

**REPÚBLICA DE COLOMBIA
SUPERINTENDENCIA DE INDUSTRIA Y COMERCIO**

Oficio N° 13891

Ref. Expediente N° NC2017/0003239

Señor(a)
HOME BOX OFFICE, INC
notificaciones@clarkemodet.com.co

Asunto: PATENTE DE INVENCION - PCT

Patente de Invención	PT	Patente de Modelo de Utilidad			
Vía Nacional					
Vía PCT (Fase Nacional)	PCT	Capítulo I	X	Capítulo II	
No. Solicitud Internacional	PCT/US2015/048418				
Fecha de Presentación Internacional	3 de septiembre de 2015				

Como resultado del examen de fondo realizado, con fundamento en el Artículo 45 de la Decisión 486 de la Comisión de la Comunidad Andina, se comunica:

1. Documentos estudiados

Descripción	Radicado N° NC2017/0003239	3 de abril de 2017
Reivindicaciones:	Radicado N° NC2017/0003239	3 de abril de 2017
	Radicado N° NC2017/0003239	2 de junio de 2017
	Radicado N° NC2017/0003239	8 de octubre de 2018
Dibujos:	Radicado N° NC2017/0003239	3 de abril de 2017
Respuesta del solicitante:	Radicado N° NC2017/0003239	8 de octubre de 2018

**REPÚBLICA DE COLOMBIA
SUPERINTENDENCIA DE INDUSTRIA Y COMERCIO**

Oficio N° 13891

Ref. Expediente N° NC2017/0003239

Prioridad(es):	País: US, ESTADOS UNIDOS DE AMÉRICA Número: 62/046,109 Fecha: 4 de septiembre de 2014
	País: US, ESTADOS UNIDOS DE AMÉRICA Número: 14/843,792 Fecha: 2 de septiembre de 2015

2. Análisis de la Prioridad

Se acepta la reivindicación de prioridad porque esta solicitud fue presentada dentro del plazo establecido y el contenido técnico de esta solicitud corresponde con el de la prioridad reivindicada.

3. Modificaciones presentadas (Art. 34 D 486 y Art. 19, 22 y 34 PCT)

Se aceptan las modificaciones presentadas comoquiera que se ajustan a las prescripciones contenidas en el reglamento del PCT y en el Art 34 de la Decisión 486.

4. Objeto de la invención

De acuerdo con lo establecido en el numeral 1.2.2.2 del Capítulo Primero del Título X de la Circular Única para efectos del examen de patentabilidad, se tendrá en cuenta la tasa pagada por el solicitante al momento de comenzar el examen para las reivindicaciones adicionales; esto es que cuando la solicitud contenga más de diez (10) reivindicaciones y no se haya pagado la tasa complementaria, sólo se estudiarán las cubiertas por la tasa pagada. Por lo tanto, se estudiarán las reivindicaciones 1 a 20 del capítulo reivindicatorio (Radicado N° NC2017/0003239 del 8 de octubre de 2018).

Título de la solicitud: “MÉTODO Y SISTEMA QUE TIENE MODELOS POPULADOS CON DATOS ASINCRÓNICOS” (Radicado N° NC2017/0003239 del 2 de junio de 2017).

El problema planteado en esta solicitud consiste en la necesidad de optimizar el proceso de recuperación o recopilación de información desde una fuente de datos hacia una vista que requiere de dicha información.

**REPÚBLICA DE COLOMBIA
SUPERINTENDENCIA DE INDUSTRIA Y COMERCIO**

Oficio N° 13891

Ref. Expediente N° NC2017/0003239

Para resolver este problema técnico, la solicitud en estudio proporciona un sistema y métodos para recopilar información a través de modelos asíncronos que vinculan a la fuente de datos con la vista. Una vista solicita a un modelo de vista para devolver datos de vista especificados; a su vez, el modelo de vista hace una o más peticiones a un modelo de datos para recuperar la información correspondiente a los datos de vista especificados. El modelo de datos se comunica con una fuente de datos, tales como un servicio web o una memoria caché para devolver la información solicitada al modelo de vista.

El objeto de la presente solicitud se relaciona con la Clasificación Internacional de Patentes (CIP): G06F 9/44.

5. Determinación del Estado de la Técnica

El estado de la técnica se determinó con base en la fecha de prioridad: 4 de septiembre de 2014, y se encontraron los siguientes documentos que anticipan el objeto de esta solicitud:

Ítem	Documento	Título	Fecha de Publicación
D1	Artículo*	"Async MVVM Modern UI"	16/Agosto/2012
D2	Artículo**	"JavascriptPromise"	22/Abril/2013

* Juan Pablo G.C. ** Martin Fowler.

