Ingeniería de Software II

Sistema "Twitteando para Ahorrar"

Grupo 6

1º Cuatrimestre 2013

LU	Nombre	email
667/06	Daniel Foguelman	dfoguelman@dc.uba.ar
767/03	Hernán Modrow	hmodrow@gmail.com
511/00	Leonardo Tilli	leotilli@gmail.com

1. Parte I

En esta primera parte del informe presentamos la planificación para el proyecto TPA para las etapas subsiguientes a lo creado en la entrega anterior. En este proceso, integraremos el Core de Twiteando para Ahorrar con múltiples servicios (Redes sociales, detección de fraude, etc), tendremos requerimientos funcionales y no-funcionales de media y gran complejidad. Esto Nos mueve del paradigma SCRUM por uno más tradicional, RUP, haciendo preponderancia en la planificación y análisis de requerimientos antes que en la adaptabilidad al cambio y visibilidad del proyecto.

El plan desarrollado para esta nueva etapa sigue el modelo RUP, y se documenta a continuaciń. Primero se describirán los casos de uso identificados. En base a esto se detallan las distintas iteraciones que se planificaron y de qué consta cada una. En la última sección se encuentra el análisis de los riesgos más importantes que se detectaron.

Casos de Uso

Listado de Casos de Uso

CU-01 Autenticándose al sistema

CU-02 Cargando oferta a través de la Web-API

CU-03 Consultando oferta a través de API-Internet

CU-04 Consultando oferta a través de página web TPA

CU-05 Enviando sugerencia sobre el sistema

CU-06 Configurando oferta sugerida

CU-07 Buscando información de oferta sugerida

CU-08 Generando reporte de ofertas dudosas con Spam-Buster

CU-09 Invalidando oferta

CU-10 Cargando/Consultando oferta por Página Web/Red Social

CU-11 Generando reporte de ofertas dudosas con Módulo Propio

CU-12 Registrar información de uso del sistema

CU-13 Asignando confiabilidad a oferta

CU-14 Asignando confiabilidad a usuario

CU-15 Comparando SpamBuster con Módulo de ofertas dudosas

CU-16 Configurando sistema de confianza personal

CU-17 Mostrando mapa oferta

CU-18 Analizando web en busca de ofertas

CU-19 Cargando/Consultando oferta por SMS

CU-20 Consultando información estadística del sistema

CU-21 Configurando sistema de ofertas

Detalle de los Casos de Uso

En esta sección incluiremos una lista de los casos de uso identificados para el sistema a implementar durante la primera iteración, con una breve descripción de alto nivel para cada uno. Se trata solamente de interacciones entre el sistema y agentes externos (es decir, el usuario y otros sistemas). Por esta razon, esta clasificación no contiene absolutamente todo el trabajo a realizar.

En particular, no se describe aquí el trabajo requerido para permitir a nuestro sistema detectar las ofertas posteadas en las diversas redes soportadas.

CU-01: Autenticándose al sistema Un usuario con una cuenta de usuario válida podrá utilizarla para autenticarse con el sistema. Está cuenta podrá ser una cuenta OpenId, de alguno de los sistemas externos soportados o de usuario interno para tareas de configuración y administración

CU-02: Cargando oferta a través de la Web-API Un usuario podrá cargar una nueva oferta de un producto válido en el sistema. En caso de que el usuario esté autenticado se le dará prioridad de acuerdo a . Los usuarios pagos podrán además incorporar más información a la oferta para que esta se ubique como un aviso esponsoreado.

CU-03: Consultando oferta a traves de API-Internet Un usuario podrá realizar consultas sobre las mejores ofertas que tiene el sistema. Los resultados de búsqueda estarán ordenados por las ofertas de mayor importancia, esto definido por las reglas internas.

CU-04: Consultando oferta a través de página web TPA Un usuario podrá realizar consultas sobre las mejores ofertas que tiene el sistema. Los resultados de búsqueda estarán ordenados por las ofertas de mayor importancia, esto definido por las reglas internas. En caso de lo usuarios auténticados, de tenerlo definido, se utilizarán además sus preferencias de confianza en las fuentes (ej: ciertos usuarios, webs).

CU-05: Enviando sugerencia sobre el sistema Un usuario del sistema, sin importar si está autenticado o no, podrá enviar sugerencias sobre el sistema. Las sugerencias Spam, de haberlas, deberán ser tratadas con una estrategia anti-spam.

Riesgos

R1: Dado que **el/los cluster/s armados pueden no tener suficiente poder de computo** existe el riesgo de que **los tiempos de respuesta y el conjunto de datos analizados no sean los esperados.**.

CONTEXTO: Se espera manejar grandes volúmenes datos y de peticiones por lo tanto necesita alto poder de procesamiento y una gran cantidad de espacio para almacenarse.

PROBABILIDAD: Media

IMPACTO: Alto

EXPOSICIÓN: Alta

MITIGACIÓN: Diseñar arquitectura que escale horizontalmente y sea fácil agregar capacidad de procesamiento y de datos.

CONTINGENCIA: Disminuir la cantidad de información que se puede recibir y enviar TPA a los usuarios. Analizar la distribución del sistema en nodos que abarquen las consultas de determinadas regiones.

R2: Dado que **la arquitectura se define con computadores de bajo costo** existe el riesgo de que **se caíga la base de datos con la información de las ofertas**.

CONTEXTO: La caída de la infraestructura de repositorios de ofertas generaría una caida del servicio completo.

PROBABILIDAD: Baja

IMPACTO: Alta

EXPOSICIÓN: Media

MITIGACIÓN: Información guardada con chequeos de integridad que permitan una fácil recuperación (RAID). Redundancia de la base de datos con información de la ofertas.

CONTINGENCIA: Recuperar la información utilizando chequeo de integridad de datos. Levantar sistemas de base de datos de backup.

R3: Dado que **el sistema se espera sea usado masivamente** existe el riesgo de que **se genere mucha carga en el uso del sistema que produzca lentitud de respuesta**.

CONTEXTO: Dado que el sistema TPA tendrá impacto a nivel nacional y regional, es esperable una gran carga de consultas.

PROBABILIDAD: Alta

IMPACTO: Alta

EXPOSICIÓN: Alta

MITIGACIÓN: Generar una arquitectura que permita la alta disponibilidad del servicio utilizando **cachés** distribuidos para minimizar la carga en cada servidor. Armar casos de tests específicos para detectar problemas de disponibilidad en el sistema.

CONTINGENCIA: Levantar clusters adicionales, en caso de no ser posible evaluar la contratación de clusters externos.

R4: Dado que **se quiere que el sistema sea usado por la mayor cantidad de gente y en la mayor cantidad de plataformas** existe el riesgo de que **no sea homogeneo entre las distintas plataformas y fácil de usar para todos los usuarios**.

