manual integradores

guía de integración y casos de uso

viafirma documents

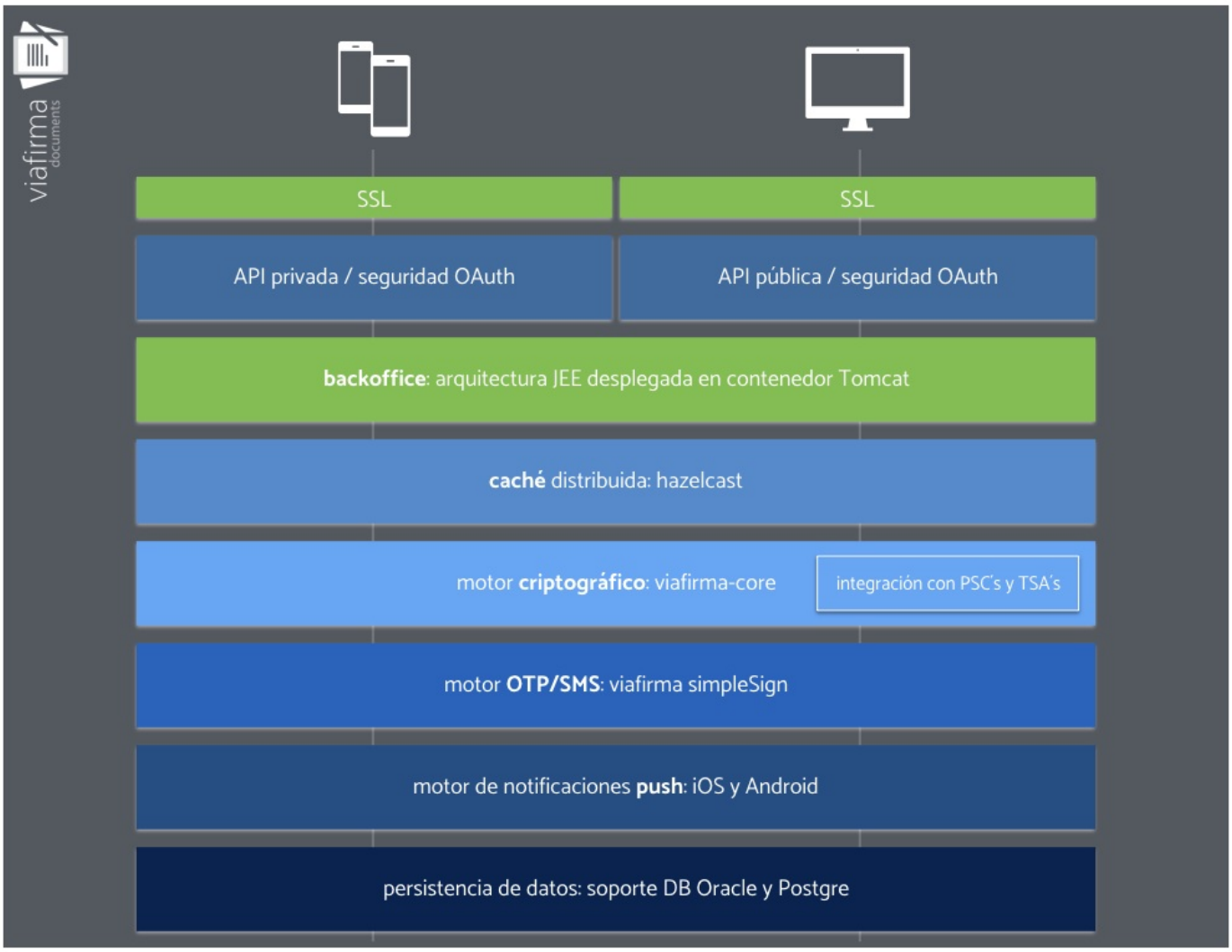
Tabla de contenido

Cómo Integrar con viafirma documents

A lo largo de los siguientes capítulos te explicaremos todo lo que necesitas saber para integrar tu sistema con los servicios ofrecidos por viafirma documents v3.5.

Si lo deseas puede descargar este manual en formato PDF aquí.

Diagrama de arquitectura.



Control de Cambios

Esta documentación técnica está sujeta a modificaciones diarias, y alguna información o configuración avanzada podría no estar reflejada. Consulte en cualquier caso con el equipo de soporte técnico.

- Fecha: 11-abril-17

Todos los servicios de integración pueden probarse en nuestro ambiente de sandbox. Si no tienes credenciales para desarrollador solicítalas en la cuenta comercial@viafirma.com

https://sandbox.viafirma.com/documents

Servicios ofrecidos por viafirma documents

Viafirma documents expone todo sus servicios públicos mediante una implementación Java de Jersey, y sobre una capa de securización OAuth, por lo que para empezar a integrar tu sistema con viafirma documents necesitarás unas credenciales.

Credenciales para Integración

Las credenciales gestionadas por viafirma documents para la integración de servicios esta compuestas por dos datos:

- OAuth Consumer Key
- OAuth Consumer Secret

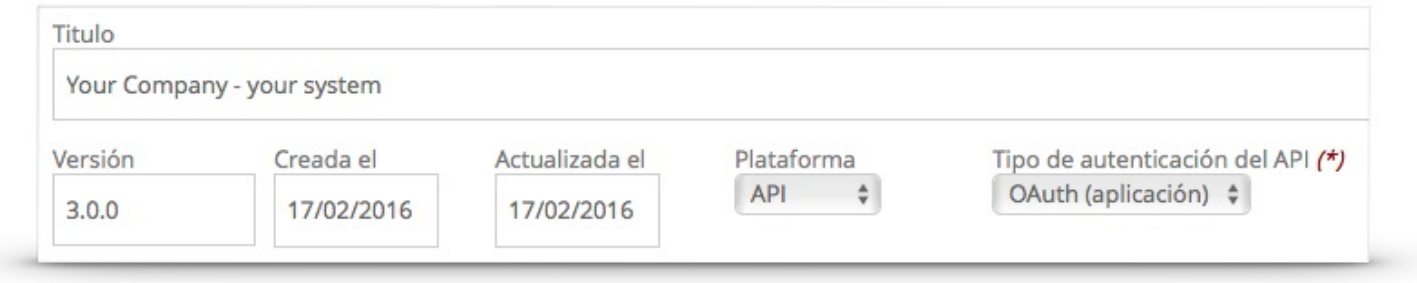
Plataformas soportadas

La credenciales podrán ser consumidas por dos tipos de plataformas:

- servidor (aplicación web o escritorio)
- móvil (app móvil)

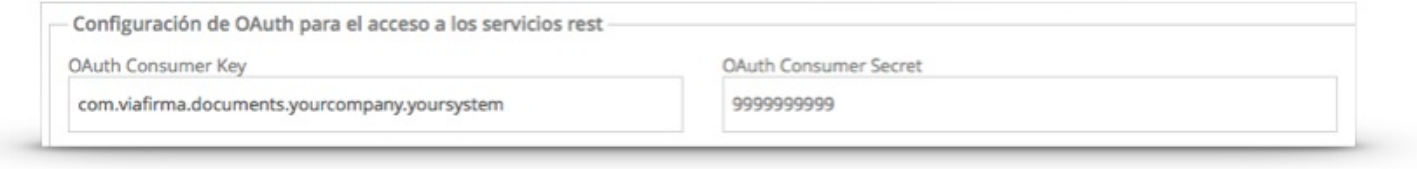
Credenciales para plataformas servidor o escritorio

Si tu sistema está basado en una aplicación web o escritorio, las credenciales que necesitas serán del tipo API.



Por ejemplo:

- OAuth Consumer Key = com.viafirma.documents.yourcompany.yoursystem
- OAuth Consumer Secret = 9999999999

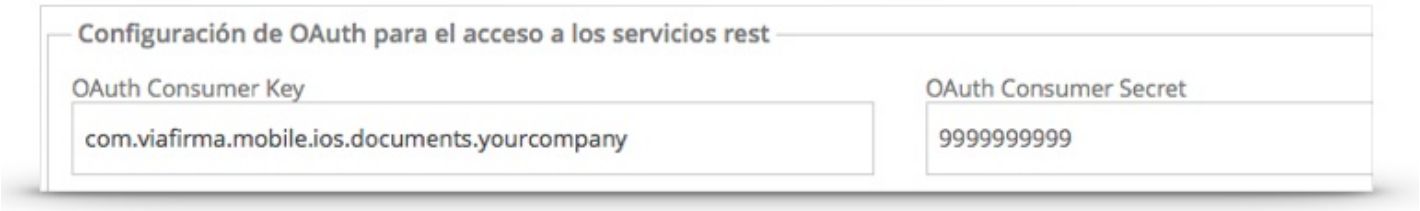


Credenciales apps móviles

Y por otro lado, se gestionarán las credenciales para las plataformas móviles, en concreto, *iOS y Android, que usarán credenciales de tipo User**.

Por ejemplo:

- OAuth Consumer Key = com.viafirma.mobile.ios.documents.yourcompany
- OAuth Consumer Secret = 9999999999



¿Cómo puedo probar los servicios?

Te facilitamos una interfaz de usuario, basada en Swagger, que te simplifica y facilita el consumo de los servicios expuestos por viafirma documents en tus pruebas.

Para acceder esta interfaz necesitarás además un usuario autorizado para acceder al backend de viafirma documents, con rol “developer”.

En tres sencillos pasos podrás revisar los servicios y consumirlos directamente desde la interfaz de usuario:

1. Accede al backend, por ejemplo https://sandbox.viafirma.com/documents
2. Usa tus credenciales de usuario (debes tener rol developer)
3. Accedes a la interfaz de swagger, por ejemplo: https://sandbox.viafirma.com/documents/api-docs/

Uso de la interfaz de usuario para testing

En esta interfaz te ayudaremos a probar los servicios expuestos por vafirma documents a sus integradores, diferenciando entre servicios públicos y servicios securizados para integradores.

Servicio público

Dispones de un método isAlive que podrás usar en tus monitorizaciones del servicio. No requiere el uso de credenciales.

