UNIVERSIDAD REY JUAN CARLOS

LICENCIATURA EN INGENIERÍA SUPERIOR EN INFORMÁTICA

PROYECTO FIN DE CARRERA

..TÍTULO PROYECTO..

AUTOR: LÓPEZ CEREZO, ALEJANDRO M.

TUTOR: FERNÁNDEZ GIL, ALBERTO

Agradecimientos

Tras ser capaz de desarrollar mis estudios universitarios de un modo satisfactorio, llegué al último paso: el Proyecto Fin de Carrera.

La presente memoria hace referencia a, por diferentes motivos, el tercer proyecto que realizo, quedando su desarrollo y elaboración compaginada con mi actividad laboral; circunstancia que ha implicado numerosas dificultades ligadas fundamentalmente a la falta de tiempo. Por ello, su finalización no hubiese sido posible sin el apoyo y ayuda de una serie de personas que me gustaría tener en cuenta en este punto final.

A mi tutor, Alberto Fernández, por haberme dado la oportunidad de desarrollar el proyecto bajo su supervisión cuando, después de una mala experiencia previa, me quedé sin ninguno asignado.

A mis padres, mi hermano y mi novia, Ana, por su comprensión y apoyo constante durante el proceso y, sobre todo, en esos fines de semana de dedicación interminables en los que no estaba para nada ni nadie después de toda la semana de trabajo.

A todos, gracias.

Resumen

El presente proyecto ha supuesto el desarrollo de una aplicación Android completa para la gestión de parques de bicicletas públicos, estableciendo una infraestructura tecnológica base para desarrollos e investigaciones futuras.

Aplicaciones como la presentada cobran gran importancia en la actualidad dados dos factores básicos: por un lado, la utilización, cada vez mayor, de la bicicleta pública como alternativa a los medios de transporte tradicionales; por otro lado, la presencia de los dispositivos móviles como herramientas de gestión de tareas diarias, como podría ser la reserva de un bicicleta. Así, se hace necesario contar con aplicaciones y sistemas eficientes, eficaces y fácilmente utilizables para la gestión de dichos parques públicos.

La herramienta presenta un mapa actualizado de las estaciones disponibles sobre el que poder operar, así como pantallas adicionales para facilitar la interacción del usuario con la aplicación a la hora de gestionar su perfil o conocer el estado del parque de bicicletas.

Desde un punto de vista técnico, la aplicación sigue una arquitectura cliente-servidor de tres capas, estructurándose la comunicación entre el ambos mediante el estándar REST de los servicios web.

El desarrollo ha seguido un modelo de proceso incremental bajo un paradigma orientado a objetos, tratando de tener en todo momento en mente las buenas prácticas establecidas por la Ingeniería del Software, las *Reglas de Oro* para el diseño de interfaces de usuario, recomendaciones de diseño e implementación Android, etc. Dado el modelo señalado, el proceso se ha dividido en una serie de incrementos jerarquizados por importancia y desarrollados de manera individual para, finalmente, quedar integrados en el producto final.

La aplicación ha sido probada mediante pruebas planificadas de unidad, de integración y de validación para tratar de lograr una elevada cobertura de errores y calidad final.

Finalmente, señalar que como se ha indicado al principio de este resumen, la herramienta presentada supone una infraestructura base para investigaciones futuras que conduzcan a modelos de gestión más eficientes y con mayor capacidad de atracción de usuarios que los utilizados actualmente.

Índice general

Índice de figuras

Índice de tablas

Capítulo 1

Introducción

1.1. Motivación

Los parques públicos de bicicletas están cada vez más extendidos en las ciudades como alternativa a los medios de transporte tradicionales. Asimismo, los dispositivos móviles representan, para gran parte de la población, la herramienta básica para el manejo de numerosas tareas diarias.

Ambos factores suponen la motivación básica para el desarrollo de una aplicación de gestión de dichos parques, de manera que los usuarios registrados puedan conocer y operar con las estaciones disponibles y se logre una experiencia de usuario positiva al mismo tiempo que se realice una gestión eficiente del parque.

El presente proyecto surge con la idea de suponer una infraestructura tecnológica base para el desarrollo e investigaciones futuras sobre el área que mejoren la gestión del conjunto de estaciones.

1.2. Metodología

El modelo de proceso seguido ha sido el *modelo incremental*, que combina elementos de los flujos de proceso lineal y paralelo. Como se puede observar en la figura 1.1, el modelo aplica secuencias lineales en forma escalonada a medida que avanza el calendario de actividades. Cada secuencia lineal produce *incrementos* de software susceptibles de entregarse de manera parecida a los incrementos producidos en un flujo de proceso evolutivo.

Cuando se utiliza un modelo incremental, es frecuente que el primer incremento sea el *producto fundamental*. Es decir, se abordan los requerimientos básicos, pero no se proporcionan muchas características suplementarias (algunas conocidas y otras no). Como resultado del uso y/o evaluación de este primer producto, se desarrolla un plan para el incremento que sigue. El plan incluye la modificación

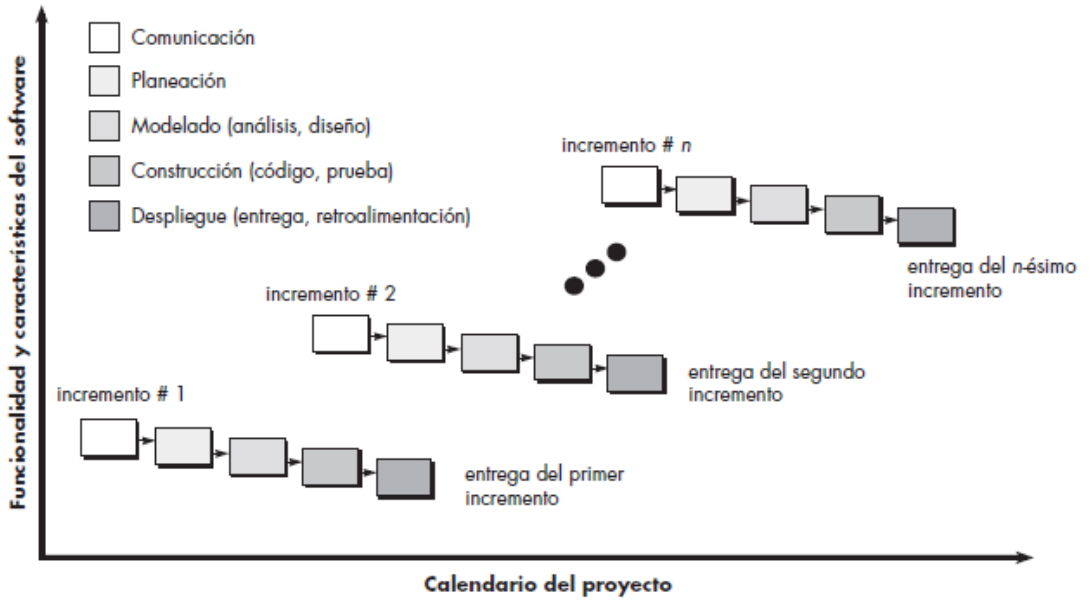

Figura 1.1: El modelo incremental

del producto fundamental para cumplir mejor las nuevas necesidades, así como la entrega de características adicionales y más funcionalidad. Este proceso se repite después de entregar cada incremento, hasta terminar el producto final.

El modelo de proceso incremental se centra en que en cada incremento se entrega un producto que ya opera. Los primeros incrementos son versiones desnudas del producto final, pero proporcionan capacidad que sirve al usuario y también le dan una plataforma de evaluación.

Considerando las características anteriores, los incrementos generales establecidos han sido los siguientes:

1. Mapa con estaciones sobre las que poder operar con datos locales.
2. Lista con el detalle de las estaciones mostradas en el mapa.
3. Gráficos con el estado del parque de estaciones.
4. Desarrollo e integración con la aplicación del Servidor y base de datos.
5. Sistema gestor de usuarios.
6. Sistema de gestión de reservas ligadas al usuario.

