Implementación de aplicación para automatizar procesos

Paseo Virtual Biblioteca UQ - Biblioteca Universidad del Quindío

SAD-Documento de Arquitectura del proyecto

Versión 1.0

Historial de Revisiones

|  |  |  |  |
| --- | --- | --- | --- |
| **Fecha** | **Versión** | **Descripción** | **Autor** |
| 23/03/2017 | 1.0 | Definición de funcionalidades arquitectónicas. | Jhoan Sebastián Gómez Medina.  Jonh Sebastián Agudelo Ospina |
| 25/03/2017 | 2.0 | Se establecen las herramientas y framework que no serán utilizados en el proyecto y se explica el porqué. | Jonh Sebastián Agudelo Ospina |
| 25/03/2017 | 3.0 | Se definen las herramientas a utilizar en el proyecto y se describen sus características. | Jhoan Sebastián Gómez Medina. |
| 25/03/2017 | 4.0 | Definición y delimitación de los atributos de calidad a utilizar. | Jhoan Sebastián Gómez Medina.  Jonh Sebastián Agudelo Ospina |
| 25/03/2017 | 5.0 | Se describen los escenarios considerados para los principales y más importantes atributos de calidad del proyecto. | Jhoan Sebastián Gómez Medina.  Jonh Sebastián Agudelo Ospina |
| 25/03/2017 | 6.0 | Drivers de arquitectura. | Jhoan Sebastián Gómez Medina. |
| 25/03/2017 | 7.0 | Patrones arquitectónicos. | Jhoan Sebastián Gómez Medina. |
| 25/03/2017 | 8.0 | Desarrollo sección 4+1 vistas arquitectónicas. Diagramas, Mockups (posibles diseños), estructura del proyecto. | Jhoan Sebastián Gómez Medina.  Jonh Sebastián Agudelo Ospina |

Tabla de Contenidos

1. Introducción 5

2. Propósito 5

3. Alcance 5

4. Funcionalidades Arquitectónicas 5

5. Decisiones, limitaciones y justificaciones 6

5.1. Componentes descartados 6

5.2. Componentes y frameworks elegidos (Restricciones del Sistema) 7

6. Atributos de Calidad 9

6.1. Requerimientos Funcionales (RF) 10

6.2. Requerimientos no funcionales (RNF) 11

7. Escenarios de Atributos de Calidad 12

7.1. Escenarios de Mantenibilidad 12

7.2. Escenario de Extensibilidad 14

7.3. Escenario de Disponibilidad 15

7.4. Escenario de Confiabilidad 16

7.5. Escenario de Usabilidad 17

8. Vistas Arquitectónicas 18

8.1. Use cases view: 18

8.1.1. Actores 19

8.2. Logical view: 19

8.3. Implementation view: 23

8.4. Deployment view: 24

8.5. Process view: 25

9. Objetivos y limitaciones de la arquitectura 26

10. Procesos de la arquitectura 26

11. Patrón o framework arquitectónico a utilizar 27

11.1. Patrón MVC (Model View Controller) 27

11.2. Microkernel 29

12. Tácticas Arquitectónicas 30

13. Trazabilidad de los Drivers arquitectónicos 32

13.1. Tácticas o patrones que satisfacen los drives más importantes del proyecto. 33

13.2. Lista completa de driver arquitectónicos 34

**Contenidos de Figuras**

[Figura 1: Estructura de los escenarios de atributos de calidad. 12](#_Toc478317922)

[Figura 2: Caso de uso que describe funcionalidades principales de la APP. 19](#_Toc478317923)

[Figura 3: Actor principal identificado para la APP que puede desarrollar varias actividades. 19](#_Toc478317924)

[Figura 4: Vista de alto nivel sobre organización de la capa lógica de la aplicación en sus módulos. 20](#_Toc478317925)

[Figura 5: Paquete principal que contiene la estructura del proyecto. 20](#_Toc478317926)

[Figura 6: Organización en paquetes del proyecto. 21](#_Toc478317927)

[Figura 7: Producciones con las cuales se desplegará la producción del proyecto. 21](#_Toc478317928)

[Figura 8: Componentes para la construcción de la capa de presentación (Front - End). 21](#_Toc478317929)

[Figura 9: Organización de lo contenido en la capa de negocio o capa lógica. 22](#_Toc478317930)

[Figura 10: Capa lógica de la aplicación. 22](#_Toc478317931)

[Figura 11: consumo de servicios de la aplicación. 23](#_Toc478317932)

[Figura 12: Diseño-vista de alto nivel sobre el modelo de la implementación del proyecto. 23](#_Toc478317933)

[Figura 13: Modelo (Mockup) de la aplicación. 24](#_Toc478317934)

[Figura 14: Deployment de la aplicación. 25](#_Toc478317935)

[Figura 15: Diseño y modelo de la Estructura de los procesos a construir. 25](#_Toc478317936)

[Figura 16: Patrón principal MVC. 28](#_Toc478317937)

[Figura 17: Descripción de la organización del patrón MVC en el royecto. 28](#_Toc478317938)

[Figura 18: procesos del patrón MVC en la aplicación. 29](#_Toc478317939)

[Figura 19: Esquema principal del patrón Microkernel utilizado en la construcción del proyecto 30](#_Toc478317940)

**Contenido de Tablas**

[Tabla 1: Escenario 1 del atributo de calidad de mantenibilidad. 14](#_Toc478317941)

[Tabla 2: Escenario 2 del atributo de calidad de mantenibilidad. 15](#_Toc478317942)

[Tabla 3: Escenario 3 del atributo de calidad de mantenibilidad. 15](#_Toc478317943)

[Tabla 4: Escenario del atributo de calidad de Extensibilidad. 16](#_Toc478317944)

[Tabla 5: Escenario del atributo de calidad de Disponibilidad. 17](#_Toc478317945)

[Tabla 6: Escenario 1 del atributo de calidad de Confiabilidad. 17](#_Toc478317946)

[Tabla 7: Escenario 3 del atributo de calidad de Confiabilidad. 18](#_Toc478317947)

[Tabla 8: Escenario del atributo de calidad de Usabilidad. 19](#_Toc478317948)

[Tabla 9: Tácticas de arquitectura aplicadas al proyecto. 33](#_Toc478317949)

[Tabla 10: Tácticas o patrones que satisfacen los drives más importantes del proyecto 35](#_Toc478317950)

[Tabla 11: Lista completa de driver arquitectónicos. 37](#_Toc478317951)

Paseo Virtual CRAI

Architecture Notebook

# Introducción

El documento de arquitectura de software, contiene los puntos más relevantes del proyecto en cuanto a establecer técnicas y los medios que encaminan al cumplimiento de los requisitos (No funcionales y Funcionales) establecidos previamente.

