Instituto Tecnológico de Costa Rica

Escuela de Computación
Curso: IC6831
Aseguramiento de la Calidad del Software
Grupo 2, II Semestre 2017
Profesor: Saúl Calderón Ramírez

Avance 2, SyRS

Integrantes:
Brandon Dinarte 2015088894
Armando López 2015125414
Andrey Mendoza 2015082908
Julian Salinas 2015114132

Domingo 08 de Octubre del 2017
ITCR, Sede Central Cartago

1. Sección 1: Introducción

Como bien es sabido, antes de ejecutar una tarea se debe realizar un proceso de análisis, recolección y definición de ciertos elementos que son importantes para lograr el objetivo deseado. Tal es el caso del proceso de definición del alcance que lo que pretende es analizar, recolectar y definir las necesidades planteadas por el usuario en la fase de recolección de requerimientos del software.

Por esta razón, es de suma importancia tener claro en todo momento hacia dónde se dirige el objetivo del proyecto, es decir, se debe tener conocimiento del contexto para evitar percepciones equivocadas con respecto a lo que el sistema realmente debe hacer, ya que de no ser así el desarrollo del programa sería totalmente a ciegas y las probabilidades de incurrir en un problema de incumplimiento de las expectativas del cliente aumentarían considerablemente.

En síntesis, esta sección provee los fundamentos y referencias necesarias para el desarrollo de requerimientos del sistema EigenFaces, los cuales representan el nivel máximo de abstracción dentro de la cadena de requerimientos. Además, busca justificar la razón del por qué el proyecto se va a realizar y el valor que el sistema producirá, tanto para la empresa o interesado como para los clientes.

1.1. Propósito del Documento

El propósito del documento consiste en identificar los requerimientos y el alcance del producto para el proyecto EigenFaces, además del análisis de la calidad, y otros aspectos importantes que brindarán un especificación de requerimientos sustentable para satisfacer las necesidades del usuario. Además, se describe cada parte del sistema de manera general, sin entrar en tantos detalles de software y hardware.

1.2. Convenciones del Documento

La convención de este documento tendrá como tipografía Arial, 12, con márgenes justificados e interlineado 1,5. En caso de que haya algún elemento subrayado o resaltado en negrita el mismo se denotará como de gran importancia. Se van a establecer requerimientos generales en donde estos van a tener dentro de sí mismos requerimientos más detallados, dando prioridad a ciertos elementos del producto.

1.3. Propósito del Sistema

El reconocimiento facial se ha convertido en los últimos años en un área de investigación activa que abarca diversas disciplinas, como el procesamiento de imágenes, reconocimiento de patrones, visión por ordenador y redes neuronales.

Debido a sus facilidades de lectura, poca invasión al usuario, precisión y agilización de procedimientos, esta tecnología ha representado un gran avance para el reconocimiento efectivo de personas, sobretodo para empresas e instituciones donde la seguridad es uno de los factores más importantes.

Estas propiedades mencionadas dan solución al problema que se desea resolver, desarrollar un sistema de bajo coste que permita a distintas instituciones, como universidades, aeropuertos, puestos de migración o incluso bancos, implementar un enfoque eficiente para identificar a las personas, ya sea por motivos de seguridad o para brindar facilidades a sus miembros.

En primera instancia, el propósito del proyecto por parte de BiomeSys abarca la creación de un sistema de reconocimiento facial usando como base un pasaporte electrónico con la información necesaria de su portador (incluyendo imágenes) en los principales puntos de entrada al país, entre estos, el aeropuerto Juan Santamaría, el paso por Peñas Blancas, Paso Canoas y Sixaola donde el mismo sea capaz de tramitar a los viajantes sin tener que disponer de otros procesos menos eficientes e intrusivos. Además, se pretende incorporar junto con otros sistemas de biometría para que de esta forma la precisión con la que se reconocen las personas sea mayor.

1.4. Alcance del Sistema

Como se menciona anteriormente el sistema de reconocimiento facial beneficiará a un contexto en específico, sin embargo, se espera que en un futuro sea implementado en otros, como por ejemplo, universidades, bancos u otras instituciones ya sean públicas o privadas. A su vez, se pretende que trabaje conjuntamente con más sistemas, ya sean también de biometría o para compartir información con otros sistemas que sean de interés.

Conforme pasan los años nuevas tecnologías aparecen, por tanto, se pretende que el atributo de mantenibilidad sea acorde, es decir, que el sistema pueda ser ajustado fácilmente respecto a estas tecnologías.

1.5. Características del Usuario

Respecto a los usuarios del sistema, se podría decir que en este caso como el producto de reconocimiento de rostros se está llevando a cabo para un aeropuerto, los usuarios de la aplicación vendrían a ser solamente los empleados de la zona de pasajeros, donde la función principal sería realizar los chequeos necesarios o rutinarios para identificar los pasajeros. Estos usuarios podrían presentar algunas condiciones particulares que se deberán tener en cuenta a la hora de desarrollar la aplicación; estas se mencionan a continuación:

1. Ocupan un tiempo de respuesta rápido, pues están bajo constante estrés y con un flujo de trabajo bastante concurrido.

2. El entorno de trabajo debe ser simple pero a la vez amigable con el usuario, esto para hacer que la utilización de la herramienta sea armoniosa y no conlleve una carga más para el empleado.

