

Plataforma de Firma Digital para Aplicaciones Web del Poder Judicial de Río Negro

Javier Villanueva¹ Enrique Molinari²

¹ Poder Judicial de Río Negro y Universidad Nacional de Río Negro, Argentina
jvillanueva@jusrionegro.gov.ar

² Poder Judicial de Río Negro y Universidad Nacional de Río Negro, Argentina
emolinari@unrn.edu.ar

Resumen. En la actualidad, existen muchas alternativas para firmar digitalmente documentos con un dispositivo criptográfico. Sin embargo, cuando integramos un software firmador en una aplicación web, el resultado puede ser una experiencia poco amigable para el usuario. Algunas de las pocas alternativas que existían, como los Applets de Java, fueron discontinuadas. La iniciativa impulsada por las autoridades del Poder Judicial de Río Negro de informatizar y digitalizar en su totalidad los procesos judiciales, hizo imperativo contar con una solución de firma digital que permita a jueces y funcionarios, firmar de forma rápida y amigable. En este artículo se analizan diferentes ventajas y desventajas de los enfoques existentes, los problemas necesarios a resolver y se describe en detalle la solución implementada en el Poder Judicial de Río Negro que permite a sus funcionarios firmar digitalmente cientos de actos procesales diarios de manera simple e integrada a su sistema de gestión de expedientes.

Keywords: Firma Digital, Token, Aplicación Web, Firmador Java, PKCS#11

1 Introducción

Jueces y funcionarias del Poder Judicial de Río Negro utilizan la firma digital a través de dispositivos criptográficos (Tokens) desde hace varios años, principalmente para firmar resoluciones importantes y sentencias. Sin embargo, en los procesos judiciales los organismos, a través de sus funcionarios y magistrados, requieren la firma de más de cien actos procesales diarios. La decisión impulsada por las autoridades del Poder Judicial de informatizar completamente los procesos judiciales, requirió del desarrollo del software de gestión de expedientes y demandó la implementación de una solución de firma digital con las siguientes características:

1. Integrada de forma transparente al software de gestión web del Proceso Judicial.
2. Simple de utilizar, es decir, que no requiera más que una selección múltiple de actos procesales a firmar y el ingreso del PIN de seguridad del Token.
3. Permitir firmar cientos de actos procesales a la vez en pocos segundos (ingresando el PIN del Token solo una vez).
4. Permitir que los actos procesales sean firmados por múltiples funcionarios.

5. Dejar visible, nombres, apellidos y fecha de firma de los funcionarios judiciales firmantes.
6. Cada acto procesal es un documento PDF.

En este artículo, se describe una solución para implementar un firmador con dichas características. La dificultad más grande recae en encontrar la forma de integrar una aplicación Web con el uso de un dispositivo como los Tokens. El Token y sus drivers corren en la computadora (PC) de un usuario, mientras que la aplicación Web corre una gran parte en uno o varios servers, y otra parte en el navegador de internet (Browser). Los Browsers corren en la PC de un usuario, sin embargo, por cuestiones de seguridad, no hay posibilidad de acceder a un dispositivo conectado a dicha PC. Actualmente, todos los navegadores de internet implementan mecanismos de seguridad que prohíben el acceso a cualquier dispositivo de hardware conectado a la computadora donde la aplicación Web está siendo utilizada. De acuerdo con lo expuesto por Ferri et al. (2010 pp. 1, 6) [1] los navegadores web deben instanciarse dentro de un ambiente virtual controlado denominado sandbox, cuyo espacio de memoria y recursos son designados de manera restringida. De tal forma, no comprometemos al sistema anfitrión durante la navegación por sitios web riesgosos. Esto se logra abriendo los hipervínculos en sesiones sandbox que virtualizan algunos recursos para que más tarde, al cerrar la sesión, los cambios efectuados dentro de dicho ambiente no persistan en el sistema anfitrión, aunque, por cierto, el usuario puede llegar a configurar cierto grado de persistencia. Este mecanismo de seguridad es implementado actualmente por todos los browsers modernos como Google Chrome, Mozilla Firefox, Microsoft Edge y Chromium.

2 Alternativas existentes

2.1 Aplicaciones de Escritorio

Las aplicaciones de escritorio no poseen las restricciones de seguridad que imponen los navegadores de internet. Sin embargo, tienen la desventaja de que no suelen integrarse del todo bien a las aplicaciones web. Generalmente el usuario tiene que descargar los documentos desde la aplicación web, firmarlos y luego volver a subirlos a ella. Dicha manera de operar es incompatible con los requerimientos expuestos anteriormente.

Existen algunas herramientas como XolidoSign [2] que en su versión gratuita permite firmar múltiples documentos a la vez. Adicionalmente, ofrece versiones pagas que proveen distintas funcionalidades que incluyen integración con aplicaciones web, sin embargo, la firma debe ser a través de certificados digitales y no permite el uso de dispositivos criptográficos como los tokens.