CONTEXTO: La aplicación TPA deberá tener alcance nacional y ser inclusiva para todos y todas. Las personas de edad avanzada o con capacidades especiales deberán poder utilizar las interfaces de usuario de manera intuitiva.

PROBABILIDAD: Bajo

IMPACTO: Bajo

EXPOSICIÓN: Baja

MITIGACIÓN: Buscar especialistas en usabilidad para generar interfaces aptas.

CONTINGENCIA: No hacer nada, el porcentaje de estos usuarios es menor en comparación a los usuarios activos.

R5: Dado que **los servicios del sistema se proveerán a través de Internet** existe el riesgo de que **recibir un ataque de Denegación de Servicio (DoS y DDoS) no siendo posible brindar los servicios**.

CONTEXTO: Existen grupos de activistas que con distintas motivaciones atacan sitios mediante ataques de denegación de servicio para hacer llegar su mensaje. Estos ataques son faciles de generar y ejecutables en todo tipo de computadoras, incluso celulares o tablets, por lo que pueden ser facilmente escables por un grupo coordinado.

PROBABILIDAD: Media

IMPACTO: Alta

EXPOSICIÓN: Alta

MITIGACIÓN: Generar una aplicación distribuida con buena distribución de carga para poder soportar alta carga de datos. Organizar periodicamente simulaciones de este tipo de ataques para probar el sistema.

CONTINGENCIA: Minimizar el tiempo de reinicialización de los servicios de TPA. Analizar la distribución del sistema en nodos que abarquen las consultas de determinadas regiones. Evaluar la contratación de sistema contra este tipo de ataques.

R6: Dado que **se almacena información de los usuarios en sistema** existe el riesgo de que **esa información sea robada**.

CONTEXTO: Dada la exposición de los usuarios así como otra información que se requiera que sea relevante al sistema, ej: configuración de confianza, hábitos de utilización, dicha información es suceptible de ser comercializada.

PROBABILIDAD: Alta

IMPACTO: Medio

EXPOSICIÓN: Alta

MITIGACIÓN: Anonimizar la información de los usuarios utilizada internamente. Aislar los datos sensibles de los usuarios de otros partes del sistema, encriptar, restringir y auditar su utilización. Realizar simulaciones de ataques para encontrar vulnerabilidades.

CONTINGENCIA: Cerrar las consultas de información y dejar solo accesible las consultas de ofertas generales.

R7: Dado que **el sistema depende en gran parte de la información provista por lo usuarios** existe el riesgo de que **haya un alto porcentaje de ofertas dudosas**.

CONTEXTO: Dado que el proyecto es esponsoreado por el gobierno nacional, es posible que un número importante de ofertas dudosas sean generadas que buscan manipular el sistema en pos de un beneficio personal.

PROBABILIDAD: Bajo

IMPACTO: Medio

EXPOSICIÓN: Media

MITIGACIÓN: Informar a los usuarios de los beneficios de usar el sistema correctamente. Buscar la adopción temprana del sistema por parte de los usuarios para que estos generen un alto número de datos. Utilizar los servicios de SpamBuster para el filtrado de ofertas dudosas mientras se genera un módulo proprio. Invalidación de ofertas manualmente por usuarios administrativos/configuración.

CONTINGENCIA: Generar listas de productos oficialmente verificados por agentes de TPA.

R8: Dado que **dado el contexto inflacionario y de polarización política** existe el riesgo de que **información no sea confiada por los usuarios**.

CONTEXTO: La poca credibilidad en los índices inflacionarios podría inducir a la falta de confianza de los usuarios a los resultados de búsqueda.

PROBABILIDAD: Baja

IMPACTO: Bajo

EXPOSICIÓN: Baja

MITIGACIÓN: Generaremos reportes de confianza para entender las necesidades de los usuarios y mejorar la confiabilidad de las ofertas sugeridas.

CONTINGENCIA: Evaluar la realización de una campaña publicitaria para generar confianza en los usuarios. Evaluar la posibilidad de la captura de ofertas mediante fotos.

R9: Dado que **el sistema depende en gran parte de la información provista por lo usuarios** existe el riesgo de que **haya un gran número de usuarios no confiables**.

CONTEXTO: Dado que el proyecto es esponsoreado por el gobierno nacional, es posible que se registren usuarios que hagan busquen el fracaso del sistema, ej: disminuyendo la confiabilidad de las ofertas.

PROBABILIDAD: Bajo

IMPACTO: Medio

EXPOSICIÓN: Baja

MITIGACIÓN: Generar estrategias de validación de usuarios para minimizar la falsificación de identidad. Evaluar la generación programas de fidelización fin de que los usuarios quieran registrarse utilizando información fidedigna y participar activamente generando información confiable.

CONTINGENCIA: Los usuarios deberán verificarse personalmente para poder utilizar el sistema.

R10: Dado que **el presupuesto es acotado y se espera que el sistema genere ingresos propios a fin de mantenerse** existe el riesgo de que **los fondos obtenidos de los servicios pagos ofrecidos sean insuficientes y el proyecto fracase**.

CONTEXTO: El proyecto depende de fondo.

PROBABILIDAD: Baja

IMPACTO: Alto

EXPOSICIÓN: Media

MITIGACIÓN: Venta de servicios para sugerir ofertas de comerciantes. Venta de información estádistica del sistema.

CONTINGENCIA: Agregar ventea publicadad al sistema. Solicitar fondos adicionales al estado nacional. Buscar fuentes de financiación adicionales.

R11: Dado que **es necesario obtener información de ofertas distintos sistemas/páginas web mediante webcrawling** existe el riesgo de que **no se descubran correctamente las ofertas de los mismos**.

CONTEXTO: La programción de analizadores de leguaje natural es una tarea compleja que requiere un alto nivel de conocmiento para implementarse correctamente.

PROBABILIDAD: Media

IMPACTO: Medio

EXPOSICIÓN: Media

MITIGACIÓN: Alguno de los miembros del equipo asistará a una capacitación sobre el tema, para que luego realice capacitación interna. Implementaremos estratégias de entrenamiento de la plataforma y generaremos múltiples casos de test para verificar la identificación de ofertas. También utilizaremos una estrategia de supervisión de la identificación de ofertas por un operario manual a fin de mejorar la detección.

CONTINGENCIA: Contrataremos personal que identifique las ofertas que no se están encontrando para poder generar nuevas estrategias de detección.

R12: Dado que **es necesario almacenar la información de las ofertas en un base de datos no relacional (NOSQL) y el equipo tiene poca experiencia** existe el riesgo de que **de que el proyecto se atrase o de que fracase debido al mal armado del modelo de datos**.

CONTEXTO: Se solitica la utilización de una base de datos no relacional a fin guardar la información de las ofertas.