Curl:

```
curl -X GET --header "Accept: application/json" "https://sandbox.viafirma.com/documents/api/v3/system/alive"
```

URL de la Solicitud:

```
https://sandbox.viafirma.com/documents/api/v3/system/alive
```

Ejemplo cuerpo de la Respuesta:

```
{
  "isAlive": "true",
  "pid": "29779"
}
```

Ejemplo código de la Respuesta:

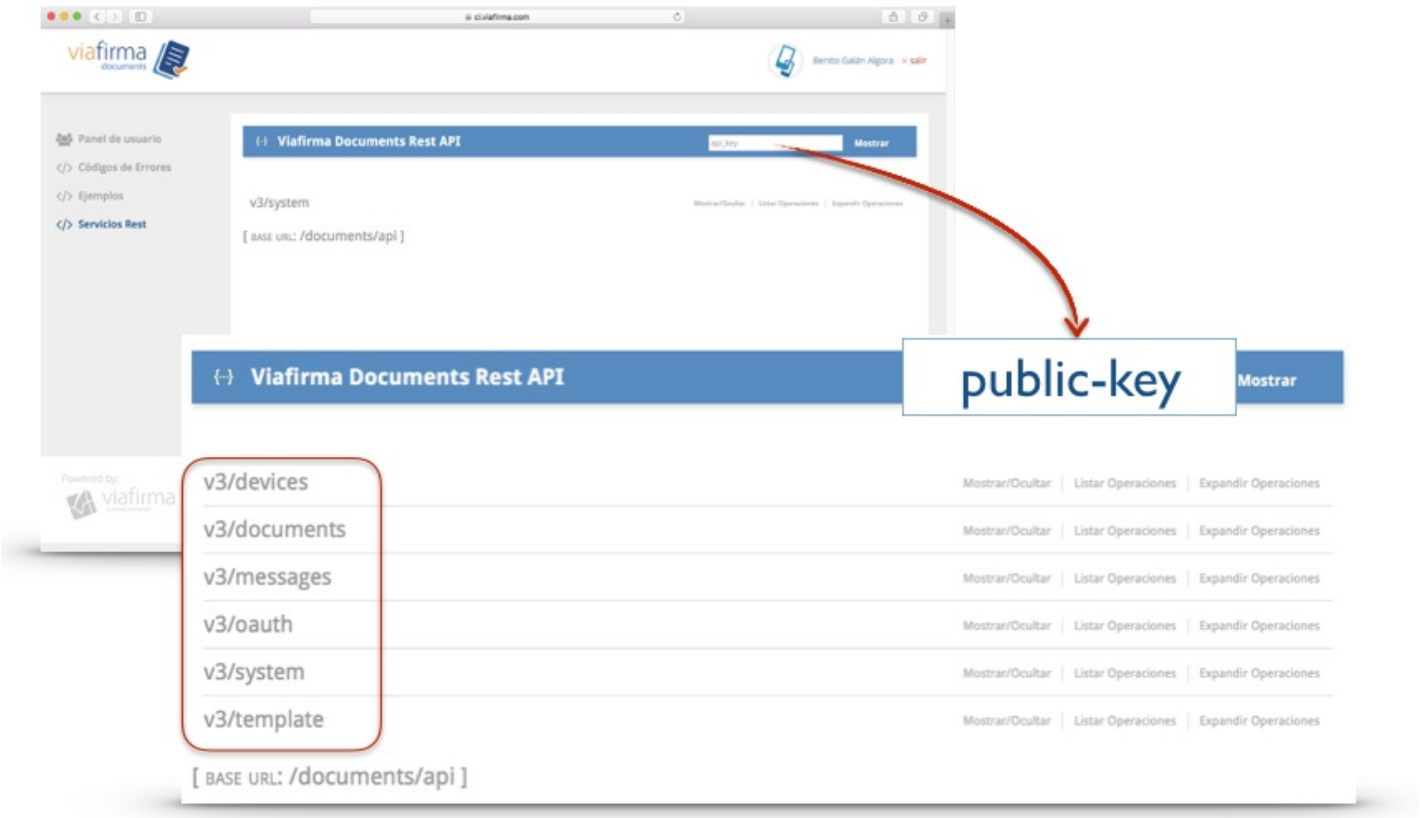
```
200
```

Ejemplo encabezados de la Respuesta:

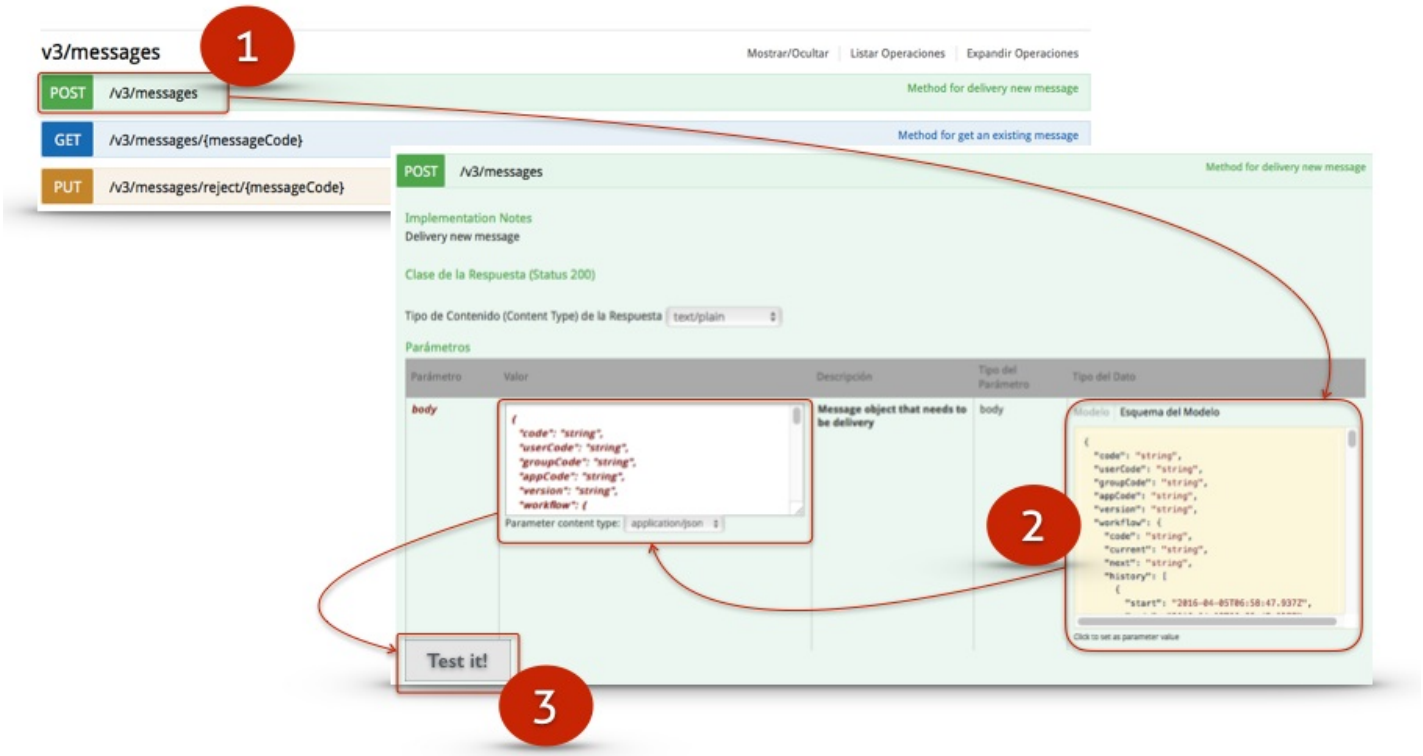
```
{
  "date": "Tue, 05 Apr 2016 11:13:30 GMT",
  "content-type": "application/json",
  "connection": "Keep-Alive",
  "keep-alive": "timeout=5, max=100",
  "transfer-encoding": "Identity"
}
```

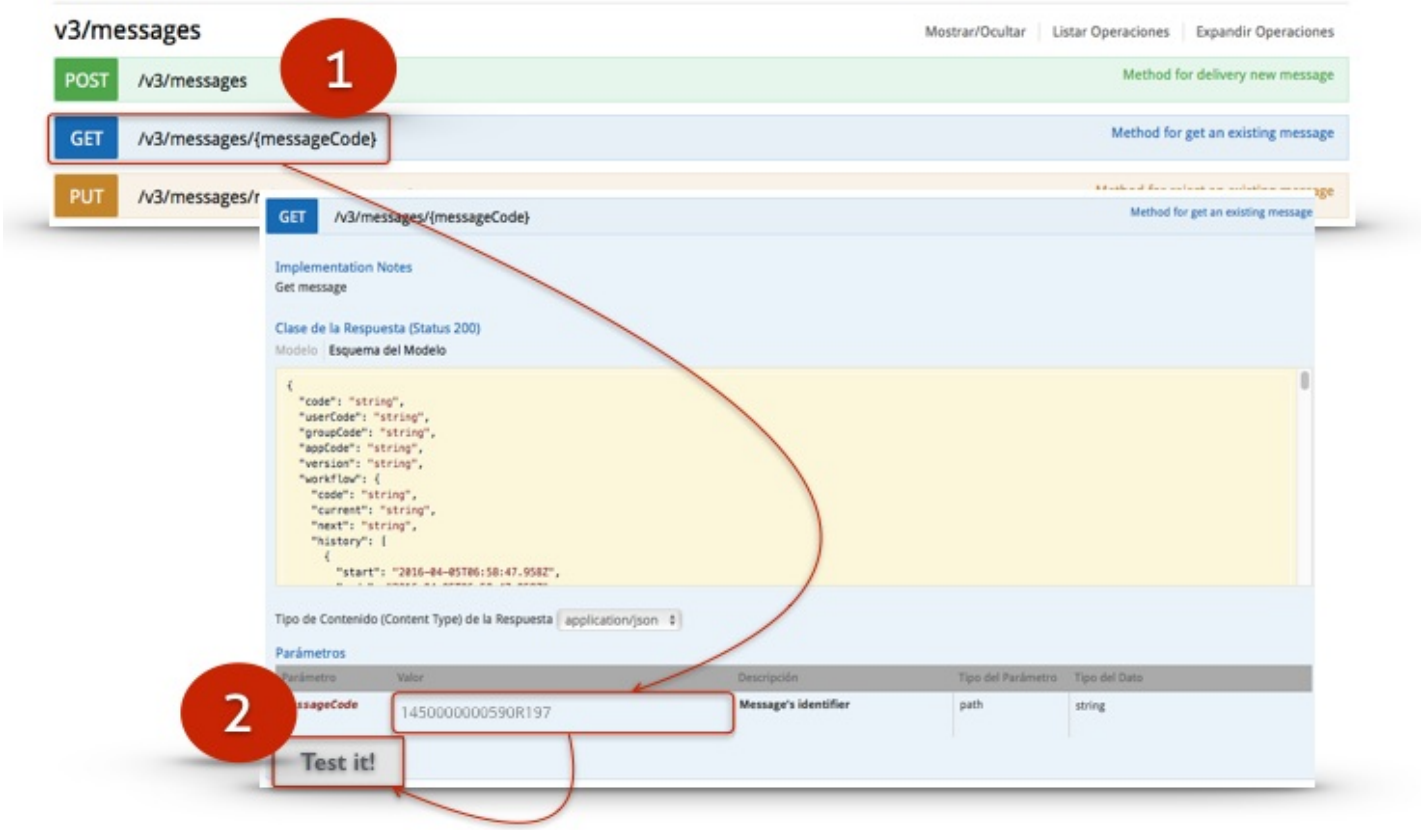
Servicios securizados para integradores

Para acceder al resto de servicios disponibles a integradores, necesitarás introducir una clave pública:



Una vez dentro, podrás consumir para cada entidad los distintos métodos implementados POST, GET y PUT


v3/messages
1
Mostrar/Ocultar
Listar Operaciones
Expandir Operaciones
POST /v3/messages
Method for delivery new message
GET /v3/messages/{messageCode}
Method for get an existing message
PUT /v3/messages/reject/{messageCode}
POST /v3/messages
Method for delivery new message
Implementation Notes
Delivery new message
Clase de la Respuesta (Status 200)
Tipo de Contenido (Content Type) de la Respuesta
text/plain
Parámetros
Parámetro
Valor
Descripción
Tipo del Parámetro
Tipo del Dato
body
Message object that needs to be delivery
body
Modelo
Esquema del Modelo
Parameter content type:
application/json
2
Test it!
3


v3/messages
Mostrar/Ocultar
Listar Operaciones
Expandir Operaciones
POST /v3/messages
1
Method for delivery new message
GET /v3/messages/{messageCode}
Method for get an existing message
PUT /v3/messages/r
GET /v3/messages/{messageCode}
Method for get an existing message
Implementation Notes
Get message
Clase de la Respuesta (Status 200)
Modelo
Esquema del Modelo
Tipo de Contenido (Content Type) de la Respuesta
application/json
Parámetros
Parámetro
Valor
Descripción
Tipo del Parámetro
Tipo del Dato
145000000590R197
Message's identifier
path
string
2
Test it!