2.2 Applets de Java.

Un applet de Java es un componente escrito en Java que se ejecuta sobre un contenedor dentro del navegador web. Por ejemplo, SIU-Toba [3] es un applet muy conocido

que permite firmar digitalmente y pertenece al Sistema de Información Universitaria integrado por Universidades Nacionales Públicas. Entre sus ventajas podemos mencionar que es open-source, funciona con tokens y permite firmar múltiples documentos a la vez. Sin embargo, la desventaja de este enfoque reside en que, para que el applet se ejecute, es necesario que el navegador tenga instalado el Plugin de Java. Pero sucede que dicho plugin funciona sobre una arquitectura NPAPI (Netscape Plugin Application Programming Interface) que los browsers han dejado de dar soporte para unificar las funcionalidades entre las versiones desktop y mobile [4]. Sin ir más lejos, el soporte que provee Mozilla Firefox es muy limitado para las versiones de 32 bits y lo abandonó para las de 64 bits, por otro lado, Google Chrome finalizó el soporte de esta tecnología en el año 2015. Finalmente, Oracle discontinuó los Applets desde Java SE9 [5].

2.3 Extensiones/Addons de navegadores.

Otra alternativa posible es desarrollar una extensión, también conocida como complemento o Add-On, para algún browser específico como Mozilla Firefox o Chrome. La extensión, una vez instalada en el navegador, hace de nexo entre la sesión de la aplicación web y el entorno del sistema operativo, logrando el acceso al dispositivo de firma digital. Si bien la firma se produce de manera casi directa, la desventaja está en que los usuarios del sistema quedan limitados a usar siempre un mismo navegador. O nos obliga a desarrollar el mismo Add-On para todos los proveedores de navegadores desde donde se acceda al sistema.

2.4 Firmar en el Servidor.

Existe la posibilidad también de realizar la firma en el servidor (en lugar de realizarla en el cliente con Token). Esto es posible únicamente si la clave privada de los firmantes es almacenada en el servidor. Esta posibilidad la tiene el estado nacional y por ello la Jefatura de Gabinete de Ministros implementó la Plataforma de Firma Digital Remota (PFDR) [6]. La PFDR permite al usuario subir documentos en PDF y firmarlos digitalmente allí, ingresando un OTP (One Time Password) como mecanismo adicional de seguridad que el sitio envía al teléfono del firmante para asegurar la autenticidad, y luego ingresar el PIN. La PFDR, a través de un conjunto de APIs, permite una buena integración con aplicaciones web de terceros, pero actualmente con la limitación de firmar de a un documento a la vez. A su vez, como mencionamos, era necesario firmar cien documentos PDF en pocos segundos, lo que no es factible de hacer con una solución remota.

3 Plataforma de Firma Digital

Entre las diferentes opciones mencionadas de firma, se optó por desarrollar una aplicación de escritorio para poder firmar con los Tokens que ya eran utilizados por fun-

cionarios y magistrados en el Poder Judicial de Río Negro. Sin embargo, para tener una integración transparente con el sistema de gestión de expedientes judiciales, fue necesario implementar varios elementos de software adicionales.

Para describir y comunicar la arquitectura del Sistema de Gestión de Expedientes Judiciales y su integración con la Plataforma de Firma Digital se utilizarán los modelos propuestos por Simon Brown denominados C4 Model [7]. En concreto, utilizaremos el modelo de contenedores y el de componentes. Los contenedores (no confundir con Docker) del modelo C4 se refieren a aplicaciones (Web, móviles, single-page, de escritorio, shell scripts, etc), repositorios de datos, esquemas y bases de datos de cualquier tipo, relacionales o no. Esencialmente es un elemento de software que ejecuta código o almacena datos. Con componente, el modelo C4 se refiere a un conjunto de funcionalidades encapsuladas y expuestas a través de una interfaz bien definida. Éstos deben mapear a una estructura sintáctica como un package, espacio de nombres o módulos. El diagrama de componentes es un nivel de detalle superior al de contenedores. Dentro de un contenedor, encontraremos uno o varios componentes.

3.1 Sistema de Gestión de Expedientes Judiciales - Diagrama de contenedores

Inicialmente, se plantea un diagrama de contenedores para describir la plataforma de manera más abstracta. El siguiente modelo está compuesto por cuatro contenedores: el Sistema de Gestión de Expedientes Judiciales (Sistema Puma), el Firmador, el Módulo Gestor de Archivos Temporales y el Web Browser. A continuación, se describen dichos contenedores y sus relaciones.

