UNIVERSIDAD DE ANTIOQUIA

Sistema de gestión de conocimiento y estrategias culturales sobre componentes SAP

Autor

Oscar David Guerra Jiménez

Universidad de Antioquia

Facultad de Ingeniería, Departamento de Sistemas

Medellín, Colombia

2019

Sistema de gestión de conocimiento y estrategias culturales sobre componentes SAP

Oscar David Guerra Jiménez

Informe de práctica como requisito para optar al título de:
Ingeniero de Sistemas

Asesora.

Diana Margot Lopez Herrera, Magíster en Ingeniería

Universidad de Antioquia
Facultad de Ingeniería, Departamento de Sistemas.
Medellín, Colombia
2019.

SISTEMA DE GESTIÓN DE CONOCIMIENTO Y ESTRATEGIAS CULTURALES SOBRE COMPONENTES SAP

Resumen

El sistema de repositorio fruto de este trabajo, permite realizar de manera ágil y flexible el almacenamiento de los código en lenguaje ABAP generados por los consultores al momento de realizar los diferentes requerimientos en el trabajo diario de la Fábrica de Software.

Para la realización del mismo se realizó la búsqueda de opciones existentes en el medio y se realizaron encuestas y reuniones con el personal de la empresa para que a través de la experiencia de cada uno de ellos se consolidará la forma adecuada de desarrollo que pudiera satisfacer sus necesidades de forma adecuada.

Con las experiencias otorgadas se procedió a esbozar el esquema de trabajo en los que se definió comenzar con el proceso de análisis en el que se identificarían las diferentes funcionalidades a implementar en el sistema con el fin de satisfacer las necesidades propuestas, continuando con el diseño en el cual se definiría la arquitectura del sistema, los diagramas considerados necesarios para el proyecto, los diferentes lenguajes y herramientas a utilizar para luego, en la etapa de desarrollo, generar todo el código pertinente junto con la retroalimentación brindada por el personal a quien se le iba mostrar los avances obtenidos.

Introducción

La gestión del conocimiento es un tema de preocupación para las empresas de desarrollo de software en la actualidad; debido al alto flujo de desarrollos que se presentan, la repetición de algunas tareas, el tiempo de reprocesos, el conocimiento tácito perdido, el conocimiento particular sobre los clientes y sus aplicaciones que se fugan en las organizaciones, a causa de la rotación de

personal y cambios de proyectos. La gestión del conocimiento que se genera en el día a día en las empresas de soporte es un activo de la organización que es necesario gestionarlo, además de generar una estrategia para la reutilización de códigos que ya hayan sido generados y probados correctamente en los diferentes requerimientos que se presentan en una fábrica de software.

En las fábricas de software hay un flujo constante de requerimientos que necesitan ser desarrollados, muchos de estos, a pesar de tener un objetivo y enfoque diferentes, tienen similitud en alguna de sus etapas de desarrollo. Generalmente en las fábricas que no tienen implantado un sistema de gestión de conocimiento suelen realizar los aplicativos desde cero sin re-utilizar ninguna parte del código, por lo que cada desarrollador tiene la tarea de realizar el análisis, diseño y posterior codificación de una parte del programa que pudo haber sido ya solucionada previamente por algún otro compañero de trabajo, generando así sobreesfuerzos y desaprovechando el tiempo que, de haber previamente conocido o reutilizado un componente desarrollado, habría significado un ahorro de tiempo y dinero que hubiera beneficiado a la compañía.

Un sistema de gestión de conocimiento permitiría almacenar e indexar los objetos intangibles resultantes después de haber dado solución a un requerimiento para posteriormente recuperarlos de una manera fácil y rápida generando un ahorro de tiempo al momento de que otro empleado pudiera reutilizar un módulo genérico previamente almacenado y que pudiera implementar en otro desarrollo nuevo.

A pesar de ello, un sistema de gestión por sí solo no causaría gran impacto debido a que no solo se requeriría una cultura de estandarización y modularización al momento de crear el código, sino que también correría por cuenta del empleado buscar en el repositorio de objetos el que sea adecuado para su interés, por lo que se necesitaría también implementar una buena indexación para que el esfuerzo del almacenamiento se vea recompensado al momento de realizar la reutilización. De lo anterior se puede concluir que debe haber una estrategia cultural que permita que todos los pasos mencionados anteriormente se lleven a cabo por parte de los empleados de la fábrica de software, esta gestión será exitosa.

Este proyecto tiene como finalidad diseñar un sistema de gestión de conocimiento y estrategias culturales para llevar a cabo su correcta implantación en la Fábrica

de Software de la empresa Perceptio S.A.S.; buscando optimizar el recurso humano y financiero utilizado en los nuevos desarrollos.