Capítulo 2

Objetivos

El objetivo primario del presente proyecto ha sido el desarrollo de una aplicación móvil para la gestión de un parque público de bicicletas, de manera desglosada:

- Desarrollar una aplicación Android comunicada con un servidor y base de datos mediante servicios web que implemente un sistema de gestión de bicicletas que permita coger, dejar o reservar bicicletas y anclajes a usuarios registrados.
- Introducir un primer nivel de adaptación dinámica de precios dependiendo de la disponibilidad individual de cada estación de bicicletas.
- Trasladar, en la medida de los posible, la lógica de las operaciones al servidor, haciendo más eficiente la aplicación instalada.
- Seguir las recomendaciones generales de diseño e implementación procedentes de fuentes especializadas para asegurar una adecuada experiencia de usuario.
- Evitar condiciones de carrera ocasionadas como consecuencia de accesos simultáneos a un mismo recurso.
- Cubrir el mayor volumen de dispositivos posible a nivel tecnológico (versiones Android), lingüístico (diferentes idiomas), etc.

Capítulo 3

Descripción informática

3.1. Especificación de Requisitos Software

3.1.1. Introducción

Propósito

El propósito de esta sección es el de presentar los requisitos de la aplicación acorde al estándar *IEEE Std. 830-1998: Especificación de Requisitos Software* (ERS en adelante), mostrando y esquematizando la funcionalidad básica del software a desarrollar.

El documento va principalmente dirigido a futuros usuarios y desarrolladores de la aplicación, de modo que cuenten con una aproximación teórica a la misma y a sus posibilidades.

Ámbito del Sistema

El futuro sistema, de nombre *Bikesmanager*, consistirá en una aplicación Android encargada de la gestión de parques públicos de bicicletas a través de usuarios previamente registrados.

Se buscará desarrollar una aplicación intuitiva y fácil de usar que cubra las necesidades de manera eficiente y eficaz, proporcionando una adecuada experiencia al usuario final.

Definiciones, Acrónimos y Abreviaturas

- ERS: Especificación de Requisitos Software.
- Bikesmanager: Nombre de la aplicación a desarrollar.

- Android: Sistema operativo diseñado principalmente para dispositivos móviles con pantalla táctil.
- Usuario: Persona que hará uso de la aplicación.
- RF: Requerimiento Funcional.
- RNF: Requerimineto No Funcional.

Referencias

- IEEE Std. 830-1998: Especificaciones de los Requisitos del Software

Visión General del Documento

Una vez realizada la introducción general previa, se aportará a continuación una primera descripción general del sistema a desarrollar para, finalmente, pasar a detallar los requisitos específicos básicos del mismo.

En el apartado dedicado a la descripción general se aporta una visión global de la aplicación, así como de las funciones básicas de la misma. Funciones que se detallan en la sección siguiente, junto con otros aspectos como los atributos del sistema sobre los que se implantará el desarrollo.

3.1.2. Descripción General

Perspectiva del Producto

La aplicación *Bikesmanager* será un producto diseñado para trabajar sobre dispositivos móviles con sistema operativo Android, donde los datos quedarán almacenados en una base de datos a la que se accederá mediante el servidor de la aplicación. La conexión de la herramienta con el servidor se realizará mediante servicios web.

Funciones del Producto

En la figura 3.1 se aporta el diagrama de casos de uso que muestra, a grandes rasgos, las funciones del futuro sistema.

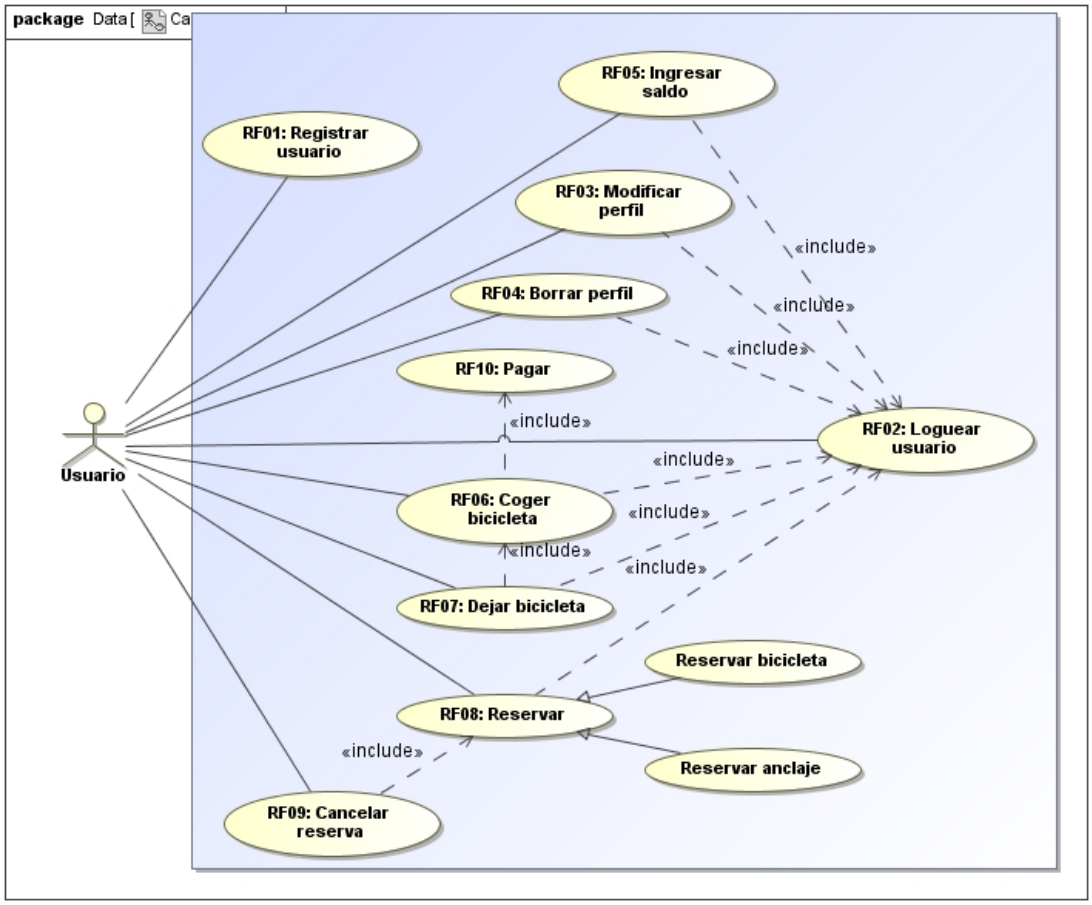

Figura 3.1: Diagrama de casos de uso

Características de los Usuarios

- Tipo de usuario: Usuario
  - Nivel educacional: irrelevante.
  - Experiencia técnica: experiencia en el manejo de *smartphones*.
  - Actividad: manejo de la aplicación.

Restricciones

- Limitaciones hardware:
  - Los servidores han de ser capaces de atender consultas concurrentes.

    - Los dispositivos móviles deberán estar gestionados por el sistema operativo Android.

- Arquitectura del sistema: cliente-servidor de tres capas, con servidor de aplicación Glassfish y SQL Server de base de datos.
- Lenguaje(s) en uso: JAVA, SQL.
- Protocolos de comunicación:
    - Aplicación – Servidor: HTTP (RESTful Web Services - JSON).
    - Servidor Aplicación – Base de Datos: MySQL Connector/JDBC.
- Consideraciones acerca de la seguridad:
    - El acceso a la aplicación se realizará mediante el par usuario-contraseña.
    - Las claves de usuario deberán almacenarse de manera segura mediante encriptación SHA-1.
    - Desde la aplicación ningún usuario tendrá acceso a la administración interna de la misma, sino que dicha tarea se realizará directamente sobre el servidor o base de datos.

Suposiciones y Dependencias

- Se asume que los requisitos aquí descritos son estables.
- Los equipos en los que se vaya a ejecutar el sistema deben cumplir los requisitos antes indicados para garantizar el adecuado funcionamiento de la aplicación.

Requisitos Futuros

- Adaptación a nuevas plataformas (iOS, versión Web, etc.).

