UNIVERSIDAD DE CHILE
FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS
DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN

SISTEMA RECOMENDADOR PARA APOYAR EL SCOPING DE PRODUCTOS DE SOFTWARE

TESIS PARA OPTAR AL GRADO DE

MAGÍSTER EN TECNOLOGÍAS DE LA INFORMACIÓN

JOAQUÍN BAUTISTA ROMERO FRANCO

PROFESORES GUÍA:
SERGIO OCHOA DELORENZI
DANIEL PEROVICH GEROSA

MIEMBROS DE LA COMISIÓN:
JOCELYN SIMMONDS WAGEMANN
FRANCISCO GUTIÉRREZ FIGUEROA
ALCIDES QUISPE SANCA

SANTIAGO DE CHILE
2023

RESUMEN DE LA TESIS PARA OPTAR
AL GRADO DE MAGISTER EN
TECNOLOGÍAS DE LA INFORMACIÓN
POR JOAQUÍN BAUTISTA ROMERO FRANCO
FECHA: 2023
PROFESORES GUÍA: SERGIO OCHOA D.
DANIEL PEROVICH G.

Sistema recomendador para apoyar el scoping de productos de software

El mercado del software crece día a día, realizándose más proyectos de software, los que necesitan dimensionar los productos que van a desarrollar. Este dimensionamiento, llamado scoping del producto, se aborda usualmente con técnicas de ingeniería de requisitos tradicionales, las cuales han mostrado ser poco costo-efectivas, particularmente aplicadas en la etapa de pre-venta de un proyecto de software.

Para ayudar a mejorar la costo-efectividad de dicha actividad, en este trabajo de tesis se diseñó e implementó un servicio de recomendación, el cual fue embebido en un tablero digital (sistema de software tipo canvas) que apoya la actividad de scoping. El servicio creado le sugiere distintos componentes o servicios a incluir en el sistema a ser presupuestado o desarrollado según sea el caso.

La herramienta, y particularmente el servicio de recomendación, fue evaluado a través de tres experimentos diferentes, realizados a pares cliente-desarrollador, y un proyecto de software simulado. El objetivo fue realizar, de forma conjunta, la definición del alcance del producto utilizando el servicio de recomendación. Al finalizar cada sesión los participantes completaron una evaluación que buscaba capturar sus impresiones.

Los resultados obtenidos fueron positivos. Los evaluadores consideraron al servicio de recomendación usable y útil, manifestando estar dispuestos a utilizarlo en futuras actividades de scoping. Los resultados también mostraron aspectos de mejora, tanto del servicio de recomendación, como la aplicación base.

De acuerdo a la opinión de los participantes en el proceso de evaluación, se cumplieron los objetivos inicialmente definidos, es decir, mejorar la costo-efectividad del proceso de scoping. También es importante reconocer que los resultados experimentales aún son pocos, producto que se buscó que los participantes tuvieran experiencia previa utilizando el tablero digital, lo que acotó mucho la población disponible para las pruebas. La continuación de este proceso de pruebas y ajuste de la solución es parte del trabajo a futuro declarado en esta tesis.

Tabla de Contenido

1 Introducción 1
1.1 Contexto del trabajo 1
1.2 Problema abordado 3
1.3 Objetivos de la tesis 4
1.4 Resumen de la solución desarrollada 5
1.5 Estructura del documento 6
2 Marco Teórico 7
2.1 Escenarios de scoping 7
2.2 Proceso de scoping 8
2.3 Desafíos del proceso de scoping 9
2.4 Descripción del sistema legado 10
2.5. Sistemas recomendadores 13
3 Concepción de la Solución 17
3.1 Requisitos y restricciones de la solución 19
3.2 Perfiles de usuarios a soportar 20
3.3 Arquitectura de la solución 20
3.4 Modelo de datos 22
3.5 Tecnologías seleccionadas 24
4 Implementación de la solución 25
4.1 Biblioteca de cápsulas 25
4.1.2 Colecciones 25
4.1.2 Endpoints expuestos 25
4.2 El sistema de recomendación 28
4.3 Actualización e integración con la aplicación legada 29
4.3.1 Seleccionar cápsulas de recomendación 30
4.3.2 Obteniendo recomendaciones en el tablero 32
4.3.3 Menú de recomendaciones 33
4.4. Nivel de cumplimiento de los requisitos 34
5 Evaluación de la solución 36
5.1 Caracterización de los participantes 36

5.2 Evaluación realizada 37
5.3. Instrumento de evaluación utilizado 38
5.4 Resultados obtenidos 40
5.5 Discusión 41
5.6 Limitaciones de la evaluación 42
6 Conclusiones y trabajo a futuro 44
Bibliografía 46
Anexos 50
Anexo A - Especificación del formato YAML para la creación de cápsulas 50
Anexo B - Caso utilizado en roleplaying 52
B.1 Cliente 52
B.2 Developer 52
Anexo C - Encuesta aplicada sobre el sistema recomendador 53
Anexo D - Resultados de la evaluación 55
D.1 Usuarios desarrolladores 55
D.2 Usuarios clientes 56

1 Introducción

1.1 Contexto del trabajo

La industria de software se ha vuelto altamente competitiva y global, lo que hace que las empresas de desarrollo (especialmente las más pequeñas) tengan que hacer un esfuerzo importante y permanente por sobrevivir a los desafíos del mercado [15]. Esta competitividad las fuerza a presupuestar proyectos con márgenes bastante acotados, principalmente en términos de tiempo de desarrollo y dinero. Para presupuestar proyectos en esas condiciones, las empresas deben lograr un alto grado de certeza, tanto del problema u oportunidad a abordar, como del objetivo y alcance del producto a desarrollar.

Para alcanzar esta certeza las empresas realizan, durante la etapa de preventa o pre-proyecto, un proceso de levantamiento y negociación de requisitos con el cliente. A esta actividad se la conoce como scoping [35], la cual busca captar las necesidades del cliente, y definir las principales características y capacidades de la solución de software, así como su objetivo.

Idealmente, la actividad de scoping debe ser de costo bajo, ya que en la etapa de preventa el costo de la exploración del problema, así como el dimensionamiento y la estimación de la solución a desarrollar, es absorbido por el proveedor. Si la oferta de desarrollo no se materializa en un contrato, entonces el scoping de ese proyecto se traduce en una pérdida de tiempo y de recursos importante para la organización oferente [25]. Por otra parte, el proceso de scoping debe ser altamente efectivo, ya que dependiendo de la certeza que se tenga sobre los requisitos capturados, se podrá proponer una solución que resuelva efectivamente el problema del cliente, y que sea construida dentro de tiempos y costos predecibles.

Para lograr una alta costo-efectividad en el proceso de scoping, se requiere usar técnicas y herramientas que puedan aumentar la certeza sobre el problema a abordar y su solución, al menor costo posible. Para este objetivo se requiere además contar con expertise en el dominio de negocio, y en el dominio técnico del tipo de sistema que se pretende desarrollar. Lamentablemente, para una empresa de desarrollo es imposible tener expertos en todos los dominios de negocio y tipos de sistema en los que debe presupuestar proyectos, por lo que estas compañías usualmente usan técnicas de Ingeniería de Requisitos para realizar esta labor. Aunque estas técnicas han demostrado su utilidad durante el desarrollo de un proyecto de software, éstas no son costo-efectivas cuando se usan durante la etapa de preventa, donde las condiciones de uso son diferentes [23, 24].

Una técnica novedosa y costo-efectiva para clarificar el alcance de un producto durante la preventa fue propuesta en [36]. Esta técnica está basada en dinámicas colaborativas donde se utilizan tableros tipo Canvas (Fig. 1).

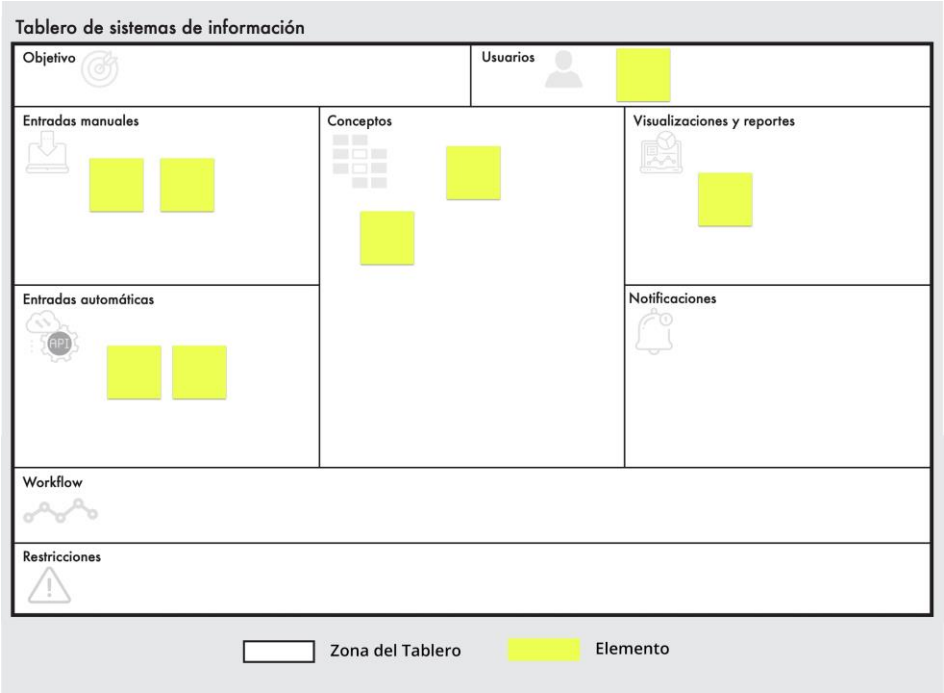

Figura 1: Ejemplo de tablero diseñado para abordar el scoping de sistemas de información.

Las dinámicas consisten en realizar sesiones de trabajo donde se capturan los componentes de la solución (sistema a construir), utilizando tableros diseñados para abordar dominios de soluciones particulares (es decir, tipos de sistemas), por ejemplo, tableros enfocados a capturar la funcionalidad y los componentes de sistemas de business intelligence, de sistemas de información, etc.

Estos tableros tienen un layout distinto dependiendo del tipo de sistema del cual se pretende capturar la funcionalidad. Cada layout tiene zonas predefinidas, que se ajustan a una arquitectura típica para ese tipo de sistema. Por ejemplo, en una sesión de trabajo para la creación de un sistema de control de mercadería para una bodega, se utilizará el canvas de la Figura 1, que corresponde a un sistema de información.

Los sistemas de información típicamente cuentan con: 1) mecanismos de alimentación de su repositorio de datos (entradas), 2) el repositorio propiamente tal (conceptos que maneja el sistema), 3) salidas de datos (visualizaciones, reportes, notificaciones, etc.), y 4) componentes autónomos que transforman o generan de datos (workflows). Estos sistemas también cuentan con restricciones, que corresponden a requisitos no-funcionales.

En las distintas zonas del tablero se identifican, a través de post-its, los distintos componentes de la solución según corresponda. Por ejemplo, en la zona de *"entradas manuales"* se podrían colocar los formularios web que permiten realizar el CRUD de los ítems en bodega. En *"entradas automáticas"* se podría colocar un post-it con las facturas que llegan desde el servicio de impuestos internos. Luego, a través de estos dos mecanismos de entrada de datos se podrían actualizar múltiples *"conceptos"* (usualmente, tablas en un modelo relacional) como, por ejemplo: facturas de venta, facturas de compra, mercadería en bodega, etc. Los conceptos también pueden actualizarse a través de procesos autónomos (workflows) que, por ejemplo, calculen índices o valores de variables particulares una vez al día.

Finalmente, estos conceptos servirán para crear *"visualizaciones y reportes"* como dashboards de gestión de inventario, y *"notificaciones"* por ejemplo para cuando el stock esté más bajo de lo recomendado.

El llenado de las distintas zonas del tablero, que representa al producto a desarrollar, es realizado por el cliente y el desarrollador, durante una o más sesiones de trabajo. Esto permite que la solución final contenga el conocimiento compartido entre ambas partes, confiando en el expertise tanto del cliente como del proveedor. Típicamente, el cliente tiene mayor conocimiento sobre los conceptos claves de su negocio (por ejemplo, los documentos tributarios que maneja), y el desarrollador los aspectos técnicos del producto a desarrollar (por ej., conoce qué API acceder para obtener los documentos de manera automática). De esa manera, ambas partes se complementan

Entre los beneficios de usar esta técnica está el hecho de que ésta tiene una barrera de entrada baja, debido a que solo requiere entender las zonas de un tablero. Además, el costo de uso es bajo, ya que la única herramienta necesaria para aplicar dicha técnica es una aplicación de software que implementa una versión digital del tablero, y la actividad de llevado del mismo se realiza en una o dos sesiones de trabajo entre el cliente y el desarrollador. Finalmente, esta técnica ha mostrado buena efectividad, ya que permite definir la solución, su alcance y sus requisitos, utilizando un lenguaje común entre el proveedor y el cliente.

1.2 Problema abordado

Las dinámicas colaborativas con el uso de tableros logran ser costo-efectivas en el scoping de proyectos de software, debido a que encapsulan el conocimiento clave sobre los tipos de componentes en las zonas del tablero, dado un tipo de sistema determinado [36]. Esto permite reutilizar los tableros y reducir el expertise necesario para dimensionar un cierto tipo de sistema; es decir, hacer el scoping de ese producto. Sin embargo, el determinar los componentes del software a construir (en este caso, los elementos que deben ir en cada zona del tablero), depende completamente del expertise que, tanto el

proveedor como el cliente, tenga sobre el dominio del problema y la solución [7, 32]. Esto limita a que, quienes tengan que realizar la dinámica en el equipo del proveedor, deban tener conocimiento del dominio abordado. De esa manera, es posible proveer valor al cliente a través de la correcta definición del problema. Dicho problema puede ser distinto al punto de dolor identificado por el cliente.

Para apoyar a los miembros del equipo que no tengan la expertise necesaria para abordar un cierto dominio de aplicación, es necesario diseñar una herramienta que permita almacenar y reutilizar el conocimiento del dominio, en forma de elementos de una zona del tablero. Junto con esto, la herramienta debe poder recomendar componentes durante una sesión de scoping, facilitando así la identificación de elementos de la solución a desarrollar. La Figura 2 muestra una sesión de trabajo entre representantes del cliente y el proveedor, en la cual ellos están completando un tablero (o sea, realizan el scoping) de un sistema particular.

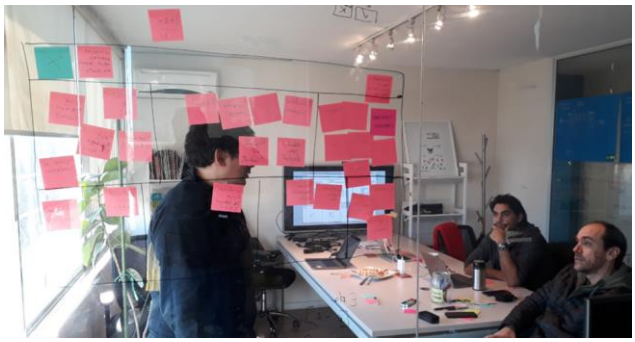

Figura 2: Sesión de scoping de un sistema de procesamiento masivo de datos (tomado de [36]).

El reúso de conocimiento permitirá aumentar aún más la costo-efectividad de esta herramienta, reduciendo la barrera de entrada para su uso, promoviendo una definición ágil y colaborativa de la solución, y mejorando la capacidad de descubrimiento de los elementos que forman parte del alcance del proyecto.

1.3 Objetivos de la tesis

El objetivo general de este trabajo de tesis es facilitar el proceso de definición de alcance de un producto, en el marco de un proyecto de software, a través del uso de recomendaciones de elementos (requisitos) para la dinámica colaborativa apoyada en un tablero digital. Particularmente, se busca apoyar de mejor manera el proceso de elicitación, cuando éste es llevado a cabo por personas con conocimiento limitado sobre el dominio técnico y de negocio abordado.

