Asignatura	Datos del alumno	Fecha
Ingeniería del Software	Apellidos: González Pradas	25/11/2020
	Nombre: Ernesto	

Índice

Actividad individual: Modelo de proceso y comprensión de requisitos

Descripción de la actividad y pautas de elaboración

Vamos a dividir en dos partes esta actividad. En primer lugar, lo relativo al **cambio de modelo de proceso,** y en segundo lugar, la **comprensión de los requisitos** del sistema a ser desarrollado.

- **Cambio de modelo de proceso**

  Supongamos una empresa de desarrollo de software de sistemas de gestión, en la que trabajas como ingeniero de software y en la que se aplica hasta el momento un modelo de proceso clásico en cascada, se decide a realizar un plan de cambio para trabajar con un modelo de proceso ágil.

  Modelo en cascada: hasta que no termina una actividad no comienza la siguiente.
  Modelo de proceso ágil: tiene como objetivo producir productos funcionales con rapidez que se irán completando, desarrollando y mejorando de forma progresiva en unidades siempre funcionales.

  Ejercicio 1: Elabora el plan de cambio en el que describas las tareas a realizar. Si obtienes información de Internet aporta referencias (no copies de Internet, en su lugar explícalo con tus propias palabras).

  Recomendación: ten en cuenta los principios del manifiesto por el desarrollo ágil del software y también el proceso de desarrollo de software Scrum.

Respuesta Ejercicio 1:

Para elaborar el plan de cambio, del modelo en cascada al modelo ágil, nos vamos a basar en el marco concreto de SCRUM, ya que dentro del modelo de proceso ágil encontramos una gran variedad de marcos, como pueden ser: SCRUM, Programación Extrema XP, Programación Extrema XP Industrial, Kanban y un largo etc.

Primero vamos a analizar, cuáles son las partes fuertes y las partes más problemáticas del modelo en cascada. El modelo en cascada está pensado para ser utilizado cuando los requisitos de un desarrollo son muy claros y no hay a penas cambios previsibles en el propio desarrollo. Esto de por sí en la realidad, es un problema, ya que es realmente complicado que un cliente nos pueda especificar a la perfección desde el inicio lo que quiere del producto y, por otro lado, esto tiene una desventaja frente a la detención de erros, ya que hasta que no entreguemos el desarrollo funcional no podremos ver muchos de los errores que se enmascaran cuando estamos desarrollando el producto.
Por el contrario, el modelo ágil, plantea una adaptación en el proceso del desarrollo, que nos permite ser más dinámicos en cuanto a los errores que puedan surgir, ya que, este modelo se basa en ir mostrando pequeños desarrollos funcionales al cliente y, por lo tanto, permite una mejor adaptación tanto a las necesidades del propio cliente como a las del equipo.

Para mostrar esto último de una forma más clara, vamos a realizar un ejemplo de cambio de modelo de proceso ágil.
Para ello, vamos a detallar las tareas a realizar en cada momento del proceso:

- Primero, los analistas del negocio y los gerentes se reúnen con el cliente para que este les cuente que producto es el que quiere o identificar cual es la necesidad del negocio. Para ello tienen que entender cuál es la problemática, hacia quién va dirigida la solución y sobre todo una comunicación fluida y constante con dicho cliente.

- Segundo, una vez entendidas las necesidades del cliente, se realiza una reunión de estado en la que se implica a todo el equipo técnico, para que todos tengan un horizonte y un fin común. Que todo el equipo sepa hacia donde tienen que ir y sugerir tantas recomendaciones como crean oportuno. Al finalizar esta parte, tendremos un listado de requerimientos.
- Tercero, realizado el paso anterior, podremos elaborar las distintas tareas a raíz del listado de requerimientos surgido. Dichas tareas, el equipo, las puntuará en función de la complejidad, tiempo de resolución o prioridad para mostrar al cliente determinadas funcionalidades en los pequeños entregables. Se darán de alta en una herramienta de gestión de tareas como puede ser Jira, y se reparten en función de lo que se quiera entregar en el primer Sprint. Los Sprint serán de entre dos semanas y un mes dependiendo de la complejidad de las tareas metidas en cada uno ellos.
- Cuarto, una vez que tenemos todas las tareas del Sprint bien definidas y repartidas entre el equipo, se realizará diariamente una reunión con el equipo (también llamadas daily), de no más de 15 minutos donde cada integrante dirá que ha realizado desde la última daily, si ha tenido o no algún obstáculo y que hará para la próxima reunión. Es muy importante que, una vez empezado el sprint, no se introduzcan cambios que puedan afectar al objetivo de dicho sprint.
- Quinto, al finalizar el sprint se reúne todo el equipo y parte del cliente para comprobar los desarrollos realizados y ver si se incorporan al producto. Para esto, los propios desarrolladores exponen los cambios o desarrollos realizados y el cliente, si lo considera necesario, realizará distintos tipos de preguntas. Por otro lado, el equipo habla de si hay algún punto en concreto que mejorar para ser más ágiles y se vuelve a planificar un nuevo sprint de la misma forma que se ha explicado más arriba en el texto.

