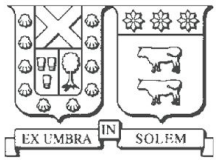
Departamento de Informática
Universidad Técnica Federico Santa María



Entregable IV

Proyecto: “Machine for learning”

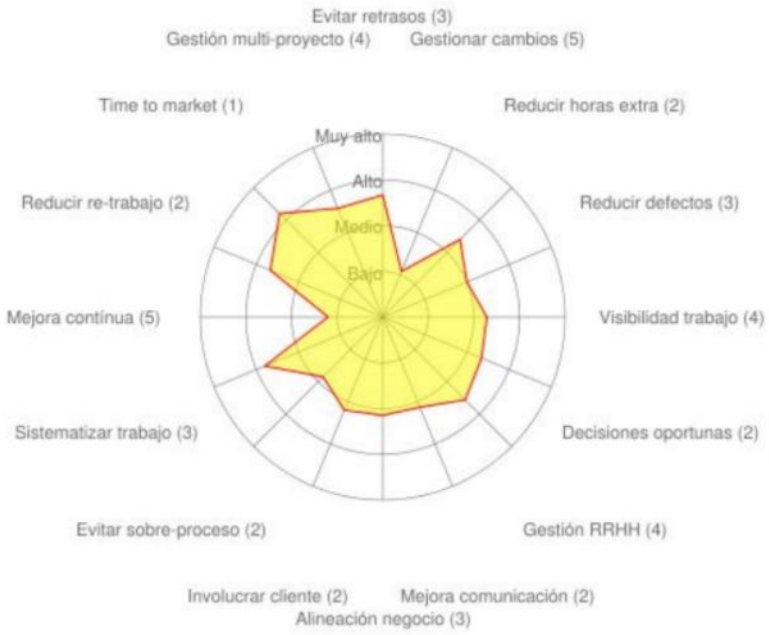
Integrantes:

Nombres y Apellidos	Email	ROL USM
Iván Caro León	ivan.caro.12@sansano.usm.cl	201273545-9
Felipe Flores Valdivia	felipe.floresv@alumnos.usm.cl	201123518-5

Post-Mortem Metodológico

- Gráfico de evaluacion por areas:

- Gráfico de evaluación por objetivos:



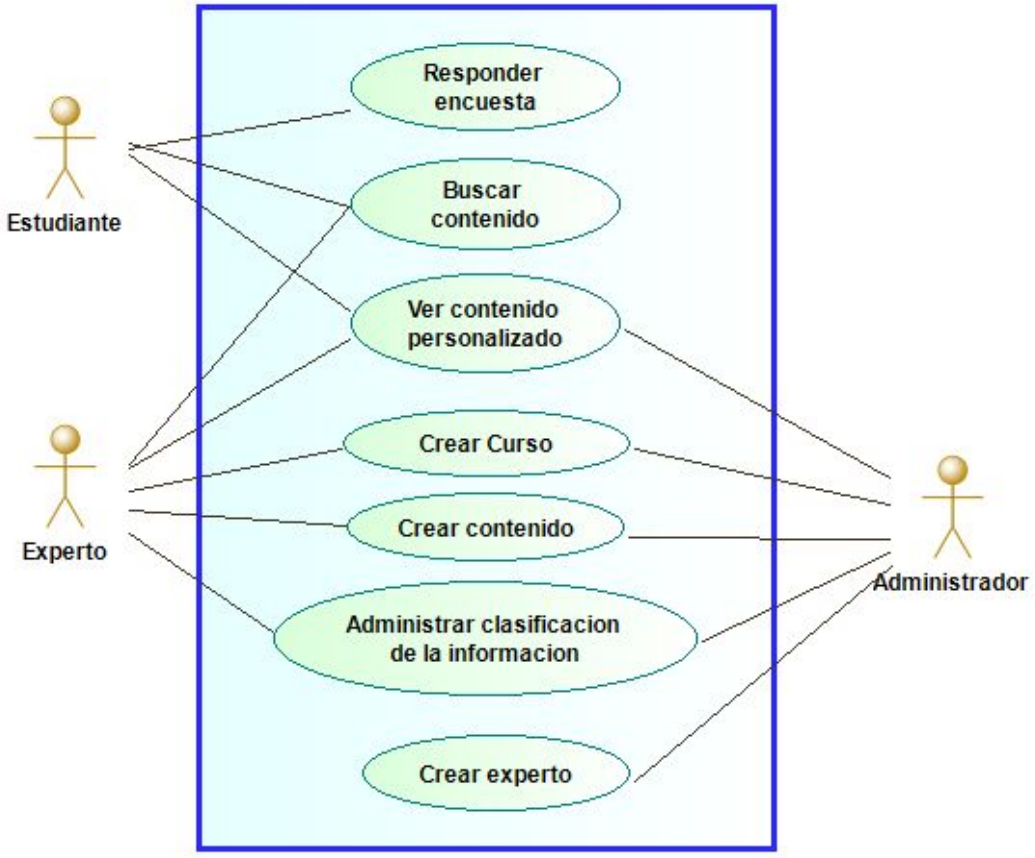
- Análisis de resultados test de agilismo:

Como se puede ver en el primer gráfico, el equipo posee un bajo nivel de agilismo en las áreas de reuniones, espacio de trabajo y liderazgo, por lo que las debilidades del equipo se encuentran en estas áreas. La mayoría de las demás áreas se encuentran en un nivel medio de agilismo por lo que no se consideran ni como debilidades ni como fortalezas. Por último se puede ver que en el área de fundamento ágil se tiene un alto nivel de agilismo lo que nos dice que el equipo privilegia la sencillez a la hora de tomar decisiones y evitar complicaciones que no son necesarias.

En el segundo gráfico se puede ver que los objetivos en los que se tiene un nivel bajo de agilismo son gestión de cambios y mejora continua lo cual consideramos como una de las mayores debilidades del equipo, esto puede mejorarse estableciendo una buena planificación del proyecto y se constante en el cumplimiento de esta, además de realizar revisiones periódicas del trabajo realizado para confirmar que el trabajo se está realizando como se planificó, y si esto no está ocurriendo tener un plan de acción para resolver el problema.

Pensando en el futuro consideramos que las mejoras más importantes y necesarias que se debe aplicar es potenciar el trabajo en equipo y la realizar una revisión periódica del trabajo que ha hecho, ya que durante el desarrollo de este proyecto estos fueron los mayores problema que detectamos, esto se puede mejorar realizando reuniones periódicas con el equipo, trabajando en conjunto en el mismo espacio físico y definiendo un líder dentro del equipo para así tener un objetivo fijo y no se pierda el foco de lo que se desea hacer y cumplir a cabalidad con la planificación que se estableció en un principio.

Diagrama de Casos de Uso (final)



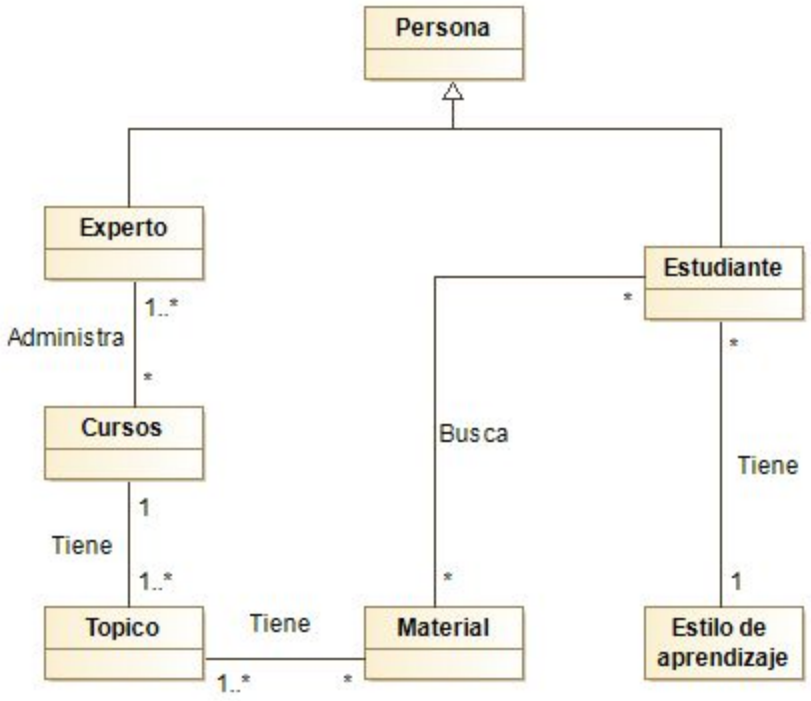
Como se puede ver en el diagrama de casos de uso final, se cumplieron los principales requerimientos que solicitó el cliente para el software.

Patrones de diseño y Frameworks (final)

Intención	Patrón de Diseño	Razonamiento
Se desea separar los componentes: modelo, vista y controlador para el desarrollo de la aplicación. Para así tener un código más legible, poder reutilizar código	MVC	Este patrón se utiliza ya que debido al tamaño del sistema que se está desarrollando, este puede llegar a ser muy complejo, por lo que dividirlo en partes hace que el desarrollo sea más simple y rápido
Se desea que el software no tenga problemas al recibir distintos formatos de material de estudio proveniente de la web y pueda clasificarlo de manera correcta independientemente del formato que este tenga	Adapter	Debido a los distintos formatos en los cuales se encuentra el material de estudio, el software debe realizar un trabajo extra al clasificar todos los distintos tipos de material, por lo que al agregar una clase que adapte este material a un formato más fácil de entender para el software, este tendrá menos trabajo al clasificar el material.
Se desea poder manejar de forma fácil y eficiente las instancias de cada clase, por lo que al utilizar este patrón facilita el manejo de los datos que usará el sistema	Active Record	Este patrón de diseño hace que el desarrollo sea mucho más simple en el sentido del manejo de los datos provenientes de la base de datos, ya que al ser tratados como objetos se hace mucho más fácil el trabajar con estos.

