

Sistemas de Computación

TRABAJO FINAL

ANALIZADOR LÓGICO

2008

Jost, Mauricio Gastón

Matrícula: 200404067

Ingeniería en Computación

Veneranda, Guillermo Federico

Matrícula: 200204446-3

Ingeniería en Computación

Índice	2
Objetivo	3
1. Marco Teórico	4
2. Requerimientos para el proyecto	5
3. Actores	6
Operador	6
Observador	6
4. Diagramas de Casos de Uso	7
Configurar el modo de muestreo	8
Configurar la freq. de muestreo	9
Iniciar el muestreo.....	10
Manipular representación.....	11
Aplicar zoom a la representación	12
Desplazar representación	13
5. Diagrama de Capas	14
6. Diagramas de Clases	16
6.1. Diagrama General de Clases.....	16
6.2. Capa de Representación y Lógica (manipulación de vistas).....	18
6.3. Capa de Representación y Lógica (manipulación del módulo externo)	19
6.4. Capa de Lógica	19
6.5. Capa de Lógica y de Mensajes.....	20
7. Diagramas de Secuencia	21
7.1. Cambio de color de un canal	21
7.2. Desplazamiento de la vista	22
7.3. Inicio de muestreo	23

Objetivo

El objetivo de este proyecto es construir un analizador lógico de 8 canales para PC. Se ha de basar en requisitos propuestos, y ha de contener la correspondiente documentación del desarrollo del mismo, así como la implementación, tanto en hardware como en software.

1. Marco Teórico

Un **analizador lógico** es un instrumento de laboratorio que sirve para obtener muestras de múltiples señales digitales, con una base de tiempo en común. Esto permite visualizar las comunicaciones entre dispositivos electrónicos (también llamado "hand-shaking") y determinar la causa de posibles fallos.

Habitualmente este dispositivo puede muestrear varias señales digitales, una por cada canal. Un canal no es más que una vía de análisis de una señal en particular. Físicamente se manifiesta con una punta similar a la de un osciloscopio. Esta debe ser conectada al conductor correspondiente a la señal a observar. Además, el analizador consta de una punta de referencia o masa, y de puntas de disparo. Estas últimas tienen como finalidad generar un muestreo únicamente ante cambio de estado de cierta señal. Esta suele ser del clock del sistema a estudiar. Otra forma de funcionamiento generalmente implementada en los analizadores consiste en utilizar un clock interno, que trabaja a una frecuencia indicada por el usuario. Así los muestreos son logrados con un período predefinido.

El analizador lógico es una herramienta muy útil en la electrónica digital, principalmente en la comunicación entre procesadores y otros dispositivos específicos, como son los displays LCD, memorias, sensores de distancia digitales, etc.

Desde los modelos más económicos hasta los más completos tienen precios entre 300 y 1.600 dólares, siendo estos modelos para PC. Modelos completamente portables cuestan entre 10.000 y 19.000 dólares (frecuencias del orden del GHz, de varias decenas de canales).

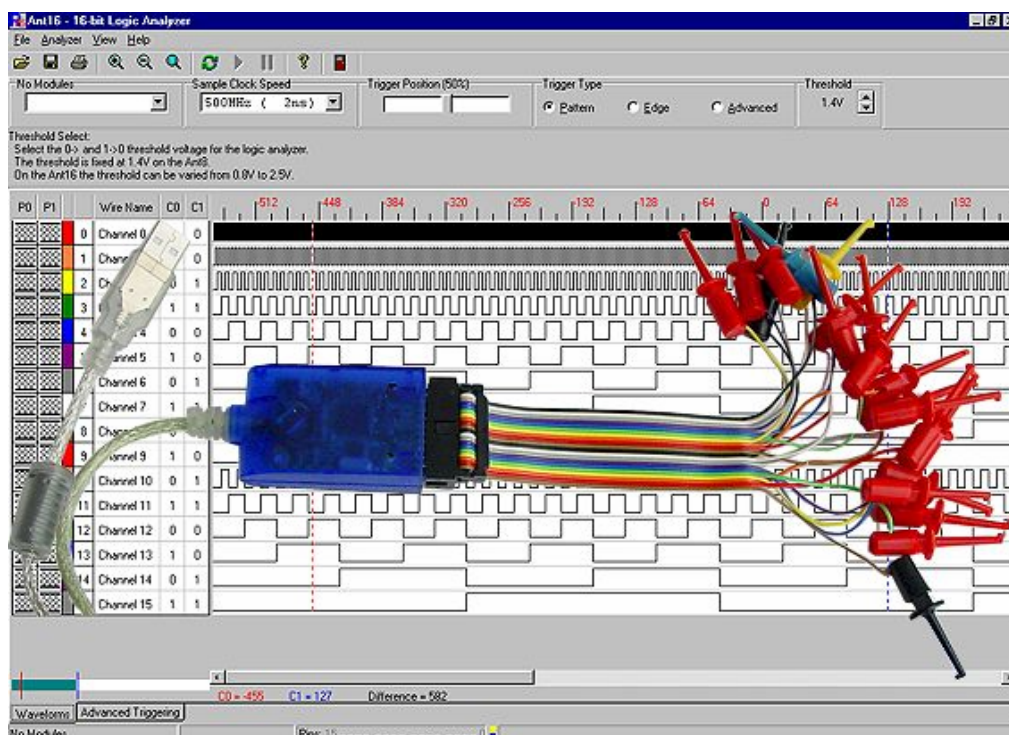


Figura 1. Analizador comercial para PC (y de fondo la imagen obtenida de una sesión de muestreo).

2. Requerimientos para el proyecto

Se deberá realizar un analizador lógico para PC, **sencillo** de usar, y de **costo reducido**. Para proveer más flexibilidad y aislamiento respecto del dispositivo a medir, se ha de hacer la implementación electrónica en un módulo externo a la PC.

El sistema deberá ser capaz de muestrear 8 canales, según dos tipos de funcionamiento.

El primer modo, de clock interno, está dado por muestreos generados periódicamente, con un periodo seleccionado por el usuario a través de una interfaz gráfica en la PC.