3. Podrían no tener muchos conocimientos acerca de la utilización de herramientas computacionales, por lo que la aplicación debe ser lo más intuitiva posible.

Por último, respecto a la relación directa que existe entre los usuarios de chequeo y el sistema estarían los siguientes escenarios:

1. Solicitar un reconocimiento de rostro

2. Ingresar un nuevo sujeto a la base de datos

3. Entrenar el sistema basado en una localización que contenga la base de datos

2. Sección 2: Descripción General del Sistema

2.1. Contexto del Sistema

En el presente apartado se especifican las diferentes acciones que puede llevar a acabo el usuario de la aplicación EigenFaces. A continuación, se muestra la secuencia de acciones y reacciones que se pueden dar entre los dos entes: Usuario EigenFaces y el app.

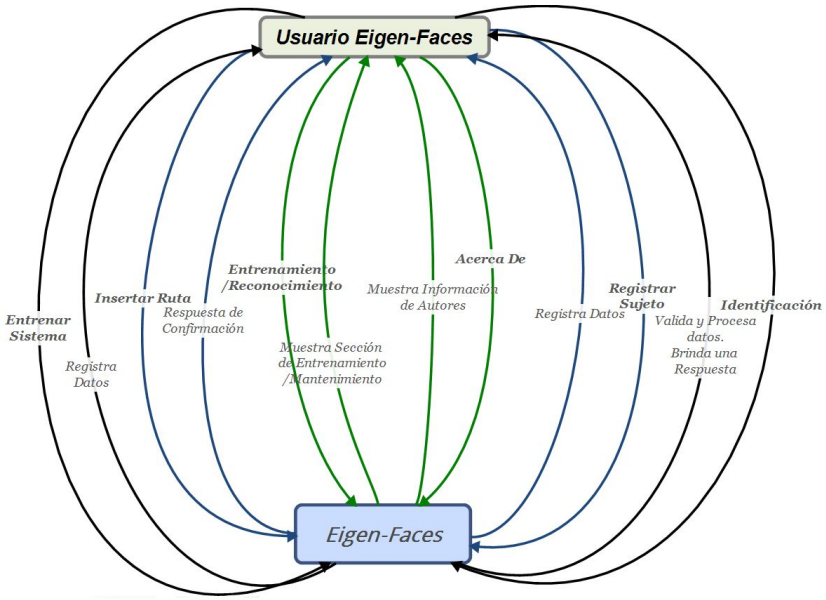

Figura 1: Diagrama de Contexto del Sistema Eigen-Faces.

2.2. Secuencia de Acciones

- **Entrenar el Sistema:** Esta acción por parte del usuario hace que el sistema dada una ruta de localización de la base de datos de sujetos, recurra a realizar el cálculo de las auto-caras o eigenfaces, así como la proyección de todas las muestras en el nuevo espacio de dimensionalidad reducida, estas acciones se resumieron como Registrar Datos.

- **Insertar Ruta:**Insertar Ruta, consiste básicamente en ingresar una localización de la base de datos, misma que servirá para llevar a cabo el entrenamiento. Respecto a la respuesta del sistema a esta acción, lo que se realizará es retroalimentar al usuario para dar una respuesta de confirmación de la ruta.

- **Entrenamiento/Reconocimiento:** lo que realizan es la selección de la pestaña a la que se quiere ir en la página web, básicamente es el proceso de enrutamiento. El sistema lo que realizará es el despliegue de la sección o también conocida como app.

- **Acerca De:** esta acción al igual que las de Entrenamiento y Reconocimiento, lo que realiza es la elección de una sección, donde en este caso el sistema desplegará información acerca de los autores.

- **Registrar un Sujeto:** esta acción es muy obvia y consiste básicamente

en ingresar un nuevo sujeto a la base de datos. El sistema deberá informar del estado final a la hora de realizar el registro.

- **Identificación:** consiste en ejecutar la identificación de un sujeto a partir de una imagen del rostro, donde el sistema procesa los datos, ejecuta el algoritmo y por último muestra una respuesta con la etiqueta del sujeto correspondiente.

2.3. Modos y Estados del Sistema

Los modos de operación del sistema se encuentran clasificados en dos grandes secciones que son el entrenamiento e identificación de rostros. Donde el entrenamiento consiste en el cálculo de las auto-caras o eigenfaces, así como la proyección de todas las muestras en el nuevo espacio de dimensionalidad reducida. Por otro lado, la identificación se basa en una fachada que emula el reconocimiento de un sujeto usando el sistema de auto-caras pre-entrenado.

Para más esclarecimiento de estos modos, se recomienda consultar la descripción detallada que se puede encontrar en la sección "*3.1 Funciones del Sistema*".

2.4. Suposiciones y Dependencias

A la hora del desarrollo e implementación del producto se van a tener ciertas suposiciones, en donde si estas no se cumplen no se podrá realizar la implementación adecuada del producto. Por lo tanto, se definieron de la siguiente manera:

1. Los entregables del Proyecto serán escritos en español.
2. Los usuarios de la aplicación de reconocimiento de rostros tienen conexión a internet.
3. El desarrollo del producto será realizado sobre el sistema operativo Windows.
4. El desarrollo de la aplicación se llevará a cabo mediante el uso de software libre.