PROBABILIDAD: Alta

IMPACTO: Alto

EXPOSICIÓN: Alta

MITIGACIÓN: Alguno de los miembros del equipo asistará a una capacitación sobre el tema, para que luego realice capacitación interna.

CONTINGENCIA: Contratar a un especialista en el tema para que participe del proyecto. Alternativamente tercerizar esa parte del proyecto.

R13: Dado que **se utilizarán máquina es necesario armar uno (o varios) clusters y que los miembros delequipo no tienen experiencia en esto** existe el riesgo de que **de que el proyecto se atrase o de que fracase en cuanto al nivel de procesamiento de datos que realiza**.

CONTEXTO: Se dispone de PCs comunes para el deploy del sistema. Dado que se requiere un alto nivel de procesamiento es que se deberán armar clusters con las PCs a fin de generar mayor poder de computo.

PROBABILIDAD: Alta

IMPACTO: Alto

EXPOSICIÓN: Alta

MITIGACIÓN: Alguno de los miembros del equipo asistará a una capacitación sobre el tema, para que luego realice capacitación interna. Destrabar la importación de los servidores dedicados.

CONTINGENCIA: Contratar a un especialista en el tema para que participe del proyecto. Evaluar la contratación de un servicio externo de procesamiento de datos.

R14: Dado que **los fondos destinados al contrato de Spam-Bust son acotados** existe el riesgo de que **haya perdida de funcionalidad control de spam**.

CONTEXTO: Dado la partida presupuestaria a destinada al pago de acotado e incluso podría no materializarse.

PROBABILIDAD: Baja

IMPACTO: Alto

EXPOSICIÓN: Medio

MITIGACIÓN: Gestionar adecuadamente los requerimientos de análisis de spam para que dichos CU no se retrasen. Contratar el servicio con la posibilidad de poder diferir/postergar los pagos.

CONTINGENCIA: Que el equipo de desarrollo se aboque al desarrollo de la funcionalidad. Evaluar la tercerización del desarrollo

R15: Dado que **equipo de trabajo es reducido** existe el riesgo de que **un miembro de equipo renuncie y el proyecto se atrase**.

CONTEXTO: El equipo de trabajo es pequeño (3 personas). Los miembros de equipo están entusiasmados por los desafíos que impone el proyecto

PROBABILIDAD: Baja

IMPACTO: Alto

EXPOSICIÓN: Medio

MITIGACIÓN: Reservar fondos para una bonificación de entrega del proyecto a tiempo. Incorporar nuevos desarrolladores como backups.

CONTINGENCIA: Negociar replanificación del proyecto con el cliente. Conseguir reemplazo.

Priorización de Riesgos

Prioridad	Riesgo	Exposición
1	R1	Alta
2	R3	Alta
3	R12	Alta
4	R13	Alta
5	R5	Alta
6	R6	Alta
7	R2	Media
8	R10	Media
10	R14	Media
11	R15	Media
9	R11	Media
11	R4	Baja
12	R7	Baja
13	R8	Baja
14	R9	Baja

Plan de Proyecto

Considerando las posturas de los stakeholders con los que se tuvo contacto, se decidió dar prioridad a la usabilidad, rendimiento, e integrabilidad y extensibilidad. Para esta segunda parte se cuenta con una dedicación *full-time* de los tres integrantes del equipo.

1.0.1. Iteración 1 - Elaboración (2 semanas / 240 horas)

- Definición de arquitectura
- CU-01: Autenticándose al sistema
- CU-02: Cargando oferta a través de la Web-API
- CU-03: Consultando oferta a través de API-Internet
- CU-04: Consultando oferta a través de página web TPA
- CU-05: Enviando sugerencia sobre el sistema

Detalle de la iteración

Los casos de uso incluidos en esta primera iteración conforman un subconjunto mínimo que nos permite tener un recorrido completo de la aplicación. Esto último implica también que varios de éstos tiene un alto impacto en la arquitectura, teniendo que tomar decisiones al respecto de la autenticación, implementación del servicio Web-API, integración del sitio Web-TPA con el servicio API, modelo de persistencia de entidades y resolución de búsquedas.

La decisión de generar un recorrido completo también tiene en cuenta los riesgos relevados, dado que los riesgos de mayor exposición están relacionado con cuestiones de performance y arquitectura. Esto junto con la búsqueda de que haya una adopción temprana del sistema por parte de los usuarios, permitirá evaluar tempranamente el funcionamiento del sistema para corregir en una fase temprana cualquier inconveniente de arquitectura que pudiera surgir y no hubiese sido relevado.

Desde el punto de la usabilidad, incluimos el envío de sugerencia para tener un feedback temprano del usuario. Desde el punto de vista del rendimiento, apuntamos a utilizar una interfaz simple, que minimice el intercambio de datos por consulta. Por último, desde el punto de vista de la integrabilidad y extensibilidad, decidimos implementar todos las funcionalidades en el servicio Web-API, reutilizándolo desde el sitio Web-TPA.

1.0.2. Iteración 2 - Elaboración (2 semanas / 240 horas)

- CU-06: Configurando oferta sugerida (24hs)
- CU-07: Buscando información de oferta sugerida (32hs)
- CU-08: Generando reporte de ofertas dudosas con Spam-Buster (64hs)
- CU-21: Configurando sistema de ofertas (56hs)
- CU-10: Cargando/Consultando oferta por Página Web/Red Social [Twitter] (64hs)

1.0.3. Iteración 3 - Elaboración (2 semanas / 240 horas)

- CU-10: Cargando/Consultando oferta por Página Web/Red Social [cont.] (64hs)
- CU-09: Invalidando oferta (40hs)
- CU-11: Generando reporte de ofertas dudosas con Módulo Propio (64hs)
- CU-12: Registrar informacion de uso del sistema (24)
- CU-13: Asignando confiabilidad a oferta (48)

1.0.4. Iteración 4 - Construcción (4 semanas / 480 horas)

Las siguientes tareas se desarrollan para finalizar con el deplyment del sistema

- CU-14: Asignando confiabilidad a usuario (40)
- CU-10: Cargando/Consultando oferta por Página Web/Red Social [cont.] (40hs)
- CU-15: Comparando SpamBuster con Módulo de ofertas dudosas (160hs)
- CU-16: Configurando sistema de confianza personal (136hs)
- CU-17: Mostrando mapa oferta (104hs)

1.0.5. Iteración 5 - Transición (2 semanas / 240 horas)

- CU-10: Cargando/Consultando oferta por Página Web/Red Social [cont.] (60hs)
- CU-18: Analizando web en busca de ofertas (60hs)
- CU-18: Cargando/Consultando oferta por SMS (64hs)
- CU-20: Consultando información estadística del sistema (56 hs)