En el segundo modo, de clock externo, el sistema a testear provoca la toma de una muestra a partir de un cambio de estado en alguno de los 4 canales superiores*. El operador deberá escoger las señales que crea conveniente para el disparo y hacer las conexiones correspondientes.

El sistema se vale de la PC para la representación de las muestras a través de una interfaz gráfica. Esta misma permite la configuración completa del módulo externo del analizador lógico. La configuración consiste en el modo de funcionamiento (síncrono o asíncrono) y la frecuencia de muestreo (sólo para modo síncrono, es decir, de clock interno).

La interconexión se ha de llevar a cabo por medio del puerto serie.

El módulo externo contiene una punta por cada canal, y tiene identificadas las puntas según números. Las puntas de 4 a 7 son las generadoras de muestreo (triggers). Sólo generarán muestreo si está el sistema en el modo correspondiente. También se ha de incluir una punta de referencia.

Una etapa o sesión de muestreo es un conjunto de muestras tomadas de forma continua. Se tiene una cantidad máxima de 1024 muestras por cada sesión de muestreo**. La frecuencia máxima de muestreo es de 500 KHz***.

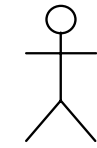
La interfaz gráfica de usuario presente en la PC, consiste en 8 ejes de coordenadas que muestran los niveles lógicos para cada canal, en función del tiempo. Luego de una sesión de muestreo, el usuario podrá recorrer en el tiempo las señales para su estudio. La escala de tiempo estará dada por el modo de funcionamiento, es decir, según cuántos (dados por el periodo de muestreo), o según momento registrado para cada muestra.

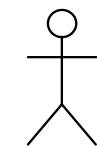
**Este modo es muy útil cuando se trata de dispositivos (a medir) con diferencias de velocidades importantes. Esto es porque uno de ellos generalmente queda a la espera del otro, no existiendo cambios interesantes durante periodos largos. Se debe recordar que la cantidad de muestras a tomar es limitada (1024 muestras).*

***Es probable que en el avance del proyecto se logren capacidades ligeramente superiores.*

****Es probable que en el avance del proyecto se pueda optimizar el algoritmo del procesador del módulo externo para obtener velocidades superiores de muestreo.*

3. Actores

 Operador	ACTOR: Operador
	TIPO: Primario
	DESCRIPCIÓN: Es quien ve al sistema desde el punto de vista electrónico. Realiza las conexiones propias para que se lleve a cabo la medición. Este actor se plantea a los efectos de contemplar el otro dominio del proyecto, que escapa al manejo de software con una interfaz gráfica.

 Observador	ACTOR: Observador
	TIPO: Primario
	DESCRIPCIÓN: Es quien tomará conclusiones a partir de la representación de las señales. Como el sistema presenta los resultados con ayuda de una PC, este actor naturalmente interactuará con el sistema a través de ella. En otras palabras, el Observador será el usuario desde la óptica de la PC.

4. Diagramas de Casos de Uso

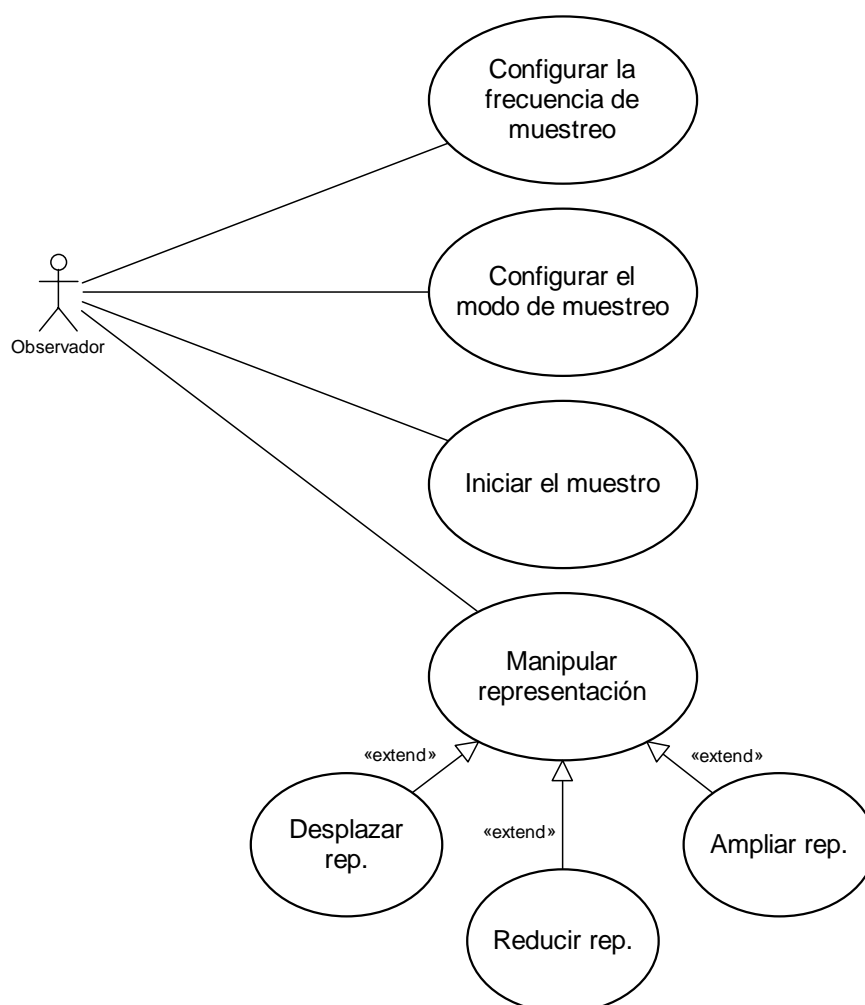
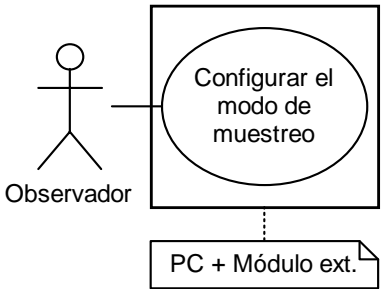
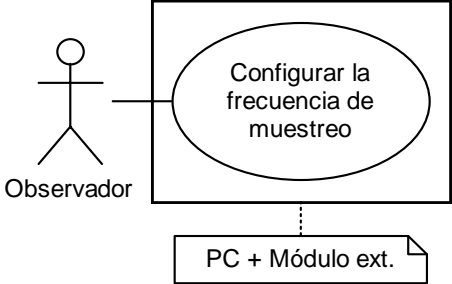
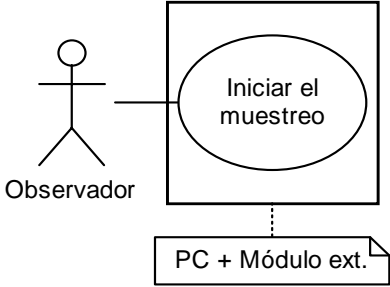


Figura 2. Diagrama General de Casos de Uso.

