BENEMÉRITA UNIVERSIDAD AUTÓNOMA DE PUEBLA

FACULTAD DE CIENCIAS DE LA COMPUTACIÓN

ERP PARA ADMINISTRAR UNA ZONA URBANA

TESIS PROFESIONAL
para obtener el título de
LICENCIADO EN CIENCIAS DE LA COMPUTACIÓN

PRESENTA
Manuel Pérez Garrido

ASESOR:
M.C. Mariano Larios Gómez

Puebla, Puebla 2019

Dedicatoria:

Llegar hasta este momento ha sido largo, no cabe duda de que he tomado varias decisiones; de las cuales algunas han sido acertadas y otras no tanto. Estas junto con el apoyo incondicional de mi familia, amigos, y personas especiales, me han permitido avanzar y en cada una de esas decisiones, mis padres siempre han estado presentes, a pesar de no estar siempre de acuerdo, me han apoyado de manera incondicional, y me han brindado las posibilidades óptimas para terminar mis estudios y seguir adelante en mi formación profesional.

Es por ello que con mi profundo agradecimiento a Dios por estar siempre presente cuando cumplo cualquiera de mis objetivos, le dedico esta Tesis a mis Padres y a las personas más especiales que han estado presentes en cada momento de mi vida, acompañándome por el complejo y emocionante camino de la vida.

Agradecimientos:

Llegar hasta este momento ha sido largo, no cabe duda de que he tomado varias decisiones; de las cuales algunas han sido acertadas y otras no tanto. Estas junto con el apoyo incondicional de mi familia, amigos, y personas especiales, me han permitido avanzar y en cada una de esas decisiones, mis padres siempre han estado presentes, a pesar de no estar siempre de acuerdo, me han apoyado de manera incondicional, y me han brindado las posibilidades óptimas para terminar mis estudios y seguir adelante en mi formación profesional.

Es por ello por lo que con mi profundo agradecimiento a mis Padres por estar siempre presente cuando cumplo cualquiera de mis objetivos, quienes han sido una guía y el camino para poder llegar a este punto de mi carrera. A Gilberto y Miguel mis queridos hermanos por ser mi apoyo incondicional.

“Muchas Gracias”

Índice general

Contenido

Índice de Ilustraciones

1 Introducción

1.1 Antecedentes

En la mayoría de las empresas privadas se requiere implementar un software especializado que les permita administrar sus propios recursos, tales como los sistemas ERP (Enterprise Resource Planning). La incorporación de un sistema ERP le dará una ventaja competitiva a una empresa ya que permitirá disponer de información para realizar una mejor toma de decisiones. Sin embargo, en los servicios públicos que se les brinda a la sociedad no ocurre lo mismo, no existe un control detallado de los bienes y la información no está al alcance de todos. Este proyecto contempla la especialización de un ERP para este caso [1].

Como cada organización tiene sus demandas particulares, este sistema brinda un servicio adaptado a los requisitos de cada organización, que no se trata de una solución uniforme e idéntica para todos los casos.

Los sistemas ERP como tales comienzan a usarse desde la década de los noventa, estos sistemas utilizan un tipo de software multi-módulo, lo que permite mejorar el rendimiento de los procesos internos de nuestro negocio. Los sistemas ERP integran las actividades y funciones de todos los departamentos de una organización [2] y la forma en que esto se estructura en el sistema de software también representa la forma en que se debe estructurar una organización.

Sin embargo, los sistemas ERP no son simples y directos, tienen una estructura compleja, lo que requiere la creación de nuevas oficinas y departamentos, y adecuarlos a la estructura ERP.

Por lo tanto, una empresa necesita un experto para trabajar con sus tomas de decisiones más importantes y tratar de simplificar la gestión de un sistema ERP en la medida de lo posible

1.2 Descripción del problema

El objetivo principal de este sistema es la integración de los diferentes departamentos, sobre todo en aquellos puntos en donde con anterioridad había un sistema de información concreta para cada parte de la organización. Los ERP tienen la capacidad de generar una base de datos limpia, donde la información se gestiona en tiempo real y se puede conseguir los datos necesarios en el momento deseado.

Los sistemas ERP se organizan a través de módulos, los cuales se enlazan a diferentes bases de datos, según lo que se exija para cada área, en una zona urbana se pueden encontrar áreas como; finanzas, en la que se gestiona el dinero que se destina a las calles, semáforos y parques; o, recursos humanos, en la que interviene el personal de limpieza, seguridad pública o privada, personal de reparación entre otros.

Otro servicio es el de las amenidades de las organizaciones, las cuales pueden ser instalaciones para actividades recreativas ubicadas dentro de cada cluster, y que tienen un costo por un determinado lapso de tiempo. Estas cuentan con un reglamento interno y algunas de ellas tienen la opción de ser apartadas en los módulos de este ERP.

Además, las distintas organizaciones pueden gestionar trámites necesarios para la realización de obras dentro de los clúster, y todos ellos se llevan a cabo físicamente en las instalaciones de la organización, previas con cita en algunos casos.

Las organizaciones también se encargan de mantener la integridad física de los clúster que administran, proporcionando a los condominios la opción de reportar desperfectos, y un seguimiento de estos para garantizar la reparación, en caso de corresponder a su administración.

El proceso administrativo, su registro y los reportes, se realizan en su mayoría por medio de formatos de hojas de cálculo, lo cual implica la dependencia absoluta del factor e interacción humana, con los riesgos de error en fórmulas, cálculos, borrado de información y tiempo que esto conlleva. A la vez que la búsqueda de información de procesos anteriores es más compleja.

1.3 Metodología de desarrollo

El proyecto que se planteó sigue el estilo de arquitectura de software REST (Representational State Transfer) [3]. REST define un conjunto de principios arquitectónicos por los cuales se diseñan servicios web incluyendo cómo se accede al estado de dichos recursos y cómo se transfieren por el protocolo HTTP (Hypertext Transfer Protocol) hacia clientes escritos en diversos lenguajes. Una implementación concreta de un servicio web REST sigue cuatro principios de diseño fundamentales [4]:

1) Utiliza los métodos HTTP de manera explícita: Hace que los desarrolladores usen los métodos HTTP explícitamente de manera que resulte consistente con la definición del protocolo. Este principio de diseño básico establece una asociación entre las operaciones de crear, leer, actualizar y borrar y los métodos HTTP.
    - POST para crear un recurso en el servidor
    - GET para obtener un recurso
    - PUT para cambiar el estado de un recurso o actualizarlo

- DELETE para eliminar un recurso

2) No mantiene estado: Una petición completa e independiente hace que el servidor no tenga que recuperar ninguna información de contexto o estado al procesar la petición. Un cliente de servicio web REST debe incluir, dentro del encabezado y del cuerpo HTTP de la petición, todos los parámetros, contexto y datos que necesita el servidor para generar la respuesta. De esta manera, el no mantener estado mejora el rendimiento de los servicios web y simplifica el diseño e implementación de los componentes del servidor, ya que la ausencia de estado en el servidor elimina la necesidad de sincronizar los datos de la sesión con una aplicación externa
3) Expone URIs con forma de directorios: Las URI de los servicios web REST deben ser intuitivas, hasta el punto de que sea fácil adivinarlas. Una forma de lograr este nivel de usabilidad es definir URIs con una estructura al estilo de los directorios. Este tipo de URIs es jerárquica, con una única ruta raíz, y va abriendo ramas a través de las subrutas para exponer las áreas principales del servicio [5].
4) Transfiere XML, JSON (JavaScript Object Notation), o ambos: Esto permite que el servicio sea utilizado por distintos clientes escritos en diferentes lenguajes, corriendo en diversas plataformas y dispositivos [6].

1.3.1 Modelo de diseño MVC

El Modelo Vista Controlador (MVC) es un modelo de diseño estándar con el que están familiarizados muchos desarrollos de la industria del software. Algunos tipos de aplicaciones salen beneficiadas con el marco de desarrollo del modelo vista controlador [7 y 8].

A continuación en la figura 1, se describe el flujo de procesamiento típico para peticiones HTTP con Spring MVC [9 y 10].

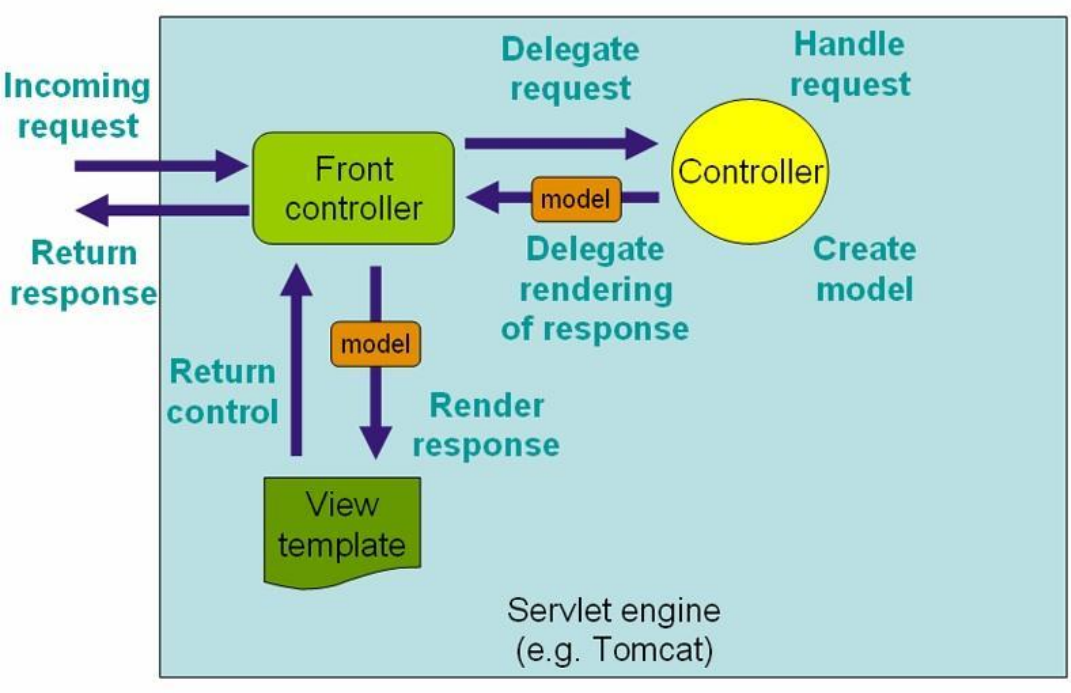

Fig. 1. Procesamiento para peticiones HTTP con Spring MVC

- Todas las peticiones HTTP se canalizan a través del Front Controller. En la mayoría de frameworks MVC que siguen este patrón no es más que un Servlet cuya implementación es propia del Frameworks. En Spring la clase se llama DispatcherServlet.
- El Front Controller busca, normalmente a partir de la URL, a qué Controller hay que llamar para servir la petición. Para esto se usa un HandlerMapping en Spring.
- Se llama Controller, que ejecuta la lógica de negocio, obtiene los resultados y los devuelve al Servlet, encapsulados en un objeto del tipo Modelo.
  Además, se devolverá el nombre lógico de la vista a mostrar.
- Un ViewResolver se encarga de averiguar el nombre físico de la vista que se corresponde con el nombre lógico del paso anterior.
- Finalmente, el front controller (en Spring es el DispatcherServlet) redirige la petición hacia la vista, que muestra los resultados de la operación realizada [12].

Administrar los recursos correctamente, poniendo al alcance toda la información de la gestión de los bienes públicos a la sociedad y hacer un correcto manejo de los mismos a través de un modelado de las diferentes áreas que contiene. El propósito fundamental de la aplicación de un ERP es otorgar apoyo a la organización con tiempos rápidos de respuesta a sus problemas, así como un eficiente manejo de la información que permite la toma oportuna de decisiones y disminución de los costos de operaciones, como objetivos en este proyecto fueron.

1. Implementar este sistema como paso inicial en una colonia de una ciudad o una zona urbana de una población inferior a 50,000 habitantes
2. Tener una base de datos centralizada
3. Tener un sistema global en vez de tener sistemas que se manejan por separado
4. Eliminación de costos y operaciones innecesarios de la organización
5. Control sobre la nómina de los empleados
6. Información automática para la fiscalización de estos bienes
7. Registro y publicación de eventos que puedan integrar a los usuarios de estos bienes
8. Automatizar las tareas que se realizan a mano y por sistemas que se manejan por separado
9. Dar información de manera rápida a los servidores públicos para la toma de decisiones

El resultado obtenido fue un sistema fácil de usar, escalable y global que abarque una gran parte de bienes que se puedan administrar con un ERP.

- Todo el proyecto es un servicio realizado mediante la arquitectura REST.
- Una base de datos centralizada que contenga la información que se genere con el uso de este sistema.
- Se dará el servicio a una aplicación web, o varias

- Se dará el servicio a dispositivos móviles que cuentan con la aplicación de cliente remoto para que pueda comunicarse con el servidor.

La implementación de un ERP en la organización ayudará a cubrir diferentes aspectos entre los cuales se destacan la confiabilidad en la información y reportes generados por el sistema, la rapidez en el proceso de información, la disminución de tiempos de entrega y gastos de operación, así como rapidez y mayor certeza para la toma de decisiones.

Debe ayudar a mejorar las prácticas de la organización y el control de sus recursos. Por otro lado, se logrará mejorar la productividad de los empleados públicos, dado que existe un plan más eficiente y realista de lo que hace y qué comprar y utilizar. Estas características permitirán manejar menores niveles de gasto público, y será más confiable a la hora de manejar los recursos de la organización a la que se le aplique un sistema de este tipo.

Además, la implantación de un sistema ERP permitirá beneficiarse de las posibilidades de gestión por internet, ya que los datos e información recogidos del sistema estarán disponibles y accesibles para cualquier empleado que necesite solicitar cualquier dato que necesite, y eso simplificará el trabajo de la organización.

Con el fin de mejorar el sistema de información en una empresa para poder realizar mejor sus servicios y operaciones, o si es el caso que requiere de un sistema diferente, producto del desarrollo de TI, es conveniente que la organización implemente un sistema ERP. Las ventajas de tal sistema son consideradas válidas por los representantes de la mayoría de las empresas, como la solución tecnológica básica para garantizar la transparencia, reducir costos y reducir los efectos del fraude u otros medios ilegales.

2 Marco Teórico.

2.1 Estándares en la Industria.

En el proceso administrativo, su registro y reportes, se realiza por medio de formatos de hoja de cálculo como en Excel en su mayoría, la dependencia absoluta del factor e interacción humano, con los riesgos de error en fórmulas cálculos, borrado de información de procesos anteriores es más compleja. Adicionalmente, los trámites ofrecidos al público deben ser realizados en forma presencial generando riesgos de inconformidad por falta de información y largas filas en ciertos días [12].

Este problema tiene una solución ofrecida con los ERP modernos que hay en el mercado y/o que desarrollan algunas empresas en requerimientos de los requisitos de la organización que los necesite, facilitando muchas de las tareas que antes se hacían con otras herramientas.

Por lo general, se hace referencia a un ERP como una categoría de software de administración de negocios, generalmente un conjunto de aplicaciones integradas, que una organización puede usar para recopilar, almacenar, administrar e interpretar datos de estas muchas actividades comerciales que realice.

En la década de 1990 los modelos empresariales habían evolucionado considerablemente, necesitando nuevas formas de gestión. El software ERP es el resultado de la evolución y adaptación de los sistemas MRP (Material Requirement Planning). Por primera vez en la historia del ERP, todas las áreas de la empresa se centralizan en una única solución, que realiza una gestión de forma íntegra. Más allá de ser simple planificador, un ERP es un contenedor de información valiosa para la organización que puede ayudar a crecer apoyando la forma de decisión. Gracias a

este concepto abierto de gestión modular, el ERP puede en este momento abrirse a múltiples perfiles de negocios [13, 14 y 15].

Los sistemas de ERP experimentaron una rápida expoliación en los años noventa. Debido al problema de Y2K (Problema informático del año 2000) [16] y a la introducción del euro en muchos países de Europa que interrumpió los sistemas heredados, muchas empresas aprovecharon la oportunidad para reemplazar sus sistemas antiguos con ERP [17].

Con la información de un sistema, los procesos son más automáticos, la información es más certera y accesible, se reducen tiempos de atención y el capital humano se puede invertir en procesos de mejora y atención a los usuarios de los servicios de las organizaciones como se muestra en la tabla 1.

Tabla 1. Componentes de un ERP

Administración u Operaciones	Contabilidad o Finanzas	Recursos Humanos o Nómina	Producción
Compras	Libro Mayor	Tipos de Contrato	MPR1
Inventarios	Activos Fijos	Tipos de Calendario	MPR2
Cuentas por pagar	Presupuestos	Múltiples modelos nómina	Planificación
Facturación o Ventas	Centros de costos	Historial Empleado	Costos
Cuentas por cobrar	Cuentas auxiliares	Vacaciones, reposos o permisos	Materia Prima
Caja y Bancos	Periodo fiscal	Tipos de Contrato	Mano de Obra

Los ERP se caracterizan por estar divididos en varios módulos que típicamente representan las áreas y procesos fundamentales de la organización como son la parte administrativa, operativa y financiera. Los módulos abarcan todos los departamentos y procesos más importantes de la organización.

Los sistemas de información son hoy en día una necesidad para procesar grandes volúmenes de datos que se requieren actualmente para la toma de decisiones de las empresas. Los sistemas ERP son sistemas integrales de gestión para la organización. Se caracterizan por estar compuestos por diferentes partes integradas en una única aplicación. Estas partes son de diferentes usos, por ejemplo, producción, ventas, compras, logística, contabilidad, nóminas etc.

2.2 Implementación de un ERP

La finalidad de la implementación del ERP es la gestión interactiva de todas sus funciones a través de la centralización y la consolidación de información estratégica [18]. Las razones que llevan a la organización a la implementación de un sistema de este tipo pueden ser las siguientes [19].

- Apoyarse en un sistema único y coherente
- Tener en cuenta las especificaciones de la organización
- Centralizar la información y facilitar su circulación
- Organizar los flujos de información internos y externos
- Disponer de funciones de coordinación y de análisis de las decisiones
- Tomar decisiones más rápidas gracias a los datos centralizados

En la mayoría de los casos en un corto o mediano plazo estos cambios, tienen como consecuencia un ahorro de costes. Hay que tener en cuenta que se deben respetar los principios fundamentales de cualquier proyecto [20 y 21].