Códigos de error

La excepción ApiException contiene los siguientes códigos de error, al que se puede acceder desde el atributo code de la excepción.

Código	Mensaje
1	Debes indicar una aplicación
2	Aplicación no encontrada con el código especificado
3	Configuración de la aplicación no disponible
4	Error al procesar la configuración de la aplicación
5	Error al guardar la aplicación
6	Error al guardar el grupo
11	Debes indicar un usuario
12	Usuario no encontrado con el identificador especificado
13	Usuario no encontrado con el correo electrónico especificado
14	Usuario no encontrado con el código especificado
15	Error al guardar el usuario
16	Los datos facilitados para el acceso son incorrectos
17	No se ha podido asociar la plantilla al usuario
18	No se ha podido solicitar la reactivación de la cuenta de usuario
19	Esta cuenta de usuario ya tiene solicitada la reactivación
20	Cuenta ya activa
300	Ya existe un usuario con el código indicado.
301	Ya existe un usuario con el correo electrónico indicado.
302	El correo electrónico es obligatorio
303	Error al enviar el correo de confirmación de la cuenta
304	El nombre es obligatorio
305	El apellido es obligatorio
306	Rol no encontrado
307	El código de grupo indicado no ha sido encontrado
308	Ya existe un usuario con el identificador indicado.
309	Ya existe un grupo con el código de grupo indicado.
21	Debes indicar un tipo de dispositivo

22	Tipo de dispositivos no encontrado
23	No existen dispositivos con el token especificado
24	No existen dispositivos con el identificador único especificado
25	Se ha producido un error al obtener el dispositivo
26	No se encontró ningún dispositivo según lo indicado en el mensaje
27	Se encontraron dispositivos de distintos usuarios según lo indicado en el mensaje
28	Se ha producido un error al registrar el dispositivo
29	Usuario no activo
30	Se ha llegado al máximo de dispositivos permitidos
31	Tipo de documento no reconocido
32	Se he producido un error al obtener el documento
33	Se ha producido un error al guardar el documento
34	Se ha producido un error al añadir la evidencia al documento
35	No se puede añadir una evidencia ya existente en el documento
36	Se ha producido un error al acceder al documento temporal
37	La notificación solo puede ser enviada a un dispositivo
38	No es posible obtener el documento solicitado porque no se encuentra firmado.
39	Error al registrar o actualizar el dispositivo
311	Se ha producido un error al actualizar el estado de la acción de firma
312	Se ha producido un error al añadir la firma al documento
313	El proceso es correcto, pero el documento firmado asociado al proceso ya no está disponible. Puede haber sido eliminado por la política de retención de documentos. Consulte la configuración del proceso.
314	Estás intentando firmar un PDF que ha sido firmado con un certificado digital emitido por un Prestador que no está autorizado en la configuración del servicio. Por favor, ponte en contacto con el administrador para que pueda proceder con la configuración adecuada.
315	Documento original no disponible
316	Se ha producido un error al añadir la firma al documento. No se ha detectado ningún trazo en la firma biométrica.
317	No es posible actualizar el estado de la evidencia.
40	Se ha producido un error al crear la plantilla solicitada
41	No se ha encontrado la plantilla con el código especificado
42	No se ha podido localizar el documento en cache
43	Plantilla no encontrado para el usuario
44	Error al recuperar la información del formulario de la plantilla solicitada
45	La fuente indicada no está configurada en el sistema

46	La plantilla no tiene formulario asociado
47	Hay un error con las posiciones de tus evidencias o firmas. Por favor, revísalas y vuelve a intentarlo.
48	No se puede utilizar alguna de las evidencias para enviar esta petición para firmar en web
110	Error al guardar información temporal
111	Error al recuperar información temporal
112	Error al borrar información temporal
80	No se ha podido localizar un token para usar esta aplicación. Contacta con el administrador del sistema
81	El token utilizado por esta aplicación no es válido. Contacta con el administrador del sistema
82	El token utilizado por esta aplicación no es válido. Contacta con el administrador del sistema
83	Ha caducado la sesión. Vuelve a ingresar en la aplicación
84	Error en la verificación de la marca de tiempo de la petición
85	Tu sesión ha sido cerrada por inactividad. Por favor ingresa nuevamente tus credenciales
86	Problemas al identificar esta aplicación en el servidor. Contacta con el administrador del sistema
87	Error en el proveedor de autenticación
88	Error al firmar el contenido de la respuesta
89	Protocolo no soportado, solo se permiten peticiones https
90	No tiene permiso para acceder al recurso solicitado
91	La aplicación no tiene permiso para acceder al recurso solicitado
92	El usuario no tiene permiso para acceder al recurso solicitado
93	No puede indicar el código de la petición
51	Se ha producido un error al obtener las notificaciones
52	Se ha producido un error al enviar la notificación
53	Se ha producido un error al actualizar la notificación
54	El usuario no existe, no tiene registrado ningún dispositivo o tiene más de uno. Especifique el usuario y dispositivo correcto.
55	Error en el envío de correo electrónico
56	El usuario no tiene correo electrónico asociado
57	No se ha podido enviar la petición porque el correo informado en la notificación no es un correo válido
58	No se ha podido enviar la petición porque el teléfono informado en la notificación no es un teléfono válido
61	Se ha producido un error al obtener la configuración de los workflows.
62	Base64 requerido
63	Se ha producido un error al generar la petición con el documento indicado en la URL

64	Se ha producido un error al generar la petición con el documento indicado en el base 64
65	No es posible conectar con la TSA
70	Se ha producido un error al recuperar la definición del documento
71	Se ha producido un error en el proceso de escaneado del documento
72	Se ha producido un error al recuperar la máscara del documento
73	Se ha producido un error en el proceso de petición de escaneado offline
100	Error al configurar viafirma core
101	Se ha producido un error al preparar la firma del documento
102	Se ha producido un error al utilizar la política de firma indicada
103	Se ha producido un error en la firma de alguna de las evidencias
104	Error al recuperar la clave publica del certificado. Por favor, comprueba si no esta caducado o revocado
105	Error al preparar la firma del documento
106	Error al preparar la firma del documento
107	Error en la configuración de la TSA
201	Se ha producido un error al validar el mensaje
202	Se ha producido un error al validar el mensaje
203	Se ha producido un error al validar el mensaje
204	Se ha producido un error al validar el mensaje
205	Se ha producido un error al validar el mensaje
206	Se ha producido un error al validar el mensaje
207	Se ha producido un error al validar el mensaje
208	Se ha producido un error al validar el mensaje
209	Se ha producido un error al validar el código
210	Se ha producido un error al validar la notificación
211	Se ha producido un error al validar la notificación
212	Se ha producido un error al validar la notificación
213	Se ha producido un error al validar la notificación
214	No se puede enviar la solicitud porque no se han definido ninguna política de firma.
230	Se ha producido un error guardar el mensaje en base de datos
231	Error al repostar el error a la url de aviso
232	No es posible rechazar un mensaje finalizado
233	No ha sido posible realizar la operación porque el mensaje ha expirado

234	Se ha producido un error al intentar modificar un mensaje ya finalizado anteriormente
235	El documento ya ha sido generado previamente
236	No es posible recuperar la información del mensaje solicitado
237	Error al asignar el mensaje a la tarea correspondiente
238	Error al procesar el json del mensaje
239	No se puede finalizar la operación solicitada
240	No se ha indicado el código de la plantilla o la url del documento externo a firmar.
241	Se ha superado el número máximo de anotaciones permitidas
242	Datos no válidos en la solicitud
243	El estado de la solicitud no es válido
250	Ha ocurrido un error con la aprobación.
251	El código de validación es incorrecto.
252	No se encuentra la aprobación.
253	El mensaje se encuentra rechazado.
254	El mensaje ya se está procesando en el backend
255	Todas las políticas deben tener al menos una firma
256	No es posible modificar el documento para este tipo de solicitud
260	Error al generar el documento firmado
261	Una de las evidencias provistas ha expirado
262	No es posible reenviar la petición. Solo se puede reiniciar si está expirada o rechazada.
263	Solo pueden deshabilitarse las peticiones que se encuentren en un estado final
270	Error al parsear el json
280	Solo pueden eliminarse evidencias opcionales en documentos pendientes
290	El estado del proceso que se pretende reenviar no es válido para el reenvío.
291	La solicitud seleccionada no tiene asociada una transferencia.
292	La solicitud seleccionada no tiene asociada un sistema de transferencia.
401	Las credenciales utilizadas no son válidas
403	No es posible aceptar la petición
501	Se ha producido un error al enviar el mensaje al servicio informado
601	No existen dispositivos para el usuario indicado
701	Error al recuperar la información de estado del sistema
710	Error en el envío de sms
711	Error en la verificación del token otp

712	El número de teléfono informado para el envío de sms no es válido
713	Error al generar la url para enviarla por sms
714	Error al generar el correo electrónico con el código de validación. Email no se enviará
715	Error al enviar el correo electrónico con el código de validación
716	El correo electrónico informado para el envío de la evidencia OTP es incorrecto o no viene informado.
717	Error al procesar la evidencia de tipo question
718	Se ha producido un error al adjuntar el fichero al mensaje
719	No se ha encontrado en el mensaje el adjunto
720	Número máximo de intentos de ingreso superado, no podrá ingresar hasta pasados 5 minutos.
801	Licencia no válida
802	Acción no permitida por licencia
995	Error al asignar la solicitud a la tarea correspondiente
996	Error en el sistema de almacenamiento temporal
997	Servicio no disponible en esta versión
998	Error al procesar la tarea {0} para el mensaje con código {1}
999	Vaya, algo no va bien y la aplicación no puede continuar. Ciérrala y vuelve a ingresar por favor