El documento D1¹ divulga un ejemplo del uso de MVVM con Windows 8 Modern UI con enlace de datos, comandos y propiedades utilizando la metodología asíncrona y manteniendo el modo de diseño con plantillas de datos (Sección Introducción).

El documento D2² divulga características y algunos ejemplos de promesas implementadas en Javascript (Todo el documento).

6. Examen de Patentabilidad (Art. 14 D486)

Nivel Inventivo (Art. 18 D486)

1 <https://www.codeproject.com/Articles/441752/Async-MVVM-Modern-UI>

2 <https://martinfowler.com/bliki/JavascriptPromise.html>

**REPÚBLICA DE COLOMBIA
SUPERINTENDENCIA DE INDUSTRIA Y COMERCIO**

Oficio N° 13891

Ref. Expediente N° NC2017/0003239

La reivindicación 1 consiste en: “*Un método para recibir, en un modelo de vista, una solicitud de datos específicos de la vista de una vista, que comprende: (...)*” (Radicado N° NC2017/0003239 del 8 de octubre de 2018).

Tabla 1. Comparación de las características técnicas esenciales, con las del estado de la técnica:

Características esenciales	D1	D2
Recibir, en un modelo de vista, una solicitud de datos específicos de la vista de una vista.	Finalmente llegamos al XAML, he creado una página simple con los botones de comando, <i>GridView</i> y <i>TextBlock</i> para el elemento seleccionado. Cuida lo siguiente: - Creo una instancia de <i>ViewModel</i> en <i>Page.DataContext</i>. - El primer botón vincula el comando con '<i>GetFilesCommand</i>' y enlaza <i>IsEnabled</i> con <i>IsIdle</i> (...) (Sección View). (...) El comando ' <i>GetFilesCommand</i> ' para leer archivos (...) (Sección Properties and Fields).	No se menciona.
Comunicar, mediante un dispositivo que comprende un procesador a través de una vista a un modelo de vista (102), una solicitud (106) de datos específicos de la vista asociados con la vista.	El ' <i>GetFilesCommand</i> ' ha implementado una acción con el prefijo <i>async</i> (lo descubrí por prueba y error, no lo encontré en ningún artículo) y en el interior llamo a ' <i>GetFilesAsync</i> ' para obtener los archivos como pueda la aplicación (Sección Constructor).	No se menciona.
Recibir (802), por el dispositivo por medio del modelo de vista (102), la solicitud de datos para los datos específicos de la vista.	Divido ' <i>GetFilesAsync</i> ' para mostrar que simplemente está llamando a un método dentro de <i>Task.Run</i> . Ese método necesita tener <i>async</i> como prefijo para saber que lo quieres en el 'segundo plano'. ' <i>addfile</i> ' es la acción que agrega archivos mientras <i>_cancel</i> no está activado (...) (Sección The heart).	No se menciona.
Hacer mediante el dispositivo a través del modelo de vista (102), una o más solicitudes asincrónicas a un modelo de datos para la información	El modelo de vista implementa una versión personalizada de <i>INotifyPropertyChanged</i> para ser asincrónico y actualiza la UI (Interfaz de Usuario) cuando es capaz (Sección ViewModel). Divido ' <i>GetFilesAsync</i> ' para mostrar que simplemente está llamando a un método	No se menciona.