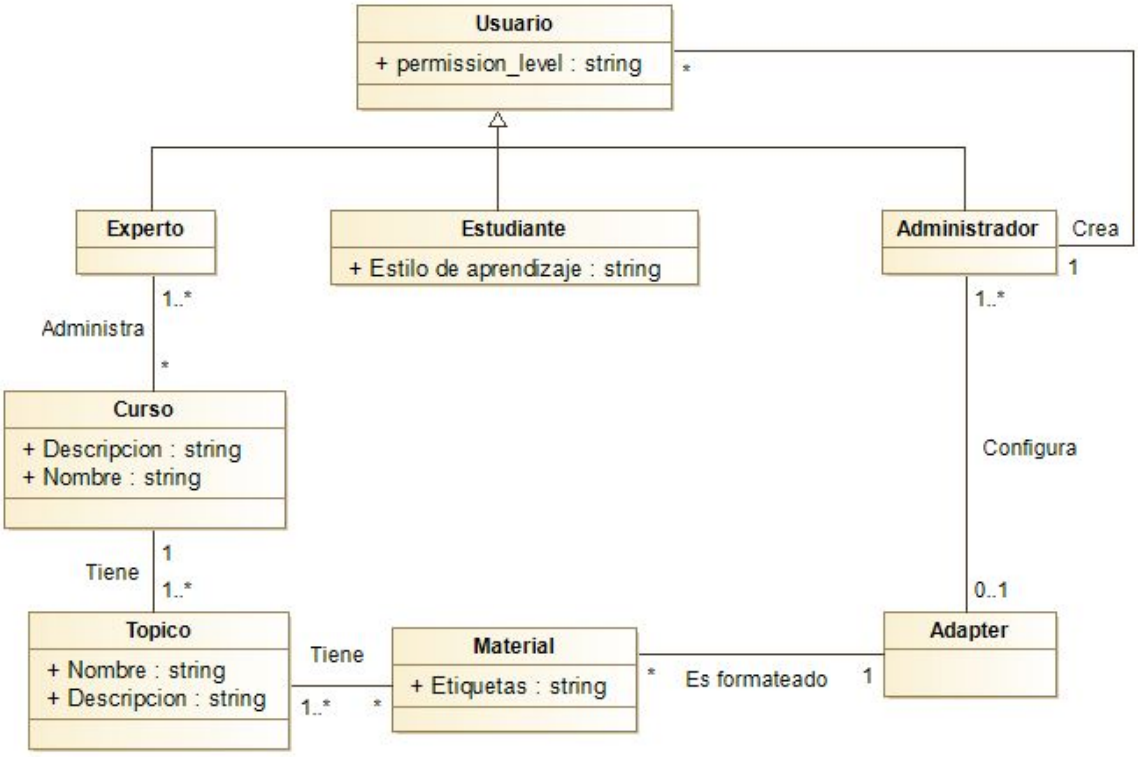
Modelo de Dominio y Diagrama de Clases (final)

- Modelo de dominio:



Se puede ver que el diagrama de dominio no cambió con respecto al presentado en los entregables anteriores, esto porque consideramos que se logró representar correctamente desde un principio el dominio del problema y no es necesario hacer cambios en el diagrama.

- Diagrama de clases:



En el diagrama de clases se muestran las clases que posee el software, como se puede ver está la clase Adapter la cual toma información de la web y la transforma en información que pueda ser mostrada por el software. El patrón de diseño MVC está utilizado en la separación de las partes del software en modelo, vista y controlador, y el patrón Active record se utiliza en la lectura y fácil manejo de datos provenientes de la base de datos, estos dos últimos patrones son provistos por el Framework que utilizó para desarrollar el software.

Pruebas de Software (actualización)

Defecto encontrado	Mitigación	Resultado obtenido	Observaciones
Registro de profesor con datos incorrectos	Se aplicó validación en frontend y en backend de los datos al realizar el registro impidiendo que se ingresen datos en un formato incorrecto	Se corrigió el error ahora solo se permite el ingreso de datos válidos en el formulario de registro	Este fue el único defecto encontrado en el entregable 3
Búsqueda distingue entre mayuscula y minuscula	Se cambió el formato en que se almacenan los datos en la base de datos haciendo que estos se guarden en minúscula y el texto a buscar tambien se pasa a minúscula al realizar la consulta en la BD	Ahora las búsquedas entregan los resultados correctos independiente de si se ingresan mayusculas o minusculas	Defecto encontrado durante el desarrollo