1. Asegurarse de contar con el compromiso de la dirección, al ser el eje central de la organización y tener una visión global, la participación es muy valiosa e importante en el proceso.
2. Implementar un liderazgo integrador, involucra a todas las áreas y todos los niveles de la organización.
3. Asignar recursos al proyecto (económico, humano, de infraestructura).
4. Definición de una manera clara y funcional los procesos de la organización.
5. Determinación de los objetivos y metas a alcanzar con un proyecto de este tipo.
6. Asignación de un administrador del proyecto interno que se asegure de dar seguimiento a los avances y compromisos del proyecto.
7. Tomar conciencia de los cambios en las tareas y responsabilidades, perfiles de los puestos y procesos los cuales surgen de la innovación tecnológica aplicada en la organización.
8. Motivación del personal para que asistan y participen activamente en las sesiones de capacitación en el uso del ERP

2.2.1 Etapas de una implementación de un ERP

Un proyecto de la implementación de un ERP consta de varias etapas, cada una con su importancia individual para el éxito del proyecto como se puede ver en [22, 23, 24, 25 y 26].

- Planeación
- Análisis
- Diseño
- Pruebas
- Capacitación
- Liberación

2.2.1.1 Planeación del proyecto ERP

El objetivo de esta etapa es poner en contacto con el cliente del producto para que en conjunto se define la planeación definitiva y detallada del proyecto, entre las actividades a realizarse están [27]:

- Presentar el equipo de trabajo al cliente que llevará a cabo la implementación
- Establecer un cronograma de actividades y ajustar todo con la misma agenda de todos los involucrados en el proyecto
- Armar una bitácora de compromisos mandatorios para poder arrancar el proyecto sin posibles contratiempos
- Preparar los formatos de cada uno de los procesos de la implementación, así como los datos iniciales que se ingresaran al sistema

2.2.1.2 Análisis del proyecto ERP

En esta fase lo principal será obtener la lista de requerimientos y funcionalidades acordadas de cada una de las áreas de la organización para la implementación y así mismo dicho listado debe ser validado por cada responsable de área para proceder a la configuración de la solución de acuerdo con los procesos y procedimientos definidos [27].

Durante esta etapa se llevan a cabo las siguientes tareas:

- Se definen los alcances específicos del proyecto
- Se elaboran diagramas de flujo de los procesos y procedimientos para cada área operativa

2.2.1.3 Diseño del proyecto ERP

En esta etapa de la implementación del ERP se realiza la configuración de la solución de acuerdo con el proceso, procedimientos y requerimientos que en la etapa anterior se definieron. Es importante en esta etapa que la persona a cargo del proyecto mantenga contacto con el cliente para que se validen cualquier duda que se puedan surgir ya que en esta etapa la solución va tomando forma para quedar adecuada a las necesidades de la organización [28].

2.1.1.4 Capacitación para utilizar el sistema

Una vez realizadas las pruebas y mejoras, se procede a capacitar a todos los usuarios finales, aquí la motivación y participación de todos los involucrados es muy importante ya que es cuando va a tener la experiencia real con el ERP y sobre todo se preparan para dominar el uso de la solución. La comunicación total con el consultor es muy relevante en este paso, se deben dar a conocer las dudas en cuanto a procesos y funciones, esto con el fin de obtener el máximo provecho de la

herramienta [29]. Al finalizar la capacitación, se deberá asegurar de encargar a cada usuario que realice operaciones diarias de práctica en el sistema, desde ese día de práctica en el sistema, desde ese día hasta la liberación [30].

Al final, un equipo de empleados y consultores trabajó durante un par de semanas para implementar los procesos y funciones del sistema ERP y mejorar su eficacia general. Los resultados de la implementación del ERP son una mejora en la eficiencia operativa y una reducción de costos para la empresa, el nuevo sistema también le brinda a la empresa la oportunidad de brindar un mejor servicio a sus clientes. Después de la implementación del ERP, cada empleado de la empresa ahora puede consultar la información en la web.

2.3 Requerimientos del sistema

En [31] se puede observar la etapa fundamental en un proyecto de ingeniería de software es la identificación y documentación de los requerimientos del sistema al comienzo del proyecto. Esta etapa se refiere a la identificación y documentación de los requerimientos de un sistema, a partir de los usuarios, clientes o interesados. A la práctica también se conoce como recopilación de requerimientos. Para obtener los requerimientos de software se realizan los siguientes pasos [32].

1. Análisis de documentación

- Consiste en obtener la información sobre los requerimientos funcionales y requerimientos no funcionales de software a partir de documentos que ya están elaborados
- Utiliza la documentación que sea relevante al requerimiento que se está levantando

2. Observación

- Consiste en observar y estudiar el entorno de trabajo de los usuarios, clientes e interesados en el sistema

3. Entrevistas

- Conforme a las entrevistas que se realizan a los usuarios finales del sistema se hacen cambios o mejoras al sistema
- Las preguntas abiertas son útiles para identificar información faltante
- Las preguntas cerradas son útiles para confirmar y validar información

2.4 Conceptos de la arquitectura

Una parte esencial para el mantenimiento y posterior desarrollo del ERP es la creación de una arquitectura diseñada de forma modular. De esta forma se da una facilidad para poder agregar nuevos módulos que fuesen posibles conforme la organización va evolucionando en sus diferentes áreas. La forma modular entiende que es la organización es un conjunto de departamentos que se encuentran relacionados uno con otro y que se genera a partir de sus procesos [33] como se muestra en la figura 2.

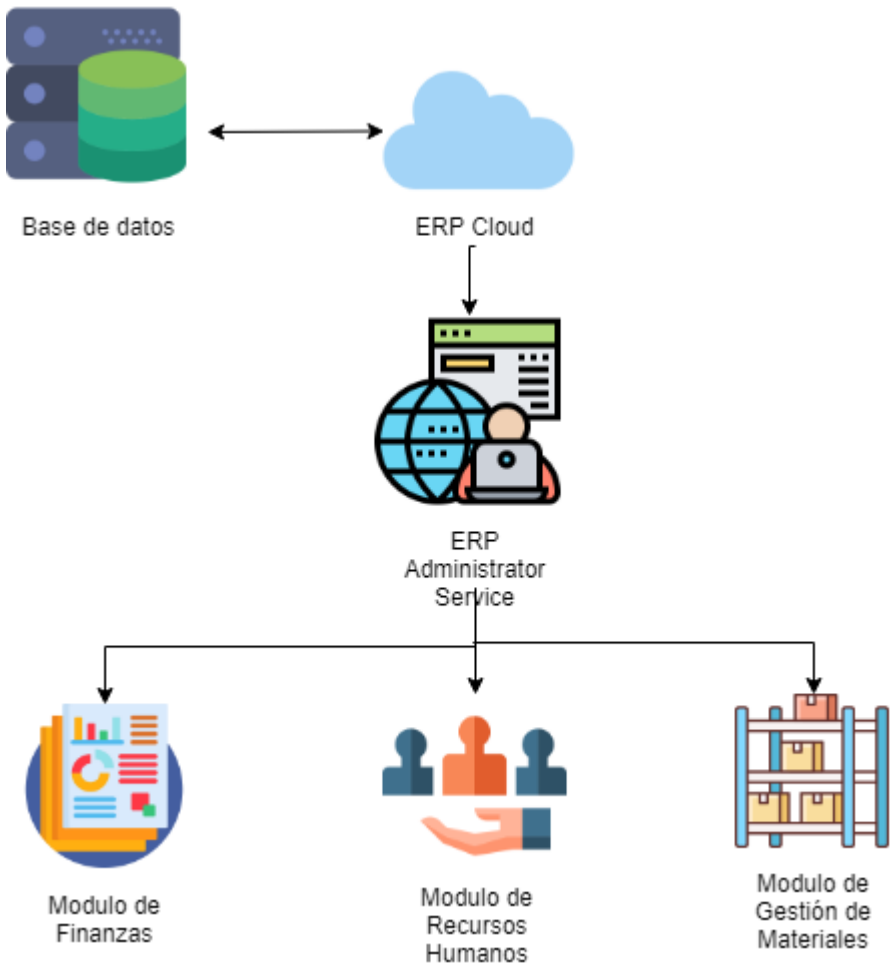

Fig. 2. Arquitectura para sistemas ERP

El sistema está compuesto de dos elementos principales: la base de datos y una arquitectura tipo cliente-servidor, esta última consiste en la implementación de un servidor que entrega servicios a un conjunto de usuarios. La información utilizará una arquitectura tipo REST, lo que facilitará en el futuro vincular de más formas los servicios que se ofrezcan por la organización ya sea con una App de otro sistema operativo móvil, u ofreciendo una API (Application Programming Interface) a terceros.

Un usuario administrador tendrá la posibilidad de poder interactuar de una forma controlada con la base de datos, tendrá la posibilidad de añadir, actualizar y

eliminar registros, la eliminación se hará de una forma lógica, de esta forma se mantienen los datos guardados y no se muestran para los usuarios de la plataforma.

2.4.1 Operaciones fundamentales del sistema

Entre las funciones del ERP [34 y 46] se encuentran estas operaciones fundamentales:

- Procesar las transacciones que se producen en todos los departamentos de la organización.
- Tiene un papel fundamental en la medición de resultados de la organización al disponer de toda la información de todas las transacciones de la organización.
- Dar a conocer acontecimientos o actividades de la organización.
- El sistema debe de ser modular para que responda en las modificaciones conforme a las necesidades futuras de la organización.
- Permitir recolectar la información de diferentes ubicaciones, procesarlas y ofrecer a los distintos departamentos y usuarios.
- Recibir soporte de algún departamento de desarrollo.
- Facilidad para poder ser utilizado por los distintos usuarios de las diferentes áreas.
- El sistema debe de estar basado en una sola base de datos que permita integridad, consistencia e integración de éstos.

2.5 Estructura de carpetas del proyecto

La estructura del proyecto seguirá una estructura por módulos en donde se incluyen las librerías en las cuales su única función será hacer uso de la base de datos mediante los métodos CRUD. Los módulos encargados de utilizar estas librerías en el

proyecto son AdminClusterERP, AppAdmnistrator, CondominioService [36, 37 y 38] como se muestra en la figura 3.

```
AdminClusterERP
    src
        main
            java
                com.project
                    erp
                        components
                        configuration
                        controller
                        pojo
                        service
                        tools
                    ERPAplication.java
            resources
                static
                templates
                application.properties
                banner.txt
        test
            java
                com.project.erp
                    TestERP.java
    target
    uploads
    pom.xml
AppAdministrator
    src
        main
            java
                com.project
                    admin
                        configuration
                        controller
                        excelView
                        pojo
                        service
                        tools
                    AppAdministratorApplication.java
            resources
                static
                    assets
                    index.html
                templates
                    datos
                        *.html
                    error
                        *.html
                    Fragments
                        *.html
                    Tablas
                        *.html
                    *.html
                application.properties
        test
            java
```

```
                com.project.erp
                    TestERP.java
        target
        pom.xml
    CondominioDB
        src
            main
                java
                    com.project
                        customtypes
                        entity
                        page.configuration
                        repository
        target
        pom.xml
    CondominioService
        src
            main
                java
                    com.project
                        controllers
                        security
                        admin
                        tools
                        CondominioServiceApplication.java
                resources
                    application.properties
                    banner.txt
            test
                java
                    com.project
                        CondominioApplicationTest.java
        target
        uploads
        pom.xml
    ErpDB
        src
            main
                java
                    com.project
                        customtypes
                        entity
                        page.configuration
                        repository
        target
        pom.xml
    Testing
        src
            main
                java
                    App.java
                    TransaccionesTest.java
        target
        pom.xml
    Uploads
        2019-05-05
        2019-05-06
        Correo
        Reglamentos
        Servicios
```

Fig. 3. Estructura de Carpetas

Es bien sabido que para las empresas en el mundo globalizado de hoy estar bajo presión permanentemente para ser exitosas y sobrevivir a sus respectivos negocios, tienen que competir a escala mundial, adaptarse a innovaciones tecnológicas y diferentes paradigmas los cuales cambian bastante rápido y deben estar en cumplimiento con un creciente número de leyes, regulaciones y normas internacionales. Las cuales deben ser lo más rentables posibles en cuestión de costos. Como la seguridad es un prerrequisito esencial, forma parte integral de las actividades del negocio y es mucho más importante desde el punto de vista del negocio, esta situación es mucho más compleja para los proveedores de servicios TIC, por que adicionalmente ellos deben cumplir con cada requerimiento específico del cliente con la suficiente garantía.

En las empresas, la organización de la seguridad usualmente se establece la responsabilidad en los altos mandos para asegurar que la seguridad se establezca y tenga el nivel apropiado para cumplir los requerimientos internos y externos con los cuales la empresa es representada. La organización de seguridad aplicará los estándares ya existentes, así como guías y ambientes de trabajos o frameworks, pero, ¿Cuáles deben ser usadas?

La elección puede ser influenciada por el enfoque tomado. Muchas organizaciones de seguridad se enfocan en la implementación de la solución (Nut-and-bolts approach) debido a que el personal tiene más afinidad tecnocrática, otros se enfocan más en directivas (Directive-and-guidance-only approach) debido a que el personal tiene más afinidad por la concepción y la metodología, sin embargo, hay vacíos o al menos debilidades con respecto a la orientación bien-equilibrada de las organizaciones de seguridad [39 y 40]. La gestión de la información y tecnología relacionada en una empresa incluye entre otra información los requerimientos de

seguridad. COBIT [41], está mencionado como un ejemplo. ISACA (Information Systems Audit and Control Association) ha desarrollado COBIT 5 para proveer a las empresas con un Frameworks completo que los ayude a manejar y gestionar la información, así como la tecnología relacionada de forma holística, incluyendo la seguridad. COBIT se utiliza para “Controlar los objetivos de la información y la tecnología relacionada”. COBIT 5 está construido sobre 5 principios y 7 facilitadores. SOX [42] (Sarbanes-Oxley Act) comprende lo legal y Gubernamental, así como los requerimientos específicos de la industria y las obligaciones que son en su mayoría de carácter general. SOX, tiene por objeto garantizar la correcta administración a través de un control de evaluación, incluyendo a la empresa y auditorías independientes. Esta directriz también cubre problemas como el control corporativo y una mayor transparencia financiera, existen leyes similares en otros países. KonTrag, por ejemplo, es una ley alemana de control corporativo y transparencia que requiere de un consejo ejecutivo para implementar un sistema de control de riesgos que provea una detección temprana de alertas, las cuales tienen que ser reportadas, existen varias leyes, regulaciones y políticas de diferentes fuentes que pueden ser consideradas y tienen una influencia en las TIC y control de seguridad de una empresa, no obstante, esta situación parece un poco confusa para la parte técnica y usualmente es requerido el asesoramiento legal.

La categoría de gestión de TI, se refiere al establecimiento de la gestión de servicios de TIC en una empresa la cual requiere servicios de seguridad, entre otros. ITIL [43] se refiere a “IT infrastructure Library,” ITIL V3 es un Frameworks de las mejores prácticas que describe la posible implementación de la gestión de servicios de TI en una empresa, ITIL recomienda que los servicios deben estar alineados a las necesidades del negocio y apuntalar los procesos del núcleo del negocio, ITIL se usa en conjunción con uno o más frameworks de las mejores prácticas como COBIT, ISO/IEC 27000-Series y TOGAF.

Las mejores prácticas de ITIL se detallan con 5 publicaciones principales que cubren el ciclo de vida completo de los servicios empezando con la identificación de las necesidades del cliente y conduce los requerimientos de TI, el diseño y la implementación de los servicios de producción y finalmente , el monitoreo y la mejora de la fase de servicio, cada publicación cubre una fase del ciclo de vida del servicio y describe los principios clave, los procesos clave y actividades, y los roles y responsabilidades clave en cada fase. ISO/IEC 27001 [44], es una prominente representación, la cual trata con los establecimientos del control de seguridad en una empresa, ISO (International Standardization Organization) e IEC (International Electronical Commission) han desarrollado ISO/IEC 27001, más precisamente la serie de estándares ISO/IEC 27000, ISO/IEC 27001 provee un modelo para el establecimiento, implementación, monitoreo, revisión, mantenimiento y mejora en un sistema de gestión de la seguridad de la información (ISMS – Information Security Management System), el diseño e implementación del ISMS de una empresa depende de situaciones y requerimientos específicos, es por ello que puede variar significativamente entre empresas. ISO/IEC 27001 sigue un enfoque de procesos adoptando el modelo “Plan do Check Act” (PDCA) con sus 4 fases:

1) Establecer el plan ISMS (Plan)
2) Implementar y operar ISM (Do).
3) Monitorear y revisar ISMS (Review)
4) Mantener y mejorar ISMS (Act).

Con el establecimiento de ISMS, la empresa debe identificar, analizar y evaluar los riesgos a los que se expone, entonces identificar y evaluar las opciones para la corrección de los riesgos y finalmente seleccionar los objetivos de control y controles para el tratamiento de dichos riesgos, ISO/IEC 27001 consiste de 8 títulos:

1) Alcance

2) Referencia normativas
3) Definiciones y términos
4) Sistema de control de seguridad en la información
5) Responsabilidad del control
6) Auditorias Internas ISMS
7) Control de revisión del ISMS
8) Mejora del ISMS

El Estándar describe el control de los sistemas que es requerido para alcanzar el nivel apropiado de seguridad. Este estándar no provee información acerca de las medidas específicas de seguridad que permitan el nivel apropiado de seguridad que debería alcanzarse, sin embargo, el anexo A provee 39 controles objetivos y 133 controles que deberían ser considerados. El estándar enfatiza que se debe adecuar si se requiere. La lista es una suma del ISO/IEC 72002 [45].