	CASO DE USO: Configurar el modo de muestreo
	ACTORES: Observador
	PROPÓSITO: Indicar al módulo externo qué modo utilizar.
	RESUMEN: El Observador establece el modo en el que debe trabajar el sistema durante el muestreo. El modo por defecto será el de clock interno. 1. Se obtiene un elemento en pantalla de configuración del módulo externo. 2. Se realiza la configuración correspondiente. 3. El módulo externo es avisado del nuevo parámetro (haya cambiado o no).
	PRECONDICIONES: Ninguna.
	EXCEPCIONES: Ninguna. La validez estará garantizada por la tipificación de los parámetros.

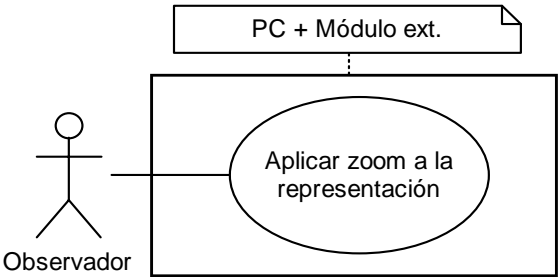
 <pre> graph LR Observador[Observador] --- Configurar(Configurar la frecuencia de muestreo) subgraph PC_Modulo_ext [PC + Módulo ext.] Configurar end </pre>	CASO DE USO: Configurar la freq. de muestreo
	ACTORES: Observador
	PROPÓSITO: Indicar al módulo externo a qué frecuencia debe tomar las muestras.
	RESUMEN: El Observador establece la frecuencia de muestreo mediante el ingreso de ciertos parámetros. 1. Se tiene una pantalla con un ítem de configuración del módulo externo. 2. Se realiza la configuración correspondiente. 3. El módulo externo es avisado del nuevo parámetro (haya cambiado o no).
	PRECONDICIONES: 1. El modo de funcionamiento debe ser el de clock interno.
	EXCEPCIONES: Ninguna. La validez de la configuración está garantizada por la tipificación de los datos.

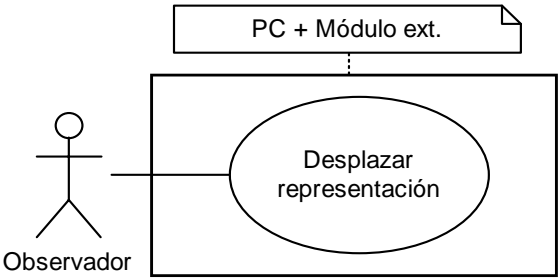
	CASO DE USO: Iniciar el muestreo
	ACTORES: Observador
	PROPÓSITO: Dar partida al inicio del muestreo por parte del módulo externo.
	RESUMEN: <p>El Observador da la orden a través de su interfaz gráfica para que comience el muestreo. Este se realizará según los parámetros ya configurados (o aquellos por defecto).</p> <ol style="list-style-type: none"> 1. El Observador genera un evento que corresponde al inicio del muestreo. 2. El módulo externo es informado de la solicitud, junto a los parámetros involucrados. 3. El módulo externo comienza con el muestreo. Una vez finalizado responde al sistema de la PC. 4. Una vez que se recibe el mensaje, la interfaz gráfica indica al usuario que el muestreo fue realizado con éxito.
	PRECONDICIONES: <ol style="list-style-type: none"> 1. El módulo debe estar correctamente conectado a la fuente de señales digitales. <p>Existen parámetros por defecto.</p>
	EXCEPCIONES: <ol style="list-style-type: none"> 1. El módulo no fue encontrado.

<p>The diagram shows a stick figure actor labeled 'Observador' connected to a use case 'Manipular representación'. Above this use case is a note box labeled 'PC + Módulo ext.' with a dashed line connecting to the use case. Below 'Manipular representación' are two other use cases: 'Aplicar zoom a la rep.' and 'Desplazar rep.'. Arrows point from these two use cases up to 'Manipular representación', with the label '«extend»' placed near the arrow from 'Aplicar zoom a la rep.'.</p>	CASO DE USO: Manipular representación
	ACTORES: Observador
	PROPÓSITO: Manejar las representaciones según parecer del Observador, en búsqueda de mejores perspectivas de la sesión de muestreo.
	RESUMEN: El usuario deberá estar con sesión activa de muestreo. Esto significa que ya se ha hecho una captura de antemano o se ha cargado un archivo de una sesión previa. La manipulación de la representación se lleva a cabo de acuerdo a diferentes procedimientos, según el tipo de manipulación requerida. En general se tiene: 1. El usuario decide el tipo de manipulación necesaria para su cometido. 2. Luego genera un evento que indica qué tipo de acción se ha de realizar, como zoom o desplazamiento. 3. El sistema genera las vistas correspondientes a los parámetros indicados por el usuario Observador.
	PRECONDICIONES: 1. Se debe tener una sesión activa de muestreo.