3.1.3. Requisitos Específicos

Interfaces Externas

La interfaz con el usuario consistirá en un conjunto de pantallas con los controles habituales: botones, campos de texto, listas, etc. sobre la que se deberá asegurar una adecuada experiencia de usuario mediante diseños que sigan las normas Android [1]. Esta interfaz deberá ser construida para el sistema especificado

[1] `https://developer.android.com/guide/practices/ui_guidelines/index.html`

y será visualizada mediante dispositivos móviles. Del mismo modo, destacar que el diseño de la interfaz gráfica de usuario deberá fundamentarse en los llamados *fragments* de Android, de modo que el mismo pueda ser reutilizable y fácilmente transportable a otras dimensiones u orientaciones de pantalla.

En relación a las interfaces hardware-software, se deberá contar con un dispositivo móvil con conexión a internet y que implemente, como mínimo, la versión 4.0 del sistema Android (llamada *Ice Cream Sandwich*[2]).

Finalmente, acerca de las interfaces de comunicación, la aplicación se comunicará con su servidor mediante el protocolo HTTP haciendo uso de RESTful Web Services y cadenas JSON. La conexión entre el servidor de aplicación y la base de datos se realizará mediante las tecnologías MySQL Connector/JDBC.

Funciones

En esta sección se desarrolla la descripción de los diferentes requerimientos funcionales y no funcionales con que deberá contar el sistema.

Se comienza por los *Requerimientos Funcionales*, para los que se aporta su definición acompañada de una descripción textual y del diagrama de actividad correspondiente, de modo que cada requerimiento quede adecuadamente descrito.

- **RF01: Registrar usuario**. El sistema deberá permitir el registro de nuevos usuarios. Para ello, se solicitarán el nombre usuario, contraseña, dirección de correo electrónico y nombre y apellidos reales. El nombre de usuario y la dirección de correo han de ser únicos, de modo que no se permita la duplicidad de los mismos en la aplicación. Asimismo, la contraseña ha de ser encriptada a la hora de ser almacenada en la base de datos.

  La descripción textual:

[2]Android denomina alfabéticamente y con nombres de dulces a cada una de las versiones de su sistema operativo. En el momento de la redacción de este documento, la versión más actualizada es la 7.0, *Nougat*.

RF01: Registrar usuario	
Usuario	**Sistema**
1. Seleccionar registro	
	2. Mostrar interfaz
3. Introducir datos	
	4. Comprobar datos obligatorios
	5. Validar campos únicos
	6. Crear registro

Tabla 3.1: Descripción textual de RF01: Registrar usuario

Y el diagrama de actividad:

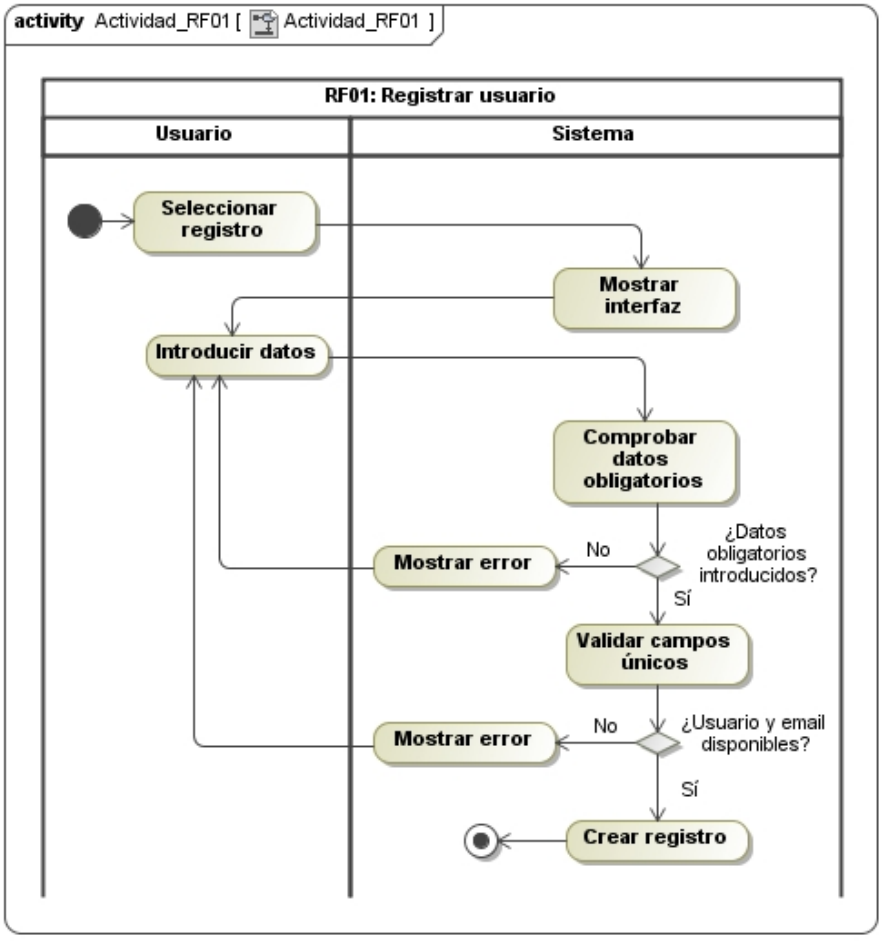

Figura 3.2: Diagrama de actividad de RF01: Registrar usuario

- **RF02: Loguear usuario**. El sistema deberá permitir la autenticación de usuarios para su acceso. Este proceso se realizará mediante el par usuario-contraseña. Señalar que, como se ha especificado en el requisito previo, la contraseña se almacena encriptada, de modo que la introducida por el usuario que trata de acceder ha de ser tratada por el mismo algortimo para poder realizar la comparación.

  La descripción textual:

RF02: Loguear usuario	
Usuario	**Sistema**
1. Seleccionar login	
	2. Mostrar interfaz
3. Introducir datos	
	4. Comprobar campos obligatorios
	5. Autenticar
	6. Completar login

Tabla 3.2: Descripción textual de RF02: Loguear usuario

Y el diagrama de actividad:

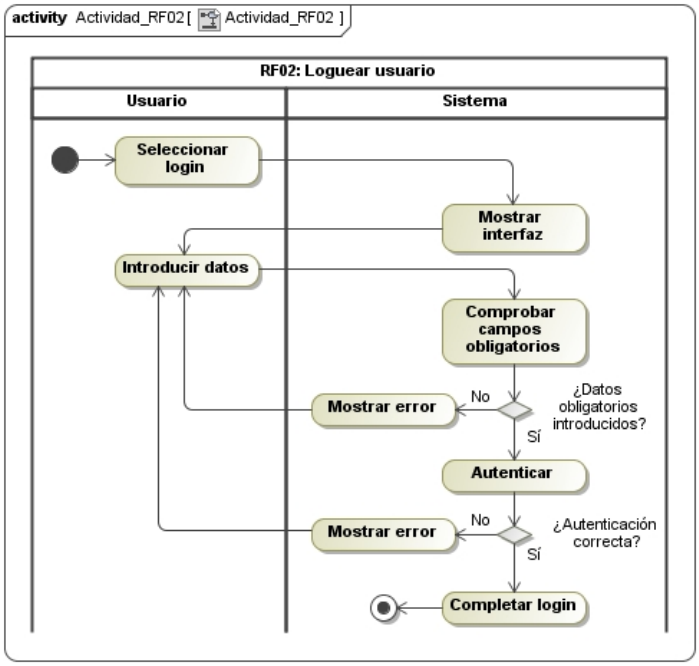

Figura 3.3: Diagrama de actividad de RF02: Loguear usuario

- **RF03: Modificar perfil**. El sistema deberá permitir la modificación del perfil de usuario solicitados en el proceso de registro, considerando que el nombre de usuario y dirección de correo han de ser únicos.