# Propósito

Este documento describe la filosofía, las decisiones, las restricciones, las justificaciones, los elementos significativos y cualquier otro aspecto general del sistema que moldea el diseño y la implementación.

Este documento está destinado a captar y transmitir las decisiones arquitectónicas significativas que se hayan introducido en el diseño y la construcción del sistema. Es un camino por el cual el arquitecto y otras personas involucradas en el proyecto pueden comprender mejor los problemas que hay que resolver y cómo se representan con este sistema.

# Alcance

El alcance del presente documento está previsto para permitir llegar a un diseño de sistema que además de dar solución a cada una de las necesidades previstas, brindará una guía a seguir dependiendo de los requisitos ya sean funcionales o no funcionales de la aplicación.

# Funcionalidades Arquitectónicas

Previo a realizar el diseño arquitectónico de la aplicación, Se necesitan definir las funcionalidades arquitectónicas con base en el modelo, la actividad de negocio y los atributos de calidad que debe poseer la aplicación. Las funcionalidades arquitectónicas como cimiento importante para definir la arquitectura del sistema, se determinan en consideración de los servicios de los módulos principales que tendrá la aplicación. Los módulos que integran la aplicación, se definen así:

* **Módulo - Paseo Virtual:** Este módulo será el encargado de mostrar información al usuario por medio de la cámara del dispositivo móvil mediante la técnica de la realidad aumentada. La información que se mostrará depende de la imagen (target) a la cual apunte la cámara del dispositivo móvil, dichas imágenes se ubicarán estratégicamente en la biblioteca central de la Universidad del Quindío.
* **Módulo - Buscar:** Este módulo localizara un libro en un estante de la biblioteca CRAI de la universidad del Quindío, la búsqueda se realizará por medio de su código único de identificación.
* **Módulo - Localizar libro:** Luego de haber realizado una búsqueda de un libro por parte de un usuario, se mostrará la ubicación del libro en una representación 2D de la distribución interna de los estantes de la biblioteca, donde, por medio de un apuntador se identificará la ubicación.

# Decisiones, limitaciones y justificaciones

Teniendo las funcionalidades se establece los requisitos del producto.

* IDE Android: Debido a que la aplicación será nativa del sistema operativo Android se requiere el uso de un IDE para esta plataforma.
* Framework de AR: Se establece la necesidad de la utilización de un framework especializado en la realidad aumentada.
* Software modelado 3D: Debido a la característica de la realidad aumentada se establece el requerimiento de modelar objetos en un espacio tridimensional.

A continuación se establecerá los componentes que se evaluaron para el desarrollo de la aplicación donde estarán clasificados en dos categorías, componentes elegidos con sus justificaciones y los componentes descartados.

## Componentes descartados

Los componentes que se encuentran a continuación fueron descartados porque no cumplen todas las necesidades para el desarrollo de la aplicación así como no tienen compatibilidad para el desarrollo en conjunto o la disposición de un SDK.

* **ARToolKit:** El Kit de herramientas de Realidad Aumentada (ARToolKit) captura imágenes de fuentes de video, marca de forma óptica los marcadores en las imágenes y las comprime con contenido generado por computadora usando OpenGL. Licencia dual, bajo la GPL, más comercialmente por ARToolworks, Inc.
* **Arneo:** Es un motor de realidad aumentada, Arneo puede identificar elementos 2D como 3D, con su interfaz intuitiva que permite transponer fácilmente un catálogo digital en el mundo real.
* **Blender:** Es un [programa informático](https://es.wikipedia.org/wiki/Programa_inform%C3%A1tico) multiplataforma, dedicado especialmente al modelado, iluminación, [renderizado](https://es.wikipedia.org/wiki/Renderizado), animación y creación de gráficos [tridimensionales](https://es.wikipedia.org/wiki/Tridimensional).

* **Amazon SimpleDB**: Amazon SimpleDB es una base de datos distribuida escrito en Erlang por Amazon.com. Se utiliza como un servicio web en concierto con Amazon Elastic Compute Cloud (EC2) y Amazon S3 y es parte de Amazon Web Services.
* **Eclipse:** El IDE de desarrollo eclipse hace uso de un plugin de ADT donde se permite programar para Android, aunque con algunas desventajas si es comparado con lo que ofrece android studio y todo su entorno de construcción de aplicaciones. Eclipse no ofrece la rapidez en renderizar pantallas para mostrar al desarrollador además de que sus actualizaciones no están evolucionando de acuerdo a como lo hace Android Studio.

## Componentes y frameworks elegidos (Restricciones del Sistema)

Las restricciones del sistema son todas aquellas condiciones bajo las cuales la plataforma debe funcionar y/o interactuar, de tal manera que se consiga una adaptación que continúe satisfaciendo los drivers de arquitectura.

Entre las restricciones que se tienen para la aplicación se encuentran:

* **Android SDK**: Android SDK contiene todo lo necesario para comenzar a crear aplicaciones. El SDK de Android también viene con un dispositivo virtual emulado que es totalmente funcional para realizar pruebas.
* **Vuforia SDK**: Para la aplicación, sus funcionalidades de realidad aumentada serán desarrolladas mediante la herramienta **Vuforia Unity**, lo cual ofrece el framework “vuforia SDK” para construir aplicaciones para dispositivos móviles Android y de este modo poder trabajar conjuntamente en su construcción haciendo uso del IDE de desarrollo Android Studio. Las principales razones por las cuales se selecciona esta herramienta y se describen a continuación (Descripción del por qué):
  + Vuforia es la plataforma líder de AR.
  + *Fiabilidad (Atributo de calidad Seguridad para el proyecto)*: Vuforia ofrece la mejor visión de equipo en su clase, asegurando experiencias robustas y confiables en una variedad de ambientes.
  + *Empoderamiento Creativo*: Vuforia ofrece a los desarrolladores la libertad creativa para crear experiencias únicas que reflejen las marcas e impulsen los resultados empresariales.
  + *Máximo alcance*: Vuforia soporta teléfonos líderes, tablets y lentes digitales en Android, iOS y UWP.
  + *Visión Avanzada*: Vuforia puede reconocer una gama de imágenes, objetos y entornos cotidianos.
  + Compatibilidad con Unity
* **Android Studio**: Es el entorno de desarrollo integrado (IDE) para el desarrollo de la aplicación, este IDE se basa en IntelliJ IDEA.

Hacer uso de Android Studio, en cuanto a la construcción de la aplicación contribuye a satisfacer un conjunto de requisitos definidos para el proyecto, donde, es una de las razones primarias de su selección, además, esta herramienta IDE permite construir aplicaciones móviles con las mejores prácticas de desarrollo tal como lo establece Google en su esencia de las APP móviles.