Configuraciones de ejemplo

Política con certificado por defecto

```
{
  "policies": [
    {
      "evidences": [
        {
          "type": "SIGNATURE | FINGERPRINT | IMAGE",
          "helpText": "Evidencia del usuario",
          "typeFormatSign": "XADES_B | XADES_T | XADES_LT | XADES_LTA",
          "positions": [
            {
              "rectangle": {
                "x": 75,
                "y": 75,
                "width": 100,
                "height": 50
              },
              "page": 1
            }
          ]
        }
      ],
      "signatures": [
        {
          "type": "SERVER",
          "helpText": "Firma del usuario",
          "typeFormatSign": "PADES_B | PADES_T | PADES_LT | PADES_LTA"
        }
      ],
      "checklist": [
          {
            "helpText": "Requiere aprobación del responsable",
            "validateCode": "12345",
            "signature": {
                "type": "SERVER",
                "helpText": "Firma del usuario",
                "typeFormatSign": "PADES_B | PADES_T | PADES_LT | PADES_LTA"
            }
          }
        ]
    }
  ]
}
```

Política con certificado configurado

```
{
  "policies": [
    {
      "evidences": [
        {
          "type": "SIGNATURE | FINGERPRINT | IMAGE",
          "helpText": "Evidencia del usuario",
```

```
            "typeFormatSign": "XADES_B | XADES_T | XADES_LT | XADES_LTA",
            "certificateAlias": "documents",
            "certificatePassword": "12345",
            "encryptionKeyAlias": "documents-cipher",
            "positions": [
              {
                "rectangle": {
                  "x": 75,
                  "y": 75,
                  "width": 100,
                  "height": 50
                },
                "page": 1
              }
            ]
          }
        ],
        "signatures": [
          {
            "type": "SERVER",
            "helpText": "Firma del usuario",
            "certificateAlias": "documents",
            "certificatePassword": "12345",
            "typeFormatSign": "PADES_B | PADES_T | PADES_LT | PADES_LTA"
          }
        ],
        "checklist": [
            {
              "helpText": "Requiere aprobación del responsable",
              "validateCode": "12345",
              "signature": {
                  "type": "SERVER",
                  "helpText": "Firma del usuario",
                  "certificateAlias": "documents",
                  "certificatePassword": "12345",
                  "typeFormatSign": "PADES_B | PADES_T | PADES_LT | PADES_LTA"
              }
            }
          ]
      }
    ]
  }
```

Plantillas de ejemplo

Integrar con viafirma documents

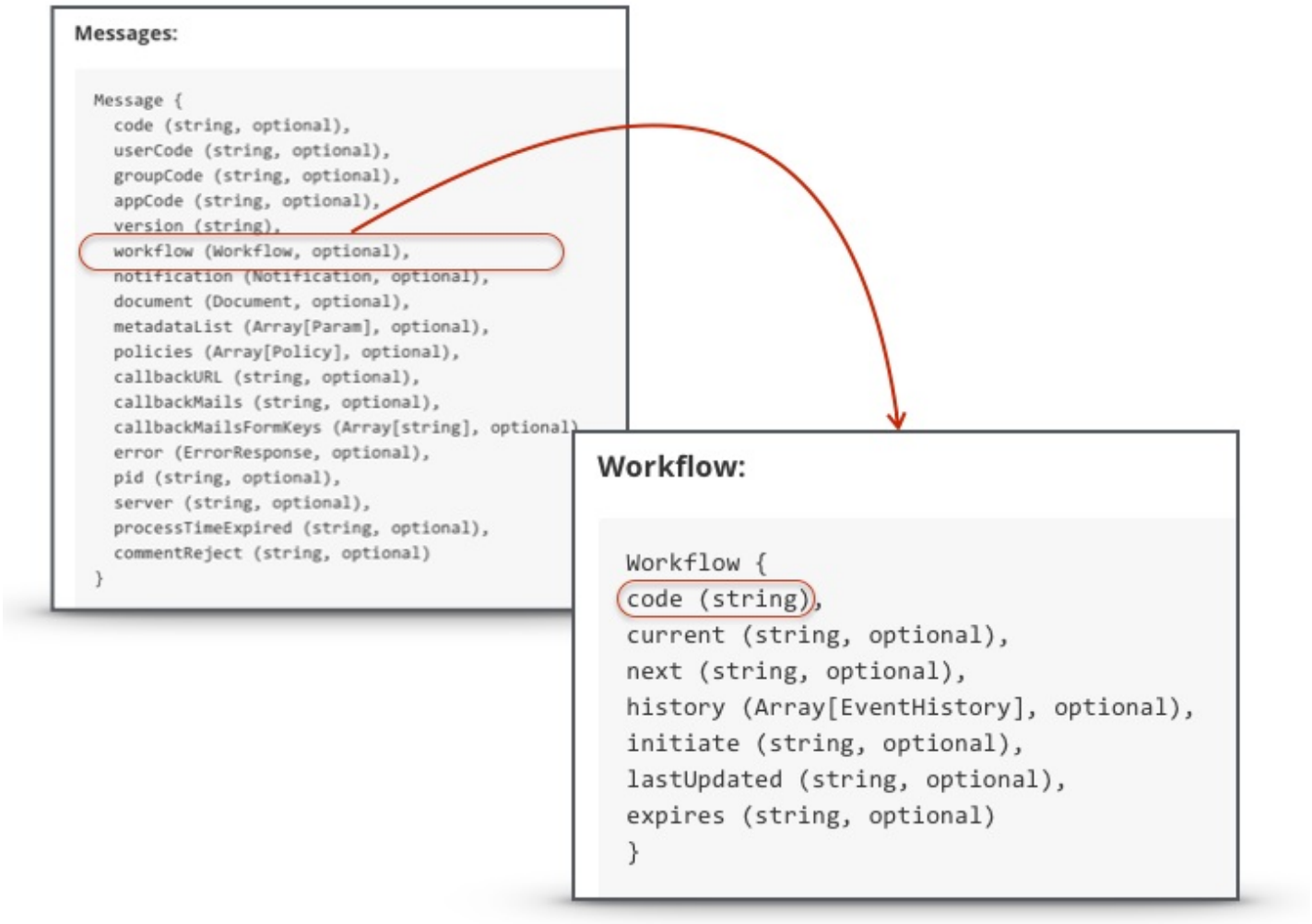
Con viafirma documents puedes implementar distintos ciclos de vida en tus documentos. A continuación te explicamos los cuatro flujos más habituales que puedes consumir.

1. captura y firma de evidencias (default)
2. registro de datos
3. firma con autorización previa
4. generación de PDF para impresión

Para elegir uno u otro worflow deberás usar el parámetro messages.workflow.code en la llamada del servicio. Si lo omites, el flujo será el completo.

Por ejemplo:

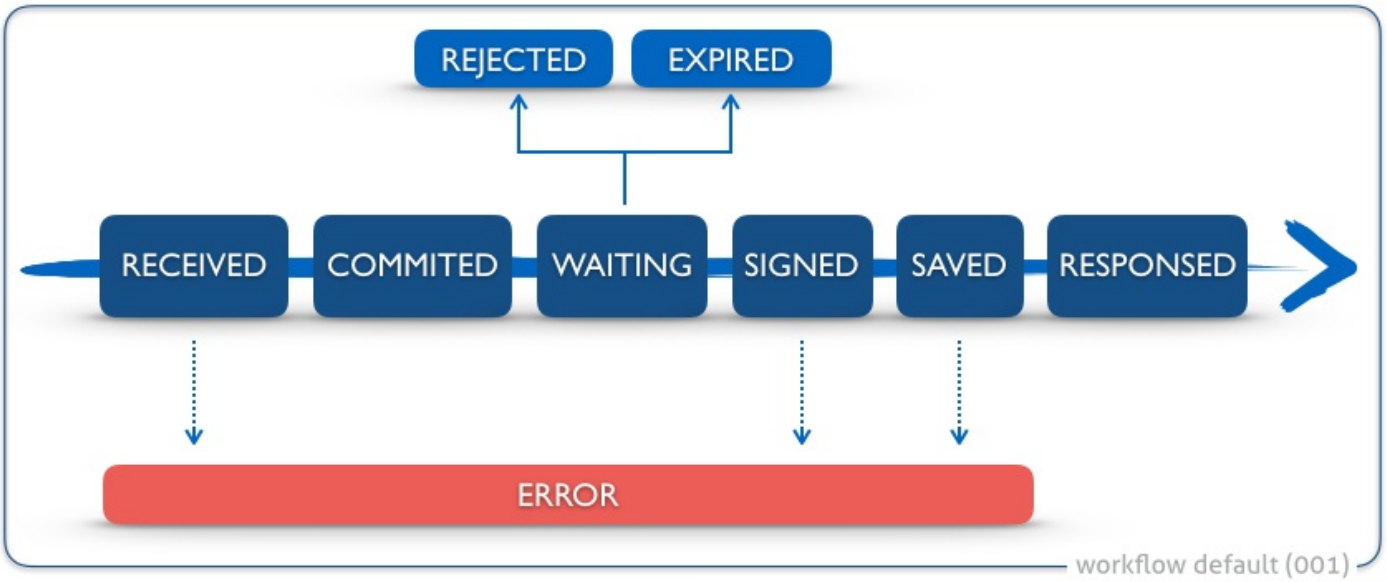
```
{
  "userCode": "string",
  "appCode": "string",
  "version": "string",
  "workflow": {
    "code": "004"
  }
}
```



A continuación la descripción de cada tipo de flujo.

Flujo por defecto

El mensaje puede caracer de workflow, y en su defecto se implementa el ciclo completo, asociado internamente al workflow.code = 001, el cual se resume en el siguiente diagrama de estados:



En pocas palabras, con este flujo tendrás:

- recibes documento en el dispositivo móvil,
- se capturan las envidencias acorde a la política definida,
- se firma,
- se hace call-back al sistema que invocó la petición,
- se persiste el documento firmado.