También se definieron una serie de riesgos y dependencias, de situaciones en las que el producto pueda fallar. Definidas de la siguiente manera:

1. No lograr el alcance deseado.
2. Que la aplicación del reconocimiento de rostros no se adapte a las necesidades de los aeropuertos.
3. Inexistencia de soporte en un futuro del software libre a utilizar.

2.5. Seguridad del Sistema

La información a manejar mediante el sistema Eigen-Faces es bastante sensible y el funcionamiento adecuado del mismo puede llegar a ser determinante en asuntos de seguridad ciudadana. Es por esto que es muy importante implementar medidas que eviten debilidades en el sistema y lo hagan vulnerable.

Como aspecto fundamental de cualquier sistema que requiere seguridad, se deberá identificar al usuario que va a interactuar directamente con el sistema con el objetivo de proveer solamente la información y funcionalidades disponibles para este tipo de usuario. Los diferentes tipos de usuario serán descritos en los requerimientos funcionales con el fin de establecer los tipos de usuario y las opciones que cada uno de estos va a poder realizar. Sin embargo, para cada uno de ellos se manejará la siguiente información:

- Nombre y Apellidos.
- Cédula.
- Número de teléfono.
- Dirección actual.
- Foto tamaño pasaporte tomada en el momento de su registro en el sistema. Esta imagen deberá ser actualizada cada año.

Debido a la sensibilidad de la información que el sistema utilizará (imágenes personales), además de la responsabilidad que este tendrá, se manejará un registro por cada operación realizada que afecte directamente el funcionamiento de Eigen-Faces. Esta bitácora guardará la siguiente información:

- Fecha y hora en la cual se hizo la solicitud
- Identificación del usuario solicitante
- Nombre de la operación (entrenamiento, inicio de sesión o identificación)
- Resultado de la operación

En ella se podrá ver la actividad del sistema en caso de cualquier anomalía y así poder llevar a cabo una investigación para tomar las medidas necesarias.

Va a existir un tipo de usuario que es administrador del sistema que va recibir notificación inmediata sobre la identificación de algún sujeto, cuando se está realizando un entrenamiento o se agrega un nuevo usuario al sistema.

2.6. Empaquetado, Manipulación y Transporte

La entrega de los productos de software no se puede realizar de manera física y el producto tampoco suele ser fácil de poner en funcionamiento sin una configuración e instalación previa adecuada. Es por esto que en este punto se describe la forma de entrega con el cliente y los diferentes módulos que componen al sistema.

El sistema está compuesto por 3 paquetes que se comunican entre ellos para cumplir con el objetivo solicitado por el cliente:

1. **Módulo de entrenamiento:** se encarga de tomar una base de datos de imágenes de sujetos y realiza los cálculos necesarios para generar las auto-caras. Este módulo es independiente de los otros dos.
2. **Módulo de identificación:** toma las auto-caras y una imagen aparte. Partiendo de la imagen aparte, se procede a realizar una comparación del sujeto de la imagen respecto a cada uno de los sujetos almacenados en la base de datos utilizando las auto-caras generadas previamente. Si se logra establecer igualdad entre ambos sujetos, el resultado de la operación tiene éxito.
3. **Interfaz de usuario:** permite la comunicación del sistema con los usuarios del mismo. Esta será desarrollada como aplicación web.

Los requerimientos de cada uno de estos módulos están descritos con detalle en la sección de requerimientos funcionales.

Una vez se proceda a realizar la entrega del producto, la empresa desarrolladora se encargará de establecer cada uno de los módulos correctamente en el hardware del cliente, esto con el fin de evitar errores a la hora de su instalación y configuración. Además, se harán las respectivas pruebas para determinar que el sistema funcione correctamente bajo las reglas descritas en el punto 5 (Verificación) de este documento.

3. Sección 3: Capacidades, Condiciones y Restricciones del Sistema

Esta sección describe detalladamente las diferentes características que debe tener el sistema, desde la parte funcional como la no funcional. Además, se incluyen algunos detalles del ambiente operativo del sistema, condiciones y restricciones que se deberán tener en cuenta a lo largo del desarrollo del producto.

3.1. Funciones del Sistema

Para poder entender a grosso modo las funcionalidades básicas del sistema Eigen-Faces, se decidió dividirlas en 3 secciones que serán explicadas a conti-

nuación:

1. **Entrenamiento:** consiste en el cálculo de las auto-caras, así como la proyección de todas las muestras en el nuevo espacio de dimensionalidad reducida de auto-caras. Para lograr utilizar esta funcionalidad, se debe tener en cuenta los pasos a seguir, que son:

   a) Cargar un conjunto de imágenes almacenada en la dirección dada por el usuario para el entrenamiento. El conjunto de imágenes debe contener una carpeta por cada sujeto, y se debe almacenar el nombre de tal carpeta como la etiqueta de tal sujeto o clase.

   b) Entrenar el sistema, generando las auto-caras y la proyección de las muestras en el nuevo espacio formado por tales auto-caras, usando el algoritmo supervisado del centroide más cercano. Además, el parámetro del entrenamiento será la cantidad de autovectores a conservar en el nuevo conjunto de vectores base.

   c) Cargar un conjunto de muestras de prueba, para medir la precisión del sistema.