Android Studio ofrece funciones que aumentan tu productividad durante la compilación de apps para Android, como las siguientes:

* + Sistema de compilación flexible basado en Gradle.
  + Un emulador rápido con varias funciones.
  + Un entorno unificado en el que puedes realizar desarrollos para todos los dispositivos Android.
  + Instant Run, para aplicar cambios mientras tu app se ejecuta sin la necesidad de compilar un nuevo APK.
  + Apoyo a la “Gestión de proyecto”: Integración de plantillas de código y GitHub, para ayudarte a compilar funciones comunes de las apps e importar ejemplos de código.
  + Herramientas Lint para detectar problemas de rendimiento, uso, compatibilidad de versión, mantenibilidad, etc.
  + Compatibilidad con el framework de Unity Vuforia.
* **Gradle**: En la elección de Usar Android Studio para el desarrollo del proyecto es el entorno de trabajo con “Gradle”, un sistema de compilación que reúne en un uno solo las mejores prestaciones de otros sistemas de compilación. Está basado en JVM (Java Virtual Machine), permitiendo así, escribir código en java, y Android Studio lo entenderá y lo usará.

Gradle es que es un plugin, lo que facilita su actualización y su exportación de un proyecto a otro. Esto significa que es posible distribuir la aplicación a otros entornos fácilmente.

* **Unity:** Es un [motor de videojuego](https://es.wikipedia.org/wiki/Motor_de_videojuego) multiplataforma creado por Unity Technologies. Unity está disponible como plataforma de desarrollo para [Microsoft Windows](https://es.wikipedia.org/wiki/Microsoft_Windows), [OS X](https://es.wikipedia.org/wiki/OS_X). La plataforma de desarrollo tiene soporte de compilación con diferentes tipos de plataformas. Unity es, sin lugar a dudas, el motor de juegos favorito del mundo para la creación de juegos para dispositivos móviles. ¿Por qué es tan popular?
* Despliegue de contenido con un solo clic en Android, iOS, Windows Phone, Tizen y Fire OS
* Cientos de optimizaciones gracias a prestaciones como occlusion culling, asset bundling y build size stripping (reducción del tamaño de la compilación).
* Compatibilidad con el SDK de Vuforia.
* Herramientas y flujos de trabajo 3D y 2D dedicados y fáciles de usar.
* **Consumo de servicios**:

La aplicación, para llevar a cabo el correcto desempeño de sus funcionalidades estará consumiendo servicios web necesarios para la conexión con bases de datos de “Vuforia” en cuanto a los targets usados en la AR.

* **Firebase:** (DBaaS) Firebase es una plataforma móvil que te permite desarrollar apps de alta calidad con rapidez, aumentar la cantidad de usuarios. Firebase contiene funciones complementarias que puedes combinar y adaptar según tus necesidades.
  + Cuando tu app sea un éxito, no tendrás que preocuparte por el escalamiento del código de tu servidor ni por ofrecer capacidad adicional; Firebase se encargará de eso.
  + Implementar Firebase es rápido y fácil. Gracias a las API intuitivas incluidas en un solo SDK, puedes concentrarte en resolver los problemas de tus clientes sin perder tiempo compilando infraestructuras complejas.
* **Java SE Development Kit 8**: Java JDK es un entorno de desarrollo para crear aplicaciones, applets y componentes utilizando el lenguaje de programación Java.

El JDK incluye herramientas útiles para desarrollar y probar programas escritos en el lenguaje de programación Java y se ejecuta en la plataforma Java haciendo uso de su máquina virtual.

El uso del JDK de java se debe a que el lenguaje de programación Android soportado por Google es basado en Java y, los IDE actuales de Android requieren debido a su constante evolución actualizaciones y los últimos JDK.

# Atributos de Calidad

Definir los atributos de calidad para el proyecto y su desarrollo, requiere que arquitectónicamente se tomen ciertas decisiones para que éstos sean cumplidos adecuadamente. Teniendo una mayor abstracción, se deben definir a partir de los requisitos del proyecto las vistas de restricciones, estructura y procesos a desarrollar.

Teniendo en cuenta las metodologías arquitectónicas, se determinan los siguientes atributos de calidad para que a partir de éstos se determine los de mayor importancia y a satisfacer en el proyecto, teniendo en cuenta de que todos son importantes, algunos en mayor o menor medida.

* Manifiestas:
  + Performance
  + Confiabilidad
  + Disponibilidad
* Operacionales:
* Throughput
* Desarrollo:
* Learnability
* Accesibilidad
* Usabilidad
* Evolutiva:
* Mantenibilidad
* Extensibilidad
* Flexibilidad

Según lo anterior y, teniendo en cuenta las funcionalidades arquitectónicas se determinan para la aplicación principalmente los atributos de calidad como atributos claves para el proyecto:

* Mantenibilidad
* Usabilidad
* Learnability
* Confiabilidad
* Extensibilidad
* Disponibilidad

Estos atributos de calidad definidos para el proyecto, fueron seleccionados considerando tanto los principales requerimientos funcionales como los no funcionales del proyecto, citados y descritos a continuación.

## Requerimientos Funcionales (RF)

El desarrollo del sistema se basa en una aplicación para realizar reconocimiento de los elementos y lugares de la biblioteca mediante realidad aumentada, la búsqueda y localización de libros.

* Cuando un usuario realice la búsqueda de un libro se localizara el estante que lo contiene y se mostrará al usuario su ubicación.

* La aplicación permitirá el acceso a la cámara de los dispositivos móviles.

* La aplicación mostrará los resultados de las consultas ejecutadas por el usuario.

* La aplicación funcionará correctamente ante el presente incremento de usuarios.

* La aplicación mostrará mediante AR información importante sobre elementos y lugares que componen la biblioteca.

* La aplicación será documentada y será posible añadir nuevas funcionalidades posteriormente.
* Debe consumir, mostrar, toda la información de los servicios web desde la app.

## Requerimientos no funcionales (RNF)

Los siguientes requerimientos no funcionales brindan detalles clave para considerar restricciones que debe cumplir el sistema, para que por consiguiente sean tenidos en cuenta en el momento de realizar el respectivo modelado.

* El sistema debe estar disponible durante el horario de atención del público por parte de la biblioteca. Lunes a viernes 7:00 a.m. a 09:00 p.m. sábados de 8:00 a.m. a 6:00 p.m.

* Aplicación implementada siguiendo el reglamento de Material Design para aplicaciones Android propuesto por Google en cuanto a cómo debe ser la “usabilidad” de las mismas.
* Las pantallas diseñadas según el “Estándar de Google” para el desarrollo de aplicaciones móviles en este caso Android, son fáciles de usar por parte de la comunidad universitaria. Para esto se tiene en cuenta:
  + La ubicación de los botones, menús y demás complementos.
  + Combinación de colores.
  + Alertas-Mensajes mostrados al usuario.
  + íconos.

* Los colores usados en la aplicación deberán ser referentes a la universidad.