Mensaje de Ejemplo

nota: puedes parametrizar el siguiente JSON de ejemplo y usarlo tal y como se explica en el capítulo 2 "Testing" a través de la UI habilitada en el backend de viafirma documents.

```
{
  "notification" : {
    "text" : "Message sent from CRM",
    "detail" : "contract sample 01",
    "devices" : [ {
      "appCode" : "com.viafirma.mobile.ios.documents",
      "code" : "your_devicecode_here",
      "type" : "IOS",
      "userCode" : "your_user_here"
    } ]
  },
  "document" : {
    "templateCode" : "301_example",
    "templateType" : "docx",
```

```
    "items" : [ {
      "key" : "KEY_01",
      "value" : "jhon"
    }, {
      "key" : "KEY_02",
      "value" : "doe"
    }, {
      "key" : "KEY_03",
      "value" : "jhondoe@mail.com"
    }, {
      "key" : "KEY_04",
      "value" : "id99282"
    } ]
  },
  "callbackURL": "http://mysystem.com/responseSample"
}
```

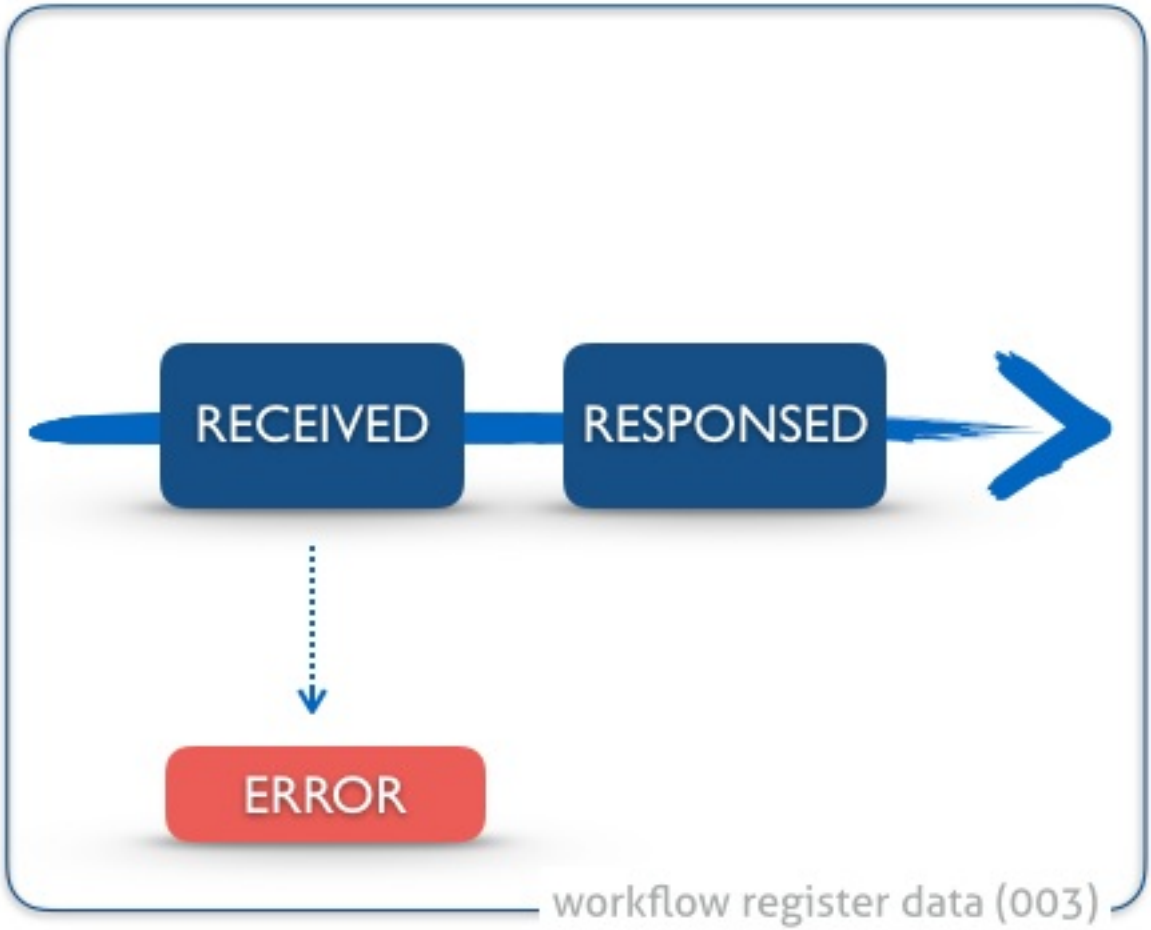
Registro de datos

Este flujo está pensado para procesos de captura de datos en formularios que:

- NO NECESITAN generar documento PDF
- NO NECESITAN capturar evidencias
- NO NECESITAN ser firmados

Los datos recogidos son persistidos en el backend y, opcionalmente, son remitidos por correo electrónico. Útiles por ejemplo para recogida de encuentas de satisfacción de clientes, formularios de invitación, etc.

Para su uso, el mensaje debe especificar workflow.code = EX_003, ejecutando el siguiente ciclo de vida del mensaje.



En pocas palabras, con este flujo tendrás:

- recibes o generas formulario en la app móvil,
- capturas datos de formulario,
- se envían y persisten al backend y,
- de forma opcional, los datos recogidos en el formulario son remitidos por correo electrónico.

Mensaje de Ejemplo

nota: puedes parametrizar el siguiente JSON de ejemplo y usarlo tal y como se explica en el capítulo 2 "Testing" a través de la UI habilitada en el backend de viafirma documents.

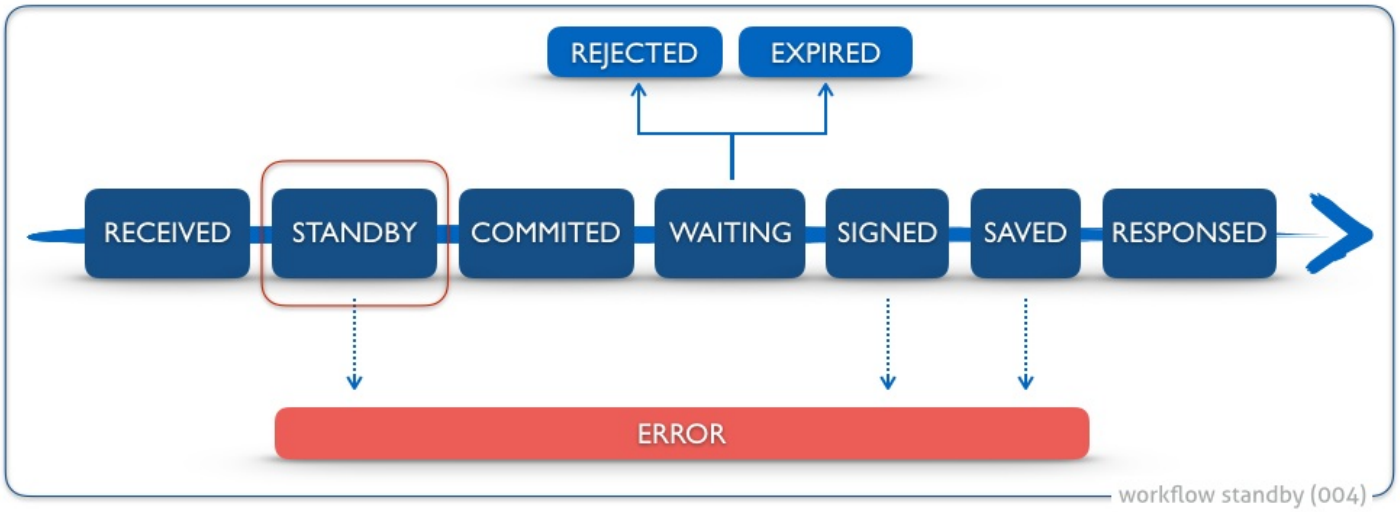
```
{
  "workflow" : {
    "code": "EX003"
  },
  "notification" : {
    "text" : "Message title",
    "detail" : "Message detail"
  },
  "document" : {
    "templateCode" : "301_example",
    "items" : [ {
      "key" : "KEY_01",
      "value" : "jhon"
    }, {
      "key" : "KEY_02",
      "value" : "doe"
    }, {
      "key" : "KEY_03",
      "value" : "jhondoe@mail.com"
    }, {
      "key" : "KEY_04",
      "value" : "id99282"
    } ]
  },
  "callbackMails": "jhondoe01@yopmail.com,jhondoe01@yopmail.com",
  "callbackURL": "http://mysystem.com/responseSample"
}
```

Preautorización

Este flujo está pensado para dejar en standby un mensaje que debe ser enviado a un dispositivo para su firma.

Es decir, delegamos un control de supervisión a un usuario que, desde el backend, debería revisar el mensaje y decidir si rechazarlo o finalmente enviarlo a su destino (app móvil).

Para su uso, el mensaje debe especificar workflow.code = EX_004, ejecutando el siguiente ciclo de vida del mensaje.



En pocas palabras, con este flujo tendrás:

- se recibe una petición de mensaje al backend,
- en backend un usuario "supervisor" decide su aprobación o rehazao,
- y en caso de aprobación, se completa un ciclo completo, similar al flujo por defecto (001).

Mensaje de Ejemplo

nota: puedes parametrizar el siguiente JSON de ejemplo y usarlo tal y como se explica en el capítulo 2 "Testing" a través de la UI habilitada en el backend de viafirma documents.

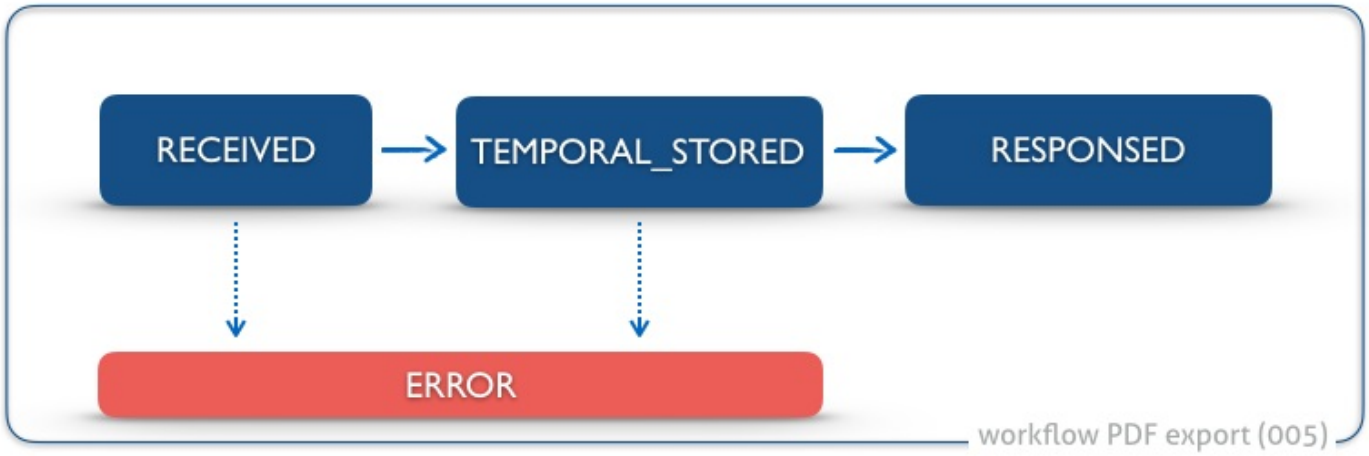
```
{
  "workflow" : {
    "code": "EX004"
  },
  "notification" : {
    "text" : "Message sent from CRM",
    "detail" : "need to be authorized",
    "devices" : [ {
      "appCode" : "com.viafirma.mobile.ios.documents",
      "code" : "your_devicecode_here",
      "type" : "IOS",
      "userCode" : "your_user_here"
    } ]
```

```
  },
  "document" : {
    "templateCode" : "301_example",
    "templateType" : "docx",
    "items" : [ {
      "key" : "KEY_01",
      "value" : "jhon"
    }, {
      "key" : "KEY_02",
      "value" : "doe"
    }, {
      "key" : "KEY_03",
      "value" : "jhondoe@mail.com"
    }, {
      "key" : "KEY_04",
      "value" : "id99282"
    } ]
  },
  "callbackURL": "http://mysystem.com/responseSample"
}
```