Gantt de la Primera Iteración

ID	Task Name	Duration
1	**Elaboración 01 - Iteración 01**	**10 days**
2	Definición de Arquitectura	**5 days**
3	Definición de Alternativas	16 hrs
4	Evaluación de acuerdo a atributos	8 hrs
5	Documentación	16 hrs
6	**Web API**	**5.5 days**
7	Definición Modelo de Entidades	12 hrs
8	Definición Interfaz	4 hrs
9	**SerDe**	**0.5 days**
10	Deserialización Parámetros	4 hrs
11	Serialización Resultados	4 hrs
12	**Login**	**1 day**
13	Integración con OpenID	8 hrs
14	**Carga de Ofertas**	**1.5 days**
15	Oferta Básica	8 hrs
16	Oferta Sponsoreada	4 hrs
17	**Consulta de Ofertas**	**2 days**
18	**Búsqueda Puntual**	**2 days**
19	Por Descripción	8 hrs
20	Por Precio	4 hrs
21	Por Zona	4 hrs
22	Sustitutas/Relacionadas	4 hrs
23	Promocionados	4 hrs
24	Sugerencias de usuarios	4 hrs
25	**Web Site TPA**	**7 days**
26	**Diseño de Look & Feel**	**3.5 days**
27	Maquetado común	12 hrs
28	Estilos generales	8 hrs
29	Imágenes comunes	8 hrs
30	**Login**	**1 day**
31	Maquetado	4 hrs
32	Estilos	4 hrs
33	Integración API	4 hrs
34	**Carga de Ofertas**	**2 days**
35	Maquetado	8 hrs
36	Estilos	4 hrs
37	Integración API	4 hrs
38	**Consulta de Ofertas**	**1 day**
39	Maquetado	8 hrs
40	Estilos	4 hrs
41	Integración API	4 hrs
42	**Sugerencias de usuarios**	**1.5 days**
43	Maquetado	4 hrs
44	Estilos	4 hrs
45	Integración API	4 hrs

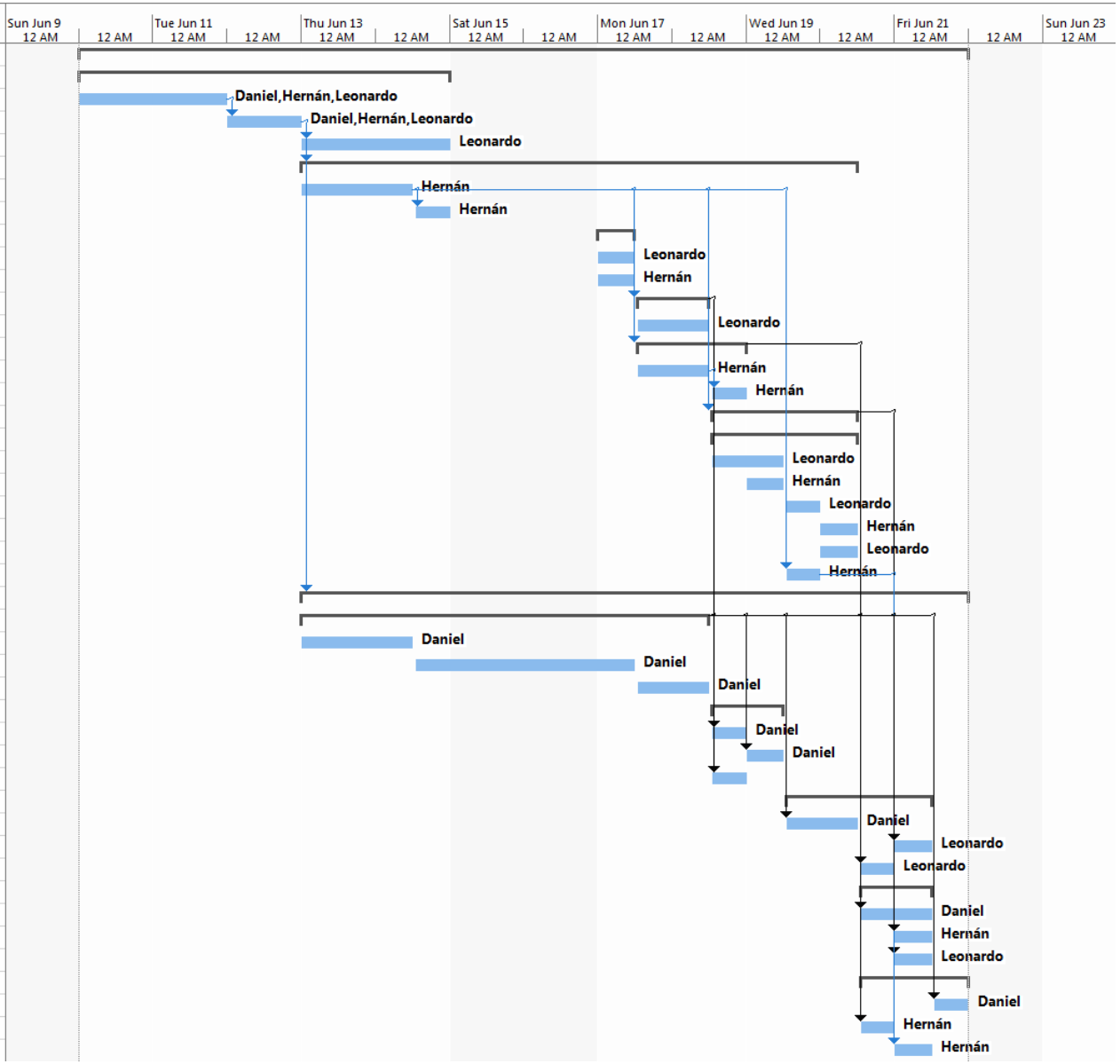


Figura 1: Diagrama de Gantt de tareas de la primera iteración, con división de tareas y asignación de recursos

2. Parte II

En esta segunda parte del informe presentamos el análisis realizado al respecto de los distintos atributos de calidad identificados, exponiendo varios escenarios para cada uno de éstos. También mostramos la arquitectura definida para la aplicación, utilizando distintos diagramas (de conectores y componentes, y de alocación), y complementamos estos últimos mediante una descripción detallada de la interacción entre los distintos componentes y artefactos. También explicamos las decisiones que nos llevaron a definir la arquitectura, y cómo ésta cubre los distintos escenarios de calidad.

Por último, compartimos nuestra apreciación al respecto de las similitudes y diferencias entre la modalidad de trabajo para el primer trabajo práctico y éste, comparando metodologías y alcances.