* La aplicación será compatible para versiones Android desde la API 19 en adelante (versión Android 4.4 hasta la actual).

* La aplicación tendrá en cuenta las características de seguridad y privacidad, en cuanto a las condiciones de uso adecuadas de las bases de datos.

* Se debe informar al usuario cuando el dispositivo no cuente con conexión a internet, y se deben inhabilitar las opciones que dependen de esta conexión.
* Se debe contar con una interfaz principal que permita la navegación entre las diferentes opciones de las app.
* Cada vez que se presione un botón se debe direccionar rápidamente al módulo solicitado.

# Escenarios de Atributos de Calidad

A continuación, se describen los escenarios de atributos de calidad considerados más importantes para el desarrollo de la aplicación, para esto se tendrán en cuenta los atributos de calidad de mantenibilidad, extensibilidad, confiabilidad, usabilidad y disponibilidad.

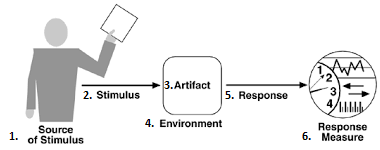


Figura : Estructura de los escenarios de atributos de calidad.

## Escenarios de Mantenibilidad

**7.1.1.** Dado el caso de que se vaya a implementar una nueva versión de la aplicación se deberá garantizar la compatibilidad con al menos dos versiones inmediatamente anteriores de cada uno de los usuarios tal y como se define en fases iniciales de la concepción del mismo (Android).

|  |  |
| --- | --- |
| **Fuente del Estímulo** | Equipo de desarrollo. |
| **Estímulo** | Liberar a producción una nueva versión del sistema. |
| **Artefacto** | Toda la aplicación. |
| **Entorno** | Etapa de preproducción de una nueva versión. |
| **Respuesta del estímulo** | Liberación de la nueva versión del sistema. |
| **Medida de la Respuesta** | La nueva versión es compatible con al menos dos versiones inmediatamente anteriores. (# de versiones compatibles) |

Tabla : Escenario 1 del atributo de calidad de mantenibilidad.

**7.1.2** La dirección de la biblioteca o procesos administrativos de la universidad del Quindío implementará una nueva medida en cuanto a la organización o ubicación de los recursos (libros) de la biblioteca, lo que impactará la aplicación, por tal razón, esta necesitará adaptarse a dicha reforma en mínimo 1 mes y máximo 2 meses.

|  |  |
| --- | --- |
| **Fuente del Estímulo** | La dirección de la biblioteca o procesos administrativos de la universidad del Quindío |
| **Estímulo** | Nueva reforma en cuanto a la organización o ubicación de los recursos (libros) de la biblioteca. |
| **Artefacto** | Toda la aplicación. |
| **Entorno** | Condiciones normales de operación. |
| **Respuesta del estímulo** | Adaptación del sistema a las nuevas medidas administrativas. |
| **Medida de la Respuesta** | Mínimo 1 mes y Máximo 2 meses. |

Tabla : Escenario 2 del atributo de calidad de mantenibilidad.

**7.1.3** El sistema debe proporcionar la posibilidad de modificar, agregar o eliminar funcionalidades según lo requiera el negocio sin impactar el normal funcionamiento del mismo, es decir, los cambios no deben afectar la capa lógica de la aplicación.

|  |  |
| --- | --- |
| **Fuente del Estímulo** | Cambio de naturaleza del negocio. |
| **Estímulo** | Modificar, agregar o eliminar funcionalidades del sistema para que éste se adapte a las realidades y necesidades del negocio. |
| **Artefacto** | Bases de datos y módulos donde se presenta la información asociada al cambio. |
| **Entorno** | Bajo condiciones normales. |
| **Respuesta del estímulo** | Funcionalidades modificadas sin afectar en el sistema. |
| **Medida de la Respuesta** | Tiempo X a definir según la naturaleza del cambio. |

Tabla : Escenario 3 del atributo de calidad de mantenibilidad.

## Escenario de Extensibilidad

**7.2.1** La biblioteca desea promocionar un evento por medio de la funcionalidad de paseo virtual de la aplicación usando la realidad aumentada.

|  |  |
| --- | --- |
| **Fuente del Estímulo** | El stakeholder - Dirección de la biblioteca. |
| **Estímulo** | Solicitud de adición de características. |
| **Artefacto** | Paseo virtual. |
| **Entorno** | Condiciones normales |
| **Respuesta del estímulo** | Acceso a los archivos de desarrollo. |
| **Medida de la Respuesta** | Inmediata. |

Tabla : Escenario del atributo de calidad de Extensibilidad.

## Escenario de Disponibilidad

**7.3.1** Existe un crecimiento exponencial de los usuarios que usan la plataforma y se hace necesario que la disponibilidad y el rendimiento de la misma no se vean afectados por dicho crecimiento.

|  |  |
| --- | --- |
| **Fuente del Estímulo** | Incremento de usuarios. |
| **Estímulo** | Peticiones y transacciones a la Base de datos mayores a las usuales. |
| **Artefacto** | Todo el sistema. |
| **Entorno** | Condiciones normales de uso del sistema. |
| **Respuesta del estímulo** | Utilización de mecanismos de gestión de peticiones (uso de balanceadores de carga en cuanto al consumo de servicios consumidos por la aplicación). |
| **Medida de la Respuesta** | Tiempos de respuesta y nivel de disponibilidad del sistema se mantienen estables. |

Tabla : Escenario del atributo de calidad de Disponibilidad.

## Escenario de Confiabilidad

**7.4.1** Para los usuarios que utilizan la aplicación para realizar búsquedas de ubicación de libros en la biblioteca CRAI de la Universidad del Quindío la aplicación por medio de la información contenida en bases de datos y lo que necesita el usuario le muestra exactamente el lugar al cual debe desplazarse mediante un entorno 2D.

|  |  |
| --- | --- |
| **Fuente del Estímulo** | El usuario final de la aplicación |
| **Estímulo** | Búsqueda de información (ubicación de libros) |
| **Artefacto** | Peticiones y transacciones a la Base de datos de la aplicación. |
| **Entorno** | Condiciones normales de uso del sistema en la. |
| **Respuesta del estímulo** | La aplicación luego de realizada la consulta del usuario muestra la opción de ver la ubicación exacta del recurso consultado en la biblioteca mediante un entorno 2D. |
| **Medida de la Respuesta** | Inmediata. La APP indica el lugar al cual debe dirigirse el usuario sin indicar lugares erróneos. |

Tabla : Escenario 1 del atributo de calidad de Confiabilidad.