  La descripción textual:

RF03: Modificar perfil	
Usuario	**Sistema**
[Pto. inclusión: RF02: Loguear usuario]	
1. Seleccionar modificar perfil	
	2. Mostrar interfaz
3. Introducir datos	
	4. Validar campos únicos
	5. Solicitar confirmación
6. Confirmar operación	
	7. Actualizar registro
	8. Confirmar operación terminada

Tabla 3.3: Descripción textual de RF03: Modificar perfil

Y el diagrama de actividad:

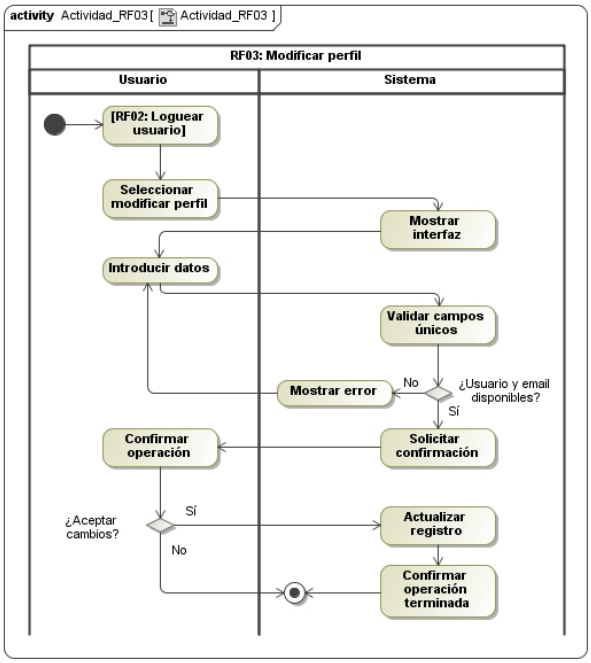

Figura 3.4: Diagrama de actividad de RF03: Modificar perfil

- **RF04: Borrar perfil**. El sistema deberá permitir el borrado completo de perfiles. Este proceso requerirá de una confirmación explícita por parte del usuario y, en el momento en que se realice, será irreversible, borrando el registro específico de la base de datos. El dinero ingresado será devuelto y las posibles reservas, canceladas.

  La descripción textual:

RF04: Borrar perfil	
Usuario	**Sistema**
[Pto. inclusión: RF02: Loguear usuario]	
1. Seleccionar borrar perfil	
	2. Solicitar confirmación
3. Confirmar operación	
	4. Devolver saldo y eliminar posibles reservas
	5. Borrar registro

Tabla 3.4: Descripción textual de RF04: Borrar perfil

Y el diagrama de actividad:

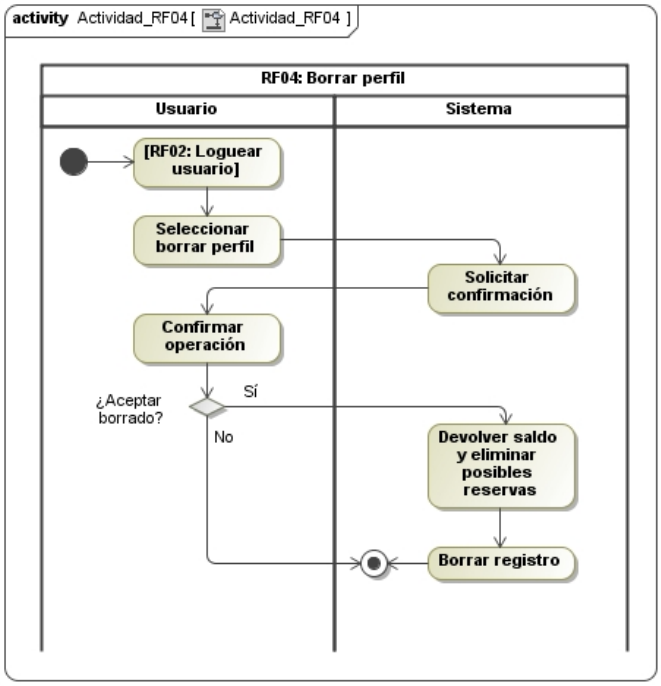

Figura 3.5: Diagrama de actividad de RF04: Borrar perfil

- **RF05: Ingresar saldo**. El sistema deberá permitir el ingreso de dinero, de modo que el usuario cuente con saldo suficiente para que pueda operar con el parque de bicicletas.

  La descripción textual:

RF05: Ingresar saldo	
Usuario	**Sistema**
[Pto. inclusión: RF02: Loguear usuario]	
1. Seleccionar ingreso	
	2. Mostrar interfaz
3. Introducir datos	
	4. Actualizar registro
	5. Confirmar operación terminada

Tabla 3.5: Descripción textual de RF05: Ingresar saldo

Y el diagrama de actividad:

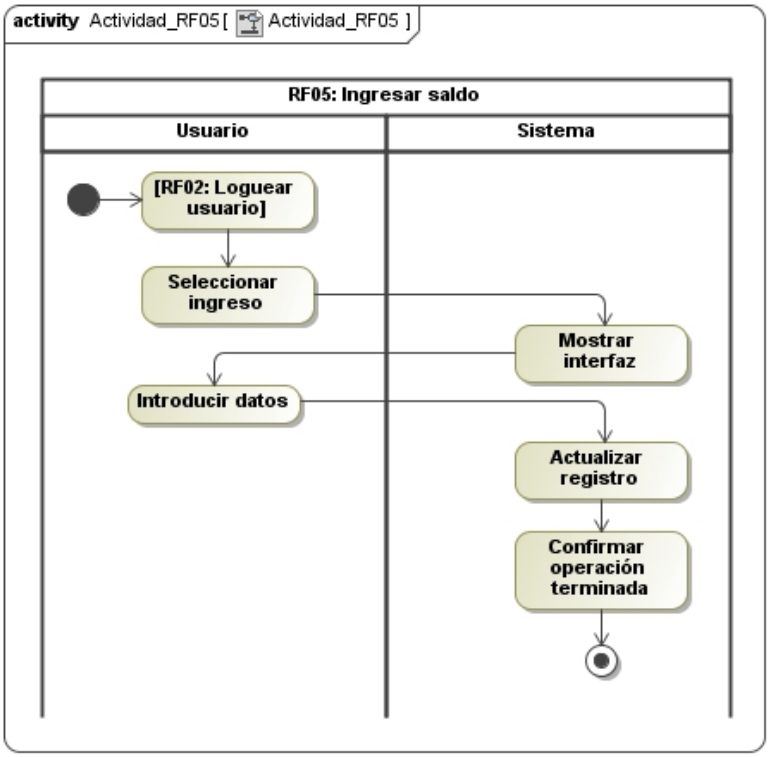

Figura 3.6: Diagrama de actividad de RF05: Ingresar saldo

- **RF06: Coger bicicleta**. El sistema deberá permitir coger bicicletas a los usuarios registrados y logueados en el sistema. Se han tener en cuenta las siguientes restricciones:

  - El usuario sólo puede tener una bicicleta cogida a la vez.
  - El usuario ha de disponer de saldo suficiente para completar la operación.
  - En caso de haber reservado previamente, el usuario sólo podrá coger la bicicleta de la estación en la que haya realizado dicha reserva.
  - Las bicicletas reservadas por otros usuarios no están disponibles.

  Señalar, a su vez, que la estación deberá mostrar una tarifa adaptada a la disponibilidad de bicicletas. Es decir, partiendo de una tarifa base, ésta se verá acrecentada a medida que las bicis disponibles se vean reducidas.

  El objetivo es introducir un primer nivel de adaptación a las condiciones del entorno, de modo que los usuarios tengan alicientes para coger bicicletas de estaciones con una mayor disponibilidad.