El “information security handbook”, es una guía para gerentes, provista por el NIST (National Institute of Standards and Technology) que muestra que la gestión de seguridad ISMS, por una parte, y los controles de seguridad (implementación), por otra, pueden ser tomados por separado, Este a su vez provee una lista de las mejores prácticas de los controles de seguridad que se describe en otra publicación especial de NIST (SP 800 Series). La evaluación de seguridad del cómputo en la nube es un documento de la ENISA (European Network and Information Security Agency) la cual provee un despliegue de nuevos modelos que requieren nuevos frameworks para ser aplicados, ENISA provee más guías, hay varias iniciativas las cuales también tratan con la nube y su seguridad, el cual se denomina, ISF (Information Security Forum) y el CSA (Cloud Security Alliance), La documentación disponible no está correctamente alineada una con otra, incluyendo , ISO/IEC 27001 y ISO/IEC 27002, ISO/IEC 27017 (Seguridad en la nube), ISO/IEC 27018 (Privacidad en la nube) no pueden ser evaluados aún. La categoría representada por TOGAF se trata con el establecimiento

de una arquitectura como un importante requisito para la gestión efectiva de la seguridad y la información. Un framework de arquitectura empresarial generalmente reconocida que incluye e integra aspectos de seguridad en el TOGAF [46], (The Open Group Architecture Framework) provee un completo framework de referencia para el diseño, la planeación, la implementación y el control de una empresa de información, este es un completo enfoque de alto nivel para diseñar el modelado típicamente en cuatro niveles: negocios, aplicación, datos y tecnología. Esta usa su estandarización, modularidad, así como los ya existentes productos y tecnologías provistas el TOGAF 9.1 consiste de 9 partes:

PARTE I Introducción
PARTE II Método de desarrollo de arquitectura (ADM)
PARTE III Lineamientos y técnicas (ADM)
PARTE IV Framework de contenido de arquitectura
PARTE V Herramientas y continuación de la empresa
PARTE VI Modelos de referencia TOGAF
PARTE VII Framework de compatibilidad de la Arquitect

El núcleo de TOGAF en sí es su método de desarrollo de la arquitectura (ADM) descrita en la parte II, esta metodología es conducida por un conjunto de principios y comprende un proceso de 8 pasos para el desarrollo y mantenimiento de la arquitectura de una empresa. Las consideraciones de seguridad que necesitan ser tomadas en cuenta durante la aplicación de TOGAF – Método de desarrollo de la arquitectura (ADM), por ejemplo en cada una de sus fases, están descritas por separado en la parte III, que también se refiera a él “White Paper Guide to Security Architecture” en TOGAF, ADM es una vista útil en la arquitecturas de seguridad, TOGAF es una poderosa herramienta para el desarrollo de la arquitectura de una empresa, ESARIS [47] trata de proveer un enfoque práctico que se centre en la seguridad de la información y asegure que los problemas de seguridad no se oculten

entre todas las medidas de calidad. Existen otros enfoques de arquitectura como los son el SABSA (Sherwood Applied Business Security Architecture), y el OSA (Open Security Architecture). SABSA es un modelo para el desarrollo de las arquitecturas de seguridad que parece estar ligada a la educación de negocios y certificaciones, OSA tiene un enfoque real de herencia y estipula una variedad de controles de seguridad adicionales. OSA, comprende en un catálogo de control, patrones y catálogo de tratamiento, este enfoque arquitectónico tiene como objetivo evitar el aislamiento de temas de interés y organizar los problemas de seguridad que correspondan a la infraestructura de TI. En la práctica, la herencia es más bien plana y los controles están basados en los mismos que NIST. OSA recolecta información útil pero no es adaptable a ningún modelo de negocio. El estándar internacional ISO/IEC 27002 [48], fue desarrollado con base en BS7799, el cual fue elaborado por la industria y publicado a mediados de los 90, el estándar británico BS7799 se convirtió en el ISO/IEC 17799, el cual ha sido revisado varias veces y eventualmente se convirtió en la serie ISO/IEC 27000, en el cual se describen 133 controles de seguridad asignados a 39 controles que están organizados en 11 cláusulas.

Clásulas de la serie 27000:

- Análisis y tratamiento de riesgos.
- Políticas de Seguridad.
- Organización de la información de seguridad.
- Controles de activos.
- Seguridad de recursos humanos.
- Seguridad física y ambiental.
- Gestión de comunicación y las operaciones.
- Control de accesos.
- Adquisición de sistemas de información y desarrollo y mantenimiento.
- Gestión de incidencia de seguridad.
- Gestión de la continuidad del negocio.

- Conformidad.

Este estándar provee un gran control de la información, pero el enfoque no considera el negocio de las empresas, así como los actuales servicios de TIC 's. Así se asume que la infraestructura de un usuario típico en una organización no podría encajar con el proveedor de servicios de TIC 's.

El ISF ha preparado el SOGP [49], (Standard of Good Practices), que aborda la seguridad de la información desde una perspectiva de negocios empresariales y provee las bases para acceder y mejorar la seguridad de la información en las organizaciones. Siendo más detallada, actual y exhaustiva que el ISO/IEC 27002, el cual puede usarse para posibilitar el cumplimiento con el ISO/IEC 27001.

El SOGP se distingue entre otros controles de seguridad fundamentales y especializados, los temas fundamentales son generalmente aplicados por todas las organizaciones para darle forma a los fundamentos en su seguridad de la información. Los temas especializados dependen del negocio y no son relevantes para cada entorno.

La mayoría de los conceptos provistos para el análisis de riesgos son como un prerrequisito para la implementación de medidas de seguridad. El BSI (Bureau of Security Information) en Alemania tomó un enfoque diferente. La IT-Grundschutz catalogues [50] estipula controles de seguridad técnicos, organizacionales, personales e infraestructurales que han sido probados para alcanzar los requerimientos “normales” de protección. Estos pueden ser usados para posibilitar el cumplimiento del ISO/IEC 27001. IT-Grundschudz Catalogues están basados en 5 Módulos, 5 categorías y 6 catálogos de resguardo. La ventaja de la IT-Grundschudtz catalogue, es al mismo tiempo su defecto, el concepto es sencillo de aplicar en un ambiente típico de TI pero mayormente muy inflexible para ser aplicado en grandes

instalaciones y negocios complejos. Muchas de las organizaciones internacionales proveen recomendaciones, estándares y guías de los aspectos más importantes y relevantes de la seguridad de la información. Una remarcable fuente es NIST, los cuales proveen una amplia recomendación de gestión, normas y directrices administrativas, técnicas y físicas para el uso público con la publicación especial de la serie 800.

Además de requerimientos y recomendaciones que han sido liberadas por las autoridades federales, como es el caso de Alemania y Estados Unidos, variando de país a país, así como las regulaciones de privacidad de la información, los proveedores de TIC, tienen que implementar las normas de seguridad y generar 36 reportes de calidad para demostrar que se cumple de acuerdo a la norma ISAE3402 [51], además de otras directivas.

Debido a que los requerimientos de seguridad han incrementado en años recientes, las tecnologías de la información y la comunicación precisan una significante reducción en los costos de seguridad, mientras que el desempeño y la flexibilidad de la infraestructura deben mejorar al mismo tiempo que la transparencia en cuestión de seguridad.

Una de las soluciones a todos estos requerimientos se fundamenta en la modernización de las TIC, en la cual se ha replanteado el paradigma de seguridad para poder definir la idea de “seguridad industrializada” y adaptar la seguridad de las TIC a un método de producción industrializada para las TIC. Una de las investigaciones más amplias en este campo ha sido por parte de la empresa T-Systems GMBH, la cual ha sentado las bases para la correcta organización y documentación de seguridad en las empresas y a su vez ha liberado la metodología y taxonomía de ESARIS [8] (Enterprise Security Architecture for Reliable ICT Services),

basada en todos los estándares mencionados anteriormente la cual ha sido muy exitosa para la misma.

2.6 Patrón de diseño de controlador de vistas del modelo

El patrón MVC [52] es un modelo de diseño estándar con el que están familiarizados muchos desarrolladores. Algunos tipos de aplicaciones web salen beneficiados con el marco de MVC. Otras seguirán usando el modelo de la aplicación ASP.NET tradicional que está basado en formularios Web Forms y devoluciones. Otros tipos de aplicaciones web combinarán las dos estrategias; una no excluye a la otra.

El marco de MVC incluye los componentes siguientes mostrados en la figura 4:

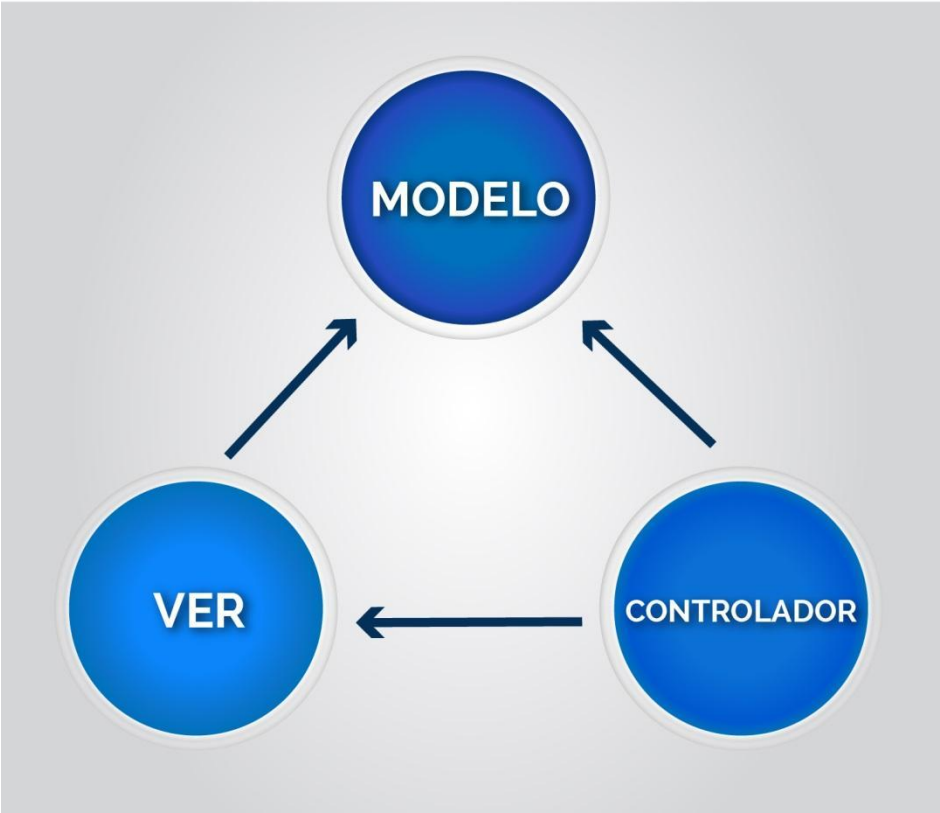

Fig. 4. Componentes del modelo MVC

Modelos. Los objetos de modelo son las partes de la aplicación que implementan la lógica del dominio de datos de la aplicación. A menudo, los objetos de modelo recuperan y almacenan el estado del modelo en una base de datos. Por ejemplo, un objeto Product podría recuperar información de una base de datos, trabajar con ella y, a continuación, escribir la información actualizada en una tabla Productos de una base de datos de SQL Server.

En las aplicaciones pequeñas, el modelo es a menudo una separación conceptual en lugar de física. Por ejemplo, si la aplicación solo lee un conjunto de datos y lo envía a la vista, la aplicación no tiene un nivel de modelo físico ni las clases asociadas. En ese caso, el conjunto de datos asume el rol de un objeto de modelo.

Vistas. Las vistas son los componentes que muestra la interfaz de usuario de la aplicación. Normalmente, esta interfaz de usuario se crea a partir de los datos del

modelo. Un ejemplo sería una vista de edición de una tabla Productos que muestra cuadros de texto, listas desplegables y casillas basándose en el estado actual de un objeto Product.

Controladores. Los controladores son los componentes que controlan la interacción del usuario, trabajan con el modelo y por último selecciona una vista para representar la interfaz de usuario. En una aplicación MVC, la vista solo muestra información; el controlador administra y responde a los datos proporcionados por el usuario y su interacción. Por ejemplo, el controlador administra los valores de la cadena de consulta y pasa estos valores al modelo, que a su vez podría usarlos para consultar la base de datos.

El modelo MVC le ayuda a crear aplicaciones que separan los diferentes aspectos de la aplicación (lógica de entrada, lógica de negocios y lógica de la interfaz de usuario), a la vez que proporciona un vago acoplamiento entre estos elementos. El modelo especifica dónde se debería encontrar cada tipo de lógica en la aplicación. La lógica de la interfaz de usuario pertenece a la vista. La lógica de entrada pertenece al controlador. La lógica de negocios pertenece al modelo.

Esta separación le ayuda a administrar la complejidad al compilar una aplicación, ya que le permite centrarse en cada momento en un único aspecto de la implementación. Por ejemplo, se puede centrar en la vista sin estar condicionado por la lógica de negocios.

El acoplamiento vago entre los tres componentes principales de una aplicación MVC también favorece el desarrollo paralelo. Por ejemplo, un desarrollador de software puede trabajar en la vista, un segundo desarrollador puede ocuparse de la lógica del controlador y un tercero se puede centrar en la lógica de negocios del modelo.

Además de administrar la complejidad, el modelo MVC hace que sea más fácil probar las aplicaciones que probar una aplicación web Spring MVC basada en formularios Web Forms. Por ejemplo, en una aplicación web Spring MVC basada en formularios Web Forms, se usa una clase única para mostrar la salida y para responder a los datos proporcionados por el usuario. A diferencia de tecnologías de Java como JSP, en Spring MVC utilizamos Thymeleaf que es el que Pivotal recomienda utilizar.

Escribir pruebas automatizadas para las aplicaciones Spring MVC basadas en formularios Web Forms puede ser complejo, ya que para probar una página individual se deben crear instancias de la clase de página, todos sus controles secundarios y las clases dependientes adicionales de la aplicación. Dado que se crean instancias de tantas clases para ejecutar la página, puede ser difícil escribir pruebas que se centren exclusivamente en partes individuales de la aplicación. Las pruebas para las aplicaciones Spring MVC basadas en formularios Web Forms pueden ser por consiguiente más difíciles de implementar que las pruebas de una aplicación MVC. Es más, las pruebas en una aplicación Spring MVC basada en formularios Web Forms requieren un servidor web. El marco de MVC desacopla los componentes y hace un uso intensivo de las interfaces, lo cual hace posible probar los componentes individuales aislados del resto del marco.

3. Información del dominio del problema

Los servicios que ofrece una organización en su mayoría actualmente ofrecen un abanico de servicios de administración de conjuntos habitacionales, ofrecidos a través de un ayuntamiento, oficina, o departamento que se ubican dentro del área urbana.

3.1 Introducción al dominio del problema

Uno de los servicios que puede ofrecer un área urbana como una zona residencial es el control de acceso peatonal y vehicular dentro de la misma. Para ello se hace uso de un sistema de chips, que permiten identificar a un habitante o vehículo, y pudiendo así permitirle el acceso o negárselo. Para obtener una forma de este acceso, los habitantes de la zona urbana requieren ir de forma presencial a las oficinas en donde se encuentren.

Otro servicio es el de amenidades, las cuales son instalaciones para actividades recreativas ubicadas dentro de los clúster, y que tiene un costo por lapso de tiempo. Estas cuentan con reglamentos internos y algunas de ellas tienen la opción de ser apartadas en módulos de las zonas urbanas.

Entre las actividades que hacen las oficinas que se encargan de administrar la zona urbana, también se encarga de gestionar trámites necesarios para la realización

de obras dentro de los clúster, y todos ellos se llevan a cabo físicamente en las instalaciones de la zona urbana, algunas requieren una cita previa.

Las oficinas administrativas de la zona urbana también tienen como actividad mantener la integridad física de los clúster que administra, proporcionando a los condominios la opción de reportar desperfectos, y un seguimiento de estos para garantizar la reparación, en caso de corresponder a su administración.

3.2 Disponibilidad de servicios

La aplicación permitirá a los usuarios acceder al siguiente catálogo de servicios. También se incluirá la posibilidad de vincular los servicios de pago de algunos proveedores (como Paypal) y la posibilidad de pagar los servicios que ofrece la organización.

- La aplicación móvil para Android y iOS podrá realizar:
    - Registro para uso de la aplicación.
    - Agendar citas para solicitudes de chips vehiculares o peatonales.
    - Información, apartado y pago para uso de amenidades.
    - Directorio con información de los servicios ofrecidos por las oficinas administrativas.
    - Directorio con información del personal de las oficinas administrativas
    - Acceso a información sobre trámites de obra, y programación de citas para realizar trámites.
    - Realizar un reporte de seguridad, mantenimiento o convivencia.
    - Recibir información sobre la comunidad, comunicados, recomendaciones y promociones.

- Sistema para administrador de clúster que contendrá:

    - Módulo de cobranza que se encargará de:
        - Recepción de pagos.
        - Ejecución de pagos.
        - Dashboard.
        - Historial de pagos.
        - Estados de cuenta por lote.
        - Reporte de cierre de mes.
    - Módulo de inventario:
        - Reporte de inventario.
        - Salidas de inventario.
        - Recepción de inventario.
        - Asignación de ubicación.
        - Ajuste de inventario.
    - Módulo caja chica
        - Registro de movimiento.
        - Balance de caja chica.

- Sistema para administrar la zona urbana que tendrá los siguientes módulos:
    - Módulo de compras que podrá realizar las siguientes actividades:
        - Requisiciones de compra y servicio.
        - Orden de compra (pedido).
        - Entrada por orden de compra.
        - Facturación de proveedor.
        - Autorizaciones.
    - Módulo de cobranza que se encargará de:
        - Recepción de pagos.
        - Ejecución de pagos.
        - Dashboard.
        - Historial de pagos.

        - Estados de cuenta por lote.
        - Reporte de cierre de mes.
    - Módulo de inventarios en el cual se realizarán las siguientes transacciones:
        - Dar alta a un material.
        - Movimientos de inventarios.
        - Traspasos.
        - Ajustes masivos.
    - Módulo de Envió de correos:
        - Avisos.
        - Confirmaciones.
    - Agenda de citas
        - Agendar citas.
        - Re-asignar citas.
        - Cancelar cita.
    - Reporte de incidencias
        - Asignar estado al reporte.
    - Módulo presupuestal
        - Capturar ingresos.
    - Caja chica
        - Registro de movimiento.
        - Balance de caja chica.
    - Business Partners
    - Dashboard de Clúster
    - Reglamentos
    - Amenidades
    - Usuarios
    - Servicios

4 Necesidades del Negocio

El objetivo principal de un sistema a desarrollar se centra en el alineamiento de las necesidades de negocio con el sistema, con las posibilidades técnicas reales. Una fase muy importante, por ende, es la identificación de las necesidades del negocio. En este capítulo se especifican las necesidades de la zona urbana que requiere satisfacer para proporcionar un excelente servicio a los residentes que vivan en ella y de igual forma facilitar el trabajo a las oficinas administrativas haciéndolo más ágil y sencillo.

4.1 Alta de usuarios

Se requiere que el ERP tenga un sistema de autenticación de usuarios, esto para llevar un control de las personas que tendrán acceso a los servicios ofrecidos (Véase Fig. 5).

Los niveles de acceso a la información dependerá del tipo de usuario, el cual puede ser: supervisor, administrativo, propietario, residente o proveedor de servicios.