2. **Identificación:** El modo de identificación de sujetos consiste en una fachada que emula la identificación de un sujeto usando el sistema de auto-caras pre-entrenado. Los requerimientos del sistema en este modo son los siguientes:

   a) Cargar las auto-caras y las muestras proyectadas de un sistema entrenado previamente.

   b) Permitir la carga de una imagen correspondiente a un rostro, y ejecutar la identificación del sujeto. El sistema debe desplegar la etiqueta correspondiente a tal sujeto.

3. **Interfaz gráfica:** existirá una interfaz gráfica amigable con usuario, la cual permitirá hacer uso del modo entrenamiento y modo identificación de nuevos sujetos según se explicó en los párrafos anteriores.

Como funcionalidad extra, se decide implementar un algoritmo de clasificación supervisado no paramétrico de los -vecinos más cercanos. El valor de K debe ser un parámetro modificable por el usuario. Además, se compararán los resultados con el algoritmo de clasificación del centroide más cercanos para distintos valores de K.

3.2. Requerimientos Funcionales

Definir los requerimientos de un proyecto siempre es importante para conocer el alcance que este debe tener para ser aceptado por el cliente y así evitar algún tipo de inconvenientes. Es por esto que en esta sección se detallarán detenidamente cada uno de los requerimientos con los que debe cumplir el producto.

3.2.1. Lista de Requerimientos

Los requerimientos han sido reordenados según su prioridad con base en las necesidades y solicitudes del cliente.

1. **Ruta del conjunto de imágenes:** el sistema deberá tomar como parámetro la ruta donde se encuentran almacenadas las imágenes sobre las que se va a realizar el entrenamiento. Este requerimiento tiene una prioridad **alta** a solicitud del cliente, es prioritario al Req. #2 pues es necesario para su cumplimiento.

2. **Carpetas por cada sujeto:** en la ruta especificada para el entrenamiento, deben existir carpetas por cada sujeto. Dentro de ellas, se encuentran las imágenes de cada sujeto. Este requerimiento tiene una prioridad **alta** a solicitud del cliente.

3. **Generación de las auto-caras:** para el entrenamiento del sistema, se utilizará el algoritmo del centroide más cercano. Este requerimiento es requisito para el Req. #4 y además tiene una prioridad **alta** a solicitud del cliente.

4. **Almacenamiento de las auto-caras:** una vez entrenado el sistema, se deben almacenar los datos, esto con el fin de luego utilizar estas auto-caras precalculadas para el modo de identificación. Este requerimiento tiene una prioridad **alta** a solicitud del cliente, pero es dependiente del requerimiento anterior.

5. **Parámetro de entrenamiento:** será la cantidad de autovectores a conservar en el nuevo conjunto de vectores base. Este requerimiento depende del Req. #3. Y tiene una prioridad alta al momento de la confección del documento.

6. **Carga de una imagen:** para la identificación de un sujeto, el usuario debe poder cargar una imagen de un rostro y luego proceder a la identificación del mismo. Este requerimiento tiene una prioridad **alta** a solicitud del cliente.

7. **Métrica de precisión:** para medir la precisión del sistema, deberá cargarse un conjunto de pruebas para y calcular los falsos positivos y negativos, la precisión y el recall. Implementando las métricas vistas en el curso, se generará un informe en un archivo con formato ".csv" que contenga los resultados. Este requerimiento tiene una prioridad **alta** a solicitud del cliente.

8. **Interfaz gráfica:** se debe realizar una interfaz que sea amigable con el usuario. Desde esta será posible llamar los dos módulos disponibles según el tipo de usuario. Este requerimiento ha sido priorizado a nivel medio.

9. **Resultados de la identificación:** una vez identificado el sujeto, se mostrará la etiqueta que tiene el mismo en la base de datos. Este requerimiento ha sido priorizado a nivel medio.

10. **Entrenamiento con otro algoritmo:** el sistema deberá ofrecer la opción de realizar el entrenamiento utilizando un algoritmo de clasificación supervisado no paramétrico de los k-vecinos más cercanos. Este valor k, debe ser un parámetro modificable por el usuario. Este requerimiento ha sido priorizado a nivel medio.

11. **Comparación de entrenamientos:** debe realizarse una comparación entre el entrenamiento mediante el algoritmo del centroide y el otro algoritmo de clasificación supervisado no paramétrico implementado. Este requerimiento ha sido priorizado a nivel medio.

3.3. Requerimientos de Usabilidad

Respecto a la entendibilidad que debe presentar el software para permitir al usuario el captar la utilidad o propósito del mismo y como puede ser usado para cumplirlo, se establece una prioridad intermedia, pues los usuarios que han de utilizar el sistema se espera que se encuentren conscientes de lo anterior, al ser empleados especializados, y que no requieran obtener las ideas con base al producto.