  La descripción textual:

RF06: Coger bicicleta	
Usuario	**Sistema**
[Pto. inclusión: RF02: Loguear usuario]	
1. Seleccionar estación	
	2. Mostrar estado estación (incluyen tarifa)
3. Seleccionar coger bici	
	4. Comprobar restricciones
	5. [RF10: Pagar]
	6. Actualizar estado global
	7. Confirmar operación terminada

Tabla 3.6: Descripción textual de RF06: Coger bicicleta

Y el diagrama de actividad:

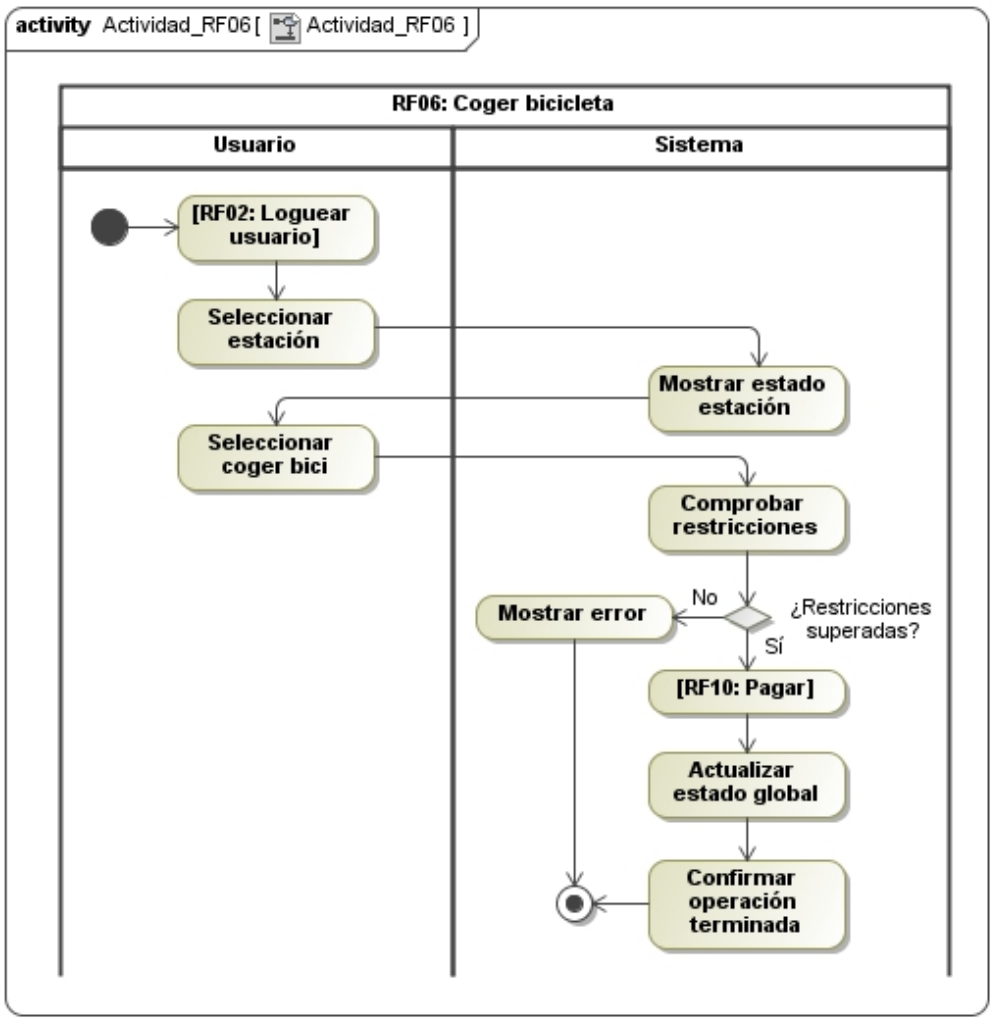

Figura 3.7: Diagrama de actividad de RF06: Coger bicicleta

- **RF07: Dejar bicicleta**. El sistema deberá permitir dejar, en estaciones con disponibilidad suficiente, bicicletas previamente cogidas por usuarios. Se han tener en cuenta las siguientes restricciones:

  - En caso de haber reservado un anclaje previamente, el usuario sólo podrá dejar la bicicleta en la estación en la que haya realizado dicha reserva.
  - Los anclajes reservados por otros usuarios no están disponibles

  La descripción textual:

RF07: Dejar bicicleta	
Usuario	**Sistema**
[Pto. inclusión: RF02: Loguear usuario]	
1. Seleccionar estación	
	2. Mostrar estado estación
3. Seleccionar dejar bici	
	4. Comprobar restricciones
	5. Actualizar estado global
	6. Confirmar operación terminada

Tabla 3.7: Descripción textual de RF07: Dejar bicicleta

Y el diagrama de actividad:

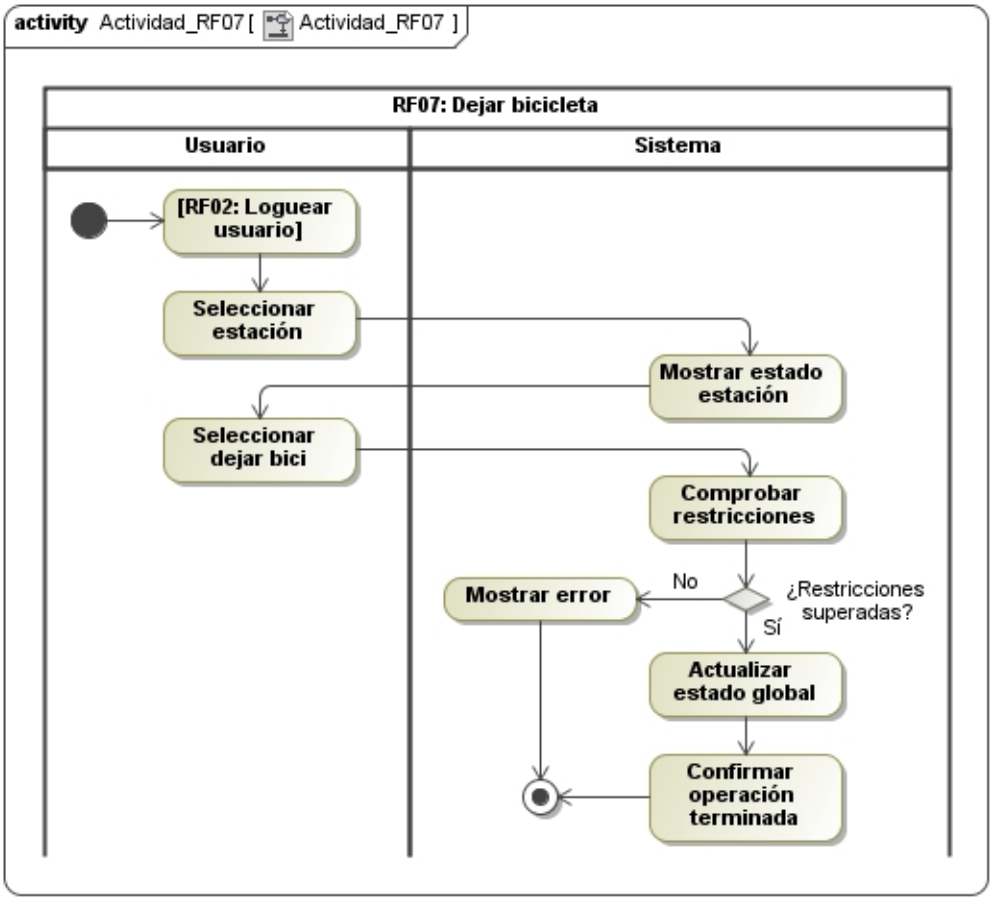

Figura 3.8: Diagrama de actividad de RF07: Dejar bicicleta

- **RF08: Reservar**. El sistema deberá permitir la reserva de recursos, que podrán ser de dos tipos: bicicletas o anclajes. Se han tener en cuenta las siguientes restricciones:
  - El usuario sólo puede tener una reserva de cada tipo a la vez. Es decir, se permiten tener, como mucho, dos reservas simultáneas por usuario: una bicicleta y un anclaje.
  - En caso de tener una bicicleta cogida, el usuario no podrá realizar la reserva de otra.