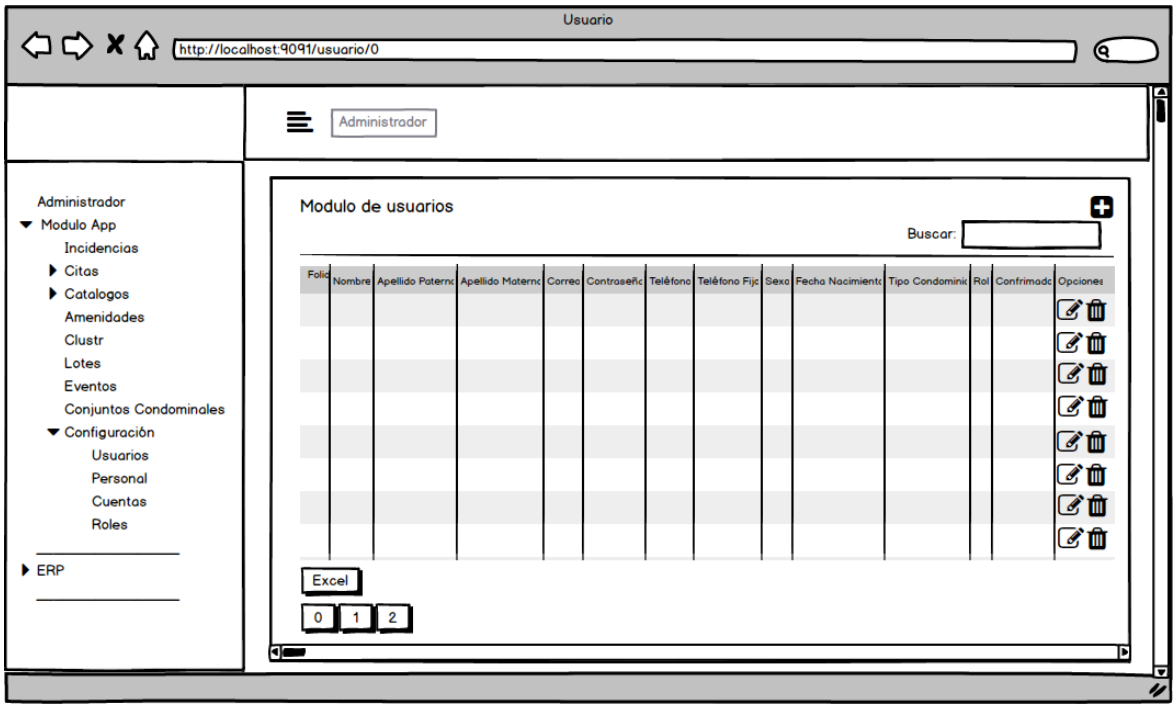

Fig. 5. Módulo de usuarios

4.2 Administración y manejo de Amenidades

Algunas zonas urbanas requieren que los condominios puedan obtener información acerca del uso y disponibilidad de las amenidades; así como ofrecer un mecanismo más práctico para el pago de aquellas que tienen costo especial.

Implementación de un mecanismo adicional para el acceso a las amenidades, el cual identifique a los usuarios que han apartado algunas de ellas para uso particular.

La información presente en este módulo requería algunas veces que los usuarios tengan que exportarla, por lo tanto, se deja un acceso en la API del sistema para que se permita descargar desde el panel del administrador.

Los documentos asociados a cada fila de información también estarán disponibles para descargar desde el panel del administrador, este podrá también eliminarlos y actualizarlos cuando este lo requiera (Véase Fig. 6.).

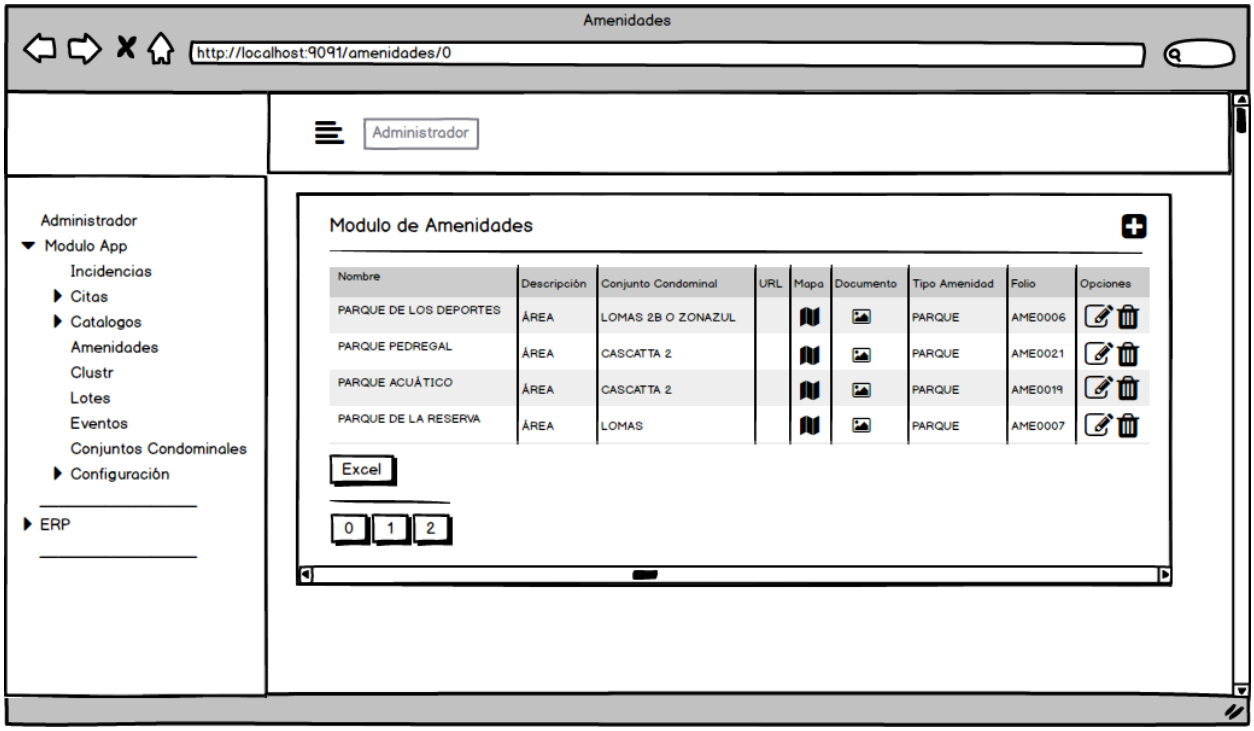

Fig. 6. Módulo de Amenidades

4.3 Trámite de obras

Las oficinas de las zonas urbanas necesitan agilizar el proceso de gestión de obras dentro de la misma zona. Por tanto, necesita un mecanismo que facilite el acceso a la información de trámites y requisitos para ejecución y liberación de obras; así como un sistema que gestione la asignación de citas con el personal correspondiente.

La información presente en esta sección tendrá una conexión con el sistema ERP para que pueda notificar a los usuarios que levantaron algún trámite desde sus dispositivos móviles y enviárselos en el caso de que tengan una cuenta vinculada con la App de la organización (Véase Fig. 7.).

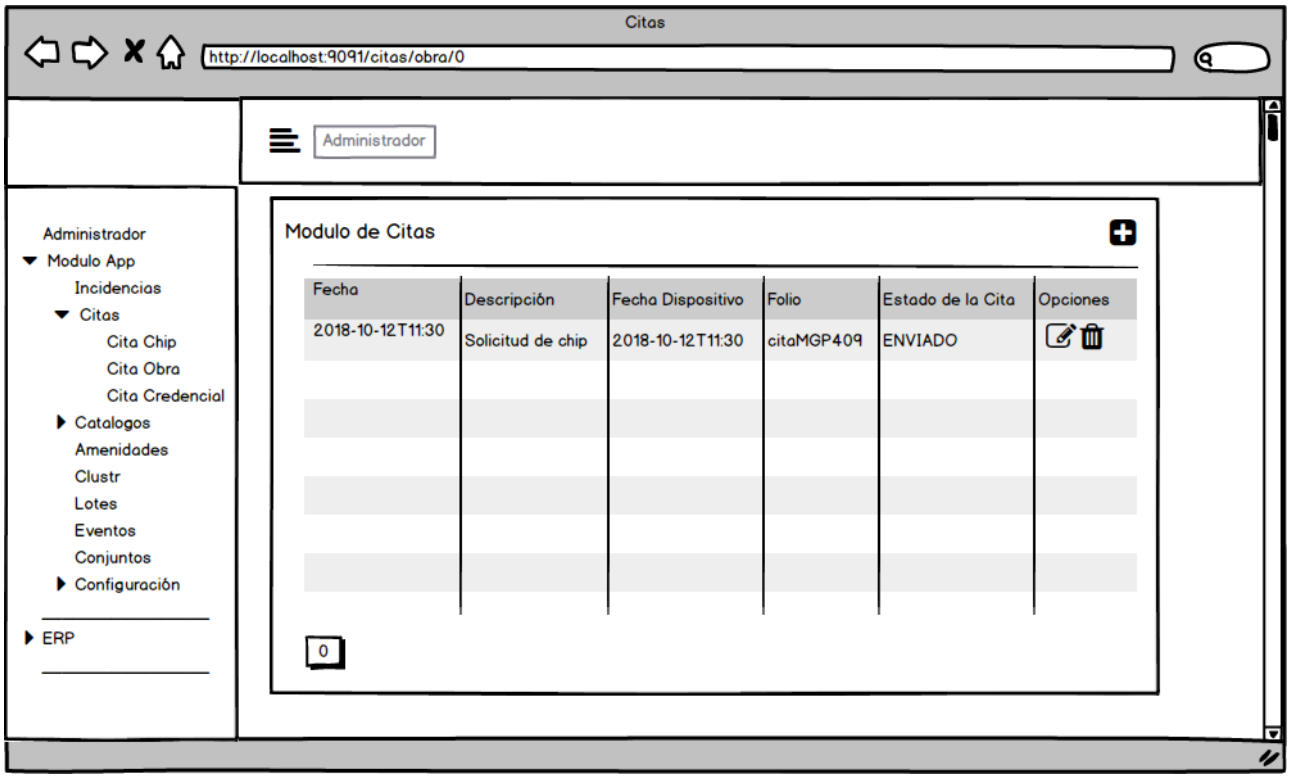

Fig. 7. Módulo de Citas

4.4 Información acerca de los servicios

Una zona urbana tiene la necesidad de presentar a los condominios un catálogo con los servicios que ofrece, por tanto, requiere un sistema sencillo, pero eficaz que describa el alcance de sus servicios.

En la aplicación se mostrarán los diferentes requisitos para el trámite de obras y asignación de citas.

En este módulo servirá para dar de alta los servicios ofrecidos por la organización con una breve descripción para que aparezcan en la App de los usuarios con una cuenta vinculada al servicio de la organización (Véase Fig. 8).

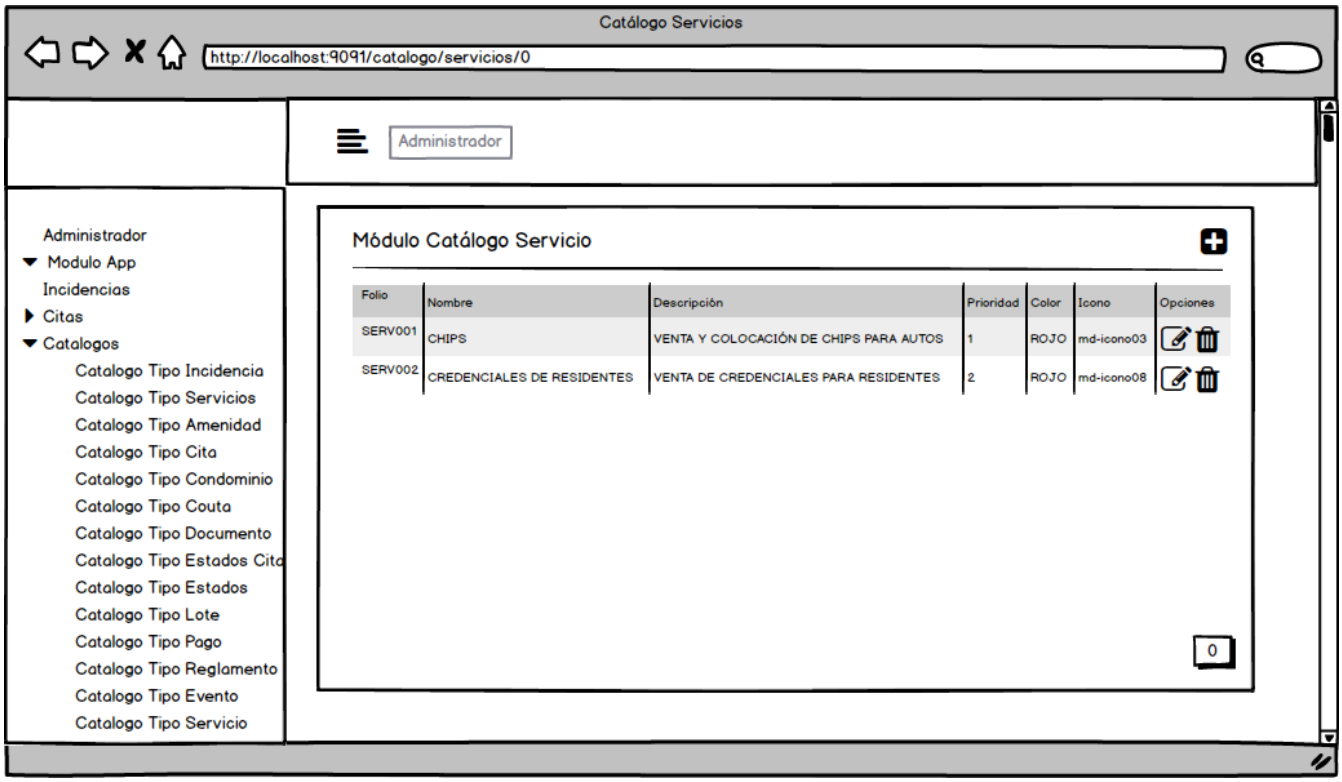

Fig. 8. Módulo de Catálogo de Servicios

4.5 Reportes de estado de instalaciones

Una zona urbana necesita un mecanismo eficiente que permita a los usuarios de los condominios realizar reportes sobre imperfectos o daños en las instalaciones, e

informar a estos el estado de las reparaciones reportadas. De la misma forma realizar reportes sobre eventos de seguridad o convivencia.

Por lo tanto, se dedicará un módulo de reportes y seguimiento de anomalías. El encargado de revisión de incidencias podrá ver los reportes realizados por los condominios a través de la App.

Una vez leído el incidente, el encargado podrá asignar el reporte a su correspondiente responsable, cerrar el reporte cuando este ya se haya solucionado, declinar algún reporte o derivarlo a alguna otra autoridad competente.

El residente que levantó el reporte recibirá información y evidencia cuando su solicitud haya sido resuelta a través de una notificación Push a su dispositivo móvil en donde haya iniciado sesión.

Se requiere almacenar la información que el usuario de la App suba al realizar un reporte de las instalaciones. Esto incluye los siguiente:

- Ubicación del incidente por GPS del celular
- Fotografía en formato digital del incidente
- Comentarios en formato de texto

También se almacena información sobre el estado de los reportes levantados, esto solamente es un texto que puede ser alguno de los especificados en la sección de reglas de negocio correspondiente. La información de la fotografía se enviará al ERP y quedará almacenada para posteriores aclaraciones.

El usuario del ERP debe categorizar los incidentes y cambiar el estado del reporte, siendo los posibles estados los siguientes:

- Abierto: Se ha recibido el reporte, pero no se ha categorizado por un usuario del ERP.
- ODT Asignado: Se ha asignado una ODT (Orden de Trabajo) a un miembro del personal.
- Cerrado: Se ha completado la ODT y se adjuntan fotos y mensajes sobre el incidente por parte del usuario del ERP.
- Derivado: No corresponde a la administración de la organización, sin embargo, se ha derivado el reporte a la autoridad correspondiente.

Cuando el estado de un incidente se encuentra derivado el seguimiento del incidente queda fuera del alcance de esta App.

La App deberá informar al condominio que levantó el reporte de incidente sobre cualquier cambio de estado que tenga ese incidente.

4.6 Módulo de cobranza

Se necesitará un módulo que lleve el control de las diferentes transacciones financieras, por lo tanto, se creará un sistema que ofrezca los siguientes módulos.

4.6.1 Submódulos

- Recepción de pagos
- Ejecución de pagos
- Dashboard
- Historial de pagos
- Estados de cuenta por lote

- Reporte de cierre de mes

Para almacenar la información se requería que el usuario administrador defina los catálogos para la cobranza como se expresa en la figura 9

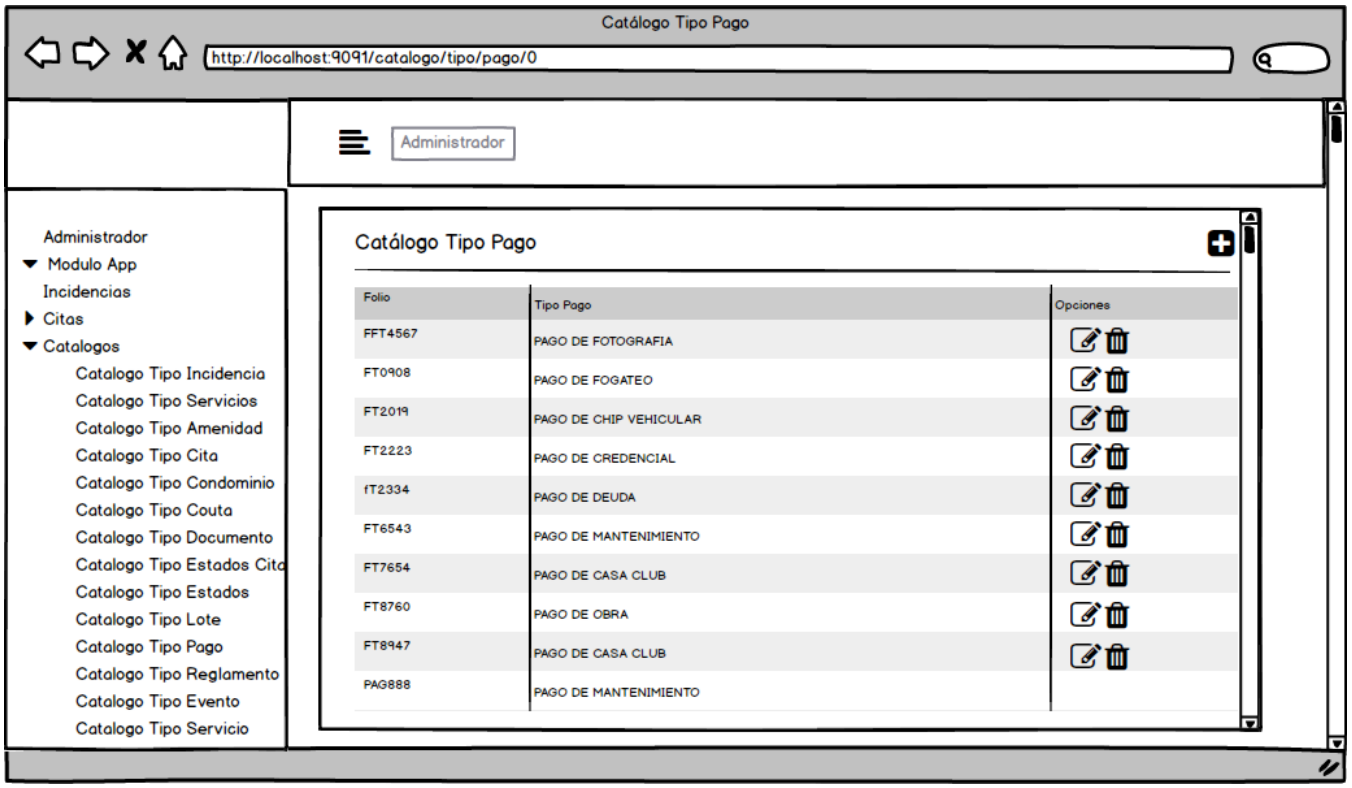

Fig. 9. Módulo de Catalogo Tipo Pago

4.7 Módulo de inventarios

El sistema podrá controlar los inventarios, para esto se desarrollarán los siguientes módulos que facilitarán el manejo y control de los artículos utilizados por el clúster.