Las referencias que he usado, para realizar este cambio al modelo de proceso ágil, las he sacado por un lado de las diapositivas vistas en clase junto con algunas partes del libro “Ingeniería del Software – Un enfoque práctico. *Roger S.Pressman*” (por ejemplo: distintos tipos de marcos ágiles como programación extrema XP, etc) y mi

propia experiencia laboral donde realizo reuniones diarias (dailys), entregamos Sprints y la manera de gestionar todas las tareas (en mi caso con herramientas como Jira) y el equipo.

- **Comprensión de los requisitos**

  Tras el cambio de modelo de proceso de la empresa de desarrollo de software de gestión, uno de los primeros proyectos software que se requieren en dicha empresa es el de construir un sistema para la gestión de una biblioteca universitaria.

  En dicha biblioteca universitaria se dispone de puestos de lectura y salas de trabajo en grupo. Los usuarios (profesores y alumnos) podrán reservar puestos o salas. Los profesores no pueden reservar más de una sala al día y los alumnos no más de una sala a la semana.

  Por otro lado, el sistema también permitirá a los usuarios la reserva de los libros de la biblioteca, algunos de los cuales son solo para lectura en sala, por lo que no pueden sacarse de la biblioteca.

  Para el desarrollo de las bases fundamentales del proyecto sobre la comprensión de los requisitos, se deben llevar a cabo los siguientes ejercicios.

  Ejercicio 2: Elabora el listado de clases y el listado de relaciones (generalización, asociación, dependencia, agregación o composición) justificando el tipo de relación. Puede no haber relaciones de alguno de los tipos.

Respuesta Ejercicio 2

Listado de Clases

Biblioteca

Puestos de Lectura

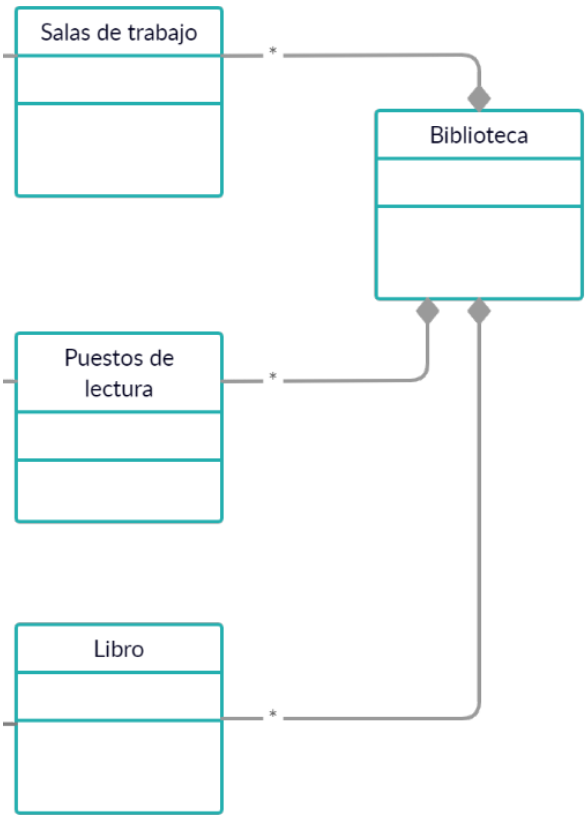
Salas de Trabajo

Usuarios (Profesores y alumnos)