REPÚBLICA DE COLOMBIA
SUPERINTENDENCIA DE INDUSTRIA Y COMERCIO

Oficio N° 13891

Ref. Expediente N° NC2017/0003239

correspondiente a los datos específicos de la vista.	dentro de <i>Task.Run</i> . Ese método necesita tener <i>async</i> como prefijo para saber que lo quieres en el 'segundo plano'. ' <i>addfile</i> ' es la acción que agrega archivos mientras <i>_cancel</i> no está activado, como puede ver, no necesito ninguna lógica especial para saber el valor de cancelar (Sección The heart).	
Comunicar, mediante un dispositivo a través del modelo de vista (102), una primera promesa a la vista para proporcionar los datos específicos de la vista en respuesta a la solicitud de los datos específicos de la vista.	No se menciona.	En Javascript, las promesas son objetos que representan el resultado pendiente de una operación asincrónica. Puede usarlos para programar más actividades después de que la operación asincrónica haya finalizado proporcionando una devolución de llamada (Párrafo 1).
Realizar, por el dispositivo a través de la vista, antes de que la vista reciba los datos específicos de la vista, una tarea relacionada con la visualización de la vista.	Finalmente llegamos al XAML, he creado una página simple con los botones de comando, GridView y TextBlock para el elemento seleccionado. Tenga cuidado de lo siguiente: - (...) El TextBlock enlazar texto a CurrentFile - GridView vincula ItemsSource con 'Files' y SelectedItem con el modo CurrentFile TwoWay (Sección View). El modelo de vista implementa una versión personalizada de <i>INotifyPropertyChanged</i> para ser asincrónico y actualiza la UI (Interfaz de Usuario) cuando es capaz (Sección ViewModel).	No se menciona.
Obtener, por el dispositivo a través del modelo de datos, un conjunto de datos (1010, 1018) que incluye la información.	El comando ' <i>GetFilesCommand</i> ' para leer archivos (...) (Sección Properties and Fields).	No se menciona.
Proveer (1022), por el dispositivo a través del modelo de datos, la información para el	El modelo de vista implementa una versión personalizada de <i>INotifyPropertyChanged</i> para ser asincrónico y actualiza la UI (Interfaz de Usuario) cuando es capaz	No se menciona.

**REPÚBLICA DE COLOMBIA
SUPERINTENDENCIA DE INDUSTRIA Y COMERCIO**

Oficio N° 13891

Ref. Expediente N° NC2017/0003239

modelo de vista en respuesta a una o más solicitudes asíncronas.	(Sección ViewModel). Para actualizar la UI cuando usa asincrónico, debe usar el despachador de ventana actual. Como puede ver, creo un método asíncrono que actualiza la UI llamada <i>'UIThreadAction'</i> y varios métodos como <i>'PropertyChangedAsync'</i> que llama <i>'PropertyChanged'</i> dentro de la acción del despachador (Sección Implementing INotifyPropertyChanged).	
Comunicar, mediante un dispositivo a través del modelo de vista (102), los datos específicos de la vista, basándose en la información devuelta al modelo de vista, desde el modelo de vista a la vista para cumplir la primera promesa.		No se menciona.

El documento D1 se considera el estado de la técnica más cercano a la invención definida en la reivindicación 1 porque divulga un ejemplo del uso de MVVM con Windows 8 Modern UI con enlace de datos, comandos y propiedades utilizando la metodología asincrónica y manteniendo el modo de diseño con plantillas de datos (Sección Introducción).

La diferencia entre la invención, definida en la reivindicación 1 y el método de D1 consiste en que en el método de la reivindicación 1 incluye el uso de promesas en el modelo de vista.

El efecto que logra el uso de promesas es gestionar las respuestas de las solicitudes de datos especificados.

Por lo tanto, el problema técnico objetivo que pretende resolver esta invención se puede formular así: “¿cómo modificar el método conocido en D1 para gestionar las respuestas de las solicitudes de datos especificados?”.

Sin embargo, el uso de promesas para gestionar las respuestas de las solicitudes de datos especificados ya está divulgado en el documento D2 que se refiere a que en Javascript, las promesas son objetos que representan el resultado pendiente de una operación asincrónica. Puede usarlos para programar más actividades después de que la operación asíncrona haya finalizado proporcionando una devolución de llamada (Párrafo 1).

En consecuencia, la persona del oficio normalmente versada en la materia incluiría el uso de promesas para gestionar las respuestas de las solicitudes de datos

REPÚBLICA DE COLOMBIA
SUPERINTENDENCIA DE INDUSTRIA Y COMERCIO

Oficio N° 13891

Ref. Expediente N° NC2017/0003239

especificados mencionado en el documento D2 en el método divulgado en el documento D1 para llegar así al objeto de la reivindicación 1 de la solicitud. Por lo cual se considera obvia.

Asimismo, los documentos D1 y D2 divulgan las características de las reivindicaciones dependientes así:

- Reivindicación 3: En Javascript, las promesas son objetos que representan el resultado pendiente de una operación asincrónica. Puede usarlos para programar más actividades después de que la operación asíncrona haya finalizado proporcionando una devolución de llamada (D2, Párrafo 1).
- Reivindicación 9: En Javascript, las promesas son objetos que representan el resultado pendiente de una operación asincrónica. Puede usarlos para programar más actividades después de que la operación asíncrona haya finalizado proporcionando una devolución de llamada (D2, Párrafo 1).
- Reivindicación 10: (...) Creo una instancia de ViewModel en Page.DataContext (...) (Sección View). (...) Aquí inicializo los Comandos, el despachador y la colección (...) (D1, Sección Constructor).

Las reivindicaciones dependientes 3 y 9 a 10 no mencionan alguna característica que, junto con las características de la reivindicación independiente, aporte nivel inventivo al método de la reivindicación 1, por lo cual se considera que no tienen nivel inventivo.