Exportar PDF

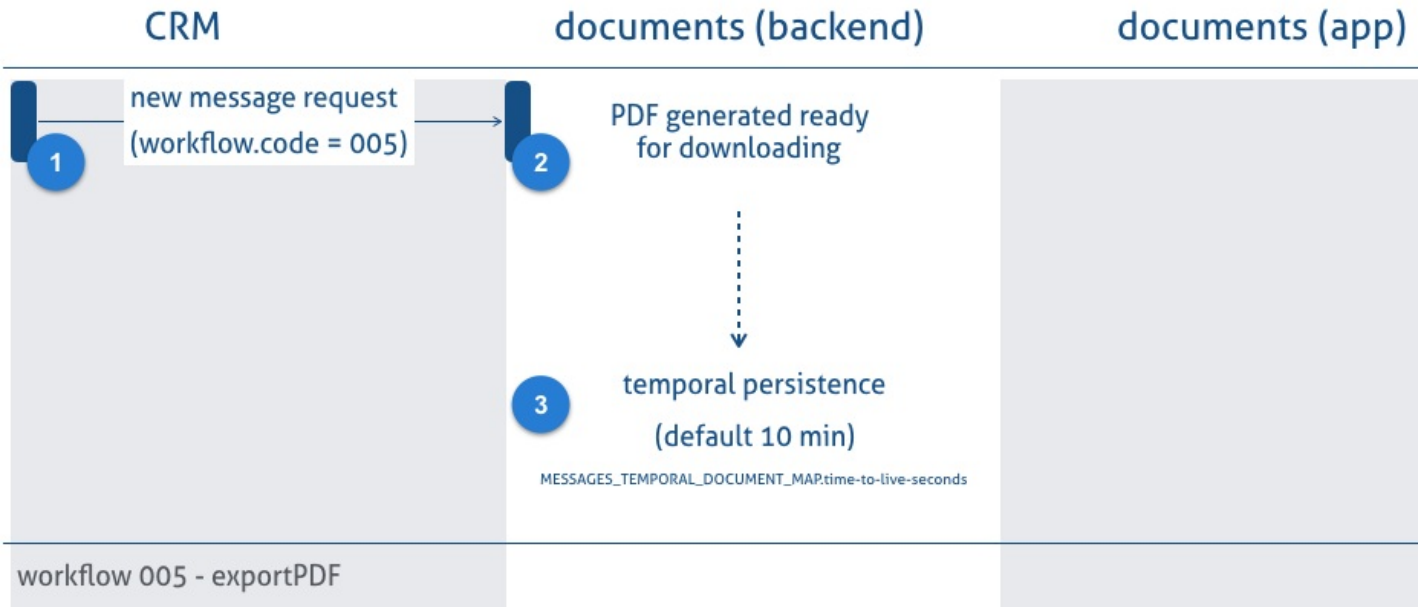
Este flujo está pensado para la impresión del PDF, bien para contingencias en caso de no disponer de dispositivos móviles, o bien porque deseas una copia en papel para que el usuario lo firme de forma "tradicional".

Para su uso, el mensaje debe especificar workflow.code = EX005, ejecutando el siguiente ciclo de vida del mensaje.



Por tanto, el ciclo de vida de este flujo se resumen en los siguientes pasos:

- se recibe nuevo mensaje (datos + plantilla),
- se genera PDF,
- permanece en el backend el tiempo definido en la configuración [1]



[1] la variable MESSAGES_TEMPORAL_DOCUMENT_MAP.time-to-live-seconds se define en el fichero de configuración hazelcast.xml. Para más información consulta la documentación de instalación y configuración del backend. Por defecto es de 10 minutos

Mensaje de Ejemplo

nota: puedes parametrizar el siguiente JSON de ejemplo y usarlo tal y como se explica en el capítulo 2 "Testing" a través de la UI habilitada en el backend de viafirma documents.

```
{
  "workflow" : {
    "code": "EX005"
  },
  "notification" : {
    "text" : "Message title",
    "detail" : "Message detail"
  },
  "document" : {
    "templateCode" : "301_example",
    "templateType" : "docx",
    "items" : [ {
      "key" : "KEY_01",
      "value" : "jhon"
    }, {
      "key" : "KEY_02",
      "value" : "doe"
    }, {
      "key" : "KEY_03",
      "value" : "jhondoe@mail.com"
    }, {
      "key" : "KEY_04",
      "value" : "id99282"
    } ]
  }
}
```

Firma desde link

Este flujo está pensado para poder firmar una petición sin necesidad de abrir sesión.

Para su uso, se debe enviar por correo electrónico un enlace a un código QR que posteriormente la aplicación móvil leerá, y desde donde se firmará la petición.

En caso de que se requiera aprobación, el usuario deberá confirmar o rechazar dicha aprobación (*Ver la sección Servicios para aprobación más abajo*).

Por tanto, el ciclo de vida de este flujo se resumen en los siguientes pasos:

- se recibe nuevo mensaje,
- se envía el enlace por correo electrónico,
- se firma la petición desde la aplicación móvil,
- si la petición requiere aprobación, desde el backend o el backoffice integrado se confirma o rechaza la aprobación,
- si la petición no requiere ninguna aprobación, el flujo finaliza.

Mensaje de Ejemplo

nota: puedes parametrizar el siguiente JSON de ejemplo y usarlo tal y como se explica en el capítulo 2 "Testing" a través de la UI habilitada en el backend de viafirma documents.

```
{
  "notification" : {
    "text" : "Firma biométricas simple con check en backend",
    "detail" : "Check en backend requerido",
    "sharedLink" : {
      "appCode" : "com.viafirma.ventamovil",
      "email" : "enduserB2B@yopmail.com"
    }
  },
  "document" : {
    "templateCode" : "329_example",
    "items" : [ {
      "key" : "KEY_01",
      "value" : "Test value"
    }, {
      "key" : "KEY_02",
      "value" : "Test value"
    }, {
      "key" : "KEY_03",
      "value" : "Test value"
    }, {
      "key" : "KEY_04",
      "value" : "Test value"
    } ]
  },
  "policies": [
      {
        "evidences": [
          {
```

```
            "type": "SIGNATURE",
            "helpText": "Firma del usuario",
            "typeFormatSign": "XADES_B"
          }
        ],
        "signatures": [
          {
            "type": "SERVER",
            "helpText": "Firma del servicio",
            "typeFormatSign": "PADES_B"
          }
        ],
        "checklist": [
          {
            "helpText": "Requiere aprobación del responsable",
            "validateCode": "12345",
            "signature": {
                "type": "SERVER",
                "helpText": "Firma del servicio",
                "typeFormatSign": "PADES_B"
            }
          }
        ]
      }
    ]
}
```

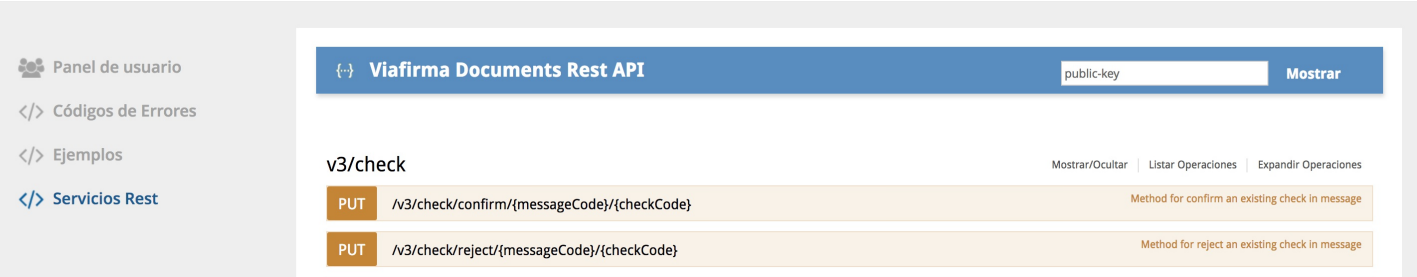
Mensaje de Ejemplo con OTP SMS

```
{
  "workflow": {
    "code": "EX006"
  },
  "notification": {
    "text": "Firma con OTP SMS",
    "detail": "Solicitud de firma mediante SMS",
    "sharedLink": {
      "appCode": "com.viafirma.documents",
      "email": "enduserB2B@yopmail.com"
    }
  },
  "document": {
    "templateCode": "350_example",
    "items": [
      {
        "key": "KEY_01",
        "value": "Test value"
      },
      {
        "key": "KEY_02",
        "value": "Test value"
      },
      {
        "key": "KEY_03",
        "value": "Test value"
      },
      {
        "key": "KEY_04",
        "value": "Test value"
      }
    ]
  },
  "policies": [
```

```
    {
      "evidences": [
        {
          "type": "OTP_SMS",
          "helpText": "SMS del usuario",
          "phone": "+34666554433",
          "typeFormatSign": "XADES_B",
          "positions": [
            {
              "rectangle": {
                "x": 535,
                "y": 10,
                "width": 50,
                "height": 50
              },
              "page": 1
            }
          ]
        }
      ],
      "signatures": [
        {
          "type": "SERVER",
          "helpText": "Documento certificado por viafirma documents (servicios de confianza)",
          "typeFormatSign": "PADES_LT",
          "stampers": [
            {
              "type": "TEXT",
              "xAxis": 565,
              "yAxis": 250,
              "rotation": "ROTATE_270",
              "page": 1
            }
          ]
        }
      ]
    }
  ],
  "callbackMails": "enduserB2B@yopmail.com"
}
```

Servicios para aprobación

Los servicios para confirmar o rechazar una aprobación son públicos, de forma que son accesibles desde la aplicación con usuario logado, en la parte de documentación de los servicios REST, introduciendo la clave public-key.