EXCEPCIONES:

1. Todavía no se han obtenido muestras.

 <pre> graph LR Observador[Observador] --- UC((Aplicar zoom a la representación)) UC -.- PC[PC + Módulo ext.] </pre>	CASO DE USO: Aplicar zoom a la representación
	ACTORES: Observador
	PROPÓSITO: Extender o reducir en tiempo las representaciones, para hacer foco en un evento particular o general con una amplia zona de la pantalla.
	RESUMEN: <ol style="list-style-type: none"> 1. El usuario genera el evento correspondiente al zoom. 2. Los parámetros asociados a esta acción son validados. 2. Luego de ser validados, todos canales son modificados según este nuevo valor de zoom. 3. El sistema utiliza los parámetros y realiza un nuevo dibujo para cada canal, con sus parámetros específicos correspondientes (como el color de la traza de la señal), y respetando los nuevos rangos. Sólo serán representadas las muestras contenidas en el rango.
	PRECONDICIONES: <ol style="list-style-type: none"> 1. Debe existir una sesión de muestreo ya activa (por captura o archivo).
	EXCEPCIONES: <ol style="list-style-type: none"> 1. No hay sesión activa. 2. Valor de zoom no válido.

 <pre> graph LR Observador[Observador] --- Desplazar([Desplazar representación]) Note[PC + Módulo ext.] -.-> Desplazar </pre>	CASO DE USO: Desplazar representación
	ACTORES: Observador
	PROPÓSITO: Mover el rango de representación sin modificar su tamaño.
	RESUMEN: <ol style="list-style-type: none"> 1. El Observador genera un evento asociado a un desplazamiento. Lo hace indicando el desplazamiento requerido. 2. El sistema valida los parámetros asociados al desplazamiento solicitado. 3. Luego de la validación, el sistema responde realizando por cada canal una nueva traza de las señales, con el nuevo rango. Sólo serán representadas aquellas muestras (para cada canal) que estén dentro del mismo.
	PRECONDICIONES: <ol style="list-style-type: none"> 1. Se tiene una sesión activa de muestreo.
	EXCEPCIONES: <ol style="list-style-type: none"> 1. No hay sesión abierta. 2. No es válido el parámetro.

5. Diagrama de Capas

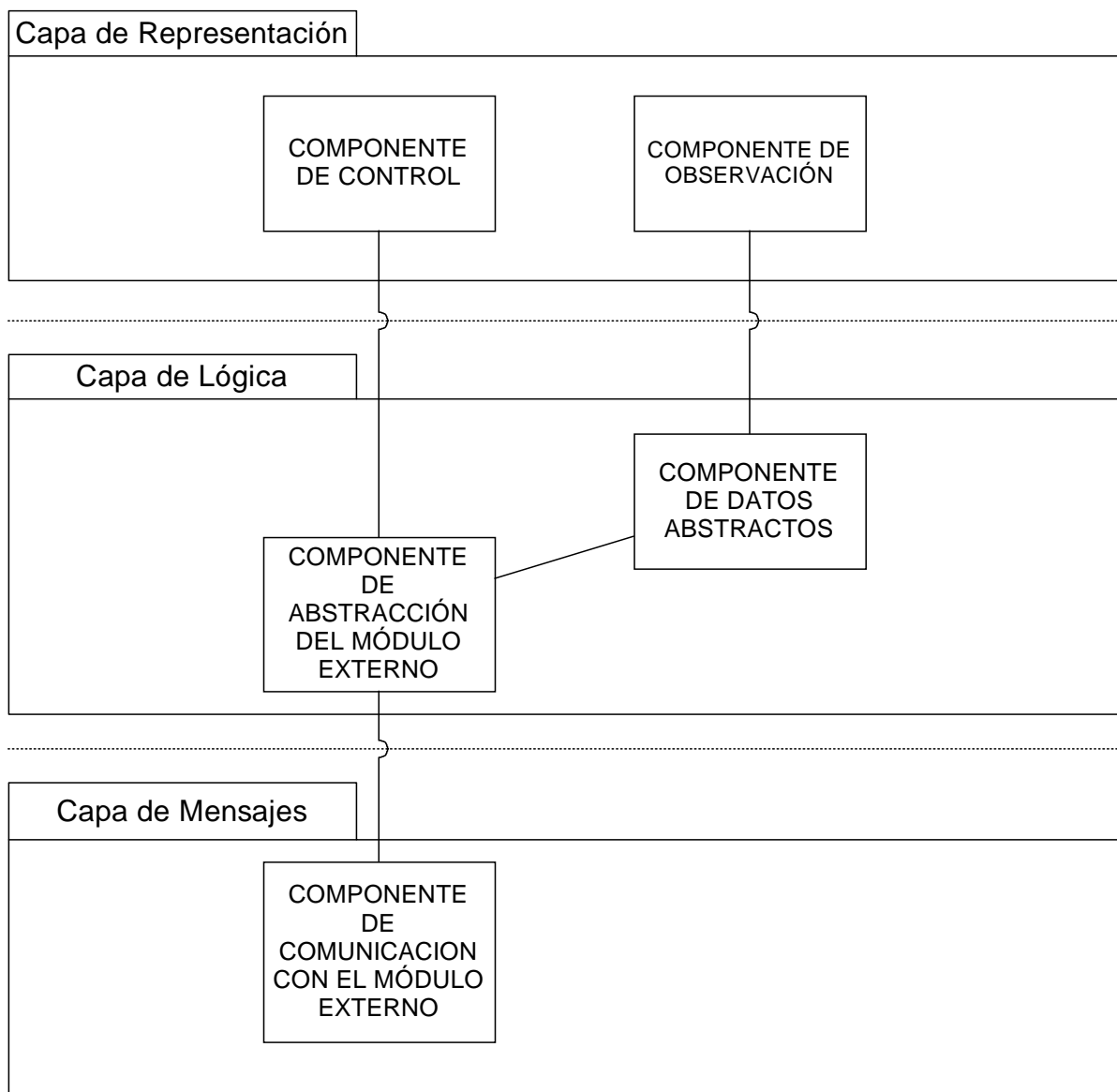


Figura 3. Patrón de capas.

El **Componente de Control** se encuentra en la Capa de Representación. Es el encargado de atender al usuario cuando sus acciones están relacionadas al manejo del módulo externo. Es decir, actividades como el inicio, la configuración del modo, y de la frecuencia de muestreo. Este componente indica sus eventos al Componente de Abstracción del Módulo Externo (que lo representa en la Capa de Lógica). Así actúa como intermediario entre el usuario (o más bien su interfaz gráfica) y el módulo externo (más bien con el componente que lo representa en la Capa de Lógica). El Componente de Abstracción del Módulo Externo no realiza validación importante a sus llamadas, por lo que es el Componente de Control quien deberá asegurar que el usuario ha ingresado parámetros válidos (en otras palabras, tiene como responsabilidad también la validación).

