Boca Analitycs

Instituto Tecnológico de Costa Rica
Escuela de computación
Ingeniería en computación
Aseguramiento de la calidad de software

Joshua Mata Araya
Adrian López Quesada
Josué Arrieta Salas

7 de septiembre de 2016

Índice

1. Especificación de Requerimientos del sistema

En esta sección se especifican los requerimientos tanto funcionales como no funcionales del sistema a implementar que se ofrecerá a la junta directiva del Boca Juniors; basados en los atributos ISO-9126 del documento anterior. Esta especificación está basada en el estándar ISO/IEC/IEE 29148-2011.

1.1. Propósito del sistema

Debido al estimulamiento del análisis de videos de forma automática en los últimos años y la necesidad de la organización argentina de fútbol Boca Juniors; SportAnalytics tiene pensado en construir un software que analice de forma automática videos deportivos (específicamente fútbol). Daniel Angelici ha contactado a SportAnalytics para tal propósito.

1.2. Alcance del sistema

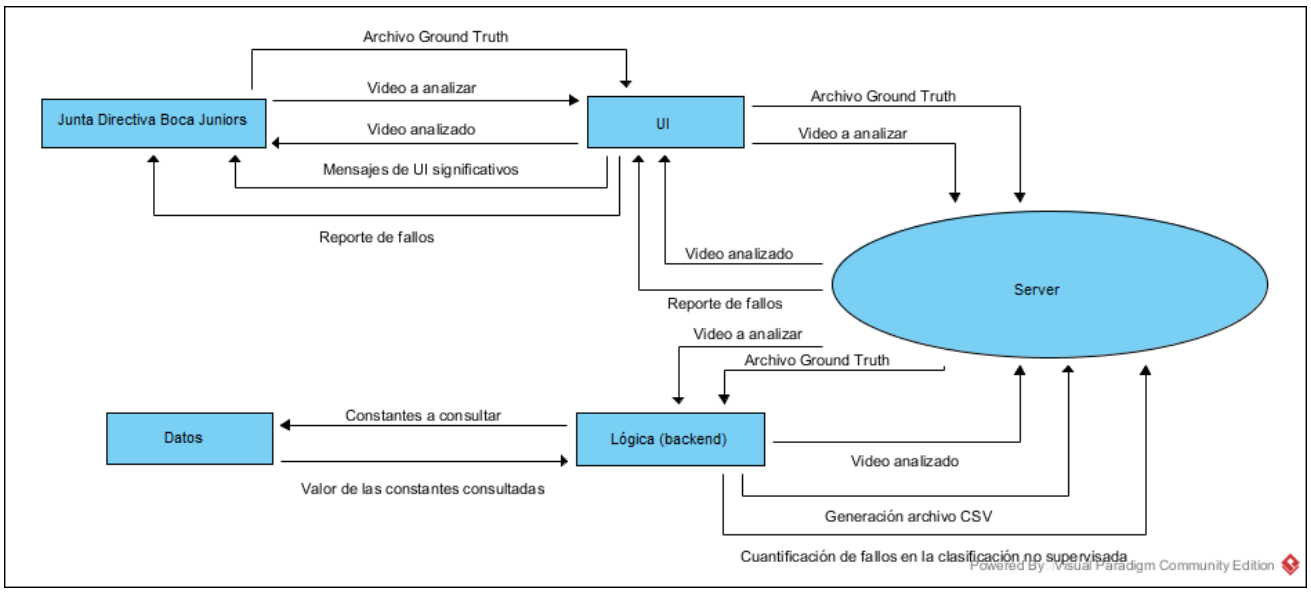
El sistema a implementar se llamará **BocaAnalytics**. Boca Juniors necesita un análisis automático o semiautomático con el fin de poder analizar diversos aspectos de desempeño del club de fútbol. Para solucionar este problema, y como objetivo principal del programa, este deberá realizar una de manera no supervisada, determinar la pertenencia de cada jugador a alguno de los dos equipos en un partido de fútbol. De esta manera se obtienen segmentos de un video con información útil, donde se puede mostrar las posiciones de cada jugador en un cuadro, de un equipo específico. Como objetivo secundario se pretende generar informes de la cantidad de jugadores en cada equipo.

Los principales beneficios de esta aplicación sería el ahorramiento de tiempo para el equipo técnico a la hora de analizar los partidos del Boca contra otros equipos; además tal software podría mejorar la toma de decisiones tácticas y estratégicas (tanto defensivas, como ofensivas) para el equipo deportivo, de tal forma que el rendimiento del equipo mejoraría. Se ha de mencionar que el sistema no pretende generar un informe con la duración por escenas, o la obtención de datos sobre la consistencia de la línea defensiva, de medio campo u ofensiva. Solo lo mencionado en el primer párrafo de esta sección.

1.3. Visión del sistema

1.3.1. Contexto del Sistema

Debido a que BocaAnalytics trabajará en un ambiente web (luego se especificará con más detalle el ambiente), se tiene una arquitectura cliente/servidor. Se selecciona esta arquitectura porque permite un sistema escalable (el número de clientes o servidores puede crecer sin problema), además permite la centralización del control (tanto accesos como recursos). Se creó el siguiente diagrama de contexto:



Se puede ver cómo es la interacción del sistema con el usuario (el usuario en este caso carga un video o archivo Ground Truth al servidor) al frontend. Se puede ver a su vez, cómo el frontend, que por medio de un servidor, se comunica con el backend. El backend es el encargado de toda la lógica de negocios requerida por la junta directiva del Boca Juniors. Una ejecución normal del programa sería: el usuario por medio de una interfaz gráfica, carga el video a analizar al servidor. Este redirecciona la entrada a un backend. En el backend se realizan los cálculos y lógica de negocio necesarios; y envía el video analizado al servidor. A su vez genera un archivo CSV (reportes generados) y se guarda en el servidor. Finalmente el servidor envía el video procesado a la interfaz, y esta le permite al usuario descargar el video final. También se podría mostrar en pantalla la cantidad de fallos en la clasificación no supervisada, si el usuario presento como entrada el archivo de ground truth.

