Universidad Técnica Federico Santa María

Ingeniería de Software

Entregable 3

Buscador de Herramientas de Aprendizaje

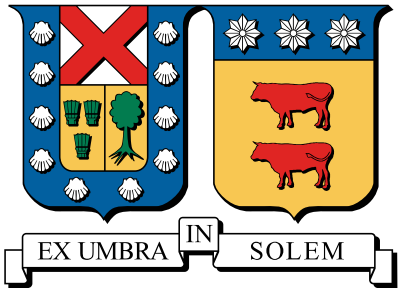
Autores:
Quantum Taco

Francisco Olivares
francisco.olivars.14@sansano.usm.cl
201473575–8

Felipe Vega
felipe.vega.14@sansano.usm.cl
201473511-1

Profesor:
Hernán Astudillo

Primer Semestre 2017



Índice

1. Post-Mortem Metodológico

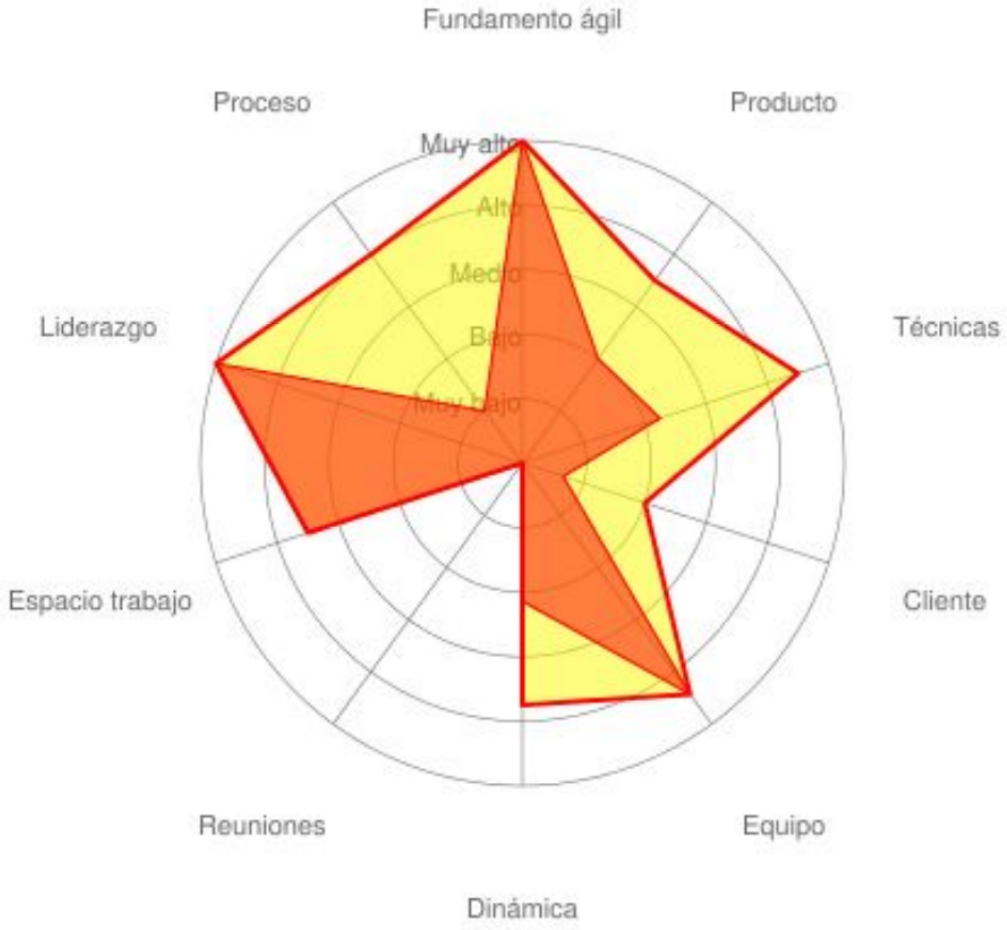


Figura 1: Resultado de la encuesta de agilismo.

