Tecnológico de Costa Rica

Ingeniería en Computación

IC6831 -Aseguramiento de la Calidad del Software

Auditoría 1: basada en el estándar IEEE-1028-2008

Profesor: Saúl Calderón Ramírez

Estudiante:

José Miguel Mora Rodríguez

Dylan Rodríguez Barboza

Karina Zeledón Pinell

Noviembre, 2017

Contents

1 Introducción

En el presente documento se detalla un proceso de inspección que se realiza al proyecto realizado por el equipo de trabajo que redacta este documento, además, se detalla la auditoría hecha hacia otro grupo de trabajo, con el fin de corregir errores o agregar elementos críticos en un documento de obtención y análisis de requerimientos. Para lograr esto, se hizo uso de listas de chequeos para comprobar que todo se haya hecho de la mejor manera posible.

2 Inspección interna a proyecto

2.1 Propósito de la inspección

El propósito de la inspección del propio proyecto, es intentar detectar alguna anomalía o detectar alguna falta que este tenga, ya sea el no seguimiento de algún estándar o la mala estructuración de algún código fuente con el que se haya trabajado. Además, será útil para obtener información del software, de tal manera que se pueda mejorar algún estándar o implementación ya hecha. Entre los puntos que se quieren comprobar con esta inspección se encuentran:

- Comprobar el uso de una herramienta para el manejo del estándar de codificación seleccionado por el equipo.
- Comprobar que el código estuviese debidamente comentado.
- Comprobar si es posible simplificar algún algoritmo ya implementado.
- Comprobar que el código no presente o pueda presentar un futuro bug.
- Comprobar que haya pruebas de unidad para cada comportamiento del sistema implementado.
- Comprobar que las pruebas cubran las condiciones para que se de un error.
- Comprobar que se manejan excepciones en caso de que hayan errores en los parámetros.
- Comprobar si se implementó un componente que se encargue de brindar información relacionada con los errores.
- Comprobar que las variables globales sean accedidas correctamente, de manera que los datos no se corrompa en algún momento de la ejecución.
- Comprobar que no hayan olores de software en el código.
- Comprobar que no haya código de depuración (prints) en los programas que se suponen completamente implementados.

Estos son los puntos más importantes que se quieren comprobar, se tratarán otros, específicamente en una sección a lo largo del documento que se encarga del manejo de una lista de chequeo de la implementación del código.

2.2 Artefactos inspeccionados

Los artefactos de implementación inspeccionados para el proyecto de reconocimiento de rostros fueron todos los archivos que contengan código fuente, en los cuales la mayoría fueron implementados en el lenguaje de programación python, además de estos, algunos también fueron implementados en html.

Sección del archivo	Nombre del archivo	Programado	Parte del repositorio en donde se encuentra
Test	Test.py	Python	Entra al repositorio: Test.py
Test	Test2.py	Python	Entra al repositorio: Test2.py
Back-End	Trozos.py	Python	Entra al repositorio: Trozos.py
Back-End	Trozos.py	Python	Entra al repositorio/reconocedor/ Trozos.py
Back-End	urls.py	Python	Entra al repositorio/reconocedor/urls.py
Back-End	views.py	Python	Entra al repositorio/reconocedor/ views.py
Front-End	header.html	Html	Entra al repositorio/reconocedor/templates/ header.html
Front-End	home.html	Html	Entra al repositorio/reconocedor/templates/ home.html
Front-End	test.html	Html	Entra al repositorio/reconocedor/templates/ test.html

2.3 Estándares seguidos por la organización inspeccionados

El siguiente cuadro muestra los estándares utilizados en los archivos de código:

Estándar	Descripción de Estándar	Elementos en cual se aplicó
PEP8	PEP8 está dedicada a la recopilación de estándares seguidos por los desarrolladores de python para la librería estándar.	Codificación de los archivos de código (.py)

Javadocs	Javadoc es una herramienta de documentación que define un formato estándar para los comentarios, y que genera archivos HTML para ver la documentación desde un navegador web.	Documentacion interna en los archivos de código (.py)

2.4 Lista de chequeo de Codificación

A continuación se presentan los aspectos evaluados en torno a la codificación del proyecto:

Documentación

1. Se utiliza una herramienta para la verificación del estándar de codificación interna
   (a) Comentarios:
      i. Se utiliza el estándar de documentación "Javadocs"
      ii. La evidencia se presentan dentro de los archivos fuentes de codificación.
2. Todos las funciones se encuentran debidamente comentados:
   (a) Comentarios:
      i. Se utiliza el estándar de codificación Javadocs con su debido formato.
3. El código fuente incluye la etiqueta @autor:
   (a) Comentarios:
      i. No se utiliza este formato debido al uso de la herramienta de administración de configuración "Git".
4. Se utiliza la etiqueta @version para definir el versionamiento del código:
   (a) Comentarios:
      i. Se define la versión una vez se encuentra lista en el repositorio.
   (b) Evidencia:

5. Se explica el funcionamiento de algoritmos complejos:
   (a) Comentarios:
      i. Se explican dentro de la documentación interna
      ii. Evidencia:

```
##
#  Se cargan n imÃ¡genes .pmg de dimensiones l * a = p, donde l son la
#  cantidad de pixeles de largo y a la cantidad de pixeles de ancho,
#  de una carpeta especifica. Posteriormente se convierte cada una
#  en vectores de forma que se guarden en una matriz M de dimensiones
#  p * n. Por Ãºltimo calcula la matriz de covarianza de la matriz M
#  y la guarda en un archivo .txt de nombre MatrizCovarianza.txt en
#  la direcciÃ³n 'root'.
#
#  <p>
#  Esta funcion es el metodo principal de la aplicaciÃ³n por lo que
#  se debe de llamar para iniciar la aplicaciÃ³n.
#
#  @param  direccion  la direcciÃ³n de la carpeta donde se encuentran las
#                     imagenes a cargar
#
#  @return
##


def cargarImagen(files):
```

6. Las dependencias de librerías externas son son claramente establecidas
   (a) Comentarios:
      i. Se utilizan las librerías numpy y OpenCV
      ii. Se hace referencia a las librerías y su versión en el respectivo repositorio
   (b) Evidencia:
      i. https://github.com/dylanrodbar/ReconocedorRostros/tree/master/reconocedor
7. El código no depende de algún error en un framework externo el cual vaya a ser arreglado posteriormente afectando el funcionamiento del sistema.
   (a) Comentarios:
      i. No se ha utilizado ninguna función de las librerías externas la cual presente algún error.
8. Se documentan las métricas utilizadas en la implementación.
   (a) Comentarios:
      i. Todas las métricas utilizadas se documentan en el SyRS del sistema.
   (b) Evidencia:
      i. https://github.com/dylanrodbar/ReconocedorRostros/blob/master/Avance%202/SyRS/SyRS.pdf

Testing

1. Se incluyen pruebas unitarias las cuales comprueban el funcionamiento adecuado de las funcionalidades implementadas.
   (a) Comentarios:
      i. Se implementan con la librería pytest.
   (b) Evidencia:
      i. https://github.com/dylanrodbar/ReconocedorRostros/blob/master/Test.py
      ii. https://github.com/dylanrodbar/ReconocedorRostros/blob/master/Test2.py
2. Las pruebas unitarias cubren los posibles casos de error.
3. No se codificaron algoritmos codificados previamente en otros API.
   (a) Comentarios:
      i. Se implementaron los algoritmos de PCA y KMeans sin utilizar APIs externos con fines didácticos.
      ii. Se pueden reemplazar con la librería SciPy.
4. Asegurarse que las pruebas unitarias verifican la codificación contra los requerimientos.
   (a) Comentarios:
      i. Los requerimientos del sistema son verificables a partir de pruebas de integración y de sistema y no por las pruebas unitarias.

Manejo de Errores

1. Se validan parámetros nulos de forma temprana en los métodos
   (a) Comentarios:
      i. No se validan los parámetros
      ii. La validación de los parámetros está programada para ser realizada en un fecha posterior al Deadline de este documento, sin embargo al finalizar el sprint actual se contará con dichas validaciones.
2. Se utilizan error handlers generales:
   (a) Comentarios:
      i. No se utilizan
      ii. Se planean utilizar para el manejo de errores al finalizar el sprint.
3. Un error handler debe liberar de forma adecuada los recursos utilizados
   (a) Comentarios:

i. No se utilizan error handlers.

4. No se utilizan RuntimeException o Exception, o subclases para evadir modificar el código con el fin de implementar un adecuado manejo de errores.
   (a) Comentarios:
      i. No se utilizan este tipo de excepciones.
5. Se crean clases de manejo de Excepciones específicas a las excepciones que pueda presentar el sistema.
   (a) Comentarios:
      i. No se implementan.

Rendimiento

1. Los objetos son duplicados solamente cuando son necesarios
   (a) Comentarios:
      i. No se implementa ninguna duplicación de objetos.
2. No se implementan ciclos "busy-wait" en vez de técnicas apropiadas de sincronización de threads.
   (a) Comentarios:
      i. No se implementan threads.
3. No se utilizan objetos de gran tamaño en memory o Strings los cuales contienen archivos de gran tamaño. Por ejemplo: No se guardan archivos XML en Strings o en el caso de imágenes no se copian todas las que se vayan a utilizar en memoria antes de ser utilizadas.
   (a) Comentarios:
      i. En el sistema se maneja una gran cantidad de imágenes.
      ii. Las imágenes se procesan con la librería OpenCV.
      iii. Una vez procesada, la imagen es liberada de memoria.
   (b) Evidencia:

```
for i in files:
    imagen = i.read()
    img = cv2.imdecode(np.fromstring(imagen, np.uint8), -1)
```