El módulo para el control de inventarios contera submódulos que facilitaran el manejo y control de los artículos utilizados por los diferentes clústeres, además de que en este módulo se podrá dar de alta un material, traspasos y ajustes masivos (Entiéndase como ajustar un lote de datos en usa sola petición por parte del usuario administrador)

4.7.1 Submódulos

- Reporte de inventario
- Salida de inventario
- Recepción de inventario
- Asignación de ubicación
- Ajuste de inventario

4.8 Módulo de caja chica

Se desarrollará un módulo de caja chica que permitirá registrar ingresos diversos por conceptos que no son los administrados en el módulo de cobranza. Permitirá tener una administración eficiente de diversos conceptos que no son los administrados en el módulo de cobranza y está compuesto de las siguientes interfaces propuestas en los submódulos.

4.8.1 Submódulos

- Registro de movimiento
- Balance de caja chica

4.9 Módulo de compras

La zona urbana necesita un control de compras, desde la revisión de catálogos de artículos con poco stock, hasta su compra e inserción en el almacén. Las compras deben ser autorizadas por 2 niveles de supervisión. Por esa razón se ofrecen los siguientes submódulos.

4.9.1 Submódulos

- Requisición de compra y servicio.
- Orden de compra (pedido).
- Entrada por orden de compra.
- Factura de proveedor.
- Autorizaciones.

4.10 Módulo de envío de correos

Se desarrolla un módulo para envió de correos masivos o unitarios que permitirá mantener informados a los condóminos a cerca de avisos o confirmaciones Las estadísticas de entrega e interacciones del usuario con el correo se consideran como un servicio adicional no contemplado en los alcances del presente contrato.

4.11 Módulo de agenda de citas

El encargado de revisión de incidencias de las oficinas administrativas de la zona urbana podrá ver las citas que falten y en base a su criterio el podrá: aceptar, reagendar o cancelar citas (Véase Fig. 10).

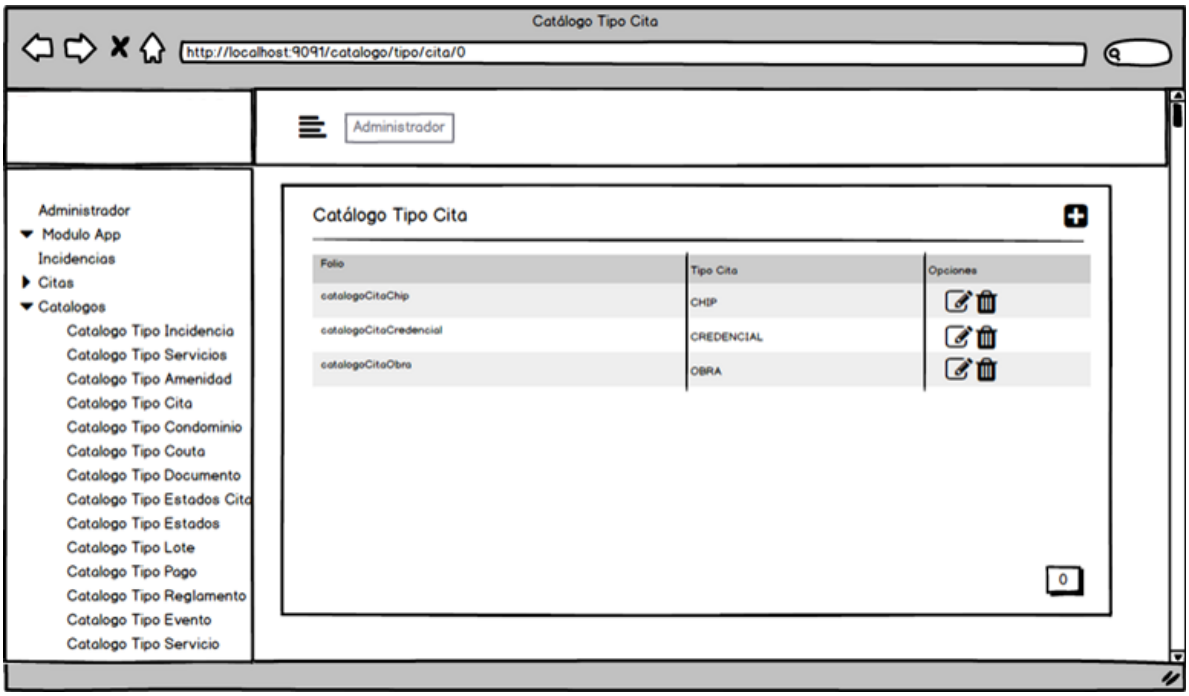

Fig. 10. Módulo de Catalogo Tipo Cita

4.12 Módulo de reglamentos

El usuario administrador podrá ver y modificar los reglamentos que actualmente la zona urbana posea en sus diferentes divisiones tales como reglamentos de clúster (Véase Fig. 10), amenidades, trámites de obra entre otros. Esto con la finalidad que los consumidores conozcan las reglas que se tienen en cada amenidad y posibles sanciones en caso de haberlas. La información por lo general se presenta en forma de documentos descargables en formato PDF. La información una vez subida al ERP será enviado a los dispositivos con la app a través de JSON, convertidos en base64 con la finalidad de seguir el uso de la arquitectura REST.

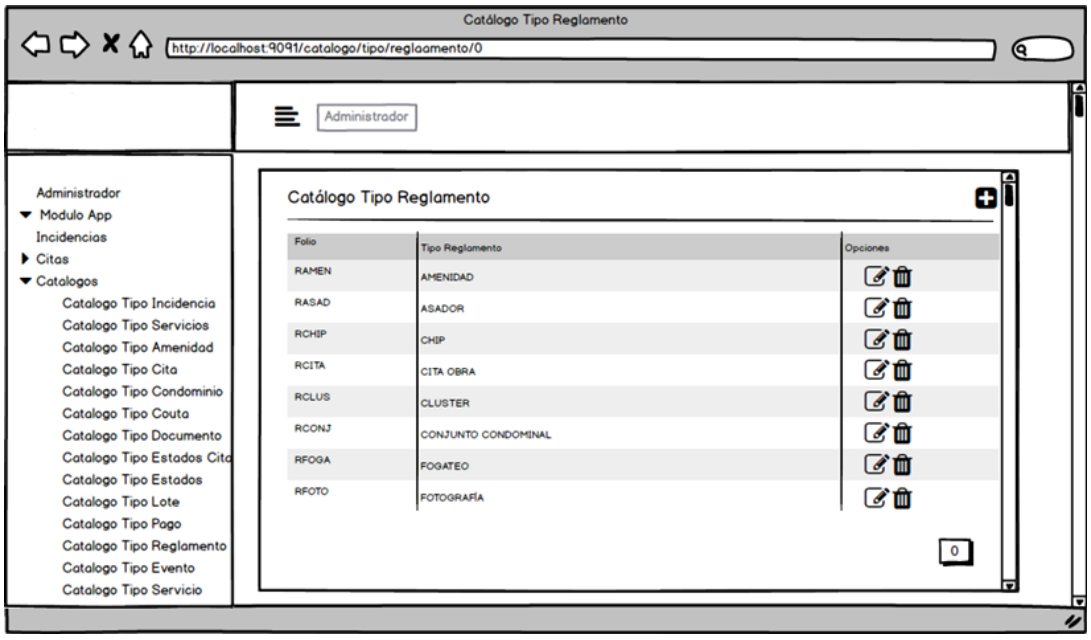

Fig. 11. Módulo de Catálogo Tipo Reglamento

5 Pruebas y resultados

5.1 App para los Condominios

Este subsistema es una App para Smartphones con sistemas operativos como Android y iOS que permite a los condominios tener interacción con las oficinas administrativas de la zona urbana sin necesidad de asistir físicamente a ellas como se muestra en la figura 12.

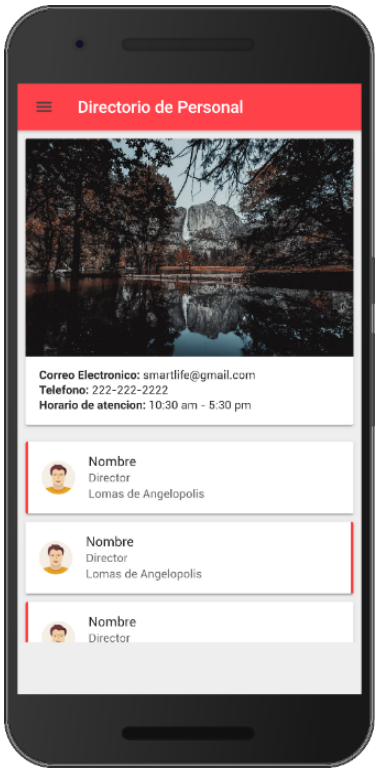

Fig. 12. Directorio Personal en App

La importancia de esto actualmente radica en la captura de documentos y de información, que consume mucho tiempo y no se tiene el suficiente personal asignado a esta tarea, por lo cual el proceso llega a ser muy tardado para los condominios.

5.2 Subsistema de administrador de clúster

Este subsistema se encarga de la administración y cobranza de los clústeres administrados por la zona urbana.

La importancia de esto es que en los clústeres administrados por las oficinas de la zona urbana tienden a tener problemas con la recaudación de cuotas de mantenimiento por casa de los lotes de los que administran, por lo cual llegan a mantener una gran cartera de clientes morosos. Actualmente se tienen que llenar varios documentos en hojas de cálculo para obtener datos sobre el presupuesto asignado por áreas, esta tarea consume mucho tiempo de personal administrativo de la zona urbana y puede resolverse con el desarrollo del ERP, logrando así una mayor flexibilidad a la hora de obtener información y generar reportes.

5.3 Requisitos específicos del subsistema

5.3.1 Casos de uso del subsistema

A continuación, se muestra un diagrama de casos de uso para el sistema descrito para especificar la comunicación y el comportamiento del sistema mediante la interacción con los usuarios y/u otros sistemas en la figura 13.

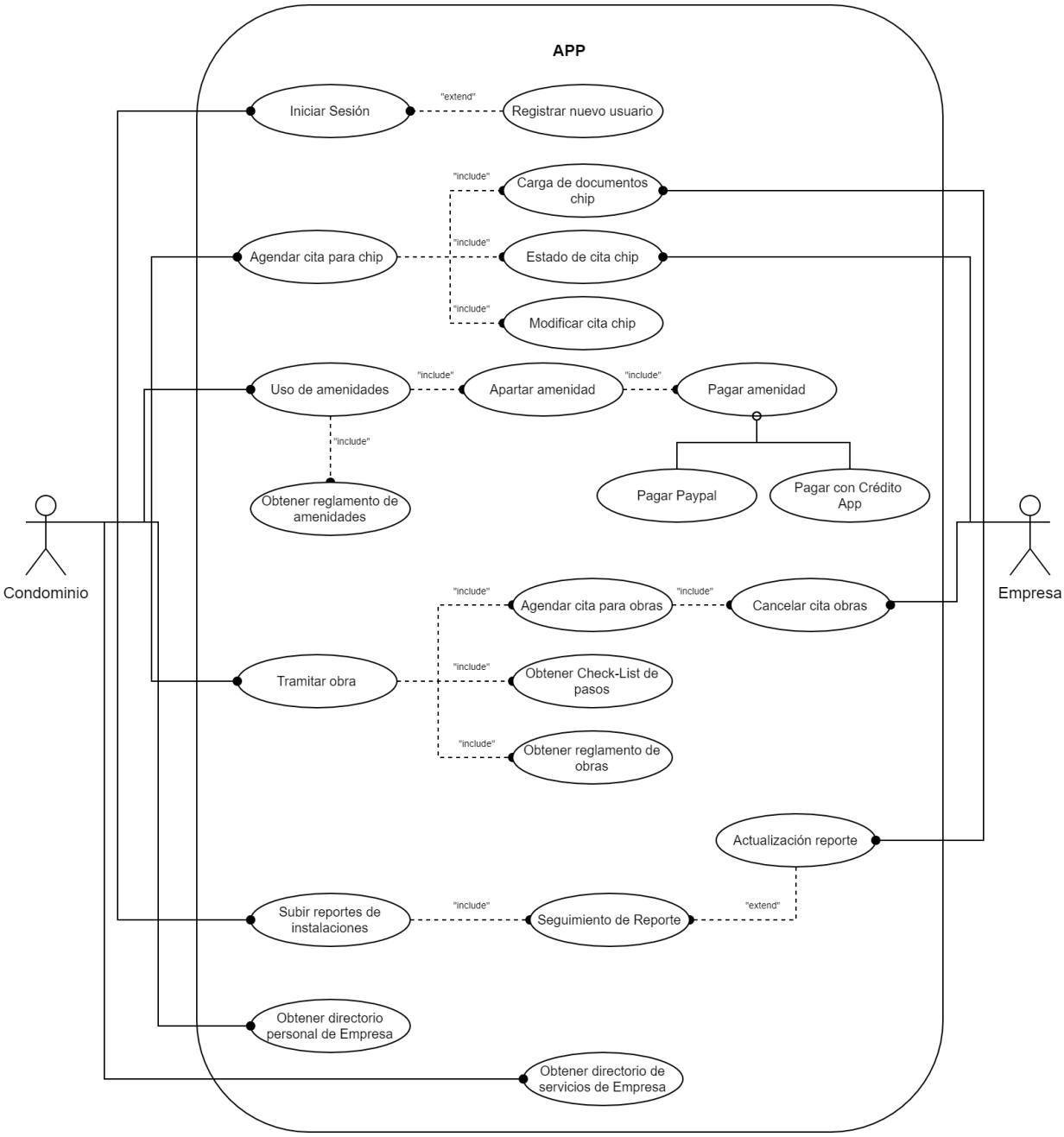

Fig. 13. Diagrama de casos de uso en App

5.4 Especificación de Actores

5.4.1 Condominio

- Condominio perteneciente a clúster administrado por las oficinas administrativas de la zona urbana.
- Trabajador de obra en clúster administrado por la zona urbana.
- El usuario de la aplicación puede solicitar alguno de los servicios ofrecidos por la zona urbana.

5.4.2 Oficinas administrativas

- Encargado de revisión de incidencias en la zona urbana.
- Encargado de chips y controles de acceso.
- Encargado de obras en la zona urbana.

5.5 Especificaciones de casos de uso del subsistema

5.5.1 Registro

El usuario de la aplicación que puede solicitar alguno de los servicios ofrecidos por la zona urbana a través de la app, actualmente en algunas zonas, los usuarios deben acudir a las oficinas administrativas o algún departamento, para realizar algún trámite, u obtener información sobre estos y es el personal administrativo que se encarga de recibir las peticiones e información de los usuarios.

Para poder realizar esto el usuario de la app debe tenerla instalada en su dispositivo móvil. En caso de que el usuario esté registrado en la app tiene la opción de registrarse proporcionando su información personal a través de un formulario. Una

vez registrado, los datos del usuario son almacenados en la base de datos, y entonces podrá iniciar sesión en la app y disponer de todos los servicios que la organización ofrezca a sus usuarios.

5.5.2 Agendar cita para Chip

Como precondición el condominio debe estar registrado en la App con un usuario activo. El proceso de agendar una cita para el chip crea un registro en la base de datos del ERP para que un usuario del ERP pueda atender a la persona que solicita el chip y poder otorgarlo. El usuario podrá realizar una de las tres siguientes opciones:

- Carga de documentos
- Modificar cita
- Estado de cita

Una vez realizada alguna cita, ésta quedará registrada en la base de datos, mostrando la información al usuario del ERP y la información con los detalles de la cita al usuario de la aplicación. Estos documentos serán subidos por el condominio mediante la app en formato de imagen digital, y almacenados en el mismo formato.
La app enviará la imagen del incidente por medio de la conversión base64 en un JSON, la imagen pasará del lado del servidor el cual volverá la imagen en su formato original almacenando con el formato originales el que este vino, solo se guardará en la base de datos el path de esta, permitiendo que las consultas de la base de datos sean más rápidas, ya que solo se almacenará el path y no toda la imagen.

En el caso de chip vehicular se requieren los siguientes documentos, siendo necesarios sólo uno de los tres primeros

- Comprobante de domicilio
- Escritura del inmueble
- Contrato de arrendatario

- Tarjeta de circulación
- Identificación oficial (INE, IFE, Pasaporte, Cédula Profesional, Cartilla Militar)

Además, el sistema almacena la información relacionada a la cita creada, es decir, la fecha y hora de inicio y fin, y un número de folio relacionado.

5.5.2.1 Carga de documentos

Para agendar una cita los usuarios deberán subir a la APP los documentos especificados en los requerimientos funcionales, de esta forma el personal de administrativo de la zona urbana podrá hacer una comprobación de la documentación previa. Los documentos viajaran a través de documentos tipo JSON, este formato solo envía texto por lo cual todos los documentos serán transformados antes de enviarse en base 64 una vez sean recibidos por el sistema ya sea el ERP o la App este se encargará de volverlo a convertir en su formato original.

5.5.2.2 Modificar cita chip

El condominio debe estar registrado en la app con un usuario activo, y con una cita para el chip asignado. El usuario podrá cambiar la fecha de la cita ya agendada, así como volver a cargar documentos en caso de que estos no hayan sido aprobados por la oficina administrativa encargada de esta tarea.

5.5.2.3 Estado de cita chip

El condominio debe estar registrado en la APP con un usuario activo y con una cita para chip asignado. En este apartado el usuario podrá verificar si los documentos subidos para la cita ya han sido pre-procesados por las oficinas administrativas o si necesita volver a cargar algún documento en caso de no haber sido aprobado

después de esto el usuario tendrá una cita asignada, e información sobre el estado de la cita.

5.5.3 Uso de amenidades

Previamente el condominio debe tener registro en la base de datos con las amenidades que ofrece. En el apartado de amenidades aparecerá una lista desplegable con todas las amenidades agrupadas por conjunto y clúster, de las cuales al seleccionar uno aparecerá una breve descripción. Así el usuario tendrá conocimiento de las amenidades disponibles.