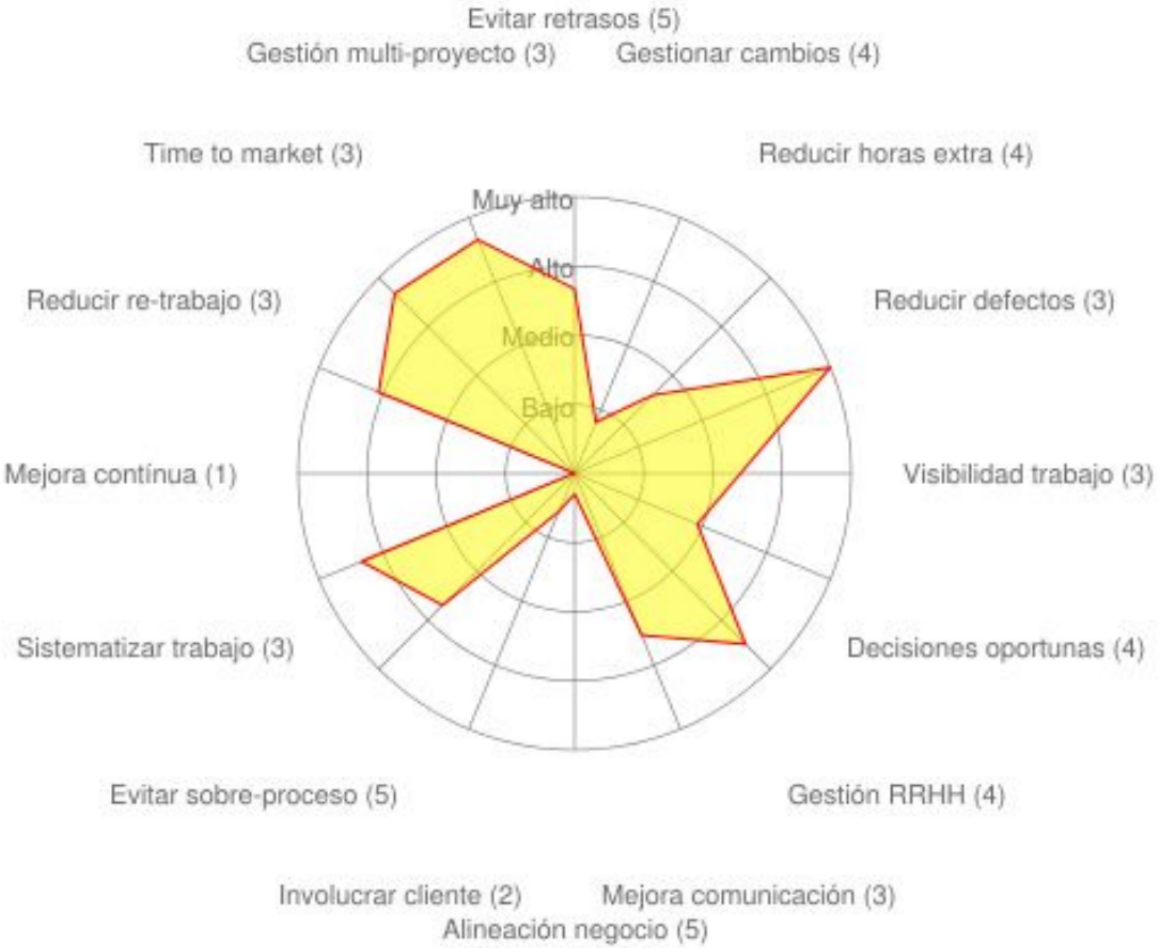


Figura 2: Resultado de la encuesta de agilismo.

Después de realizar el test se aprecia que el equipo tiene fortaleza en Fundamento ágil, Liderazgo, Equipo, Espacio de trabajo, Time to market, Reducción de defectos, Gestión de RRHH. Estas fortalezas eran de esperarse ya que al trabajar siempre se intentó hacerlo como equipo, distribuyendo la carga equitativamente y de manera de realizar las cosas lento pero seguro. Siempre se respetaron los horarios de trabajo, para que así lo que se pedía estuviera a la fecha que se necesitaba.

En las debilidades se tienen Proceso, Producto, Técnicas, Reuniones, Involucrar al cliente, Mejora continua, Gestionar cambios, Reducir horas extra, Alineación negocio. Estas debilidades se deben a que en la medida que se desarrolló el proyecto no se puso importancia a mejorar el proceso interno de desarrollo de software, también al ser un proyecto pequeño desarrollado por 2 personas no se gestionaron reuniones debido a que no se estimaron necesarias. El contacto con el cliente se le atribuye principalmente a que como equipo sentimos que el cliente no tenía el tiempo o no estaba lo suficientemente interesado en el proyecto, entonces por esta razón no se pudo establecer una buena comunicación.

Para mejorar estas debilidades se estima que en proyectos más grande y más complejos se realicen reuniones de equipo y con el cliente (si este tiene el tiempo y motivación) para poder hablar sobre el camino a seguir en el desarrollo. La mejora de procesos se puede dar al aplicar metodologías si es que se estima conveniente o sino seguir con lo más clásico (aplicar una metodología en pleno semestre de clase puede ser complicado).