La reivindicación 11 consiste en: “*Un sistema para recibir, en un modelo de vista, una solicitud de datos específicos de la vista de una vista, que comprende: (...)*” (Radicado N° NC2017/0003239 del 8 de octubre de 2018).

Tabla 2. Comparación de las características técnicas esenciales, con las del estado de la técnica:

Características esenciales	D1	D2
Recibir, en un modelo de vista, una solicitud de datos específicos de la vista de una vista.	Finalmente llegamos al XAML, he creado una página simple con los botones de comando, <i>GridView</i> y <i>TextBlock</i> para el elemento seleccionado. Cuida lo siguiente: - Creo una instancia de <i>ViewModel</i> en <i>Page.DataContext</i>. - El primer botón vincula el comando con '<i>GetFilesCommand</i>'	No se menciona.

Página 7 de 17

REPÚBLICA DE COLOMBIA
SUPERINTENDENCIA DE INDUSTRIA Y COMERCIO

Oficio N° 13891

Ref. Expediente N° NC2017/0003239

	y enlaza <i>IsEnabled</i> con <i>IsIdle</i> (...) (Sección View). (...) El comando ' <i>GetFilesCommand</i> ' para leer archivos (...) (Sección Properties and Fields).	
Un procesador. Una memoria acoplada comunicativamente al procesador, la memoria que tiene almacenada en ella componentes ejecutables por computadora.	Para la implementación de los patrones de arquitectura de software descritos es evidente la utilización de una arquitectura física para su realización donde dicha arquitectura corresponde con una computadora general que comprende los elementos básicos como son un procesador, una memoria, entre otros.	Para la implementación de los patrones de arquitectura de software descritos es evidente la utilización de una arquitectura física para su realización donde dicha arquitectura corresponde con una computadora general que comprende los elementos básicos como son un procesador, una memoria, entre otros.
Un modelo de vista (102) configurado para ser acoplado a un objeto de vista (106).	El modelo de vista implementa una versión personalizada de <i>INotifyPropertyChanged</i> para ser asíncrono y actualiza la interfaz de usuario cuando es capaz (Sección View Model).	No se menciona.
Un modelo de datos (110) configurado para ser acoplado al modelo de vista.	El modelo es muy simple, por lo general prefiero agregar un presentador para la vista en lugar de agregar un convertidor, use la opción que desee. Tenga en cuenta que en <i>W8RP</i> el formato de fecha ha cambiado a <i>GetDateTimeFormats</i> (Sección Model).	No se menciona.
En donde el modelo de vista está configurado además para recibir una solicitud (802) de datos específicos de la vista desde el objeto de vista.	El ' <i>GetFilesCommand</i> ' ha implementado una acción con el prefijo <i>async</i> (lo descubrí por prueba y error, no lo encontré en ningún artículo) y en el interior llamo a ' <i>GetFilesAsync</i> ' para obtener los archivos como pueda la aplicación (Sección Constructor).	No se menciona.
Solicitar (810) de forma asíncrona la información correspondiente a los datos específicos de la vista desde el modelo de datos.	El modelo de vista implementa una versión personalizada de <i>INotifyPropertyChanged</i> para ser asíncrono y actualiza la UI (Interfaz de Usuario) cuando es capaz (Sección ViewModel). Divido ' <i>GetFilesAsync</i> ' para mostrar que simplemente está llamando a un método dentro de <i>Task.Run</i> . Ese método necesita tener <i>async</i> como prefijo para saber que lo	No se menciona.