Se ha de mencionar que en presente diagrama no se modela ningún motor de base de datos. Los "datos"presente son un conjunto de variables estáticas en una clase de java, simulando una base de datos.

1.3.2. Funciones del Sistema

El sistema debe de tener las siguientes capacidades:

- El usuario debe poder cargar un video.
- El sistema deber permitir al usuario descargar el video analizado con los blobs y etiquetas por equipos marcadas.
- El sistema debe generar un reporte interno en formato CSV con la cantidad de jugadores por equipo, por frame.
- La interfaz de usuario debe poder mostrar el tiempo en procesar el video en pantalla.
- El usuario debe poder cargar un archivo de Ground Truth.
- El sistema debe poder cuantificar la cantidad de fallos en la clasificación no supervisada.

El sistema posee las siguientes restricciones o condiciones:

- El video cargado por el usuario debe estar en formato: .mp4, .webm, .ogg.
- El video producido por el sistema será en formato .mp4 únicamente.
- El archivo CSV generado no será mostrado en pantalla ni procesado. Se deberá acceder del servidor.

1.3.3. Características del Usuario

El usuario de la aplicación será toda la junta directiva del Boca Juniors, específicamente el cuerpo técnico del equipo de fútbol; donde cabe destacar el actual presidente Daniel Angelici, vicepresidentes: Horario Paolini, Rodolfo Ferrari, Dario Richarte y secretarios: Gustavo Ferrari, Carlos Aguas y Pedro Orgambide. Su función principal sería la de subir un video de fútbol para ser analizado, y descargar el video analizado automáticamente anteriormente subido. Se espera que este grupo de usuarios sea de alrededor unas 15 personas. Debido a que no son usuario técnicos, se espera que su dispositivo de uso sea una página web con interfaz de usuario amigable (en secciones posteriores se especificarán los distintos requerimientos relacionados al tema), para su fácil operabilidad.

1.4. Requerimientos Funcionales

Se tienen los siguientes requerimientos funcionales:

- La junta directiva podrá cargar un video de fútbol para ser analizado.
- La junta directiva podrá descargar un video analizado: se debe de poder visualizar los blobs o regiones correspondientes a jugadores junto con una etiqueta correspondiente por equipo al cuál el jugador pertenezca. También se muestra en el video el número de jugadores detectados en cada cuadro.
- Generación de reportes de la cantidad de jugadores en cada equipo, por frame.
- La junta directiva deberá poder visualizar el tiempo en procesar el video.
- La junta directiva podrá cargar un archivo de ground truth: el sistema debe analizarlo para cuantificar la cantidad de fallos encontrados.

Para garantizar tales requerimientos, se tienen las siguientes métricas:

Característica	Métrica	Nivel Requerido	Resultado Actual	Herramienta
Idoneidad	Los requerimientos implementados están de acuerdo a lo implementado	La cantidad de requerimientos implementados entre la cantidad de requerimientos funcionales establecidos es mayor a 0.90, abarcando los más importantes dentro de lo implementado.	-	Selenium
Exactitud	Cantidad de errores en los resultados de los algoritmos	Cantidad de errores entre cantidad de pruebas menor o igual a 0.10	-	JUnit

Nota: la justificación de selección de estas métricas se especificaron en el documento de métricas previo a este.

1.5. Requerimientos de Usabilidad

Se espera que el sistema, como requerimiento de usabilidad, posee una interfaz gráfica de usuario amigable, y muestre en pantalla mensajes significativos para el usuario (se debe de poder mostrar el resultado del ground truth: cantidad de fallos en la clasificación no supervisada). Se espera que sea simple de usar y de aprender; además que la interfaz mantenga consistencia entre sí. Esto debido al

tipo de usuarios de la aplicación (no técnicos), es de suma importancia que no tengan problemas al utilizar la aplicación. Para garantizar estos requerimientos de usabilidad se utilizarán las siguientes métricas:

Característica	Métrica	Nivel Requerido	Resultado Actual	Herramienta
Capacidad de ser aprendido (externa)	Tiempo requerido por usuario para usar el programa (de entrada de video a salida de video)	Menos de 160 segundos	-	Java System Timer
Capacidad de ser entendido (externa)	El usuario entiende lo que es requerido cómo entrada al programa, así cómo la salida del mismo	$T = \frac{A}{B} = 1$,donde A es el número de entradas y salidas que el usuario entiende y B es el número total de entradas y salidas del programa	-	Google Forms
Capacidad de ser operado (externa)	Consistencia de los componentes de la interfaz de usuario (funcionalidades y mensajes de interfaz)	$T = 1 - \frac{A}{B} > 0{,}85$,dónde A es el número de componentes de interfaz considerados inconsistentes y B es el número total de componentes de interfaz	-	Google Forms

Nota: la justificación de selección de estas métricas se especificaron en el documento de métricas previo a este. La consideración de incluir más requerimientos relacionados con la usabilidad se pueden agregar en futuras iteraciones.

Como requerimiento específico relacionado con la usabilidad se tiene:

- La funcionalidad de mostrar en pantalla la cantidad de errores en la supervisión no autorizada deberá ser mostrada en pantalla al final. Este requerimiento será de prioridad baja, ya que no es un aspecto requerido por la Boca Juniors.