El **Componente de Observación** es el encargado de atender al usuario cuando sus acciones estén relacionadas al manejo y visualización de vistas de una sesión de muestreo. Este componente es el más importante desde el punto de vista del usuario (Observador). Atenderá a eventos que tiendan a manipular las representaciones (véanse los casos de uso para los tipos de manipulación), y proveerá gráficamente las mismas.

El **Componente de Datos Abstractos** es un componente dedicado a disponer de la información relacionada a una sesión de muestreo, para proveerla cuando le sea solicitada. Estará constituido principalmente por estructuras y conceptos como canal, muestra, etc. Su responsabilidad es dar facilidades a la carga de datos para con la capa inferior, y dar facilidad para la obtención de los datos por parte de la capa superior. Actúa como un buffer, hace las veces de nexo entre el Componente de Abstracción del Módulo Externo y el Componente de Observación, que se vale de los datos allí alojados para llevar a cabo su representación.

El **Componente de Abstracción del Módulo Externo** (CAME) tiene como funcionalidad dar a la capa superior acceso sencillo al módulo externo, de modo que sea fácil interactuar con el mismo sin preocuparse por el protocolo involucrado en la comunicación. Se encuentra en la Capa de Lógica, constituye para la capa superior un punto de desacople respecto del protocolo de comunicación con el módulo. Así la interfaz usada por la Capa de Representación permanece inmutable ante cambios en el protocolo de comunicación. Este componente conoce las primitivas dadas por el Componente de Comunicación con el Módulo Externo, y se vale de ellas (como su interfaz) para controlar al módulo según sea requerido por el Componente de Control.

El **Componente de Comunicación con el Módulo Externo** es utilizado únicamente por el CAME. Está constituido por un conjunto de primitivas de comunicación con el módulo externo. Estas están especificadas en el protocolo mismo, que ha de ser respetado tanto por el sistema de la PC como por el sistema del Módulo Externo.

6. Diagramas de Clases

6.1. Diagrama General de Clases

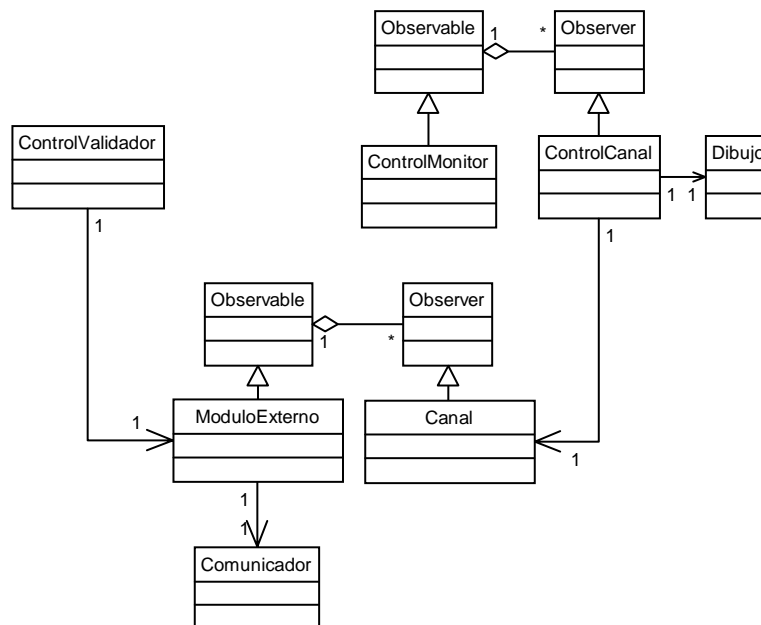


Figura 4. Diagrama General de Clases. Notar que tanto Canal (sus instancias) como ModuloExterno y, ControlMonitor con los ControlCanal, constituyen patrones *Observer*.

Como se puede observar en el mapa de clases, estas siguen las líneas propuestas por el diagrama de capas presentado anteriormente. El diagrama presenta las clases relacionadas al analizador. El Comunicador se asocia estrictamente con el ModuloExterno, es este último quien permite cierta abstracción para el envío de comandos. Es el ModuloExterno quien traduce una solicitud de comportamiento del módulo físico, a una serie de comandos entendibles por el comunicador, y por el Hardware Externo.

El ModuloExterno informa de sus cambios (aquellos pertinentes, correspondientes a una nueva sesión de muestreo) a cada uno de los 8 canales. Estos obtienen los datos actualizados respecto de la última sesión. Presentan los datos de manera que sea fácil su proceso de manipulación.

Por una solicitud a manos del propio Observador, el ControlValidador es quien informa al ModuloExterno de la necesidad de pedir un nuevo muestreo. Así mismo, informa de cambios en la configuración del mismo, como modo o frecuencia de muestreo.

Por otra parte, el Observador intentará visualizar la sesión. Para ello indirectamente interactuará (los objetos de interacción directa serán botones, scrolls, etc.) con el mismo ControlMonitor. Este debe informar los requerimientos para la vista general a cada ControlCanal, para que la solicitud sea tratada por cada canal independientemente.

Los objetos Dibujo responden a solicitudes del ControlCanal asociado, solicitudes de actualización de la representación misma. En sí, su responsabilidad radica en presentar gráficamente lo que se le solicite. Los 8 objetos Dibujo están sincronizados por medio del ControlCanal al ControlMonitor que gestiona la vista general.

A continuación se dará una explicación más minuciosa de lo anteriormente introducido.

6.2. Capa de Representación y Lógica (manipulación de vistas)

Realizado el estudio de los escenarios de los Casos de Uso, y tenidos en cuenta los componentes de cada capa de la arquitectura anteriormente planteada, se pueden obtener las siguientes clases para este diseño.