En ese escenario, la mejora planteada consiste en extender una aplicación existente del tablero digital con un sistema recomendador, que utilice una base de conocimiento de

elementos reutilizables, con el fin de entregar sugerencias de funcionalidad a incluir en el producto a desarrollar. Se espera que esta extensión ayude a reducir el tiempo de elicitación, aumentar el número de requisitos identificados durante la actividad, y bajar el nivel de expertise necesario para abordar el scoping de un sistema para un dominio de aplicación particular.

Para alcanzar el objetivo general, se han definido los siguientes objetivos específicos:

1. Definir la estructura de un repositorio de conocimiento reusable, que ayude a generar sugerencias de funcionalidad a incluir en el producto, durante el proceso de elicitación de un proyecto. En una primera instancia, este repositorio contendrá los elementos potencialmente reutilizables en sistemas de un cierto dominio de negocio. A futuro se podrán incorporar componentes de varios dominios.
2. Desarrollar un recomendador de elementos de un tablero (componentes reutilizables), utilizando la estructura definida anteriormente como repositorio disponible.
3. Extender la aplicación actual del tablero digital con el recomendador desarrollado, para que ésta facilite la construcción colaborativa del alcance de un producto; particularmente, para personas con conocimiento limitado del dominio técnico y de negocio abordado.

1.4 Resumen de la solución desarrollada

Como se mencionó antes, en este trabajo de tesis se extendió la aplicación llamada tablero digital, que implementa tableros tipo canvas para apoyar la captura de requisitos, de forma colaborativa, entre el cliente y el desarrollador. La extensión involucró el desarrollo, y su posterior integración, de sistema recomendador de elementos (post-its) para las distintas zonas del tablero (Fig. 3).

El servicio de recomendación cuenta con un cuerpo de conocimiento estructurado (recomendaciones almacenadas en una BD), en base al tipo de tablero seleccionado y al dominio de negocio abordado. Éste se despliega en la zona gris del tablero, y los participantes deciden si aceptan o no las recomendaciones sugeridas por el sistema. Aquellas que sean aceptadas deben ser colocadas en el tablero, en la sección que corresponda. De esa manera se facilita el proceso de construcción colaborativa del alcance del producto a desarrollar.

Además del desarrollo del mecanismo de recomendación, se tuvieron que realizar múltiples cambios a la interfaz de usuario del sistema, y cambios menores a la aplicación de backend, para poder integrar la selección de cuerpos de conocimiento y recomendaciones.

Figura 3: Escenario de uso del sistema recomendador

1.5 Estructura del documento

Este documento de tesis está estructurado en 6 capítulos. El Capítulo 2 presenta el marco teórico, donde se ahonda en la especificación de requisitos en la etapa de scoping, y en el potencial reúso de conocimiento en esta etapa. Además, se introducen los sistemas recomendadores, y un conjunto de desafíos de diseño que éstos presentan, similares a los abordados en esta tesis.

El Capítulo 3 describe la solución diseñada, y el Capítulo 4 presenta la implementación de dicha solución. En el Capítulo 5 se introduce el instrumento de evaluación utilizado, sus resultados y limitaciones. Finalmente, en el Capítulo 6 se exponen las conclusiones de este trabajo y se detallan las alternativas de trabajo a futuro.

2 Marco Teórico

En este capítulo se explican en detalle los conceptos involucrados en este trabajo de tesis, iniciando con el escenario y el proceso de scoping, y terminando con los sistemas de recomendación que pueden ser utilizados para apoyar este proceso.

2.1 Escenarios de scoping

Un ciclo de vida puede ser definido como la evolución de un sistema, producto, servicio o proyecto. Esta evolución va desde su concepción hasta su término (o retiro del mercado/organización) [19]. Los ciclos de vida pueden ser divididos en etapas, que usualmente se relacionan con el estado de avance del proyecto o producto. Por ejemplo, desde una perspectiva muy general, en el ciclo de vida de un proyecto de software se identifican 3 etapas: *pre-contrato*, *contrato* y *post-contrato* [10, 31], donde en las dos primeras se genera la actividad de scoping del producto (Fig. 4).

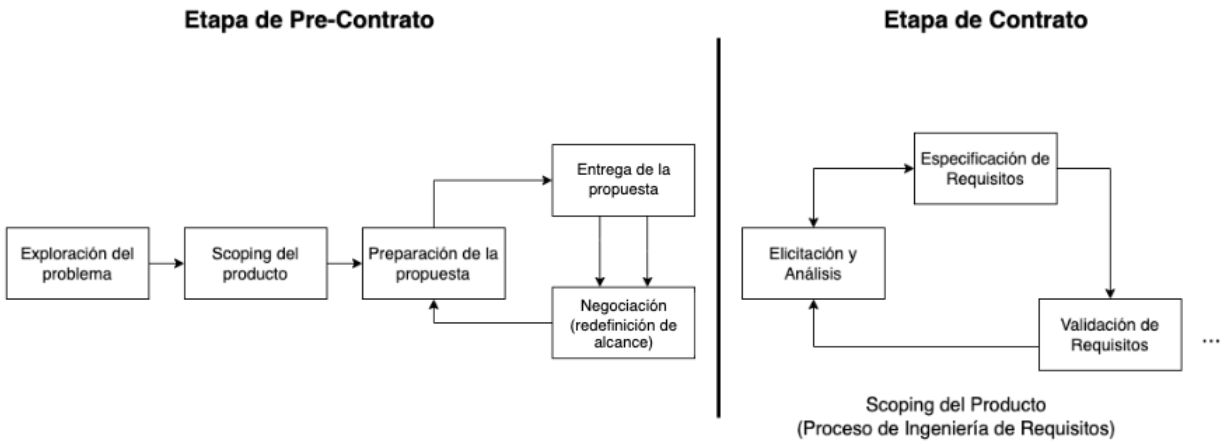

Figura 4: Etapas del ciclo de vida de un proyecto de software donde se realiza scoping (basado en [31]).

La etapa de *pre-contrato* involucra todas las actividades que se realizan antes de la venta del proyecto al cliente, e incluye todo lo necesario para determinar (de manera preliminar) el alcance del producto requerido, para así poder realizar una oferta. Esta etapa se subdivide en distintas actividades [31]: la *exploración del problema*, donde se identifica el dolor u oportunidad que tiene el cliente y el contexto en que ésta se desarrolla. Una vez se realiza el análisis del problema, el proveedor junto con el cliente deben *determinar el alcance del producto o solución* a construir (conocido como scoping).

Una vez que las funcionalidades y los límites de la solución están definidos, el proveedor debe desarrollar una propuesta de proyecto. Durante la presentación de la propuesta se

deben estimar los costos, tiempos y recursos a utilizar. En esta etapa se pueden realizar ciclos de refinamiento del alcance del producto o proyecto, a través de una negociación entre las partes, con el fin de llegar a una propuesta más adecuada y factible para ambos.

Es importante destacar que el scoping realizado en la etapa de pre-contrato, es considerado como uno de los factores más importantes que inciden en el éxito o fracaso de un proyecto de software [2, 34, 29]. Realizar mal la etapa de scoping, ya sea durante el pre-contrato, como durante el contrato mismo, genera deuda de requisitos (que es parte de la deuda técnica del proyecto) [12].

En la etapa de *contrato* es cuando la empresa proveedora comienza la implementación de la solución. Independientemente de la metodología a utilizar para el desarrollo, en esta etapa se realiza nuevamente la actividad de scoping, pero a diferencia de la etapa anterior, esta vez se realiza un análisis profundo de los requisitos de la solución, es decir, la ingeniería de los requisitos.

2.2 Proceso de scoping

El PMBOK define el scoping o alcance del producto como las características y las funcionalidades que deben ser incluidas como parte del producto o servicio a desarrollar [27]. El trabajo reportado en [35] muestra que la etapa de scoping involucra tres macros/actividades (Fig. 5): el *descubrimiento de la solución*, la *definición del alcance del producto*, y la *validación de dicho alcance*. Las tres actividades requieren interacción e intercambio de conocimiento entre el cliente y el proveedor. Esto se hace sin un workflow claro (o predefinido) entre las actividades, pues la dinámica se va generando en el momento de la reunión, en base a las necesidades de información que necesita abordar cada parte (es decir, el cliente o el proveedor).

Usualmente, el conocimiento intercambiado entre las partes tiene que ver con aspectos de negocio; por ejemplo, entidades de datos, conceptos, procesos/servicios, e indicadores que el sistema debería manejar. En dominios de negocios maduros, estos componentes deberían poder identificarse y reutilizarse. Por ejemplo, en el ámbito inmobiliario uno puede hablar de inmuebles, dueño, arrendatario, contrato, pago, vencimiento, servicios, etc.

Lo anterior no impide que ambas partes intercambien conocimiento sobre aspectos técnicos del proyecto o del producto. Sin embargo, eso es menos frecuente, pues se requiere que el cliente cuente con personal técnico, el cual tenga el expertise y disponibilidad suficiente como para participar de esas sesiones de trabajo. Por esa razón, en general se da lo primero.

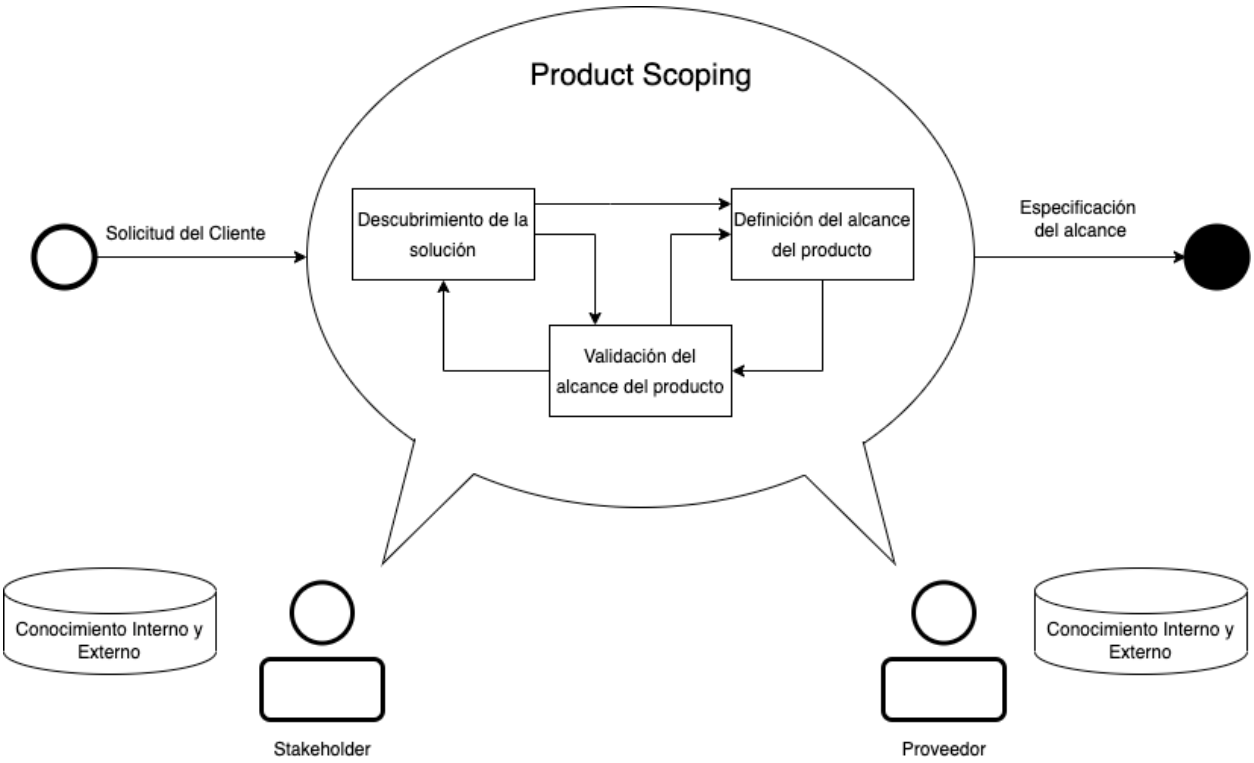

Figura 5: Proceso de scoping (basado en [35]).

Para evitar diferencias de interpretaciones respecto al significado de los elementos que forman parte del scope del producto, así como a la relación entre ellos, estos aspectos deben ser especificados [5]. Para definir el scope e identificar los elementos de negocio que se van a considerar en el proyecto, a menudo se utilizan distintas técnicas de ingeniería de requisitos [6], como, por ejemplo: entrevistas, workshops, e inclusive proyectos piloto [17].

2.3 Desafíos del proceso de scoping

A continuación, se describen los principales riesgos y desafíos del proceso de scoping en la etapa de pre-contrato:

- El *objetivo cambia frecuentemente,* pues tanto el proveedor como el cliente están en la etapa temprana, donde clarifican tanto el problema como la solución a éste [25].
- *Los tiempos para analizar el problema/solución son limitados,* debido a que aún no se ha generado un contrato, y por ende no se puede correr el riesgo de asignar muchos recursos a esta actividad. La escasez de tiempo genera dificultades a la hora de hacer el análisis, lo que agrega un alto nivel de incertidumbre [23].

- El scoping representa una *inversión previa a cualquier retorno,* tanto para el equipo proveedor como para el cliente. Por ende, si el trato no se llega a cerrar, todo esfuerzo significa una pérdida de recursos [25, 30]. Debido a esto, ambas partes usualmente intentan minimizar riesgos utilizando la menor cantidad de recursos posibles para alcanzar sus objetivos individuales.

- *El resultado del scoping depende fuertemente de la expertise de los participantes* [4]. Esto hace que los proveedores con poco expertise en el dominio del cliente, ingenieros novatos, o miembros del equipo del cliente sin el suficiente conocimiento, tengan pocas chances de realizar el scoping de un proyecto de manera costo-efectiva.

- *La confianza entre proveedor y cliente es limitada* debido a que frecuentemente representa el inicio de una relación de negocios. Si es una relación nueva hay que considerar acrecentar la confianza durante las actividades de scoping [23].

Dados los desafíos y riesgos existentes al momento de iniciar un proyecto que aún no ha sido vendido, se hace imperativo para el proveedor y el cliente realizar la actividad de scoping de forma lo más costo-efectiva posible. Una manera de acercarse a esa costo-efectividad requerida, es utilizando dinámicas colaborativas o workshops de requerimientos. En la siguiente sección se detalla el tablero digital (sistema legado), el cual apoya la dinámica descrita en [36] para llevar a cabo el scoping en la etapa de pre-contrato.

2.4 Descripción del sistema legado

El tablero digital permite definir de forma colaborativa el objetivo y el alcance de un producto a desarrollar, posibilitando así la realización del scoping. A continuación, se describe el flujo típico de uso de la herramienta, y sus principales funcionalidades.

El acceso a la aplicación requiere autenticación, y por ende, el primer paso es crear un usuario e ingresar al sistema con las credenciales correspondientes. Una vez ingresado, el sistema muestra el panel principal que le permite al usuario crear un tablero nuevo o seleccionar uno ya creado (Fig. 6).

Para crear un tablero nuevo es necesario ingresar un título (nombre corto del tablero), una descripción, y elegir un tipo de layout (template). Este último muestra la disposición de un conjunto de paneles en el tablero, donde cada panel tiene un significado particular. Cada tipo de layout representa un tipo de sistema (por ej. un sistema de información, de procesamiento masivo de datos, o de internet de las cosas), y sus tipos de componentes. La estructura de un layout está basada en la arquitectura típica (o de referencia) del tipo de sistema que éste representa. La aplicación también permite crear distintos layouts de

tableros utilizando el módulo de edición de layouts, el cual forma parte de la aplicación Tablero Digital, y está disponible para los usuarios de tipo *administrador*.