1.6. Requerimientos de Rendimiento

Se espera que espera que el sistema sea eficiente y con una utilización baja en memoria. Esto porque se piensa que si el sistema es lento, podría considerarse mas bien un estorbo para la Boca Juniors; ya que necesitan reportes además de automáticos, que sean rápidos. Los requerimientos y métricas se muestran en la siguiente tabla:

Característica	Métrica	Nivel Requerido	Resultado Actual	Herramienta
Capacidad de comportamiento temporal (externa)	Tiempo de procesamiento requerido por frame	Menos de 20ms	-	Java System Timer
Utlización de recursos (interna)	Porcentaje de memoria usado por el programa	Máximo 5 %	-	Task Manager

Nota: la justificación de selección de estas métricas se especificaron en el documento de métricas previo a este. La consideración de incluir más requerimientos relacionados con la eficiencia se pueden agregar en futuras iteraciones.

1.7. Interfaces del Sistema

Según el diagrama de componentes del POC,en al sección 1, pagina 2. Existen tres 4 módulos grandes que necesitan establecer 3 comunicaciones internas entre ellos.

La primer Deberá existir una conexión entre el servidor y la interfaz web. Por lo tanto se piensa utilizar el protocolo intercambio de datos JSON. Se escogió este protocolo ya que es bastante sencillo de escribir y entender por lo humanos, ademas de que es una manera de envió de datos de bajo consumo de memoria. Ademas, es sencillo para computar y generarlo de manera automática.

Para el envió de mensajes informativos, se utilizara:
{msg: mensaje de ayuda informativo}

Para el cambio de urls para acceder archivos, se utilizara:
{url: /nuevo_url}

Para la manipulación de tiempos, se utilizara:
{time: tiempo_procesamiento}

Para siguientes iteraciones se especificaran mas protocolos entre el servidor y la interfaz en caso de ser necesarios.

Se utilizara el protocolo de manejo de datos DTO para poder comunicar el servidor con la lógica. Se escogió utilizar este protocolo para generar una mayor separación de capas, así el servidor solo se encargara de establecer el protocolo necesario según la Interfaz Gráfica, en este caso por JSON, y aunque no se deban enviar muchos datos del usuario al software, a un futuro ayudaría a modularizar el envió de estos datos.

Por otro lado, debe existir una interfaz

Para la comunicación del paquete lógico con los datos no se implementara ningún tipo de protocolo de comunicación ya que se estará tratando con valores estáticos, sin acceso a una base de datos o archivo con información.

Finalmente, estaría la comunicación con los usuario la cual no esta especificada en el diagrama de componentes, sin embargo se establecerá inicialmente de manera a prosa los componentes para que el usuario pueda utilizar para comunicarse e interactuar con el sistema. Inicialmente, el usuario deberá tener acceso a un computador con mínimo un mouse, teclado y monitor como dispositivos de salida y entrada. Posteriormente, al acceder a la pagina web por medio de una URL, se establece el siguiente protocolo de comunicación:

1. Un botón para subir el video
2. Un boton para subir el GroundTruth
3. Una barra del progreso, mensajes informativos y tiempo de espera
4. Un botón para descargar el video
5. Visualizador del video en linea.

En caso de mas requerimientos funcionales en futuras iteraciones es posible modificar o agregar protocolos de comunicación entre el usuario y el sistema.

1.8. Operaciones del Sistema

1.8.1. Requerimientos de Integración Humana

No hay documentos aplicables, ni tampoco ningún requerimiento único ni especial relacionado con la integración humana (referida a interacciones entre personal/equipo). Esto debido a la naturaleza

propia del sistema a implementar, y los usuarios del mismo (no técnicos). Tampoco no hay ningún área, estación o equipo específico que necesite ingeniería humana concentrada (en caso de una operación sensible o crítica). Mas bien al contrario, se espera que el sistema sea lo más sencillo de usar, y se cumplan los requerimientos especificados en la sección de requerimientos de usabilidad.

1.8.2. Mantenibilidad

Se pretende que el sistema sea fácil de entender, mantener, debuggear y testear. Se pretenden estos requerimientos para que el equipo de desarrollo, si se quiere realizar un cambio o corregir un bug, lo pueda hacer rápido y esto mejor el tiempo de desarrollo. Como conclusión la organización Boca Juniors recibe un sistema de mayor calidad. Para garantizar la mantenibilidad se tienen las siguientes métricas y requerimientos:

Característica	Métrica	Nivel Requerido	Resultado Actual	Herramienta
Capacidad para ser analizado (interna)	Indice de complejidad ciclomatica	Igual o por debajo de 10	-	JHawk5 Java Metrics
Capacidad para ser probado (interna)	Cobertura de las pruebas por el codigo (Code Coverege)	Al menos 80 %	-	EclEmma
Capacidad de ser cambiado (interna)	Indice de mantenibilidad	Al menos un 80 %	-	JHawk5 Java Metrics

Nota: la justificación de selección de estas métricas se especificaron en el documento de métricas previo a este.