Atributos de Calidad y Escenarios

Atributo: **Usabilidad - Aprender a usar el sistema**
Comentario: Que Todos y Todas lo puedan usar

Fuente:	Usuario
Estímulo:	Desea realizar por primera vez una consulta simple (ofertas de un producto o cercanas a una dirección)
Artefacto:	Sistema
Entorno:	En ejecución
Respuesta:	Realiza la consulta sin inconvenientes
Medida:	Realiza la consulta en menos de 30 segundos

Atributo: **Usabilidad - Usar sistema eficientemente**
Comentario: Los usuarios contarán con un sistema de confianza configurable

Fuente:	Usuario
Estímulo:	Desea agregar un ítem a su listado de confianza personal
Artefacto:	Sistema
Entorno:	En ejecución
Respuesta:	El ítem es configurado exitosamente en la posición indicada de la lista
Medida:	El ítem se agrega en menos de 2 minutos

Atributo: **Usabilidad - Minimizar errores del usuario**
Comentario: El sistema debe tener la capacidad de anticiparse a las búsquedas de los usuarios

Fuente:	Usuario
Estímulo:	Desea ingresar bien el producto o dirección
Artefacto:	Sistema
Entorno:	En ejecución
Respuesta:	Se sugieren valores para el producto o dirección en base a valores aceptados por el sistema
Medida:	Las sugerencias aparecen en menos de 1 s. de ingresar una letra

Atributo: **Usabilidad - Personalizar sistema**
Comentario: El sistema tendrá un mecanismo interno de reputación

Fuente:	Usuario
Estímulo:	Desea asignar confiabilidad a un usuario u oferta
Artefacto:	Sistema
Entorno:	En ejecución
Respuesta:	Se asigna la confiabilidad solicitada por el usuario
Medida:	El usuario encuentra satisfactorio poder personalizar la configuración de confianza

Atributo: **Rendimiento**
Comentario: Una característica distintiva del sistema debe ser su inmediata respuesta

Fuente:	Usuarios
Estímulo:	Realiza consulta sobre ofertas de un producto
Artefacto:	Sistema
Entorno:	Modo normal
Respuesta:	El sistema procesa la consulta y obtiene el listado de ofertas acuerdo a la confianza personal del usuario
Medida:	El pedido es procesado en promedio en menos de 2 segundos

Atributo: **Rendimiento**
Comentario: Las ofertas deben poder visualizase en un mapa

Fuente:	Usuarios
Estímulo:	Solicita visualizar el resultado de una consulta en un mapa
Artefacto:	Sistema
Entorno:	Sistema sobrecargado
Respuesta:	El sistema ingresa las ofertas en el mapa y envía el mapa al usuario
Medida:	El pedido es procesado en promedio en menos de 10 segundos

Atributo: **Disponibilidad**
Comentario: Tiempo de disponibilidad

Fuente:	Usuario
Estímulo:	Realiza una consulta de ofertas
Artefacto:	Sistema
Entorno:	Normal
Respuesta:	El sistema responde a la consulta
Medida:	El sistema devuelva la información el 99,9 % de las veces

Atributo: **Disponibilidad**
Comentario: Es posible utilizar el sistema sin conexión

Fuente:	Usuario
Estímulo:	Realiza consulta / envía oferta
Artefacto:	Aplicación cliente
Entorno:	Sin conectividad
Respuesta:	El sistema guarda la operación para sincronizar cuando vuelva la conectividad.
Medida:	Disponibilidad offline

Atributo: **Disponibilidad - Tiempo de Reparación**
Comentario: El sistema no debe caerse

Fuente:	Nodo del cluster
Estímulo:	Se invalida, queda inutilizado
Artefacto:	Sistema
Entorno:	Normal
Respuesta:	El sistema se adapta a los n-1 nodos, si fuera un nodo distinguido reasigna esta responsabilidad en otro nodo que se encuentre online.
Medida:	El sistema se adapta en menos de 1 minuto desde la detección de la caída. Se detecta la caída en menos de 30 segundos.

Atributo: **Modificabilidad - Interoperabilidad**
Comentario: Interactibilidad con redes sociales

Fuente:	Desarrollador
Estímulo:	Integrar el sistema con una nueva red social
Artefacto:	Sistema
Entorno:	Desarrollo
Respuesta:	Interfaz modificada o nueva interfaz para integrar con red social
Medida:	El esfuerzo de desarrollo es menor a 40 horas

Atributo: **Modificabilidad - Modularidad**
Comentario: SpamBust será reemplazado por una módulo propio

Fuente:	Desarrollador
Estímulo:	Intercambiar Spambust por implementación propia
Artefacto:	Sistema
Entorno:	Desarrollo
Respuesta:	Intercambio de implementación de interfaz (inyección de nueva dependencia)
Medida:	El esfuerzo de desarrollo es menor a 8 horas

Atributo: **Modificabilidad - Portabilidad**
Comentario: Interacción con usuarios de múltiples plataformas

Fuente:	Desarrollador
Estímulo:	Adaptar interfaz web a un tipo de dispositivo nuevo
Artefacto:	Interfaz de usuario web
Entorno:	Desarrollo
Respuesta:	Adapta exitosamente la interfaz
Medida:	El esfuerzo de es menor a 40 horas

Atributo: **Modificabilidad - Capacidad**
Comentario: El sistema debe ser rápido y se cuenta con computadoras de escritorio

Fuente:	Administrador del sistema
Estímulo:	Agregar un nodo al cluster
Artefacto:	Sistema
Entorno:	En ejecución
Respuesta:	El nodo se agrega y comienza a procesar exitosamente
Medida:	El nodo está operativo en menos 2 horas

Atributo: **Modificabilidad - Content Management**
Comentario: La aplicación deberá recomendar productos. Las reglas se cambiarán con frecuencia.

Fuente:	Administrador de configuración
Estímulo:	Modifica relación entre productos
Artefacto:	Modulo de sugerencias
Entorno:	En ejecución
Respuesta:	El sistema utiliza la nueva configuración para relacionar productos a sugerir
Medida:	La modificación demora menos de 2 minutos

Atributo: **Seguridad - Auditoría**
Comentario: Se deben poder anular ofertas y las asociaciones de consumidores quieren poder ver cuáles fueron

Fuente:	Administrador de configuración
Estímulo:	Anular oferta
Artefacto:	Sistema
Entorno:	En ejecución
Respuesta:	El sistema audita la anulación
Medida:	Se identifica con una probabilidad mayor al 99 % al responsable de la modificación.

Atributo: **Calidad de datos**
Comentario: Se quiere detectar spam/ofertas falsas

Fuente:	Usuarios
Estímulo:	Envían ofertas spam
Artefacto:	Detector de Spam
Entorno:	En ejecución
Respuesta:	El sistema marca las ofertas como spam.
Medida:	El sistema detecta spam con un 95 % de probabilidad.