4. No hay código correspondiente a procesos de debugging
   (a) Comentarios:
      i. Al finalizar el sprint actual no se encontrará código de este tipo
5. No hay impresiones a consola innecesarias
   (a) Comentarios:
      i. Al finalizar el sprint actual no se encontrará código de este tipo

2.5 Artefactos corregidos

Los artefactos que se corrigieron a lo largo del proceso de auditoria se encuentra en el repositorio: https://github.com/dylanrodbar/ReconocedorRostros/Inspeccion

3 Auditoría a proyecto externo

Se realizó una auditoría a un artefacto del proyecto del sistema de reconocedor de rostros desarrollado por un grupo externo, a continuación se presentan más detalles al respecto

3.1 Propósito de la auditoría

El propósito de esta auditoría será proveer una evaluación de cómo se conforma el producto de software del equipo al que se debe auditar, además de los procesos que se aplican a estándares, regulaciones, planes, especificaciones y procedimientos. Para esta auditoría se evaluarán los requerimientos que recolectó la organización auditada.

Entre los puntos que se quieren comprobar con esta inspección se encuentran:

- Comprobar si hay una estructura adecuada para la recolección de requerimientos.
- Revisar estándares de SyRS y actividades usadas.
- Comprobar si se definen todos lo términos y unidades de medida.
- Comprobar si cada requerimiento está descrito individualmente y con una prioridad fijada.
- Comprobar las referencias que hay entre los requerimientos.
- Comprobar si todas las clases de usuario están debidamente descritas.
- Comprobar si se definieron adecuadamente los atributos de calidad.

Estos son los puntos más importantes que se quieren comprobar, se tratarán otros, específicamente en una sección a lo largo del documento que se encarga del manejo de una lista de chequeo de la recolección de requerimientos.

3.2 Organización auditada

La organización auditada corresponde al grupo conformado por:

- Luis Alonzo Cascante Franco.
- Olman Leivin Castillo Picado.
- Maria Laura de Jesús Pizarro Moreno.
- Gabriel Josué Venegas Castro.

3.3 Artefactos auditados

Dentro de la auditoría se auditaron los siguientes ICS:

ICS	Descripción	Dirección del repositorio
SyRS	Documento de requerimientos del proyecto	https://github.com/Hamdog28/Asdecalidad-1
Atributos de Calidad	Documento con los atributos de calidad del estándar IEEE-730-2002 especificando las métricas y umbrales de cumplimiento	https://github.com/Hamdog28/Asdecalidad-1/blob/master/1503897631-Atributos_de_Calidad_y_M%C3%A9tricas.pdf

3.4 Estándares seguidos por la organización auditada

Los siguientes estándares fueron seguidos por la organización auditada y serán tomados en cuenta a la hora de realizar la auditoria correspondiente:

Estándar	Descripción de Estánda	Elementos en cual se aplico
ISO-9126	Establece un modelo de calidad y su uso como marco para la evaluación de software, definiendo seis características.	Documento de especificación de los requerimientos del sistema (SyRS)
IEEE-730-2002	Este estándar establece los requisitos para iniciar, planificar, controlar y ejecutar los procesos de Aseguramiento de calidad de software (ACS) de un proyecto de desarrollo o mantenimiento de software.	Actividades de Aseguramiento de calidad
IEEE-828-1998	El estándar IEEE 828 define la información mínima requerida para llevar un Plan de Gestión de la Configuración del Software	Plan de administración de la configuración

PEP8	PEP8 está dedicada a la recopilación de estándares seguidos por los desarrolladores de python para la librería estándar.	Codificación de los archivos de código (.py)

3.5 Lista de chequeo

La siguiente lista fue utilizada para la auditoría del proyecto:

Id del Ítem	Descripción	Cumplimiento y observaciones
1	Se siguieron los estándares de documentación establecidos de forma correcta.	Sí, se hizo una revisión del SyRS y se siguió el estándar fijado en el momento (ISO/IEEE - 29148-2011), por ejemplo, se describió correctamente el propósito del sistema, su alcance, definieron correctamente los requerimientos funcionales, de manera que no se veían ambigüos, finalmente, definieron correctamente las interfaces y operaciones del sistema.
2	Se definen con claridad todas las métricas.	Sí, se puede notar en la especificación de adaptabilidad y condiciones ambientales definidos en el SyRS.
3	Todos los requerimientos cuentan con un nivel apropiado y consistente de detalle.	Son entendibles, sin embargo, se pudo haber manejado una tabla en la que se indique un poco más claro los casos que pueden ocurrir por cada requerimiento.
4	Los requerimientos poseen su respectiva descripción y nivel de prioridad.	Sí, cada requerimiento tiene su respectiva descripción, en la cual se detalla que debería hacer cada módulo del programa final.
5	Se presentan las referencias cruzadas entre requerimientos.	Sí, se puede evidenciar en las secciones de los requerimientos funcionales, de usabilidad y de rendimiento.

