



ESCUELA DE INGENIERÍA ELÉCTRICA Y ELECTRÓNICA

InVi Fusion – ETAPA DE DISEÑO

Tabla de contenido

1	InVi Fusion – Aplicación de Calibración y Reporte en procesos de inspección termográfica ...	1
1.1	DIAGRAMAS DE INTEGRACIÓN.....	1

1 INVI FUSION – APLICACIÓN DE CALIBRACIÓN Y REPORTE EN PROCESOS DE INSPECCIÓN TERMOGRÁFICA

Esta es una aplicación permite automatizar el proceso de calibración intrínseca y extrínseca para un arreglo de una cámara de espectro visible (EV) y una cámara de espectro infrarrojo (IR). El arreglo de las cámaras debe ser claramente especificado, de tal forma que el usuario pueda implementarlo y obtener los resultados deseados a partir de su propio montaje en el cual las características de las cámaras pueden variar siempre y cuando se tenga una cámara IR y una EV.

El objetivo es generar una imagen, resultado de fusionar las características más importantes de cada espectro, para obtener imágenes donde se puedan apreciar los límites y bordes que ofrece el espectro visible y las diferencias de temperatura que ofrece un termograma.

A partir de la imagen (o imágenes) allí obtenidas debe ser posible generar un reporte en un formato de documento que permita que el software pueda ser utilizado en un proceso de inspección termográfica real.

Este documento expone la etapa de diseño de InVi Fusion siguiendo la metodología RUP de diseño de software. En este documento se presentan:

- Diagramas de secuencia
- Diagramas de clase

1.1 DIAGRAMAS DE INTEGRACIÓN

Los diagramas de secuencia que se muestran a continuación hacen parte de la aplicación de InVi Fusion de calibración y reporte en inspección termográfica. En ella el usuario tiene a su disposición gran variedad de funciones que cumplen el objetivo de calibrar un arreglo de cámaras de espectro visible e infrarrojo para encontrar la matriz de transformación que permita fusionar las imágenes provenientes de ambas cámaras. Las funciones de esta aplicación se han organizado en tres grupos que comprenden: Interfaz Gráfica de Usuario, Calibración, Análisis de Imágenes y Reporte.

1.1.1 Interfaz Gráfica de Usuario

En el grupo de Interfaz Gráfica de Usuario se concentran las funciones que permiten la entrada y salida de archivos a la aplicación, además de las funciones para modificar configuraciones, ver manual y versión, y el registro que se lleva de los comandos ejecutados. Durante la ejecución de la

aplicación todas estas funciones están disponibles en las tres pestañas que agrupan el resto de las funciones. Las áreas que permanecen visibles durante todo momento en la aplicación se ven en la Figura 1.1, y son: los espacios de trabajo, el registro de comandos (Log), la barra de herramientas y las barras de menú y pestañas.

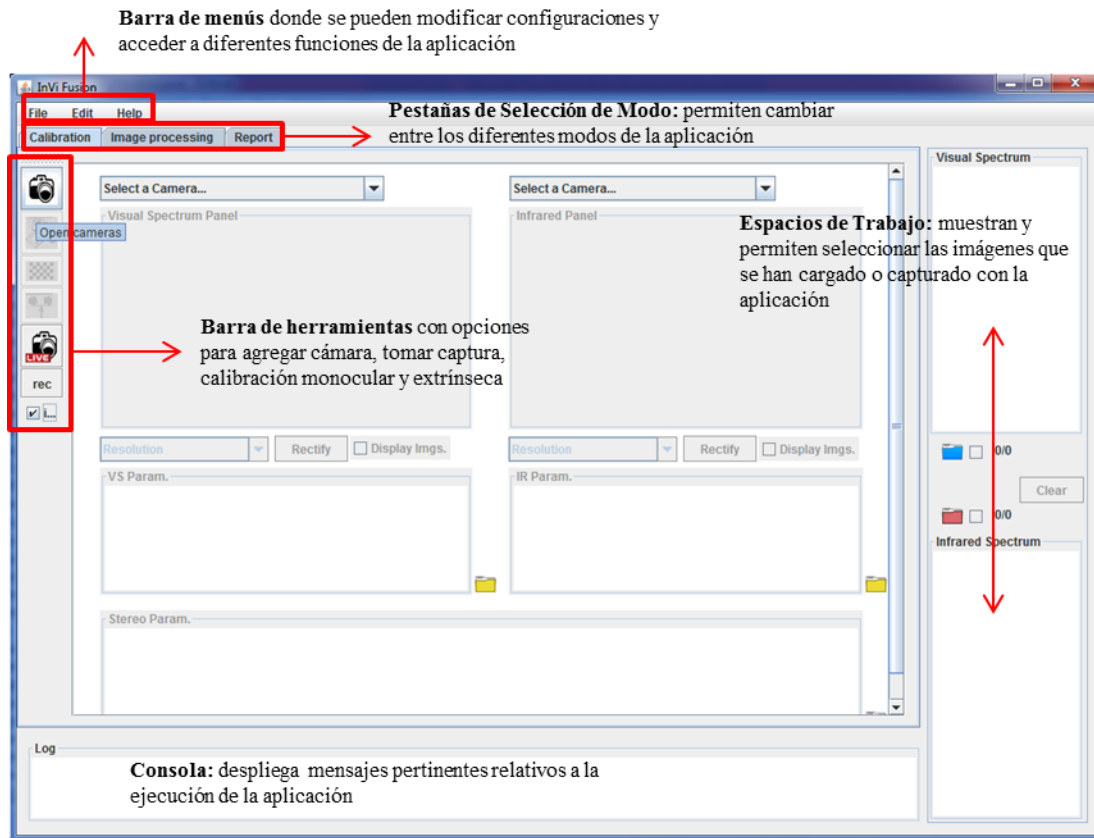


Figura 1.1 Interfaz Gráfica de Usuario de InVi Fusion

En la Figura 1.2 se expone el diagrama de secuencia para cargar imágenes a la interfaz. A través de una intervención del usuario se ejecuta el método `loadImage()` que crea una ventana de selección de archivos. En esta ventana el usuario puede seleccionar una o más imágenes, que son adquiridas por la interfaz por medio de la creación de un arreglo de objetos tipo `File`. Por último, la clase `imageIO` ejecuta el método `showImagePanel()` que se encarga de crear y mostrar el objeto gráfico que representa la imagen cargada a la interfaz. Este objeto muestra una miniatura de la imagen seleccionada por el usuario.

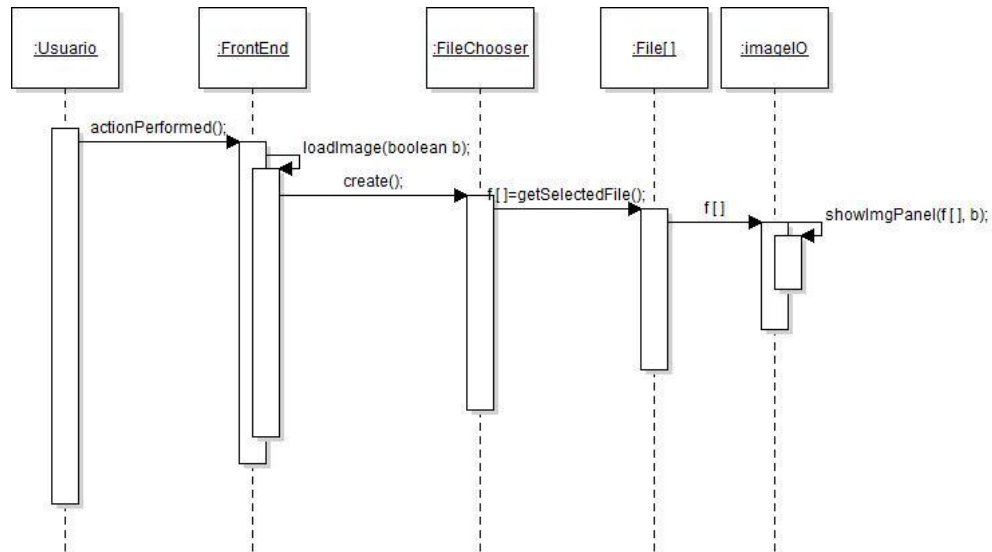


Figura 1.2 Diagrama de Secuencia para cargar Imágenes a la Interfaz

En la Figura 1.3 se presenta el diagrama de secuencia para cargar parámetros. El usuario ejecuta el método *loadParam()* que crea una ventana de selección de archivos del sistema. A través de esta ventana el usuario elige el archivo de parámetros que desee cargar. Al presionar Aceptar, la aplicación captura la información del archivo y la utiliza para cargar los datos al sistema, además de mostrarlos en su respectiva área de texto de la pestaña de calibración.

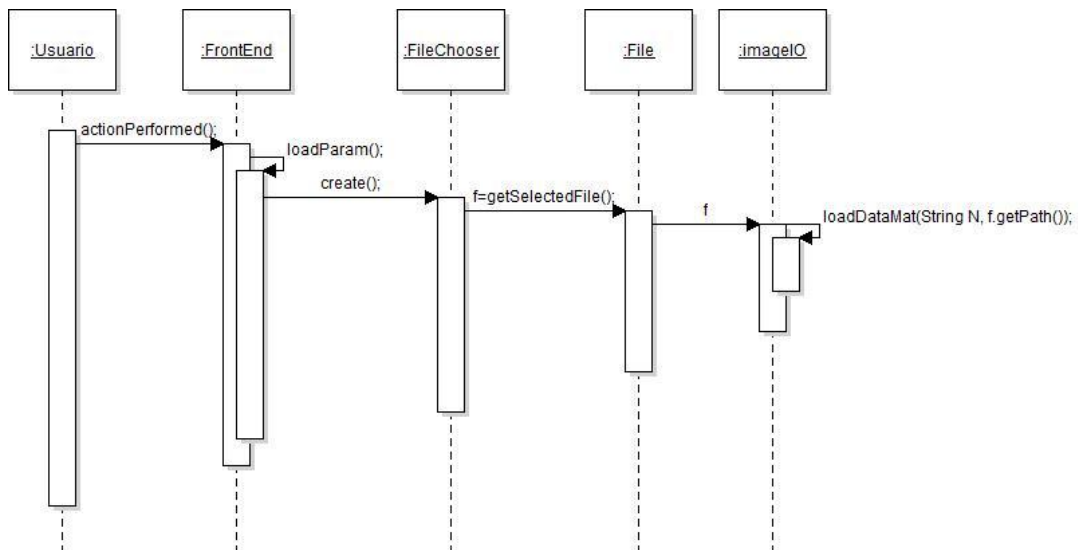


Figura 1.3 Diagrama de Secuencia para cargar Parámetros

En la Figura 1.4 se muestra el diagrama de secuencia para guardar los archivos del espacio de trabajo. Al ejecutar el método *saveWorkspace()*, el usuario crea una ventana de selección de archivos que le permite seleccionar la carpeta donde serán guardados los archivos. Simultáneamente se crea un objeto de la clase *Path* que usa las clases *DirectionStream* y *Files* para copiar todos los archivos que han sido cargados o generados por la aplicación en la carpeta destino seleccionada por el usuario.