Libros

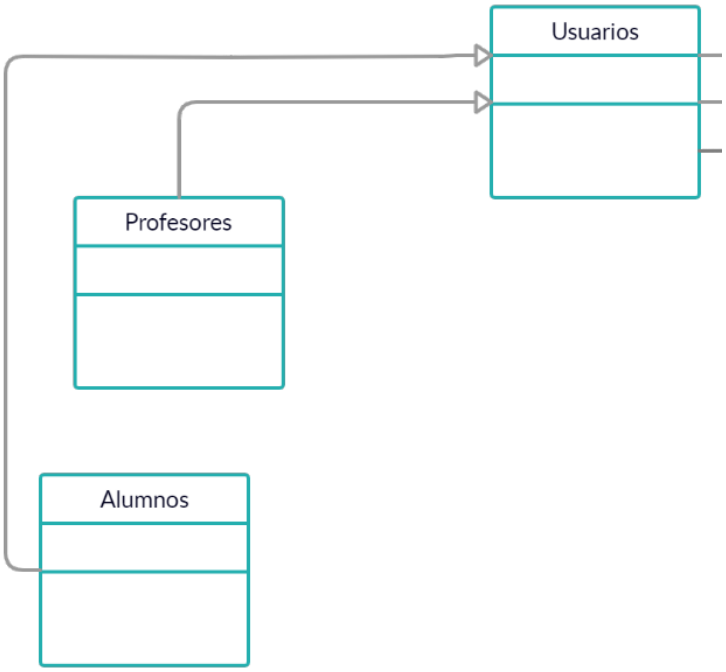
Listado de Relaciones

La **biblioteca** tiene **Puestos de lectura**, **Salas de trabajo** y **Libros**. Esto sería una relación de **Composición**, ya que, si por un casual se demoliera la biblioteca, las clases "Puestos de lectura", "Salas de trabajo" y "Libros" no tendrían razón de existir **para este enunciado**:

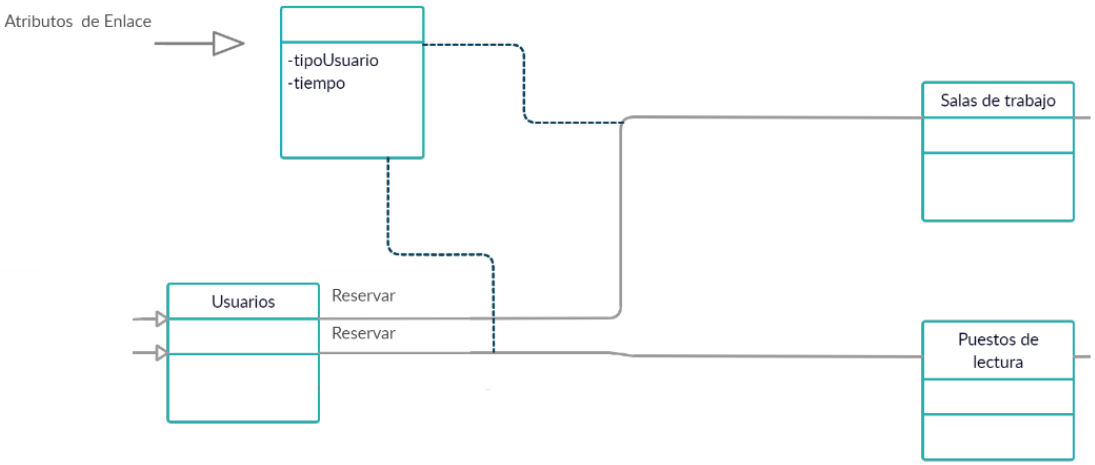


Además, tiene una multiplicidad de 1-*, ya que en una biblioteca puede haber una o varias salas de trabajo, uno o varios puestos de lectura y uno o varios libros.

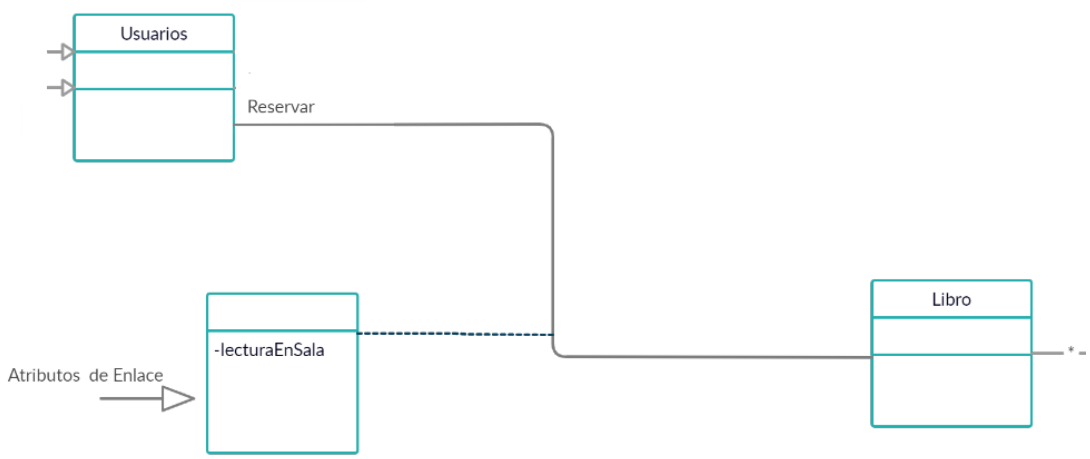
Otra relación sería la que hay entre las **clases Profesores y Alumnos** con la clase **Usuarios**. Esta relación sería de **Generalización** ya que heredarían aspectos o atributos de la clase padre que sería usuarios:



La siguiente relación sería la que tienen los **Usuarios** con las **Salas de Trabajo** y los **Puestos de Lectura**. Esta sería una relación de **Asociación**, ya que, los objetos de una clase están conectados con los de la otra y es bidireccional, "los usuarios pueden reservar salas" y "las salas son reservadas por usuarios". Esta relación de asociación también tiene unos atributos de enlace que nos especifican que tipo de usuario (Profesor o Alumno) puede reservar por cuanto tiempo las salas de trabajo y los puestos de lectura:

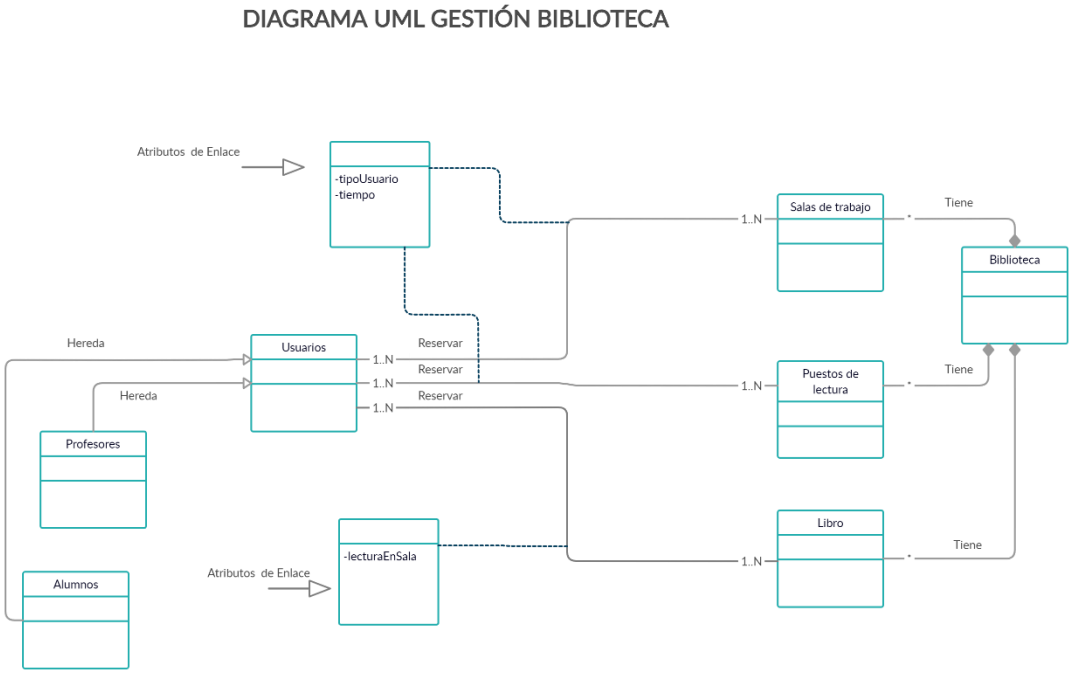


Por último, tenemos la relación de los **Usuarios** que pueden reservar **libros**, aunque algunos de ellos solo se pueden leer dentro de las salas. Es una relación de Asociación por el mismo motivo que la anterior entre los usuarios y las salas de trabajo y los puestos de lectura. Los objetos son accesibles desde cada una de las clases contrarias y son bidireccionales, “los usuarios reservan libros” y “los libros son reservados por los usuarios”. Esta relación también tiene un atributo de enlace que marca si los libros reservados se pueden o no sacar de la biblioteca:



Ejercicio 3: Representa el diagrama UML de clases que contenga tanto estas como las relaciones.

Respuesta Ejercicio 3



Objetivos

- Evaluar los conocimientos adquiridos respecto a la introducción a la ingeniería del software.
- Preparar los fundamentos para el modelado, el proceso del software y la comprensión de los requisitos.
- Comprender cómo aplicar la ingeniería del software en un proyecto software.

Entrega y extensión máxima

Se deberá entregar un **documento de diez páginas en fuente Calibri 12 e interlineado 1,5,** conteniendo los diferentes ejercicios bien explicados y detallados. Además, se requieren **capturas o imágenes** que muestren los **resultados de los diagramas.** Se valorará la capacidad de análisis, un adecuado plan de cambio de proceso, más las clases y relaciones representadas en el diagrama de clases correspondiente, así como la presentación del documento.

También se considerará la necesidad justificada de una mayor extensión (siempre que el contenido sea adecuado y necesario). El documento deberá contener un **índice,** así como una **estructura adecuada, conclusiones** finales, y **referencias bibliográficas.**

Rúbrica

Modelo de proceso y comprensión de requisitos	Descripción	Puntuación máxima (valor real: 5 puntos)	Peso %
Criterio 1	Plan de cambio de modelo de proceso	3	30 %
Criterio 2	Listado de clases y relaciones	3	30 %
Criterio 3	Diagrama UML de clases	4	40 %
		10	**100 %**

Para alcanzar la máxima puntuación, habrá que superar correctamente los diferentes criterios, que incluyen no solo el diseño, sino también el explicarlo adecuadamente en el documento a ser entregado.