Objetivo General

Construir un Sistema de Gestión del Conocimiento que permita clasificar, conservar y recuperar los componentes software generados para los clientes SAP en la fábrica de software y definir las estrategias culturales necesarias para el uso del mismo, en la empresa Perceptio S.A.S

Objetivos Específicos

- Indagar acerca de las teorías de las fábricas de software y la gestión del conocimiento.
- Divulgar la importancia de la implantación de un sistema de gestión del conocimiento en el ambiente de la compañía Perceptio S.A.S en su enfoque como Fábrica de Software.
- Diseñar el sistema de gestión de conocimiento de componentes de software.
- Desarrollar el sistema de gestión de conocimiento de componentes de software.
- Definir la estrategia cultural para adecuado uso del sistema de gestión de conocimiento en la empresa
- Validar la funcionalidad del sistema.
- Presentar el sistema desarrollado y socializar las estrategias definidas a la compañía.

Planteamiento del problema

Actualmente Perceptio S.A.S. funciona bajo el modelo de Fábrica de Software, en la cual se gestionan, a demanda, los requerimientos de múltiples clientes, los cuales utilizan el ERP de SAP para la gestión de sus propias compañias. El sistema, al ser de un tercero, contiene una base común que es personalizada dependiendo de las necesidades requeridas por cada uno y los desarrollos

ejecutados por Perceptio en múltiples ocasiones usan lógica de programación similar incluso cuando son realizados para áreas y clientes diferentes.

Perceptio al no contar con un repositorio central de conocimiento del que se pueda extraer códigos de programación en lenguaje ABAP previamente realizados, depurados y correctamente implementados en requerimientos anteriores, genera muchos escenarios en los que distintos analistas de desarrollo puedan realizar lógicas de programación similares de forma diferente y con tiempos invertidos en investigación individualmente.

Marco Teórico

SAP es la compañía líder del mercado en software de aplicaciones empresariales, con soluciones para todas las necesidades de una organización de todos los tamaños y cualquier sector industrial; cuenta con 20.000 instalaciones en más de 120 países. Su sede principal está localizada en Walldorf, Alemania. En 1993 ingresó a América Latina, mediante la subsidiaria SAP Andina y del Caribe y, con una planta de 150 empleados, ya cuenta más de 166 clientes.

En Colombia existen 49 compañías que ya han implantado el software SAP entre las cuales se encuentran Avianca, Bavaria, Corona, Universidad de Antioquia, Sofasa, Grupo Nutresa etc, los cuales son potenciales clientes para cualquier compañia de consultoria en servicios SAP [1].

Perceptio SAS nace a principios del año 2000 a partir de una necesidad creciente en el mercado, que demandaba el desarrollo de software orientado a objetos y el uso de componentes reutilizables. En el 2011 adquirió el título de Partner de SAP debido al gran auge que tenía la plataforma y hoy cuenta con más de 35 certificaciones siendo el primer SAP Partner de Colombia certificado en Movilidad y HANA.
El área de desarrollo de aplicaciones basadas en SAP de la empresa Perceptio SAS trabaja bajo el modelo de Fábrica de Software registrando un alto flujo de requerimientos por parte de múltiples clientes los cuales a pesar de sus propias particularidades tienen generalidades que no son desarrolladas para ser re-utilizadas. Esto conlleva a que Perceptio desarrolle soluciones y adaptaciones a

la medida de los clientes, cuyo conocimiento una vez entregado el componente se pierde con la rotación del personal, el cambio de proyecto, entre otros.

El desarrollo de software realizado por una Fábrica de Software tiene un enfoque en componentes reutilizables y las Líneas de Productos de Software. Para mayor claridad es necesario acudir a algunas definiciones:

Fábrica de Software

Un modelo de referencia o forma de trabajo que se puede implementar en empresas que desarrollan productos de software, basados en equipos de trabajo multidisciplinarios conformados por profesionales especializados con roles y responsabilidades claras, que sigue procesos bien definidos para producir una familia de productos de software [2].

Los enfoques de las fábricas de software se enfatizan en la estandarización de los métodos y herramientas de desarrollo, la reutilización sistemática de los componentes o diseños de programa, división de departamentos por funcionalidades y la gestión interdisciplinaria de proyectos, así como el control de la calidad del producto [3].

Línea de Producto de Software

Se basa en el ensamblaje de partes de software previamente elaboradas la cual se inspira en los procesos de producción de sistemas físicos fundamentada en la reutilización de software y en la cual se asuma la existencia de una industria de partes [4].

Gestión del Conocimiento

De [5] y [6] Se infiere que un sistema de gestión del conocimiento es usado para administrar la recopilación, organización, refinamiento, análisis y diseminación del conocimiento de cualquier tipo dentro de una organización. No solo soporta funciones de organización, sino que también se encarga de satisfacer las necesidades del trabajador de conocimiento.

Un sistema de gestión de conocimiento es fundamental para lograr los objetivos de producción necesarios para la sostenibilidad y efectividad del modelo de fábrica de software.