- **3.3.1** Se define la necesidad de un apartado breve dentro del sistema que provea información acerca del uso del mismo para esclarecer los objetivos con los que cuenta y los procedimientos para conseguirlos. Esto con el fin de facilitar al usuario el comprender la utilidad del sistema, en caso de no estar claro al momento del uso.

  De igual forma para la aprendibilidad del sistema se establece un prioridad intermedia por motivos similares, ya que los empleados han de estar familiarizados con el proceso realizado para permitir la entrada al país, y se espera tenga experiencia con el mismo con lo cual el software habrá de resultar familiar y fácil de asimilar por naturaleza.

- **3.3.2** El operador del software debe tener el control sobre toda configuración del mismo de una forma transparente, por lo que todo aspecto configurable estará disponible para edición de los usuarios con el permiso adecuado dentro de la compañía cliente.

  Lo anterior establece una alta prioridad del cumplimiento del atributo de operabilidad, pues es de vital importancia que los empleados puedan controlar las funcionalidades presentadas de manera que obtengan los resultados esperados.

3.4. Requerimiento de Rendimiento

La utilización del tiempo tiene una alta prioridad para el sistema por la naturaleza de las circunstancias en las que se requerirá la identificación de sujetos.

- **3.4.1** Con el tiempo como vital factor en el desempeño del sistema, se define la necesidad de que la ejecución de las funcionalidades principales como es la identificación tarde entre diez y quince segundos para su completitud. Los rangos definidos han de variar ante el crecimiento de la base muestral, con lo cual se acuerda generar un posterior estudio para la redefinición de los márgenes.

- **3.4.2** Para la funcionalidad principal de entrenamiento del sistema, aplican los mismos principios que para la identificación, con la diferencia de que los márgenes de tiempo empieza desde diez a quince minutos en el estado actual de la base muestral.

  En términos de recursos el contexto del sistema no establece restricciones específicas que deban ser tomadas como prioridades reales.

- **3.4.3** Con base en el actual análisis de proyección de consumo de recursos del sistema, se define que no ha de consumir más de dos gigabytes de memoria principal, y para su almacenamiento no sobrepasar los cien megabytes en su versión inicial.

- **3.4.4** Todo requerimiento de tiempo debe ser cumplido según lo especificado en el presente documento, por otra parte los requisitos de utilización de recursos puede verse incumplidos en caso de presentar una justificación evidenciada.

3.5. Requerimientos de Integración con Sistemas Humanos

Como parte de las interacciones entre el sistema y el empleado de la estación de control fronteriza, son definidos aspectos tomados en cuenta para asegurar un apto funcionamiento del mismo.

- **3.5.1** El sistema requiere de un empleado como máximo para ser operable, en otras palabras, el funcionamiento del sistema no requiere de la coordinación directa entre empleados para presentar los resultados esperados.

- **3.5.2** La ejecución de las funciones principales del sistema está disponible para todo empleado identificado, sin la necesidad de que sean otorgados permisos de forma directa por empleados de mayor privilegio.

3.6. Mantenibilidad

La creación de un sistema debe realizarse con la visión de poder extender sus funcionalidades, modificar su estructura por errores de planificación o razones que no fueron tomadas en cuenta durante su diseño, o simplemente porque se desea actualizar o cambiar la manera actual en la que se realizan algunos procesos. Para facilitar estos cambios, se implementan algunos principios de software. El sistema Eigen-Faces no es la excepción y es por eso que a continuación se describirán los puntos a tomar en cuenta para lograr un grado de mantenibilidad adecuado.

- **Facilidad de análisis:**se tendrán diferentes diagramas en formato UML que permitan conocer la estructura del sistema. El código debe apegarse por completo a lo que se planificó en el diagrama de clases. Esto evitará que sea necesario ir directamente al código para revisar la estructura el sistema.
- **Testabilidad:** para lograr verificar que el sistema funcione correctamente tras una modificación sobre el mismo, este debe proveer una manera sencilla de esta verificación. Para ello, se implementará un pequeño programa que hace pruebas sobre las funcionalidades de entrenamiento e identificación. Este programa utilizará la herramienta UnitTest para indicar si los resultados son correctos o no. Además, cada uno de los módulos podrá ser probado por aparte para de esta forma poder identificar de manera más sencilla el origen de algún eventual fallo.
- **Flexibilidad:** para realizar cambios de manera sencilla sobre el sistema, se hará uso de la mayor cantidad de patrones de diseño disponible, ya que estos siguen los principios de SOLID y GRASP que permiten la modificación del código de manera sencilla.

3.7. Fiabilidad

La robustez de un sistema es fundamental para evitar que el funcionamiento del mismo se vea afectado. Debido al contexto en el que será implementado el sistema Eigen-Faces, es extremadamente importante que siempre se mantenga activo, además de la exactitud de la identificación para evitar inconvenientes por falsos positivos o poner en riesgo la seguridad de las personas por falsos negativos. Es por ello que se utilizarán principios que eviten en la medida de lo posible que los casos mencionados anteriormente ocurran.