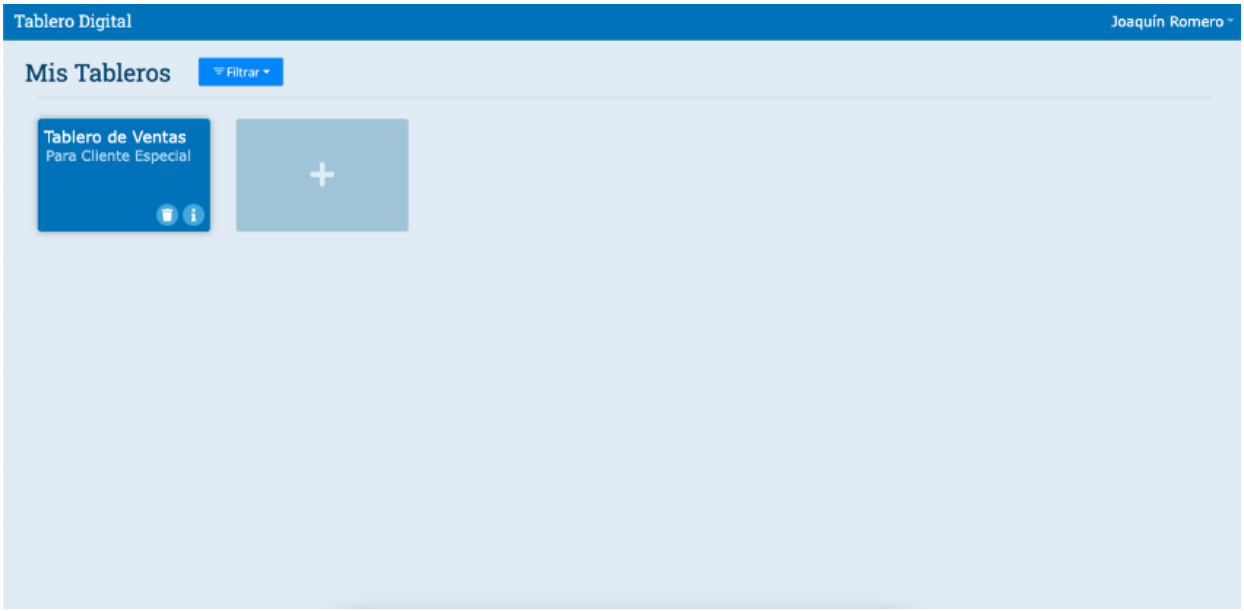

Figura 6: Dashboard de aplicación de tableros.

Una vez creado un tablero en base a un cierto layout, el sistema muestra las distintas zonas del mismo. En la Figura 7 podemos ver un tablero para definir los componentes de un sistema de procesamiento masivo de datos. En dicho tablero las zonas son las siguientes: *Objetivos*, *Usuarios*, *Fuentes de Datos*, *Generador de Conceptos*, *Conceptos*, *Generador de Métricas, Métricas, Visualización*, *Notificación y Acción*, y *Restricciones*. Estos espacios son los contenedores de elementos del tablero, por ende, aquí es donde se permite agregar los requisitos (o componentes de la solución).

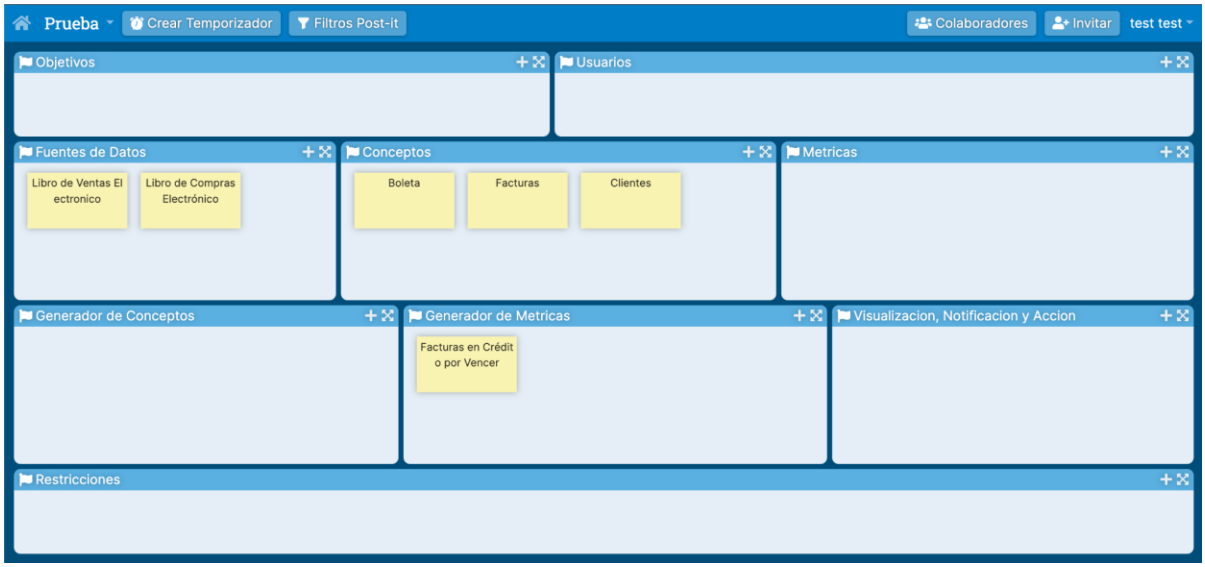

Figura 7: Tablero de prueba de aplicación de tableros (versión legada).

El sistema considera a cada tablero como una entidad diferente, a la cual pueden acceder varias personas por invitación. Estos invitados pueden ser usuarios registrados o no registrados. En este último caso, a las personas se les comparte por correo un link para que puedan acceder al tablero al cual se los está invitando. Al ingresar un nuevo usuario a un tablero particular, se le pregunta a éste si es parte del equipo de los stakeholders (cliente) o de los desarrolladores (proveedores), tal como se muestra en la Figura 8.

Para mantener la sincronía en las actualizaciones de post-its sobre un tablero, se utiliza un servidor de websockets. Esto permite ver en tiempo-real el estado del tablero mientras los usuarios actualizan los post-it o crean nuevos elementos.

Actualmente la aplicación solo guarda los registros en su base de datos, acerca de qué usuario genera o actualiza cada post-it. La secuencia de cambio sobre un post-it, por el momento, no se ve reflejado en ninguna parte de la interfaz o registrado en un logfile. Se decidió hacerlo de esa manera, dado que la construcción del tablero es un proceso creativo y evolutivo, que involucra muchos cambios sobre los post-its. Hasta el momento, el uso del tablero por parte de clientes y desarrolladores no ha mostrado la necesidad de mantener un logfile, pero éste se puede implementar fácilmente en caso de ser requerido.

Figura 8: Selección de equipos en aplicación de tableros.

Además de crear, modificar y eliminar post-its, algunos usuarios, particularmente los líderes de equipo, pueden votar acerca de la inclusión o no de un post-its como parte del alcance del producto a desarrollar. Una vez que un líder de equipo ha votado por mantener un cierto post-it, éste no puede ser eliminado mientras no se retire el voto. Los post-its no votados, es decir, no acordados entre las partes, pueden ser modificados y/o

eliminados sin restricciones. Por otra parte, los líderes de equipo pueden sobrescribir su voto en el momento que lo deseen, si lo estiman conveniente.

2.5. Sistemas recomendadores

Una manera de ayudar a bajar la barrera de entrada para el uso del tablero digital, y facilitar el descubrimiento de elementos a incluir en un proyecto, es a través de recomendaciones asistidas por un software. Estos sistemas son conocidos como *Sistemas Recomendadores* (o *Recommender Systems* - RS) [22].

Los RS son utilizados principalmente para sugerir/recomendar "*ítems"* (es decir, cualquier tipo de objeto) a un usuario, a través de alguna técnica, buscando que la recomendación le sea de utilidad para conseguir un propósito [21]. Los RS están enfocados principalmente a usuarios que no tienen la suficiente experiencia personal, o las competencias para evaluar múltiples alternativas. Ejemplos comunes de RS son los que ofrece Amazon (para sus artículos de retail) [33], Netflix (para series y películas) [16], o Spotify (para canciones) [20].

En [18] se definen las principales razones por las que un usuario quisiera utilizar un RS, éstas son las siguientes:

> *Anotaciones en contexto:* Enfatizar ítems dentro de una lista de varios de estos, dado un contexto. Por ejemplo, dentro de una lista de artículos a leer, mostrar los más interesantes dado el perfil del usuario.
>
> *Encontrar ítems útiles:* Recibir listas de sugerencias de ítems específicos que el usuario puede considerar útiles.
>
> *Encontrar todos los ítems útiles:* Recomendar todos los ítems que pueden considerarse útiles para el usuario dentro de un catálogo completo de ítems.
>
> *Recomendar una secuencia:* Recomendar una secuencia de ítems que presenten valor al usuario, en vez de recomendar solo uno; por ejemplo, la recomendación de una secuencia de canciones o videos que el usuario podría encontrar atractiva.
>
> *Solo navegar:* Si bien los recomendadores son evaluados en base a cuando un usuario toma una acción final respecto al ítem recomendado, por ejemplo, comprar, o consumir, algunos usuarios utilizan los recomendadores para navegar el catálogo de ítems recomendados.
>
> *Encontrar un recomendador creíble:* Dado que algunos usuarios suelen no confiar inmediatamente en las recomendaciones obtenidas, es común que el usuario "juegue" con el recomendador para evaluar la calidad de sus recomendaciones.

Mejorar el perfil: En el caso de recomendadores que utilizan las preferencias de los usuarios, los usuarios suelen utilizar el recomendador para proveer inputs y obtener una mejor personalización en sus recomendaciones.

Expresarse: A algunos usuarios no les importan las recomendaciones, si no que buscan contribuir con su input (ratings, o opiniones) para expresar su opinión.

Ayudar a otros: Algunos usuarios buscan contribuir con su input, ya que creen que la comunidad se beneficia con su contribución.

Influenciar a otros: Algunos usuarios buscan explícitamente influenciar a través de su input a otros usuarios para que obtengan o no una recomendación dada. Por ejemplo, mejorar el rating de una serie en Imdb (una página web que es un catálogo de películas), para que esta serie aparezca recomendada a más usuarios, y por ende, sea más vista.

Como fue mencionado anteriormente la función principal de un RS es sugerir un ítem que el usuario encuentre útil. Si bien el proceso de recomendación puede variar dependiendo de la técnica utilizada (lo veremos más adelante), en [1] se presenta una generalización del proceso de cada recomendador. Considerando eso, en [26] se presenta una revisión de los tipos de sistemas de recomendación, los desafíos aún pendientes, y las directrices de investigación futura en esta área.

El proceso de recomendación puede resumirse como se indica a continuación. Dado un conjunto de todos los usuarios C y un conjunto de todos los posibles ítems I, la función del recomendador es utilizar una función de utilidad u que obtenga el valor para un ítem i para el usuario:

$$u : C \times I \rightarrow R$$

donde R es el conjunto ordenado de recomendaciones.

Finalmente, para cada usuario c, queremos escoger items i' que maximicen la utilidad. Esto se representa formalmente de la siguiente manera:

$$\forall\, c \in C,\ i'_c = arg\, max_{i \in I}\, u(c, i)$$

Se presenta además que el problema central de los recomendadores es que la función de utilidad, no solo está dada por el espacio de usuarios por ítems, si no que por un subconjunto del espacio con algún tipo de característica. Entonces la implementación final de u varía según la técnica utilizada por el recomendador; los RS son clasificados de acuerdo a esto.

Las clasificaciones más comunes de RS [30] son las siguientes:

Colaborativos: recomiendan en base a la similitud de preferencia de ítems entre uno o más usuarios del sistema. Estos no requieren información específica sobre el ítem a recomendar, pues sólo consideran el comportamiento pasado de los usuarios respecto al ítem considerado (o ítems similares).

Basados en contenido: buscan recomendar en base a características de los ítems y de los usuarios. Para ello, utilizan algún mecanismo de recuperación y filtrado de información, creando un ranking de los ítems que pueden entregar mayor utilidad, dada una consulta u oportunidad de sugerir/recomendar. Estos sistemas intentan obtener, de manera automática, información del comportamiento de los usuarios, e ir mejorando la calidad de las recomendaciones en la medida que se conozca más al potencial destinatario de la recomendación.

Demográficos: Estos recomiendan en base al perfil demográfico del usuario, por ende, asume que distintos nichos demográficos necesitan distintas recomendaciones.

Basados en una comunidad: Este tipo de sistema recomienda ítems basados en las preferencias de otros usuarios, relacionados con el usuario que busca la recomendación; por ejemplo, las recomendaciones de sus amistades en redes sociales.

Basados en conocimiento: Estos sistemas recomiendan en base a información detallada y estructurada de los ítems sobre los cuales se va a operar. Esta información es extraída desde una base de conocimiento, usualmente construida y mantenida por expertos en el dominio de negocio abordado. Existen dos subtipos de RS basados en conocimiento, los basados en caso y los basados en restricciones. La diferencia entre los dos es como determinan las recomendaciones, el primero las determina en base a métricas de similitud, y el segundo explota bases de conocimientos que definen reglas de cómo relacionar los requisitos del usuario con los de la recomendación.

Estos sistemas no sufren problemas al realizar arranques en frío y recomendaciones sobre nuevos ítems [8], es decir, pueden realizar recomendaciones sin datos previos sobre el comportamiento del usuario, y sin tener algún tipo de historial sobre el ítem a recomendar.

En este trabajo de tesis se decidió implementar un sistema recomendador basado en conocimiento, debido a los siguientes puntos:

1) La necesidad de tratar requisitos para dominios específicos y relacionados en base a la expertise de los usuarios requiere que, tanto los datos como las relaciones de entre estos, para el recomendador sean estructurados y curados.

2) Al utilizar tableros vacíos sin ningún elemento disponible, cualquier otro tipo de sistema recomendador tendría los problemas de arranque en frío.

3) Al aplicar nuevos cuerpos de conocimiento, cualquier otro tipo de sistema recomendador no podría recomendar ningún elemento al tablero, si es que éste no fue agregado de manera manual previamente, dado el problema de ítems nuevos.

3 Concepción de la Solución

Para conseguir el objetivo de mejorar la costo-efectividad de la dinámica de scoping, se propuso disminuir la barrera de la expertise para las dinámicas. El mecanismo para esta reducción es el reúso de conocimiento, en forma de los elementos que se incluyen en las zonas de los tableros de la dinámica, ya que son los únicos elementos variantes, y potencialmente reutilizables, dado ciertos contextos de proyectos; es decir, en proyectos que aborden un cierto tipo de sistema (por ejemplo, sistemas de información, de procesamiento de datos, etc.) para un dominio de negocio particular (salud, retail, finanzas, etc.). En consecuencia, la solución propuesta debe contener un repositorio de conocimiento reusable que encapsule los elementos bajo las dimensiones de un dominio de negocio y un tipo de tablero específico.

El repositorio de conocimiento reusable debe tener una estructura definida para ser utilizada por el recomendador (objetivo específico 1). Llamaremos cápsulas de conocimiento al conjunto de elementos reutilizables (potenciales componentes de la solución, es decir, requisitos), definidos para un cierto dominio de aplicación. Además, llamaremos biblioteca de cápsulas al repositorio de cápsulas que contiene los elementos reutilizables.

Los elementos de las cápsulas contendrán las relaciones con otros elementos dentro de la misma cápsula, y estarán asociados a una zona del tablero. Por ejemplo, una cápsula para un tablero de procesamiento de datos, sobre el dominio de las prácticas legales, podría ser utilizada para generar la solución de un dashboard de gestión para un estudio de abogados. Dicha cápsula contendría, en la zona de conceptos, elementos como, por ejemplo: *abogado* y *causas judiciales.* Estos dos elementos se podrían relacionar con la métrica *horas trabajadas en una causa judicial.*

Las cápsulas serán definidas por expertos en el dominio de aplicación (usualmente, desarrolladores con gran conocimiento del dominio de negocio vinculado a la cápsula), y guardadas en la biblioteca, que a su vez será administrada por otros usuarios. Este trabajo de tesis solo se encargará de crear el mecanismo de construcción de cápsulas, y no de construir cápsulas en particular.

Diferentes cápsulas de conocimiento componen la base de conocimiento de un tablero en específico. Es decir, las cápsulas que se están utilizando en una instancia de un tablero.

Para que las cápsulas cumplan su utilidad, los elementos de éstas serán procesadas por un sistema recomendador (objetivo específico 2), que consumirá las cápsulas que contiene un tablero, y el estado actual de esté para obtener las recomendaciones disponibles a los usuarios mientras éstos construyen el tablero. Este sistema recomendador será un sistema basado en conocimiento, funcionando en base a conceptos y relaciones estructuradas generadas por expertos, y no en filtros colaborativos sobre lo que los distintos usuarios prefieran.