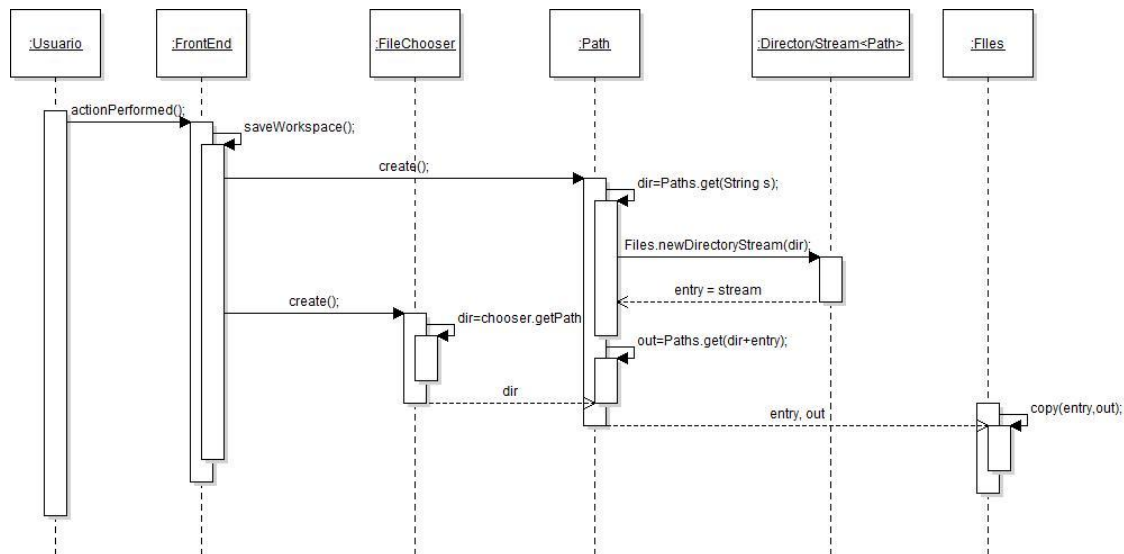


Figura 1.4 Diagrama de Secuencia para guardar los Archivos del Espacio de Trabajo

En la Figura 1.5 se puede ver el diagrama de secuencia que explica el proceso de limpiar el espacio de trabajo. El usuario llama al método *clear()* que se encarga de revisar las miniaturas seleccionadas en el espacio de trabajo. Aquellas que estén seleccionadas son removidas del espacio de trabajo y de la carpeta donde se almacenan temporalmente, durante la ejecución del programa.

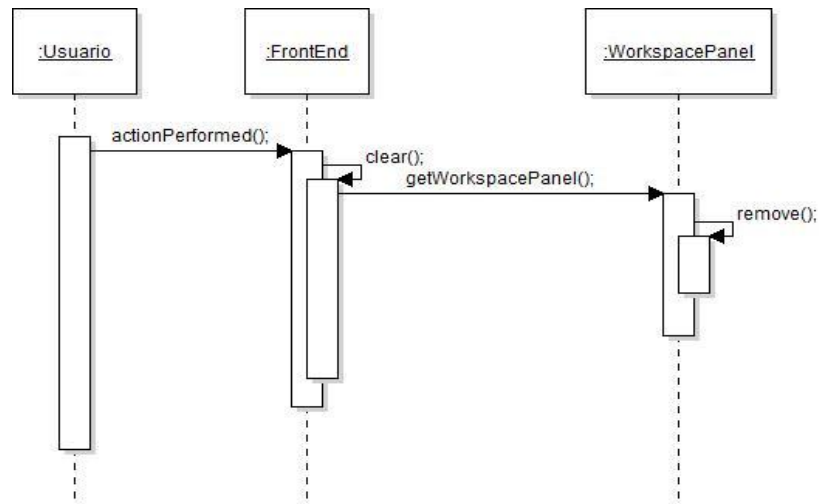


Figura 1.5 Diagrama de Secuencia para limpiar el Espacio de Trabajo

En la Figura 1.6 se expone el diagrama de secuencia para abrir las configuraciones modificables por el usuario. A partir de una acción del usuario se crea un objeto de la clase *appConfiguration*, el cual es una ventana donde se pueden cambiar los valores a diferentes parámetros y activar o desactivar la visualización de variedad de resultados.

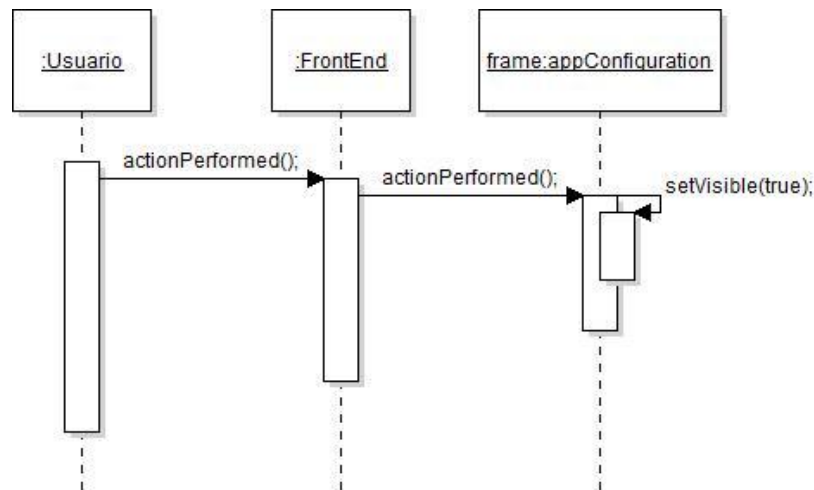


Figura 1.6 Diagrama de Secuencia para abrir las Configuraciones

En la Figura 1.7 se puede ver el diagrama de secuencia para ver la versión de la aplicación. El usuario ejecuta una acción que llama a un objeto de la clase *Version*. Este objeto se muestra al usuario como una ventana con toda la información de la versión de la aplicación e información de contacto de los desarrolladores.

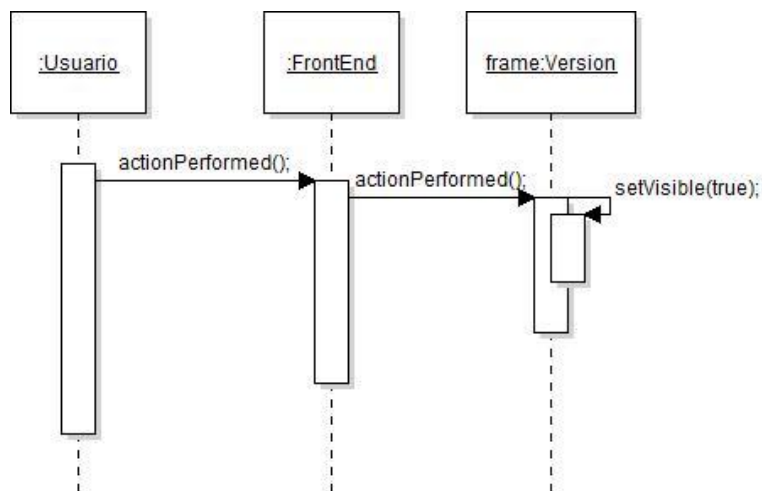


Figura 1.7 Diagrama de Secuencia para abrir la Versión de la Aplicación

En la Figura 1.8 se expone el diagrama de secuencia para abrir el manual de la aplicación. A partir de una acción del usuario se crea una instancia de la clase *PDFViewer* que abre el archivo “*Manual.pdf*”.

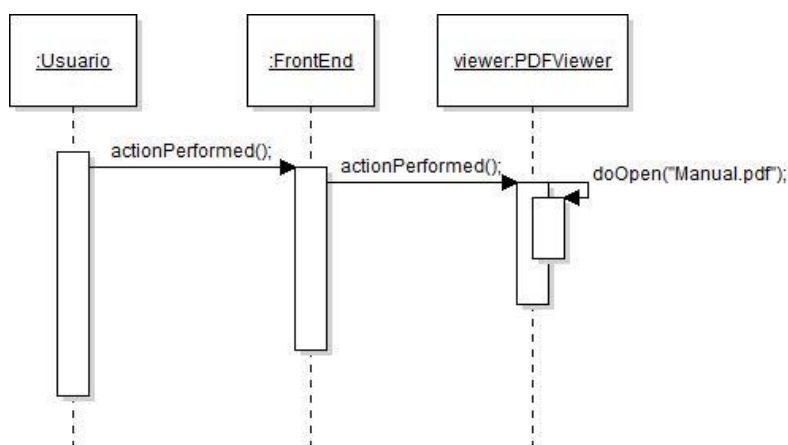


Figura 1.8 Diagrama de Secuencia para ver el Manual de la Aplicación

En la Figura 1.9 se expone el diagrama de secuencia que representa la ejecución del proceso de muestra del registro de eventos importantes en un área de texto. En el diagrama se ejemplifica una de las formas en las que la aplicación ejecuta dicho proceso. Inicialmente ante una acción del usuario, la aplicación carga su interfaz gráfica principal, creando los objetos para iniciar una calibración, además del panel donde se mostrará el registro de eventos. Al crearse el objeto vS de la clase *camCalibrator* se genera un mensaje del sistema que es capturado por el panel *log* y es mostrado consecuentemente al usuario en el área de texto descrita.

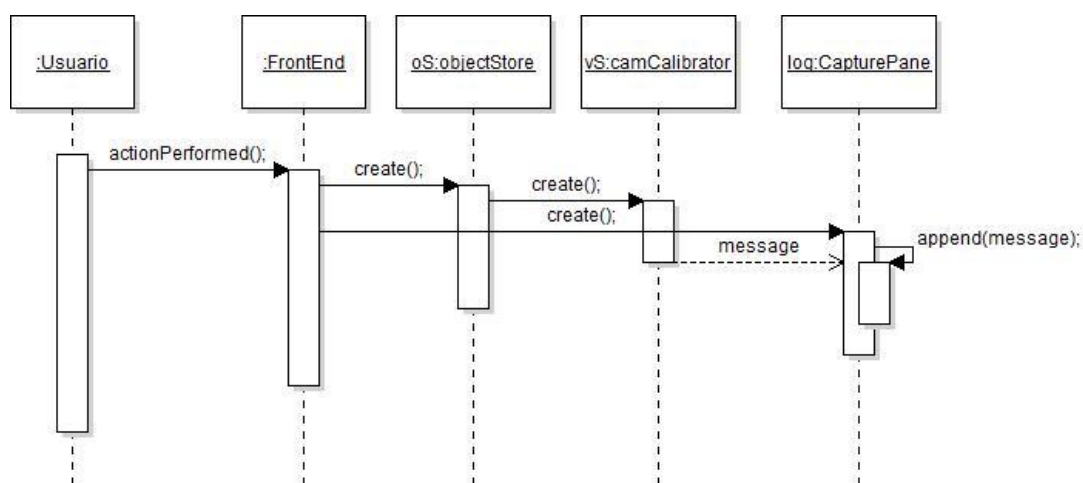


Figura 1.9 Diagrama de Secuencia para mostrar el Registro de Eventos de la Aplicación

1.1.2 Calibración

En la pestaña de calibración se ponen a disposición del usuario las funciones que permiten calibrar el arreglo de cámaras para el posterior análisis de imágenes. La Figura 1.10 muestra la interfaz gráfica de APL. En la barra de la izquierda se encuentran los botones que corresponden a las funciones de calibración más importantes. La ventana principal se ha dividido en dos partes correspondientes a los dos espectros que se desea fusionar, de forma que sea posible agregar cámaras, y se calcular parámetros correspondientes al espectro visible o infrarrojo. Por último en la parte inferior, aparece el panel de parámetros estéreo donde se verán dichos parámetros cuando sean

calculados. Y en la parte derecha con la etiqueta de *Visual Spectrum* o *Infrared Spectrum*, aparecerán las diferentes imágenes que se estén utilizando en la sesión iniciada por el usuario. Afuera de la pestaña activa, aparece el panel de consola que mostrará mensajes durante la ejecución de cualquier comando.