- **Tolerancia a fallos:** se implementará una funcionalidad de pruebas que se ejecute luego de cada entrenamiento, esto con el fin de verificar si el entrenamiento se realizó adecuadamente. Luego de cada entrenamiento, se debe calcular el margen de error que existe en la identificación de los sujetos. Esto permitiría determinar que tan fiable es la respuesta del sistema.

- **Recuperabilidad:** para mantener el sistema siempre activo, se debe implementar en el código posibles errores que pueden ocurrir, utilizando rutinas de escape y notificar detalladamente a los usuarios administradores sobre el error para que estos realicen la acción necesaria. Se debe mantener el archivo de auto-caras calculadas respaldado, así en caso de ocurrir un error durante el entrenamiento, el sistema deberá cargar las auto-caras anteriores.

- **Madurez:** para poder arreglar los errores ocurridos durante la ejecución del programa, se tendrá una bitácora aparte donde se podrá observar la fecha y hora y la descripción del error retornada el lenguaje de programación. Con estos datos, será posible estudiar las causas y así poder estos fallos vuelvan a ocurrir en el sistema.

3.8. Requerimientos Físicos

El sistema Eigen-Faces entrenamiento e identificación será programado para ejecutarse sobre un mismo servidor, por lo que es importante aclarar que para conjuntos de imágenes grandes, es posible que el proceso tarde bastante. A futuro se recomienda la implementación de este sistema sobre clusters con el fin de distribuir el trabajo de procesamiento y poder realizar la ejecución de entrenamiento de manera más rápida.

Para ofrecer acceso desde cualquier lugar, es necesario la compra de un dominio público para acceder a la página web. El servidor de la página web se encargará de hacer los llamados a la lógica y retornar al usuario el resultado de la operación.

3.9. Requerimientos de Adaptabilidad

- **3.9.1** Para asegurar la capacidad del sistema de ejecutarse en la mayor cantidad posible de equipos disponibles en las instalaciones fronterizas, se establece que parte de la adaptabilidad del mismo es la utilización de un lenguaje interpretado independiente de la arquitectura del equipo que lo ejecuta. A petición informal de los clientes, se establece el lenguaje de implementación como Python, sin especificar una versión determinada.

- **3.9.2** Ante el inminente crecimiento de la base de muestras gráficas sobre los individuos, el sistema ha de ser capaz de trabajar con cantidades de datos múltiplos de los actuales sin perder rendimiento promedio en gran escala. Este aspecto al estar relacionado con las especificaciones de rendimiento del sistema, puede ser complementado con la información detallada en dicha sección.

3.10. Políticas y Regulaciones

El sistema Eigen-Faces utiliza información sensible de los ciudadanos, por lo que queda prohibido la manipulación de cualquier tipo de la misma, a excepción de su uso dentro de los algoritmos a implementar.

El objetivo de la implementación de este sistema en las líneas fronterizas es mantener la seguridad de la ciudadanía, por lo que se trata de hacer de la manera más discreta posible, esto con el fin de evitar que las posibles amenazas evadan el sistema entrando de manera ilegal al país. Por esta razón, la empresa Biomesys se compromete a mantener de forma secreta toda la información dada.

3.11. Gestión de Información

Entre los datos definidos como posible información relevante o de valor real para la ejecución del sistema, se considera el contenido de la base de sujetos muestral, los datos de identificación de los empleados que poseen acceso al sistema y las configuraciones de mayor uso que requiere el software para su funcionamiento. Con lo anterior como base, se definen los requerimientos de gestión de dicha información de la forma siguiente.

- **3.11.1** La base muestral de sujetos debe ser revisada por el sistema ante posibles alteraciones con un margen de tiempo definido por el cliente posteriormente, para evitar corrupción en los datos de identificación de personas.
- **3.11.2** De lo definido anteriormente deriva la necesidad de que exista un respaldo original del que se obtendrá la base muestral para la ejecución del entrenamiento del sistema.
- **3.11.3** El software de entrenamiento funcionará únicamente bajo la seguridad de que la base muestral se encuentra en estado óptimo no corrupto.
- **3.11.4** Con la versión inicial del sistema, se proporciona una base muestral previamente probada para la ejecución exitosa del sistema.

3.12. Mantenimiento del Ciclo de Vida del Sistema

Para hablar del ciclo de vida del proyecto primero se debe tener claro que la metodología de desarrollo a implementar será el modelo Scrum. Estos procesos se caracterizan por estar basados en las etapas del ciclo de vida del software tradicionales, pero la gran diferencia es que se combinan algunas técnicas y por ende se hacen aún más entrelazadas en cuanto al orden en que se deben ejecutar, ya que no se sigue una ejecución lineal de las diferentes fases del desarrollo del proyecto.

Es importante tener claro que el modelo Scrum, está basado en el tipo de desarrollo incremental, es decir, que no es del todo lineal ya que para ir avanzando en el desarrollo del proyecto, el mismo debe pasar por una serie de iteraciones, las cuales al ir ejecutándose reiteradas veces incrementarán el tamaño del proyecto que se esté desarrollando, es por eso que se recomienda tener un equipo de desarrollo altamente capacitado, ya que esto garantiza un excelente funcionamiento de la aplicación a lo largo de cada sprint.