El sistema almacena la información relacionada a las amenidades, incluyendo:

- La dirección dentro de la organización
- Reglamento Interno

La información se guarda del apartado de una amenidad incluye lo siguiente:

- Usuario que apartó la amenidad
- Horario de apartado
- Código único para apertura de amenidad
- Información de pago

La información de pago puede ser almacenada para el condominio, puede realizar posibles pagos posteriores. Esta información incluye, en caso de pago con tarjeta, lo siguiente:

- Número de tarjeta
- Fecha de vencimiento de tarjeta
- Código de seguridad

En caso de pago con PayPal, se almacena la información básica del usuario que este servicio proporciona para realizar un cobro.

5.5.3.1 Obtener reglamento de amenidades

El reglamento debe estar registrado en la base de datos, debe de estar colocado por los usuarios del ERP, cada amenidad posee un reglamento el cual se le mostrará al usuario de la APP. Los reglamentos podrán ser cambiados cada vez que los usuarios del ERP lo deseen. Los cambios se verán reflejados de igual forma en la App una vez que el usuario de esta misma vuelva a entrar.

5.5.3.2 Pagar amenidad

El condómino debe estar registrado en la base de datos con un usuario activo, y haber apartado una amenidad para un periodo determinado por el usuario que la apartó. El pago de la reserva puede darse en pago con tarjeta de crédito o de débito o a través de PayPal y OpenPay. El sistema registrará el apartado de la amenidad con una fecha y hora de uso de inicio y fin.

5.5.4 Tramitar obra

La opción de trámite de obra ayuda a las personas que están construyendo en la zona urbana a agilizar los trámites necesarios haciendo uso de los siguientes submenús

- Agendar cita para obras
- Obtener check-list de pasos
- Obtener reglamento de obras

El sistema proporcionará información sobre los trámites de la obra

5.5.4.1 Agendar cita para obras

El usuario podrá agendar una cita para aclarar sus dudas con los encargados de obra de las oficinas de la zona urbana, el intervalo en minutos por cita es configurable por el administrador del sistema, estas mismas serán mostradas al usuario de la App para que pueda escogerlas. El sistema registra la cita creada con una fecha y hora de uso de inicio y fin.

5.5.4.2 Obtener check list de pasos

Se muestra un check list con los pasos necesarios para completar el registro para agendar la cita de obra, esto depende de cada clúster. Una vez hecho esto el sistema proporcionará información sobre los trámites de obra y será enviado al ERP para que un administrador cambie el estado.

5.5.4.3 Obtener reglamento de obras

El usuario podrá descargar todos los reglamentos que se tengan para realizar obra por clúster, esto depende de la información que el usuario del ERP haya subido a la base de datos. El sistema proporcionará información sobre los trámites de obra.

5.5.4.4 Cancelación de citas para obras.

La cita de obras puede ser cancelada por ambas partes, tanto por el condominio, como por las oficinas administrativas, todos esto a través del ERP, de esta manera se le enviará una notificación y un correo al usuario de la App que la solicitud explicándole lo ocurrido.

El mensaje donde se le explicará lo ocurrido lo colocará el usuario administrador, este será a discreción del mismo personal de la organización, no habrá plantillas personalizadas ni autocompletado del mismo.

5.5.5 Subir reporte de instalaciones

El usuario puede reportar a través de la App reportes sobre las instalaciones y otras incidencias dentro de la zona urbana (Véase Fig. 14), tomando como parámetro de entrada los siguientes datos.

- Ubicación de incidente por GPS del celular
- Fotografía
- Comentario

El registro del reporte se almacenará en la base de datos del ERP, de esta manera las oficinas administrativas podrán dar seguimiento al reporte.

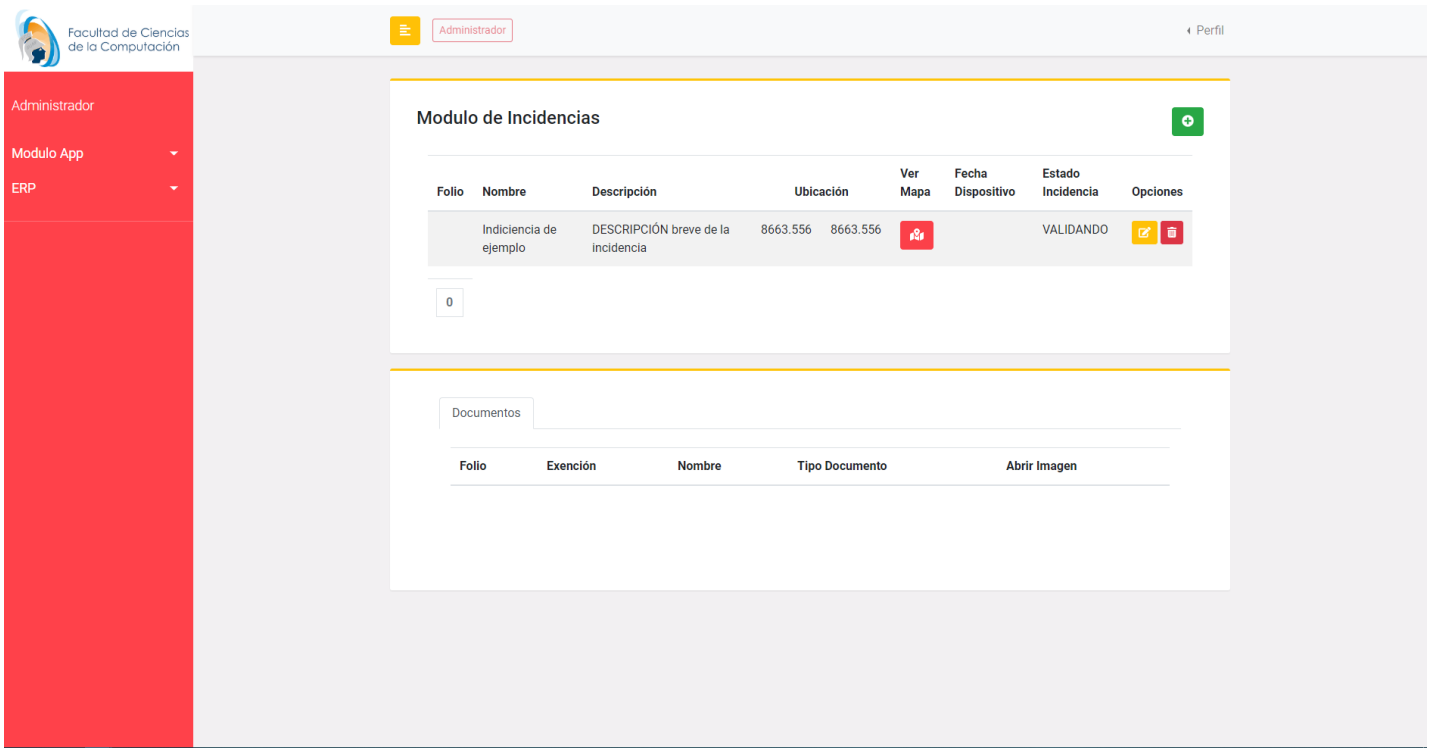

Fig. 14. Resultado Módulo de Incidencias

5.5.5.1 Seguimiento de reporte

El condominio debe estar registrado en la App con un usuario activo, y haber levantado un reporte. Los incidentes podrán verse reflejados en el ERP con el usuario que tenga los permisos necesarios para ver el módulo de reportes de incidentes, a su vez, así como el estado de actualización de este. El condominio revisa el estado del reporte levantado desde el ERP y actualiza el estado desde ahí. El usuario que levantó el incidente recibirá una notificación push cada vez que sea actualizado el estado de este desde el ERP

5.5.5.2 Actualización de reporte

Las oficinas administrativas deben categorizar los incidentes y cambiar el estado del reporte, siendo los posibles estados, detallados en los requisitos funcionales los siguientes:

- Abierto
- Asignado
- Cerrado
- Derivado

5.5.6 Obtener directorio de personal de las oficinas administrativas

El directorio de personal es una lista con el personal a cargo de los servicios de las oficinas administrativas, y es configurable a través de un usuario con privilegios de administrador en el sistema. El directorio contendrá nombre del personal, teléfonos del personal, dirección y horarios de oficinas, y correos electrónicos, siendo opcional poner a disposición los números personales de cada uno del personal.

5.5.7 Obtener directorio de servicios de la organización

El directorio de servicios es una lista con los servicios que ofrece la organización, y es configurable a través de un usuario del sistema.

5.6 Requisitos funcionales del sistema

5.6.1 Ingreso y registro de usuario

El sistema almacena la información correspondiente a los usuarios registrados, en concreto:

- Nombre – Texto
- Sexo – Opción múltiple
- Fecha de nacimiento – Fecha
- Teléfono – Texto
- Correo – Texto
- Arrendatario/Propietario – Opción múltiple
- Conjunto – Lista desplegable
- Clúster – Lista desplegable
- Lote – Texto
- Edificio/Casa – Opción múltiple
- Número – en casa de edificios

Cuando se registra un usuario el sistema enviará al usuario registrado un correo de confirmación con un enlace de activación, al no entrar al enlace de activación en un plazo de 24 horas el usuario será dado de baja por completo del sistema a no ser que tenga en curso alguno de los siguientes trámites.

- Agendar cita para control de acceso
- Agendar cita para tramites de obra

El usuario de igual forma podrá ser validado desde el ERP por un administrador sin la necesidad de entrar al vínculo del correo electrónico, cambiando el estado de aceptación del usuario.

Los usuarios podrán ver los menús de reglamentos y estado de cuenta solo si pertenecen a los clústeres administrados por la organización. Esta información será variable si el administrador del ERP cambia los reglamentos y añade más tipo de conceptos de pagos desde el sistema

5.6.2 Reporte de instalaciones

El usuario del ERP de la organización debe categorizar los incidentes y cambiar el estado del reporte, siendo los posibles estados los siguientes:

- Abierto: Se ha recibido el reporte, pero no se ha categorizado por un usuario del ERP.
- ODT Asignada: Se ha asignado una ODT (Orden de trabajo) a un miembro del personal.
- Cerrado: Se ha completado la ODT y se adjuntan fotos y mensajes sobre el incidente por parte del usuario del ERP.
- Derivado: No corresponde a la administración de la organización, sin embargo, se ha derivado el reporte a la autoridad correspondiente.

Cuando el estado de un incidente se encuentra derivado el seguimiento del incidente queda fuera del alcance de la App.

La App deberá informar al condominio que levantó el reporte de incidente sobre cualquier cambio de estado que tenga ese incidente. La información será cambiada conforme el administrador actualice esa información desde el sistema, enviando al instante una notificación push al dispositivo que posea la aplicación y este haya iniciado sesión.

Las citas pueden ser canceladas por ambas partes, el intervalo en minutos por cita es configurable por la organización.

5.7 Requisitos específicos del subsistema

5.7.1 Casos de uso del Subsistema

A continuación en la figura 15, se describen los casos de uso para el subsistema descrito en las anteriores secciones.

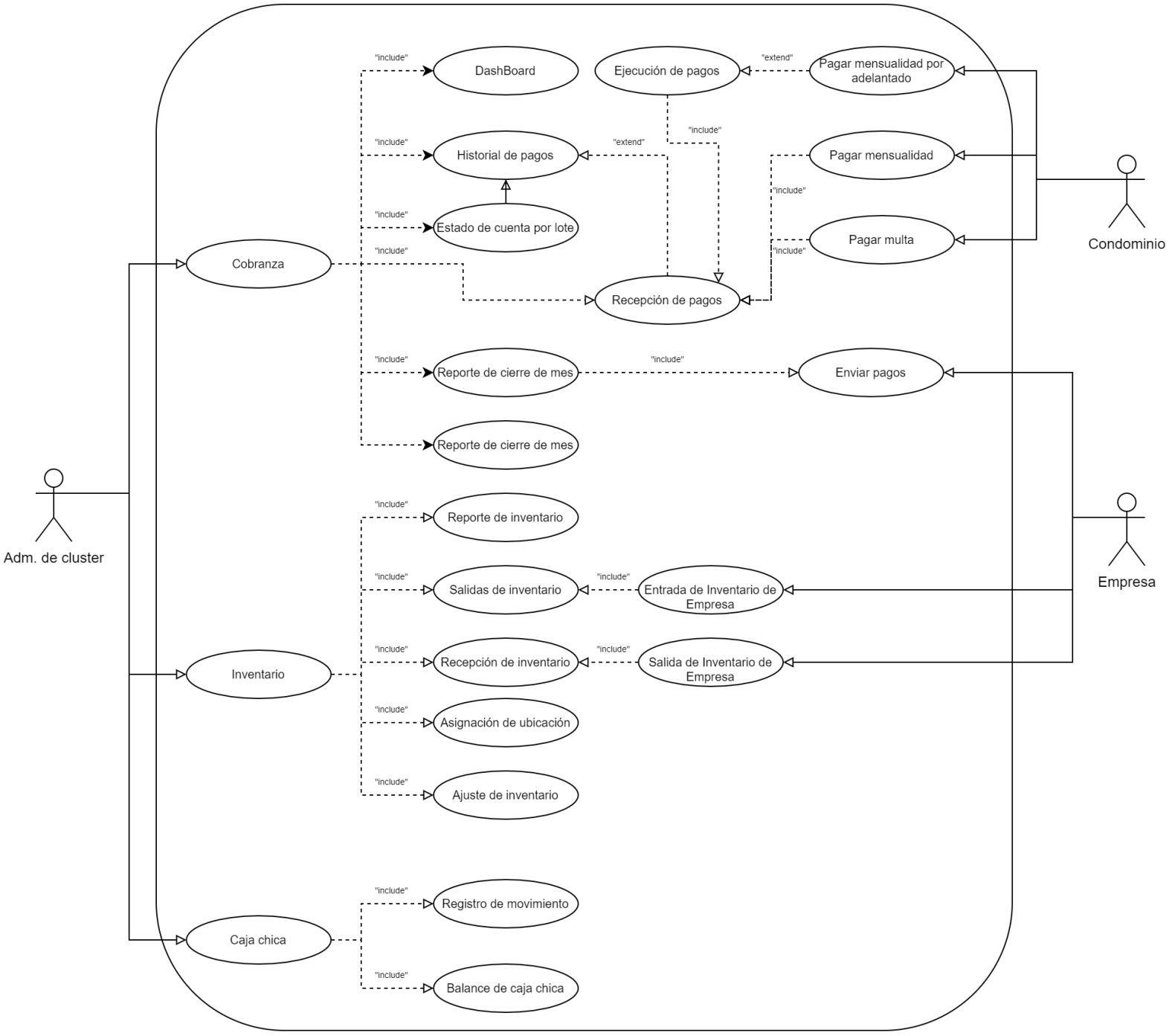

Fig. 15. Diagrama de casos de uso Módulo 2 En Condominio

5.7.2 Especificaciones de actores del subsistema

- Administrador de clúster

  Este actor está encargado de recibir los pagos, ingresarlos al sistema, generar los reportes y mantener en orden el intervalo

  - Encargado de Inventario
  - Encargado de finanzas del cluster
- Condominio

  El condominio es aquella persona propietaria de un lote o casa que debe pagar el mantenimiento

  - Pagar servicios
  - Pagar multas
  - Adelantar pagos de servicios
- Organización administrativa

  El actor se encarga de recibir los pagos de los clúster realizar los pagos y hacer movimientos de inventarios entre clúster

  - Se encarga de recibir el pago del clúster
  - Se le envían reporte de los pagos
  - Mueve inventario entre los clúster

5.8 Especificación de casos de uso del subsistema

5.8.1 Cobranza

El usuario debe de estar registrado previamente en el ERP.

El módulo de cobranza sirve para que el administrador del clúster pueda capturar y visualizar la cartera de clientes en su propio clúster después el sistema mostrará los menús de cobranza.

La información que se presentará por lo general será con el nombre del condominio al que pertenece el pago realizado, con los montos que se cobrarán, esta información puede ser variable dependiendo si es editada por el administrador de la base de datos, los pagos se pueden modificar en los registros pero no se tendrá manera de saber si el monto actual es menor o mayor al monto anterior, no habrá ninguna forma de saber si se regresará el dinero o se abonará más de la cantidad.

5.8.2 Dashboard

El usuario antes debe de estar registrado previamente en el ERP.
El dashboard correspondiente a cobranza debe mostrar en una gráfica de pastel el porcentaje de pagos de cuotas de mantenimiento ya realizados contra los pagos faltantes de realizar, así como la estadística de los pagos atrasados correspondientes a meses anteriores, incluyendo número de lotes con pagos atrasados, número de pagos atrasados en total, cantidad en efectivo por lote de pagos atrasados en promedio y cantidad en efectivo total de pagos atrasados. Se mostrará la gráfica de pastel con los gráficos previamente detallados.

5.8.3 Estado de cuenta por lote

El usuario antes debe de estar registrado previamente en el ERP. El administrador de clúster puede ver un histórico de pagos por lote, de igual forma puede realizar, como único movimiento posible en esta interfaz, traspasar un concepto de fianza a concepto de multa, después mostrar la lista de pagos realizados.

5.8.4 Historial de pagos

El usuario debe de estar registrado previamente en el ERP. Este módulo trata de un histórico de pagos con fecha de pago y dirección de clúster que realizó el pago, después mostrará una lista de todos los pagos realizados (Véase Fig. 16).