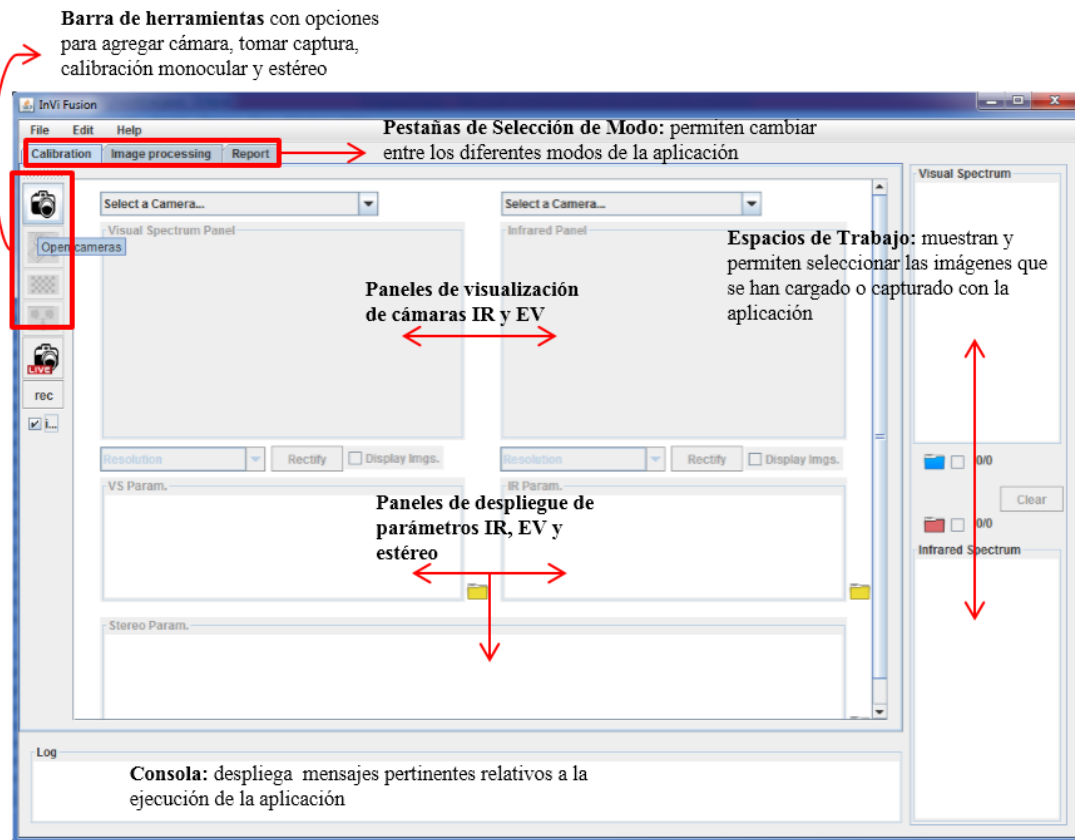


Figura 1.10 Funciones de Calibración de InVi Fusion.

En la Figura 1.11 se muestra el diagrama de secuencia para capturar una fotografía a partir de una entrada de vídeo a la aplicación. El usuario realiza una acción que activa la aplicación y le permite elegir de un listado, la cámara o entrada de vídeo que desea capturar. Posteriormente elige una resolución de un listado de resoluciones disponibles que depende de la cámara seleccionada. A continuación activa la(s) cámara(s) seleccionada(s). Por último, activa la opción capturar que obtiene la imagen a partir de la(s) entrada(s) de vídeo activa(s).

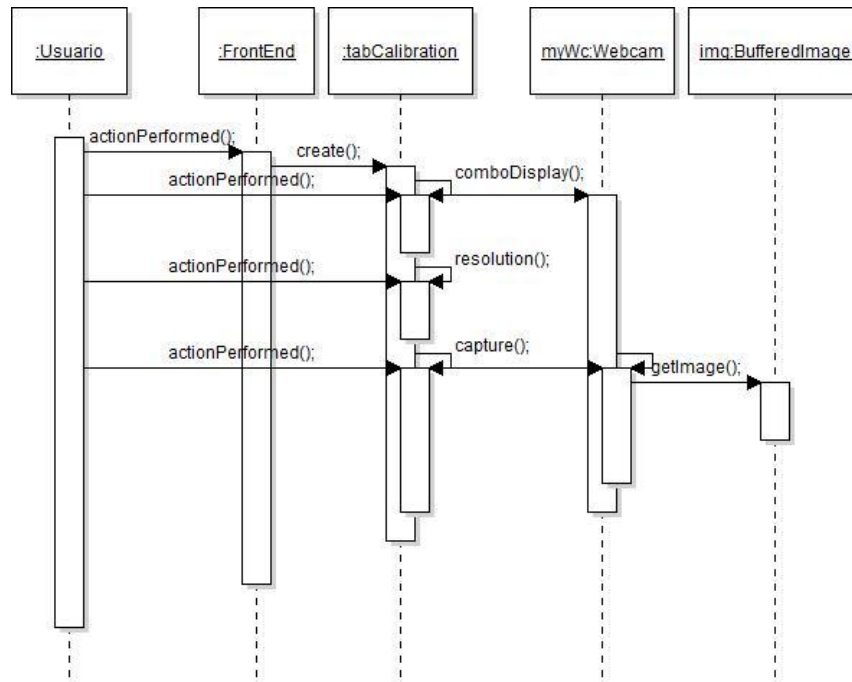


Figura 1.11 Diagrama de Secuencia para Capturar Imágenes desde Cámara

En la Figura 1.12 se muestra el Diagrama de Secuencia para realizar la calibración de una cámara. Para lograrlo el usuario activa la opción de calibración que lanza una ventana de solicitud de información (*Request Dialog*), en donde el usuario debe ingresar los valores de ancho y alto del patrón de calibración. Teniendo en cuenta esta información y la lista de imágenes seleccionadas, se crea un objeto de la clase *cameraCalibrator* y se ejecuta el método *calibrateMono()*. Los resultados son mostrados en el área de texto *monoPText*.

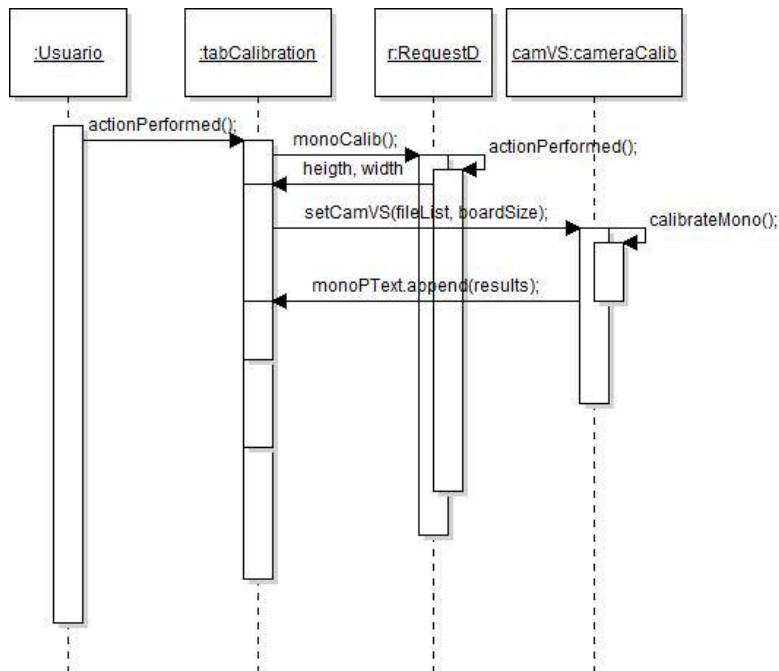


Figura 1.12 Diagrama de Secuencia para realizar una Calibración Monocular

En la Figura 1.13 se expone el Diagrama de secuencia para corregir las distorsiones de una imagen, a partir de los resultados de calibración de una cámara. Para realizar la corrección el usuario activa la función *rectify()*, y el programa obtiene el objeto *camVS* (para el caso de la cámara de espectro visible), donde se han almacenado los resultados de calibración. Posteriormente se ejecuta el método *undistortImages()* y se crean las imágenes utilizando el método *getUndistortedImages()*.

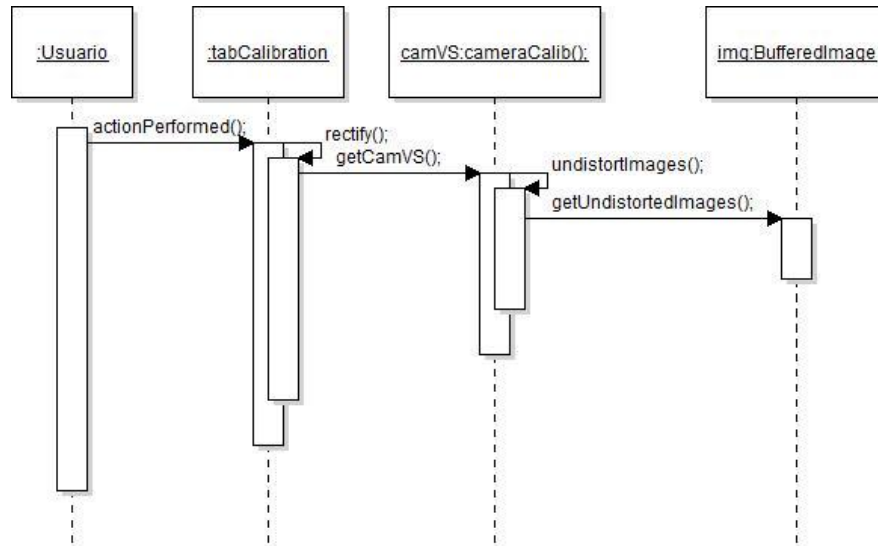


Figura 1.13 Diagrama de Secuencia para rectificar una Imagen

En la Figura 1.14 se muestra el Diagrama de secuencia para hallar la Homografía entre dos cámaras. Para realizar este proceso, el usuario ejecuta el método *stereoCalibration()*, que se encarga de establecer la información de las cámaras en una instancia de la clase *stereoCameraCalibrator*. Utilizando los resultados de la calibración individual de las cámaras, el método *calibrateStereo()* halla los valores de Homografía y el error de proyección. Por último, los resultados se muestran en el área de texto *stPText*, y se muestra el error de proyección en instancias de la clase *BufferedImage*.