Otros apartes a tener en cuenta en la gestión del conocimiento, son:

- La aproximación tecnológica se refiere a la técnica utilizada para desarrollar componentes.
- El enfoque metodológico representa la manera como se integra la reutilización dentro del proceso mismo de desarrollo.
- La modificación sugiere la forma como los componentes son accedidos y manipulados para adecuarlos a nuevas necesidades.
- El alcance de desarrollo indica si la reutilización sólo se aplica a componentes internos o permite integrar componentes de otras bibliotecas.
- El alcance del dominio delimita la reutilización a una familia de sistemas o permite su aplicación entre diferentes dominios de aplicación.
- El objeto de reuso se puede considerar desde el punto de vista de desarrollo, considerando el nivel de abstracción del componente o según el tipo de conocimiento que se reutiliza.

Reutilización Composicional

Se orienta a reusar productos; enfatiza la creación de nuevo software a partir de componentes almacenados en bibliotecas de componentes.

- Para manipular los componentes se pueden distinguir las siguientes formas:
- Por caja blanca: Sin observar el interior.
- Por caja negra: Se puede modificar el componentes
- Adaptativo: Se pueden observar y modificar ciertas partes adaptándolo a la nueva necesidad.

Reutilización Generativa

Se orienta a reutilizar el proceso de esfuerzos previos de desarrollo de software, generalmente concretados en herramientas que automatizan parte del trabajo de seleccionar, adecuar y generar nuevos componentes [7].

Metodología

El proyecto fue planteado a Perceptio como proyecto de prácticas para la Universidad de Antioquia siendo ejecutado como trabajador de planta y no como estudiante con contrato de aprendizaje como es usualmente realizado. Después de trabajar varios meses en la empresa se evidenció la problemática de la falta de un repositorio de conocimiento para los Analistas de Desarrollo por lo que se realizó la propuesta, la cual fue aceptada para ser realizada externamente, esto es, por fuera del horario laboral actualmente establecido.
Realización de entrevistas para establecer requerimientos y a continuación se desarrolló la metodología de ciclo de vida clásico al interior de un desarrollo ágil.

- Encuestas definición de requisitos
- Definición de funcionalidades
- diseño
- Desarrollo
- Pruebas
- Puesta en marcha

De acuerdo a la metodología se realizó el cronograma de actividades.

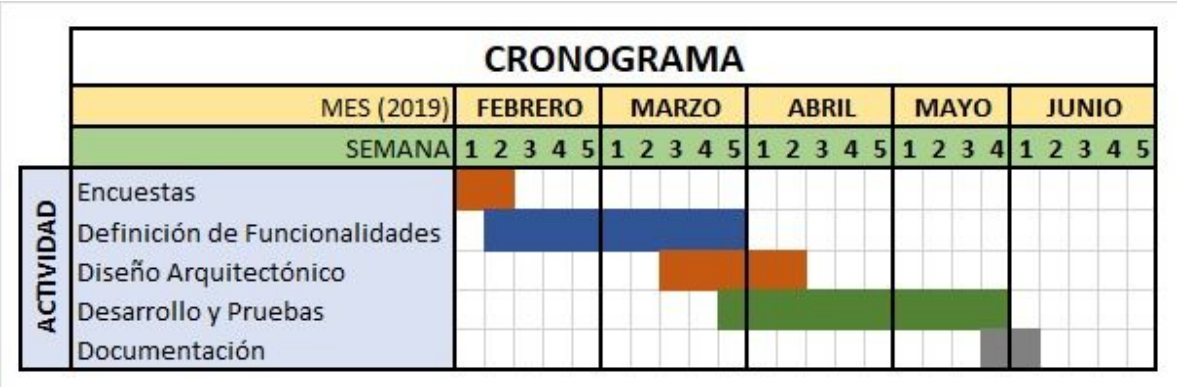

Fig. 1: Cronograma

Se realizó la ejecución del cronograma completando las actividades en los tiempos designados y recibiendo retroalimentación por parte de los compañeros analistas para posteriormente realizar los cambios adecuados y llegar a la presentación al gerente del área de tecnología el cual definiría la estrategia de implementación y

masificación, contando con la participación de la alta gerencia, con el fin de generar un proceso cultural y de rápida adopción del concepto de Fábrica de software, donde esta parte solo compete con el reuso de software.

DESARROLLO DEL PROYECTO

Después de realizada la propuesta se procedió a definir los requerimientos que debía tener el sistema, para lo cual se realizaron entrevistas y encuestas a los compañeros analistas evidenciando la falta de este sistema no solo en esta misma compañía sino también en otras empresas en la que el personal había trabajado previamente.
A continuación se relacionan las preguntas de la entrevista, las personas entrevistadas y la información obtenida:

Nombre	Cargo	Consultoras Trabajadas	Sistema de Gestión de Conocimiento	Herramienta de Repositorio Personal
Hernan Camilo Mejía Gaviria	Analista de Desarrollo en SAP	➢ TATA Consultancy Services ➢ DYNPRA ➢ EXPERT PARTNERS ➢ ABAPCOL ➢ Perceptio SAS	No tienen sistema	OneNote
Andrés Zapata Carmona	Analista de Desarrollo en SAP	➢ TATA Consultancy Services ➢ Perceptio SAS	No tienen sistema	Directorio de Archivos
Edgar Antonio Guzmán Marín	Analista de Desarrollo en SAP	➢ Abapcol ➢ TATA Consultancy Services ➢ Perceptio SAS	No tienen sistema	Directorio de Archivos
Jorge Alberto Jaramillo Pemberthy	Lider de Fábrica	➢ TLM ➢ IT-Nova ➢ GI-Proyectos ➢ Perceptio SAS	No tienen sistema	Directorio de Archivos
Andrés Felipe Ochoa	Analista de Desarrollo en SAP	➢ TATA Consultancy Services ➢ NetW ➢ Perceptio SAS	No tiene sistema	Documento de Texto y Google Drive

Fig. 2: Resultados de la encuesta

De las entrevistas realizadas se evidenció la manera como cada analista tenía creado para sí mismo su propio repositorio que les permite realizar búsquedas rápidas de los códigos, información y soluciones recopiladas después de afrontar múltiples proyectos y requerimientos; pero dicha información era de uso personal y solo accesible a cada funcionario que la creaba. Entre las herramientas utilizadas por los analistas se encontraban OneNote de Microsoft, Google Drive y directorios

de archivos y carpetas en Windows. Este es un activo importante y valioso de la compañía que se encontraba administrado a nivel personal; en caso de que uno de estos analistas abandonaran la compañía, este activo de conocimiento se iría con ellos

A partir de este punto, con la información recopilada de los compañeros y la experiencia propia en la que se trabajó para una empresa en la cual se contaba con un potente pero desaprovechado repositorio de código, debido a que era lento, poco intuitivo y desorganizado a causa de falta de control y estructuración de la información almacenada; se procedió entonces a crear los requerimientos para el sistema en los que se definía la agilidad, experiencia de usuario, velocidad, estructuración e integración como bases para un proyecto exitoso.

Agilidad

Se define una manera rápida de acceder al repositorio a través de accesos directos y por medio de extensión del navegador Google Chrome además de múltiples manera de acceso al sistema tal como con una aplicación de escritorio y un sitio web.

Experiencia de usuario

Creación de interfaces minimalistas y cotidianas con Material Design, como también la creación, edición, visualización y búsqueda de forma amena.

Velocidad

Utilizar la plataforma de Firebase para que el acceso a la información sea rápido y flexible.

Estructuración

Definición de estándares y limitación de las posibilidades de la información almacenada.

Integración

Conexión con la plataforma de Azure utilizada por Perceptio en el inicio de sesión y almacenamiento de archivos.

Durante la etapa de análisis se realizó la definición de componentes, usuarios y los diagramas considerados necesarios, de los cuales se muestran los siguientes:

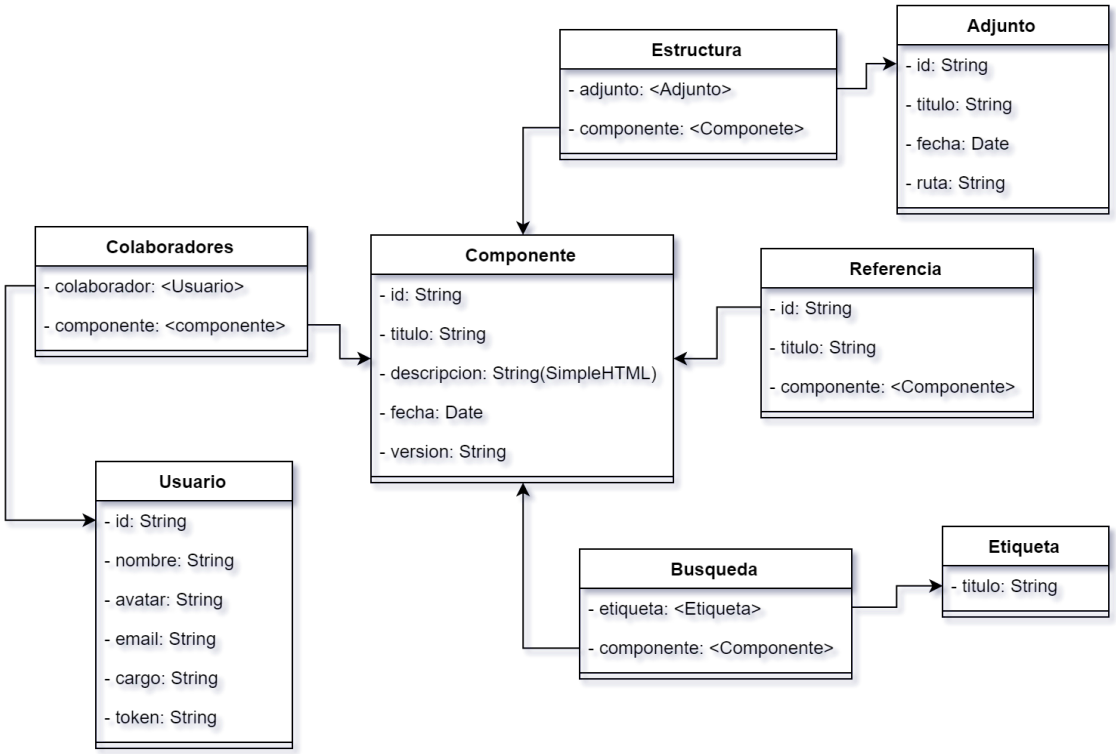

Fig. 3: Base de Datos

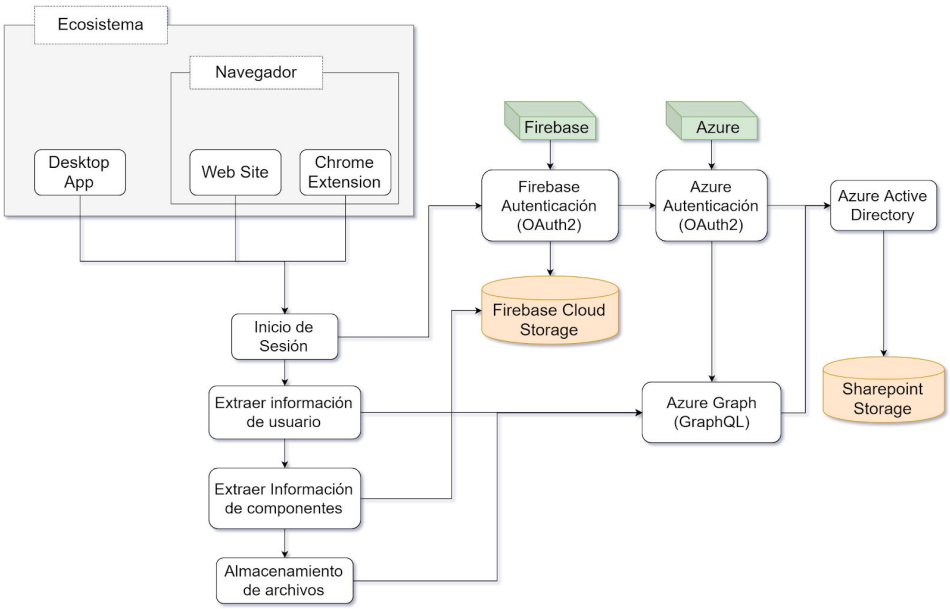

Fig. 4: Arquitectura del sistema

En la etapa de diseño en la que se escogieron los lenguajes de programación, se definió la arquitectura del sistema, la etapa de desarrollo para la creación del código y por último la etapa de documentación en la que se crearon los documentos requeridos junto con las estrategias acerca de cómo estructurar la información que será almacenada para su posterior reutilización.

Diseño de la Interfaz

A continuación se muestra el diseño de la interfaz de usuario del repositorio de código, lo cual consiste en una estructura principal formada por una cabecera común en todas las páginas en la que se encuentra el logo empresarial , el nombre del sistema “Repositorio de Código”, el botón para acceder a la información de perfil y el cierre de cuenta, el botón de acciones para acceder a la ayuda y configuración del sitio y el botón para abrir una pestaña al lado izquierdo que servirá para incluir secciones en futuras modificaciones.

Las páginas que utilizará el sistema serán las siguientes:

- Página de búsqueda la cual está conformada por una barra de búsqueda en la que se podrán ingresar las palabras claves que filtraran los componentes agregados al sistema por el título y etiquetas asociadas y la sección inferior donde se encuentran cada uno de los componentes con el título, una descripción corta, los íconos que indicará cuántos archivos se han adjuntado al componente, cuántos usuarios se ha asociados como colaboradores y en cuantas etiquetas ha sido marcado el componente, además de un botón para acceder a la información más detallada del componente y su edición