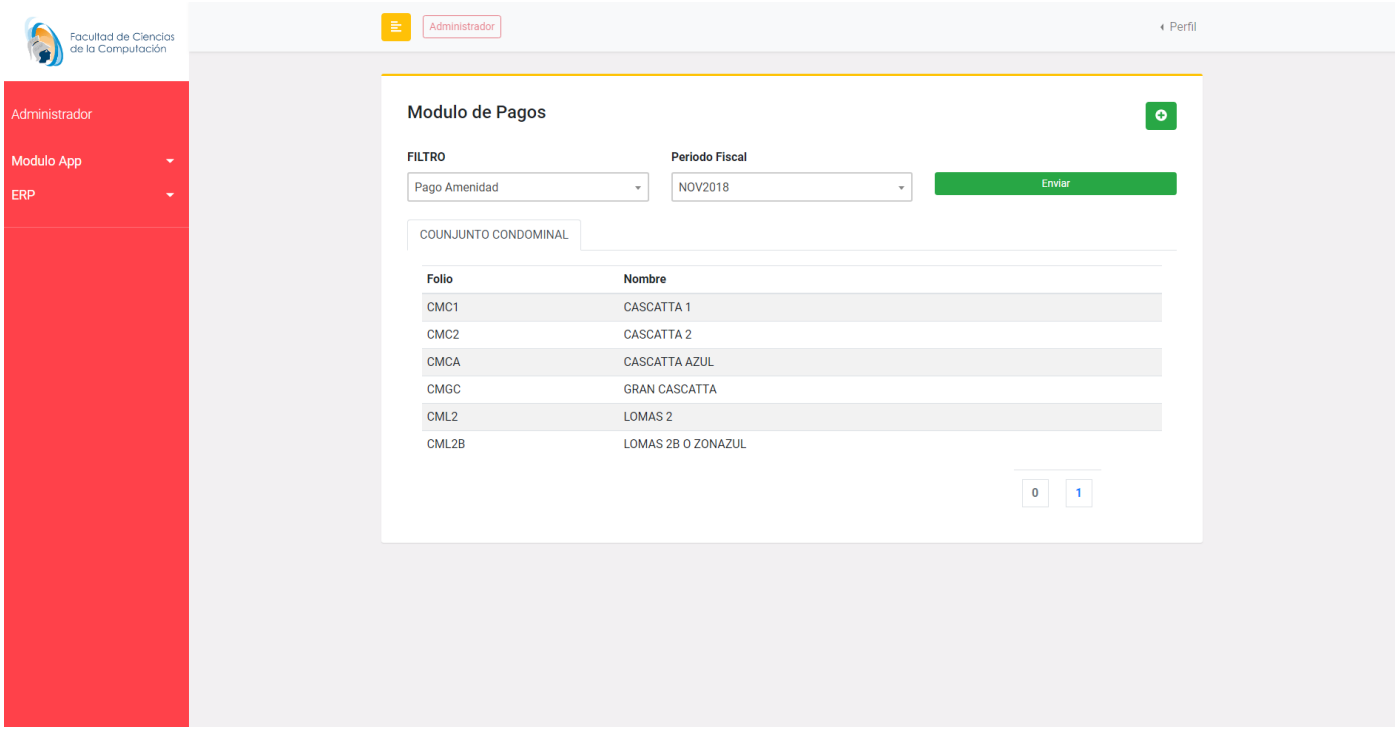

Fig. 16. Resultado Modulo de Pagos

5.8.5 Recepción de pagos

El usuario antes debe de estar registrado previamente en el ERP. El sistema deberá generar un reporte de cierre de mes teniendo en cuenta las entradas por:

- Fianzas.
- Multas.
- Cuotas de mantenimiento del periodo actual.
- Cuotas de mantenimiento de periodos anteriores.

- Abono a cuentas de cliente por cuota de mantenimiento.

Del mismo modo el sistema debe mostrar las salidas por:

- Devolución de fianzas.
- Pago por mantenimiento a la organización.

Al condominio se le enviará un correo de pago realizado y quedará registrado en el ERP del administrador de clúster.

5.8.6 Ejecución de pagos

El condominio pagará por adelantado los servicios. Se trata de un servicio en segundo plano que se encargará de aplicar pagos de mantenimiento a las cuentas de los condominios. El condominio se le marcará como pagado, los servicios correspondientes al mes actual y el administrador de clúster habrá registrado su pago (Véase Fig. 17).

Fig. 17. Resultado Ejecución de Pagos

5.8.7 Pagar mensualidad por adelantado

El condómino se dispondrá a realizar un pago adelantado de los servicios después el condominio adelantará el pago de sus servicios. El condómino habrá pagado por adelantado y el administrador del clúster habrá registrado su pago y el monto de este mismo.

5.8.8 Pagar mensualidad

El condominio antes deberá estar debidamente dado de alta, después el condominio se dispone a pagar los servicios del mes en curso, el pago estará en proceso de ser registrado. La información se registrará con el identificador del condominio, método de pago y cantidad de pago.

5.8.9 Pagar multa

El condómino antes debió haber atrasado con el pago de algún servicio. El condominio se dirige a pagar las multas por servicios no pagados en tiempo y forma. El pago pasará a ser registrado en el ERP.

5.8.10 Reporte de cierre de mes

Se deben de haber registrado los pagos correspondientes a los de los condóminos y estos haber registrado. Este módulo recopila la información de los pagos previamente realizados y los pondrá en un PDF para que puedan ser revisados. El PDF con detalle del mes de los diferentes ingresos y egresos.

5.8.11 Envío de pagos

Se debe tener el reporte previamente generado después se enviará un reporte de los ingresos y egresos del clúster a la organización.

5.8.12 Inventario

El administrador de clúster debe tener un usuario válido.

El módulo de manejo de inventario permite administrar la ubicación y cantidad de los ítems entregados al clúster por la organización.

5.8.13 Reporte de inventario

Antes se debe iniciar sesión de administrador de clúster debidamente, después genera un reporte a partir de la información contenida en el ERP, mostrará el reporte en la pantalla y en formato PDF.

5.8.14 Salidas de inventario

Deberá existir un inventario que se desea traspasar o dar de baja del clúster, el usuario administrador deberá iniciar sesión correspondiente al clúster debidamente, se enviará una solicitud por la organización para la transferencia del inventario.

Este módulo registra cualquier baja de inventario, ya sea por transferencia entre clúster, desgaste de materiales, o pérdidas, después actualiza las cantidades en el inventario.

5.8.15 Entradas de inventario a la organización

Deberá iniciar sesión de administrador de la organización de clúster debidamente realizado, solicitando al clúster para transferencia de inventario, después de esto se restará del inventario del clúster y este se enviará a la organización, el inventario de artículos se actualizará.

5.8.16 Recepción de inventario

Antes habrá una solicitud de entradas de inventario por parte de la organización, se registrará nuevo inventario en el clúster en cuestión, después el inventario de clúster se actualizará.

5.9.17 Salida de inventario de la organización

La organización crea una nueva petición para asignación de nuevo inventario al clúster. La organización envía nuevo inventario al clúster, después el inventario del clúster se actualiza.

5.8.18 Asignación de inventario

Antes debe iniciar sesión de administrador de clúster debidamente, después se realizará un movimiento de inventario interno para el clúster, se asignará una nueva ubicación donde se encuentran los artículos.

5.8.19 Ajuste de inventario.

Antes debe iniciar sesión de administrador debidamente, después se permite aumentar o reducir la cantidad de los artículos del inventario dando una razón del cambio. La cantidad del inventario del clúster se actualiza después la cantidad del inventario del clúster se actualiza.

5.8.20 Caja chica

El administrador del clúster debe tener una cuenta debidamente asignada y autorizada. El módulo de caja chica permite registrar ingresos diversos por conceptos que no son los administrados en el módulo de cobranza, después habrá muestra de menús de este módulo.

5.8.21 Registro de movimientos

Inicio de administrador de clúster debidamente realizados. Permite registrar movimientos de entrada o salida de efectivo guardando el concepto de pago, fecha de movimiento, monto de pago, proveedor y si corresponde a una entrada o salida.
Historial de movimiento actualizado al momento de la acción.

5.8.22 Balance de caja chica

Antes debe de iniciar sesión de administrador de clúster debidamente realizado.
Después muestra el balance actual de la caja chica, así como saldo actual de la misma y registros de movimientos del último mes. El módulo permite registrar

ingresos diversos por conceptos que no son los administrados en el módulo de cobranza y consta de las siguientes interfaces propuestas:

- Registro de movimiento: Permite registrar movimientos de entrada o salida de efectivo, guardando el concepto de pago, fecha del movimiento, monto de pago, proveedor y si corresponde a una entrada o salida
- Balance de caja: Muestra el balance de la caja actual

5.8.23 Cobranza

El módulo de cobranza sirve para que el administrador del clúster pueda capturar y visualizar la cartera de clientes en su propio clúster teniendo como interfaces las propuestas a continuación:

- Dato Maestro de cobranza
    - El dato maestro de cobranza en este módulo es el lote, el cual puede pertenecer a una de las siguientes categorías;
        - Terreno
        - Obra
        - Casa

- Recepción de pagos: El sistema permite registrar pagos recibidos por condominio. Los pagos deben ir a una de las cuentas definidas por el administrador del clúster (efectivo, bancos, etc.) y dividiéndolos en los conceptos de:
    - Fianza, fijo definido por el administrador
    - Multa, variable
    - Pago por mantenimiento – Fijo definido por el administrador

- Ejecución de pagos: Se trata de un servicio en segundo plano que se encargará de aplicar pagos de mantenimiento a las cuentas de los condominios para marcarlo como saldo o aplicar el pago en caso de tener saldo positivo.

- Dashboard: El Dashboard corresponde a cobranza, debe mostrar en una gráfica de pastel el porcentaje de pagos de cuotas de mantenimiento ya realizados contra los pagos faltantes de realizar, así como las estadísticas de los pagos atrasados correspondientes a meses anteriores incluyendo número de lotes con pagos atrasados, número de pagos atrasados en total, cantidad en efectivo por lote de pagos atrasados en promedio y cantidad en efectivo total de pagos atrasados.

- Log de pagos: Trata de un histórico de pagos con fecha de pago y dirección de clúster que realizó el pago

- Estado de cuenta por lote: El administrador del clúster puede ver un histórico de pagos por lote, de igual forma puede realizar, como único movimiento posible en esta interfaz, traspasar un concepto de fianza a concepto de multa

- Reporte de cierre de mes: El sistema debe generar un reporte de cierre de mes teniendo en cuenta las entradas por:
    - Fianzas.
    - Multas.
    - Cuotas de mantenimiento del periodo actual.
    - Cuotas de mantenimiento de periodos anteriores.
    - Abono a cuentas de cliente por cuota de mantenimiento.

- Del mismo modo el sistema debe mostrar las salidas por
    - Devolución de fianzas.
    - Pago por mantenimiento a la organización.

- Los administradores de clúster podrán
    - Ejecutar pagos
    - Ver Dashboard
    - Historial de pagos
    - Realizar y ver los estados de cuentas por lote
    - Realizar y ver los reportes de cierre por mes

  Si el usuario del ERP es un administrador de clúster y tiene rol de mismo

5.8.24 Inventario

El módulo de manejo de inventario permite administrar el inventario entregado al clúster de la organización, tiene como dato maestro el artículo que corresponde tanto herramienta como a insumo, y cuenta con las siguientes interfaces propuestas:

- Reporte de inventario: Muestra el inventario disponible en el clúster en cada una de sus posiciones.
- Salidas de inventario: Permite dar de baja el inventario por consumo de insumo o por pérdida.
- Recepción de inventario: Muestra el inventario enviado por la organización para su aprobación.
- Asignación de ubicación: Permite realizar ajustes de inventario (cantidad de inventario) basados en el inventario actual.

6. Acercamiento a la arquitectura de Microservicios con Spring Cloud

El concepto de arquitecturas de microservicios surgió por primera vez en 2005, cuando Peter Rodger, fundador de Resource Oriented Computing (ROC), acuñó por primera vez el término “micro-servicios web”. En un intento de diseñar una arquitectura de software flexible.

En la última década, los llamados microservicios, también conocidos como arquitecturas de microservicios, han causado altas expectativas. Y esto no es de extrañar, las aplicaciones de software no solo son cada vez más complejas, sino también más engorrosas, lo que hace que la gestión, actualización y mantenimiento de estas aplicaciones sea un gran reto para muchas empresas. Las arquitecturas de microservicios son un enfoque relativamente nuevo para el desarrollo de software que divide las aplicaciones complejas en bloques de construcción o servicio individuales e independientes. La ventaja de usar microservicios es que los equipos de desarrollo pueden crear rápidamente. Estos servicios modulares se pueden implementar y administrar individualmente, lo que hace que las aplicaciones grandes sean más resistentes al cambio.

6.1 Descripción de la Arquitectura

Las arquitecturas de microservicios describen un estilo específico por el cual se desarrollan las aplicaciones de software. A diferencia de los estilos arquitectónicos tradicionales que tienen como objetivo construir software como una sola unidad, las arquitecturas de microservicios adoptan un enfoque modular. Aquí, las aplicaciones se

desarrollan como suites que consisten en una variedad de microservicios que se pueden implementar y administrar de forma independiente entre sí.

Si bien no existe una definición clara para los microservicios, todavía hay algunas características específicas que caracterizan este revolucionario enfoque de desarrollo de software.

Una de las principales características de los microservicios es la forma en que este estilo de arquitectura divide las aplicaciones complejas en diferentes componentes. Por lo tanto el software se divide en sus componentes, que consisten en servicios intercambiables y expandibles de forma independiente. Estos servicios se pueden definir como procesos que a menudo se comunican a través de una red utilizando protocolos independientes de la tecnología. Además, los microservicios se desarrollan de acuerdo con las capacidades del negocio. Y debido a que los microservicios son descentralizados y pequeños, pueden adaptarse de manera flexible a una variedad de lenguajes de programación, hardware y entornos de software.

En entornos los cuales las organizaciones son muy exigentes con la cantidad de transacciones que realizan, ya sea por la gran cantidad de usuarios que manejan o por la cantidad de información que genera, existen formas de hacer un ambiente más escalable con los microservicios, como por ejemplo implementar microservicios con Docker, creando un contenedor para cada uno, y escalando dependiendo de que tanto se sature un microservicio.

6.2 Funcionamiento de los microservicios

La idea de servicios poco acoplados que sean fáciles de mantener y, al mismo tiempo, fáciles de probar suena particularmente fascinante para las empresas cuyos

desafíos de TI se encuentran en aplicaciones que son demasiados complejas. Pero antes de que el sistema existente se revise por completo, las empresas primero deben comprender cómo funcionan los microservicios en primer lugar. Para construir una arquitectura modular, las empresas primero deben identificar sus capacidades comerciales y definir los servicios que se ejecutan de forma independiente. Un diagrama de arquitectura de microservicios es necesario para mapear dependencia que de otro modo afectarían a la flexibilidad de cualquier servicio. Es imperativo asegurarse de que una falla en una sola área no afecte a todo el sistema.

Aunque los microservicios están diseñados para funcionar de forma independiente, en algunos casos siguen siendo interdependientes y necesitan comunicarse entre sí. Esta comunicación se realiza a través de API; protocolos síncronos o asíncronos como HTTP/REST. Spring ofrece una forma de conectar y de comunicar los servicios entre sí con Spring Getaway o una variante como Zuul Gateway. Estas librerías permiten que los microservicios se comuniquen sin necesidad de que el equipo de desarrollo tenga que saber las rutas completas de los servicios, solo se requiere saber el contexto al cual quieren hacer peticiones.

En el mejor de los casos, cada microservicio tiene una propia base de datos, por lo que la responsabilidad puede descentralizarse y la actualización se puede llevar a cabo individualmente. Siguiendo el enfoque de “Base de datos como servicio”, se configura un patrón de saga, es decir, una transacción que abarca varios servicios para garantizar la coherencia de los datos.

Sin embargo, no es raro que cada microservicio desarrolle más dependencias de servicio con el tiempo. Esto es un proceso normal a medida que los servicios crecen e impactan en un número cada vez mayor de interfaces, ubicaciones y protocolos. Las puertas de enlace del API toman todas las solicitudes de otras API y las envían al microservicio que mejor se adopte para manejar la solicitud. Las puertas

de enlace de API también llaman a los servicios back-end y agregan los resultados para determinar la mejor ruta. Por lo tanto, el equipo de desarrollo es responsable de procesar todas las solicitudes, organizar todas las entradas y diseñar una experiencia de usuario simple.

6.3 Ventajas de una arquitectura de microservicios

Los microservicios pueden ofrecer a las empresas una serie de beneficios. Después de todo, los sistemas grandes son populares y requieren una arquitectura de software que satisfaga las necesidades de las actividades empresariales, los desarrolladores y los usuarios finales. Los 10 principales beneficios de las arquitecturas de microservicios son:

1. **Acceso**: Debido a que los microservicios se implementan por separado y se comunican a través de API, el código y las aplicaciones se vuelven más accesibles para los empleados que no forman parte de los equipos de desarrollo. Gracias al enfoque modular, las empresas pueden comprender sus diversos servicios, incluidos sus funciones específicas sin tener que preocuparse por otras partes del software.

2. **Gestión de errores**: Cuando una arquitectura de microservicios se implementa correctamente, facilita a los equipos de desarrollo la detección de errores u otros problemas. Cada vez que ocurre un error, los desarrolladores saben exactamente dónde buscar, ahorrando un tiempo valioso y, en general, reduciendo el esfuerzo. Además, la solución de problemas de un servicio único e independiente significa que no es necesario reparar ningún otro código ni implementar toda la aplicación como tal.

3. **Mantenimiento**: Esta ventaja va de la mano con los dos primeros puntos mencionados. Las arquitecturas de microservicios facilitan el mantenimiento y la gestión de los servicios. De hecho, cada servicio individual puede ser administrado por un equipo diferente, ya que estos servicios representan un proceso único con su propia base de datos. Además se pueden llevar a cabo actualizaciones individuales sin paralizar todo el sistema de software.

4. **Adaptabilidad**: Si estos servicios independientes se desarrollan adecuadamente, no tendrán ningún efecto entre ellos. Esto significa que si un componente falla, a diferencia del enfoque monolítico, toda la aplicación no es afectada de inmediato.

5. **Evolutivo**: Los requisitos del negocio están sujetos a cambios constantes, por lo que los servicios deben adaptarse y agregarse nuevas características. En las arquitecturas tradicionales, donde el software se ve como un todo, esto significa una intervención seria en todo el software empresarial y la posible corrupción de las características ya existentes. El enfoque modular de la arquitectura de microservicios evita estos problemas.

6. **Escalabilidad**: A medida que aumenta la demanda de ciertos servicios, se pueden implementar de manera flexible en múltiples servidores e infraestructuras.

7. **Pruebas y monitoreo**: Los componentes de software individuales son mucho más fáciles de probar que las aplicaciones complejas. Si cada microservicio se prueba adecuadamente, entonces el software como tal suele ser más robusto y confiable. Y dado que sólo puede considerar ciertas partes de una aplicación de forma aislada, esto conduce a una mejor administración de la seguridad.

8. **Provisión**: En últimos años, las técnicas de automatización han mejorado drásticamente, lo que también ha impulsado el desarrollo de la nube y AWS. Gracias a las técnicas de automatización de la infraestructura y al principio de implementación continua, la complejidad operativa de crear, implementar y probar microservicios se ha reducido significativamente.

9. **Integración**. Un software debe integrar tantas características y servicios como sea necesario. Aunque esto puede parecer lógico a primera vista, la integración de un nuevo componente de software en una aplicación existente puede ser un desafío en el desarrollo de software tradicional. Mediante el uso de herramientas de integración continua, los microservicios han revolucionado fundamentalmente este proceso. Para facilitar la implementación, el equipo de trabajo debe invertir en automatización, ya que la complejidad de los microservicios puede ser abrumadora para la implementación humana.

10. **Reusabilidad**: Una ventaja demasiado a menudo descuidada de los microservicios es el hecho de que los componentes independientes que componen el software se pueden utilizar para otros proyectos en el futuro al reutilizar procesos exitosos y solo adaptarlos ligeramente a los nuevos requisitos, los desarrolladores no tiene que reinventar la rueda para cada proyecto y ahorrar recursos valiosos.