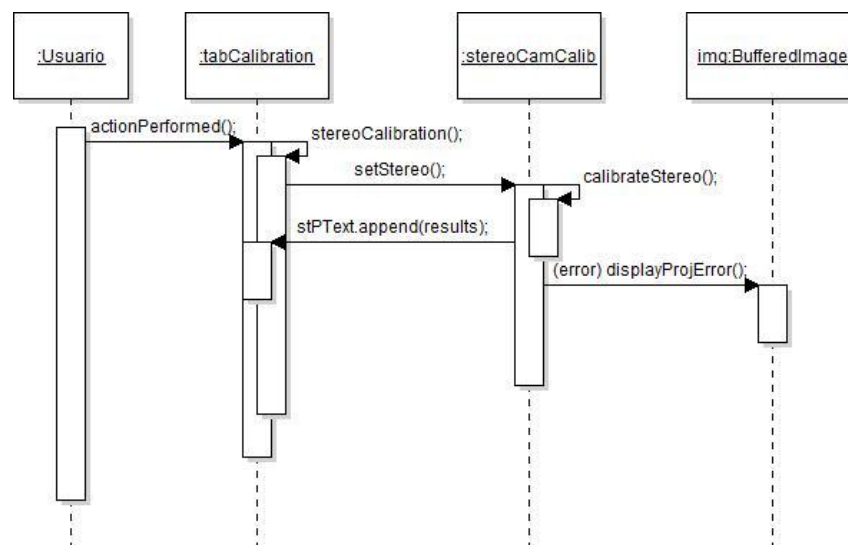


Figura 1.14 Diagrama de Secuencia para hallar la Homografía entre dos Cámaras

En la Figura 1.15 se muestra el Diagrama de Secuencia para hallar manualmente la Homografía entre dos cámaras. A partir de una acción del usuario, se llama una instancia de la clase *stereoCameraCalibrator* que se encarga de solicitar al usuario la realización de dos pasos

indispensables para la obtención de la matriz de Homografía. En cada paso el usuario debe señalar un conjunto de puntos en las imágenes. Una vez terminado el proceso, se muestra un mensaje.

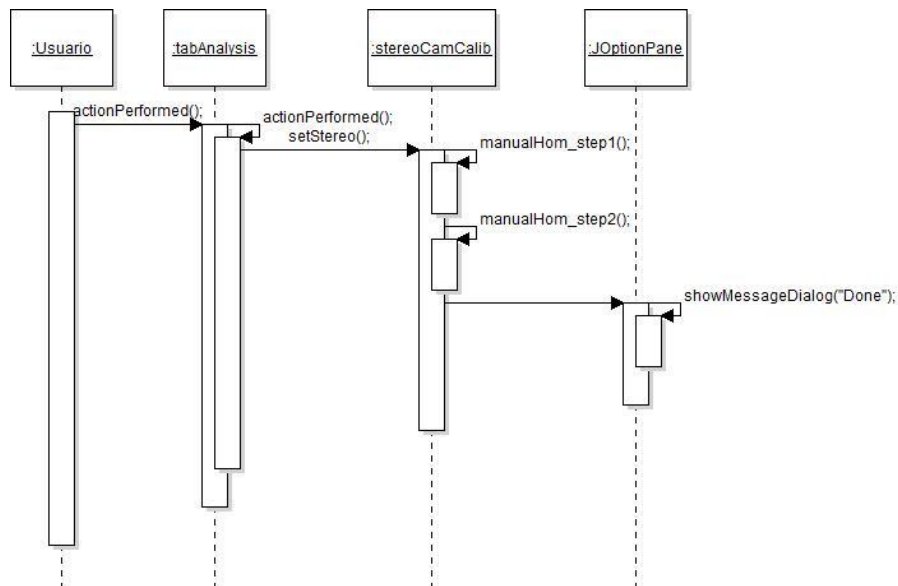


Figura 1.15 Diagrama de Secuencia para hallar manualmente la Homografía entre dos Cámaras

1.1.3 Pestaña de Análisis de Imágenes

La Figura 1.16 muestra la pestaña de Análisis de Imágenes donde se implementa la función principal de la aplicación, que es la fusión de imágenes. Luego de haber hallado una Homografía, es posible fusionar cualquier imagen que esté disponible en los espacios de trabajo de la aplicación.

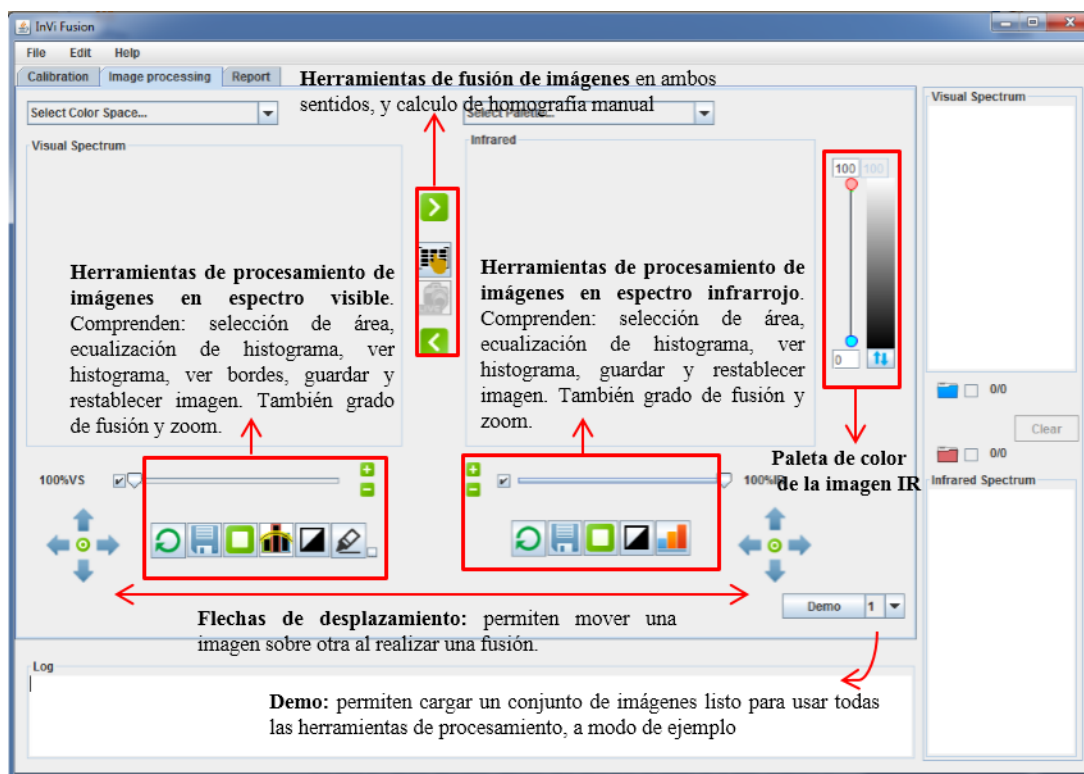


Figura 1.16 Funciones de Análisis de Imágenes InVi Fusion

Las funciones de procesamiento de imágenes permiten resaltar las características más importantes del espectro visible e infrarrojo, y seleccionar áreas específicas de ambos espectros, para ser transformadas. También existe la opción alternativa de hallar una homografía manual, o calibrar un par de imágenes sin necesidad de usar un patrón. Por último, se pone a disposición del usuario un conjunto de imágenes ya calibradas, que permiten usar todas las herramientas de la pestaña de análisis, lo que permite familiarizarse con las opciones de la aplicación.

En la Figura 1.17 se muestra el Diagrama de secuencia para mostrar imágenes en la pestaña de Análisis. El usuario hace doble clic sobre cualquiera de las miniaturas del área de trabajo. La pestaña de análisis establece la imagen en el área correspondiente al espectro visible o infrarrojo según sea el caso.

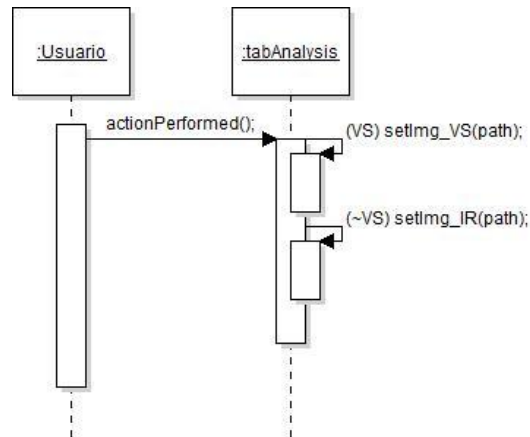


Figura 1.17 Diagrama de Secuencia para visualizar Imágenes en la pestaña de Análisis

En la Figura 1.18 se muestra el Diagrama de secuencia para realizar una fusión de imágenes a partir de los resultados de Homografía. El usuario selecciona la opción de fusión en la pestaña de Análisis, la clase *stereoCameraCalibrator* aplica la matriz de homografía a las imágenes activas utilizando el método *applyHomography(...)*. El resultado de la fusión según el sentido en que se haya realizado, de EV a IR o de IR a EV, se verá en el área de espectro IR o en el área de EV, respectivamente.

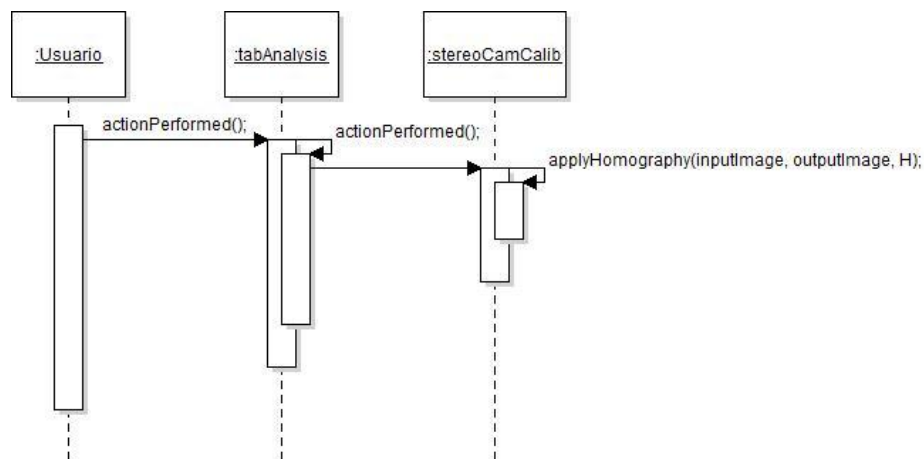


Figura 1.18 Diagrama de Secuencia para realizar una Fusión de Imágenes

La Figura 1.19 muestra el diagrama de secuencia para iniciar una visualización de la fusión de dos entradas de video. En el diagrama se muestra el curso de acciones que se siguen para generar un panel de fusión. Este panel pertenece a la clase *FusionPanel*, que internamente está configurada

para agregar las entradas de video seleccionadas por el usuario y superponerlas de acuerdo a las características de cada cámara, en este caso sus resoluciones. En caso de haber calculado una homografía, la superposición se hace teniendo en cuenta estos parámetros, y se corrigen los efectos de las diferentes perspectivas en el vídeo resultante.