  La descripción textual:

RF08: Reservar	
Usuario	**Sistema**
[Pto. inclusión: RF02: Loguear usuario]	
1. Seleccionar estación	
	2. Mostrar estado estación
3. Seleccionar reservar	
	4. Mostrar opciones de reserva
5. Seleccionar tipo de reserva	
	6. Comprobar restricciones
	7. Actualizar estado global
	8. Confirmar operación terminada

Tabla 3.8: Descripción textual de RF08: Reservar

Y el diagrama de actividad:

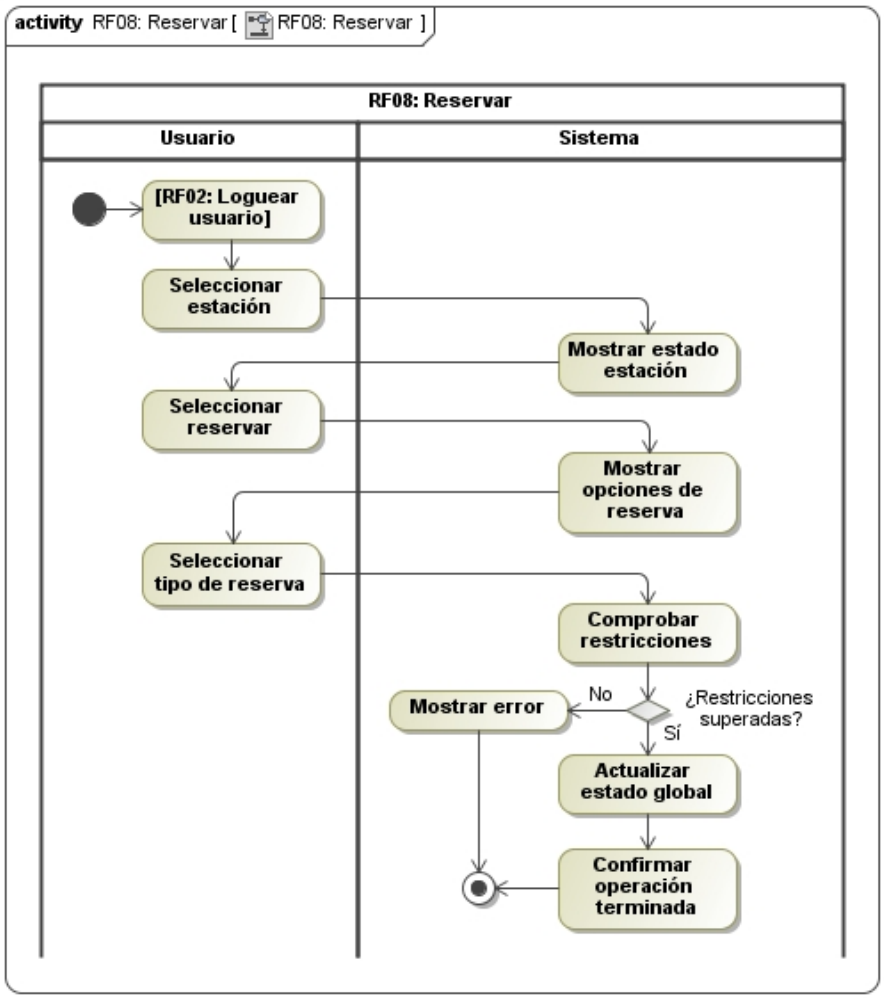

Figura 3.9: Diagrama de actividad de RF08: Reservar

- **RF09: Cancelar reserva**. El sistema deberá permitir la cancelación de reservas, tanto de bicicletas como de anclajes. Esta cancelación se podrá realizar por las siguientes vías:

  - Cancelación explícita por parte del usuario. Este tipo de cancelación implica una confirmación por parte del usuario.
  - Cancelación implícita al cabo de 30 minutos del momento de la reserva.

  Si un usuario ejerce su derecho sobre una reserva, la misma queda automáticamente cancelada.

  Se aporta la descripción textual del requisito referido a la cancelación explíci-

ta, puesto que es la única en la que interviene el usuario de manera directa:

RF09: Cancelar reserva (explícita)	
Usuario	**Sistema**
[Pto. inclusión: RF08: Reservar]	
1. Seleccionar cancelar reserva	
	2. Solicitar confirmación
3. Confirmar operación	
	4. Actualizar estado global
	5. Confirmar operación terminada

Tabla 3.9: Descripción textual de RF09: Cancelar reserva (explícita)

Y el diagrama de actividad:

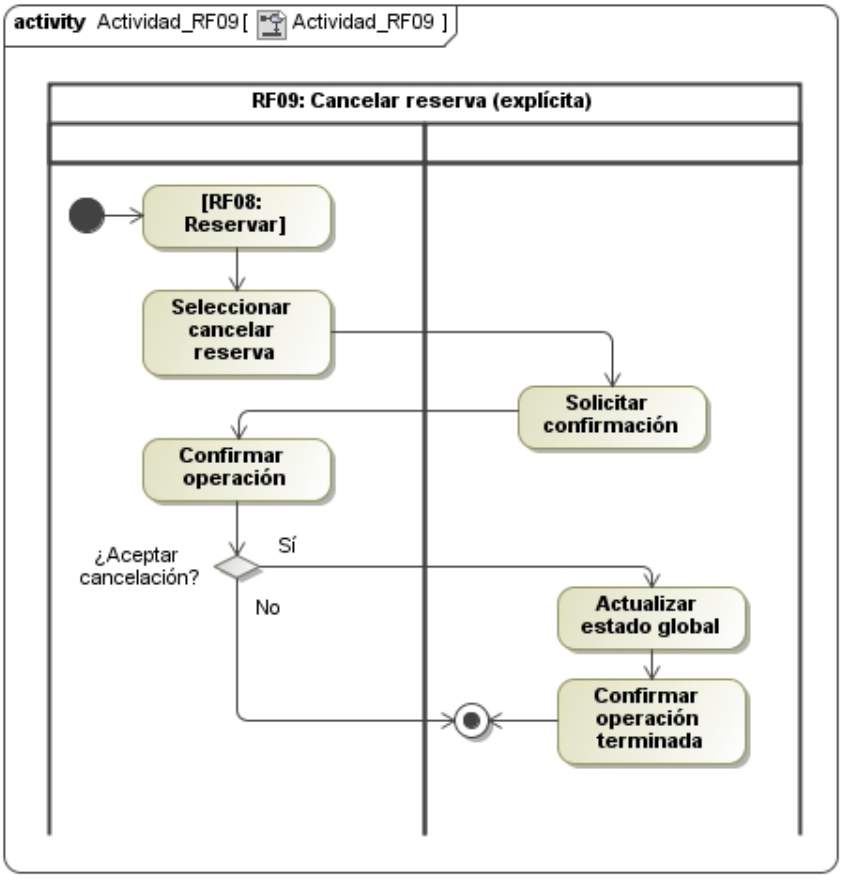

Figura 3.10: Diagrama de actividad de RF09: Cancelar reserva (explícita)

- **RF10: Pagar**. El sistema deberá permitir realizar el pago de las operaciones desarrolladas por los usuarios.

  La descripción textual:

RF10:Pagar	
Usuario	**Sistema**
[Pto. inclusión: RF06: Coger bicicleta]	
	1. Realizar pago
	2. Actualizar usuario
	3. Confirmar operación terminada

Tabla 3.10: Descripción textual de RF10: Pagar

Y el diagrama de actividad:

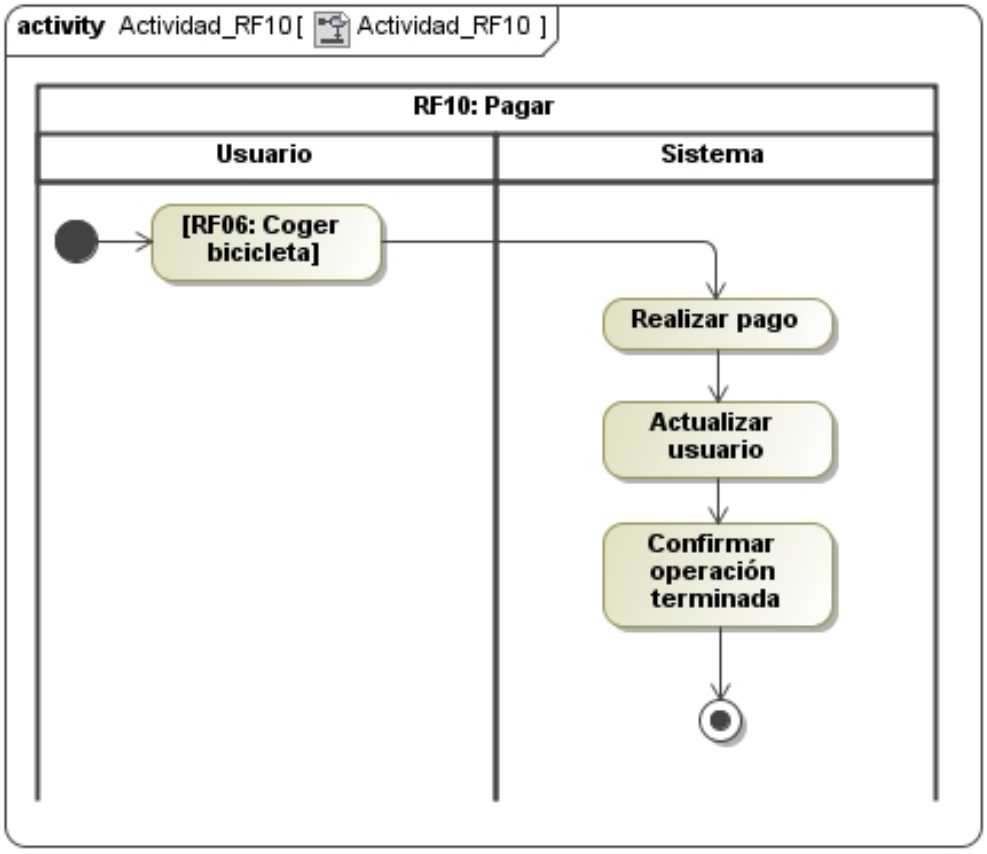

Figura 3.11: Diagrama de actividad de RF10: Pagar

Finalmente, se pasan a detallar los *Requerimientos No Funcionales* más relevantes:

- **RNF01: Usabilidad**. El sistema ha de presentar una interfaz sencilla e intuitiva, respetando las normas de diseño Android y proporcionando una adecuada retroalimentación al usuario de las operaciones terminadas y de posibles errores en su manejo.

- **RNF02: Eficiencia**. El sistema ha de trasladar toda operación posible a la capa de servidor, de modo que el cliente suponga la menor carga posible para el dispositivo móvil y se reduzcan los tiempos de espera.

- **RNF03: Seguridad**. El sistema ha de asegurar la seguridad en el tratamiento de la información del usuario.

  El detalle de este RNF se recoge en el apartado posterior referido a los Atributos del Sistema

- **RNF04: Fiabilidad**. El sistema ha operar según lo esperado y minimizando en la medida de los posible la probabilidad de aparición de fallos. E

  El detalle de este RNF se recoge en el apartado posterior referido a los Atributos del Sistema

- **RNF05: Mantenibilidad**. El sistema se ha de desarrollar de modo que pueda ser conservado y restituido (en caso necesario) con facilidad.

  El detalle de este RNF se recoge en el apartado posterior referido a los Atributos del Sistema

- **RNF06: Portabilidad**. El sistema ha de desarrollarse de modo que pueda ser ejecutado en diferentes plataformas.

  El detalle de este RNF se recoge en el apartado posterior referido a los Atributos del Sistema

- **RNF07: Disponibilidad**. El sistema ha de permanecer en funcionamiento continuo para prestar una servicio adecuado a los usuarios.

  El detalle de este RNF se recoge en el apartado posterior referido a los Atributos del Sistema

Requisitos de Rendimiento

Desde el punto de vista de la aplicación móvil, no se espera una carga excesiva en el dispositivo por gestionar únicamente los datos del usuario registrado.

El servidor, por su parte, ha de ser capaz de dar respuesta a una serie de peticiones concurrentes que pueden escalar a la cantidad de dispositivos que tengan instalada la aplicación.

Finalmente, en relación a la cantidad de registros almacenados en la base de datos, se espera que queden registrados de manera individual tanto usuarios como puestos de bicicletas, además de posibles tablas adicionales.

Restricciones de Diseño

Para el diseño de la aplicación se deberán utilizar componentes compatibles con la versión 4.0 de Android, en caso de no estar disponibles, se deberá recurrir a librerías de soporte para dar servicio a dicha versión, cumpliendo con ella, las restricciones hardware quedan, a su vez, cubiertas.

En relación al servidor de aplicación, éste ha de ser capaz de gestionar accesos concurrentes a recursos compartidos, controlando las posibles condiciones de carrera que puedan aparecer y dando un respuesta oportuna al usuario.

Atributos del Sistema

- Seguridad.
  - El acceso a la aplicación se realizará mediante el par usuario-contraseña.
  - Las claves de usuario deberán almacenarse de manera segura mediante encriptación SHA-1.
  - Desde la aplicación ningún usuario tendrá acceso a la administración interna de la misma, sino que dicha tarea se realizará directamente sobre el servidor o base de datos donde sólo el desarrollador o mantendor de la aplicación podrá acceder.
- Fiabilidad.
  - El sistema ha de quedar adecuadamente testeado previa distribución para obtener una adecuada cobertura de fallos.
  - La interfaz de usuario ha de ser sencilla e intuitiva y, en caso de ser necesario, deberá informar con el resultado de las operaciones para una adecuada experiencia en su uso.
- Mantenibilidad.
  - La aplicación ha de quedar desarrollada siguiendo las buenas prácticas del desarrollo software (código adecuadamente organizado y comentado, utilización de patrones de diseño estandarizados, utilización de *idioms*, etc.).

    - La herramienta deberá dejar *logs* internos de las áreas que se consideren más relevantes o susceptibles a fallos de modo que puedan ser consultados por los desarrolladores para realizar tareas de depuración.

- Portabilidad.

    - La aplicación podrá ser instalada en cualquier dispositivo con sistema operativo Android (versión mínima 4.0).
    - Los servidores de aplicación y base de datos se podrán desplegar sobre cualquiera de los sistemas operativos más extendidos (Windows, Linux, iOS, etc.).
    - Queda como tarea futura la adaptación de la aplicación móvil a entornos web e iOS.

- Disponibilidad.

    - Los servidores han de estar operativos 24x7 para atender las necesidades de uso.

Otros Requisitos

No se consideran otros requisitos de los especificados en secciones previas.

3.2. Análisis

3.2.1. Introducción

El análisis de los requerimientos especificados en la sección anterior ha de dar como resultado la especificación de las características operativas del software, indicando la interfaz de éste y otros elementos del sistema, y estableciendo las restricciones que limitan al software.

Este proceso de análisis otorga la información que se traduce en diseños de arquitectura, interfaz y componentes, así como los medios para evaluar la calidad del software una vez construido.

Por lo tanto, el modelo de análisis supone un puente entre la descripción del sistema y su diseño posterior. Esta relación queda esquematizada en la figura 3.12.

Este modelo se desarrollará en dos pasos:

1. En primer lugar, se presentará la vista estática general del sistema mediante un diagrama de clases de análisis que represente el modelo conceptual de datos. De este modo se pretende obtener una perspectiva global de las entidades básicas presentes en el software.

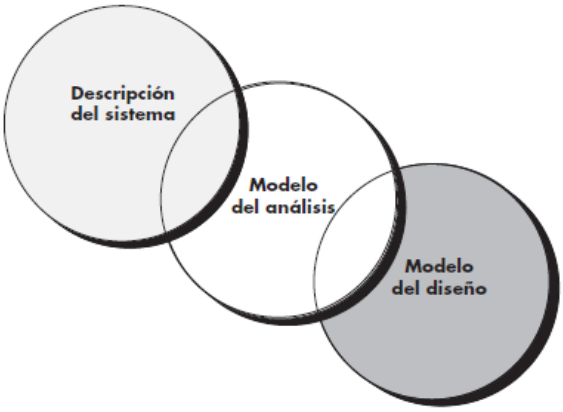

Figura 3.12: El modelo de análisis como puente entre los requerimientos y el diseño del software

2. En segundo lugar, los requisitos descritos en la sección anterior se comenzarán a desarrollar mediante diagramas de colaboración (o comunicación) de modo que se tenga una primera visión de la futura implementación de cada uno.