Atributo: **Calidad de datos**
Comentario: Se quiere detectar spammers/usuarios que informan ofertas falsas

Fuente:	Confianza de usuario
Estímulo:	Supera el threshold de confianza
Artefacto:	Sistema
Entorno:	En ejecución
Respuesta:	El sistema marca al usuario como spammer y lo blacklistea
Medida:	Los usuarios sin confianza se detectan con probabilidad del 90 %

Atributo: **Seguridad - Autenticación (Assurance)**
Comentario: Los usuarios podrán autenticarse usando OpenId

Fuente:	Usuario
Estímulo:	Se autentica
Artefacto:	Sistema
Entorno:	En ejecución
Respuesta:	El sistema valida las credenciales de usuario.
Medida:	El sistema resiste ataques de fuerza bruta sobre las credenciales y ataques de MiTM mediante certificados.

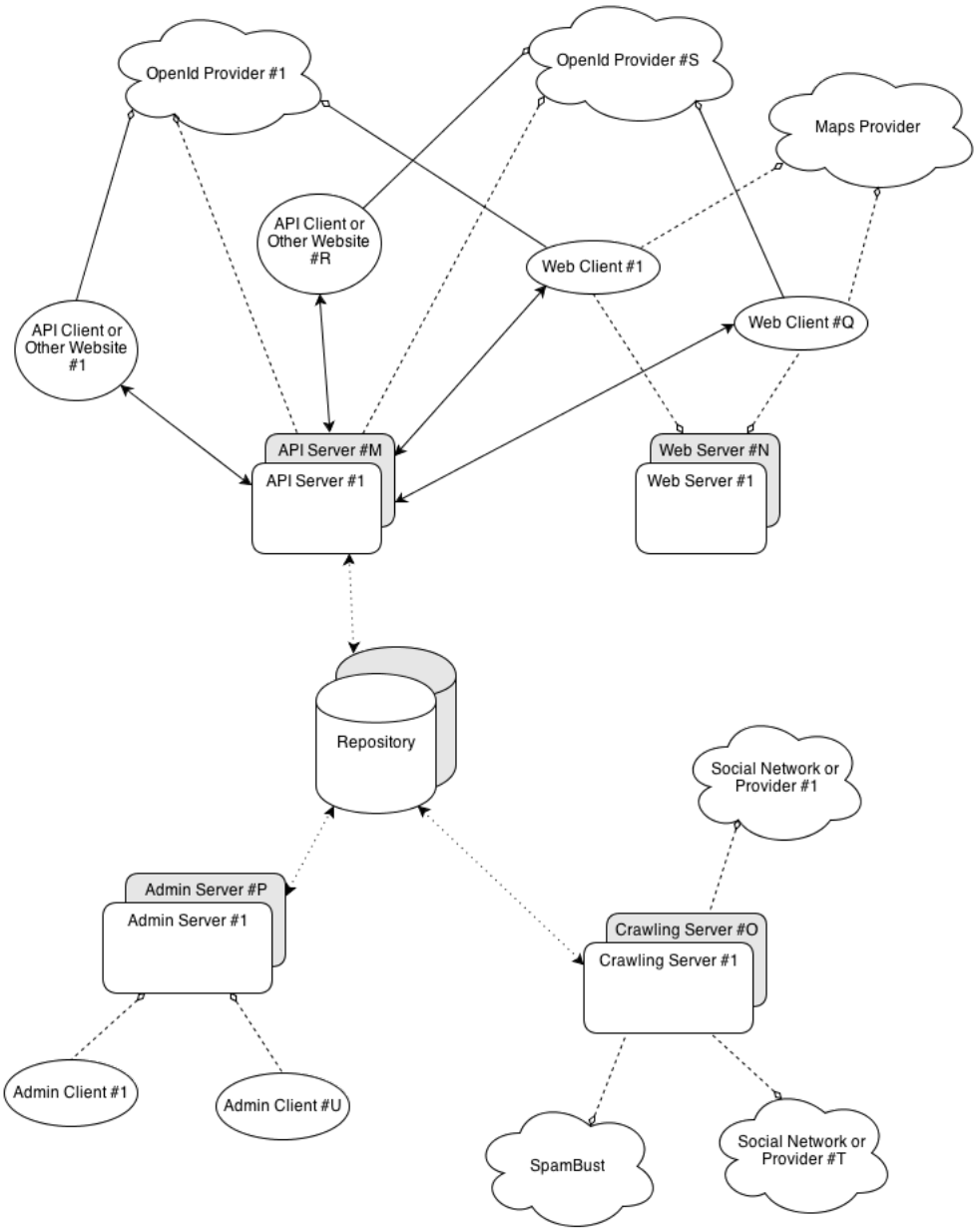
Atributo: **Seguridad - Ataques/Disponibilidad**
Comentario: El sistema no debe caerse, esto incluye ataques de DDoS

Fuente:	Atacante
Estímulo:	Denegación de servicio distribuido
Artefacto:	Sistema
Entorno:	En ejecución
Respuesta:	El sistema informa de sobrecarga en el sistema anómala
Medida:	El sistema detecta el ataque en menos de 2 minutos (?)