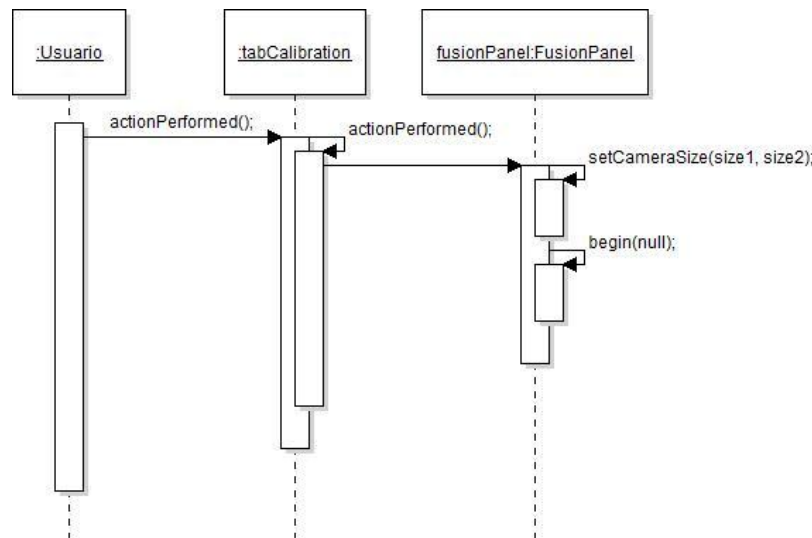


Figura 1.19 Diagrama de Secuencia para realizar una fusión en vivo.

La Figura 1.20 muestra el Diagrama de secuencia para variar el grado de fusión o transparencia de un espectro en una fusión de imágenes. En el diagrama se muestra el curso de acciones para el caso de EV, que es análogo al caso IR. Para la ejecución del proceso, el usuario realiza una acción sobre el objeto gráfico *slider_VS* que al cambiar su estado cambia el valor de la variable *alpha* y modifica la imagen utilizando el método *fusion(...)* de la clase *ImgProcessing*.

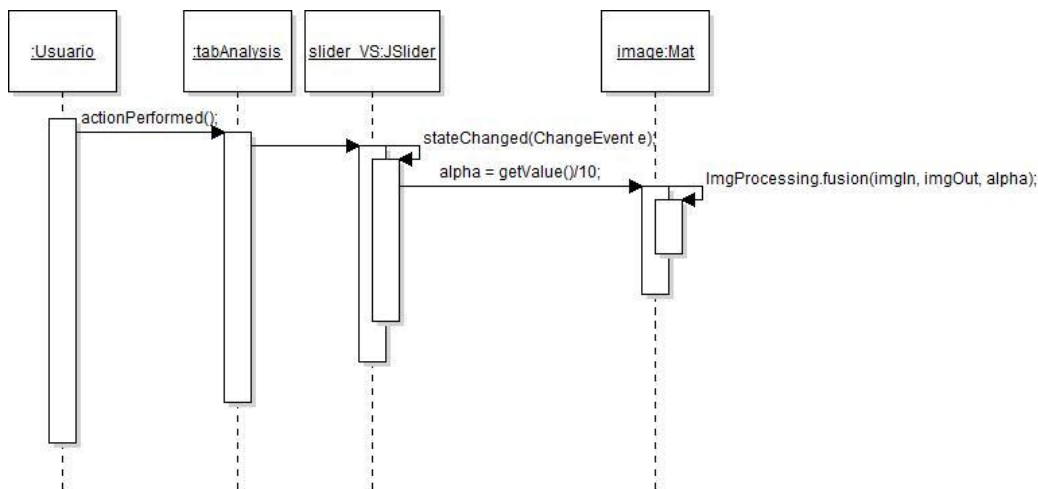


Figura 1.20 Diagrama de Secuencia para variar la Transparencia de un Espectro en una Fusión de Imágenes

La Figura 1.21 expone el Diagrama de secuencia para seleccionar un área en cualquiera de las imágenes de las áreas de imagen de la pestaña de Análisis. En el diagrama se muestra el curso de acciones para el caso de EV, que es análogo al caso IR. Para la ejecución del proceso, el usuario realiza una acción sobre el botón de área, lo que añade una instancia de la clase *MouseListener* al área de la imagen de EV. Al hacer clic sobre el área de la imagen EV, el usuario podrá ahora seleccionar un sector de la imagen, tan grande como desee. El tamaño y ubicación de este sector se

almacena en la variable *area*, que utilizará la clase *ImageProcessing* para dibujar un área seleccionada sobre la imagen de EV.

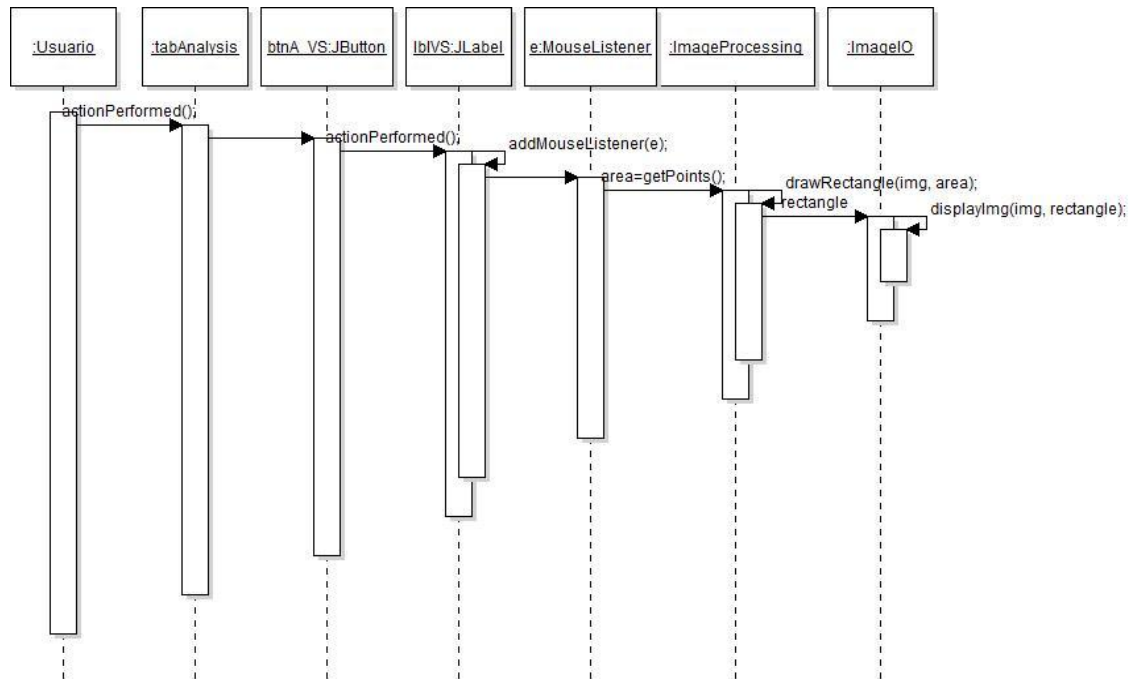


Figura 1.21 Diagrama de Secuencia para seleccionar un Área específica de una Imagen

La Figura 1.22 muestra el Diagrama de secuencia para extraer los bordes de una imagen de espectro visible. El usuario presiona el botón, y la imagen presente en el área de imagen de espectro visible es reemplazada por la imagen resultante de ejecutar el método *borderDetection(img)* de la clase *ImageProcessing*. La imagen de extracción de bordes se muestra en el área de la imagen de espectro visible utilizando el método *displayImg(img)*, de la clase *imageIO*.

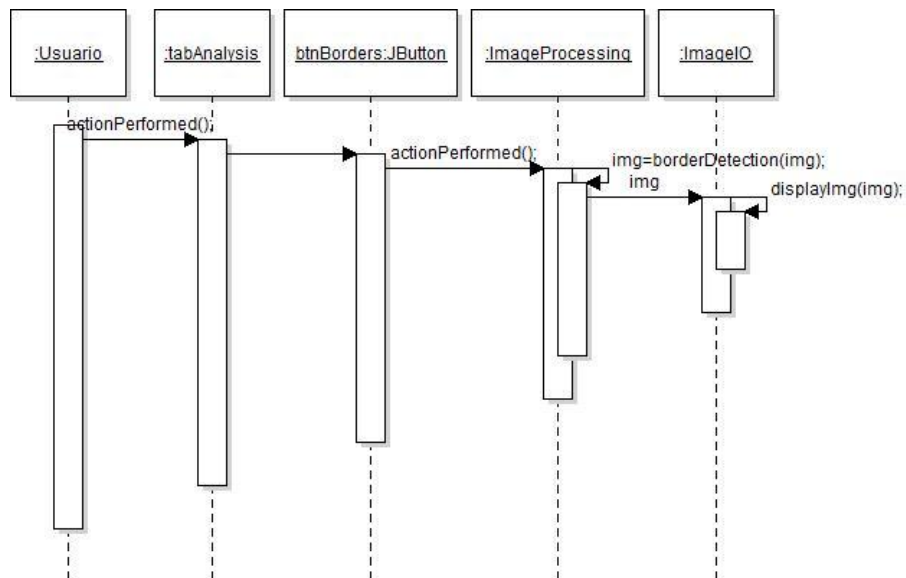


Figura 1.22 Diagrama de Secuencia para extraer los Bordes de una Imagen de Espectro Visible

La Figura 1.23 muestra el Diagrama de secuencia para mostrar la distribución de la densidad de píxeles en forma de Histograma para una imagen IR. Para iniciar el procesamiento, el usuario activa

un botón, y este ejecuta el método *drawHistogram(...)* de la clase *ImageProcessing*. El histograma se muestra en una pequeña ventana emergente.