1.8.3. Confiabilidad

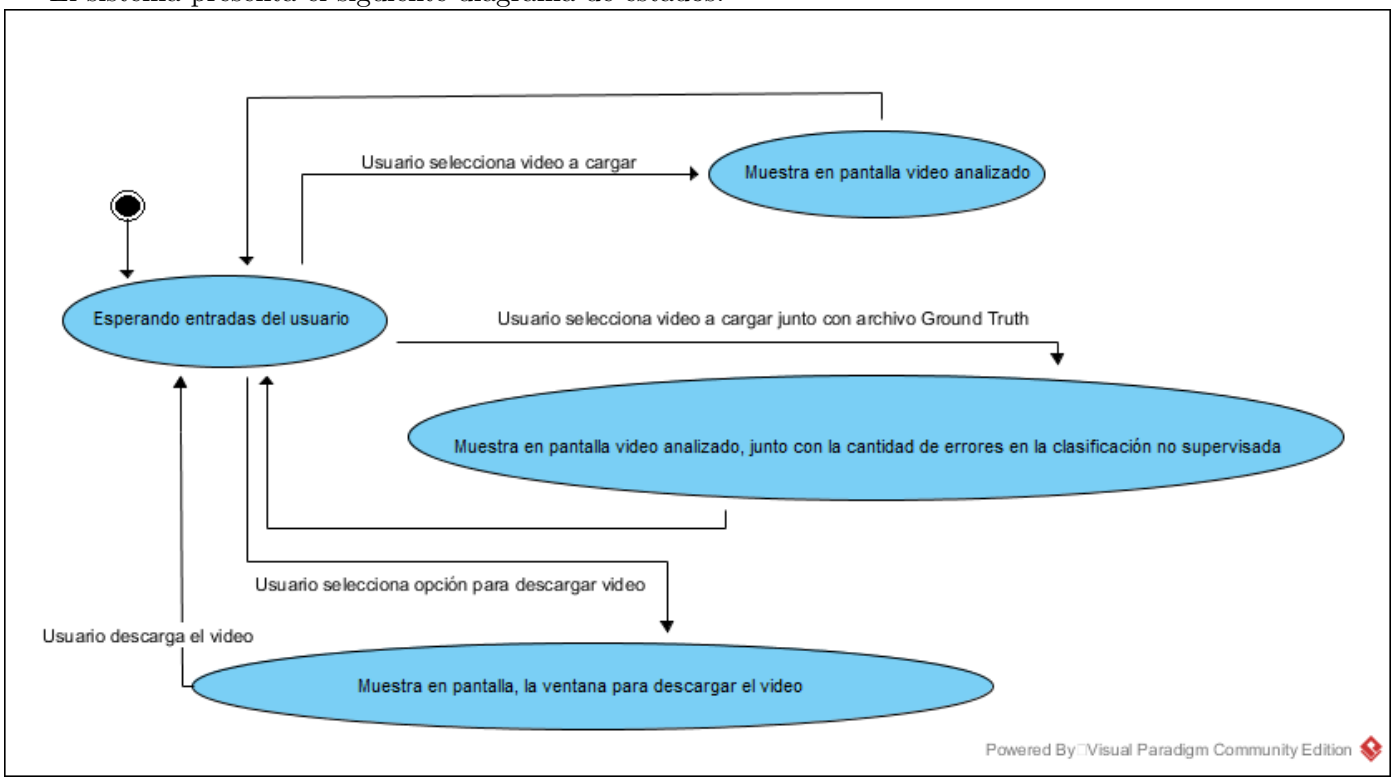
Se espera que el sistema trabaje opere bajo condiciones dadas (el ambiente operacional de la organización Boca Juniors) con la cantidad mínima de errores. Se espera un sistema robusto con buena recuperabilidad y tolerancia a fallos. Esto porque si el sistema presenta fallas y constantemente se cae, la asociación Boca Juniors podría perder la credibilidad en SportAnalytics. Como resultado se podría cancelar el proyecto. Se muestran las métricas y requerimientos relacionados con la confiabilidad:

Característica	Métrica	Nivel Requerido	Resultado Actual	Herramienta
Madurez	Cantidad de elementos probados en el software	La cantidad de historias de usuario que presentan un test automatizado entre la cantidad de historias de usuario definidas debe de ser mayor a 0.85	-	JUnit, Selenium
Tolerancia a fallos	Cuantos fallos es capaz el sistema de soportar sin necesidad de caerse por completo	La cantidad de caídas del sistema entre la cantidad de fallos detectados en el sistema debe de ser menor al 0.05	-	Log del sistema
Recuperabilidad	Tiempo que el sistema logra estar sin caerse	La cantidad de horas que el sistema se encuentre caído entre la cantidad de horas que lleva el software en producción debe de ser menor a 0.03	-	Log del sistema

Nota: la justificación de selección de estas métricas se especificaron en el documento de métricas previo a este. La consideración de incluir más requerimientos relacionados con la confiabilidad se pueden agregar en futuras iteraciones.

1.9. Nodos del Sistema y Estados

El sistema presenta el siguiente diagrama de estados:



El sistema solamente presenta cuatro estados. El primero sería cuando el sistema inicia y esa esperando una entrada del usuario (estado inicial). Los otros tres estados muestran en pantalla el resultado de procesamiento ejecutado por el sistema. Esto debido a la naturaleza propia del sistema: el usuario presenta ciertas entradas al sistema (en este caso son tres: 1) un video para analizar, 2) un video para analizar junto con un archivo de ground truth y 3 el usuario desea descargar un vídeo del servidor; probablemente el video resultado de los estados anteriores) y el sistema muestra la salida en pantalla. No hay estados intermedios.

1.10. Características Físicas

1.10.1. Requerimientos Físicos

Debido a que el sistema es un software a implementar no hay restricciones en cuanto al peso, dimensión o volumen del producto a diseñar. El sistema será usado en las computadoras del equipo técnico del Boca Juniors, y para poder instalar el sistema se necesitan los siguientes requisitos:

- JDK 8.0 o superior.
- Tomcat 8.0
- OpenCV-3.1

Nota: La consideración de incluir más requerimientos físicos se pueden agregar en futuras iteraciones.