**7.4.2** Para los usuarios que utilizan la aplicación para realizar un paseo virtual (mediante realidad aumentada) en la biblioteca CRAI de la Universidad del Quindío, la aplicación por medio de la información contenida en bases de datos (targets contenidos en Vuforia unity) y los identificadores ubicados en el espacio de la biblioteca muestra al usuario información exacta y relevante de dicho sitio.

|  |  |
| --- | --- |
| **Fuente del Estímulo** | El usuario final de la aplicación |
| **Estímulo** | Realizar un paseo virtual (mediante realidad aumentada) |
| **Artefacto** | Uso del módulo de Realidad aumentada. |
| **Entorno** | Condiciones normales de uso del sistema en la biblioteca CRAI de la Universidad del Quindío. |
| **Respuesta del estímulo** | La aplicación muestra a los usuarios información relevante sobre sitios de la biblioteca que sean enfocados en la cámara del dispositivo usado por el usuario y que tengan identificadores establecidos para el correcto funcionamiento de la aplicación. |
| **Medida de la Respuesta** | Inmediata. LA APP muestra información de acuerdo al sitio que se requiere. |

Tabla : Escenario 3 del atributo de calidad de Confiabilidad.

## Escenario de Usabilidad

**7.5.1** Estudiantes nuevos usan por primera vez la aplicación.

|  |  |
| --- | --- |
| **Fuente del Estímulo** | Estudiantes. |
| **Estímulo** | Uso de la aplicación. |
| **Artefacto** | Aplicación |
| **Entorno** | Condiciones normales |
| **Respuesta del estímulo** | La aplicación presentará un estructura simple y fácil de utilizar |
| **Medida de la Respuesta** | El usuario debe aprender a utilizar la aplicación en el menor tiempo posible |

Tabla : Escenario del atributo de calidad de Usabilidad.

# Vistas Arquitectónicas

A continuación se realizará un breve descripción de las vistas arquitectónicas tal como se aplicarán en el desarrollo del proyecto, que actuarán como una “columna vertebral” de todo el entorno de la aplicación y, de esta forma estará velando por unas correctas prácticas y medidas de construcción.

Para modelar, implementar y documentar un sistema requiere verlo desde diferentes perspectivas. Debido a esto, la arquitectura de la aplicación estará representada en un enfoque de cinco vistas: Vista de caso de uso, la vista Diseño, vista de proceso, Vista de componentes, y Despliegue. La siguiente es una breve descripción de cada uno de los puntos de vista:

## Use cases view:

La aplicación Paseo Virtual CRAI contará con dos casos de uso con los cuales el estudiante puede realizar diferentes tareas describidas en las funcionalidades arquitectónicas (Punto 4).

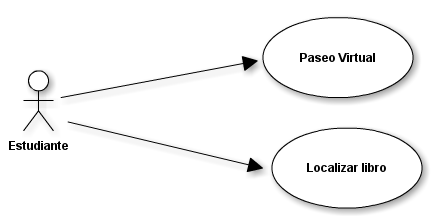


Figura : Caso de uso que describe funcionalidades principales de la APP.

## Actores

Como se describe en el diagrama de casos de uso, los actores presentes en el sistema de gestión son los siguientes:



Figura : Actor principal identificado para la APP que puede desarrollar varias actividades.

*Estudiante*: Tiene la posibilidad de llevar a cabo la acción de realizar el paseo virtual haciendo uso de un teléfono móvil en la biblioteca y realizar la localización de un libro consultado para obtener su localización exacta dentro de dicho espacio físico

## Logical view:

Esta vista contiene la definición del sistema y los objetos que representan los servicios que el sistema proporcionará a sus usuarios finales.

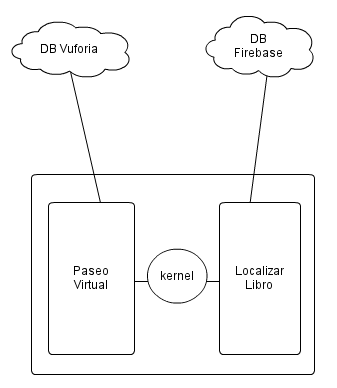


Figura : Vista de alto nivel sobre organización de la capa lógica de la aplicación en sus módulos.

## Estructura del proyecto

* + - 1. **Paquete principal de proyecto**:

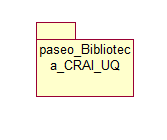


Figura : Paquete principal que contiene la estructura del proyecto.

* + - 1. **Estructura lógica de interfaz de usuario y lógica de dominio:**

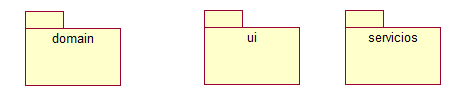


Figura : Organización en paquetes del proyecto.

* + - 1. **Producciones principales:**

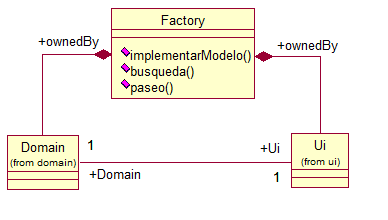


Figura : Producciones con las cuales se desplegará la producción del proyecto.

* + - 1. **Interfaz de usuario (módulos principales, capa de presentación):**

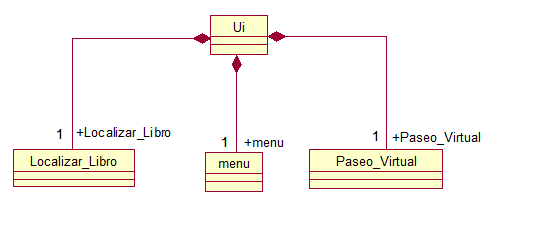


Figura : Componentes para la construcción de la capa de presentación (Front - End).

* + - 1. **Paquetes principales a considerar en el modelado de dominio:**

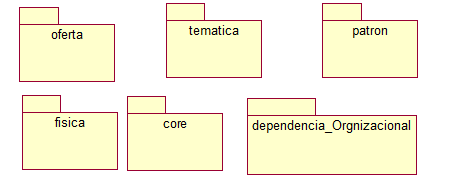


Figura : Organización de lo contenido en la capa de negocio o capa lógica.

* + - 1. **Organización, clases principales en el modelado de dominio (capa lógica de la aplicación):**

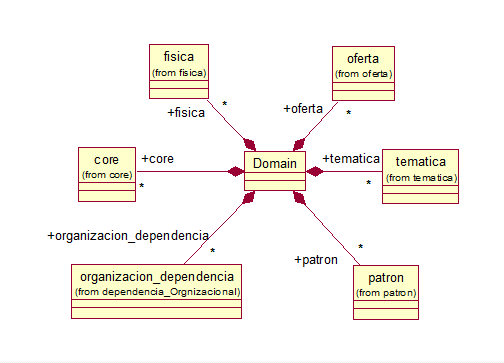


Figura : Capa lógica de la aplicación.