Un aspecto importante a rescatar es que a lo largo del ciclo de vida del sistema, el mismo podrá sufrir ciertos cambios ya sean por parte del cliente o a nivel interno o de desarrollo. Esto implica que el equipo debe ser lo suficientemente adaptativo para así hacer que los siguientes sprints (después de los cambios) no alteren el flujo de ejecución de las actividades.

Por último, pero no menos importante se encuentra la comunicación, que durante el ciclo de vida del sistema es muy importante, ya que el equipo está en constante interacción ya sea a nivel interno como externo, esto porque muchas veces esta metodología se presta para que representantes del cliente o él mismo estén involucrados en todas las fases de desarrollo, por lo que se debe tener una serie de medidas para mantener un clima organizacional adecuado entre las partes, y esto se logra desde lo más sencillo que es la comunicación adecuada.

A continuación, se muestran el conjunto de procesos que conformarán el ciclo de vida en el modelo Scrum que se está implementando:

1. Product Backlog
2. Sprint Backlog
3. Sprint Planning Meeting
4. Daily Scrum o Stand-up Meeting
5. Sprint Review
6. Sprint Retrospective

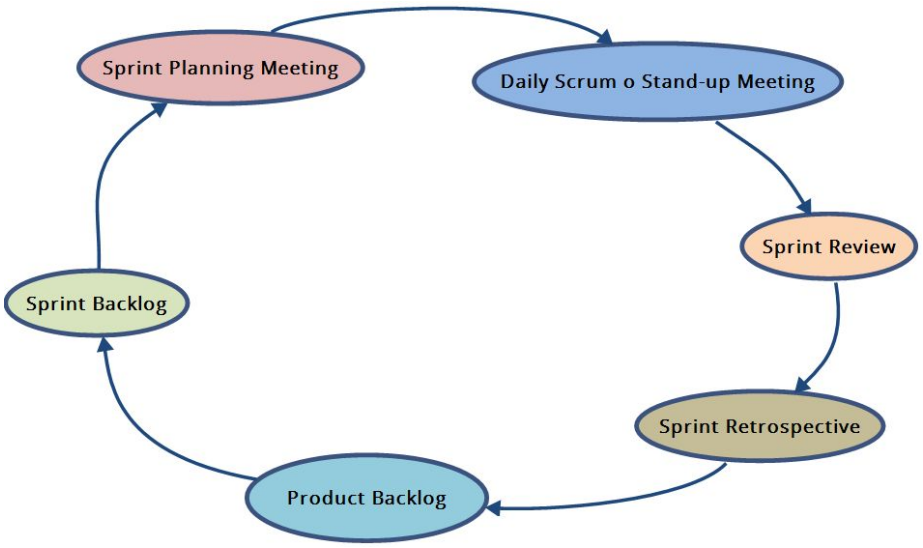

Figura 2: Ciclo de vida del modelo Scrum.

Estas son las fases del ciclo de vida del software en la metodología Scrum, donde el proceso que se lleva a cabo consiste básicamente en realizar un análisis de los requerimientos del sistema (Product Backlog) mismos que debieron ser definidos en un proceso de elicitación de requerimientos, además se señalan los objetivos a corto y mediano plazo dentro de un sprint. Posteriormente los desarrolladores empiezan la etapa de creación de las actividades definidas en el sprint, se realizan algunas pruebas y por último se realiza una retroalimentación de acuerdo a lo conseguido al terminar la última fase.

3.13. Condiciones Ambientales

Por el contexto de utilización del sistema se definen mayormente consideraciones legales y políticas.

- **3.13.1** El uso de imágenes de identificación de personas está sujeto a las leyes del país o estado en el que se encuentre el puesto fronterizo. Por lo que las imágenes muestrales proporcionadas con la primer versión del sistema, están siendo utilizadas bajo los permisos necesarios para el caso.
- **3.13.1** El sistema tiene objetivos meramente de reconocimiento de potenciales sujetos buscados por la ley, por lo que su diseño se establece para dicha situación, independientemente del uso que le dé el cliente luego de la entrega del producto.

4. Sección 4: Interfaces del sistema

Para comprender las diferentes interfaces del sistema, primero se deben mencionar las herramientas que se están implementando en el desarrollo de la aplicación. En primera instancia se encuentra el lenguaje de desarrollo python, que fungirá como backend de todo el app, y respecto al frontend se utilizará Django que es un framework especializado para trabajar con python, por lo que su utilización es casi obvia en esta aplicación.

Respecto a Django, este utilizará una base de datos sqlite, pues el desarrollo con esta es mucho más sencillo y como no se precisa tener una base de datos cargada con todas las imágenes existentes de los sujetos, el desarrollo en este manejador suele ser el adecuado y por ende el que se utilizará en el desarrollo de producto.

Otro aspecto a tener en cuenta son las hojas de estilos, pero en general los elementos visuales, pues en la aplicación implementará bootstrap y font-awesome, donde el primero será para adquirir una serie de estilos ya predefinidos pero que podrán ser modificados a nivel interno si se descarga la hoja de estilos completa, y el segundo es una especie de repositorio de íconos, mismos que se utilizarán para dar ese aspecto visual deseado por todo usuario.