Confirmar

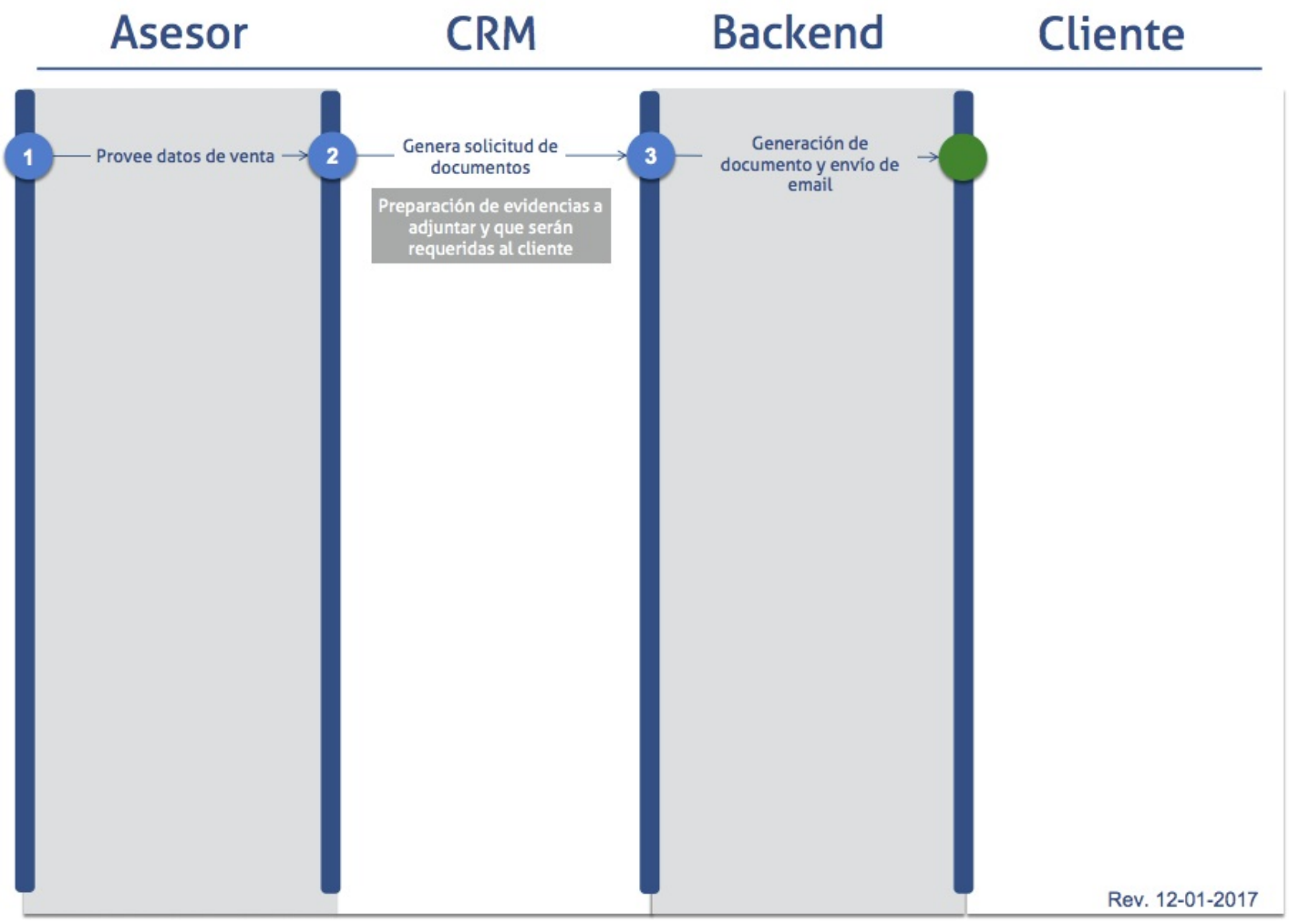
Para confirmar la aprobación, será necesario indicar el código de mensaje, así como el código de la aprobación. Además, se podrá requerir un código de validación configurable desde la política de la petición.

Rechazar

Para rechazar la aprobación, será necesario indicar el código de mensaje, así como el código de la aprobación. De forma opcional, el usuario podrá indicar un comentario con el motivo de rechazo. Además, se podrá requerir un código de validación, configurable desde la política de la petición.

Ejemplo de generación de documento

A continuación se detallarán los datos necesarios para realizar la integración que permita generar un documento según el siguiente flujo:



URL de la Solicitud:

La URL de la solicitud será:

```
https://sandbox.viafirma.com/documents/api/v3/messages
```

Además, indicaremos que la petición HTTP es de tipo POST y cuyo Content-Type tendrá valor *application/json*.

Ejemplo cuerpo de la Petición:

El CRM indicará los datos necesarios acordes a la plantilla utilizada dentro del parámetro items, mediante el sistema *clave/valor*.

```
{
  "groupCode": "my_group_code",
  "notification" : {
    "text" : "Cuerpo del correo",
    "sharedLink" : {
      "appCode" : "com.viafirma.app",
      "email" : "enduserCode@example.com",
      "subject" : "Asunto del correo"
    }
  },
   "document" : {
    "templateCode" : "301_example",
    "items" : [ {
      "key" : "KEY_01",
      "value" : "Test value"
    }, {
      "key" : "KEY_02",
      "value" : "Test value"
    }, {
      "key" : "KEY_03",
      "value" : "Test value"
    }, {
      "key" : "KEY_04",
      "value" : "Test value"
    } ]
  }
}
```

Respuesta:

Los datos obtenidos consistirán en:

- el cuerpo de la *Respuesta*, que indicará el código único del documento generado:

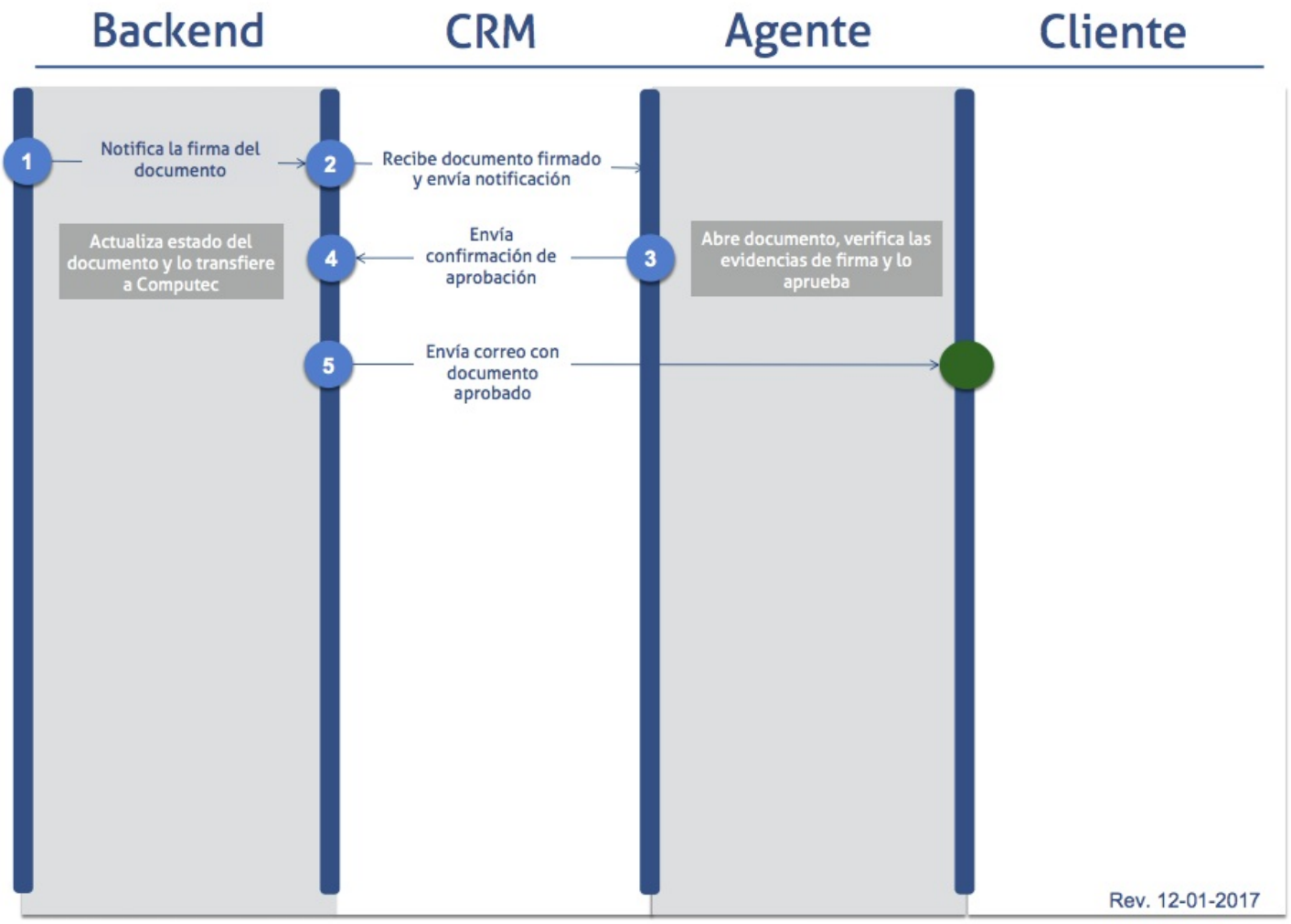
```
1493028491652R956
```

- el código http del estado de la operación:

```
200
```

Ejemplo de aprobación de documento firmado

A continuación se detallarán los datos necesarios para realizar la integración que permita aprobar un documento firmado según el siguiente flujo:



Generación del documento

Previamente debemos generar el documento.

URL de la Solicitud:

La URL de la solicitud será:

```
https://sandbox.viafirma.com/documents/api/v3/messages
```

Además, indicaremos que la petición HTTP es de tipo POST y cuyo Content-Type tendrá valor *application/json*.

Ejemplo cuerpo de la Petición:

Tenemos dos formas de indicar que el documento que queremos generar implica acción de aprobación/rechazo desde el CRM.

- Mediante el código de una plantilla predefinida. Esta plantilla deberá configurarse apropiadamente y todo documento generado a raíz de dicha plantilla, requerirá la acción de aprobación/rechazo desde el CRM.

```
{
  "groupCode": "my_group_code",
  "notification" : {
    "text" : "Cuerpo del correo",
    "sharedLink" : {
      "appCode" : "com.viafirma.app",
      "email" : "enduserCode@example.com",
      "subject" : "Asunto del correo"
    }
  },
   "document" : {
    "templateCode" : "329_example",
    "items" : [ {
      "key" : "KEY_01",
      "value" : "Test value"
    }, {
      "key" : "KEY_02",
      "value" : "Test value"
    }, {
      "key" : "KEY_03",
      "value" : "Test value"
    }, {
      "key" : "KEY_04",
      "value" : "Test value"
    } ]
  }
}
```

- Consultando la política de la plantilla que vamos a utilizar y configurando el atributo *checklist*. Para ello, necesitamos hacer una petición adicional que nos devuelva esta información:

  URL (petición *GET*)

  ```
  https://sandbox.viafirma.com/documents/api/v3/template/{code}
  ```

  Parámetros:

  code *(requerido)*: código de la plantilla que vamos a consultar (ejemplo: 301_example).