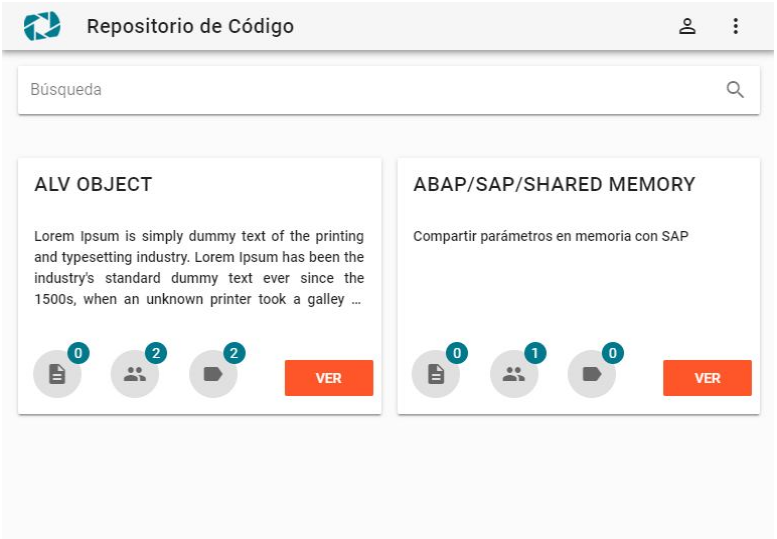

Fig. 5: Página de búsqueda

- Página de detalle y edición la cual servirá para ver la información extendida del componente (etiquetas incluídas, archivos adjuntos, usuarios asociados y referencias de utilización en proyectos internos de la empresa) y está conformada por las etiquetas en las que fue incluído, el título, la descripción creada, los archivos principales de código descargables en lenguaje ABAP, los archivos anexos descargables que ayudarán a complementarlo para su mejor entendimiento, las referencias o enlaces a la plataforma JIRA que indicará en cuáles proyectos se ha utilizado el componente, los usuarios asociados que pueden editar el componente y por último de un botón de edición que permitirá alternar la vista de detalle para poder agregar o eliminar los diferentes elementos descritos anteriormente.

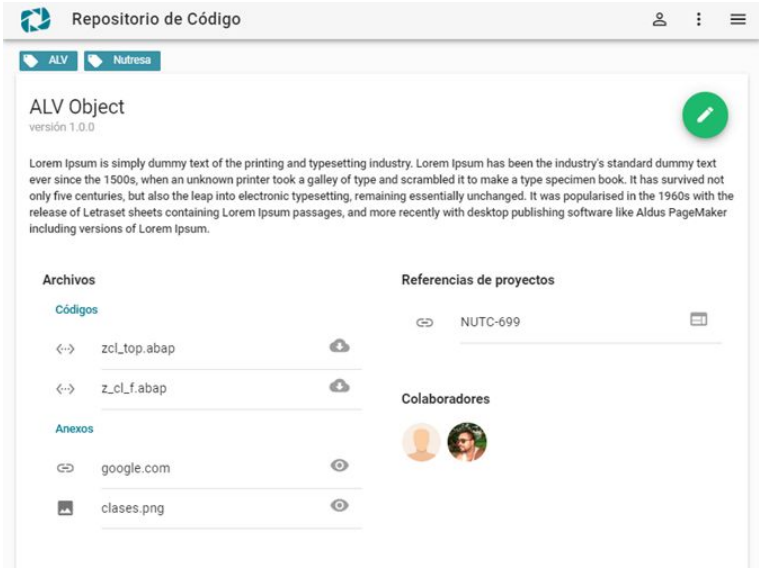

Fig. 5: Página de detalle y edición

La aplicación fue realizada utilizando el framework Javascript VueJS junto con el framework Vuetify; el primero se especializa en la creación de aplicaciones web por componentes lo cual permite la modulación del sistema con posibilidad de escalamiento, el segundo se enfoca en implementar el estándar de Material Design[8] el cual define un diseño de interfaz de usuario de acuerdo a lo establecido por Google tanto para aplicaciones de dispositivos móviles como para aplicaciones web.Todo con el fin de crear un sistema escalable con una interfaz amigable.

- Cultura de Uso

  https://docs.google.com/document/d/1p-qbuntw4jEj9AyOxnnUHN9cMMzC_VhtYyCrylxDxFU/edit?usp=sharing

- Manual de Usuario

  https://docs.google.com/document/d/1GDkmttJRxq-y94gUvwqwepMwXIajGYsdugTnrGDz3oY

Cultura de Uso

Uno de los principales temas a tener en cuenta para que la implementación del sistema sea exitoso a lo largo del tiempo, es que los objetos almacenados en el repositorio estén estructurados correctamente para poder ser reutilizados en múltiples desarrollos sin la necesidad de que se tengan que realizar cambios radicales; por lo que se consideró necesario definir una guía el la cual sean plasmadas las mejores prácticas de cómo deben

ser programados los objetos con el fin de que se lleven a cabo correctamente las tareas de reutilización descritas anteriormente. Esta guía está definida dentro del sistema como la sección de ayuda lo que permitirá ser accesible a los Analistas en cualquier de manera rápida.

Ingreso de un Componente al sistema:

Al momento de subir un componente al sistema se tienen diferentes secciones que corresponden a las características que lo definirán.

- **Títulos**: Nombre otorgado al componente.
- **Etiquetas**: Son las clasificaciones que se deben agregar al componente para realizar búsquedas más rápidas.
- **Descripción General**: Se describe a grandes rasgos las características del objeto cargado.
- **Códigos**: Son los archivos que contienen toda la lógica que debe implementarse para utilizar correctamente el componente. Hay que tener en cuenta que se podrá tener una visualización de los códigos anexados en lenguaje ABAP.
- **Anexos**: Son los archivos adicionales que servirán para extender la descripción en caso de que sea necesario. Es totalmente recomendable realizar diagramas que ilustran el flujo del componente, adicionar documentación util o anexar sitios web que permitan una fácil implementación del componente.
- **Colaboradores**: Usuarios que podrán realizar ediciones sobre el componente.