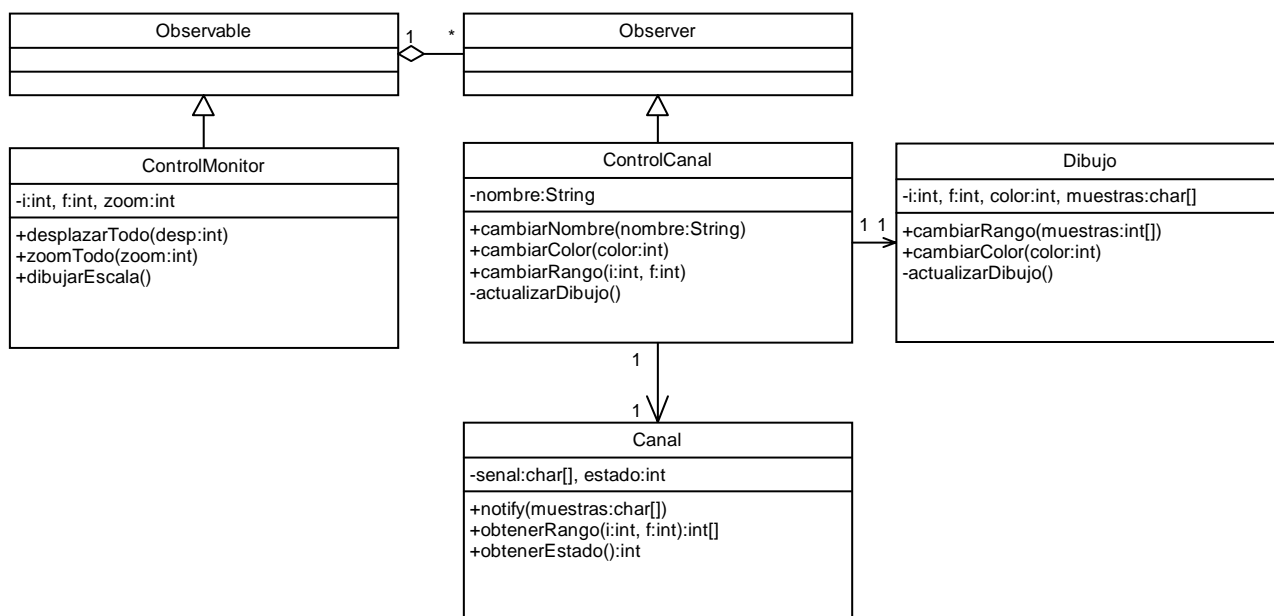


Figura 5. Diagrama de Clases de la Capa de Representación, Componente de Observación (y Componente de Datos Abstractos, Capa de Lógica).

Este diagrama muestra cómo interactúan los objetos relacionados a las Capas de Representación y de Lógica.

La Capa de Representación está dada por los objetos **ControlMonitor**, **ControlCanal** y **Dibujo**. El primero atiende los eventos del usuario (por medio de objetos de interfaz gráfica), pero aquellos asociados al "monitor", a los objetos de representación de las señales. En otras palabras el **ControlMonitor** atiende eventos en relación a actividades como el desplazamiento, el zoom, etc.

Se tienen dos zonas importantes para el control del usuario, la zona de monitor (visualización de señales), y de configuración del muestreo (previa a la anterior).

En cuanto a la zona de monitor, un cambio en el **ControlMonitor** notifica a cada **ControlCanal**, que toma datos de su **Canal** asociado y los vuelca en su **Dibujo** asociado, según los cambios sucedidos en **ControlMonitor**. **ControlCanal** también atiende a los eventos del usuario, pero sólo a aquellos relacionados a un canal en particular (nombre del canal, color). Ante un cambio en sí mismo, el **ControlCanal** lleva a cabo las acciones correspondientes de manera directa (esta vez sin patrón Observador de por medio).

6.3. Capa de Representación y Lógica (manipulación del módulo externo)

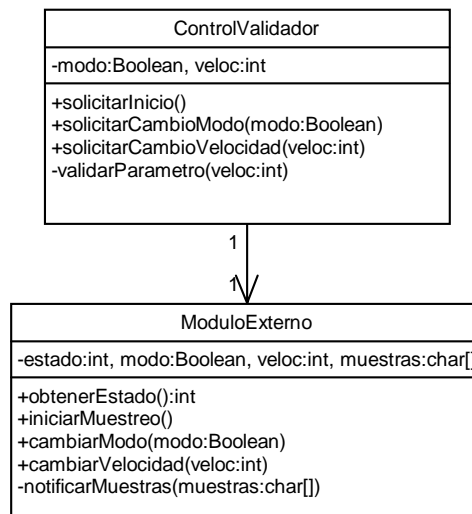


Figura 6. Diagrama de Clases de la Capa de Representación y Lógica (Componentes de Control y de Abstracción del Módulo Externo).

Este diagrama muestra cómo están conformadas estáticamente las relaciones entre las Capas de Lógica y de Representación, pero únicamente en lo que respecta a configuración del módulo externo. El **ControlValidador** atiende al usuario y valida sus parámetros, luego de ello, sin más, se vale de los métodos del **ModuloExterno** para provocar la comunicación y luego el efecto deseado. En lo que sigue del trabajo se tratarán los diagramas dinámicos (por ejemplo los de secuencia) y se hará más hincapié en estas secuencias.

El **ModuloExterno** es la interfaz que tienen los objetos (de la misma capa y de la superior) para comunicarse con la extensión de hardware del analizador lógico. Así provee de métodos para realizar cambios de modo, de frecuencia, inicialización, etc.

6.4. Capa de Lógica

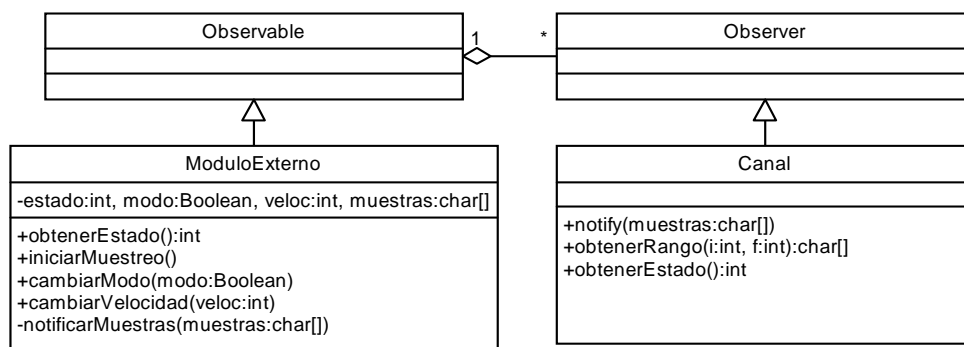


Figura 7. Diagrama de Clases de la Capa de Lógica.