6.4 Implementación de una arquitectura de microservicios

Pasar de una arquitectura de software tradicional a un nuevo estilo de arquitectura puede parecer un trabajo muy complicado cuando se piensa en la arquitectura tradicional de software monolítico. Pero dado que los microservicios ya han sido probados y utilizados por muchas otras empresas tales como Netflix o

Spotify, algunas mejores prácticas han surgido en los últimos años. Sin embargo, antes de implementar una arquitectura de microservicios hay que tener en cuenta que cada microservicio se puede desarrollar en un lenguaje de programación diferente o frameworks como Spring que ofrece un conjunto de librerías para llevar a cabo esta tarea como lo es Spring Cloud.

Cuando las organizaciones deciden adoptar en sus equipos de trabajo una arquitectura de microservicios, cada miembro del equipo involucrado en esta transición debe estar completamente informado y preparado. Después de todo, este estilo de arquitectura no se trata solo de subdividir un gran sistema en sus componentes individuales, sino que requiere un cambio en la cultura y la forma de hacer las cosas en la empresa que desea implementarlo. Esto requerirá que todos los empleados del equipo entiendan completamente el cambio y estén listos para aprender las nuevas formas de hacer las cosas.

Cada equipo de trabajo debe ser verdaderamente independiente, mientras que los estándares de comunicación, la documentación común de la API y los formatos de registro son esenciales para crear una buena arquitectura de microservicios. Esto significa que más desarrolladores están trabajando en la misma aplicación al mismo tiempo, lo que reduce significativamente el tiempo de desarrollo.

Las arquitecturas de microservicios están diseñadas para hacer que el software de las organizaciones sean menos complejos Pero si no se construyen, depuran o administran correctamente, los microservicios pueden incluso agregar complejidad a las arquitecturas. Antes de trabajar con microservicios, primero se debe definir la arquitectura de destino. Es fundamental eliminar las dependencias conflictivas y garantizar la coherencia de los datos en todos los microservicios. El ERP en este proyecto tiene la ventaja de que los módulos que lo componen no están fuertemente relacionados, ya que se separaron en paquetes y las librerías en común se extrajeron

así como los objetos de transacción entre API que resulta fundamental para la comunicación entre los servicios.

Separar un sistema grande en fragmentos de aplicaciones pequeñas también significa que cada una de estas pequeñas aplicaciones genera una gran cantidad de protocolos que seguir, que a su vez tienen que ser analizados por software especializado.

Existen distintas formas de analizar el resultado de una arquitectura de microservicios, se puede analizar los logs que estos generan, las rutas que siguen los servicios para llegar a la información que necesitan y los microservicios que están activos en el momento. Por lo tanto, se debe crear un plan claro para la supervisión regular de microservicios individuales a través del tiempo de respuesta a los mensajes de error de servicios. Los dashboards que muchas de estas herramientas ofrecen, los registros, así inteligencia artificial en la nube facilitan estas tareas complejas y crean la transparencia necesaria. Además, cada vez son más populares las nuevas plataformas que analizan las aplicaciones comerciales para brindar una descripción detallada de la arquitectura de los microservicios.

6.5 Relación de Spring Cloud con los microservicios

Como se viene diciendo Spring Cloud es un conjunto de tecnologías las cuales permiten el desarrollo de una arquitectura de microservicios. Para crear una arquitectura se requieren de las dependencias más importantes para realizar una proyecto de este tipo.

Una de las tecnologías base para desarrollar en Spring Cloud sería inicializar el proyecto con Spring Boot 2, cada microservicio tendría que implementar esta versión

empaquetando las librerías comunes que permiten realizar la lógica de negocio y la configuración de un entorno Cloud.

Cada microservicio tendrá que conectarse a un servidor que registra sus nombres permitiendo así que estos entren dentro del mismo ecosistema y puedan reconocerse entre sí, para esta tarea se utiliza comúnmente Eureka Server el cual es el más popular de la industria. Las aplicaciones simplemente describen en su archivo de propiedades la dirección del servidor de Eureka y estas se conectarán ahí registrándose. Antes el equipo de desarrollo tendrá que implementar la dependencia de eureka en el pom.xml del microservicio.

Cuando se tenga que exponer al exterior una API que otros servicios periféricos quieran utilizar para hacer peticiones a nuestro ecosistema de microservicios utilizaremos Spring Cloud Gateway, este se encargará de proporcionar un punto de entrada a nuestros microservicios. Spring Cloud Gateway permite un enrutamiento dinámico, también proporciona librerías de seguridad y da herramientas para monitorear el estado de nuestros microservicios y como se realizan las llamadas. Los microservicios no tendrán que implementar el Gateway, este solo funcionara como otro microservicio aparte el cual se conectara a nuestro ecosistema apuntando al servidor donde se encuentre el servidor de Eureka. En el Gateway podrá configurar cuáles serán las rutas que queremos exponer al mundo, esto permite que solo demos a conocer mediante ciertos privilegios la información. Permitirá que tengamos un conjunto de microservicios los cuales solo serán herramientas para otros microservicios, ya sea para realizar operaciones o para validar cierta información, sin necesidad de que estos estén al alcance de cualquier usuario de la web, agregando una capa más de seguridad.

Una de las dudas que surge es cómo saber cuál es la mejor instancia para realizar las peticiones y que este no colapse. Dentro de las ventajas de utilizar Eureka

permite utilizar Ribbon el cual es un balanceador de carga automático. Ribbon también nos proporciona Integración de descubrimiento de servicios, tolerancia a fallos y reglas de equilibrio de carga configurables.

6.5.1 Dependencias de un ecosistema Spring Cloud

Spring Boot Starters son descriptores de dependencia, aumenta la productividad disminuyendo el tiempo de configuración para los desarrolladores, la configuración es realizada mediante archivos de propiedades en caso de que sea necesario:

```
<dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter</artifactId>
</dependency>
```

Spring Cloud Config es una dependencia que permite unificar la configuración del ecosistema de microservicios en un solo ambiente, lo que facilita la administración de la configuración:

```
<dependency>
    <groupId>org.springframework.cloud</groupId>
    <artifactId>spring-cloud-starter-config</artifactId>
</dependency>
```

Spring Cloud Starter Gateway, es la dependencia que permite crear un Gateway de nuestro ecosistema, este deberá estar en un proyecto aparte:

```
<dependency>
    <groupId>org.springframework.cloud</groupId>
    <artifactId>spring-cloud-starter-gateway</artifactId>
</dependency>
```

Spring Config Server tiene como propósito obtener las configuraciones desde un repositorio de git o desde un archivo de propiedades externo:

```
<dependency>
  <groupId>org.springframework.cloud</groupId>
  <artifactId>spring-cloud-config-server</artifactId>
</dependency>
```

6.6 Seguridad OAuth2 con Microservicios

La tecnología de OAuth 2.0 es un protocolo de la industria actualmente que utilizan las grandes empresas de la tecnología para dar acceso a los servicios para los usuarios. Esta tecnología de seguridad vía web maneja el flujo de la seguridad para distintos dispositivos como equipo de escritorio, teléfonos móviles y dispositivos de sala de estar.

OAuth 2.0 es un estándar abierto que le permite adoptar el modelo de seguridad y utilizarlo para aplicaciones web y móviles. Le permite crear soluciones de inicio de sesión único, que comúnmente se denominan "autenticación en dos pasos".

Roles de OAuth

Define 4 roles:

1) Propietario del recurso
2) Cliente
3) Servidor de recursos
4) Servidor de autorización

Conclusiones

La presente tesis tuvo como objetivo comprobar que la administración de zonas urbanas puede ser llevada con más facilidad con una herramienta como un ERP, sistema el cual está pensado únicamente para organizaciones de este tipo, las cuales tienen requerimientos especiales con diferencia a empresas o entidades gubernamentales, los cuales ya tienen la mayor parte de sus necesidades cubiertas por ERP actuales de la industria.

El enfoque de este proyecto fue principalmente a los usuarios finales, los cuales son las oficinas administrativas y en último los usuarios de los condominios. Se terminó desarrollando una interfaz amigable para ellos en el uso del sistema.

Se dejó una arquitectura extensible la cual permite agregar o quitar más módulos los cuales se implementarán dependiendo de las necesidades únicas que tienen las zonas urbanas ya que no todas son iguales.

La API desarrollada para la App también está abierta para cambios dependiendo de las necesidades que tengan las zonas urbanas para los usuarios finales los cuales serían las personas que viven en ellas.

Se espera que con el paso del tiempo este sistema o la idea de uno de este tipo puedan solucionar más necesidades en el futuro. El impacto económico que tiene en los trabajos de las oficinas administrativas puede ser muy grande dependiendo de los módulos que ellos requieran o de los sistemas a los cuales deseen conectarlo. Cada necesidad que ésta requiera se trabajará conjuntamente con el cliente y con los consumidores de este servicio, para poder tener una idea más clara de las nuevas

necesidades que surjan. Cada empresa es distinta, y su uso requiere la disponibilidad de aplicaciones o lógica de negocios única para cumplir con las regulaciones de la industria u otras necesidades distintas

Todo el núcleo principal se ha implementado con Spring Boot, tiene como objetivo hacer más fácil el despliegue de aplicaciones y módulos durante las fases de desarrollo y de puesta a producción cualquier cambio que se realice no tendrá muchas dificultades para ponerse en marcha.

Quizás la mejora de los desarrollos en esta fase precise de una revisión a profundidad, cada función tiene que ser probada por los usuarios finales de cada plataforma. Monitorear constantemente durante los primeros días en el primer mes después de la puesta en producción, así con la finalidad de buscar los bugs y corregirlos inmediatamente para no afectar a los usuarios. Así como mejorar la seguridad y la integridad de los datos.

Para cada uno de los módulos que se implementan en el ERP se pueden separar para que estos sean servicios especializados en su tarea, comúnmente llamados microservicios. Estos microservicios se pueden comunicar entre sí para realizar las transacciones y cada microservicios puede ser replicado para que la carga de trabajo no caiga en uno solo, sino que este se pueda balancear.

En caso de que la arquitectura monolítica pase a ser una arquitectura de microservicios de alto rendimiento se tiene como ventajas que Spring cuenta con librerías dedicadas para estas actividades, tales como Spring Cloud, Spring Gateway y Eureka, para que se encarguen del balanceo, enrutamiento y exposición del API los cuales serán utilizados por sistemas periféricos que requieran consumir la información del ERP. Teniendo esto en mente al añadir un nuevo módulo lo hace aún más fácil ya que no dependen de todo una estructura compleja a la que los programadores tengan

que dedicar tiempo en entender y aprender a modificar, y en caso de algún cambio, solo afectaría al componente que se está trabajando y no con los demás.

Referencias

1. Jacobs, F. Robert. "Enterprise resource planning (ERP)—A brief history." *Journal of operations management* 25.2 (2007): 357-363.
2. Moon, Young. "Enterprise Resource Planning (ERP): a review of the literature." (2007).
3. Jakl, Michael. "Rest representational state transfer." (2008).
4. Ong, Shyue Ping, et al. "The Materials Application Programming Interface (API): A simple, flexible and efficient API for materials data based on REpresentational State Transfer (REST) principles." *Computational Materials Science* 97 (2015): 209-215.
5. Rodriguez, Alex. "Restful web services: The basics." *IBM developerWorks* 33 (2008): 18.
6. Fielding, Roy. "Representational state transfer." *Architectural Styles and the Design of Netowork-based Software Architecture* (2000): 76-85.
7. Li, Hongjun. "RESTful Web service frameworks in Java." *2011 IEEE International Conference on Signal Processing, Communications and Computing (ICSPCC)*. IEEE, 2011.
8. Deacon, John. "Model-view-controller (mvc) architecture." *Online][Citado em: 10 de março de 2006.] http://www. jdl. co. uk/briefings/MVC. pdf* (2009).
9. Olanrewaju, Rashidah F., Thouhedul Islam, and Nor'ashikin Ali. "An empirical study of the evolution of PHP MVC framework." *Advanced Computer and Communication Engineering Technology*. Springer, Cham, 2015. 399-410.
10. Bawiskar, Ankur, et al. "Spring Framework: A Companion to JavaEE." *IJCEM* 1 (2012): 41-49.
11. Radu, A. Corso, G. Lehmann Miotto, and L. Magnoni. "The electronic logbook for the information storage of ATLAS experiment at LHC (ELisA)." *Journal of Physics: Conference Series*. Vol. 396. No. 1. IOP Publishing, 2012.
12. Alfonzo, Pedro L., and Sonia I. Mariño. "Los estándares internacionales y su importancia para la industria del software." (2013).
13. Plossl, George W., and Joseph Orlicky. Orlicky's material requirements planning. McGraw-Hill Professional, 1994.
14. Hastings, Nicholas AJ, Peter Marshall, and Robert J. Willis. "Schedule based MRP: An integrated approach to production scheduling and material requirements planning." Journal of the Operational Research Society 33.11 (1982): 1021-1029.
15. Baker, Kenneth R. "Requirements planning." Handbooks in operations research and management science 4 (1993): 571-627.
16. Beveridge, David L., and Kevin J. McConnell. "Nucleic acids: theory and computer simulation, Y2K." Current opinion in structural biology 10.2 (2000): 182-196.

17. Morrissey, J. "Y2K: ready or not... survey: millennium bug is sapping time and resources, but execs are glancing beyond 2000." Modern healthcare 29.8 (1999): 52-4.
18. Stratman, Jeff K., and Aleda V. Roth. "Enterprise resource planning (ERP) competence constructs: two-stage multi-item scale development and validation." Decision Sciences 33.4 (2002): 601-628.
19. Ahituv, Niv, Seev Neumann, and Moshe Zviran. "A system development methodology for ERP systems." Journal of Computer Information Systems 42.3 (2002): 56-67.
20. Kan, Stephen H. Metrics and models in software quality engineering. Addison-Wesley Longman Publishing Co., Inc., 2002.
21. Abran, Alain. Software project estimation: the fundamentals for providing high quality information to decision makers. John Wiley & Sons, 2015.
22. Holt, George. "Risk Management Fundamentals in Software Development." What Makes Software Process Improvement Happen? 801 (2000): 12.
23. Humphrey, Watts S. A discipline for software engineering. Addison-Wesley Longman Publishing Co., Inc., 1995.
24. Floyd, Christiane, Fanny-Michaela Reisin, and Gerhard Schmidt. "STEPS to software development with users." European Software Engineering Conference. Springer, Berlin, Heidelberg, 1989.
25. Jacobson, Ivar. The unified software development process. Pearson Education India, 1999.
26. Booch, Grady. "Software Architecture and the UML." Slides available at< http://www. rational. com/uml/index. jtmpl (1998).
27. Ehie, Ike C., and Mogens Madsen. "Identifying critical issues in enterprise resource planning (ERP) implementation." Computers in industry 56.6 (2005): 545-557.
28. Goyal, D. P., and Gurbinder Randhawa. "Design of a planning model for ERP systems: an empirical study of Indian organisations." International Journal of Business Information Systems 3.2 (2008): 201-215.
29. Vilpola, Inka Heidi. "A method for improving ERP implementation success by the principles and process of user-centred design." Enterprise Information Systems 2.1 (2008): 47-76.
30. Van Gurp, Jilles, Sjaak Brinkkemper, and Jan Bosch. "Design preservation over subsequent releases of a software product: a case study of Baan ERP." Journal of Software Maintenance and Evolution: Research and Practice 17.4 (2005): 277-306.
31. Soffer, Pnina, Boaz Golany, and Dov Dori. "Aligning an ERP system with enterprise requirements: An object-process based approach." Computers in industry 56.6 (2005): 639-662.
32. Yang, Jyh-Bin, Chih-Tes Wu, and Chiang-Huai Tsai. "Selection of an ERP system for a construction firm in Taiwan: A case study." Automation in construction 16.6 (2007): 787-796.
33. Chan, Susy. "Architecture choices for ERP systems." AMCIS 1999 Proceedings (1999): 75.

34. Park, Kusiak, and Andrew Kusiak*. "Enterprise resource planning (ERP) operations support system for maintaining process integration." International Journal of Production Research 43.19 (2005): 3959-3982.
35. Chakherlou, T. N., B. Abazadeh, and Jeffrey Vogwell. "The effect of bolt clamping force on the fracture strength and the stress intensity factor of a plate containing a fastener hole with edge cracks." Engineering Failure Analysis 16.1 (2009): 242-253.
36. Boykin, Raymond F., and Wm Benjamin Martz Jr. "The integration of ERP into a logistics curriculum: applying a systems approach." Journal of enterprise information management 17.1 (2004): 45-55.
37. Zhezhnych, Pavlo, and Dmytro Tarasov. "Methods of data processing restriction in ERP systems." 2018 IEEE 13th International Scientific and Technical Conference on Computer Sciences and Information Technologies (CSIT). Vol. 1. IEEE, 2018.
38. Al-Ani, Mohammed Bashar Husham. "RESTFUL WEB SERVICES FOR AN ERP SYSTEM FOR SOCIAL SERVICES."
39. Baharum, Zirawani, Mohd Salihin Ngadiman, and Habibollah Haron. "Critical factors to ensure the successful of OS-ERP implementation based on technical requirement point of view." 2009 Third Asia International Conference on Modelling & Simulation. IEEE, 2009.
40. Faber, Eberhard, and Wolfgang Behnsen. Secure ICT Service Provisioning for Cloud, Mobile and Beyond: A Workable Architectural Approach Balancing Between Buyers and Providers. Springer Science & Business Media, 2012.
41. ISACA, COBIT 5. A Business Framework for the Governance and Management of Enterprise IT, 2012.
42. Corporate responsibility, Sarbanes-Oxley, 2002, p. 5 USC 7201 note.
43. S. Ahmad K, ITIL: Service Management Implementation and Operation, Auerbach Publications, 2010.
44. ISO/IEC 27001, Information technology. Security techniques Information security management systems Requirements, 2005.
45. ISO/IEC 27002, Information technology – Security techniques – Code of practice for information security management,, 2008-09.
46. V. H. Publishing, Ed., TOGAF Version 9.1 Enterprise Edition, 2011.
47. W. B. Eberhard Von Faber, Secure ICT Service Provisioning for Cloud, Mobile and Beyond - A Workable Architectural Approach Balancing Between Buyers and Providers, P. Hohl, Ed., Ingelheim: Springer Vieweg, 2013.
48. SO/IEC 27002, Information technology – Security techniques – Code of practice for information security management, 2008 – 09.
49. Information Security Forum (ISF), The Standard of Good Practice for Informatio Security, 2011.
50. G. c. b. t. a. l. b. protection. [En línea]. Available: www.bsi.bund.de/EN.
51. ISAE, nternational standard on assurance Engagements (ISAE) 3402, 2011.

52. Leff, Avraham, and James T. Rayfield. "Web-application development using the model/view/controller design pattern." Proceedings fifth IEEE international enterprise distributed object computing conference. IEEE, 2001.