6	Se describen todos los tipos de usuarios. Se describen las características de dichos usuarios.	Sí, se puede evidenciar en la sección del alcance del sistema en el SyRS.
7	La especificación incluye todas las necesidades de usuario o de sistema. Se especifican todas las funcionalidades a ser utilizadas por los usuarios del sistema.	Sí, se puede evidenciar en las secciones de modos y estados del sistema y las interfaces del sistema.
8	Los requerimientos funcionales especifican entradas, salidas y su debida descripción de funcionalidad.	Sí, en la sección de requerimientos funcionales se pueden apreciar las entradas y salidas. Sin embargo, como se mencionó en un punto anterior, con el uso de una tabla sería más clara este tipo de información.
9	Se especifican los requerimientos de documentación y entrenamiento de usuarios.	Sí, se evidencia en la sección de la adaptabilidad de los requerimientos.
10	Se especifican los entornos de ejecución del software.	Sí se especifican, se puede evidenciar en la sección de requerimientos de usabilidad.
11	Se especifica la forma de soporte y mantenimiento durante todo el ciclo de vida del proyecto.	Sí, se posee una sección que solamente se dedica a explicar el ciclo de vida y mantenimiento del sistema.
12	Se incluyen restricciones de diseño e implementación.	Sí, en la sección de visión general del sistema se tiene un diagrama que explica el MVC y un diagrama de casos de uso en la sección de modos y estados del sistema.

13	Se especifican requerimientos de seguridad.	Sí, explican la seguridad según el tipo de usuario, como sugerencia se puede añadir, de nuevo, el uso de tablas que permitan un mejor entendimiento de lo que se explica.
14	Se incluyen requerimientos de privacidad de datos.	No se incluyen, o no se especifican de manera separada en alguna sección.
15	Se identifican requerimientos relacionados al rendimiento y comportamiento temporal del sistema.	Sí, se pueden encontrar en la sección de requerimientos del rendimiento del SyRS de la organización auditada.
16	Se especifican de forma apropiada todos los atributos de calidad.	Sí, definieron un documento solamente dedicado a tratar los atributos de calidad, en donde definieron cuánto peso tiene cada una y qué métricas se aplicarían para cada caso.
17	No existen requerimientos duplicados ni conflictos entre los mismos.	Sí, cada requerimiento era único, por lo que no podrían considerarse duplicados.
18	Cada requerimiento tiene solo una intrepretacion	Sí, no hay ambigüedad en los requerimientos, fueron fáciles de interpretar, sin embargo, como se ha mencionado antes, se recomienda que tengan los requerimientos especificados mediante tablas, esto permite que se entienda mejor.
19	Los requerimientos son verificables.	Sí, incluso definieron las maneras y programas utilizados para las pruebas, destaca el unittest de python.
20	Se presentan criterios de aceptación medibles para los requerimientos tanto funcionales como no funcionales.	Sí, se definieron criterios de aceptación, es decir, los casos y resultados deseables para cada módulo por implementar.

21	Cada requerimiento presentan un identificador único.	Sí, manejan los requerimientos por medio de un id único, por lo que cada uno podrá ser ubicado mediante esta modalidad.
22	Cada requerimiento es rastreable a su fuente.	Sí, se incluyen capturas y manuales que permiten llegar a los entornos en los que se desarrollan los requerimientos.
23	Todos los requerimientos se encuentran dentro del alcance del proyecto.	Sí, no definieron ni más ni menos que lo definido en la sección del alcance del sistema en el documento del SyRS de la organización auditada.
24	Los requerimientos no son ideas ni soluciones de diseño e implementación.	Sí, son requerimientos únicamente, no se incluye un nivel más detallado o de implementación, por lo que el objetivo es más entendible sin tener que ir por conceptos técnicos.

3.6 Recomendaciones

Las principales recomendaciones para la organización auditada son relacionadas con el formato que le dieron a los documentos relacionados con la recolección de los requerimientos y definición de atributos de calidad, entre los cuales destacan:

- Se recomienda hacer uso de las tablas, con secciones significativas, de esta forma, será más fácil su comprensión.
- Ampliar un poco más algunas explicaciones, a pesar de que la mayoría estuvo presente en el documento, a algunos puntos le faltaron un poco de detalle, tales como la sección de las interfaces del sistema.
- Se recomienda agregar referencias bibliográficas de los estándares que sigan para la realización de los documentos.

4 Bibliografía

1. IEEE. (2008). IEEE Standard for Software Reviews and Audits. New York: IEEE