Este diagrama muestra otra situación en la que también se usa el patrón Observador. La clase Canal es parte del Componente de Datos Abstractos. El ModuloExterno establece comunicación con el Canal únicamente al finalizar un muestreo, para realizar la carga de datos sobre este.

El Canal recibe una notificación de cambio en el Módulo Externo, junto a una copia del buffer original de datos (con todos los canales). En la misma carga de datos, el Canal descarta la información que no corresponde a su canal, dejándola con formato un tanto más manipulable para la representación.

Notar que el Módulo Externo tiene atributos asociados a la extensión, varios. Esto es para evitar establecer comunicaciones con el módulo en momentos distintos al de finalización del muestreo. Con esto mejora sustancialmente la velocidad de la aplicación.

6.5. Capa de Lógica y de Mensajes

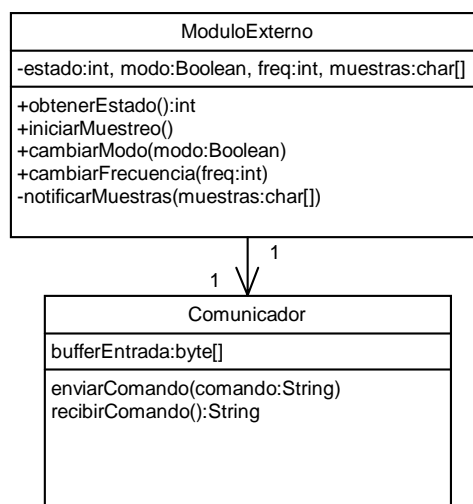


Figura 8. Diagrama de Clases de la Capa de Lógica y de Mensajes.

Este diagrama muestra la importancia del ModuloExterno en cuanto a la abstracción del protocolo utilizado para comunicarse con el módulo externo. Notar que mientras el ModuloExterno provee métodos de alto nivel como *obtener estado*, *iniciar muestreo*, *cambiar el modo*, etc. el comunicador solamente conoce cómo enviar y recibir bytes (“comandos”). El Comunicador también provee un servicio: almacena aquellos bytes que aún no fueron solicitados con un `recibirComando()`.

7. Diagramas de Secuencia

7.1. Cambio de color de un canal

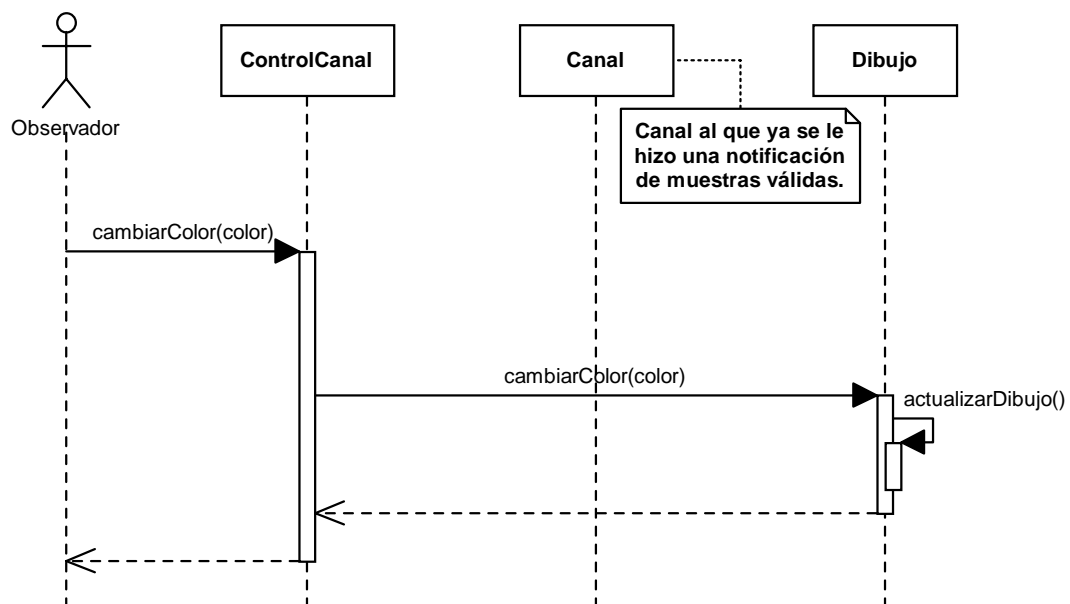


Figura 9. Cambio del color de un canal.