1.10.2. Requerimientos de Adaptabilidad

Se espera que el sistema sea fácil de adaptar de un entorno a otro. Debido a que es un sistema web, para este caso, que sea adaptable de un navegador web a otro; y que además, sea compatible con un buen número de navegadores. Se necesita que sea fácil de instalar, esto porque el usuario no es un usuario técnico; y se quiere lograr la mejor experiencia de usuario posible con la Boca Juniors. También se necesita que el sistema pueda coexistir con otros sistemas. Dichos requerimientos y métricas

se muestran en la siguiente tabla:

Característica	Métrica	Nivel Requerido	Resultado Actual	Herramienta
Atributo de Adaptabilidad (externa)	Sistema de Software adaptable al entorno	Al menos 3 navegadores diferentes con aprobación de $X \geqslant 90\,\%$	-	Manual, X= 1-$\frac{A}{B}$ donde A: prueba funcional fracasada, B: cantidad de pruebas funcionales realizadas
Atributo de Instalabilidad (externa)	Tiempo de instalación desde cero sin ayuda externa	$X \geqslant 75\,\%$ Donde el tiempo máximo de instalación sera de 45 minutos.	-	Manual, X = 1-$\frac{A}{B}$.A: tiempo requerido para la instalación, sin contar tiempos de descarga. B: Tiempo máximo de instalación.
Capacidad de coexistencia (externa)	Coexistencia disponible	$X \geqslant 90\,\%$, 90 % de disponibilidad.En el transcurso de 3 horas.	-	X = $\frac{A}{B}$ donde A: cantidad de errores encontrados, B: tiempo transcurrido de funcionamiento. Se utilizara los logs del software y detección de errores del Sistema Operativo para monitorear errores de consistencia

Nota: la justificación de selección de estas métricas se especificaron en el documento de métricas previo a este. La consideración de incluir más requerimientos relacionados con la adaptabilidad se pueden agregar en futuras iteraciones.

1.11. Condiciones del Ambiente

El sistema operará en un en las facilidades u oficinas de la organización Boca Juniors, a una temperatura óptima ya que se espera que haya aire acondicionado. Por lo tanto condiciones como: lluvia, viendo, temperatura, fauna, hongos, maleza, tierra, spray, polvo, radiación, químicos, calor, movimientos, cargas electromagnéticas, shocks, no serán tomados en cuenta. Es un ambiente controlado. También será un sistema legítimo en el tema político, legal; ya que seguirá todas las regulaciones impuestas. En el ámbito social, si el sistema causa que la Boca Juniors mejore sus tácticas, mejorará el desempeño del equipo; y traerá entretenimiento a la sociedad ya que la calidad del fútbol mejora. Si se toma un punto de vista económico (tomando en cuenta el argumento anterior), traerá más ganancias a la Boca por su éxito.

1.12. Seguridad del Sistema

El sistema no implementará ningún procedimiento de loggeo, ni contraseñas, ni cuentas de usuario. Esta fuerte decisión se toma ya que se asume que la red en donde se instalará BocaAnalytics será la de la organización Boca Juniors, y se asume que este red es completamente segura. La aplicación dependerá de la seguridad de esta red. Se utiliza la siguiente métrica:

Característica	Métrica	Nivel Requerido	Resultado Actual	Herramienta
Seguridad	Qué tan frecuente se corrompen datos en el sistema.	La cantidad de veces que el sistema presenta errores de corrupción de datos entre la cantidad de veces que el usuario ingresa datos es menor o igual a 0.10	-	Selenium y JUnit

Nota: la justificación de selección de esta métrica se especificó en el documento de métricas previo a este. La consideración de incluir más requerimientos relacionados con la seguridad se pueden agregar en futuras iteraciones.

También se tomarán medidas de copias de seguridad y recuperación de los archivos del servidor, esto para garantizar la protección de los datos. Se espera que se realice una copia de seguridad del servidor diario, en la noche, cuando el sistema no está en uso.

1.13. Manejo de Información

El sistema debe recibir la siguiente información:

- Video a analizar: con formato .mp4, .webm, .ogg y debe pesar como máximo 200 mb.
- Archivo Ground Truth: otro video con formato .mp4, .webm, .ogg y debe pesar como máximo 200 mb.

El sistema debe generar y guardar (pero no mostrar en pantalla) la siguiente información

- Informe de la cantidad de jugadores de cada equipo: se guardará en un archivo .csv y se espera que pese menos de 2mb.

El sistema debe exportar la siguiente información:

- Video Analizado: será de formato .mp4 solamente y se espera que debe pesar como máximo 200mb.

Además para mantener la integridad de la información se espera que se le haga un backup al sistema diario por las noches, cuando el sistema no está en uso.

1.14. Políticas y Reglamentos

Las políticas de la organización, o cualquier regulación o restricción relacionado con prácticas normales del negocio externa que puedan afectar la operación o rendimiento del sistema; serán definidas en futuras iteraciones del desarrollo del proyecto.

Respecto a la salud y seguridad de usuario, no hay criterios a tomar en cuenta. Esto debido a la naturaleza propia del sistema. Tampoco se tiene que manejar equipo que pueda perjudicar la salud humana, ni ningún método de operación. También es amigable con el ambiente, ya que no produce ningún residuo tóxico, ni radiación electromagnética.

1.15. Logística del Ciclo de Vida del Sistema

Para asegurar la calidad del sistema, dentro de la logística del ciclo de vida del mismo, se puede acceder a las sección de las actividades a Aseguramiento de calidad de este mismo documento. Se tienen las siguientes secciones que se podrían acceder:

- Actividades relacionadas con el Planeamiento de proceso de ACS

- Actividades relacionadas para asegurar el Producto
- Actividades relacionadas para asegurar el Proceso

No se menciona ninguna actividad relacionada con proveer instalaciones fijas para el soporte operacional, pero sí se considera algún tipo de entrenamiento de logística o especial; para los ingenieros encargados de darla mantenimiento al sistema. Esto porque se considera que las herramientas que utiliza el sistema (opencv) se consideran poco comunes, y se necesita que la curva de aprendizaje sea la más baja posible. De esta manera se tiene un mejor proceso de mantenimiento y se tiene un proceso de mayor calidad.