Para que se maximice las posibilidades de reutilización de los componentes ingresados en el sistema se definen entonces n cualidades básicas que deben tener los mismos:

- **Orientación a objetos**: Los códigos que sean ingresados deben estar separados por clases o en su defecto que puedan ser aplicados dentro de una sin presentarse ningún conflicto.
- **Independencia de Cliente**: Cada componente debe ser independiente del cliente utilizado o en su defecto especificar cada una de las dependencias necesarias para funcionar correctamente.
- **Abap Unit**: Implementar en los códigos Abap Unit para favorecer las pruebas.
- **Nombramiento de Variables**: Un buen nombramiento de variables permitirá adaptar fácilmente a nombramientos específicos para cada cliente.

Estructura de los Objetos

Para guardar los objetos de diccionario se utilizará la siguiente estructura

- Nombramiento de Objetos:
  La siguiente estructura dará ejemplo de como el siguiente: **Z[TT][####]_[#]**

  TT: Tipo de Objeto:
  - RP -> Reportes
  - MP -> Modul Pool
  - FG -> Grupo de Funciones
  - CL -> Clase
  - FM -> Módulo de Función
  - MD -> Método
  - IF -> Interface
  - #### -> Descripción corta
  - # -> Es un consecutiv

Estructura de los Códigos:

Para el código interno de cada objeto se define la siguiente forma de nombramiento de variables
G: Global
L: Local

- **Nombramiento**: [Variable] + [Tipo] + '_' + [Descripción corta]
  [Tipo]:

Tabla Interna	-> TI
Estructura	-> WA
Parámetro	-> PA
Select Option	-> SO
Field Symbol	-> FS
Constante	-> CTE
Type	-> TY
Text Field	-> TF
Group Box	-> GB
Radio Button	-> RB
Push Button	-> BT
Sub Screen	-> SS
TabScript	-> TS
Table Control	-> TC
GUI Status	-> GS
GUI Title	-> GT

- **Versionamiento**:
    - El versionamiento deberá realizarse mediante 3 números X.Y.Z y cada uno indicará lo siguiente:
    - El primero (X) será la versión mayor e indicará la versión principal del software. Ejemplo: 1.0.0, 3.0.0
    - El segundo (Y) será versión menor e indicará nuevas funcionalidades. Ejemplo: 1.2.0, 3.3.0
    - El tercero (Z) será la revisión e indicará que se hizo una revisión del código por algún fallo. Ejemplo: 1.2.2, 3.3.4

Validación de Solución

Para la validación se realizó una reunión con únicamente con Jorge Jaramillo al cual se le mostró el sistema en funcionamiento y el potencial de implementarlo de la forma como se realizó para futuros proyectos en Perceptio. Después de esta reunión el validó que todo estaba correcto y se aprobó como proyecto para la empresa. Además se consideró que el se encargaría de todo el proceso de masificación del sistema.

Solución

Para dar solución al problema inicial, se creó entonces un sistema con la capacidad de almacenar objetos de desarrollo en lenguaje ABAP con capacidad de realizar búsquedas y accesible de manera ágil para los Analistas de Perceptio.

Para este propósito se creó entonces los siguientes elementos:

- **Extensión de Chrome**: Es un componente interno de navegadores basados en Cromium [9] que permite extender las funcionalidades del navegador de diferentes maneras dependiendo de las necesidades. Se utiliza principalmente para agregar la agilidad al sistema por medio de búsquedas a través del Omnibox, Ícono de extensión para acceder de manera ágil, así como Hotkeys que permiten hacerlo sin necesidad de tener el navegador abierto.
- **Aplicación Web**: Sobre la extensión de Chrome se ejecuta una sitio web desarrollado en VueJS el cual se puede ejecutar tanto en la extensión, como en un hosting normal o inclusive como aplicación de escritorio por medio del programa NW.js[10]
- **Base de Datos Relacional en Tiempo Real**: Firebase[11] provee una Base de Datos que permite guardar y recuperar información almacenada en forma No Relacional de manera rápida por lo que en el proyecto se utiliza para almacenar la información de los componentes de software además de proveer el servicio de

autenticación a múltiples servicios tal y como es el caso de Microsoft Azure que es el que está implementado en este proyecto.