**REPÚBLICA DE COLOMBIA
SUPERINTENDENCIA DE INDUSTRIA Y COMERCIO**

Oficio N° 13891

Ref. Expediente N° NC2017/0003239

	quieres en el 'segundo plano'. 'addfile' es la acción que agrega archivos mientras <u>cancel</u> no está activado, como puede ver, no necesito ninguna lógica especial para saber el valor de cancelar (Sección The heart).	
Comunicar una primera promesa desde el modelo de vista a la vista para proporcionar los datos específicos de la vista en respuesta a la solicitud.	No se menciona.	En Javascript, las promesas son objetos que representan el resultado pendiente de una operación asincrónica. Puede usarlos para programar más actividades después de que la operación asíncrona haya finalizado proporcionando una devolución de llamada (Párrafo 1).
En donde el objeto de vista está configurado para realizar, antes de que la vista reciba los datos específicos de la vista, una tarea relacionada con la representación del objeto de vista.	Finalmente llegamos al XAML, he creado una página simple con los botones de comando, GridView y TextBlock para el elemento seleccionado. Tenga cuidado de lo siguiente: - (...) El TextBlock enlazar texto a CurrentFile - GridView vincula ItemsSource con 'Files' y SelectedItem con el modo CurrentFile TwoWay (Sección View). El modelo de vista implementa una versión personalizada de <i>INotifyPropertyChanged</i> para ser asincrónico y actualiza la UI (Interfaz de Usuario) cuando es capaz (Sección ViewModel).	No se menciona.
En donde el modelo de datos está configurado además para obtener (1010, 1018) un conjunto de datos que incluye la información de una fuente de datos (116) externa al sistema.	El comando ' <i>GetFilesCommand</i> ' para leer archivos (...) (Sección Properties and Fields).	No se menciona.
Devolver (1022) la información al modelo de vista.	El modelo de vista implementa una versión personalizada de <i>INotifyPropertyChanged</i> para ser asincrónico y actualiza la UI	No se menciona.

**REPÚBLICA DE COLOMBIA
SUPERINTENDENCIA DE INDUSTRIA Y COMERCIO**

Oficio N° 13891

Ref. Expediente N° NC2017/0003239

En donde el modelo de vista está configurado además para comunicar una respuesta (904) que comprende los datos específicos de la vista al objeto vista en base a la información para cumplir la primera promesa.	(Interfaz de Usuario) cuando es capaz (Sección ViewModel). Para actualizar la UI cuando usa asincrónico, debe usar el despachador de ventana actual. Como puede ver, creo un método asíncrono que actualiza la UI llamada <i>'UIThreadAction'</i> y varios métodos como <i>'PropertyChangedAsync'</i> que llama <i>'PropertyChanged'</i> dentro de la acción del despachador (Sección Implementing <i>INotifyPropertyChanged</i>).	No se menciona.
--	---	-----------------

El documento D1 se considera el estado de la técnica más cercano a la invención definida en la reivindicación 11 porque divulga un ejemplo del uso de MVVM con Windows 8 Modern UI con enlace de datos, comandos y propiedades utilizando la metodología asincrónica y manteniendo el modo de diseño con plantillas de datos (Sección Introducción).

La diferencia entre la invención, definida en la reivindicación 11 y el sistema de D1 consiste en que en el sistema de la reivindicación 11 se incluye el uso de promesas en el modelo de vista.

El efecto que logra incluir el uso de promesas en el sistema es gestionar las respuestas de las solicitudes de datos especificados.

Por lo tanto, el problema técnico objetivo que pretende resolver esta invención se puede formular así: “¿cómo modificar el sistema conocido en D1 para gestionar las respuestas de las solicitudes de datos especificados?”.

Sin embargo, la inclusión en el sistema del uso de promesas para gestionar las respuestas de las solicitudes de datos especificados ya está divulgado en el documento D2 que se refiere a que en Javascript, las promesas son objetos que representan el resultado pendiente de una operación asincrónica. Puede usarlos para programar más actividades después de que la operación asíncrona haya finalizado proporcionando una devolución de llamada (Párrafo 1).

En consecuencia, la persona del oficio normalmente versada en la materia incluiría el en el sistema el uso de promesas para gestionar las respuestas de las solicitudes de datos especificados mencionado en documento D2 en el sistema divulgado en el documento D1 para llegar así al objeto de la reivindicación 11 de la solicitud. Por lo cual se considera obvia.

REPÚBLICA DE COLOMBIA
SUPERINTENDENCIA DE INDUSTRIA Y COMERCIO

Oficio N° 13891

Ref. Expediente N° NC2017/0003239

Asimismo, el documento D1 divulga las características de las reivindicaciones dependientes así:

- Reivindicación 12: Divido '*GetFilesAsync*' para mostrar que simplemente está llamando a un método dentro de *Task.Run*. Ese método necesita tener *async* como prefijo para saber que lo quieres en el 'segundo plano'. '*addfile*' es la acción que agrega archivos mientras *_cancel* no está activado, como puede ver, no necesito ninguna lógica especial para saber el valor de cancelar (Sección The heart).
- Reivindicación 13: El modelo de vista implementa una versión personalizada de *INotifyPropertyChanged* para ser asincrónico y actualiza la UI (Interfaz de Usuario) cuando es capaz (Sección ViewModel). El modelo es muy simple, por lo general prefiero agregar un presentador para la vista en lugar de agregar un convertidor, use la opción que desee. Tenga en cuenta que en W8RP el formato de fecha ha cambiado a *GetDateTimeFormats* (Sección Model).