También además de las capacitaciones se piensa suministrar una gran cantidad de documentación: documento de métricas, procesos de instalación del sistema, actividades para asegurar la calidad de software, documentos de estándares usados, diagramas de componentes, documentación de trozos de código. No se espera contar con abastecimiento ni suministro de partes físicas ya que el sistema no es hardware.

Nota: La cosideración de añadir más documentación necesaria se puede considerar en las siguientes iteraciones.

1.16. Empaquetado, manipulación, envío y transporte

Debido a que el sistema es software, no se tiene que tomar en cuenta ningún aspecto de empaquetado, manipulación, envío y transporte. El sistema estará disponible en un repositorio Github, el cuál el sistema podrá descargar e instalar posteriormente.

1.17. Verificación

Las descripción de la verificación de las métricas y requerimientos con detalle descritos de este documento, está en un documento previo a este: de métricas de calidad. En dicho documento se pueden acceder las siguientes secciones (organizado por familia de métricas, según el ISO-9126):

- Funcionalidad: Sección 1, páginas 1-2.
- Confiabilidad: Sección 2, páginas 2-3.
- Usabilidad: Sección 3, páginas 3-4.
- Eficiencia: Sección 4, páginas 4-5.
- Mantenibilidad: Sección 5, páginas 5-6.
- Portabilidad: sección 6, páginas 6-7.

1.18. Supuestos y Dependencias

Se tienen las siguientes supuestos y dependencias de los requerimientos del sistema:

- Los jugadores para ser detectados por su color, no podrán tener uniforme verde, debido a que se podría confundir con el campo.
- Los jugadores de los equipos para ser detectados por su color, no podrán tener uniformes del mismo color, ya que no se podrían distinguir los jugadores unos con otros.
- El arbitro deberá de tener una camisa distinta a los jugadores, ya que este se podría confundir con un jugador.
- Los jugadores para ser detectados, deberán estar **completamente** rodeados por el pixeles verdes. Si alguna extremidad, esta tocando la gradería del estadio, se asumirá como parte de la gradería. Lo mismo pasará al detectar el cambo, tales zonas, serán tomadas como parte de la gradería.
- Se necesitan las herramientas como pre-requisitos: opencv 3.1, JDK 0.8 o superior, Tomcat 8.0.

2. Estándares de Codificación

2.1. Javascript

Para Javascript se consultaron tres estándares de código, el de Google, W3Schools y ECMAScript. Ambos comparten varias características y difieren un poco en otras, sin embargo al final se decidió utilizar ECMAScript. Se escogió una modificación de este estándar debió a la herramienta JSLint, la cual utiliza este estandar pero le hace algunas modificaciones, las cuales se adaptan mas a como generalmente el equipo de trabajo utiliza. La herramienta es online, por lo que no es necesario hacer ningún tipo de instalación y permite poner directivas en medio de comentarios para modificar un poco la evaluación que hace, lo cual brinda un poco de configuración comparado a otras herramientas.

2.2. AngularJS

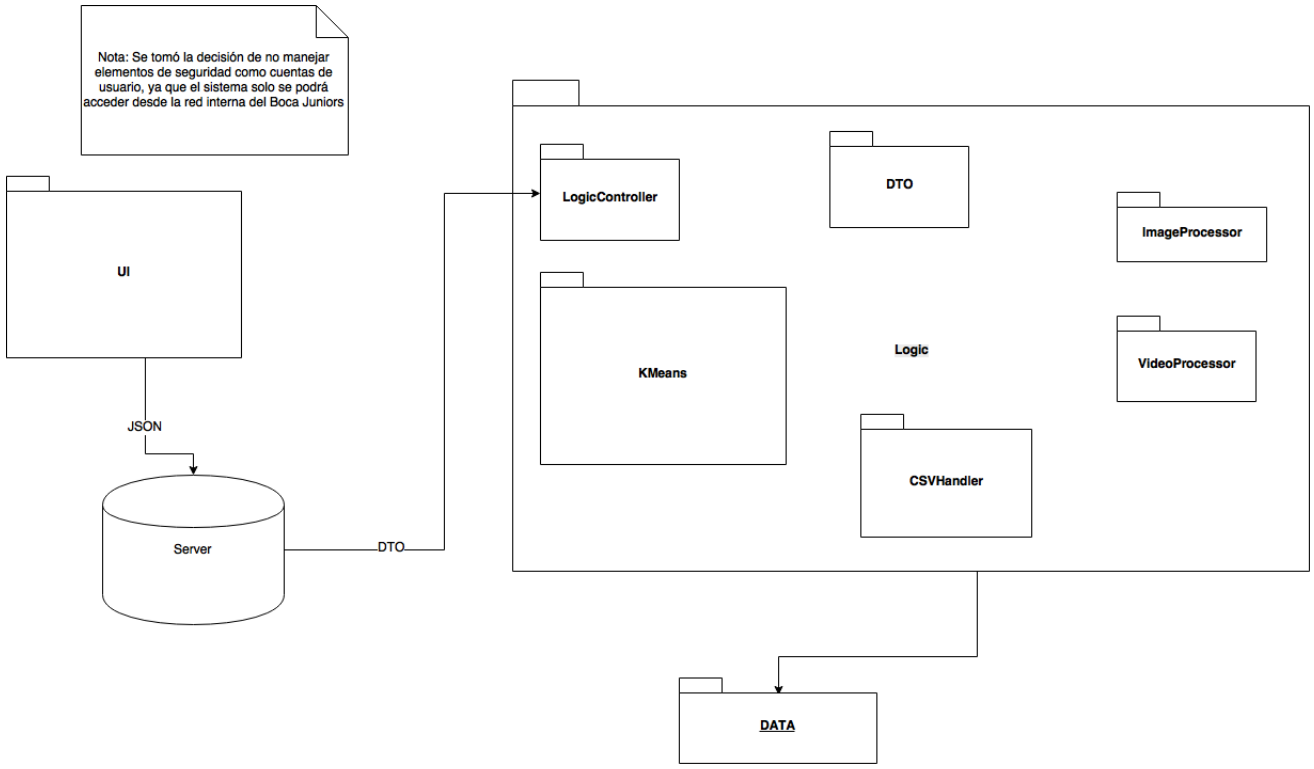
Debido a que se planea utilizar el framework de AngularJS, este incluye muchas modificaciones y estructuras al código. Por lo tanto se decidió establecer un estándar para todo lo referido a este framework. Se estudio el uso del de Google, y comparado con el de CodeStyle, es mas robusto y abarca mas detalles, sin embargo CodeStyle acopla los factores mas generales del de Google y agrega un archivo de configuración para una herramienta, lo cual ayuda a la automatización al verificar el cumplimiento del estandar. Se utilizara esta herramienta mencionada llamada JSHint. Para un proyecto en donde la parte robusta sea el uso de angular se recomendaría el uso del estándar de google.