Finalmente, se deben integrar las cápsulas, y la recomendación de elementos dentro del tablero digital (objetivo específico 3), a través de modificaciones a la interfaz en la aplicación legada, como, por ejemplo, selectores de cápsulas y de recomendaciones.

La Figura 9 muestra la relación conceptual entre los componentes antes enunciados. La biblioteca de cápsulas, que contiene todas las cápsulas y a su vez estas los conceptos, alimenta a la base de conocimiento, la cual representa las cápsulas seleccionadas por el usuario para ser usadas en la construcción de un tablero. El sistema recomendador extrae de la base de conocimiento los elementos de conocimiento reusable que entrega al usuario al usar la aplicación.

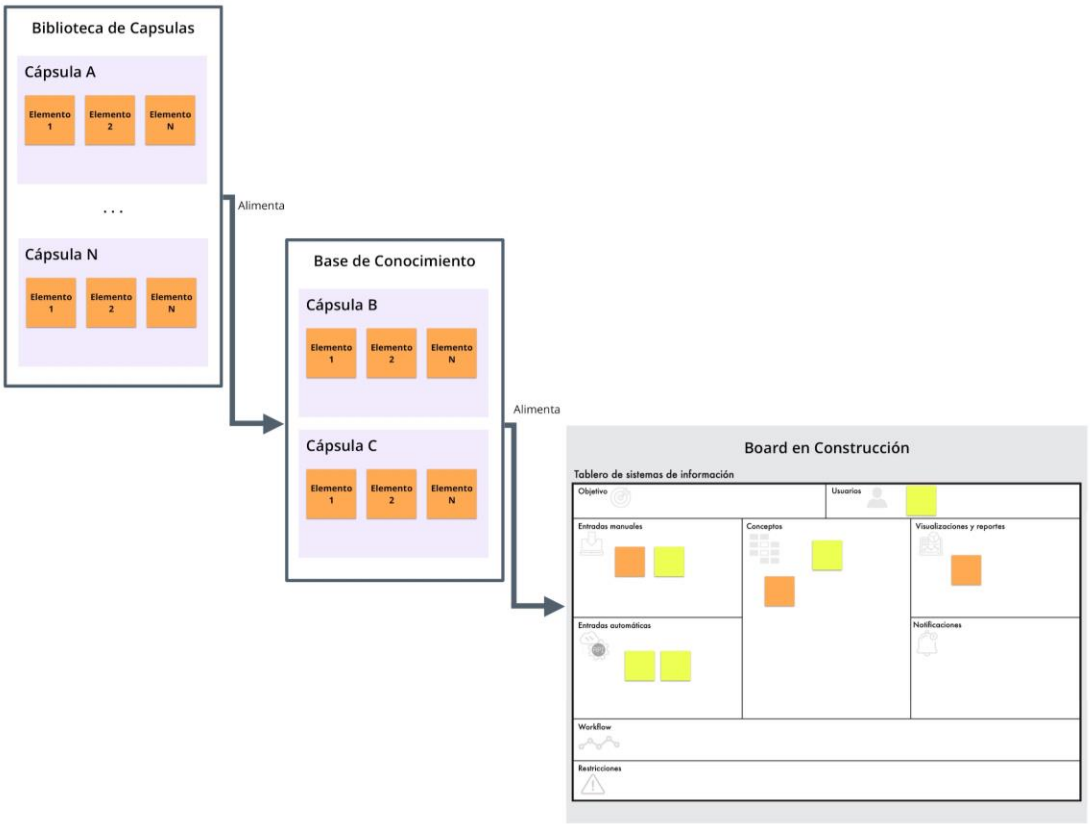

Figura 9: Diagrama general de la solución.

3.1 Requisitos y restricciones de la solución

Los requisitos y restricciones de la solución giran en torno a tres componentes principales: la biblioteca de cápsulas, el recomendador, y la integración de éstos al tablero de aplicaciones.

Requisitos y restricciones de la biblioteca de cápsulas

- Debe permitir obtener, crear y actualizar las distintas cápsulas. Estas cápsulas deben ser identificables por sí mismas.
- Debe permitir crear relaciones entre los elementos de las cápsulas.
- Debe permitir obtener, crear y actualizar otros recursos necesarios para la creación de cápsulas como: tipos de tablero y sus zonas, tipos de dominios, y palabras claves para la búsqueda de cápsulas.

Requisitos y restricciones del recomendador

El motor de recomendación debe ser capaz de:

- Realizar recomendaciones en frío (es decir, sobre un tablero vacío) y sin considerar otros tableros que hayan sido creados en el sistema (por privacidad de la información).
- Recomendar elementos potencialmente más probables (candidatos), para incluir en una zona particular del tablero.
- Recomendar los elementos candidatos para una zona del tablero, a partir de aquellos disponibles en la base de conocimiento.
- Dada la selección de un elemento en una zona del tablero, indicar otros elementos potencialmente relevantes para la misma u otra zona.

Además de estos requisitos, el motor de recomendación debe dar una cantidad controlada de recomendaciones (debido a que muchas recomendaciones no son de utilidad), y funcionar con una rapidez aceptable para el usuario.

Requisitos y restricciones de la integración en la aplicación tablero digital

El flujo dentro de la aplicación se iniciará en el momento en que el usuario seleccione el tipo de tablero a utilizar. Luego, dentro de la aplicación, el usuario debe ser capaz de:

- Revisar y cargar las cápsulas de conocimiento desde la biblioteca de cápsulas.
- Revisar la descripción de las cápsulas y elementos.

- Revisar las recomendaciones para las distintas zonas del tablero, para los elementos ya recomendados, y revisar una lista de todas las recomendaciones disponibles.

3.2 Perfiles de usuarios a soportar

Dentro de la dinámica de scoping con tableros, y en la misma aplicación, existen dos tipos de usuarios, los proveedores o equipo de desarrollo, y el cliente. Los usuarios de tipo *Proveedor* son quienes llevan la dirección de la dinámica. Estos usuarios serán los encargados de revisar y seleccionar las cápsulas adecuadas para el tipo de dominio y sistema de la sesión de scoping. Además de eso puede seleccionar los conceptos entregados para el recomendador y agregarlos a las distintas zonas del tablero.

Los usuarios tipo *Cliente* son quienes necesitan la solución de software, y pueden apoyar a la creación del tablero revisando, agregando y confirmando elementos recomendados en las distintas zonas.

3.3 Arquitectura de la solución

A continuación se describe la arquitectura del sistema a desarrollar, utilizando los viewpoints del framework C4 Model [9]. En la Figura 10 se muestran dos sistemas de software: 1) la aplicación de tablero, utilizada por el equipo proveedor para realizar la dinámica colaborativa, y 2) la biblioteca de cápsulas, que contiene las distintas cápsulas de conocimiento y entrega las recomendaciones a la aplicación de tablero.

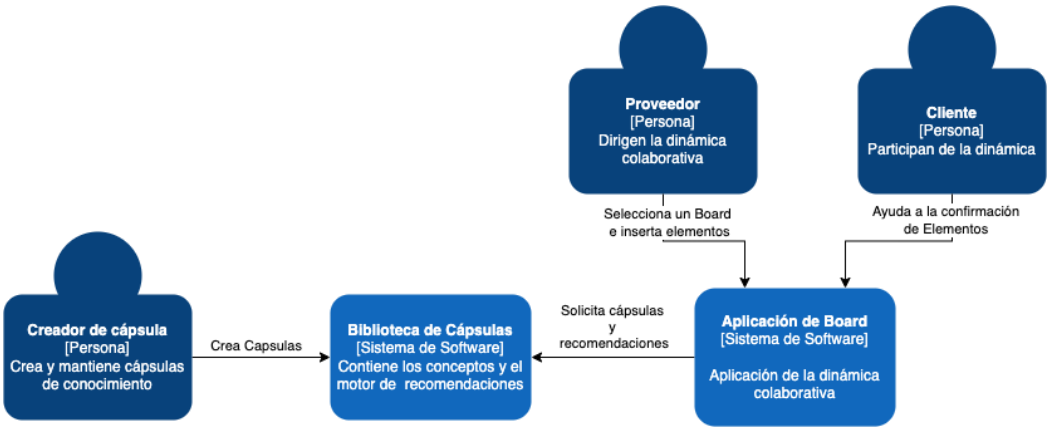

Figura 10: Diagrama de Sistemas del proyecto.

El sistema incluye una aplicación web (Fig. 11), a través de la cual interactúan los dos tipos de usuarios; es decir, proveedores (desarrolladores) y clientes (stakeholders).

La single page application (SPA) es el front-end de la aplicación y está desarrollado con el framework VueJs. En esta aplicación es donde los usuarios interactúan para ir construyendo el tablero. Aquí es donde se integrará la selección de las cápsulas y se visualizarán las recomendaciones.

Los modelos (tablero, zonas, y elementos de las zonas) y su persistencia se encuentran en la aplicación de back-end, un servicio web creado en Python utilizando el framework Django; y en la base de datos que utiliza el motor PostgreSql. Se incluye también un servicio de web socket desarrollado en Node.js, que sincroniza todas las actualizaciones del tablero en las distintas sesiones abiertas por usuarios.

Estos 3 componentes (la SPA, el servidor Django y el servidor de websockets) son parte de la aplicación legada.

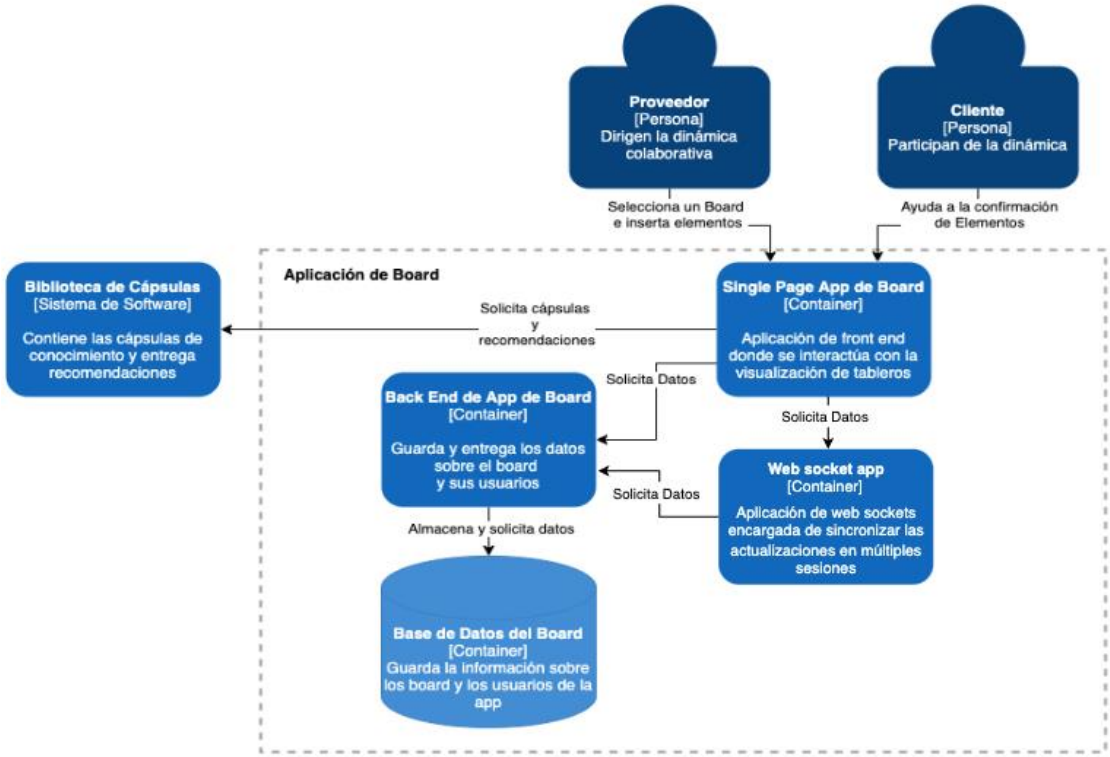

Figura 11: Diagrama de Contenedor de la Aplicación del Board.

La aplicación de biblioteca (Fig.12) contiene una aplicación web con una API REST. A través de ésta, los usuarios pueden crear y mantener cápsulas, además de buscar y entregar las recomendaciones a la aplicación del tablero.

Se tomó la decisión de crear un servicio separado de la aplicación de tablero, el cual se encarga del manejo de las cápsulas y las recomendaciones. Se hizo de esa manera debido a que: 1) los dominios de las aplicaciones son diferentes, 2) el desarrollo sería

más rápido si no estaba acoplado a la aplicación legada, 3) los cambios en una de las aplicaciones no afectarían a la otra, y 4) la biblioteca de cápsulas podría integrarse a cualquier otra aplicación de tablero por medio de su API.

El sistema recomendador se creó como un servicio interno en el servidor web, debido a que éste es una maqueta que no recibirá una gran cantidad de peticiones, y por ende, no requiere escalar independientemente de la API de la biblioteca.

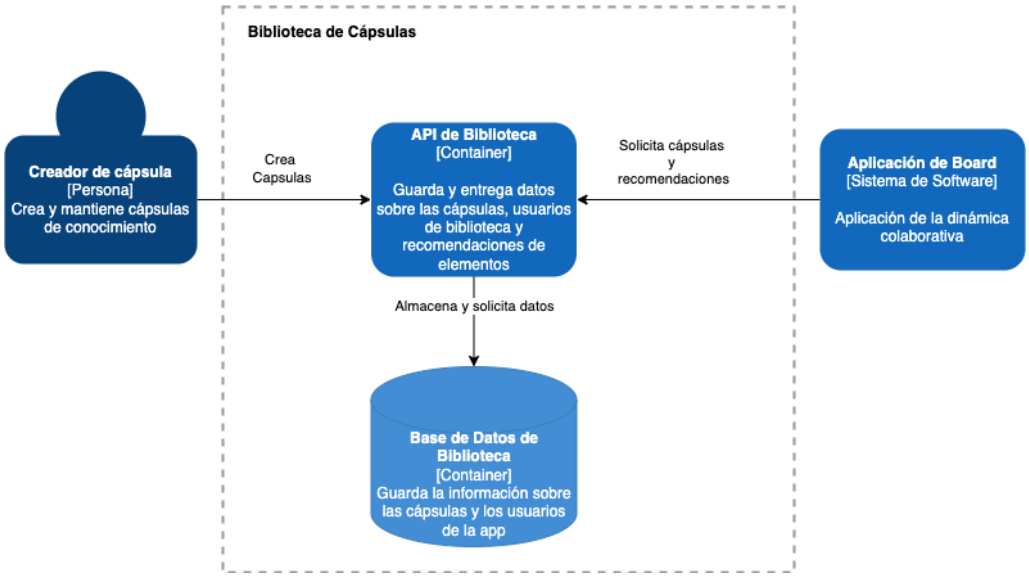

Figura 12: Diagrama de Contenedor de la Biblioteca de Cápsulas.

3.4 Modelo de datos

El modelo de datos de la biblioteca se detalla en la Figura 13. Para mejorar la legibilidad del diagrama, en dicha figura no se incluyeron algunos de los elementos de implementación, también heredados desde el framework utilizado, como, por ejemplo, los usuarios, roles, etc.

Los elementos principales del modelo son los siguientes:

Dominio: Representa un área específica del conocimiento que abarca una cápsula. Entre sí, los dominios pueden estar compuestos de otros dominios, generando así subdominios de conocimiento, por ejemplo, dentro del dominio de *Sistemas de Planificación de Recursos Empresariales (ERP, por las siglas en inglés de Enterprise Resource Planning)* se podría encontrar el subdominio de *logística* o *ventas*.

Tipo de Tablero: Representa a un tipo de tablero único con su configuración de múltiples zonas. Actualmente, el sistema tiene definidos dos tipos de tableros, uno

para *sistemas de procesamiento masivo de datos* y otro para *sistemas de información*.