Las reivindicaciones dependientes 12 y 13 no mencionan alguna característica que, junto con las características de la reivindicación independiente, aporte nivel inventivo al sistema de la reivindicación 11, por lo cual se considera que no tienen nivel inventivo.

La reivindicación 17 consiste en: “*Un método para recibir, en un modelo de vista, una solicitud de datos específicos de la vista de una vista que comprende: (...)*” (Radicado N° NC2017/0003239 del 8 de octubre de 2018).

Tabla 3. Comparación de las características técnicas esenciales, con las del estado de la técnica:

Características esenciales	D1	D2
Recibir, en un modelo de vista, una solicitud de datos específicos de la vista de una vista.	Finalmente llegamos al XAML, he creado una página simple con los botones de comando, <i>GridView</i> y <i>TextBlock</i> para el elemento seleccionado. Cuida lo siguiente: - Creo una instancia de <i>ViewModel</i> en <i>Page.DataContext</i>. - El primer botón vincula el comando con '<i>GetFilesCommand</i>' y enlaza <i>IsEnabled</i> con <i>IsIdle (...)</i> (Sección View).	No se menciona.

**REPÚBLICA DE COLOMBIA
SUPERINTENDENCIA DE INDUSTRIA Y COMERCIO**

Oficio N° 13891

Ref. Expediente N° NC2017/0003239

	(...) El comando 'GetFilesCommand' para leer archivos (...) (Sección Properties and Fields).	
Comunicar, mediante una vista a un modelo de vista, una solicitud de datos específicos de la vista asociados con la vista.	El 'GetFilesCommand' ha implementado una acción con el prefijo async (lo descubrí por prueba y error, no lo encontré en ningún artículo) y en el interior llamo a 'GetFilesAsync' para obtener los archivos como pueda la aplicación (Sección Constructor).	No se menciona.
Recibir (802), en el modelo de vista, la solicitud de una vista para los datos específicos de la vista.	Divido 'GetFilesAsync' para mostrar que simplemente está llamando a un método dentro de <i>Task.Run</i> . Ese método necesita tener <i>async</i> como prefijo para saber que lo quieres en el 'segundo plano'. 'addfile' es la acción que agrega archivos mientras <i>_cancel</i> no está activado (...) (Sección The heart).	No se menciona.
Devolver (804) por medio del modelo de vista, una primera promesa a la vista para proporcionar los datos específicos de la vista.	No se menciona.	En Javascript, las promesas son objetos que representan el resultado pendiente de una operación asincrónica. Puede usarlos para programar más actividades después de que la operación asíncrona haya finalizado proporcionando una devolución de llamada (Párrafo 1).
Realizar, a través de la vista, antes de que la vista reciba los datos específicos de la vista, una tarea relacionada con la visualización de la vista.	Finalmente llegamos al XAML, he creado una página simple con los botones de comando, GridView y TextBlock para el elemento seleccionado. Tenga cuidado de lo siguiente: - (...) El TextBlock enlazar texto a CurrentFile - GridView vincula ItemsSource con 'Files' y SelectedItem con el modo CurrentFile TwoWay (Sección View). El modelo de vista implementa una versión personalizada de <i>INotifyPropertyChanged</i> para ser asincrónico y actualiza la UI (Interfaz de Usuario) cuando es capaz (Sección ViewModel).	No se menciona.