2. Diagrama de casos de uso (final)

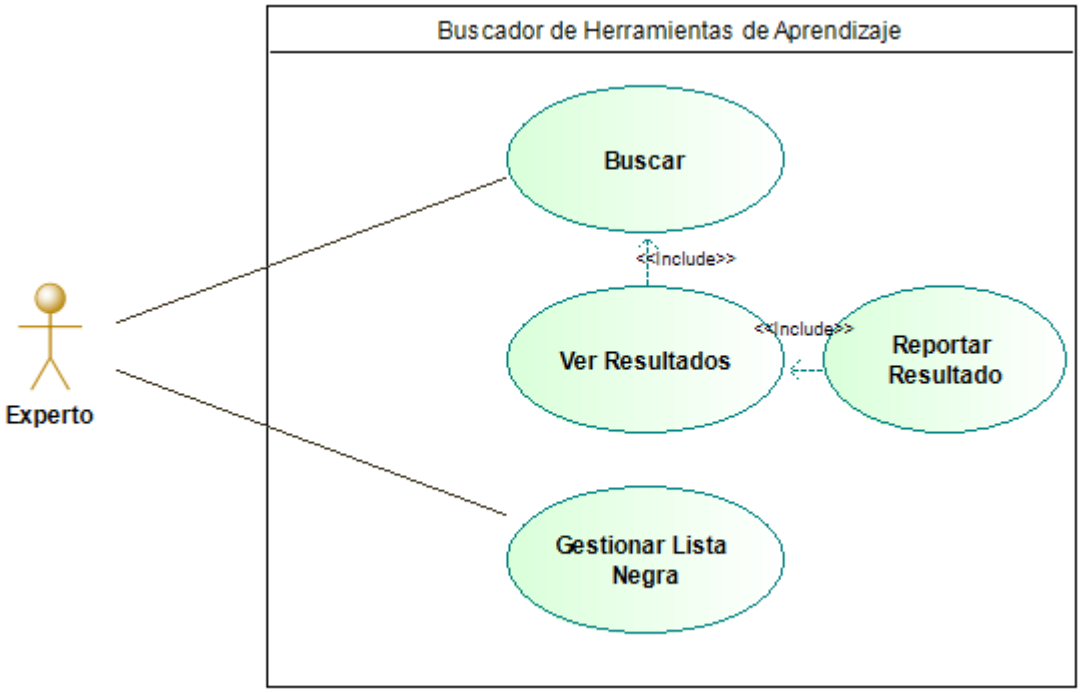


Figura 3: Modelo de casos de uso.

Caso de uso: Reportar resultado.	
Descripción: Una vez entregados los resultados al usuario, este tendrá la capacidad de reportar si parte del contenido retornado no es lo que él buscaba, agregando la página desde donde se obtuvo dicho contenido a una lista negra y así no ser considerada en futuras búsquedas.	
Precondiciones:	Haber realizado una búsqueda anteriormente.
Postcondiciones:	Agregar los falsos positivos a una "lista negra".
Secuencia exitosa.	
Experto	**Sistema**
1. Este caso de uso comienza cuando el experto encuentra un falso positivo dentro de la lista de retorno y selecciona el número de la herramienta	
2. El usuario hace click al botón "**Reportar**".	
	3. El sistema agrega la página a la "blacklist". Termina el proceso.
Secuencias alternativas.	
2.1 El usuario se arrepiente de reportar dicha herramienta y presiona "**Volver al inicio**"	

Caso de uso: Buscar.	
Descripción: El usuario busca herramientas de aprendizaje con respecto a un tópico, este debe elegir el tipo de herramienta que busca y el tópico.	
Precondiciones:	No tiene.
Postcondiciones:	Se despliegan los resultados
Secuencia exitosa.	
Experto	**Sistema**
1. Este caso de uso comienza cuando el usuario rellena los campos necesarios (palabras claves y tipo de herramienta) para la búsqueda y presiona **buscar**.	
	2. El sistema encuentra los resultados según lo ingresado y lo muestra en pantalla. Termina el proceso.
Secuencias alternativas.	
	2.1.1 El sistema no es capaz de encontrar lo pedido y se muestra un mensaje en pantalla. Termina el proceso.
	2.1.2 El sistema detecta que no se ingresó string de búsqueda y muestra un mensaje de advertencia. Termina el proceso.