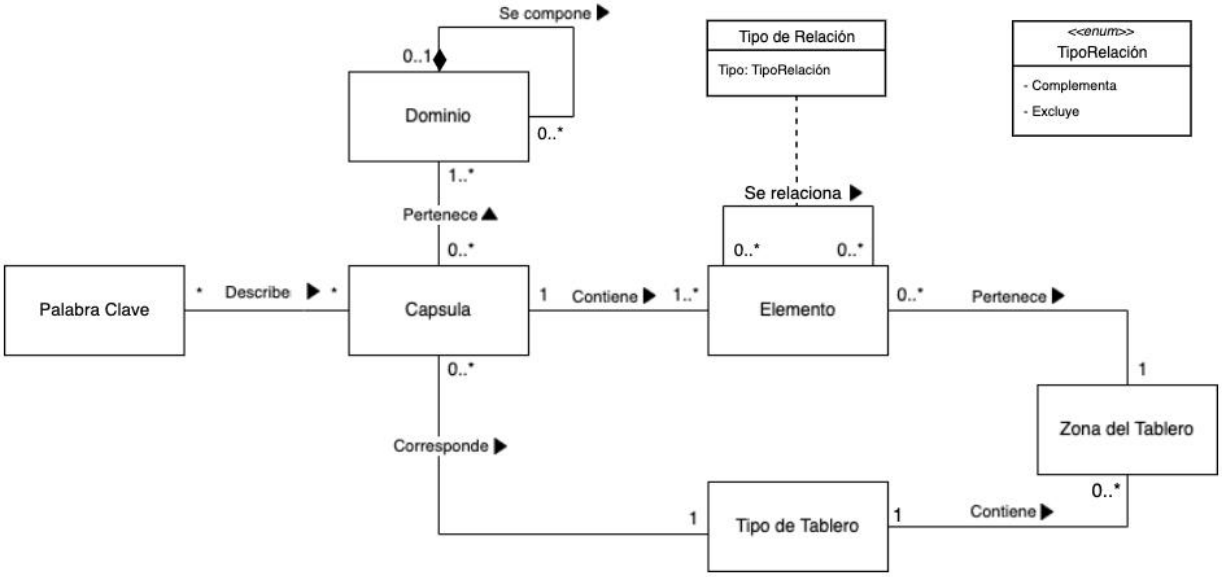

Figura 13: Modelo de Datos de la Biblioteca de Cápsulas (simplificado).

Zona del Tablero: Es una zona específica del tipo de tablero, por ej. *conceptos* en el tablero de *sistemas de procesamiento masivo de datos*.

Cápsula: Son repositorios de elementos asociados a uno o más dominios, y a un único tipo de tablero. Las cápsulas tienen un nombre único, una versión y una descripción. Por ejemplo, puede existir una cápsula de *Clínica Veterinaria de Animales Menores*, que contenga todos los elementos útiles sobre este tema.

Palabra Clave: Palabras clave utilizadas para describir una cápsula. Su principal uso es poder mejorar la búsqueda y filtrado mediante el uso de éstas.

Elemento: Representan el conocimiento reusable dentro de una cápsula, estos elementos son utilizados por el recomendador para la creación de post-its en los tableros construidos por los usuarios en la aplicación front-end. Tienen un nombre y una descripción. Los elementos pueden relacionarse con otros elementos, agregando un tipo y prioridad de relación. Además, cada elemento debe pertenecer a una zona de un tipo de tablero. La relación con otros elementos y la zona del tablero son las principales características utilizadas por el sistema recomendador.

3.5 Tecnologías seleccionadas

A continuación, se describen las principales tecnologías utilizadas, para el entorno de ambos proyectos se utilizó Docker para manejar fácilmente el despliegue de las aplicaciones tanto en desarrollo como en producción.

Biblioteca y Recomendador. La Biblioteca de Cápsulas y el Sistema Recomendador fueron desarrollados en *Python*, específicamente con el framework *Django* principalmente por la facilidad y rapidez para generar una API HTTP con la que comunicar los recursos entre esta aplicación y la del Board. Para persistir la data se utilizó el motor PostgreSQL debido a mayor expertis utilizando el motor frente a otras alternativas. Otra razón para utilizar estas dos tecnologías, fue que coinciden con las de la aplicación legada.

Aplicación del Board. La aplicación del board es un proyecto ya existente y fue modificado para poder utilizar las recomendaciones, además de algunos otros cambios. Por ende, se mantuvieron las tecnologías ya utilizadas en el proyecto sin agregar ninguna nueva. La aplicación está construida con una API HTTP en el framework Django, utilizando una base de datos PostgreSQL. Tal como se indicó antes, ésta cuenta con un servidor de websockets en NodeJS y con una Single Page Application en el framework de javascript VueJS.

4 Implementación de la solución

4.1 Biblioteca de cápsulas

La biblioteca de cápsulas fue desarrollada en Python como un servicio web REST [14]. Para esto se utilizó la librería Django y un add-on a ésta con Django Rest Framework (DRF), para implementar los endpoints. Estos endpoints consumen datos y retornan las peticiones en formato JSON.

Cada uno de los endpoints está protegido por una autenticación basada en token. Cada usuario al ser creado obtiene un token, al cual debe incluir en el encabezado de *Authorization* para realizar las peticiones.

4.1.2 Colecciones

Cada petición GET para un endpoint de una colección (por ejemplo, el endpoint de cápsulas */capsules*) implementa paginación de los recursos, retornando un conteo total de la colección, un link a la dirección de la siguiente página, un link a la dirección de la página previa (de existir), y los recursos paginados (ver ejemplo en la Figura 14).

```
{
    "count": 80,
    "next": "http://tablero-digital.app/concepts/?limit=10&offset=20",
    "previous": "http://tablero-digital.app/concepts/?limit=10",
    "results": [...]

}
```

Figura 14: Ejemplo de archivo JSON paginado en la aplicación de recomendaciones

Además, cada colección implementa filtros por cada uno de los atributos del recurso que representan; por ejemplo, si las cápsulas tienen el atributo *name* para representar su nombre. Una petición de cápsulas puede filtrarse con el parámetro *name=clientes.* para encontrar todas las cápsulas con la palabra *clientes* en su nombre. Este esquema de implementación viene dado por el framework DRF.

4.1.2 Endpoints expuestos

Los endpoints expuestos son los siguientes:

*/auth***:** Endpoint para obtener el token de autenticación de un usuario. Solo implementa en método POST y requiere que se envié como cuerpo un JSON con las

llaves *username* y *password* correspondiente al nombre de usuario y contraseña del usuario que se requiere obtener el token.

/users: Implementa tanto la colección de usuarios como recursos individuales, este endpoint solo es utilizable por usuarios que tengan la calidad de *superuser*. Cada resultado retorna el id, y nombre de usuario. Para crear un usuario se solicita los parámetros username (nombre de usuario), password (contraseña), email (correo electrónico), y groups. Este último parámetro es implementación de permisos de Django, la aplicación sólo contiene un grupo que da acceso a todos los permisos de recursos (ver, crear, actualizar y borrar), menos a usuarios.

/domains: Implementa la colección de dominios. Permite listar, crear, actualizar y borrar los dominios existentes. No se puede eliminar un dominio si éste está siendo utilizado por una cápsula. Cada resultado retorna el id y nombre del dominio. Para la creación de un dominio solo es necesario enviar el nombre, que debe ser único.

/board-types: Implementa a los distintos tipos de tableros a los que se asocian las cápsulas. Estos tableros se construyen a partir de un archivo JSON, que solicita el nombre del tipo de tablero y las distintas zonas a crear. Por ejemplo, un tablero de "sistema de procesamiento masivo de datos", al momento de crearse, incluirá zonas predeterminadas, tales como "fuentes de datos", "conceptos", "generadores de indicadores", etc.

/capsules: Implementa las cápsulas. Al crear una cápsula, el usuario debe enviar como parámetros obligatorios el nombre, descripción, versión de la cápsula, el tipo de tablero, y asociar al menos un dominio existente a través de un arreglo de dominios. Como valor opcional puede asociar distintas palabras clave a través de un arreglo de éstas. Las palabras clave representan el dominio de conocimiento de la cápsula.

/elements: Implementación del conocimiento reusable de las cápsulas. Para crear un elemento es obligatorio que éste contenga un nombre, una descripción, una referencia a una cápsula existente y a una zona de un tipo de tablero.

Como valores opcionales uno puede agregar aliases y relaciones. Los aliases son agregados a través de un arreglo con todos los nombres opcionales de un elemento, por ejemplo, el elemento *estudiante* podría tener como alias *alumno*, *escolar*, o *colegial*. Estos alias luego son seleccionables como nombres opcionales en la aplicación del tablero.

Las relaciones son agregadas a través de un arreglo con objetos que deben contener una referencia al elemento relacionado (id), la importancia de la relación (alta, media

o baja), y el tipo de relación (complementa o excluye). Las relaciones son utilizadas para generar las recomendaciones.

Además de los endpoints mencionados, se creó el endpoint */capsule-by-file* para crear cápsulas y elementos de manera más sencilla. Este endpoint permite consumir un archivo *YAML* en el que se pueden definir los parámetros para la creación de una cápsula completa con los elementos de conocimiento reutilizable que irán en cada zona. Se optó por este tipo de archivo ya que son archivos estructurados y fáciles de analizar, además de ser mucho más legibles por humanos, que por ejemplo los archivos tipo *JSON* [13].

El archivo consumido (ver ejemplo en la Figura 15) utiliza texto como principal recurso de creación, evitando así referencias a identificadores numéricos. La especificación completa del formato puede ser encontrada en el Anexo A.

Finalmente, se implementó el endpoint */recommendations* en la aplicación que permite obtener recomendaciones. Los detalles de este endpoint se describen en la siguiente sección.

```yaml
capsule: "Cuentas contables SII Chile"
description: "Cuentas contables extraídas del manual del SII Chile"
version: "1.0"
author: "Nombre de Autor"
board-type: "Datos"
domains:
  - "Contabilidad"
keywords:
  - "Cuentas Contables"
  - "SII"
elements:
  - name: "Remuneraciones por Pagar"
    description: "Sueldos devengados pendientes de pagos"
    board-zone: "Conceptos"

  - name: "Entidades Previsionales por Pagar"
    description: "En esta cuenta se incluye descuentos como: Cotización Previsional
                  Obligatoria, de Ahorro, APV, Salud Obligatoria, Salud voluntaria, Seguro
                  Cesantía."
    board-zone: "Conceptos"
    relationships:
      - concept: "Remuneraciones por Pagar"
        priority: "High"
        relationship: "Complementa"
```

Figura 15: Ejemplo de archivo YAML para creación de cápsula.

Los endpoints generados fueron considerados los mínimos viables para hacer funcional la aplicación.

4.2 El sistema de recomendación

El sistema recomendador desarrollado es del tipo “basado en conocimiento”. Como fue explicado en la Sección 2.5, este tipo de sistema es el que mejor se adecua para realizar recomendaciones en base al conocimiento de expertos.

Para obtener una recomendación se debe hacer una petición GET a la API. Esta petición debe incluir el “contexto del tablero”, el que incluye las cápsulas seleccionadas por el usuario, y los post-it creados a partir de un elemento recomendado. Como argumento extra se puede recibir el tipo de recomendación necesaria, la cual puede estar destinada a una zona del tablero, o bien basarse en una recomendación previa, que fue ya aceptada por el usuario.

Una vez recibida la petición, el sistema recomendador selecciona una estrategia para encontrar las recomendaciones. Si la recomendación solicitada está basada en un elemento, (por ej. *Remuneraciones por pagar* mostrado en la Figura 15), entonces se buscan todos los elementos relacionados que cumplan con las siguientes condiciones: 1) deben estar dentro de la cápsula del elemento, 2) no deben encontrarse ya en el tablero, y 3) no deben tener un tipo de relación excluyente (para el mismo ejemplo anterior, se regresaría el elemento *Entidades Previsionales por Pagar*, si es que no estuviera en el tablero).

Por otro lado, si la recomendación está basada en una zona del tablero, se buscan todas las recomendaciones disponibles para esa zona. Para ello, se explora los elementos dentro de todas las cápsulas agregadas, que no se encuentren en el tablero y que no tengan relaciones excluyentes con los elementos presentes.

Las recomendaciones de zona se pueden realizar en frío (es decir sobre un tablero vacío), cumpliendo así uno de los requisitos básicos de este tipo de recomendador, entregando simplemente una lista de los elementos de la cápsula para esa sección.

Si el tablero contiene otros elementos (sea o no en la misma zona seleccionada) las recomendaciones a ser entregadas son ordenadas (priorizadas) en base a la cantidad de relaciones que éstas tienen con los elementos del tablero (post-its). Por ejemplo, si el usuario del tablero solicita una recomendación, mientras realiza el scoping de su proyecto, entonces el sistema deberá determinar qué elemento le recomienda primero y cuál después. En ese caso, si un elemento “A” tiene dos relaciones activas en el tablero,

y otro elemento “B” tiene tres relaciones, entonces el elemento “B” tendrá mayor prioridad que el elemento “A” en la lista de recomendaciones.

Finalmente, si la recomendación solicitada no tiene ninguno de estos filtros, se entregan todas las recomendaciones disponibles. En este caso se utiliza el mismo método que en las filtradas por zona, pero para todas las zonas disponibles.

Una vez encontradas las recomendaciones, el endpoint responderá con un archivo JSON en el que se encuentran los resultados. En la Figura 16 se puede observar un esquema del del proceso de recomendación, para los distintos tipos de solicitudes.

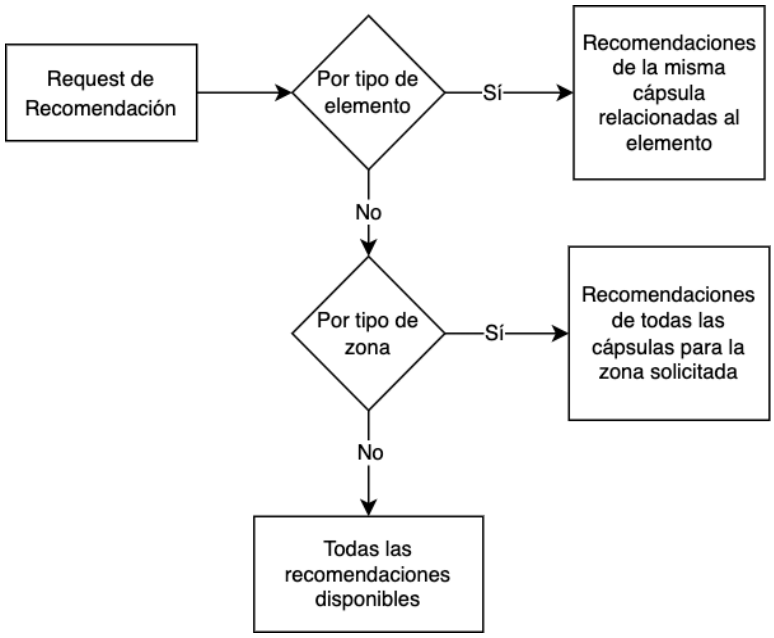

Figura 16. Proceso de las distintas respuestas de recomendaciones.

4.3 Actualización e integración con la aplicación legada

La aplicación legada del tablero digital (explicada en la Sección 2.4), fue modificada a nivel de interfaces de usuario. Aunque las pantallas de ingreso e inicio se dejaron intactas, las vistas del tablero fueron cambiadas casi completamente. El rediseño del tablero se muestra en la Figura 17. Este rediseño se hizo basado en los estilos y principios de Ant Design [3].

Además, se agregaron tres flujos importantes a la vista del tablero: 1) seleccionar cápsulas de recomendación, 2) obtener recomendaciones en el tablero, y 3) obtener recomendaciones a través del panel de recomendaciones.