**REPÚBLICA DE COLOMBIA
SUPERINTENDENCIA DE INDUSTRIA Y COMERCIO**

Oficio N° 13891

Ref. Expediente N° NC2017/0003239

Hacer (810) por medio del modelo de vista, una solicitud asíncrona a un modelo de datos para información correspondiente a los datos específicos de la vista.	El modelo de vista implementa una versión personalizada de <i>INotifyPropertyChanged</i> para ser asincrónico y actualiza la UI (Interfaz de Usuario) cuando es capaz (Sección ViewModel). Divido ' <i>GetFilesAsync</i> ' para mostrar que simplemente está llamando a un método dentro de <i>Task.Run</i> . Ese método necesita tener <i>async</i> como prefijo para saber que lo quieres en el 'segundo plano'. ' <i>addfile</i> ' es la acción que agrega archivos mientras <i>_cancel</i> no está activado, como puede ver, no necesito ninguna lógica especial para saber el valor de cancelar (Sección The heart).	No se menciona.
Recibir (812) por medio del modelo de vista, una segunda promesa del modelo de datos para proveer la información.	No se menciona.	En Javascript, las promesas son objetos que representan el resultado pendiente de una operación asincrónica. Puede usarlos para programar más actividades después de que la operación asíncrona haya finalizado proporcionando una devolución de llamada (Párrafo 1).
Hacer (1010) por medio del modelo de vista, una solicitud asíncrona a una fuente de datos externa al dispositivo para un conjunto de datos correspondientes a la información.	Divido ' <i>GetFilesAsync</i> ' para mostrar que simplemente está llamando a un método dentro de <i>Task.Run</i> . Ese método necesita tener <i>async</i> como prefijo para saber que lo quieres en el 'segundo plano'. ' <i>addfile</i> ' es la acción que agrega archivos mientras <i>_cancel</i> no está activado, como puede ver, no necesito ninguna lógica especial para saber el valor de cancelar (Sección The heart).	No se menciona.
Obtener (1018) por medio del modelo de vista, el conjunto de datos en respuesta a la solicitud a la fuente de datos, el conjunto de datos que incluye la información.	El comando ' <i>GetFilesCommand</i> ' para leer archivos (...) (Sección Properties and Fields).	No se menciona.
Devolver (1022) por medio del modelo de	El modelo de vista implementa una versión personalizada de <i>INotifyPropertyChanged</i>	No se menciona.

**REPÚBLICA DE COLOMBIA
SUPERINTENDENCIA DE INDUSTRIA Y COMERCIO**

Oficio N° 13891

Ref. Expediente N° NC2017/0003239

vista, la información al modelo de vista para cumplir con la segunda promesa.	para ser asincrónico y actualiza la UI (Interfaz de Usuario) cuando es capaz (Sección ViewModel). Para actualizar la UI cuando usa asincrónico, debe usar el despachador de ventana actual. Como puede ver, creo un método asíncrono que actualiza la UI llamada <i>'UIThreadAction'</i> y varios métodos como <i>'PropertyChangedAsync'</i> que llama <i>'PropertyChanged'</i> dentro de la acción del despachador (Sección Implementing INotifyPropertyChanged).	
Devolver (904) por medio del modelo de vista, los datos específicos de la vista, basándose en la información devuelta al modelo de vista, a la vista para cumplir la primera promesa.		No se menciona.

El documento D1 se considera el estado de la técnica más cercano a la invención definida en la reivindicación 17 porque divulga un ejemplo del uso de MVVM con Windows 8 Modern UI con enlace de datos, comandos y propiedades utilizando la metodología asincrónica y manteniendo el modo de diseño con plantillas de datos (Sección Introducción).

La diferencia entre la invención, definida en la reivindicación 17 y el método de D1 consiste en que en el método de la reivindicación 17 incluye el uso de promesas en el modelo de vista.

El efecto que logra el uso de promesas es gestionar las respuestas de las solicitudes de datos especificados.

Por lo tanto, el problema técnico objetivo que pretende resolver esta invención se puede formular así: “¿cómo modificar el método conocido en D1 para gestionar las respuestas de las solicitudes de datos especificados?”.

Sin embargo, el uso de promesas para gestionar las respuestas de las solicitudes de datos especificados ya está divulgado en el documento D2 que se refiere a que en Javascript, las promesas son objetos que representan el resultado pendiente de una operación asincrónica. Puede usarlos para programar más actividades después de que la operación asíncrona haya finalizado proporcionando una devolución de llamada (Párrafo 1).

En consecuencia, la persona del oficio normalmente versada en la materia incluiría el uso de promesas para gestionar las respuestas de las solicitudes de datos especificados mencionado en documento D2 en el método divulgado en el documento

**REPÚBLICA DE COLOMBIA
SUPERINTENDENCIA DE INDUSTRIA Y COMERCIO**

Oficio N° 13891

Ref. Expediente N° NC2017/0003239

D1 para llegar así al objeto de la reivindicación 17 de la solicitud. Por lo cual se considera obvia.

Asimismo, los documentos D1 y D2 divulgan las características de las reivindicaciones dependientes así:

- Reivindicación 18: El comando '*GetFilesCommand*' para leer archivos (...) (D1, Sección Properties and Fields). En Javascript, las promesas son objetos que representan el resultado pendiente de una operación asincrónica. Puede usarlos para programar más actividades después de que la operación asíncrona haya finalizado proporcionando una devolución de llamada (D2, Párrafo 1).
- Reivindicación 19: El modelo de vista implementa una versión personalizada de *INotifyPropertyChanged* para ser asincrónico y actualiza la UI (Interfaz de Usuario) cuando es capaz (D1, Sección ViewModel).

Las reivindicaciones dependientes 18 y 19 no mencionan alguna característica que, junto con las características de la reivindicación independiente, aporte nivel inventivo al método de la reivindicación 17, por lo cual se considera que no tienen nivel inventivo.