Caso de uso: Gestionar lista negra.	
Descripción: El usuario podrá gestionar los elementos presentes en la lista negra del sistema.	
Precondiciones:	Estar en la pantalla de gestión de la lista negra Tener elementos en la lista negra.
Postcondiciones:	Reflejar los cambios en la lista negra
Secuencia exitosa.	
Experto	**Sistema**
1. Este caso de uso comienza cuando el usuario desea quitar un elemento(s) de la lista negra y selecciona todos los que quiera	
2. El usuario presiona el botón **Desbloquear seleccionados**.	
	3. El sistema elimina el elemento(s) seleccionado(s) de la lista negra. Termina el proceso
Secuencias alternativas.	
2.1 El usuario se arrepiente de su decisión y presiona el botón "**Volver al inicio**".	
	3.2 El sistema vuelve a la pantalla de inicio. Termina el proceso.

El diagrama de casos de uso no presenta cambios desde su versión inicial. Todos los casos de uso presentados son cumplidos en su totalidad.

3. Patrones de diseño y Frameworks (final)

Intención	Patrón de Diseño	Razonamiento
Se desea mostrar en el diagrama de clases una interacción ordenada y separada del sistema	MVC	El patrón MVC satisface dicha necesidad ayudando a separar el sistema en tres componentes principales (modelo, vista y controlador) lo cual facilitará el desarrollo para cumplir todos los requisitos funcionales.
Se desea mostrar en el diagrama de clases la interacción con la base de datos para manejar falsos positivos	Active Record	Se decide utilizar Active Record para gestionar las interacciones con la base de datos.

Tabla 1: Patrones de diseño

Además se utiliza el framework "Scrappy" para realizar la recolección de información desde páginas web de confianza. Sin embargo, no se encuentra disponible en la documentación los patrones de diseño utilizados por este.

4. Modelo de Dominio y Diagrama de clases (final)

4.1. Modelo de dominio

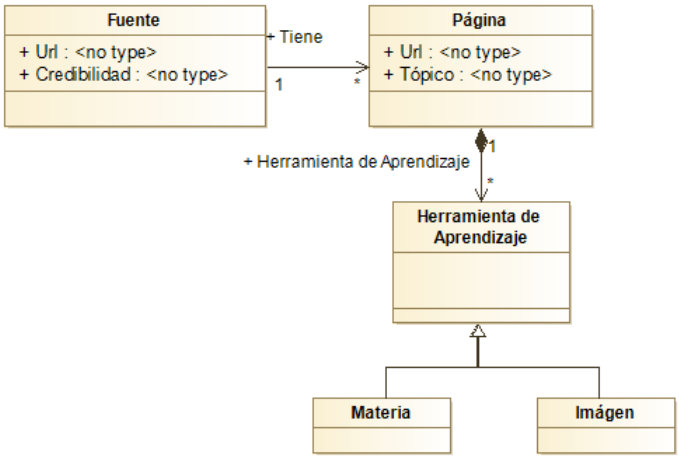


Figura 4: Modelo de dominio versión 2.0. Para ver con más detalle revisar el diagrama de clases en el proyecto de Modelio "Quantum Taco".

El modelo de dominio presenta un solo cambio desde la versión 1 (ver entregable 1), se elimina la herramienta de aprendizaje "Gráfico" dado que no se implementó como herramienta de aprendizaje.