- **Microsoft Azure Active Directory**: Dado que el Sharepoint es el sistema principal de Perceptio, el sistema se adaptó para que tanto el inicio de sesión como el almacenamiento de los archivos de código utilizarán este servicio permitiendo así solo el ingreso de usuarios con el dominio de email @perceptio.net ,ademas de guardar los archivos en directorios de archivos comúnmente usados en la empresa

Conclusiones

Este sistema da respuesta entonces a la necesidad de contar con un repositorio de objetos de desarrollo que permita a los Analistas de Desarrollo realizar búsquedas de información sobre desarrollos previamente realizados y que estén estructurados con buenas prácticas para que puedan ser reutilizados en los requerimientos actuales así como tener la posibilidad de almacenar en el mismo los códigos propios que considere que puedan ser utilizados por otros Analistas en un futuro lo que en un futuro permitirá alivianar la curva de mantenimiento al hacer re-uso de los componentes de desarrollo al interior de la compañía

Realizar la implementación de este proyecto representó un gran enriquecimiento profesional y personal, poniendo a prueba las habilidades, conocimientos y creatividad en todos los retos que se enfrentaron.

El sistema será utilizado por los Analistas de Desarrollo en SAP - ABAP de la compañía el cual les permitirá consultar y visualizar de manera ágil los códigos bien estructurados, con buenas prácticas y en lenguaje ABAP de desarrollos realizados previamente optimizando tiempo de investigación y evitando copias de código extraídos de internet no verificados correctamente.

Las limitaciones presentadas durante la realización del proyecto se presentaron como:

- Estructuración y definición de la arquitectura por la falta de acompañamiento debido a que se desarrolló como proyecto externo.
- Falta de apoyo de un equipo de trabajo debido a la separación entre las actividades de la empresa y el desarrollo del proyecto.

- Falta de conocimiento a fondo de los frameworks utilizados en el desarrollo.
- Integración con de las diferentes herramientas y framework escogidos para facilitar el acceso al repositorio.
- Dificultad de la integración con las herramientas de la compañía debido a la arquitectura escogida y a los permisos para acceder a las mismas.
- Estructuración de la Base de Datos de forma No Relacional para facilitar la flexibilidad y velocidad en la programación.

A la fecha se realizó la entrega de el sistema ya implementado con acceso a entornos de desarrollo con cuentas personales por lo que faltaría realizar la migración a cuentas empresariales, también del documento con las estrategias que se definieron para el almacenamiento de los objetos de desarrollo. Los artefactos son los siguientes:

- Proyecto en Firebase con dependencia de los módulos de Autenticación, Cloud Firestore y Storage [11].
- Código fuente de la Extensión de Chrome[12] lista para ser subida en la tienda de Google Chrome [13].
- Código fuente de la Aplicación Web desarrollada en VueJS[14] con VuetifyJS[15].
- Manual de usuario del sistema con estrategia de almacenamiento de objetos de desarrollo.

Referencias bibliográficas

1. Revista Dinero - “SAP, líder en soluciones empresariales”, [Online]. Available: https://www.dinero.com/edicion-impresa/institucional/articulo/sap-lider-soluciones-empresariales/14033
2. Straccia, Luciano; Maulini, Adriana; Pytel, Pablo; Masci, Marcelo; Vegega, Cinthia; Pollo-Cattaneo, Ma. Florencia - “La Gestión del Conocimiento en Pequeñas y Medianas Fábricas de Software en el Área Metropolitana de Buenos Aires, [Online]. Available: ”http://sedici.unlp.edu.ar/bitstream/handle/10915/62080/Documento_completo.pdf-PDFA.pdf?sequence=1
3. Michael A. Cusumano - “Software Factory”, [Online]. Available: https://onlinelibrary.wiley.com/doi/full/10.1002/0471028959.sof323
4. Jonás A. Montilva C. - “Desarrollo de Software Basado en Líneas de Productos de Software”, [Online]. Available: http://www.ieee.org.ar/downloads/2006-montilva-productos.pdf

5. Jonás A. Montilva C., Ph.D - “Desarrollo de Software Basado en Líneas de Productos de Software”, [Online]. Available: http://www.ieee.org.ar/downloads/2006-montilva-productos.pdf
6. Dr. C. Israel A. Núñez Paula1 y Ing. Yiny Núñez Govín2 - “Propuesta de clasificación de las herramientas - software para la gestión del conocimiento”, [Online]. Available: http://scielo.sld.cu/scielo.php?script=sci_arttext&pid=S1024-94352005000200003
7. Raquel Anaya de Páez - “Un acercamiento a la reutilización en la ingeniería de software”, 2012, [Online]. Available: http://publicaciones.eafit.edu.co/index.php/revista-universidad-eafit/article/download/1074/966/
8. Material Design, [Online]. Available: https://material.io/design/
9. Proyecto Chromium, [Online]. Available: https://www.chromium.org/
10. Proyecto NW.js, [Online]. Available: https://nwjs.io/
11. Plataforma de Firebase, [Online]. Available: https://firebase.google.com
12. Api de desarrollo de extensiones para Google Chrome, [Online]. Available: https://developer.chrome.com/apps/api_index
13. Tienda de extensiones para Google Chrome, [Online]. Available: https://chrome.google.com/webstore
14. VueJS - Framework para Javascript, [Online]. Available: https://vuejs.org/
15. Librería de Material Design para VueJS, [Online]. Available: https://vuetifyjs.com/en/