7. Conclusión

Los requisitos de Patentabilidad de la presente solicitud están afectados así:

Ítem	Documento	Reivindicaciones afectadas	Requisito que afecta
D1	Artículo	1, 11 a 13 y 17 a 19	Nivel Inventivo
D2	Artículo	1, 3, 9 y 17 a 18	Nivel Inventivo

8. Respuesta a los argumentos del señor solicitante

El solicitante expone que "(...) D1 dice: *“El modelo de vista implementa una versión personalizada de *INotifyPropertyChanged* para ser asíncrono y actualiza la interfaz de usuario cuando es capaz”*, y simplemente describe un proceso de actualización asíncrono genérico donde la interfaz de usuario se actualiza después de que se completa la operación asíncrona (...)", además menciona que "(...) D2 dice: *“Puede*

Página 15 de 17

**REPÚBLICA DE COLOMBIA
SUPERINTENDENCIA DE INDUSTRIA Y COMERCIO**

Oficio N° 13891

Ref. Expediente N° NC2017/0003239

usar estos para programar una actividad adicional después de que la operación asíncrona haya finalizado mediante el suministro de una devolución de llamada” (...)” y afirma que “(...) En total contraste con D1 y D2, los elementos de las reivindicaciones citadas permiten claramente que la vista realice de forma sincrónicamente una tarea relacionada con la visualización de la vista antes de que la vista reciba los datos específicos de la vista en respuesta a la primera promesa asociada con la operación asíncrona (...)”.

Es oportuno aclarar que, la realización de síncrona de una tarea relacionada con la visualización de la vista antes que esta reciba los datos específicos en respuesta a la primera promesa de la operación asíncrona no se encuentra divulgada en el capítulo reivindicatorio allegado y por lo tanto no forma parte del estudio de nivel inventivo realizado, este capítulo únicamente se refiere a la realización de la tarea sin especificar la forma en la cual se lleva a cabo por lo que el documento D1 anticipa dicha característica al mencionar la actualización de la visualización de la interfaz de usuario después de la realización o identificación de un cambio en las propiedades, adicionalmente, el documento D2 se refiere a la realización de operaciones asíncronas para completar o devolver la promesa establecida con los datos requeridos por lo que dicho documento en combinación con las enseñanzas del documento D1 revela la característica de utilizar promesas para devolver la información requerida asíncrónicamente afectando así el nivel inventivo de la solicitud.

En cumplimiento del artículo 45, de la Decisión 486 de la Comisión de la Comunidad Andina, se le indica que debe dar respuesta dentro del término de sesenta (60) días contados a partir de la fecha de notificación de la presente comunicación. En caso de no dar respuesta al presente requerimiento, o si a pesar de la respuesta subsistieran los impedimentos para la concesión, se denegará la patente.

Si como respuesta a este requerimiento el solicitante llegara a modificar: i) el capítulo reivindicatorio, ii) la descripción de la invención a proteger, iii) o presente un nuevo capítulo reivindicatorio, y desee que se realice un nuevo examen de patentabilidad a esta modificación, no deberá pagar la tasa de modificación voluntaria pero sí la correspondiente a la tasa de examen de patentabilidad. En todo caso, el número de requerimientos por no patentabilidad estará supeditado a que el término del estudio de patentabilidad no sea superior a 18 meses, contados desde la publicación de la solicitud de patente de invención. Adicionalmente, se le recuerda al solicitante que si como respuesta a este requerimiento llegara a modificar el capítulo reivindicatorio y

**REPÚBLICA DE COLOMBIA
SUPERINTENDENCIA DE INDUSTRIA Y COMERCIO**

Oficio N° **13891**

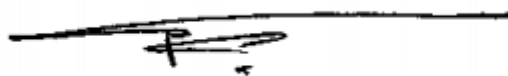
Ref. Expediente N° NC2017/0003239

éste contenga más de diez (10) reivindicaciones deberá pagar la tasa por reivindicación adicional.

Se invita al solicitante se sirva indicar, si así lo considera, su renuncia al término faltante para los sesenta (60) días.

Notifíquese el presente oficio conforme a lo dispuesto en el numeral 6.2 del Capítulo Sexto del Título I de la Circular Única, informándoles que contra el mismo no procede el recurso de la vía gubernativa.

Atentamente,



**JOSE LUIS SALAZAR LOPEZ
DIRECTOR DE NUEVAS CREACIONES**

Elaborado: CARLOS EDUARDO GUEVARA NICHÓY
Revisado: SONIA ANDREA DÍAZ PIRAZÁN