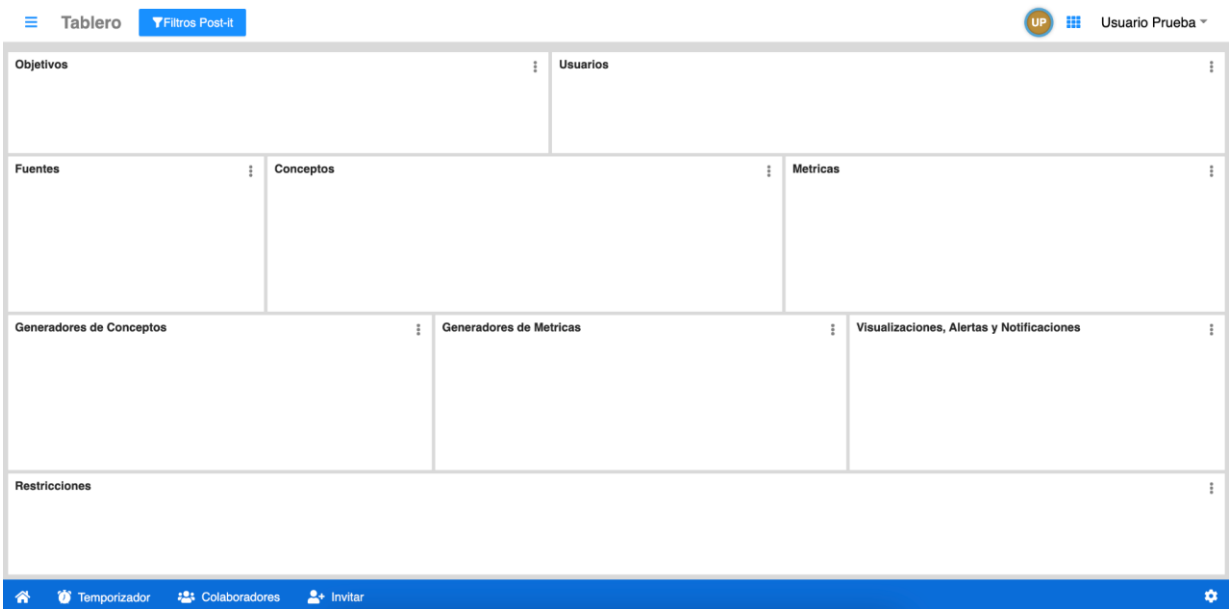

Figura 17. Vista del nuevo tablero de la aplicación.

4.3.1 Seleccionar cápsulas de recomendación

El flujo de acciones para seleccionar cápsulas se inicia con el usuario realizando un clic en el icono de menú, en la esquina superior izquierda de la barra de navegación. El menú se encuentra divido en dos secciones (Fig. 18). La primera permite la búsqueda de cápsulas, y la segunda muestra los resultados encontrados y agregados.

Figura 18. Vista del tablero de la aplicación.
(A) barra de búsqueda del panel; (B) el listado de cápsulas agregadas o disponibles.

La barra de búsqueda realiza una llamada al endpoint de cápsulas a la API del backend, buscando los términos ingresados en los títulos, descripciones, palabras clave y autores de las cápsulas. Si se encuentran resultados, entonces éstos se muestran en una lista, detallando el título, descripción, autor, rating de usuarios, y un indicador sobre si la cápsula se encuentra agregada o no (Fig. 19). Al hacer clic en agregar, la cápsula es agregada al estado del tablero, y sus recomendaciones aparecerán en el listado de recomendaciones disponibles.

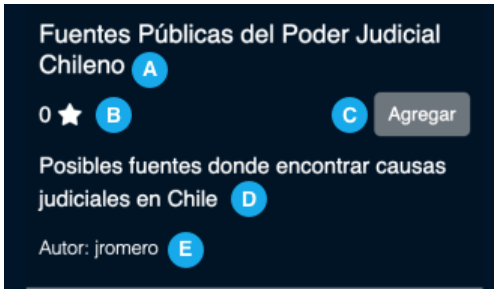

Figura 19. Cápsula listada en el tablero.
(A) título de la cápsula; (B) rating de la cápsula; (C) botón para agregar cápsula; (D) descripción de la cápsula; (E) autor de la cápsula.

Los elementos de la lista de cápsulas contienen un evento de clic, que expande el detalle de la cápsula. En el detalle se muestra (Fig. 20): en la sección superior el título, rating de usuarios, las veces que fue utilizada por otros usuarios, la versión de la cápsula y la última vez que fue actualizada. Además, se muestran los dominios, las palabras clave asociadas, la descripción y un botón para agregar o quitar la cápsula.

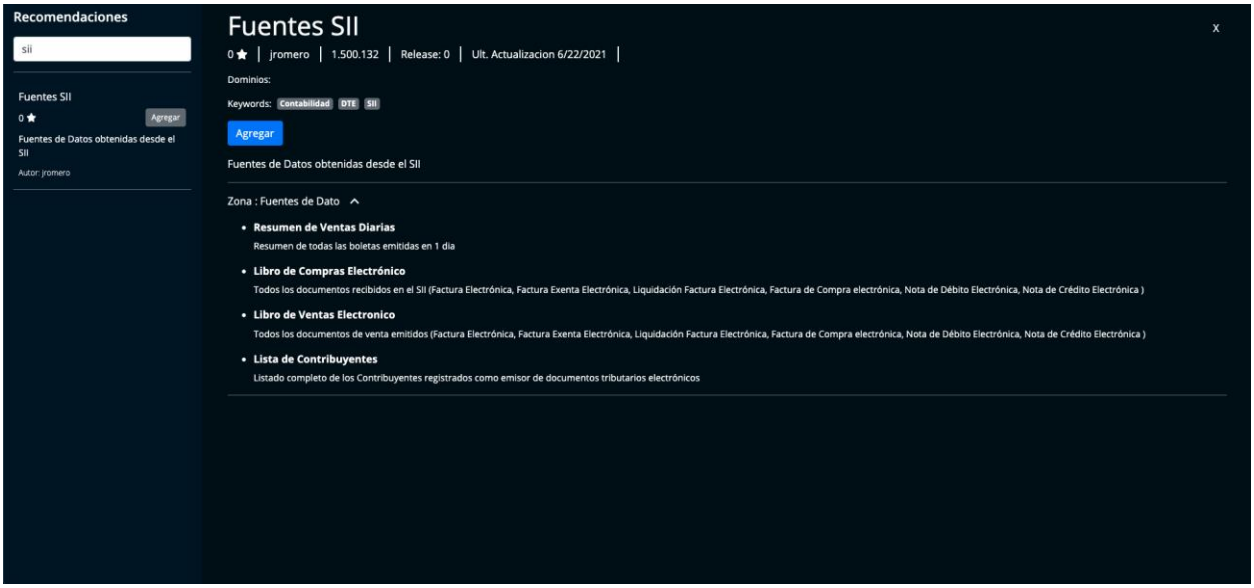

Figura 20. Detalle de una cápsula expandida.

En la sección inferior se muestran distintos menús colapsables, con cada una de las recomendaciones disponibles en la cápsula, divididas por zonas. Las recomendaciones se listan, y éstas incluyen la descripción del elemento al que pertenecen.

4.3.2 Obteniendo recomendaciones en el tablero

Una vez que el usuario ha agregado alguna cápsula, éste puede obtener recomendaciones en el tablero a través de dos mecanismos principales: haciendo clic con el botón secundario del mouse (botón derecho), o bien ingresando un nuevo post-it. Al hacer clic con el botón secundario sobre una cierta zona del tablero, el sistema despliega una lista de las recomendaciones disponibles para dicha zona (Fig. 21). Por ejemplo, cuando el usuario hace clic sobre la zona “conceptos”, obtiene múltiples elementos recomendados para agregar en dicha zona; o sea, el sistema le sugiere nuevos “conceptos a agregar”.

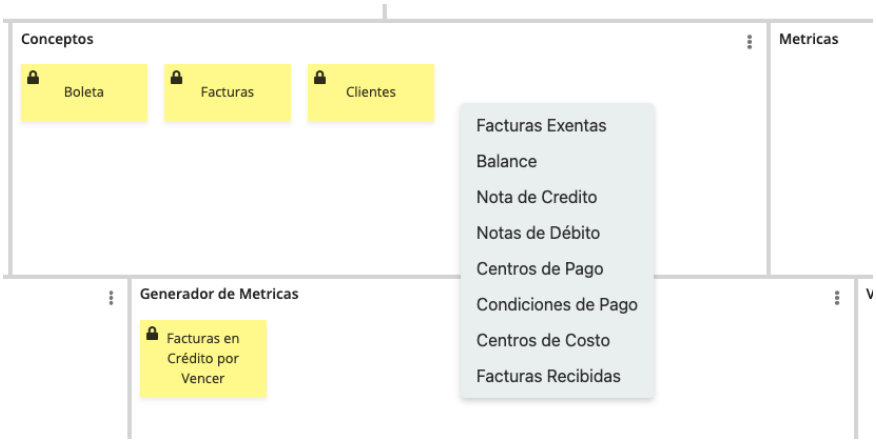

Figura 21. Obtención de recomendaciones sobre una zona del tablero.

Por otra parte, cuando se hace clic sobre una recomendación utilizando el botón secundario del mouse, se despliega una lista de las recomendaciones disponibles, relacionadas al elemento seleccionado (Fig. 22). Por ejemplo, si el usuario hace clic sobre “Facturas”, obtiene la recomendación de “Facturas Recibidas por Vencer”. Si algún contexto no tiene recomendación disponible, se muestra un ítem no seleccionable indicando que el sistema no tiene ninguna recomendación para hacer al usuario respecto al ítem seleccionado.

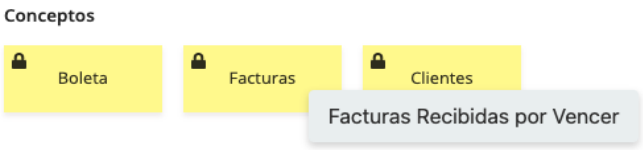

Figura 22. Obtención de una recomendación según un post-it seleccionado.

La otra manera de obtener una recomendación en el tablero, es al momento de crear un nuevo post-it. Al ingresar el título a crear, se realiza una búsqueda de texto simple para encontrar coincidencias con las recomendaciones disponibles. Por ejemplo, en la Figura 23 se muestran las sugerencias que el sistema le hace al usuario en la medida que éste va ingresando el título del elemento. Si el usuario selecciona un ítem de la lista de sugeridos, el nuevo post-it creado será contado como una recomendación. Cuando un post-it es contado como una recomendación, ésta deja de aparecer dentro de la lista de elementos recomendados. Además, este tipo de post-it es el que permite obtener recomendaciones a partir del mismo, haciendo clic en el botón secundario del mouse. Esta mejora viene dada de un proceso de calibración explicado más adelante en la sección 5.1.1.

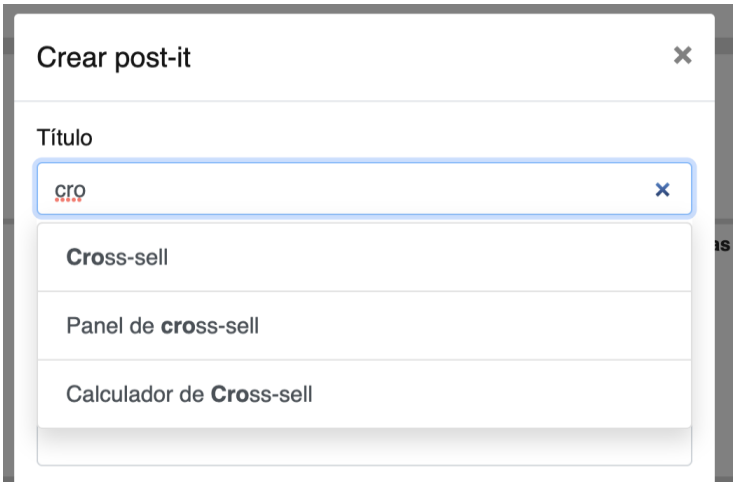

Figura 23. Obtención de una recomendación a partir del texto ingresado al crear un post-it.

4.3.3 Menú de recomendaciones

Además de las opciones anteriores para agregar una recomendación, existe un menú desplegable que lista todas las recomendaciones disponibles (Fig. 24). Para abrir dicho menú, el usuario debe hacer clic en el icono a la izquierda de su nombre de usuario en la barra superior. Una vez desplegado, se visualizan las recomendaciones en forma de post-it, divididas en secciones que representan la zona del tablero a la que corresponden. Los post-it son arrastrables y se pueden agregar en la zona correspondiente. También se puede inspeccionar la descripción de la recomendación haciendo clic en ésta.

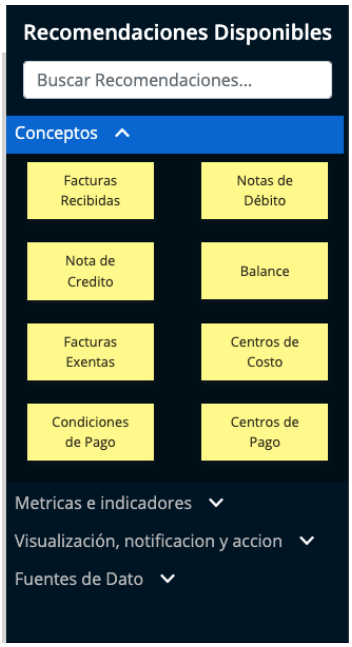

Figura 24. Panel que muestra todas las recomendaciones disponibles.

4.4. Nivel de cumplimiento de los requisitos

A continuación, se detalla brevemente el nivel de cumplimiento de los requisitos antes enunciados.

Tabla 1. Nivel de cumplimiento de requisitos

Requisitos y restricciones sobre la biblioteca de cápsulas	**Cumplimiento**	**Comentario**
Debe permitir crear relaciones entre los elementos de las cápsulas.	Completo	Modelo soporta relaciones, además de nivel de importancia (alta, media, baja) y tipos (complementa, excluye).
Debe permitir obtener, crear y actualizar otros recursos necesarios para la creación de cápsulas como: tipos de tablero y sus zonas, tipos de dominios, y palabras claves para la búsqueda de cápsulas.	Completo	Los modelos para cada uno de los elementos listados existen, en conjunto con sus endpoints REST respectivos.
Requisitos y restricciones del recomendador	**Cumplimiento**	**Comentario**
Realizar recomendaciones en frío (es decir, sobre un tablero vacío) y sin	Completo	Una vez cargada una cápsula, se puede seleccionar una

considerar otros tableros que hayan sido creados en el sistema (por privacidad de la información).		recomendación inmediatamente.
Recomendar elementos potencialmente más probables (candidatos), para incluir en una zona particular del tablero.	Completo	La API de recomendación soporta un filtro de zona.
Recomendar los elementos candidatos para una zona del tablero, a partir de aquellos disponibles en la base de conocimiento.	Completo	La API de recomendación toma en cuenta el contexto del tablero para las recomendaciones.
Dada la selección de un elemento en una zona del tablero, indicar otros elementos potencialmente relevantes para la misma u otra zona.	Completo	La API de recomendación soporta un filtro de elementos para buscar sus relacionados.
Requisitos y restricciones de la integración en la aplicación tablero digital	**Cumplimiento**	**Comentario**
Revisar y cargar las cápsulas de conocimiento desde la biblioteca de cápsulas.	Completo	Se pueden realizar todas estas acciones en la UI (Fig. 18, 19 y 20).
Revisar la descripción de las cápsulas y elementos.	Completo	Se puede realizar esta acción (Fig. 20).
Revisar las recomendaciones para las distintas zonas del tablero, para los elementos ya recomendados, y revisar una lista de todas las recomendaciones disponibles.	Completo	Se pueden realizar todas estas acciones en la UI (Fig. 21, 22 y 23, y 24)

5 Evaluación de la solución

En este capítulo se describe el proceso de evaluación realizado al sistema de recomendación embebido en el tablero, y los resultados obtenidos. El objetivo de la evaluación realizada fue determinar si la extensión desarrollada efectivamente tiene la capacidad de facilitar el proceso de scoping, comparado con un escenario sin recomendación. En ambos casos se hizo uso de la dinámica colaborativa ya definida, apoyada por el tablero digital.

La evaluación involucró sesiones de role playing, simulando una sesión de trabajo entre proveedor y cliente, cuyo fin era determinar el alcance de un proyecto de desarrollo. En estas sesiones participaron usuarios desarrolladores, representando a los proveedores de una solución de software, y usuarios con dominio en el problema presentado en la simulación, representando al cliente.

5.1 Caracterización de los participantes

Los participantes fueron miembros de compañías de desarrollo de software, que usaban el tablero digital de forma frecuente o permanente para realizar el scoping de producto. Para ello se definieron criterios de inclusión, tanto para clientes como para proveedores.