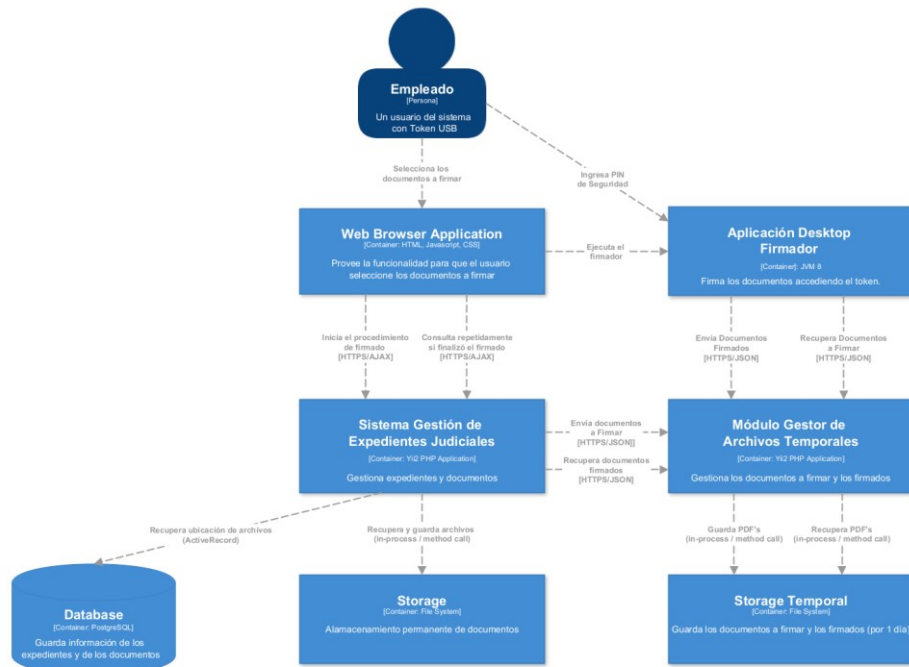


Fig. 1. Diagrama de Contenedores del Sistema de Expedientes Judiciales integrado a la Plataforma de Firma Digital

La persona, el empleado del poder Judicial, desde su navegador de internet, accede a la aplicación web Sistema de Gestión de Expedientes Judiciales para trabajar con los expedientes. Desde allí, el usuario selecciona los actos procesales a firmar, luego cuando presiona el botón de firmar, el navegador llama al sistema de gestión de expedientes para que inicie el proceso de firmado. El navegador, además, inicia la ejecución de la aplicación desktop firmadora Java como se verá en detalle más adelante.

El sistema de gestión de expedientes se comunica con el Módulo Gestor de Archivos Temporales (MGDT) para enviarle los actos procesales en formato PDF a firmar y luego para recuperarlos firmados. Adicionalmente avisará al navegador cuando el proceso de firmado haya finalizado.

El Módulo Gestor de Archivos Temporales es una aplicación de apoyo. Se ejecuta en un contenedor separado del sistema de software de gestión en un mismo server o diferente. Básicamente mantiene los actos procesales en PDF que se van a firmar y los que ya fueron firmados de manera temporal. El módulo implementa una API que es utilizada por la aplicación desktop firmadora y por el sistema de gestión de expedientes.

El contenedor Aplicación Desktop representa a la aplicación firmadora Java. Se ejecuta en la PC del usuario y se encarga de descargar los actos procesales a firmar para luego enviárselos firmados al Módulo Gestor de Archivos. Adicionalmente, le

solicita al usuario que ingrese el PIN de seguridad del su Token para luego iniciar el proceso de firmado.

3.2 Sistema de Gestión de Expedientes Judiciales - Diagrama de componentes

Se presenta a continuación un diagrama de componentes para representar el diseño de la arquitectura anterior con mayor detalle. De este modo, el siguiente modelo muestra los componentes de cada uno de los contenedores que implementan la solución.