2.3. Java

Para Java se planea utilizar el estándar de Google. Hay dos factores por los cuales se escoge dicho estándar, primero es bastante especifico respecto a todo, no deja brechas para malas interpretaciones. Segundo, este estándar ya se había utilizado por todos los miembros del proyecto, por lo cual ya todos están familiarizados con este y fue bastante aceptado por todos. Se planea utilizar el plugin de eclipse CheckStyle, el cual sigue un poco este estandar por defecto, pero tiene las ventajas de que es posible hacerle modificaciones y se puede utilizar nativamente en el IDE de Eclipse que el equipo de desarrollo utilizara.

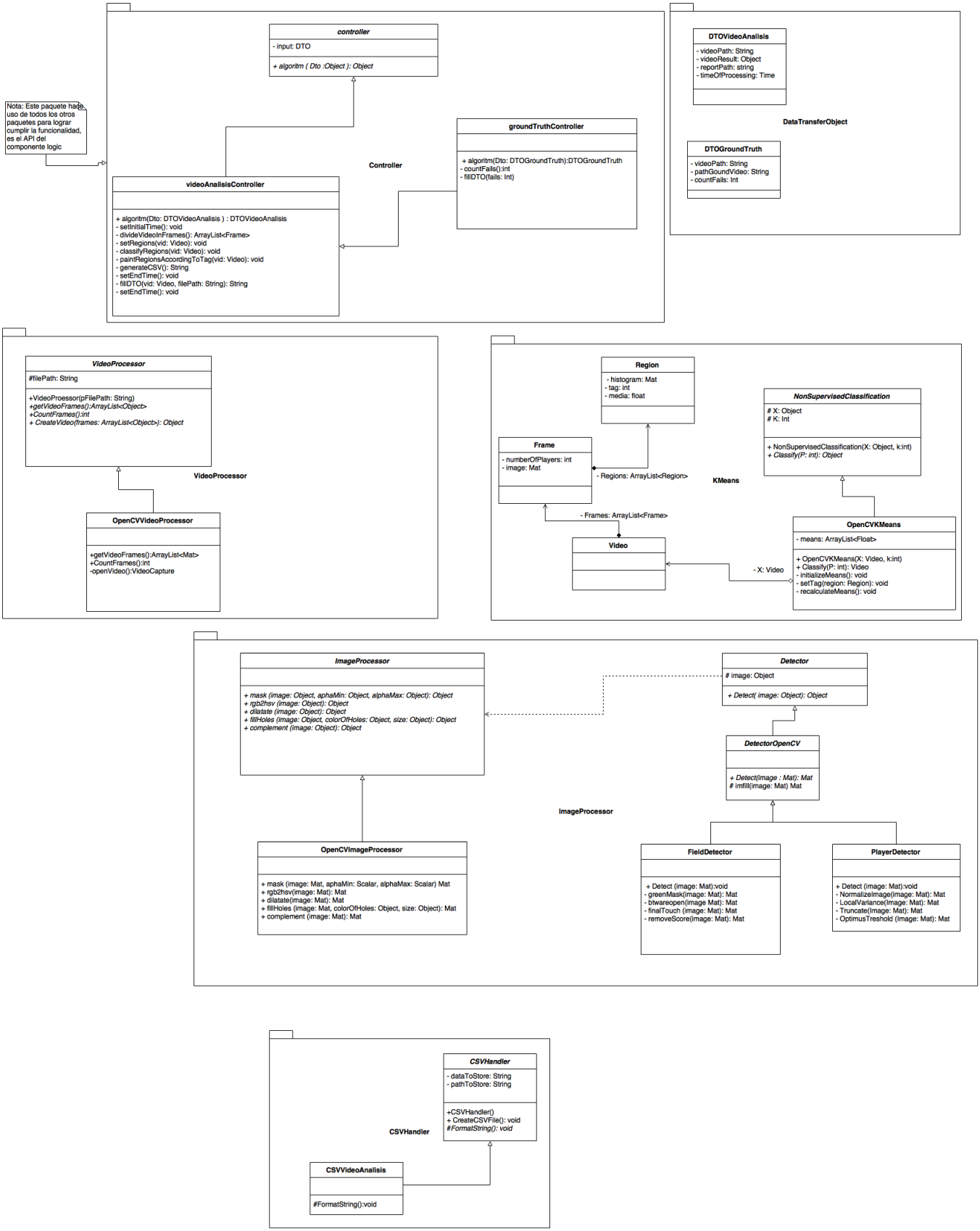
3. Arquitectura del sistema

3.1. Diagrama de componenetes



El diagrama se puede consultar en línea para poder verlo más a detalle.