Los criterios de inclusión para los proveedores fueron los siguientes:

- Tener experiencia previa, (al menos 3) en situaciones productivas, es decir con clientes reales, en el uso de la dinámica para completar el tablero.
- Pertenecer a organizaciones que realizan la dinámica de manera activa. Se solicitó esto con el fin de contrastar las experiencias de realizar una dinámica de scoping, sin y con el mecanismo de recomendación, con usuarios representativos en el universo de uso de la dinámica con apoyo de tableros.

El criterio de inclusión para los clientes fue tener el dominio del caso de negocio abordado; o sea, el ámbito de negocio para el cual se iba a desarrollar el software.

Para conseguir a los participantes desarrolladores se contactaron directamente por correo electrónico a 11 personas que cumplían con el perfil señalado, y pertenecían a organizaciones que utilizan el canvas activamente dentro de sus procesos, ya sea para scoping con clientes o scoping interno de desarrollo de soluciones. De estas, 3 personas accedieron a participar en la dinámica.

Para conseguir a los participantes clientes se hizo una invitación a 44 personas relacionadas con el área de ventas, marketing y relaciones con clientes, coordinando a 3 personas para las fechas en que se realizó la simulación.

Finalmente, se llevaron a cabo 3 sesiones, cada una con un cliente y un desarrollador distinto. La interacción entre los participantes se realizó a través de videoconferencia, y usaron el tablero digital extendido. Las sesiones fueron grabadas en video, con la debida autorización de los participantes, y el autor de este trabajo de tesis actuó como registrador del evento.

Antes de llevar a cabo las sesiones de evaluación a todas ellas se les envió un video explicativo de la nueva herramienta (disponible en https://youtu.be/o9qPa52FpEU). También se les envió un link para acceder a la aplicación, con las respectivas credenciales de ingreso, para que pudieran probarla. A cada uno se le realizó una breve inducción a la nueva versión del tablero, para que supieran cómo seleccionar cápsulas y encontrar recomendaciones.

5.2 Evaluación realizada

El problema de negocio abordado en la evaluación (descrito en el Anexo B) estuvo orientado a la definición de una solución tipo dashboard, usando datos relacionados al dominio de negocio de la gestión de los recursos de clientes. Para hacer esta evaluación, se preparó una cápsula específica que incluyó conceptos generales sobre gestión de clientes, ventas y otros relacionados. Dicha cápsula se utilizó en conjunto al caso de role playing, y al tablero digital mejorado. También se agregaron otras cápsulas no relacionadas al caso en cuestión, para así simular una biblioteca de cápsulas que permitiera ser navegable.

Antes de dar inicio a cada sesión, se le entregó al desarrollador y al cliente una descripción de alto nivel del caso de negocio a abordar, el cual se trataba de la extensión a los servicios de un CRM para clínicas veterinarias. Para ello, el cliente contactó al proveedor para indicarle la necesidad de agregar un dashboard a su CRM, para que le permitiera monitorear aspectos clave del negocio. Las necesidades más importantes de la empresa hoy en día tienen que ver con mantener el churn[1] controlado, y los cumplimientos de las metas de venta.

Al iniciar la sesión de scoping, se le indicó al proveedor que él era el encargado de explicar la dinámica al cliente, y de ir agregando los post-its en el tablero (Fig. 25), con

[1] El churn es un indicador que muestra el porcentaje de suscripciones canceladas o no renovadas para un cierto servicio.

la ayuda de las recomendaciones que pudiera obtener (visibles para ambos usuarios). Además, se explicó que la sesión de scoping tendría dos posibles condiciones de término, según lo que ocurriera primero: 1) el alcance de la solución especificada era suficiente para resolver el problema planteado por el cliente, o 2) se alcanzó el tiempo máximo de la sesión, que en este caso se definió en 30 minutos a partir de la colocación del primer post-it en el tablero.

Figura 25: Extracto de un tablero llenado por un usuario desarrollador, apoyado por el sistema de recomendación.

Al inicio de la sesión el proveedor contaba con un tiempo máximo de 5 minutos para preparar un tablero, y seleccionar las cápsulas de conocimiento que considerara pertinentes según el dominio abordado. Además, se le pidió que compartiera su pantalla para realizar la dinámica con el cliente. Una vez realizado esto, se invitó al cliente a ingresar a la reunión. Todas las sesiones se llevaron a cabo utilizando la nueva versión del tablero digital y Google Meet como soporte a las videoconferencias.

Una vez concluida la sesión de trabajo, los participantes completaron la encuesta que se presenta en la siguiente sección.

5.3. Instrumento de evaluación utilizado

El instrumento de evaluación utilizado en las sesiones de role playing fue constaba de dos secciones, la primera iba enfocada a la evaluación del recomendador, y solo fue contestada por los usuarios desarrolladores, quienes utilizaron el recomendador en la

dinámica, la segunda parte estaba enfocada a la percepción del usuario respecto a la mejora que entregaba el recomendador a la dinámica, y fue contestada por ambas partes.

La primera sección de preguntas estuvo basada en *ResQue* [28]. Este es un framework de evaluación centrado en el usuario, diseñado para sistemas recomendadores. Dicho framework, basado en TAM (Technology Acceptance Model) [11], se compone de un cuestionario con 15 criterios distintos a evaluar.

La decisión de utilizar este framework para la evaluación se tomó considerando que la mayoría de los otros instrumentos de evaluación existentes para recomendadores se enfocan directamente en la calidad y exactitud de la recomendación [1, 18]. Sin embargo, ResQue está orientado a la usabilidad de recomendador, dado a que en este trabajo se buscó evaluar el impacto del recomendador como instrumento de apoyo a la dinámica de scoping, y no de la calidad de las recomendaciones de los elementos de una cápsula.

De los 15 criterios se incluyeron 9, particularmente, se dejaron fuera todos los constructos que hicieran referencia a esto último (calidad de las recomendaciones). Además, se modificaron ligeramente algunas preguntas, para incluir la terminología que fue parte de los instrumentos y de la dinámica utilizados, por ejemplo, palabras como “elemento” y “tablero”.

Los constructos analizados en el cuestionario fueron las siguientes:

- *Novedad en las recomendaciones*: Evalúa si las recomendaciones entregadas por el sistema eran elementos que el cliente y el proveedor no habían identificado, de no ser por la recomendación.
- *Suficiencia de la interfaz*: Evalúa el diseño de cómo son presentadas las recomendaciones; por ejemplo, etiquetas, disposición en el espacio del tablero, etc.
- *Explicabilidad*: Evalúa la habilidad que tiene el sistema para explicarle al usuario los resultados de una recomendación, y por qué se le está recomendando un ítem en particular.
- *Suficiencia de la información*: Evalúa si la información contenida en las recomendaciones es suficiente para que el usuario pueda tomar una decisión (aceptar o no la recomendación).

- *Facilidad de uso percibida*: Evalúa qué tan fácil es usar la aplicación. En el ámbito de la recomendación evalúa qué tan fácil es elegir un nuevo ítem para el tablero a partir de las sugerencias entregadas por el recomendador.
- *Transparencia*: Evalúa qué tanto el sistema le permite entender al usuario por qué le está recomendando ciertos ítems.
- *Utilidad percibida*: Determina qué tanto mejora o empeora la dinámica de completar el tablero, comparando cuando se realiza con el sistema legado y con el nuevo sistema (con recomendaciones).
- *Confianza*: Evalúa el nivel de confiabilidad que tiene el sistema, según la percepción de los usuarios respecto a las recomendaciones.
- *Intención de uso*: Evalúa si el usuario usaría el recomendador nuevamente, o de manera frecuente.

Cada ítem del cuestionario se evaluó utilizando una escala de Likert de 1 a 5, donde 1 indica “muy en desacuerdo” y 5 “muy de acuerdo”.

La segunda sección de 3 preguntas, las cuales estaban relacionadas netamente con la facilidad para realizar la dinámica, evaluaron si el usuario percibe que al usar el recomendador mejora alguno de los siguientes aspectos: fluidez, completitud, y eficacia en la colocación de post-its.

La encuesta aplicada se encuentra en el Anexo C.

5.4 Resultados obtenidos

Los resultados para cada constructo de la entrevista son los siguientes:

- *Novedad en las recomendaciones*. Los resultados fueron unánimes; los participantes evaluaron la ayuda del recomendador en encontrar elementos novedosos con un “muy de acuerdo”.
- *Suficiencia de la interfaz.* Sobre la claridad e idoneidad de los elementos recomendados en la interfaz, los entrevistados respondieron en forma positiva a estas características. Por otra parte, la interfaz del recomendador en general, recibió reacciones neutras a positivas, con 12 puntos de 15 posibles.
- *Explicabilidad.* La capacidad de explicar y justificar las recomendaciones al usuario recibió respuestas mixtas, con resultados en “desacuerdo” y “de acuerdo”, con 9 puntos de 15 posibles.

- *Suficiencia de información.* La suficiencia de información entregada por el recomendador para agregar elementos fue evaluada de manera positiva con 14 puntos de 15 posibles.
- *Facilidad de uso percibida.* La facilidad de uso del recomendador recibió respuestas de neutras a positivas con 12 puntos de 15 posibles. Por otro lado, la facilidad para encontrar los elementos recomendados fue evaluada mayoritariamente como neutra, con 11 puntos de 15 posibles.
- *Transparencia.* El entendimiento por parte del entrevistado del por qué se le recomendaban los elementos del tablero, fue evaluado de manera unánime como positivo ("de acuerdo").
- *Utilidad percibida.* La utilidad percibida por el recomendador al momento de entregar elementos deseados fue evaluada de manera positiva. Los resultados se encontraron en el rango de "de acuerdo" y "muy de acuerdo", con 13 de 15 puntos posibles.
- *Confianza.* La confianza entregada por el recomendador al momento de completar el tablero fue positiva, con resultados en el rango entre "de acuerdo" y "muy de acuerdo", obteniendo 13 de 15 puntos posibles.
- *Intención de uso.* Todos los entrevistados estuvieron "muy de acuerdo" en que utilizarían el recomendador nuevamente, y de manera frecuente.

El resultado para la mayoría de las dimensiones evaluadas fue unánimemente positivo, excepto las dimensiones de *explicabilidad* y *facilidad de uso percibida* donde hubo discrepancia, pero igual la evaluación promedio fue positiva o neutra.

Respecto a las preguntas que evalúan el efecto que generan las recomendaciones en la dinámica de scoping, los usuarios desarrolladores fueron unánimes, indicando una mejora percibida respecto a fluidez, mayor completitud y menores tiempos para realizar la labor. Los usuarios clientes si bien no fueron unánimes, tuvieron puntajes positivos, sumando 14 puntos en la evaluación de fluidez y completitud, y 13 en la mejora de tiempo. Por lo tanto, a juicio de los evaluadores, la extensión realizada estaría alcanzando el objetivo definido para ella. En el Anexo D se muestra el detalle de los resultados de la evaluación.

5.5 Discusión

El análisis de los resultados fue dividido en dos partes: la evaluación de usabilidad del recomendador, y el efecto de éste en la dinámica de construcción del tablero. Como se mencionó antes, el resultado para la mayoría de las dimensiones de ResQue fue positivo,

consistente y congruente con lo esperado, demostrando que la experiencia de uso del recomendador fue adecuada para realizar la tarea asignada. Solo la dimensión de *explicabilidad* y *facilidad de uso percibida* tuvieron resultados negativos en alguna de las respuestas. Esto muestra que existen puntos de mejora a nivel de experiencia de usuario (UX), como visualizar de manera más clara la relación entre elementos recomendados, o poder agregar de manera más fácil las recomendaciones del menú de recomendaciones. Una vez resueltos estos temas de usabilidad, la aplicación debería volverse más explicable y amigable.

Como se plantea en [36], la dinámica de tableros hace al scoping más costo-efectivo respecto a alternativas más tradicionales usadas en la etapa de pre-venta. La mejora percibida por los evaluadores, debido al uso del recomendador, fue medida con las 3 últimas preguntas de la encuesta. Los resultados fueron consistentes y positivos, indicando que la extensión desarrollada mejora la fluidez, completitud y tiempo requerido para realizar el scoping del producto. Esto refleja el potencial del recomendador para cumplir con el objetivo planteado; es decir, facilitar la actividad de scoping, en comparación con el uso del tablero digital sin recomendaciones.

Considerando los aspectos descritos en la sección 2.3, el recomendador facilita la dinámica de scoping a través de las mejoras en tiempo, y la rebaja de la expertise necesaria para realizar la tarea. Además, se impacta positivamente la confianza entre proveedor y cliente, debido a que se mejora en fluidez del proceso de captación y validación de requisitos (post-its).

5.6 Limitaciones de la evaluación

En base al experimento realizado, también se deben considerar las siguientes limitaciones:

1. *Tamaño de la muestra.* El tamaño de la muestra de evaluadores fue pequeño, debido a la necesidad de utilizar desarrolladores que hayan tenido experiencia previa con la dinámica de scoping usando el tablero, y clientes reales que conocieran el dominio de negocio abordado. Sin estas exigencias para los participantes, no habría sido posible evaluar el valor agregado por el sistema de recomendación.

2. *Evaluación supeditada a la funcionalidad ofrecida por una aplicación legada.* El hecho de haber ocupado una aplicación legada como base para la nueva dinámica del tablero, no permite identificar si la experiencia de usuario respecto al uso del sistema (agregar, editar, eliminar, y arrastrar post-its) pudo afectar los resultados relacionados con la experiencia de la recomendación.

Claramente, se requieren más evaluaciones para poder sacar conclusiones más firmes y transversales respecto a las ventajas que aporta el servicio de recomendación desarrollado.

6 Conclusiones y trabajo a futuro

En vista del aumento de demanda y competitividad en la industria del software, las empresas que realizan desarrollos a medida deben ser altamente costo-efectivas a la hora de iniciar un proyecto. Con este motivo se utilizan una serie de herramientas a la hora de capturar requisitos de los potenciales proyectos en la etapa de pre-venta. Una de estas herramientas es la dinámica de tableros para realizar el scoping de productos durante la etapa de pre-proyecto y de proyecto. Una definición precisa y completa de los requisitos en esta etapa permite obtener mejores propuestas, que incluyen estimaciones más cercanas a la realidad. Además, la co-construcción del alcance del producto permite llegar a acuerdos más fácilmente entre el cliente y el desarrollador.

En este trabajo de tesis se buscó facilitar aún más el proceso de scoping, agregándole un servicio de recomendación de post-its al tablero digital existente. El servicio de recomendación utiliza un repositorio de elementos (cápsulas de conocimiento) para recomendar posibles elementos a incluir en el tablero. Estos elementos (post-its) representan funcionalidad específica que podría incluirse o no en el producto cuyo alcance se está definiendo.

La implementación del servicio de recomendación utilizó como base una aplicación web legada (tablero digital), la cual apoyaba la dinámica de scoping sin utilizar recomendaciones. Dicha aplicación fue extendida (inclusive su interfaz de usuario) para dar cabida al servicio de recomendación de post-its. El desarrollo de esta aplicación fue iterativo. En dicho proceso se realizaron revisiones periódicas con desarrolladores (potenciales usuarios) y diseñadores gráficos. Esto permitió que la versión de la herramienta sometida a evaluación ya acogiera la retroalimentación de un buen número de usuarios reales.

Este servicio fue evaluado a través de experimentos, donde participaron desarrolladores reales y también clientes reales; sin embargo, el proyecto abordado era simulado. Estos participantes no tuvieron participación alguna durante el proceso de desarrollo del servicio de recomendación, pero sí conocían y usaban frecuentemente la versión legada del tablero digital. Al término de la evaluación, se les aplicó una encuesta a los desarrolladores y clientes, donde ellos indicaron su nivel de satisfacción con el proceso de scoping realizado.

La experiencia de usar el recomendador a la hora de realizar la captura de requisitos fue positiva, solo encontrándose algunas limitaciones en el diseño de la UI de la aplicación legada. Los evaluadores percibieron una mejora en tiempos, completitud y fluidez de la

dinámica, con respecto a la misma actividad cuando ésta es realizada sin el recomendador. Por lo tanto, se cumple así con el objetivo de este trabajo.

Tanto en el proceso de desarrollo de la herramienta, como en su evaluación, se pudo constatar que la usabilidad del sistema es un aspecto muy importante para todos los involucrados; probablemente igual de importante que la utilidad del servicio de recomendación. Estos resultados son preliminares, y se debe continuar evaluando el servicio de recomendación a una escala mayor para sacar conclusiones más firmes.