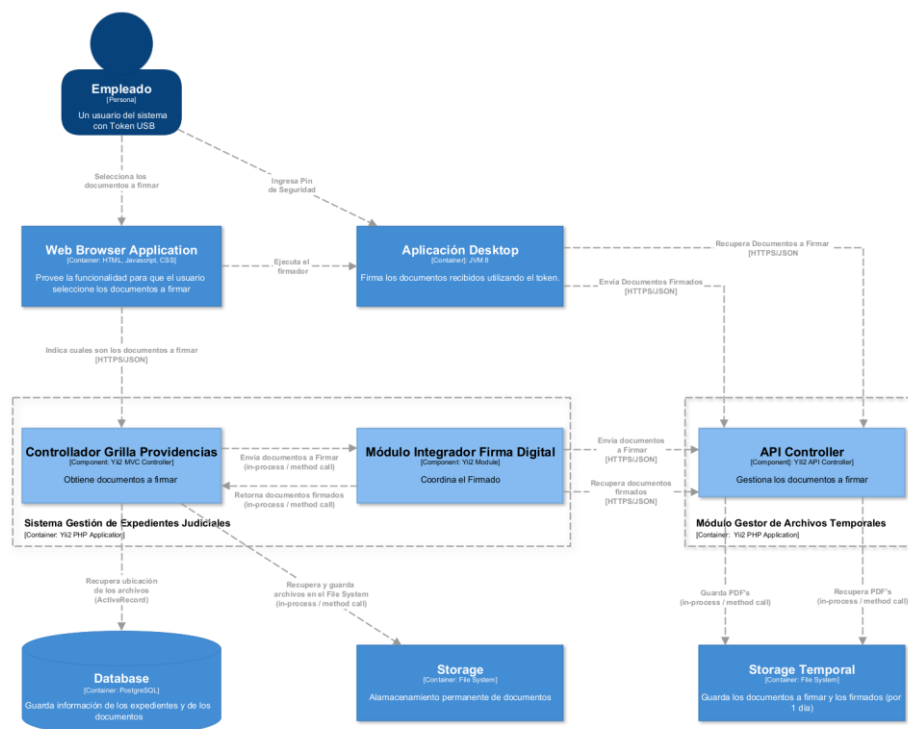


Fig. 2. Diagrama de Componentes de la Plataforma de Firma Digital para Aplicaciones Web

Sistema de Gestión de Expedientes Judiciales (PUMA). El Sistema de Gestión de Expedientes Judiciales, es una aplicación web desarrollada con el framework de PHP Yii2. El sistema recibe y maneja la acción de firmado que proviene del navegador, por medio de una llamada HTTP POST que se dispara cuando el empleado presiona el botón de firmar. Dicha llamada es manejada por un componente controlador específico encargado de iniciar el procedimiento de firmado. La aplicación web, desde el lado del servidor, debe coordinar y sincronizar algunas de las tareas que efectúan el módulo gestor de archivos, el firmador de escritorio y el navegador. La creación del Módulo Integrador de Firma Digital (MIFD) para Aplicaciones Web representa una

solución que permite extraer dicha funcionalidad e independizarla para poder reutilizarla en otras aplicaciones que requieran firma digital. Este módulo representa un componente integrador que se encarga de indicar qué hacer en cada momento y efectuar el vínculo con nuestra aplicación web.

El módulo es iniciado por el controlador del sistema ya mencionado. Primero, envía los actos procesales en PDF a firmar al módulo gestor de archivos. Luego, se queda esperando en un ciclo en el que repetidamente le consulta al MGDТ si recibió los documentos PDF ya firmados. Por último, cuando detecta que se cumple dicha condición, recupera los documentos firmados desde el módulo gestor de archivos temporales, se los entrega al controlador para que los persista y le avisa al browser que el proceso de firmado finalizó con éxito.

Módulo Gestor de Documentos Temporales. El MGDТ es un módulo independiente del sistema de gestión de expedientes web y de la aplicación de escritorio firmadora. Está desarrollado en PHP, también con el framework Yii2. Se encarga de manejar los documentos en PDF que se van a firmar y los documentos que ya fueron firmados. Expone una API (Application Programming Interface) REST (Representational State Transfer) que provee un servicio para gestionar el manejo de los documentos que necesitan ser firmados.

Este módulo debe disponer de su propio storage para persistir temporalmente los documentos. Para que los otros componentes persistan y recuperen documentos por medio llamadas API Rest, el módulo define una interfaz que implementa las siguientes operaciones:

- Guardar documentos para firmar: recibe los documentos que hay que firmar codificados en base64 y los almacena temporalmente en una carpeta específica del servidor para que luego se firmen.
- Guardar documentos firmados: recibe los documentos ya firmados y codificados en base64 y los almacena temporalmente en una carpeta específica del servidor para que luego la aplicación los recupere.
- Recuperar documentos firmados: recupera todos los documentos ya firmados que se encuentran en una carpeta específica codificados en base64.
- Recuperar documentos para la firma: recupera todos los documentos para firmar contenidos en una carpeta específica.
- Consultar si los documentos ya fueron firmados: determina si todos los documentos en PDF de una carpeta específica fueron firmados.
- Consultar si el firmador aún está firmando y no terminó con su tarea: determina si la aplicación Java leyó los archivos para firmar de una carpeta específica y aún están siendo firmados.
- Finalizar firmado: se utiliza para indicar que terminó con el proceso de firmado de todos los documentos incluidos en una carpeta específica.

Es importante resaltar que el almacenamiento de los documentos en este módulo es temporal, puesto que luego de que el proceso de firmado finaliza, los documentos procesados son almacenados por la aplicación web en su propio storage. Por tal moti-

vo dejan de ser necesarios en este módulo y deben ser eliminados gradualmente por un proceso batch diariamente.