Por otra parte, Se hará uso de un ambiente de desarrollo integrado específico para el lenguaje de programación Python llamado PyCharm, este provee herramientas necesarias para cumplir con los estándares mencionados (PEP 8 y PEP 257). PyCharm analiza el código fuente de forma estática, haciendo revisiones básicas de módulos, clases, funciones y variables, así como, de documentación interna y formato.

Además, para maximizar su utilidad se usará la extensión SolarLint, la cual proporciona de forma instantánea información útil sobre nuevos errores encontrados y problemas de calidad inyectados en el código fuente, lo cual, disminuye el tiempo que corresponde a las revisiones de la calidad del código. Cabe destacar que es una herramienta versátil, ya que puede analizar código en diferentes lenguajes usando como base el server de SonarQube, esto permitirá que distintos módulos del sistema, sin importar el lenguaje usado, puedan ser analizados. La extensión SolarLint es de gran utilidad, sin embargo, no cubre ciertas características o reglas puesto que su objetivo no es ralentizar el IDE, por tanto, cada vez que se alcance un punto de control del sistema, este será analizado por medio de SonarQube Scanner, el cual requiere un poco más de tiempo tanto para inicializar como para realizar el análisis completo.

En resumen, las dependencias serían las que acontecen pero no necesariamente son definitivas ya que pueden surgir cambios o conflictos a la hora de implementarse alguna de estas.

- Numpy
- Opencv
- Django
- Unittest
- Pydoc

En cuanto a las herramientas a utilizar están:

- SonarLint
- SonarQube
- PyCharm
- Estádares PEP8 y 257

Para finalizar sería pertinente mencionar los sistemas futuros, de los cuales se dependerá en un futuro o se utilizarán para dar nuevas implementaciones. Pero en este no se han definido expectativas de evolución de la aplicación, por lo que no se podría definir con claridad cuáles son los sistemas potenciales a utilizar ni mucho menos sus dependencias.

5. Sección 5: Verificación (Pruebas)

Para la verificación de la calidad del software se establecen las pruebas siguientes clasificadas según su necesidad de ejecución del sistema.

5.1. Pruebas Estáticas

Se define para cada uno de los módulos del sistema, las siguientes pruebas con un enfoque de caja blanca, mediante la revisión de la existencia de los aspectos listados a continuación dentro del código desarrollado, sin la necesidad de la ejecución del sistema para su comprobación

5.1.1. Análisis de la robustez

comprobación de la existencia de fragmentos de código dedicados a atrapar posibles excepciones esperadas durante la ejecución del sistema, relacionados a errores de ausencia de datos o erroneidad de los mismos.

5.1.2. Análisis de rendimiento

Análisis general y subjetivo de las funcionalidades principales, realizado por un supervisor sobre las prácticas implementadas y su enfoque al rendimiento del sistema.

5.1.3. Análisis del modelo

Inspección del código para confirmar el desarrollo del mismo con base al modelo definido en las etapas de diseño previas.

5.2. Pruebas Dinámicas

Se define para la funcionalidades principales del sistema como son la fase de entrenamiento y la fase de identificación o clasificación, así como sus componentes, una serie de pruebas con enfoque de caja negra, para confirmar su correcto funcionamiento mediante la ejecución del sistema y el ingreso de datos, buscando las salidas acordes esperadas.

5.2.1. Ejecución de las pruebas de Unit Testing

Posterior a la finalización de cambios dentro de la estructura del sistema, se deben ejecutar las pruebas de unit testing correspondientes al módulo que alberga los cambios realizados.

5.2.2. Pruebas de navegación del sistema

Se define la prueba de la navegación del sistema como una rutina de acceso a todas las interfaces disponibles en el mismo para asegurar la correcta disponibilidad de cada función desarrollada.

5.2.3. Pruebas de robustez

Ejecuciones del sistema sin restricción de procedimiento, para buscar las posibles fallas de robustez del mismo, buscar los puntos que requieren de manejo de excepciones y posibles fallos inesperados producto de fuentes diferentes a código erróneo.

6. Sección 7: Glosario

- **EigenFaces:** Auto Caras, que es el nombre de la aplicación y también del algoritmo que se está utilizando para reconocer los rostros.
- **APP:** Application, programa informático diseñado para permitir a uno o varios usuarios realizar ciertas tareas.
- **Scrum:** Metodología ágil y de avance incremental implementada para desarrollar el proyecto.
- **Product Backlog:** Bitácora del producto.
- **Sprint Backlog:** Bitácora de Sprint.
- **Sprint Planning Meeting:** Reunión de planificación del sprint.
- **Daily Scrum Meeting:** Reunión del scrum diario.
- **Sprint Review:** Revisión del Sprint.
- **Sprint Retrospective:** Retrospectiva del Sprint

7. Sección 6: Referencias

Referencias

[1] Berzal, F.(2006). *El ciclo de Vida de un Sistema de Informática*, Recuperado de: http://flanagan.ugr.es/docencia/2005-2006/2/apuntes/ciclovida.pdf

[2] Clemente, E. (2016). *Estándares de codificación*, Recuperado de: https://okhosting.com/blog/el-ciclo-de-vida-del-software/#Modelo_Scrum