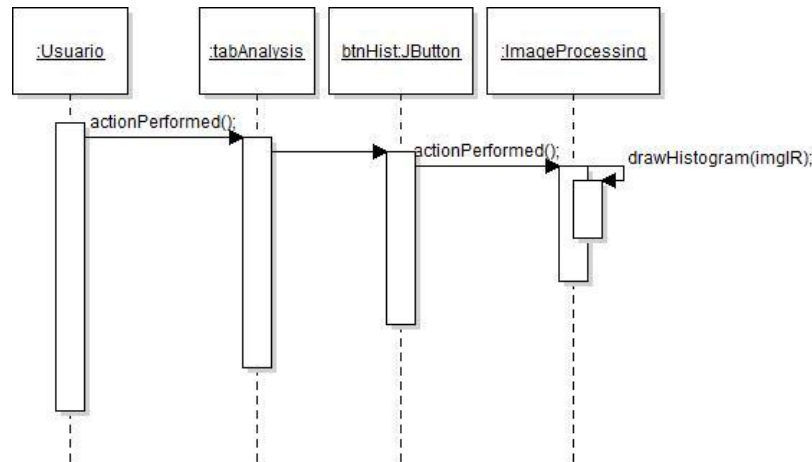


Figura 1.23 Diagrama de Secuencia para mostrar el Histograma de una Imagen de Espectro Infrarrojo

La Figura 1.24 muestra el Diagrama de secuencia para modificar una imagen de EV ecualizando los valores de su histograma. Para llevar a cabo este proceso, el usuario activa un botón que ejecuta el método *eqHistogram(...)* de la clase *ImageProcessing*. El método acepta como parámetro una imagen y retorna la imagen ecualizada, que se muestra en el área de imagen EV de la pestaña de análisis.

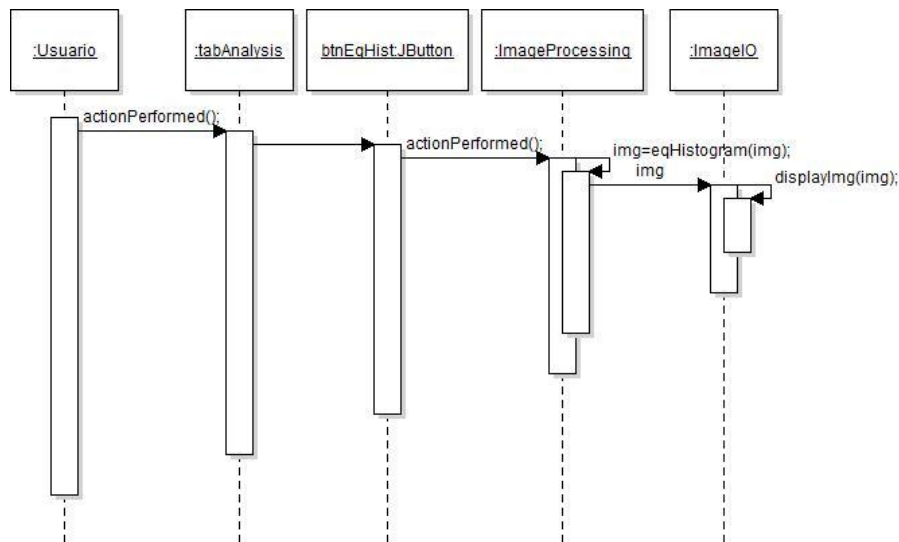


Figura 1.24 Diagrama de Secuencia para modificar una Imagen de Espectro Visible Ecualizando su Histograma

La Figura 1.25 muestra el Diagrama de secuencia para visualizar los píxeles que estén en un rango de intensidad definido por el usuario. Para activar la ejecución de este proceso, el usuario modifica la posición de alguno de los botones del objeto *rangeS*. Como resultado los valores de rango cambian y a partir de ellos se genera una máscara que es sustraída de la imagen IR, dando como resultado una imagen que muestra solo los píxeles que están dentro del rango de intensidad o color, definidos por el usuario.

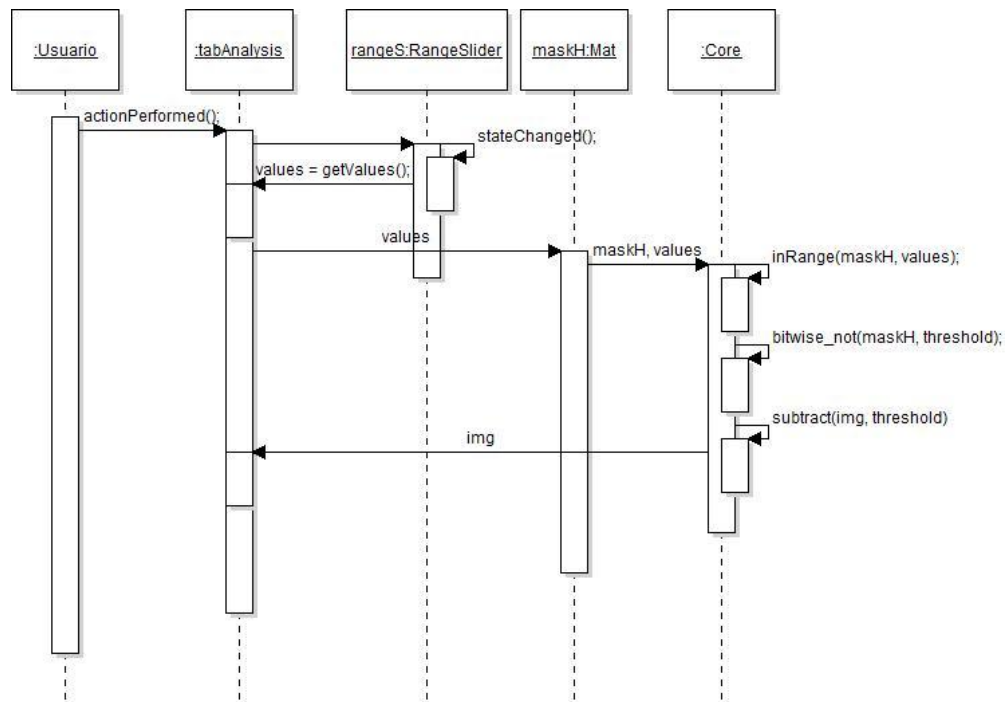


Figura 1.25 Diagrama de Secuencia para visualizar un Rango de Píxeles de cierta Intensidad

La Figura 1.26 expone el Diagrama de secuencia para desplazar la proyección (o resultado de fusión) de una imagen sobre otra. En el diagrama se muestra el caso en el que se presiona el botón de desplazamiento vertical superior, para los otros casos el diagrama de secuencia es análogo. Para ejecutar el proceso, el usuario presiona el botón de desplazamiento, esto genera un cambio de valor en la variable *shift*, la clase *ImageProcessing* ejecuta el método *shiftImage*, modificando la imagen EV o IR según sea el caso. La imagen desplazada se muestra utilizando el método *displayImage(...)*.

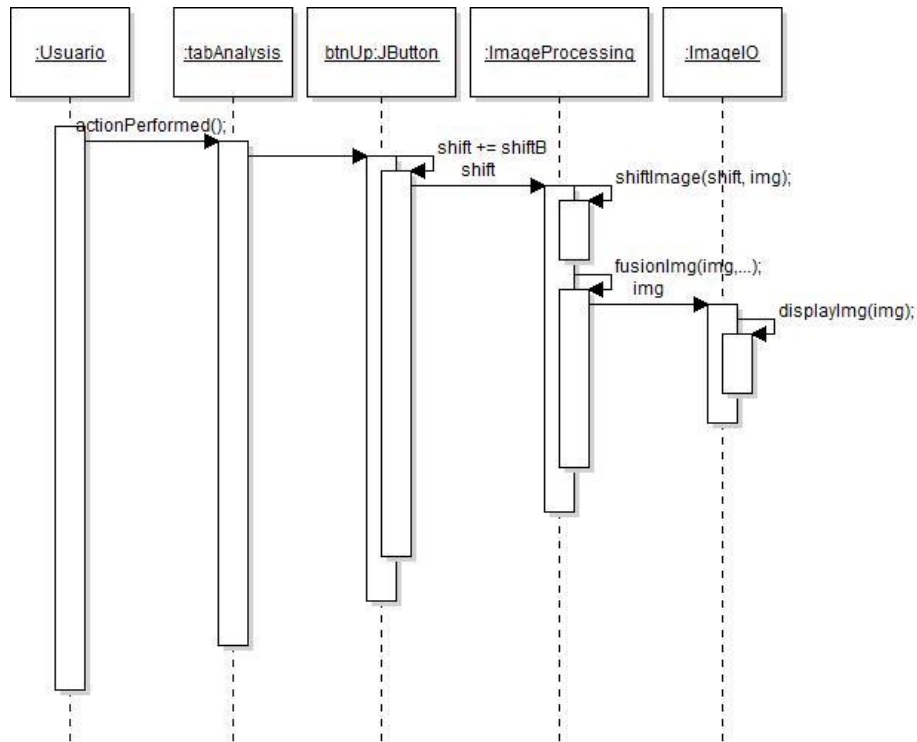


Figura 1.26 Diagrama de Secuencia para desplazar una Imagen sobre otra

La Figura 1.27 muestra el Diagrama de secuencia para restablecer las características originales de las imágenes. Para llevar a cabo este proceso, el usuario activa el botón *btnRst*, que toma la imagen que se muestra en el área de imagen IR o EV según el caso y la iguala a la imagen original que es guardada en el objeto *img_original*. Posteriormente se muestra usando el método *displayImage(...)*.

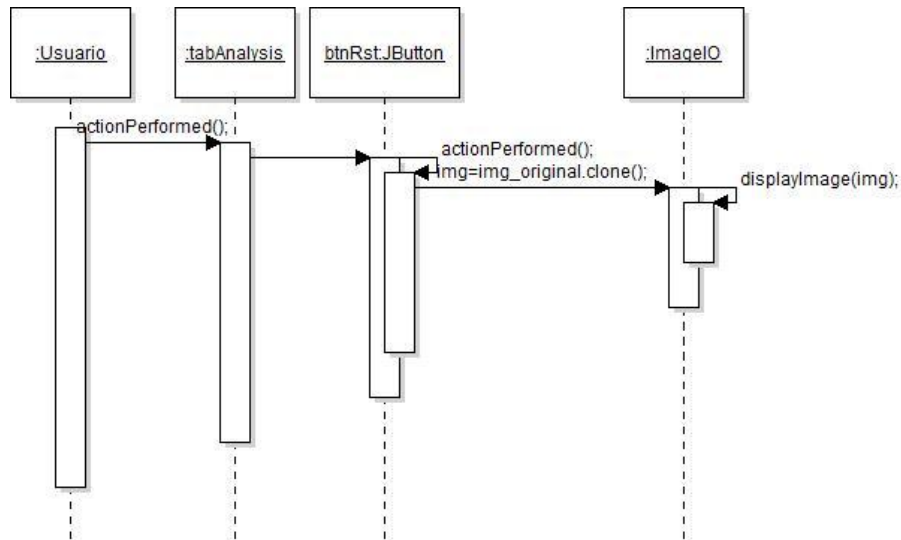


Figura 1.27 Diagrama de Secuencia para restablecer las Características de las Imágenes a su Condición inicial

La Figura 1.28 muestra el Diagrama de secuencia para ampliar o reducir el tamaño de las imágenes. El usuario presiona el botón de aumentar o reducir según desee, y la aplicación le asigna el valor de un múltiplo de 2 a la variable *zoomScale*. Usando el método *resize(...)* de la clase *Improc*, se

cambia el tamaño de la imagen y se muestra por medio de *displayImage(...)*.