Aplicación Desktop Java. El firmador es una aplicación de escritorio Java que corre sobre una Java Virtual Machine V8. Permite firmar documentos en PDF y para ello utiliza iText 7 Suit [8], una SDK (Software Development Kit) de biblioteca de PDF. La función principal de la aplicación es el firmado de los documentos que fueron seleccionados desde la aplicación web. Para efectuar dicha tarea primero le solicita al usuario el código PIN de seguridad para acceder su Token. Seguidamente, descarga en background desde el MGDТ los documentos a la computadora del cliente, luego inicia el proceso de firmado. Cuando termina de firmar, le devuelve los documentos firmados al MGDТ. Todo el proceso es automático y transparente para el usuario, sólo debe ingresar su PIN/clave para que inicie la tarea y luego esperar la finalización de toda la operación.

Acceso al token. Para que la aplicación Java pueda firmar, es necesario que acceda al token del usuario para recuperar su certificado y así poder firmar con su clave privada. La comunicación se logra utilizando el estándar PKCS#11 (Public-Key Cryptography Standards), que define una API multiplataforma para gestionar hardware criptográfico de seguridad. Comúnmente llamamos a estos dispositivos tokens criptográficos, o también "Cryptoki " (de Cryptographic Token Interface).

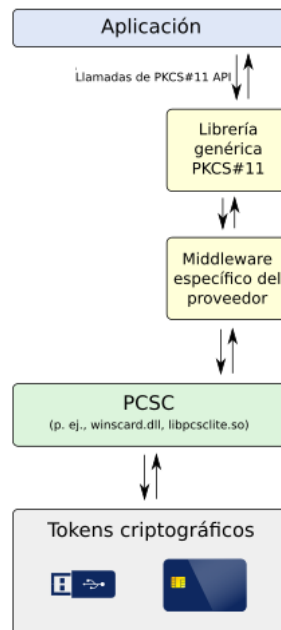


Fig. 3. Esquema genérico de la pila de software subyacente para la comunicación entre la aplicación y el hardware criptográfico [9].

En nuestro caso la aplicación es el firmador Java que efectúa las llamadas de firmado a través de una librería genérica capaz de manejar el hardware criptográfico de una gran variedad de proveedores. Específicamente, el firmador utiliza la librería `sunpkcs11.jar` (PKCS#11) de Sun que actúa de puente entre las API's de Java y la API PKCS#11 nativa del dispositivo. Dicha librería, se encarga de traducir las llamadas y las convenciones entre las API's de Java y la API del dispositivo. Por tal motivo, no implementa los algoritmos criptográficos en sí y, en consecuencia, requiere tener instalada una implementación de PKCS#11 en el sistema operativo a través de otra librería. Esta librería adicional puede ser una librería dinámica (.dll en Windows) o una librería de objetos compartidos (.so en Linux).

Además, es necesario disponer del software middleware provisto por el proveedor de nuestro token que incluye los drives necesarios para que la computadora detecte automáticamente al dispositivo criptográfico y pueda gestionar los certificados.

Por último, la implementación de PKCS#11 debe emplear PC/SC (Personal Computer/Smart Card) para poder comunicarse con el hardware criptográfico. PC/SC es un estándar para interactuar con lectores de tarjetas inteligentes en entornos informáticos. Esta interacción es llevada a cabo por librerías provistas por el proveedor del token.

4 La Plataforma de Firma Digital Web en funcionamiento

4.1 Modelo Dinámico para el firmado

Partiendo de la estructura estática definida en el modelo anterior, se muestra a continuación un modelo dinámico que desarrolla la secuencia de pasos que intervienen en runtime durante el firmado desde el momento en que el usuario, desde su navegador, selecciona los documentos a firmar y luego presiona el botón de firmar:

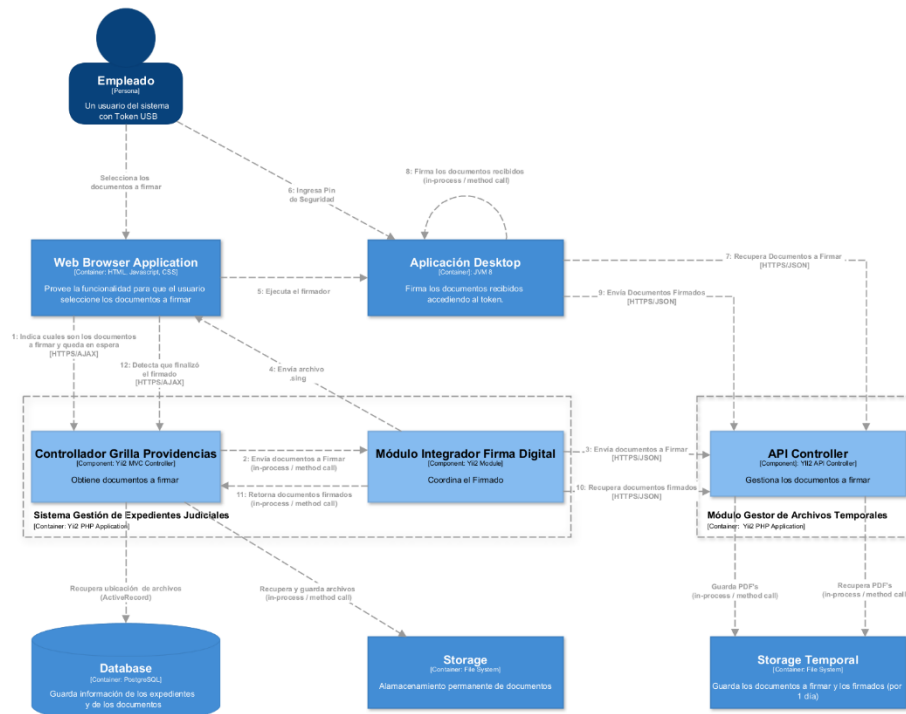


Fig. 4. Diagrama dinámico que resume el procedimiento de firmado

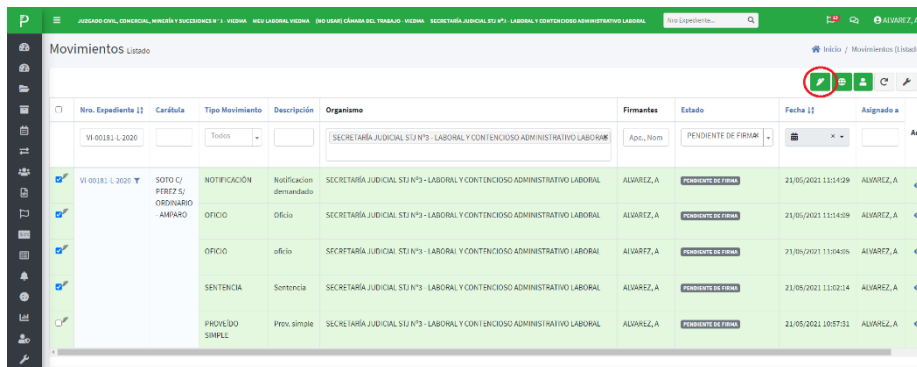
A continuación, se explican en detalle los pasos del modelo de la figura anterior:

1. El navegador llama al sistema de gestión de expedientes para iniciar el procedimiento de firma. La llamada es manejada por un controlador específico que recibe además los identificadores de los documentos a firmar. La aplicación web, del lado del cliente, muestra una señal de espera (con un spinner), mientras consulta repetidamente al sistema hasta que detecta que finalizó el procedimiento de firmado.
2. El controlador recupera los documentos a firmar del storage y luego inicia al Módulo Integrador de Firma Digital (MIFD) para delegarle el control del proceso.
3. El módulo MIFD integrador envía los documentos a firmar al Módulo Gestor de Archivos para que los guarde en una carpeta específica que representa esta instancia de firmado en particular.
4. El módulo MIFD genera un archivo con extensión *.sing* con la información de cuáles son los documentos para firmar y genera la descarga del mismo en el navegador del usuario. Luego el módulo queda a la espera de que termine el proceso de firmado.
5. La PC del usuario, previamente, tiene instalada la aplicación de escritorio firmadora y asociada ésta a los archivos *.sign*. El navegador al descargar el archivo con extensión *.sing*, abre automáticamente la aplicación desktop firmadora Java.

6. El firmador Java solicita el pin/clave al usuario para luego acceder al token.
7. El firmador lee del archivo *.sing* cuáles son los documentos que hay que firmar y los recupera desde el módulo gestor de archivos temporales por medio de un llamado por API REST.
8. El firmador accede al token y firma digitalmente todos los documentos PDF.
9. El Firmador envía los documentos PDF firmados al módulo gestor de archivos indicando que terminó el proceso de firmado por medio de una llamada a su API REST.
10. Cuando el módulo integrador MIFD detecta que el módulo gestor de archivos temporales ya tiene todos los documentos firmados, finaliza su espera y los recupera del MGD.
11. El módulo integrador entrega los documentos firmados al controlador del sistema de gestión de expedientes para que los persista.
12. El navegador detecta que el proceso de firmado terminó y muestra un mensaje de éxito al usuario.