7.2. Desplazamiento de la vista

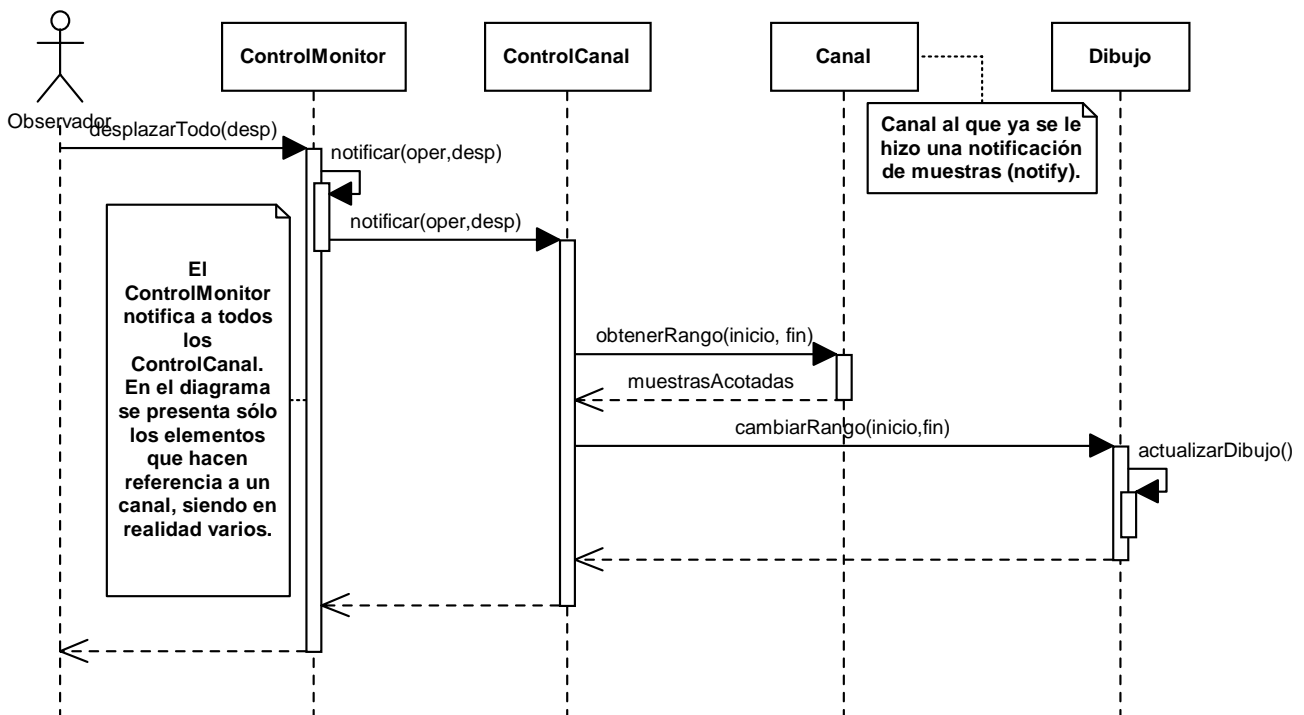


Figura 10. Desplazamiento de la vista.

En esta operación, están involucrados varios elementos. El Observador inicia la secuencia solicitando un desplazamiento. Esto es captado por el objeto ControlMonitor, el cual atiende a esta clase de eventos. El ControlMonitor junto con el ControlCanal, es parte de un patrón Observador (*Observer*). El ControlMonitor informa a sus observadores del cambio, su tipo y valor. Para el caso de desplazamiento, el valor es un entero que indica cuántas muestras se han de desplazar respecto de la posición original. El ControlCanal, procede a solicitarle un rango de muestras, aquellas requeridas para volver a graficar esta señal. Cuando obtiene los valores de esas muestras, se las envía al objeto Dibujo, quien las recibe y procede a actualizar su representación.

7.3. Inicio de muestreo

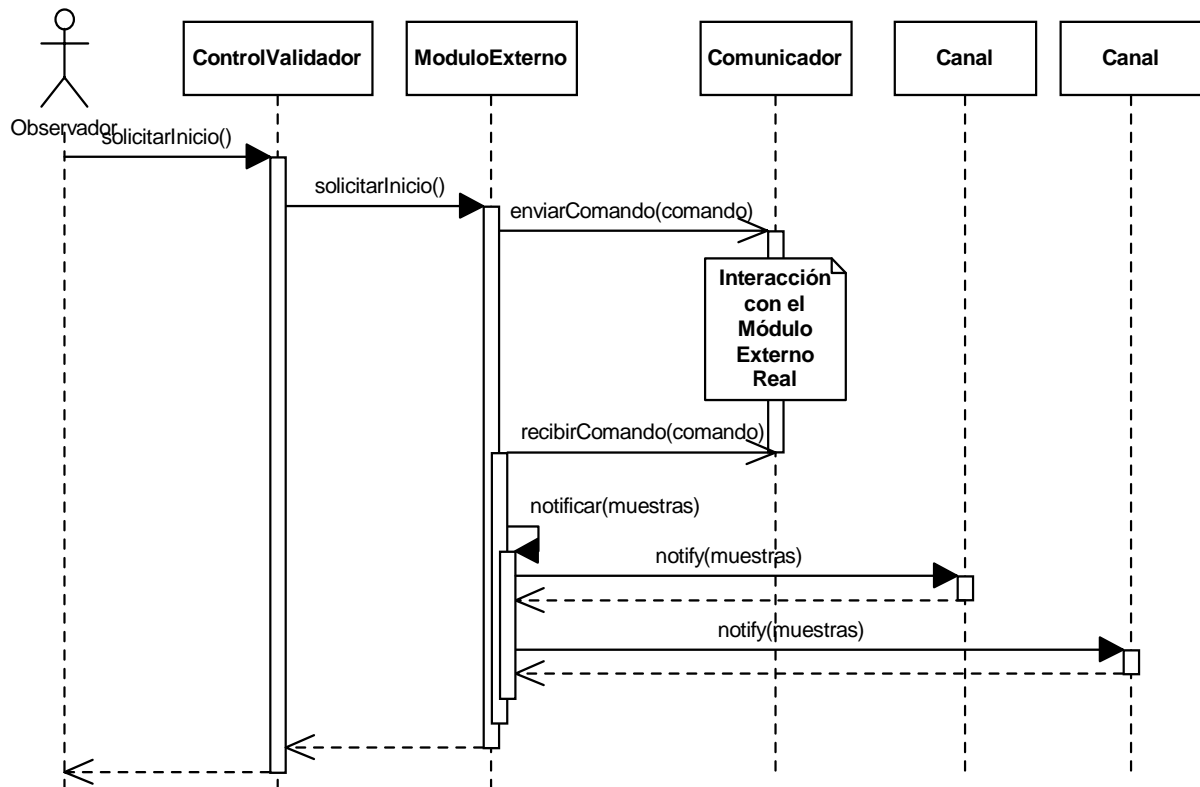


Figura 11. Inicio de un muestreo.

El inicio del muestreo involucra también varios elementos. La secuencia es iniciada por el Observador, que genera un evento capturado por el ControlValidador. Este comunica al objeto ModuloExterno de la solicitud. A través de una serie de comandos conocidos (como protocolo de comunicación de más alto nivel) indica finalmente al Módulo Externo real que es necesario arrancar el muestreo. Terminado el muestreo, el ModuloExterno recibe los datos del Comunicador. El ModuloExterno, bajo el patrón Observer, notifica a sus observadores (los diferentes objetos Canal) de los cambios. Cada canal procesa las muestras obtenidas y descarta las señales que corresponden a los demás canales, quedándose sólo con las que corresponden a él mismo.