* + - 1. **Modelo esencial de la estructura del consumo de servicios de la aplicación (capa de servicios):**

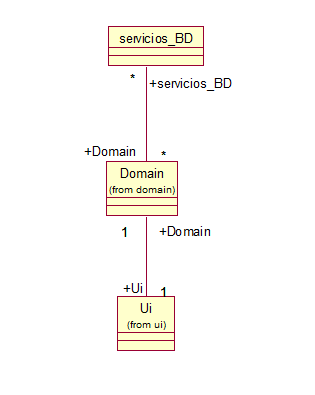


Figura : consumo de servicios de la aplicación.

## Implementation view:

La aplicación Paseo Virtual CRAI no contará con nodos de hardware en la implementación, estos serán reemplazados por servicios en la nube.

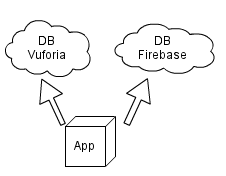


Figura : Diseño-vista de alto nivel sobre el modelo de la implementación del proyecto.

## Deployment view:

Las especificaciones del sistema se han definido en los requisitos funcionales (RF punto 6.1) y los requisitos no funcionales (RNF punto 6.2). A continuación se presenta un prototipo de la interfaz gráfica de acuerdo a las especificaciones de un dispositivo móvil (celular).



Figura : Modelo (Mockup) de la aplicación.

La vista de implementación y despliegue del sistema muestra los nodos físicos en que se ejecuta. En este caso está compuesto por tres nodos físicos: el dispositivo móvil, consumo de servicios de la Aplicación y base de datos. La simplicidad de la vista física puede verse en el siguiente diagrama:

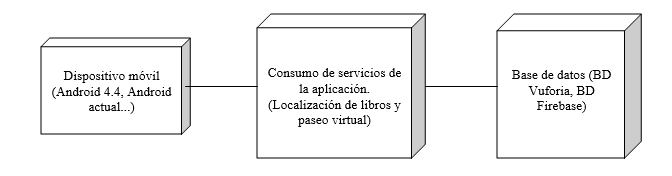


Figura : Deployment de la aplicación.

## Process view:

Este punto de vista mostrará los procesos generales que forman la aplicación y cómo el usuario puede interactuar con ellos.

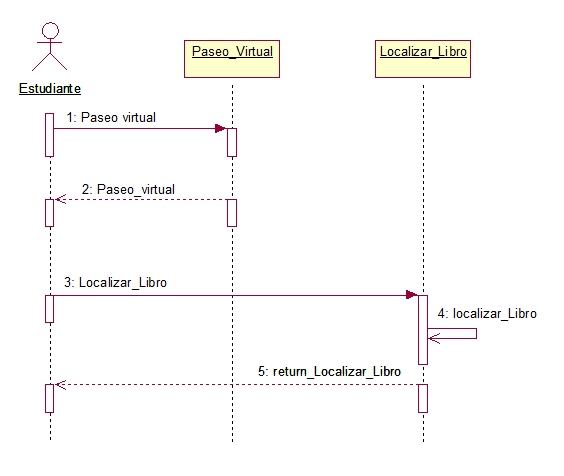


Figura : Diseño y modelo de la Estructura de los procesos a construir.

Como se ve en el diagrama anterior, los requisitos, o la vista de casos de uso, son la principal fuerza impulsora de este y cualquier otro sistema de arquitectura de software.

# Objetivos y limitaciones de la arquitectura

La arquitectura ha sido diseñada con los siguientes objetivos:

1. A fin de facilitar el proceso de consulta de libros, su ubicación y el reconocimiento del espacio físico de la biblioteca CRAI de una Universidad del Quindío mediante la incorporación de la aplicación software (móvil) que se desea realizar.
2. Para permitir mejora de procesos, ofreciéndole información detallada y fácil de entender basada en los datos disponibles.

Las principales limitaciones de diseño e implementación para el sistema son:

1. Simplicidad.
2. Usabilidad.
3. Integración con información o framework externos.
4. Disponibilidad de la información, debido a que la aplicación consumirá servicios externos.

# Procesos de la arquitectura

El sistema que se está desarrollando sigue la metodología de desarrollo RUP y Mobile-D cuyo el objetivo es permitir la producción de software de alta calidad que satisfaga las necesidades del usuario final en el menor tiempo y aprovechando al máximo los recursos como presupuesto.

Este sistema se desarrolla de la siguiente manera: incremental, con entregas pequeñas de software, con ciclos rápidos, cooperativas, cliente y desarrolladores, donde, se trabaja conjuntamente con una cercana comunicación, sencillo: el método en sí mismo es simple, fácil de aprender y modificar. Está bien documentado y es adaptable y permite realizar cambios de último momento.

Sus elementos claves son: poca documentación, simplicidad, análisis como una actividad constante, diseño evolutivo, integraciones, testeos constantes.

La metodología de desarrollo que se aplica en el proyecto presenta diversas ventajas como rápida respuesta a cambios de requisitos a lo largo del desarrollo, entrega continua y en plazos cortos , se hacen entregas de productos mínimos viables que den satisfacción al cliente y pueda empezar a interactuar desde la primer entrega, trabajo conjunto entre el cliente y el equipo de desarrollo lo cual minimiza los costos frente a cambios, es de suma importancia la simplicidad se elimina el trabajo innecesario, atención continua a la excelencia técnica y al buen diseño, mejora continua de los procesos y el equipo de desarrollo, se evitan malentendidos de requerimientos entre el cliente y el equipo, el equipo de desarrollo no malgasta el tiempo desarrollando soluciones innecesariamente generalmente complejas que en realidad no son un requisito del cliente y cada componente del producto final ha sido probado y satisface los requerimientos.

El análisis de negocio incluye los criterios necesarios para el buen desarrollo del sistema, anteriormente se realizó una estimación presupuestaria de los recursos necesarios para completar el sistema y del esfuerzo necesario para llevarlo a cabo, también se estableció un cronograma con los principales hitos.

El modelo de negocio y documentos de visión fueron definidos anteriormente allí se definió el documento el propósito del proyecto y los principales grupos de interés.

El desarrollo implica la cobertura completa de los requisitos de software siguiendo los diferentes diagramas definidos. Al final de esta fase, el sistema de gestión debe ser un sistema completamente diseñado, implementado y probado que está listo para ser desplegado para su uso.

# Patrón o framework arquitectónico a utilizar

En esta sección se describe el patrón arquitectónico que se usará como planteamiento de la arquitectura esencial para la aplicación. El patrón a utilizar de modo general sobre toda la aplicación describe cómo se estructurará el desarrollo de los los principales componentes del sistema además de cómo éstos deben ser agrupados y tratados. En necesario resaltar que primero que todo se reescribirá el patrón por el cual se guiará el desarrollo de toda la aplicación y que segundo, se tendrán patrones adicionales como complemento a funcionalidades y módulos que lo requieran tal como es la integración con frameworks bases de datos, etc.