4.2. Diagrama de clases

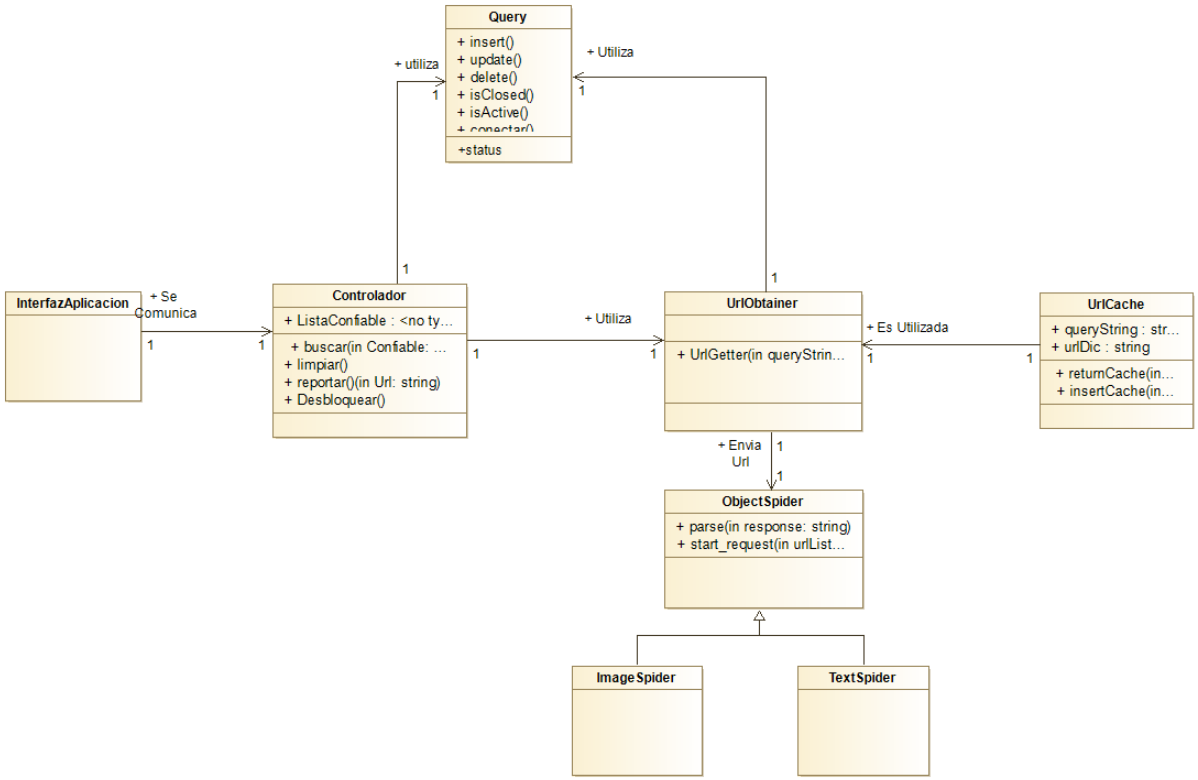


Figura 5: Diagrama de clases versión 2.0 Para ver con más detalle revisar el diagrama de clases en el proyecto de Modelio "Quantum Taco"

Este diagrama presenta varias diferencias con la versión 1.2 (ver entregable 2). Se optó por manejar las fuentes (lista de páginas confiables y páginas banneadas) como atributos dentro del controlador, dado que no tenía sentido mantenerlas como clases al no tener métodos. También se agrega la clase "UrlCache" al detectar la necesidad de mantener un banco de url para reutilizar en vez de realizar las consultas que ya se han realizado anteriormente. Finalmente se elimina la sub–clase "Table Spider" al no implementar la búsqueda de dicha herramienta de aprendizaje.

El modelo MVC se ve con la InterfazAplicación (vista), Controlador (controlador) y el modelo sería el conjunto de clases que rescata info desde las páginas web de dominios de confianza(sacadas desde la base de datos).

De igual manera, se muestra en el diagrama el patrón Active record, donde la clase Query es la encargada de interactuar con la base de datos.

5. Pruebas de Software (actualización)

Dadas las condiciones de término de semestre no fue posible coordinar el trabajo conjunto con el consultor QA. Se adjunta la tabla resumen del trabajo anterior realizado.

Defecto encontrado	Mitigación	Resultado obtenido	Observaciones
Problema al intentar buscar con un string vacío. Se genera un error al salir de los índices de una tupla.	Detectar cuando el campo de búsqueda está vacío, si esto ocurre se muestra un mensaje de alerta por pantalla.	Se corrige correctamente el defecto, de esta manera se evita que se caiga el sistema.	Este defecto fue detectado por el equipo de desarrollo, el trabajo del consultor QA sirvió para recordar que este problema no había sido arreglado.
Problema con búsqueda sin resultados. Cuando una búsqueda no arroja resultados se muestra una vista vacía y no se notifica al usuario de que su búsqueda no obtuvo resultados.	Se detecta cuando no se obtienen resultados, se muestra un mensaje de notificación en la pantalla principal, no permitiendo acceder a la vista de resultados	Una vez realizada esta modificación se mitiga correctamente este defecto.	Al igual que el defecto anterior, este ya había sido detectado anteriormente. Sin embargo, al arreglar este defecto surge un nuevo defecto que se detalla en la siguiente fila.
Problema con string puntual. Al momento de realizar la prueba anterior, el consultor QA utilizó el string "asd", cuando se realiza la búsqueda se obtenía un url que no estaba dentro de los dominios de confianza, pero se detectaba como resultado. Internamente no tenía contenidos relevantes por lo que no se desplegaban en la vista de resultados	Se agregó una nueva verificación en la clase urlObtainer para detectar si efectivamente la url encontrada corresponde a un dominio de confianza.	Una vez implementada la segunda verificación se arregló correctamente este defecto.	Este defecto se produjo por una mala coincidencia, el trabajo realizado por el QA fue de gran ayuda para detectar y arreglar este defecto.