Como parte del trabajo a futuro, estimamos que se debería incluir un mayor análisis de la usabilidad y utilidad percibida para el recomendador y el tablero digital; especialmente pensando en usuarios que nunca se han enfrentado al proceso de scoping utilizando estas herramientas. Además, se requiere facilitar el proceso de creación de cápsulas de conocimiento, para que éstas puedan ser desarrolladas por casi cualquier persona que tenga conocimiento acabado de un cierto dominio de aplicación.

Bibliografía

[1] Adomavicius, G., & Tuzhilin, A. (2005). Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions. IEEE Transactions on Knowledge and Data Engineering, 17(6), 734–749. doi:10.1109/tkde.2005.99

[2] Agarwal, N., & Rathod, U. (2006). Defining 'success' for software projects: An exploratory revelation. International Journal of Project Management, 24(4), 358-370. doi:10.1016/j.ijproman.2005.11.009

[3] Ant Design. https://ant.design/docs/spec/introduce. Última Visita: Abril de 2023.

[4] Aranda, A.M., Dieste, O., & Juristo, N. (2016). Effect of domain knowledge on elicitation effectiveness: An internally replicated controlled experiment. IEEE Transactions on Software Engineering, 42(5), 427-451. doi:10.1109/tse.2015.2494588

[5] Arduin, P.A. (2015). On the measurement of cooperative compatibility to predict meaning variance. CSCWD 2015: 42-47

[6] Aurum, A., & Wohlin, C. (2005). Requirements Elicitation: A Survey of Techniques, Approaches, and Tools. In Engineering and managing software requirements (pp. 19-46). Berlin: Springer.

[7] Bano, M., Zowghi, D., & da Rimini, F. (2018). User Involvement in Software Development: The Good, the Bad, and the Ugly. IEEE Software, 35(6), 8–11. doi:10.1109/ms.2018.4321252

[8] Bouraga, S., Jureta, I., Faulkner, S., & Herssens, C. (2014). Knowledge-based recommendation systems. International Journal of Intelligent Information Technologies, 10(2), 1-19. doi:10.4018/ijiit.2014040101

[9] C4 Model. (2022). URL: https://c4model.com/. Simon Brown. Última visita: Junio de 2023.

[10] Chou, D.C., & Chou, A.Y. (2009). Information systems outsourcing life cycle and risks analysis. Computer Standards & Interfaces (Vol. 31, Issue 5, pp. 1036–1043). Elsevier BV. https://doi.org/10.1016/j.csi.2008.09.032

[11] Davis, F. (1989) Perceived Usefulness, Perceived Ease of Use, and User Acceptance of Information Technology. MIS Quarterly, 13, 319-340. https://doi.org/10.2307/249008

[12] Duarte, V., Barreto, F. (2022). Requirements debt: causes, consequences, and mitigating practices. Proc. of 33th International Conference on Software Engineering and Knowledge Engineering (SEKE'22). pp. 13-18. KSI Research Inc., ISBN 1-891706-54-3.

[13] Eriksson, M., & Hallberg, V. (2011). Comparison between JSON and YAML for data serialization. The School of Computer Science and Engineering Royal Institute of Technology, 1-25.

[14] Fielding, R. T. (2000). Architectural styles and the design of network-based software architectures. University of California, Irvine.

[15] Giardino, C., Unterkalmsteiner, M., Paternoster, N., Gorschek, T., & Abrahamsson, P. (2014). What Do We Know about Software Development in Startups? IEEE Software, 31(5), 28–32. doi:10.1109/ms.2014.129

[16] Gomez-Uribe, C.A., & Hunt, N. (2016). The Netflix recommender system. ACM Transactions on Management Information Systems, 6(4), 1-19. doi:10.1145/2843948.

[17] Hadar, I., Soffer, P., & Kenzi, K. (2012). The role of domain knowledge in requirements elicitation via interviews: An exploratory study. Requirements Engineering, 19(2), 143-159. doi:10.1007/s00766-012-0163-2.

[18] Herlocker, J. L., Konstan, J. A., Terveen, L. G., & Riedl, J. T. (2004). Evaluating collaborative filtering recommender systems. ACM Transactions on Information Systems, 22(1), 5–53. doi:10.1145/963770.963772

[19] ISO/IEC/IEEE International Standard (2017). Systems and software engineering -- Software life cycle processes. (s.f.). IEEE. https://doi.org/10.1109/ieeestd.2017.8100771

[20] Jacobson, K., Murali, V., Newett, E., Whitman, B., & Yon, R. (2016). Music personalization at spotify. Proceedings of the 10th ACM Conference on Recommender Systems. doi:10.1145/2959100.2959120.

[21] Jannach, D. (2012). Recommender systems: An introduction. Cambridge, New York: Cambridge University Press.

[22] Mobasher, B., & Cleland-Huang, J. (2011). Recommender systems in requirements engineering. AI Magazine, 32(3), 81-89. doi:10.1609/aimag.v32i3.2366

[23] Oemig, C. (2014). Pre-Sales Requirements Engineering based on Miller Heiman's Sales Approach. Proc. of the International Conference on Requirements Engineering: Foundation for Software Quality (REFSQ), vol 1138. 2014.

[24] Oemig, C. (2015). The Box Fight Analogy: A Blueprint for Pre-Sales Requirements Engineering Workshops.

[25] Oemig, C. (2015). When Analysts turn into Boxers: An Introduction to Pre-Sales Requirements Engineering. En Complex Systems Informatics and Modeling Quarterly (Issue 3, pp. 1–14). Riga Technical University. https://doi.org/10.7250/csimq.2015-3.01.

[26] Pradeep, S., Pramanik, D., Dey, A., & Choudhury, P. (2021). Recommender Systems: An Overview, Research Trends, and Future Directions. International Journal of Business and Systems Research 15. 14–52. 10.1504/IJBSR.2021.10033303.

[27] Project Management Institute (2012). A Guide to the Project Management Body of Knowledge (PMBOK Guide). Seventh Edition.Project Management Institute.

[28] Pu, P., Chen, L., & Hu, R. (2011). A user-centric evaluation framework for recommender systems. Proceedings of the Fifth ACM Conference on Recommender Systems - RecSys '11. doi:10.1145/2043932.2043962.

[29] Purna Sudhakar, G. (2012). A model of critical success factors for software projects. Journal of Enterprise Information Management, 25(6), 537–558. doi:10.1108/17410391211272829.

[30] Ricci, F., Rokach, L., & Shapira, B. (2016). Introduction to Recommender Systems Handbook. In Recommender Systems Handbook Springer-Verlag. doi:10.1007/978-0-387-85820-3.

[31] Savolainen, P., Ahonen, J. J., & Richardson, I. (2015). When did your project start? – The software supplier's perspective. Journal of Systems and Software (Vol. 104, pp. 32–40). Elsevier BV. https://doi.org/10.1016/j.jss.2015.02.041.

[32] Sharp, H. (1991). The role of domain knowledge in software design. Behaviour & Information Technology, 10(5), 383–401. doi:10.1080/01449299108924298

[33] Smith, B., & Linden, G. (2017). Two decades of recommender systems at Amazon.com. IEEE Internet Computing, 21(3), 12-18. doi:10.1109/mic.2017.72.

[34] Thomas, G., & Fernández, W. (2008). Success in IT projects: A matter of definition? International Journal of Project Management, 26(7), 733-742. doi:10.1016/j.ijproman.2008.06.003.

[35] Vera, T., Ochoa, S.F., & Perovich, D. (2021). Requirements engineering in the pre-contract stage: Exploring the processes and practices used in small and medium-sized software enterprises. Proceedings of the 36th ACM/SIGAPP Symposium on

Applied Computing (SAC'21), March 2021, pp. 1346–1353 https://doi.org/10.1145/3412841.3442009.

[36] Vera, T., Perovich, D., & Ochoa, S.F. (2021). An Instrument to Define the Product Scope at Pre-selling Time. Proceedings of the 24th IEEE International Conference on Computer Supported Cooperative Work in Design (IEEE CSCWD'21), pp. 604-608, doi: 10.1109/CSCWD49262.2021.9437717.

Anexos

Anexo A - Especificación del formato YAML para la creación de cápsulas

A continuación se muestra el esquema del archivo en formato YAML que es utilizado para la creación de cápsulas de recomendaciones.

```
capsule:
  - tipo: String
  - descripción: Nombre de la cápsula
description:
  - tipo: String
  - descripción: Descripción de la cápsula
version:
  - tipo: String
  - descripción: El versionamiento de la cápsula
author:
  - tipo: String
  - descripción: El autor de la cápsula. Debe ser un usuario existente
board-type:
  - tipo: String
  - descripción: El tipo de tablero en el que puede ser utilizada la cápsula. Debe
    hacer match con uno de los tipos de tableros existentes
domains:
  - tipo: Array
  - descripción: Lista de todos los dominios que corresponden a la cápsula
  - items:
    - tipo: String
    - descripción: El nombre de un dominio en particular, por ejemplo “Ventas”
keywords:
  - tipo: Array
  - descripción: Lista de palabras claves que describen a la cápsula
  - items:
    - tipo: String
    - descripción: El nombre de la palabra clave a utilizar
elements:
  - tipo: Array
  - descripción:
  - items:
    - tipo: Objeto
    - descripción: Objeto que identifica un elemento de la cápsula
    - propiedades:
      - nombre:
```

  - tipo: String
  - descripción: Nombre del elemento
- descripción:
  - tipo: String
  - descripción: Descripción del elemento
- board-zone:
  - tipo: String
  - descripción: Zona del tablero a la que pertenece el elemento. Debe ser una de las zonas del tablero declaradas para el tipo de tablero definido en este mismo archivo
- relationships:
  - tipo: Array
  - descripción: Todas las relaciones de un elemento dado
  - items:
    - Tipo: Objeto
    - descripción: Objeto que representa una relación de un concepto
    - propiedades:
      - concept:
        - tipo: String
        - descripción: Nombre del concepto con el que se está relacionado. Debe ser un concepto existente en el mismo archivo.
      - priority:
        - tipo: String
        - descripción: Prioridad de la relación. Acepta un string con el valor “High”, “Medium”, “Low”
      - relationship:
        - tipo: String
        - descripción: Tipo de la relación. Acepta un string con el valor “Complementa” o “Excluye”

Anexo B - Caso utilizado en roleplaying

El caso involucró la entrega de información distinta, en función de si el participante estaba jugando el rol de desarrollador o de cliente. A continuación se explica cada uno.

B.1 Cliente

Asuma que usted es el gerente de ventas de PetSoft, una empresa de venta de software como servicio para veterinarias. Actualmente su empresa está utilizando un sistema de facturación, más un CRM a medida que representa un sistema. La empresa utiliza estos productos de software desde hace unos 7 años aproximadamente.

PetSoft posee información en detalle de cada una de las ventas facturadas, sus clientes, de los leads de venta en curso, y los fallidos. Toda esta información está segmentada por vendedores.

En este momento la empresa se encuentra limitada respecto a la información que puede rescatar de sus datos, debido a que son muchos para poder procesarlos en las planillas de cálculo que regularmente utiliza el área de ventas. Es por esto que PetSoft se contactó con una empresa desarrolladora, para intentar iniciar un proyecto de datos que le permita implementar un dashboard de ventas. Las necesidades más importantes de la empresa hoy en día tienen que ver con mantener el churn2 controlado, y los cumplimientos de las metas de venta.

B.2 Developer

Asuma que usted está a punto de entrar a una entrevista con el gerente de ventas de PetSoft, y que usted se dedica a venta de software como servicio para veterinarias.

PetSoft está utilizando un ERP/CRM a medida (sistemas legado), que utilizan desde hace unos 7 años aproximadamente. Ellos tienen acceso directo a la base de datos del software, y desean obtener información. Con este proyecto de datos buscan construir un dashboard que les permita mantener la gestión de la venta.

[2] El churn es indicador que muestra el porcentaje de suscripciones canceladas o no renovadas para una cierta organización

Anexo C - Encuesta aplicada sobre el sistema recomendador

Utilizando una escala Likert de 5 puntos, donde 1 es “muy en desacuerdo” y 5 es “muy de acuerdo”, por favor califique los ítems que se indican a continuación[3]:

Novedad en las recomendaciones

El sistema recomendador me ayudó a descubrir nuevos elementos.

Suficiencia de la interfaz

La presentación de los elementos recomendados por el sistema es clara.
La presentación de los elementos recomendados por el sistema es adecuada.
La interfaz del recomendador es atractiva.

Explicabilidad

El sistema recomendador explica por qué los productos me fueron recomendados.

Suficiencia de la información

La información que entrega el recomendador es suficiente para agregar los elementos al tablero.

Facilidad de uso percibida

Me familiaricé con el recomendador de manera rápida.
Pude identificar de manera fácil los elementos recomendados.

Transparencia

Entendí por qué los elementos me fueron recomendados.

Utilidad percibida

El recomendador me ayudó a buscar los elementos deseados.

Confianza

El recomendador me dio confianza a la hora de completar el tablero.

Intención de uso

Usaré el recomendador nuevamente.
Usaré el recomendador de manera frecuente.

[3] Los ítems fueron formulados alternadamente de forma positiva y negativa para evitar sesgos en las respuestas. Sin embargo, aquí se presentan todos los ítems formulados de forma positiva para facilitar su comprensión, y principalmente, para hacer más evidente la relación entre ellos.

Sobre la dinámica (preguntas proveedor)

1) ¿Cree que la identificación de tarjetas (post-its) es más fluida con ayuda del recomendador?
2) ¿Considera que lograría una mayor completitud usando el recomendador?
3) ¿Considera que el recomendador le ayudaría a obtener mejores resultados en menor tiempo?

Sobre la dinámica (preguntas cliente)

1) ¿Cree que la identificación de tarjetas (post-its) fue más fluida gracias a la ayuda del recomendador?
2) ¿Considera que se logró una mayor completitud usando el recomendador?
3) ¿Considera que el recomendador le ayudó a obtener mejores resultados en menor tiempo?

Anexo D - Resultados de la evaluación

D.1 Usuarios desarrolladores

Este anexo muestra las respuestas obtenidas de la evaluación con los usuarios desarrolladores. Las respuestas utilizan una escala Likert de 5 puntos, donde 1 es “muy en desacuerdo” y 5 es “muy de acuerdo”.

Sobre el sistema recomendador

Ítem / Puntaje	1	2	3	4	5
El sistema recomendador me ayudó a descubrir nuevos elementos					3
La presentación de los elementos recomendados por el sistema es clara				2	1
La presentación de los elementos recomendados por el sistema es adecuada				2	1
La interfaz del recomendador es atractiva			1	1	1
El sistema recomendador explica por qué los productos me fueron recomendados		1		2	
La información que entrega el recomendador es suficiente para agregar los elementos al tablero				1	2
Me familiaricé con el recomendador de manera rápida			1	1	1
Pude identificar de manera fácil los elementos recomendados			2		1
Entendí por qué los elementos me fueron recomendados				3	
El recomendador me dio confianza a la hora de completar el tablero				2	1
Usaré el recomendador nuevamente					3
Usaré el recomendador de manera frecuente					3

Sobre la dinámica, apoyada por el servicio de recomendación

Pregunta / Evaluación	1	2	3	4	5
¿Cree que la identificación de tarjetas (post-its) es más fluida con ayuda del recomendador?					3
¿Considera que lograría una mayor completitud usando el recomendador?					3
¿Considera que el recomendador le ayudaría a obtener mejores resultados en menor tiempo?					3

D.2 Usuarios clientes

Este anexo muestra las respuestas obtenidas de la evaluación con los usuarios clientes. Las respuestas utilizan una escala Likert de 5 puntos, donde 1 es “muy en desacuerdo” y 5 es “muy de acuerdo”.

Sobre la dinámica, apoyada por el servicio de recomendación

Pregunta / Evaluación	1	2	3	4	5
¿Cree que la identificación de tarjetas (post-its) fue más fluida gracias a la ayuda del recomendador?				1	2
¿Considera que se logró una mayor completitud usando el recomendador?				1	2
¿Considera que el recomendador le ayudó a obtener mejores resultados en menor tiempo?			1		2