4.2 Ejemplo de firmando

La Plataforma de Firma Digital para Aplicaciones Web comenzó a utilizarse por empleados, secretarios y jueces (del fuero laboral) del Poder Judicial de Río Negro junto con la puesta en producción del Sistema de Gestión de Expedientes Judiciales en marzo del año 2021. A continuación, se detalla un ejemplo en donde el empleado firma cuatro documentos de diferentes actos procesales de un expediente judicial. Inicialmente el usuario accede a una grilla de movimientos/actos procesales y filtra por aquellos que están en estado pendiente de firma. Luego, selecciona los que desea firmar y presiona el botón de firmado indicado por un círculo rojo (ver Fig. 5).



Nro. Expediente	Carátula	Tipo Movimiento	Descripción	Organismo	Firmantes	Estado	Fecha	Asignado a
VI-00181-L-2020	SOTO C/ PEREZ S/ ORDINARIO AMBARGO	NOTIFICACIÓN	Notificación demandado	SECRETARÍA JUDICIAL STJ N°9 - LABORAL Y CONTENCIOSO ADMINISTRATIVO LABORAL	ALVAREZ, A	PENDIENTE DE FIRMA	21/05/2021 11:14:28	ALVAREZ, A
		OFICIO	Oficio	SECRETARÍA JUDICIAL STJ N°9 - LABORAL Y CONTENCIOSO ADMINISTRATIVO LABORAL	ALVAREZ, A	PENDIENTE DE FIRMA	21/05/2021 11:14:59	ALVAREZ, A
		OFICIO	Oficio	SECRETARÍA JUDICIAL STJ N°9 - LABORAL Y CONTENCIOSO ADMINISTRATIVO LABORAL	ALVAREZ, A	PENDIENTE DE FIRMA	21/05/2021 11:14:59	ALVAREZ, A
		SENTENCIA	Sentencia	SECRETARÍA JUDICIAL STJ N°9 - LABORAL Y CONTENCIOSO ADMINISTRATIVO LABORAL	ALVAREZ, A	PENDIENTE DE FIRMA	21/05/2021 11:15:14	ALVAREZ, A
		PROVELO SIMPLE	Prov. simple	SECRETARÍA JUDICIAL STJ N°9 - LABORAL Y CONTENCIOSO ADMINISTRATIVO LABORAL	ALVAREZ, A	PENDIENTE DE FIRMA	21/05/2021 10:57:31	ALVAREZ, A

Fig. 5. El empleado selecciona los actos procesales a firmar desde la grilla de movimientos.

Una vez accionado el botón de inicio de firma, el sistema le pide al empleado que ingrese el PIN de seguridad del Token USB que debe tener conectado a su computadora. Como se muestra a continuación.

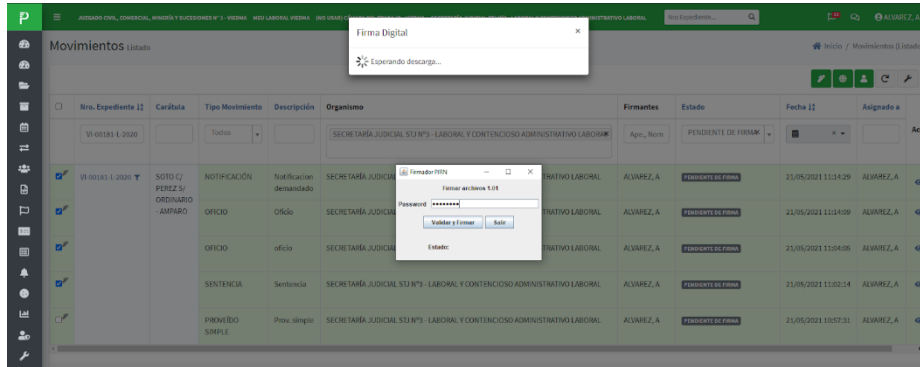


Fig. 6. El empleado ingresa su PIN de seguridad

Luego, el sistema recupera los archivos a firmar, accede al token y comienza a firmarlos digitalmente.

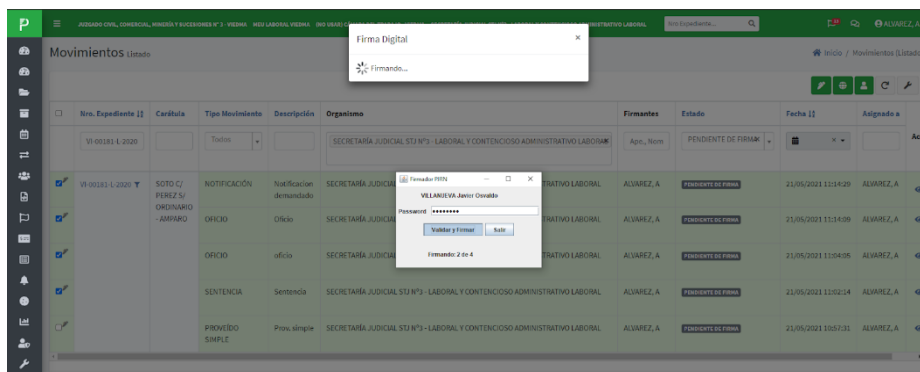


Fig. 7. El firmador inicia el firmado de los cuatro documentos.