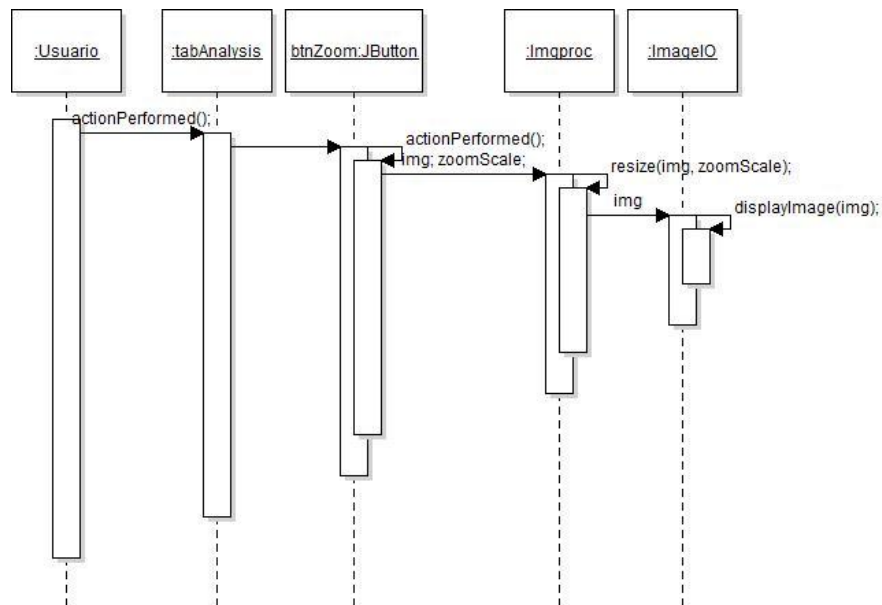


Figura 1.28 Diagrama de Secuencia para ampliar o reducir el Tamaño de las Imágenes

En la Figura 1.29 se muestra el Diagrama de secuencia para cambiar el color de una imagen de espectro visible. En este diagrama se muestra el caso en el que se selecciona escala de grises para aplicar a la imagen. El otro caso es análogo a este. Para iniciar la ejecución de este proceso, el usuario despliega la lista de opciones y selecciona el color que desee. Si el caso es el primero, la aplicación ejecuta el método *cvtColor(...)* de la clase *Imgproc*. Y le añade el parámetro *COLOR_BGR2GRAY*, que determina la conversión de color a escala de grises. El método retorna la imagen en escala de grises que se muestra nuevamente en la pestaña de Análisis.

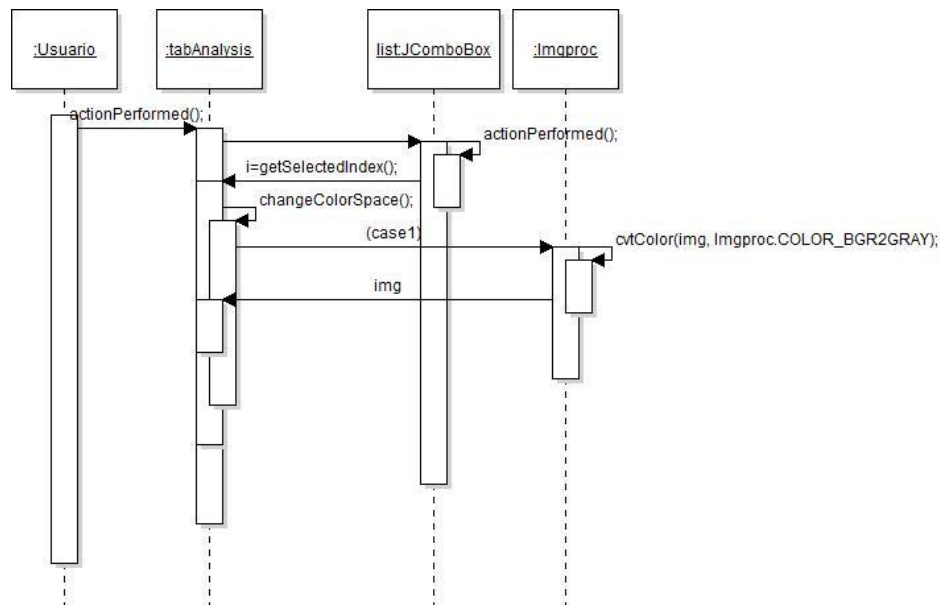


Figura 1.29 Diagrama de Secuencia para cambiar el Color de una Imagen de Espectro Visible

La Figura 1.30 muestra el Diagrama de secuencia para cambiar la paleta de color de una imagen IR. En este diagrama se muestra la ejecución para el caso específico de la selección de la paleta

Rainbow, los otros casos son análogos a este. Para llevar a cabo el proceso de cambio de paleta, el usuario selecciona un valor de la lista desplegable. La selección de la paleta acciona los métodos *processingIR()* y *changeColor()* que llaman al método *applyColorMap(...)* de la clase *Contrib*, con el valor *RAINBOW* como parámetro. Este método retorna la imagen modificada con la paleta de color seleccionada.

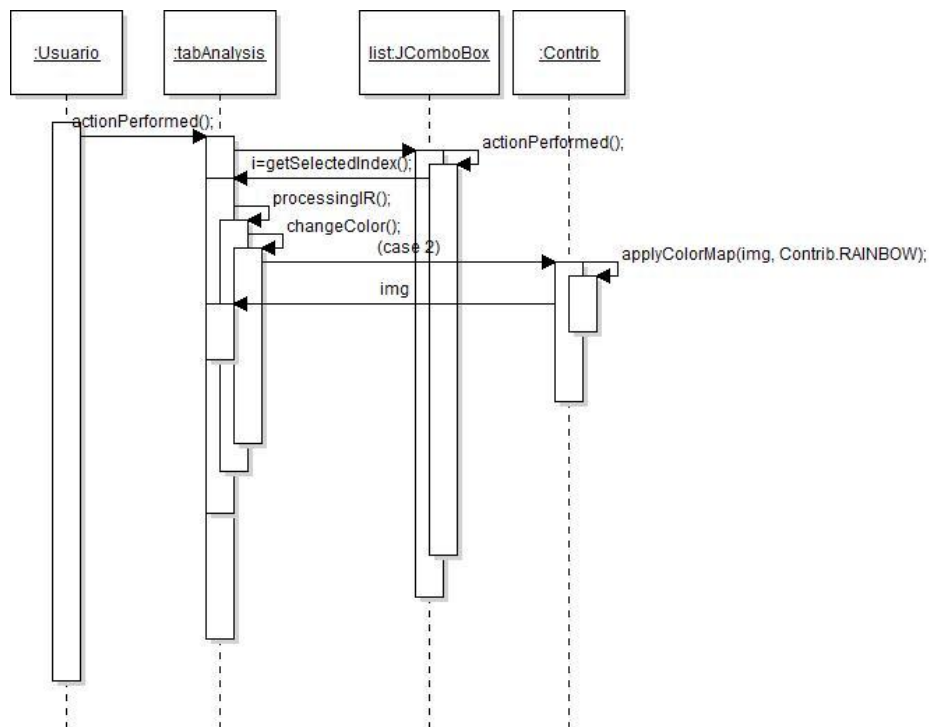


Figura 1.30 Diagrama de Secuencia para cambiar la Paleta de Pseudocolor de una Imagen de Espectro Infrarrojo

1.1.4 Pestaña de Reporte

La Figura 1.31 muestra la pestaña de reporte donde se permite al usuario modificar campos de información preestablecidos para generar un reporte de inspección termográfica. En la parte izquierda se encuentran los formularios que el usuario podrá modificar según desee. En la parte derecha, se encuentran los paneles de imágenes que permiten definir si se añadirá una o dos imágenes en cada tabla que se vaya a agregar al reporte. Los botones bajo el área de selección de imágenes permiten modificar la información de unos campos preestablecidos para cada tabla, además de ver las imágenes que se agregarán al informe, o eliminarlas. El botón “*Select File...*” permite seleccionar un logo para que sea adicionado al reporte. Por último, los botones “*Preview*” y “*Generate Report*” generan un reporte en formato pdf para que sea visualizado por el usuario antes de ser guardado, o para guardarlo en una ubicación elegida por éste, respectivamente.



Figura 1.31 Funciones de Generación de Reporte InVi Fusion

La Figura 1.32 muestra el Diagrama de secuencia para la edición de los campos de la pestaña de Reporte. Para ejecutar este proceso el usuario escribe la información pertinente en cada campo, y al realizar una acción como generar un reporte o una vista previa, la aplicación extrae la información de los campos y los guarda en variables y arreglos de variables para su posterior, como por ejemplo *companyData*.

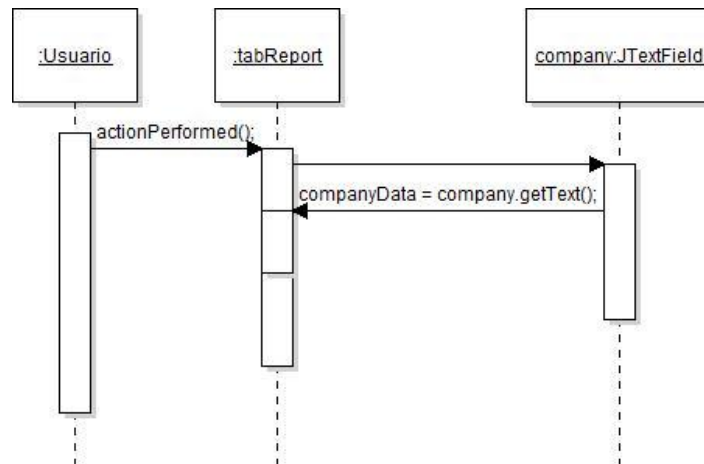


Figura 1.32 Diagrama de Secuencia para editar los Campos de la Pestaña de Reporte

La Figura 1.33 muestra el Diagrama de secuencia para añadir imágenes desde los espacios de trabajo a la pestaña de Reporte, permitiendo seleccionar las imágenes que se desean incluir en un reporte.

Para llevar a cabo este proceso, el usuario hace doble clic sobre una imagen del espacio de trabajo, y esta es añadida al objeto *table_1*, en forma de miniatura. Para añadir imágenes al objeto *table_2*, el usuario debe seleccionar al menos dos paneles que se muestren en *table_1*, y presionar el botón correspondiente. Lo que removerá los paneles de *table_1* y los añadirá a *panel_2*. Estos procesos se llevan a cabo para permitirle al usuario seleccionar la forma en que quiere que aparezcan las imágenes en el reporte. El panel *table_1* genera una tabla por cada imagen agregada, y el panel *table_2*, una tabla por cada dos imágenes.