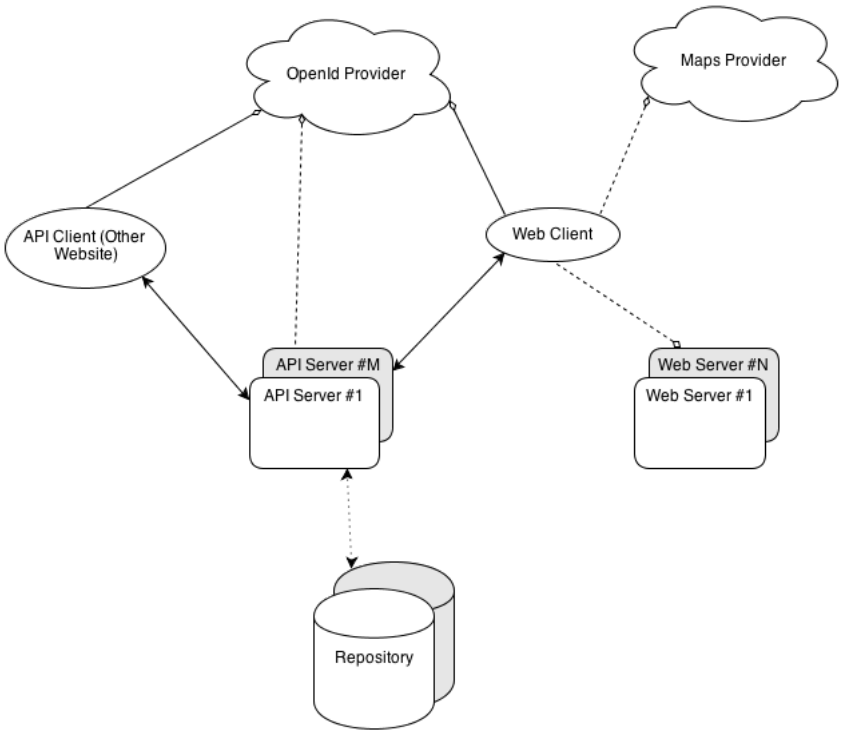
Explicación de la Arquitectura

Diagrama de Componentes y Conectores

Arquitectura Global



Consulta Web o API



En el diagrama de la consulta web se identifican dos tipos de clientes, que son los clientes del sitio Web y los que consumen únicamente la API web. Tenemos también dos tipos de componenetes servidor propios, que son los servidores del sitio Web y el servidores de la API web. Además, el diagrama nos muestra el repositorio de datos de configuración y consulta. Por último, tenemos distintos tipos de servicios externos que son consumidos por los componentes anteriores, como son los proveedores de OpenId y el proveedor de mapas.

Los clientes Web consumen el contenido estático directamente de los servidores del sitio Web, como puede ser el contenido HTML, JS, CSS e imágenes; se puede ver que el conector es de tipo cliente/servidor, lo que representa la sesión que arma el navegador con el servidor, y el bloqueo del thread de UI principal entre sucesivos requests (GET principalmente). En principio, este tipo de interacción es lo más simple posible, sin nigún tipo de autenticación; la mayor parte del contenido es cacheable y compone la UI de la aplicación (no tiene datos sensibles).

Tanto los clientes Web como los clientes de la API (como pueden ser otros sitios web) interactúan con el componente servidor API.

La autenticación es resuelta como parte de esta interacción, de acuerdo a la especificación de OpenId (`http://openid.net/specs/openid-authentication-2_0.html`):

- el cliente informa al componente servidor API el proveedor de OpenId que va a utilizar
- el componente API establece una sesión con el proveedor elgido (representamos esta sesión con un conector de tipo cliente/servidor) que podrá ser reutilizada para subsiguientes validaciones o para obtener datos extra del mismo cliente
- el cliente es redireccionado con el proveedor elegido para realizar la autenticación (representamos esta interacción como un envío de un mensaje a través de una conexión sincrónica)

- el componente API reutiliza la sesión establecida con el proveedor para verificar la información de autenticación y para obtener datos adicionales

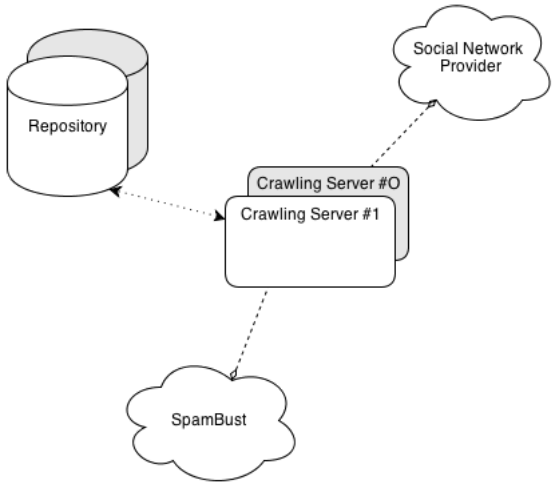
Más allá de manejar la autenticación, el componente servidor API responde a todos los pedidos de búsqueda o datos de configuración, de manera indistinta entre clientes Web o de API (usarían el mismo puerto); el tipo de conexión en este caso es asincrónica en ambos sentidos, para representar llamados tipo AJAX.

La interacción con el proveedor de mapas asumimos que se puede resolver directamente desde el cliente, mediante algún tipo de plugin descargado estáticamente desde el sitio Web.

El acceso al repositorio de datos y configuración lo hace únicamente el componente servidor API, para poder satisfacer los distintos requests, que agruparían todo el contenido dinámico de la aplicación.

Por último vale destacar que la cardinalidad del componente servidor API puede ser distinta a la del componente servidor Web, y dado que el primero atiende a consultas dinámicas, con workflows más complejos (como el de autenticación), y con mayor utilización de recursos, podría ser necesario escalarlo en mayor medida que al segundo, que sólo responde a requests de contenido estático. Además sólo es necesario implementar el proceso de autenticación en el componente servidor API, sustentado por el hecho de que sólo éste tiene acceso al repositorio, simplificando la arquitectura.

Crawling de Redes Sociales y Sitios Web



En el diagrama de crawling se identifica el componente de Crawling, el servicio de SpamBust, y el proveedor de red social (o sitio web de proveedor). Nuevamente, el diagrama nos muestra el repositorio de datos de configuración y consulta.

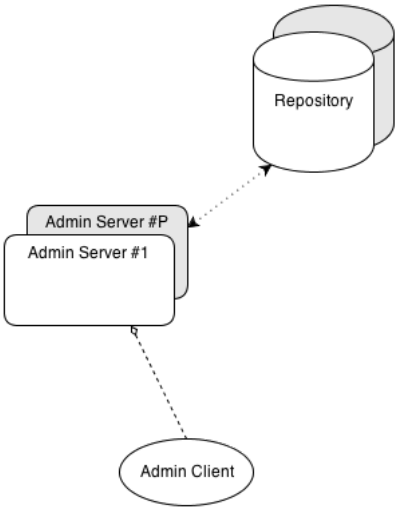
Los componentes de Crawling utilizan las distintas APIs provistas por las redes sociales para hacer las búsquedas de productos y ofertas; representamos esta interacción mediante un conector de tipo cliente/servidor, para generalizar el comportamiento de las distintas APIs. Podemos tener distintos componentes de Crawling, trabajando de manera independiente, cada uno realizando búsquedas distintas sobre la misma red social, o a distintas redes sociales.

El servicio de SpamBust es utilizado por los componentes de Crawling para filtrar ofertas de dudosa integridad, o que provienen de usuarios con baja reputación. Esta interacción también es representada mediante un conector tipo cliente/servidor, para generalizar las posibles implementaciones.

Por último, el repositorio es utilizado para consultar configuración de proveedores, usuarios, y redes sociales,

y también para persistir todos los datos de ofertas obtenidos en el proceso de crawling.

Administración



En el diagrama se identifican los componentes servidor y cliente de Administración. Además, el diagrama nos muestra el repositorio de datos de configuración.

El componente servidor de Administración provee tanto la UI como el contenido dinámico de la aplicación de adminsitración, que se utiliza para el ABM de proveedores, publicidad, y otras variables que puedan afectar el comportamiento de la aplicación. Además, maneja un esquema de autenticación y autorización propio, independiente de la aplicación principal, ya que la administración es realizada por usuarios internos. El acceso del componente cliente a la aplicación de adminitración se representa mediante un conector de tipo cliente/servidor.