## Patrón MVC (Model View Controller)

El patrón MVC se utilizará utilizado para implementar la aplicación para la Biblioteca CRAI de la Universidad del Quindío. El patrón MVC permite separar claramente el comportamiento, la presentación y el control de la aplicación. La modularidad de este patrón de diseño permite una fácil reutilización de código, un control más centralizado, más fácil de localizar y permite que el código sea más fácil de modificar (permite el cumplimiento de atributos de calidad establecidos para el proyecto). La presentación o vista, del sistema de gestión se ha implementado primero teniendo presentes unos Mockups definidos y aceptados por el cliente y posteriormente se procede a su implementación.

Al igual que con cualquier otro elemento, un diseño puede ser utilizado para satisfacer las necesidades funcionales, no funcionales o estéticas en un sistema de software. La aplicación, en particular, sigue el estilo arquitectónico de tres capas: capa de presentación, capa de negocio, y el nivel de datos. La siguiente es una descripción sencilla de lo que se incluirá en cada uno de los niveles:

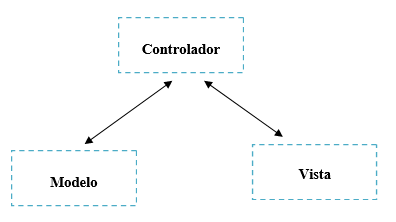


Figura : Patrón principal MVC.

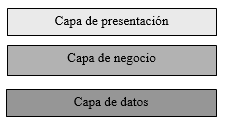


Figura : Descripción de la organización del patrón MVC en el royecto.

Para tener una vista general de cómo sería el funcionamiento de la aplicación implementando este patrón, se tiene:

En general los procesos se describen mediante la solicitud de un servicio a la aplicación y ésta haciendo uso de un controlador permite realizar el proceso de las solicitudes para que cuando ésto sea realizado sea mostrado al usuario.

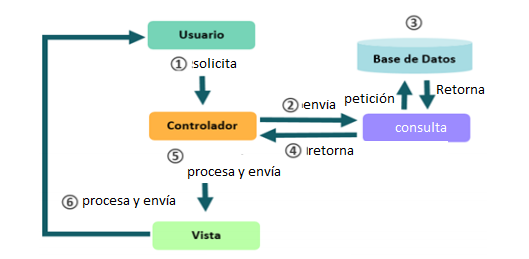


Figura : procesos del patrón MVC en la aplicación.

## Microkernel

El patrón Microkernel arquitectónicamente se incluye como importante para el desarrollo de la aplicación debido a que principalmente en la capa lógica se tienen implementaciones que pueden ser susceptibles a cambios, aumentan su complejidad (por ejemplo base de datos de la ubicación de libros) o dependen de herramientas externas (Vuforia) para el consumo de servicios de la aplicación. De esta forma se busca que se puedan realizar modificaciones (conexión y desconexión de elementos) sin afectar la base CORE de la aplicación con la cual se concibe el negocio.

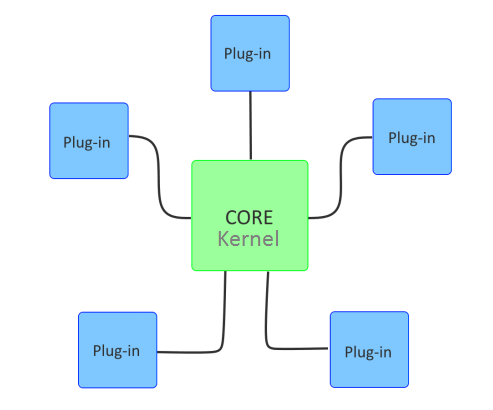


Figura : Esquema principal del patrón Microkernel utilizado en la construcción del proyecto

En la sección de *Views (*Figura 4*)* se puede observar claramente la organización que tendrían los diferentes módulos de la aplicación de acuerdo a lo filosofía microkernel.

# Tácticas Arquitectónicas

A continuación se describen las principales tácticas desde una vista arquitectónica a implementar en el proyecto con el fin de satisfacer los atributos de calidad y por ende requerimientos funcionales y no funcionales definidos y priorizados previamente.

|  |  |
| --- | --- |
| **Concepto de diseño (nombre de**  **patrón / táctica)** | **Justificación de elección** |
| Gestor de módulos | Se realiza con el fin apalancar temas de disponibilidad, desempeño y escalabilidad. Y soportar la gestión de contenedores. |
| Restauración | Debido a que el sistema debe, en caso de caerse, retornar al flujo natural del mismo afectando lo menos posible la experiencia del usuario. En caso de fallar la aplicación por sí misma solicitará cerrar y volver a reiniciar su funcionamiento rápidamente. |
| Servicios | Con el fin de tener una aplicación móvil, es necesario contar con este tipo de servicios para ahorrar tiempo de desarrollo y generar lógica de negocio manejada de forma separada de la aplicación y brindar disponibilidad a la aplicación. |
| Material Design | El uso de material Design permite tener unas pautas enfocadas al diseño utilizado en Android y de esta forma crear diseños e interfaces adecuadas al tipo de usuarios y dispositivos que la usarán, dado a que este estándar brinda la guía necesaria para ubicar las formas y colores acorde a las APP nativas. |
| Patrón MVC | Se debe definir claramente una separación de responsabilidades entre los componentes del sistema con el fin de hacer más sencilla la mantenibilidad del sistema. |
| Incrementar eficiencia del dispositivo móvil | Se elige esta táctica debido a que ayuda a optimizar la aplicación. Se implementa un monitoreo del comportamiento de la aplicación (Fse de pruebas) en cuanto a accesos a bases de datos para identificar operaciones lentas o más usadas que deben ser priorizadas y optimizadas; con esto se apalancará el desempeño y el procesamiento de peticiones que deben ser tendidas dentro de un límite de tiempo específico en para mostrar resultados al usuario. |
| Patrón Microkernel | Debido a que las funcionalidades de la aplicación pueden tornarse mucho más complejas o incluso extenderse se permite agregar modificaciones sin afectar la base Core con la cual se concibe el negocio con el cual funciona la aplicación. |
| Procesamiento Asíncrono | Se implementa esta táctica debido a que se propone dividir los eventos de solicitudes de servicios en operaciones asíncronas para su procesamiento; se asignan prioridades a las solicitudes que se hacen en el sistema y de esta forma de procesan apalancando así el rendimiento. Se tiene en cuenta que los usuarios no van a realizar al mismo tiempo la misma operación sobre el sistema ya sea paseo virtual o localización de libros. |
| Ping/Eco | Se realiza con el fin de determinar que el sistema está funcionando correctamente, o en caso de que la misma no responda, verificar en qué momento vuelve a estar arriba para continuar con su normal ejecución. Lo que apalancará la disponibilidad de la aplicación de cada uno de los contenedores. |
| Arquitectura orientada a eventos | Se aplicarán conceptos de la Arquitectura orientada a eventos, Con el fin de apalancar la gestión de transacciones del sistema, y la gran demanda de servicios a nivel interno y externo en cuanto al uso de las consultas de libros y realidad aumentada principalmente. |

Tabla : Tácticas de arquitectura aplicadas al proyecto.