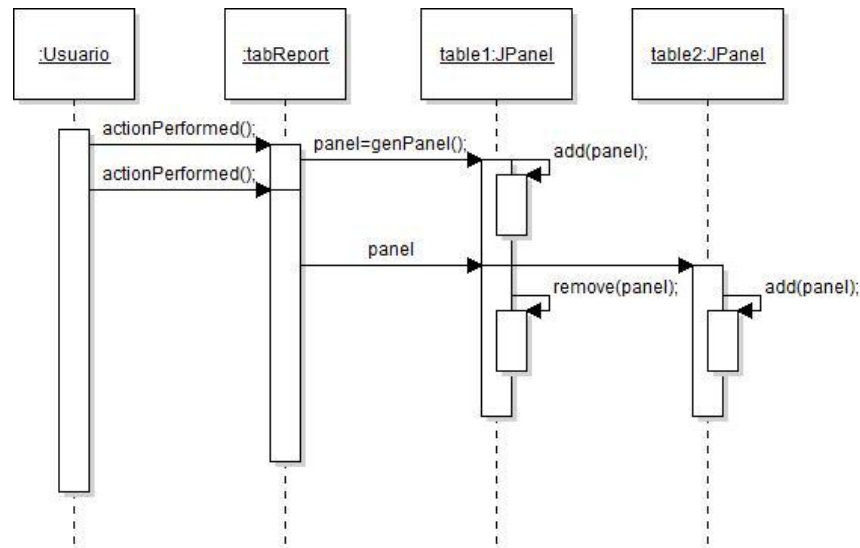


Figura 1.33 Diagrama de Secuencia para añadir Imágenes desde el Espacio de Trabajo a la Pestaña de Reporte

La Figura 1.34 expone el Diagrama de secuencia para modificar los datos de las tablas de imágenes del reporte. Para ejecutar este proceso, el usuario presiona el botón que activa el método *imgDataTableGen()*. Este método genera un *frame* en donde se añaden un número de pestañas que corresponden a las tablas de imágenes que se vayan generar. Una vez modificados los campos de cada pestaña, el usuario presiona un botón que permite extraer la información de los campos y guardarla en un arreglo de *Strings*.

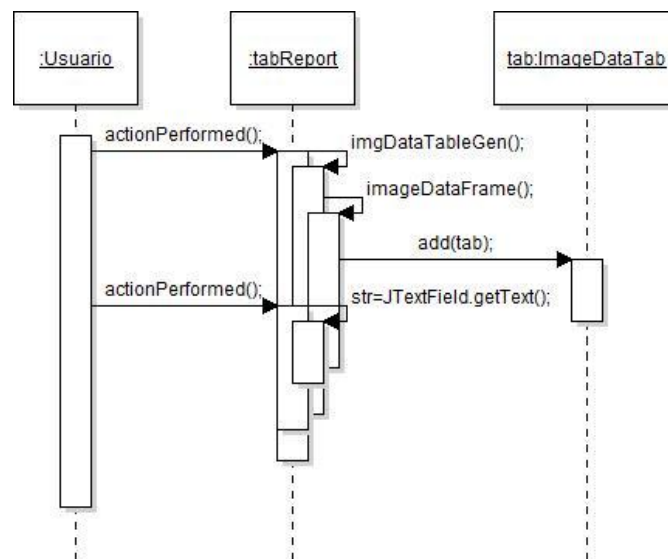


Figura 1.34 Diagrama de Secuencia para modificar la Información de las Imágenes del Reporte

La Figura 1.35 muestra el Diagrama de secuencia para agregar un logotipo al reporte. Para realizar este proceso, el usuario presiona un botón que crea una instancia de la clase *FileChooser*. Esta ventana permite seleccionar un archivo de tipo imagen cuya dirección es almacenada en un objeto de la clase *File*. La imagen es finalmente mostrada en un área de la pestaña de Reporte usando el método *read(File)* de la clase *ImageIO*.

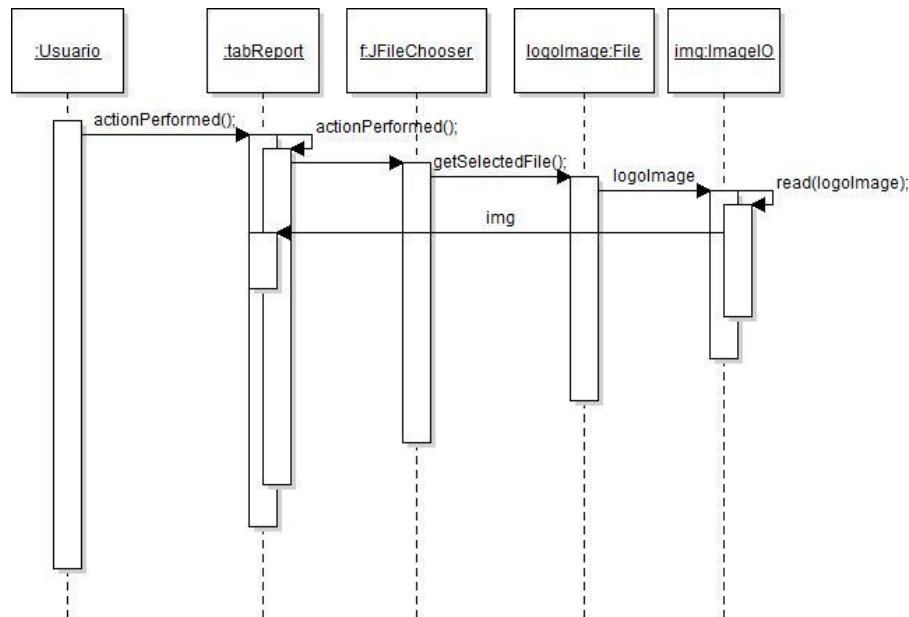


Figura 1.35 Diagrama de Secuencia para agregar un Logotipo al Reporte

La Figura 1.36 muestra el Diagrama de secuencia para realizar una vista previa del reporte en un visor de pdf de la aplicación. Para realizar este proceso, el usuario selecciona el botón *Preview*, lo que abre una instancia de la clase *Document*. Este objeto se modifica a partir de los datos agregados por el usuario, utilizando los métodos de la clase *Report*. Posteriormente, la clase *PDFWriter* genera un archivo *pdf* y lo guarda a una ubicación temporal. Este archivo es utilizado por una instancia de la clase *PDFViewer*, para ser mostrado al usuario en un visor de *pdf*.

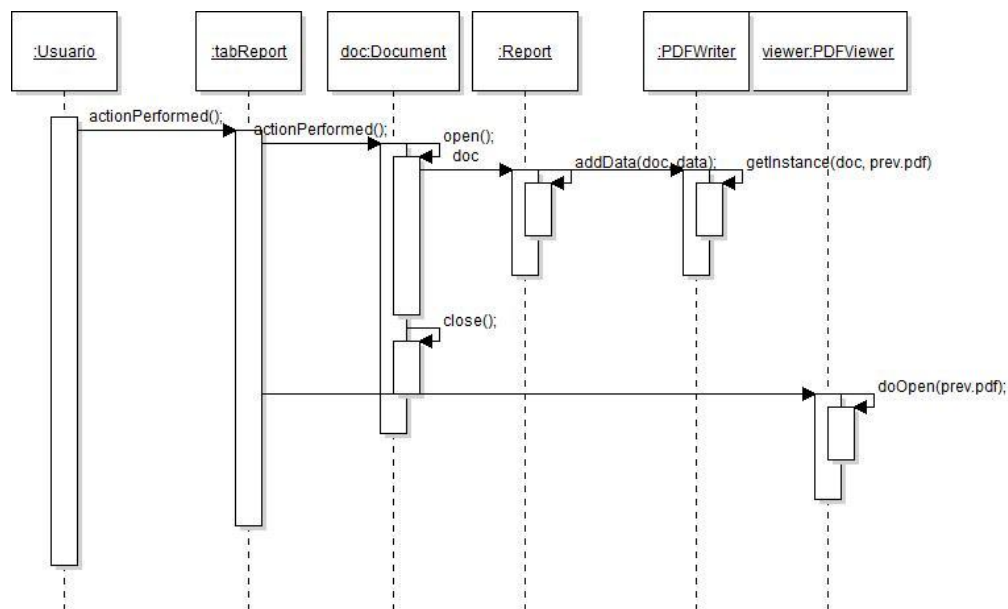


Figura 1.36 Diagrama de Secuencia para realizar una Vista previa de un Reporte

La Figura 1.37 muestra el diagrama de secuencia para generar un reporte y guardarlo en una ubicación de disco. La ejecución de esta acción es muy parecida a la explicada para el anterior diagrama de secuencia y comprende los mismos pasos, añadiendo la creación de una ventana de la clase *FileChooser*, que permite elegir al usuario una ubicación y un nombre para el reporte en formato *pdf* a ser creado.

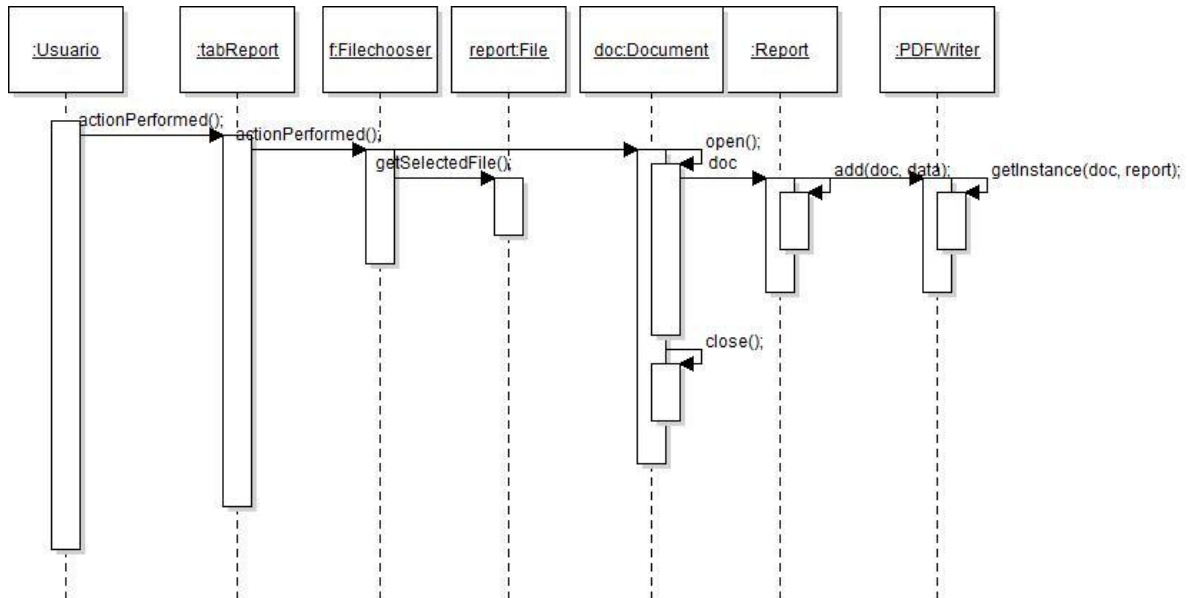


Figura 1.37 Diagrama de Secuencia para Guardar un Reporte