Finalmente, cuando el sistema de gestión de expedientes recuperó todos los documentos ya firmados, si todo salió bien, informa que finalizó con éxito. Por cierto, los documentos de los actos procesales ya firmados por el empleado, pueden ser firmados por otros funcionarios y/o jueces. Es decir, al firmar un documento, éste no pierde las firmas previas.

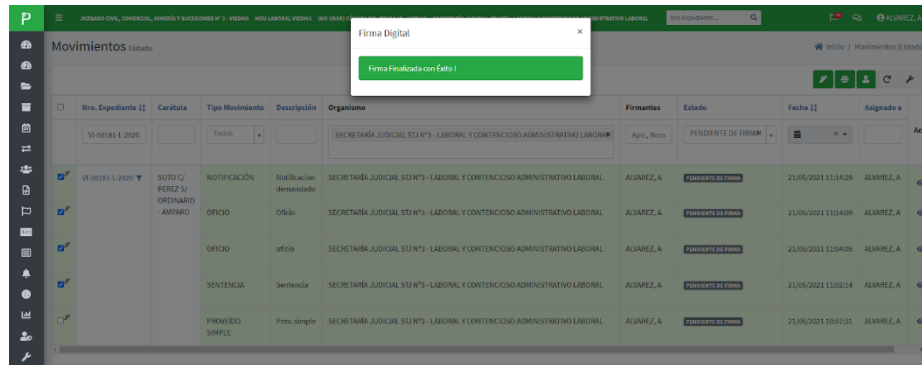


Fig. 8. El sistema informa que el firmado finalizó con éxito.

5 Conclusión

La tarea de firmar digitalmente desde una aplicación web con un dispositivo criptográfico no suele resultar una experiencia muy amigable para el usuario. En este artículo se revisaron las alternativas existentes y, si bien algunas se acercan en mayor o menor grado a resolver la necesidad de firmar de manera transparente múltiples documentos con un token, todas presentan alguna desventaja.

Adicionalmente, se analizó el motivo por el cual no es posible acceder al token del usuario desde su navegador web. Los browsers modernos renderizan las páginas web dentro de sandboxes que impiden acceder a los recursos y dispositivos del sistema por motivos de seguridad, protegiendo al usuario de posibles riesgos y ataques. En consecuencia, se mostró una alternativa que logra la comunicación entre la aplicación y el hardware criptográfico utilizando el estándar PKCS#11.

A partir de los análisis anteriores, se mostró el diseño y la arquitectura del Sistema de Gestión de Expedientes Judiciales integrada a la Plataforma de Firma Digital para Aplicaciones Web. Ambas soluciones fueron desarrolladas por empleados del Poder Judicial de Río Negro y puestas en producción en marzo del corriente año. Particularmente, la plataforma de firma digital fue diseñada con el objetivo de poder reutilizarla e integrarla desde cualquier aplicación Web, no solo desde dentro del Poder Judicial de Río Negro, sino que también, desde cualquier organismo público, privado o empresa.

El Poder Judicial de Río Negro actualmente gestiona en su totalidad los expedientes judiciales correspondientes al fuero Laboral y al fuero Originario, firmando alrededor de 1200 actos procesales diarios en toda la provincia, utilizando la solución descrita. Para el próximo año se provee incorporar a los demás fueros para finalmente realizar la gestión judicial despapelizada.

Referencias

1. Ferri, L., Pichetti, L., Secchi, M., & Secomandi, A. (2010). U.S. Patent Application No. 12/359,457. Sandbox Web Navigation.
2. XolidoSign, <https://bit.ly/3yFbL4G>, consultado en marzo 2021.
3. SIU-Toba Firmador, <https://bit.ly/3oL35VA>, consultado en marzo 2021.
4. Oracle. Migrating from Java Applets to plugin-free Java technologies [White paper]. (2016).
5. Oracle. Java Client Roadmap Update, [White paper]. (2020).
6. Plataforma de Firma Digital Remota, <https://bit.ly/3oYiPVD>, consultado en marzo 2021.
7. C4 Model Home Page, <https://bit.ly/3fLVRMS>, consultado en marzo 2021.
8. iText7. Addendum to Digital signatures for PDF documents, <https://bit.ly/3bRgx1P>, consultado en marzo 2021.
9. Microsom, <https://bit.ly/3fM5sDs>, consultado en marzo 2021.