# Trazabilidad de los Drivers arquitectónicos

Principales atributos de calidad en los cuales se basan los requerimientos del proyecto y por los cuales se tomaron ciertas decisiones arquitectónicas que impactan el desarrollo de la aplicación.

* Mantenibilidad
* Usabilidad
* Learnability
* Confiabilidad
* Extensibilidad
* Disponibilidad

## Tácticas o patrones que satisfacen los drives más importantes del proyecto.

|  |  |  |
| --- | --- | --- |
| **Driver-ID** | **Descripción del Driver** | **Táctica/Patrón que lo satisface** |
| DRI-01 | Cuando un usuario realice la búsqueda de un libro se localizara el estante que lo contiene y se mostrará al usuario su ubicación. | Patrón Microkernel, mvcd / Arquitectura orientada a eventos |
| DRI-02 | La aplicación mostrará los resultados de las consultas ejecutadas por el usuario. | Servicios/Arquitectura orientada a eventos |
| DRI-03 | La aplicación funcionará correctamente ante el presente incremento de usuarios. | Incrementar eficiencia del dispositivo móvil/Gestor de módulos |
| DRI-04 | La aplicación mostrará mediante AR información importante sobre elementos y lugares que componen la biblioteca. Permitirá el acceso a la cámara de los dispositivos móviles. | Ping-Eco/ Arquitectura orientada a eventos/ Incrementar eficiencia del dispositivo móvil |
| DRI-05 | La aplicación por su construcción será posible añadir nuevas funcionalidades posteriormente. | Gestor de módulos/Patrones mvc y microkernel |
| DRI-06 | Debe consumir, mostrar, toda la información de los servicios web desde la app. | Servicios/Procesamiento Asíncrono/Ping-Eco |
| DRI-07 | Usabilidad-Aplicación implementada siguiendo el reglamento de Material Design para aplicaciones Android propuesto por Google en cuanto a cómo debe ser la “usabilidad” de las mismas. | Material Design |
| DRI-08 | Usabilidad-Las pantallas diseñadas según el “Estándar de Google” para el desarrollo de aplicaciones móviles en este caso Android, son fáciles de usar por parte de la comunidad universitaria. Para esto se tiene en cuenta:   * + La ubicación de los botones, menús y demás complementos.   + Combinación de colores.   + Alertas-Mensajes mostrados al usuario.   + íconos. | Material Design |
| DRI-10 | La aplicación tendrá en cuenta las características de seguridad y privacidad, en cuanto a las condiciones de uso adecuadas de las bases de datos. | Patrones MVC /microkernel (en su modularidad)  Gestor de módulos |
| DRI-11 | Se debe informar al usuario cuando el dispositivo no cuente con conexión a internet, y se deben inhabilitar las opciones que dependen de esta conexión. | Arquitectura orientada a eventos |

Tabla : Tácticas o patrones que satisfacen los drives más importantes del proyecto

## Lista completa de driver arquitectónicos

**Calificación de Prioridad**:

* 1 a 2, la prioridad es baja.
* 3, la prioridad es media.
* 4 a 5, la prioridad es alta.

|  |  |  |  |
| --- | --- | --- | --- |
| **Driver-ID** | **Tipo de Driver** | **Descripción del Driver** | **Prioridad** |
| DRI-01 | RF-01 | Cuando un usuario realice la búsqueda de un libro se localizara el estante que lo contiene y se mostrará al usuario su ubicación. | 4 |
| DRI-02 | RF-02 | La aplicación deberá permitir la consulta de libros y saber su ubicación. | 4 |
| DRI-03 | RF-03 | La aplicación permitirá el acceso a la cámara de los dispositivos móviles. | 5 |
| DRI-04 | RF-04 | La aplicación mostrará una ubicación sobre los libros que se tienen en la biblioteca. | 5 |
| DRI-05 | RF-05 | La aplicación mostrará los resultados de las consultas ejecutadas por el usuario. | 4 |
| DRI-06 | RF-06 | La aplicación funcionará correctamente ante el presente incremento de usuarios. | 4 |
| DRI-07 | RF-07 | La aplicación mostrará mediante AR información importante sobre elementos y lugares que componen la biblioteca. | 4 |
| DRI-08 | RF-08 | La aplicación será documentada y será posible añadir nuevas funcionalidades posteriormente. | 3 |
| DRI-09 | RF-09 | La aplicación brindará a los usuarios información precisa de acuerdo a lo consultado y ubicaciones requeridas. | 4 |
| DRI-10 | RF-10 | Debe consumir, mostrar, toda la información de los servicios web desde la app. | 4 |
| DRI-11 | RNF-01 | El sistema debe estar disponible durante el horario de atención del público por parte de la biblioteca. Lunes a viernes 7:00 a.m. a 09:00 p.m. sábados de 8:00 a.m. a 6:00 p.m. | 4 |
| DRI-12 | RNF-02 | Aplicación implementada siguiendo el reglamento de Material Design para aplicaciones Android propuesto por Google en cuanto a cómo debe ser la “usabilidad” de las mismas. | 5 |
| DRI-13 | RNF-03 | Las pantallas diseñadas según el “Estándar de Google” para el desarrollo de aplicaciones móviles en este caso Android, son fáciles de usar por parte de la comunidad universitaria. Para esto se tiene en cuenta:   * + La ubicación de los botones, menús y demás complementos.   + Combinación de colores.   + Alertas-Mensajes mostrados al usuario.   + íconos. | 5 |
| DRI-14 | RNF-04 | Los colores usados en la aplicación deberán ser referentes a la universidad. | 3 |
| DRI-15 | RNF-5 | La aplicación será compatible para versiones Android desde la API 19 en adelante (versión Android 4.4 hasta la actual). | 4 |
| DRI-16 | RNF-6 | La aplicación tendrá en cuenta las características de seguridad y privacidad, en cuanto a las condiciones de uso adecuadas de las bases de datos. | 4 |
| DRI-17 | RNF-07 | Se debe informar al usuario cuando el dispositivo no cuente con conexión a internet, y se deben inhabilitar las opciones que dependen de esta conexión. | 3 |
| DRI-18 | RNF-08 | Se debe contar con una interfaz principal que permita la navegación entre las diferentes opciones de las app. | 3 |
| DRI-19 | RNF-9 | Cada vez que se presione un botón se debe direccionar rápidamente al módulo solicitado. | 4 |

Tabla : Lista completa de driver arquitectónicos.