Cabe señalar que, en este punto, la atención se centra en el *qué*, no en el *cómo*, con lo no se entrará en detalles técnicos, de diseño o de implementación.

3.2.2. Diagrama de clases de análisis

El diagrama de clases de análisis representa el el modelo conceptual de datos, dando lugar a la vista estática general del sistema e identificación de sus entidades básicas. Si bien se cuenta con un cierto grado de subjetividad, las clases potenciales a incluir en este punto deberán cubrir todos (o casi todos) los puntos siguientes, de acuerdo a Pressman [1]:

1. *Información retenida.* Debe recordarse la información sobre la clase para que el sistema pueda funcionar.

2. *Servicios necesarios.* La clase potencial debe tener un conjunto de operaciones identificables que cambien en cierta manera el valor de sus atributos.

3. *Atributos múltiples.*

4. *Atributos comunes.* Para la clase potencial se define un conjunto de atributos y se aplican éstos a todas las instancias de la clase.

5. *Operaciones comunes.* Se define un conjunto de operaciones para la clase potencial y éstas se aplican a todas las instancias de la clase.

6. *Requerimientos esenciales*. Las entidades externas que aparezcan en el espacio del problema y que produzcan o consuman información esencial para la operación de cualquier solución para el sistema casi siempre se definirán como clases en el modelo de requerimientos.

Quedan identificadas las siguientes entidades básicas:

- **Gestor conexión remota**. Desde el momento en el que la aplicación contará con una arquitectura cliente-servidor y la comunicación entre ambas entidades se realizará de manera remota, se requiere de una entidad que gestione dicha comunicación y sepa dar respuesta a las operaciones CRUD.

- **Entidad remota**. La entidad que quedará gestionada por el gestor anterior, supone la generalización de alguna de las siguientes entidades:

  - **Usuario**. Se deberán recoger los atributos básicos para poder realizar el login, como el nombre de usuario y contraseña, datos personales, como el nombre y apellidos, y datos operativos, como el saldo o datos acerca de si se tiene una reserva o bicicleta. Del mismo modo, el usuario deberá ser capaz de interactuar con las estaciones de bicicletas, cogiendo y dejando las mismas o gestionando reservas.
  - **Estación**. La estación de bicicletas, deberá contar con datos identificativos, como la dirección donde se encuentre ubicada, y con datos operativos referidos al total de bicicletas o anclajes disponibles, reservados, etc., así como la tarifa base de esa estación. La estación deberá poder gestionar las operaciones comenzadas por los usuarios y mostrar su estado convenientemente actualizado.
  - **Reserva**. Se considera relevante contar con una clase para almacenar las reservas de usuarios, de modo que su gestión global sea más sencilla. Esta entidad deberá contar con los datos de la reserva (tipo (anclaje o bicicleta), nombre de usuario, fecha y estación) y con la constante que defina el tiempo máximo por reserva (30' de acuerdo a los requisitos).

Este conjunto de entidades básicas junto con sus potenciales atributos y operaciones básicas queda modelizado en la figura 3.13.

3.2.3. Diagramas de colaboración

Aquí se desarrollan los requisitos

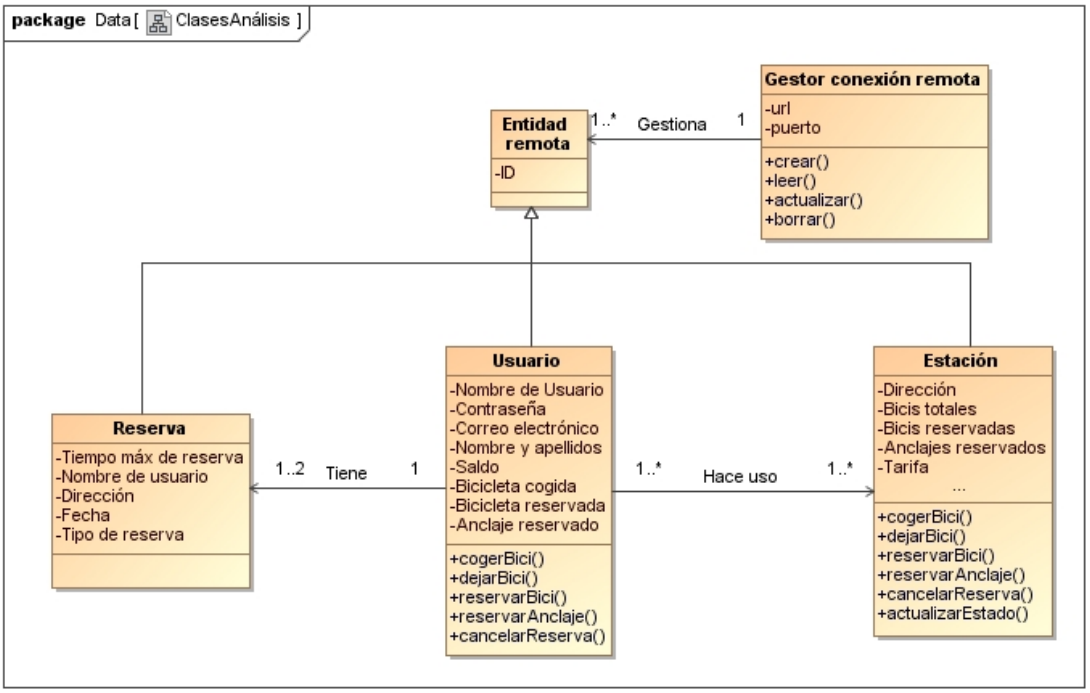

Figura 3.13: Diagrama de clases de análisis

3.3. Diseño

Aquí va la arquitectura, diagramas de clases...

3.4. Implementación

Sin enrollarse mucho con detalles... Herramientas utilizadas... Un diagrama de despliegue a lo mejor...

Capítulo 4

Conclusiones

NO ME GUSTA MUCHO ESTO, NO SÉ SI LAS CONCLUSIONES SE ENFOCAN TANTO A LO PERSONAL

El presente proyecto ha supuesto la elaboración de una aplicación Android completa comunicada de manera remota con su servidor y base de datos mediante el estándar REST de los servicios web.

Dadas las diferentes tecnologías y capas de datos utilizadas, el conocimiento técnico adquirido ha sido amplio, permitiéndome conocer, comparar y seleccionar diferentes soluciones de bases de datos, servidores, transmisión y serialización de datos... dando lugar a una visión final del desarrollo software para dispositivos Android bastante sólida y completa.

Desde el punto de vista de las capacidades personales, cabría destacar la mejora y asentamiento de las capacidades propias de organización, análisis y síntesis, que me han permitido llevar a buen puerto la herramienta desarrollada.

Más allá de los apartados personales, se espera que la herramienta pueda ser utilizada como base de estudio para la mejora de los servicios de gestión de parques públicos de bicicletas... SEGUIR AÑADIENDO COSAS...

4.1. Líneas futuras

Como se ha comentado anteriormente, y de manera general, la aplicación supone una base tecnológica para el desarrollo y mejora de los servicios de gestión de parques públicos de bicicletas.

De manera concreta, se consideran las siguientes líneas futuras de desarrollo y estudio:

- Desarrollo de la adaptación de precios a la hora de coger, dejar o reservar bicicletas, dependiendo de la disponibilidad de cada estación.

- Desarrollo de versiones sobre diferentes plataformas (web, tablet, etc.), con el objetivo de llegar a un público más amplio.

Apéndice A

Manual de usuario

Aquí podemos poner un manual de usuario.

Bibliografía

[1] Pressman, Roger S. *Ingeniería del software: un enfoque práctico*, ed.7, McGrawHill, 2010

[2] Oetiker T. et al, *The Not So Short Introduction to LaTeX 2ε*, Version 5.05, 2015.

[3] *IEEE Std. 830-1998: Especificaciones de los Requisitos del Software*,