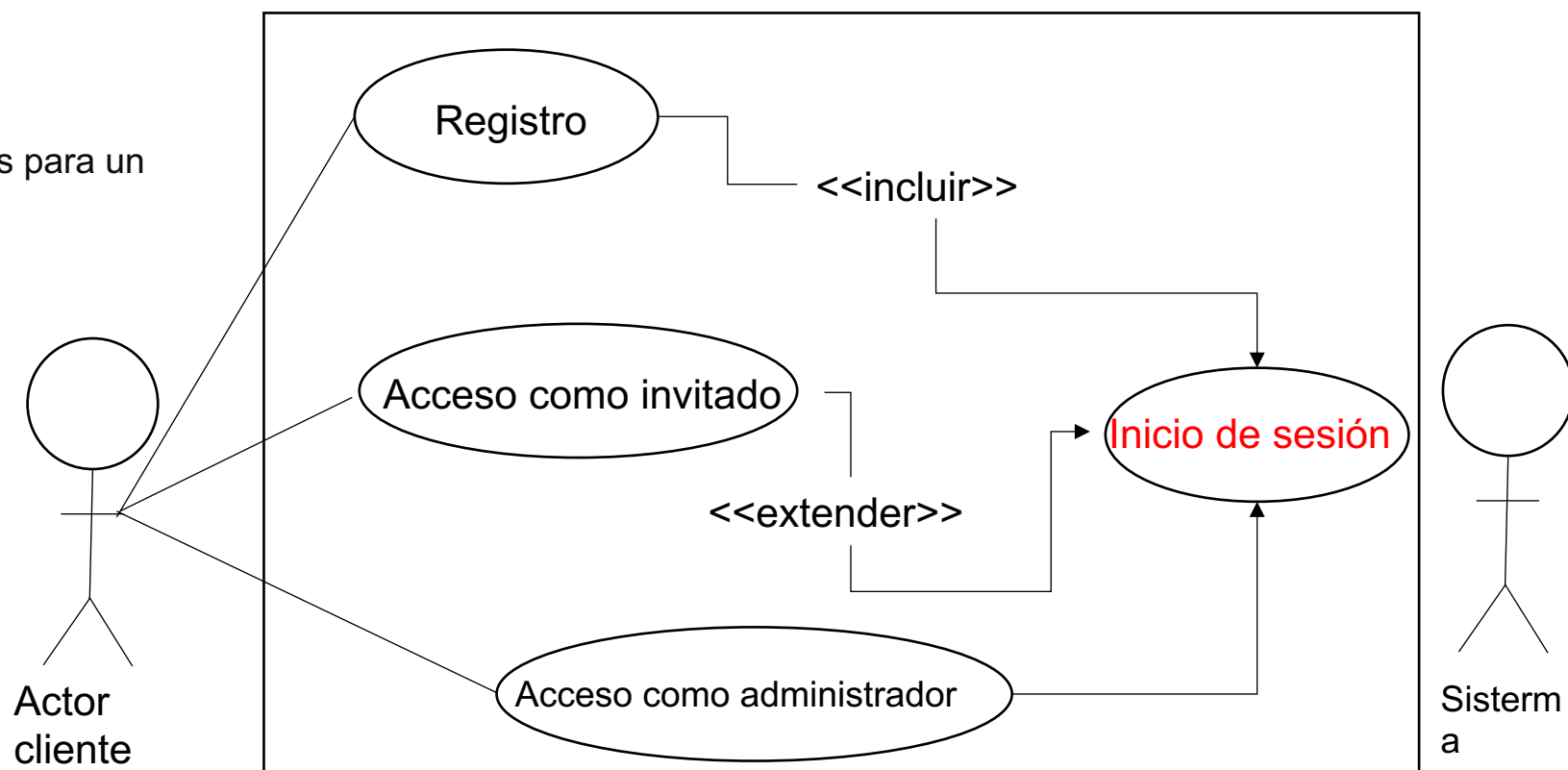
Las relaciones entre un *actor* y un *caso de uso*, se dibujan con una línea simple.

Para relaciones entre *casos de uso*, se utilizan flechas etiquetadas "incluir" o "extender."

Una relación "incluir" indica que un *caso de uso* es necesitado por otro para poder cumplir una tarea.

Una relación "extender" indica opciones alternativas para un cierto *caso de uso*.

Recuperar datos de registro





Buenas prácticas de software



Diagrama Clases

En estos diagramas se describe la estructura estática de un sistema.



Buenas prácticas de software



3.1.4 Herramientas y marcos de trabajo código libre para el modelado.

¿Qué son las herramientas para el modelado de código libre?

Cuando se habla de licencia para el uso de un software se le está otorgando al usuario el derecho legal al uso del software, hoy en día están disponibles dos tipos de licencias: libres y comerciales.

La Free Software Foundation (Fundación del Software Libre) «el software libre se refiere a la libertad de los usuarios para ejecutar, copiar, distribuir, estudiar, cambiar y mejorar el software»,



Buenas prácticas de software



3.1.5 Herramientas y marcos de trabajo para el modelado.

¿Qué son las herramientas para modelado comerciales?

Son aquellas que se desarrollan por una persona y para utilizarlas se requiere de adquirir (pagar) la licencia. Por lo que el autor tiene los derechos comerciales.



Buenas prácticas de software



Comparación de algunas herramientas para modelado UML.

- Compatibilidad con sistemas operativos.

Herramienta	Windows	Linux	MacOS
ArgoUML	x	x	
BOUML	x	x	x
Lucidchart	Web		
GModeler	x	x	
MagicDraw	x	x	x
MonoUML		x	
StarUML	x		
Umbrello	x	x	
Netbeans UML	x	x	x

- Libre o comercial.

Herramienta	Libre	Costo
ArgoUML	x	
BOUML	x	
Lucidchart		x
GModeler	x	
MagicDraw		x
MonoUML	x	
StarUML		x
Umbrello	x	
Netbeans UML	x	



Buenas prácticas de software



Comparación de herramientas para modelado UML.

Diagramas que realizan.

Herramienta	CU	CL	S	CO	A	O	P	E	ET
ArgoUML	x	x	X	X	x			x	
BOUML	x	X	X	x		x			
Lucidchart	x	x	x		x	x	x	x	x
GModeler		x					x		
MagicDraw	x	x	x	x	x			x	
MonoUML	x	x							
StarUML	x	x	x	x	x				x
Umbrello	x	x	x	x	x			e	
Netbeans UML	x	x	x	x	x	x	x	x	x

CU: casos de uso.
CL: clases.
S: secuencia.
CO: colaboración.
A: actividades.
O: objetos.
P: paquetes.
E: estados.
ET: estructura.



Buenas prácticas de software



Comparación de herramientas para modelado UML.

Generan código con base en el diagrama.

Herramienta	Java	C++	C#	PHP	Python	Ruby
ArgoUML	x	x	x	x	x	
BOUML	x	x		x	x	
MagicDraw	x	x	x			
MonoUML			x			
StarUML	x	x	x			
Umbrello	x	x	x			x
Netbeans UML	x	x	x			