3.2. Diagrama de clases



El diagrama se puede consultar en línea para poder verlo más a detalle.

3.3. Patrones de diseño

Se han usado dos patrones de diseño principalmente en la arquitectura del sistema. El primero es llamado .Estrategia", que consiste en un mismo algoritmo pueda ser implementado de maneras diferentes. Esto lo podemos observar en la clase DetectorOpenCV (Dentro del paquete de ImageProcessor), existe el algoritmo para detectar imágenes, pero este se comporta distinto dependiendo de la clase que usemos para llamarlo, si se usa fieldDetector, el algoritmo detectará el campo, pero si se usa la clase PlayerDetector, el algoritmo se comportará distinto y detectará las regiones posibles a ser jugadores.
Otro patrón de diseño utilizado Data Transfer Object, que consiste en un objeto de comunicación entre componentes, en este caso este objeto facilitará el paso de parámetros a la hora de utilizara alguno de las funcionalidades implementadas en este módulo.
Además a raíz del diagrama de componentes podemos ver que se espera utilizar el patrón de diseño fachada, esto consiste en brindar una interfaz de sólo lo útil para los clientes de un módulo, por lo cual todas las funcionalidades a las que un cliente de este módulo puede acceder, se encuentran encapsuladas en las clases del paquete controller.

4. Actividades de ACS a realizar al finalizar cada Sprint

Las siguientes actividades se tienen que realizar al finalizar cada sprint. Estas estarán basadas en el estándar IEEE-730-2002. Este estándar define tres grupos de actividades: en el planeamiento del proceso de ACS, aseguramiento del producto y aseguramiento del proceso.

4.1. Actividades de Planeamiento de proceso de ACS

Actividad	Justificación	Responsable	¿Para cuándo?
Evaluar y asegurar la objetividad del equipo de ACS	Es de suma importancia entregar un sistema de calidad a la Boca Juniors; y la mejor manera para lograrlo es tener un equipo enfocado solamente en la ACS; y no solo esto, sino que también debe ser independencia técnica, administrativa y económica. También de esta manera se divide de manera más eficiente el trabajo y el proyecto progresa más rápido	Josué Arrieta	Siempre: al finalizar cada Sprint

4.2. Actividades de Aseguramiento del Producto

Actividad	Justificación	Responsable	¿Para cuándo?
Evaluar la conformidad del diseño respecto a los requerimientos	Se escoge debido que para garantizar que el producto siga las especificaciones y expectativas del Boca Juniors; en primer lugar, se debe garantizar que lo siga en la base y estructura del sistema: el diseño. Además como se está utilizando una metodología Scrum, se tiene planeado la revisión luego de cada Sprint ya que se planea diseñar siempre	Joshua Mata	Siempre: al finalizar cada Sprint
Evaluar la conformidad de la implementación respecto a los requerimientos	Se escoge para garantizar, estilo caja negra, que el código cumple con las necesidades descritas por la Boca Juniors; y se entregue un sistema de calidad. Además como se está utilizando una metodología Scrum, se tiene planeado la revisión luego de cada Sprint ya que se planea programar siempre	Joshua Mata	Siempre: al finalizar cada Sprint
Evaluar la conformidad de la implementación respecto al diseño	Se escoge para asegurar doblemente las últimas dos actividades descrita; y asegurarse que el producto entregado cumpla las expectativas de la Boca. SportAnalytics está decidido en entregar justamente lo que Boca Juniors necesita y está pidiendo. Además como se está utilizando una metodología Scrum, se tiene planeado la revisión luego de cada Sprint ya que se planea programar siempre	Joshua Mata	Siempre: al finalizar cada Sprint
Evaluar la conformidad de los test unitarios, de integración y de aceptación	Además de asegurarse de que el sistema siga los requerimientos, es importante asegurarse que los sigan correctamente, y la manera para garantizarlo es con esta actividad descrita. Es importante asegurarse que el sistema funciona y cumpla las expectativas del Boca Juniors. Al utilizar una metodología Scrum es importante que se realice al final de cada Sprint y de esta manera se reducen defectos en el producto de software final	Joshua Mata	Siempre: al finalizar cada Sprint

Nota: se considera que estas cuatro actividades garantizarán un producto de calidad ya que aseguran que el producto sigan los requerimientos especificados por la boca Juniors tanto a nivel de diseño como a nivel de código y diseño, y que además estén implementadas correctamente.

4.3. Actividades de Aseguramiento del Proceso

Actividad	Justificación	Responsable	¿Para cuándo?
Evaluar la conformidad de los ambientes de desarrollo y de prueba respecto a los planes del proyecto	Se escoge porque es de suma importancia revisar constantemente las herramientas y programas a ser usados durante el desarrollo y testo del proyecto. Esto para garantizar que los empleados de SportAnalytics se sientan cómodos y además para asegurarse de que las herramientas o programas escogidos son los correctos para realizar el proyecto.	Joshua Mata	Siempre: al finalizar cada Sprint
Evaluar la conformidad y cumplimiento de los estándares de codificación	Se escoge debido a que es de suma importancia asegurar de qué el estándar escogido sea el más factible para SportAnalytics; además de que si sí lo es, sea usado. Esto para garantizar transparencia en todo el equipo y que el mismo este familiarizado con el mismo	Adrián López	Siempre: al finalizar cada Sprint

Referencias

[1] Gamma E. and Helm R. and Johnson R. and Vlissides J. *Design Patterns*, 1994.

[2] Larman, Craig. *UML y Patrones*. 2da edición. Prentice Hall. 2003.

[3] El código latex de este documento se puede consultar en el siguiente link