Por último, el repositorio es utilizado para persistir toda la información de configuración, para ser utilizada por el resto de los sistemas.

Diagrama de Alocación

Arquitectura y Calidad

La arquitectura descripta en las secciones anteriores fue definida para cubrir los distintos escenarios detallados al comienzo del informe, y así cumplir con los atributos de calidad identificados.

Desde el punto de vista de usabilidad, el mayor aporte de la arquitectura es la separación total de la interfaz de usuario. Esto permite, en gran medida, independizar características de la UI que inciden directamente en la experiencia de usuario, como puede ser el maquetado.

Esta mayor independencia en el desarrollo de la UI nos ofrece varias ventajas:

- proveer distintas vistas que se adapten mejor a las preferencias del usuario, atacando temas como el uso eficiente, la personalización, y la sensación de confort
- ofrecer contenido estático de ayuda para toda la UI, accesible rápidamente y cacheable, cubriendo necesidades como el aprendizaje del sistema
- incluir validaciones simples y complejas del lado de cliente, minimizando la posibilidad de errores

Otro aspecto que aporta a la adaptabilidad y personalización, es el soporte de la arquitectura para identificar usuarios y permitirles persistir configuración propia.

El siguiente atributo de calidad más relevante, es el de rendimiento; en este caso la arquitectura lo ataca desde varios puntos. La arquitectura presenta varios componentes con responsabilidades diversas, y con bajo acoplamiento entre sí, facilitando partir las estrategias de performance entre todos estos componentes.

Desde el punto de vista de la experiencia de usuario, la arquitectura separa todos los request de contenido estático, de los requests de contenido dinámico, permitiendo paralelizar con distinta cardinalidad cada uno de esto puntos, destinando mayor cantidad de recursos a los componentes que más lo requieren.

En el caso de los componentes servidor API, se impulsa un aplicación stateless, en donde cada request tiene toda la información necesaria para ser atendido (header de autenticación y parámetros), lo que permite repartirlos de acuerdo a una estrategia de cola predefinida (round robin, recursos disponibles, etc.), aumentando el throughput (y disminuyendo el delay al no saturar recursos). Al alocar componentes relacionados en el mismo recurso físico, como una partición del repositorio y un componente servidor API, se minimiza el tráfico de red para requests que se puedan responder con esa partición, y así también el delay.

Desde el punto de vista de los componentes de crawling, se paraleliza el trabajo en distintos recursos físicos, aumentando el throughput.

El atributo de integrabilidad/extensibilidad se cubre desde varios puntos:

- el bajo acoplamiento entre componentes, permite su fácil modificación, en particular si el cambio no requiere cambiar la interfaz, y aún en estos casos se puede recurrir al uso de patrones conocido como el de adapter, bridge, etc
- la separación de la UI, da mayor flexibilidad a la hora de soportar nuevos dispositivos y plataformas
- los componentes de Administración dan soporte a la capacidad del sistema de modificarse en tiempo de ejecución.
- el acceso al servicio de SpamBust como un componente separado facilita el reemplazo de éste por una implementación propia, mientras que se respete la interfaz
- el escalamiento horizontal de recursos críticos, como el repositorio, permite adaptar su capacidad en tiempo de ejecución
- el acceso a las redes sociales de manera agnóstica de la API (utilizando patrones como Adapter, etc), agiliza la incorporación de nuevas redes sociales y sus APIs

La arquitectura cubre los aspectos de disponibilidad ofreciendo redundancia en cada uno de los subsistemas.

En el caso de los componentes servidor Web y API, se ofrece alta disponibilidad paralelizando los requests entre varias instancias de éstos, controlando activamente el estado de cada uno para detectar caídas y redireccionar cuando sea necesario.

En cuanto al repositorio, además de escalar horizontalmente mediante nuevas particiones, ofrece redundancia activa para cada una de éstas; desde el punto de vista de la alocación, esta redundancia va de la mano con la replicación de los otros componentes que conviven en el mismo recurso (como el servidor API).

Por último, tanto el sistema de Crawling como los componentes de Administración ofrecen un esquema de alta disponibilidad, replicándose cada uno en varios recursos.

Respecto a los aspectos de seguridad y auditoría, todo el acceso a contenido dinámico se hace desde componentes que implementan un esquema estricto de autenticación, y en el caso de administración implementan

también un esquema de autorización. Esto último permite que todo acceso al repositorio esté acompañado de credenciales que identifican al usuario responsable por el acceso, y así poder auditar la interacción.

Al separar la API de acceso público de los componentes de Administración, podemos llevar la mayor parte de las operaciones de modificación de la aplicación a un ambiente privado y controlado, dejando en la API pública únicamente operaciones de consulta y de configuración personal del usuario.

Para finalizar, respecto a la calidad e integridad de datos, la arquitectura presenta un esquema modular para el intercambio del componente Spambust, de acuerdo a la evaluación de resultados; además éste se encuentra conectado de una manera que lo acerca lo más posible al origen de los datos, que es al momento de recolección.

Conclusión

Al momento de comparar las metodologías utilizadas para la primera y segunda parte, notamos cómo éstas se ajustan a las características de cada uno de los escenarios: tiempo de planeamiento, velocidad de incorporación de cambios, dependencias entre los distintos hitos, criticidad de los entregables (calidad y responsabilidades).

En el caso de la primera parte, hay varios puntos importantes a tener en cuenta: el entregable es un prototipo, se incorporan nuevas tecnologías que deben ser investigadas, se desea tener la funcionalidad en un tiempo acotado, puede haber dependencias entre los componentes, pero son desconocidas de antemano. En este contexto, utilizar una metodología ágil como Scrum nos permite adaptarnos mejor a estos puntos, ya que, si bien hay una identificación inicial de los requerimientos y casos de uso, no nos exige tener un plan detallado de dependencias, y se pueden agregar tareas nuevas (y quitar otras) de acuerdo a la necesidad, manteniendo la duración de la iteración como la única constante. Además la flexibilidad en la planificación inicial es complementada con los tiempos cortos de reacción, producto de las prácticas como la reunión diaria standup.

En la segunda parte, hay otros puntos que favorecen el uso de prácticas como RUP: la longitud del proyecto es mucho mayor, los entregables son para uso productivo (mayor requisito de calidad y más responsabilidades), y entre estos últimos la arquitectura, que requiere un tiempo de diseño superior, y que con certeza define un árbol de dependencias mucho más marcado. Al utilizar metodologías como RUP se tiene un control mayor de los tiempos totales del proyecto, y los recursos involucrados; además se resuelven dependencias y factores de riesgo que pueden tener un impacto mucho mayor en la totalidad del proyecto.