  Respuesta:

  En el cuerpo de la respuesta obtendremos todos los datos de la plantilla:

```
{
  "code": "301_example",
  "title": "Firma biométricas simple",
```

```
  "description": "1 firma biométrica sin sello de tiempo",
  "form": {
    "version": "0001",
    "containers": [
      {
        "name": "Container 1",
        "rows": [
          {
            "items": [
              {
                "key": "KEY_01",
                "type": "text",
                "label": "KEY_01"
              }
            ]
          },
          {
            "items": [
              {
                "key": "KEY_02",
                "type": "text",
                "label": "KEY_02"
              }
            ]
          },
          {
            "items": [
              {
                "key": "KEY_03",
                "type": "text",
                "label": "KEY_03"
              }
            ]
          },
          {
            "items": [
              {
                "key": "KEY_04",
                "type": "text",
                "label": "KEY_04"
              }
            ]
          }
        ]
      }
    ],
    "settings": {
      "policies": [
        {
          "evidences": [
            {
              "type": "SIGNATURE",
              "helpText": "Firma del usuario",
              "typeFormatSign": "XADES_B"
            }
          ],
          "signatures": [
            {
              "type": "SERVER",
              "helpText": "Firma del servicio",
              "typeFormatSign": "PADES_B"
            }
          ]
        }
      ]
    }
  },
  "version": "1",
```

```
  "type": "docx"
}
```

De los cuales incluiremos el fragmento de política en el cuerpo de la petición para generar el documento, y configuraremos el atributo *checklist* para indicar que requerimos la acción de aprobar. Un ejemplo del cuerpo de la petición, con la política modificada, sería:

```
{
  "groupCode": "my_group_code",
  "notification" : {
    "text" : "Cuerpo del correo",
    "sharedLink" : {
      "appCode" : "com.viafirma.app",
      "email" : "enduserCode@example.com",
      "subject" : "Asunto del correo"
    }
  },
   "document" : {
    "templateCode" : "301_example",
    "items" : [ {
      "key" : "KEY_01",
      "value" : "Test value"
    }, {
      "key" : "KEY_02",
      "value" : "Test value"
    }, {
      "key" : "KEY_03",
      "value" : "Test value"
    }, {
      "key" : "KEY_04",
      "value" : "Test value"
    } ]
  },
  "policies": [
        {
          "evidences": [
            {
              "type": "SIGNATURE",
              "helpText": "Firma del usuario",
              "typeFormatSign": "XADES_B"
            }
          ],
          "signatures": [
            {
              "type": "SERVER",
              "helpText": "Firma del servicio",
              "typeFormatSign": "PADES_B"
            }
          ],
          "checklist": [
            {
                "signature": {
                    "type": "SERVER",
                    "helpText": "Firma del servicio",
                    "typeFormatSign": "PADES_B"
                }
            }
          ]
        }
      ]
}
```

Como cualquier otra solicitud de generación de documento, el CRM indicará los datos necesarios acordes a la plantilla utilizada dentro del parámetro items, mediante el sistema *clave/valor*.

Respuesta:

Los datos obtenidos consistirán en:

- el cuerpo de la *Respuesta*, que indicará el código único del documento generado:

```
1493028491652R956
```

- el código HTTP del estado de la operación:

```
200
```

Aprobar el documento firmado

Los datos para realizar la petición de aprobación son:

URL de la Solicitud (aprobar documento):

La URL de la solicitud será:

```
https://sandbox.viafirma.com/documents/api/v3/check/confirm/{messageCode}/{checkCode}
```

Además, indicaremos que la petición HTTP es de tipo PUT.

Parámetros:

Los parámetros aceptados son:

- messageCode *(requerido)*: código único del documento (ejemplo: 1493028491652R956).
- checkCode *(requerido)*: código de la acción que vamos a aprobar (ejemplo: 1493028491652R956P001C001).
- validateCode *(opcional)*: código de validación para proteger la acción a realizar (ejemplo: k3W28)

Respuesta:

Los datos obtenidos consistirán en:

- el cuerpo de la *Respuesta*, indicará el estado del documento, así como los datos de la acción realizada:

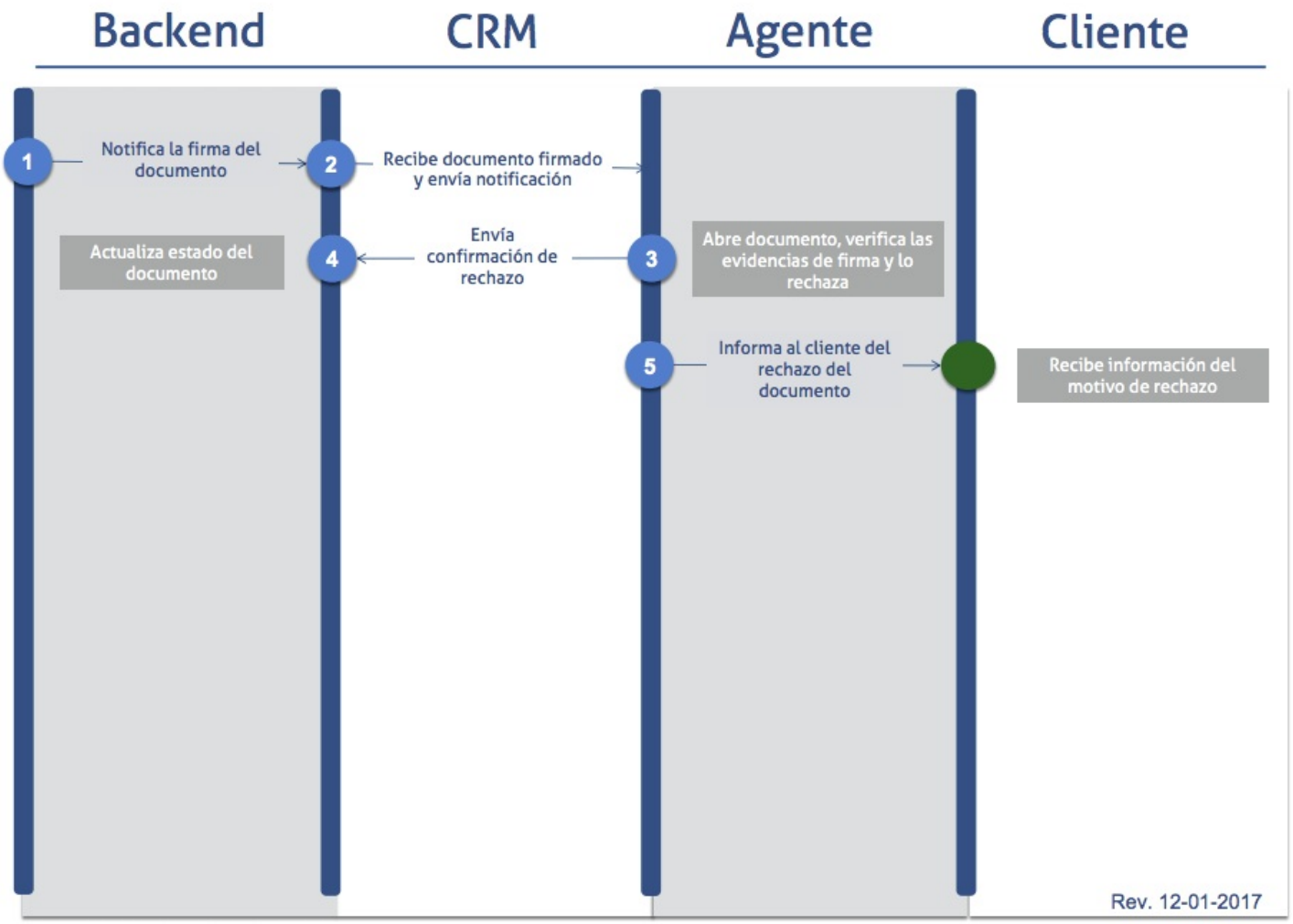
```
{
  "status": "PENDING",
  "code": "1493028491652R956P001C001",
  "signature": {
    "type": "SERVER",
    "helpText": "Firma del servicio",
    "typeFormatSign": "PADES_B"
  }
}
```

- el código HTTP del estado de la operación:

```
200
```

Ejemplo de rechazo de documento firmado

A continuación se detallarán los datos necesarios para realizar la integración que permita rechazar un documento firmado según el siguiente flujo:



Generación del documento

Previamente debemos generar el documento.

URL de la Solicitud:

La URL de la solicitud será:

```
https://sandbox.viafirma.com/documents/api/v3/messages
```

Además, indicaremos que la petición HTTP es de tipo POST y cuyo Content-Type tendrá valor *application/json*.

Ejemplo cuerpo de la Petición:

Tenemos dos formas de indicar que el documento que queremos generar implica acción de aprobación/rechazo desde el CRM.

- Mediante el código de una plantilla predefinida. Esta plantilla deberá configurarse apropiadamente y todo documento generado a raíz de dicha plantilla, requerirá la acción de aprobación/rechazo desde el CRM.

```
{
  "groupCode": "my_group_code",
  "notification" : {
    "text" : "Cuerpo del correo",
    "sharedLink" : {
      "appCode" : "com.viafirma.app",
      "email" : "enduserCode@example.com",
      "subject" : "Asunto del correo"
    }
  },
   "document" : {
    "templateCode" : "329_example",
    "items" : [ {
      "key" : "KEY_01",
      "value" : "Test value"
    }, {
      "key" : "KEY_02",
      "value" : "Test value"
    }, {
      "key" : "KEY_03",
      "value" : "Test value"
    }, {
      "key" : "KEY_04",
      "value" : "Test value"
    } ]
  }
}
```

- Consultando la política de la plantilla que vamos a utilizar y configurando el atributo *checklist*. Para ello, necesitamos hacer una petición adicional que nos devuelva esta información:

  URL (petición *GET*)

  ```
  https://sandbox.viafirma.com/documents/api/v3/template/{code}
  ```

  Parámetros:

  code *(requerido)*: código de la plantilla que vamos a consultar (ejemplo: 301_example).

  Respuesta:

  En el cuerpo de la respuesta obtendremos todos los datos de la plantilla:

```
{
  "code": "301_example",
  "title": "Firma biométricas simple",
```

```
  "description": "1 firma biométrica sin sello de tiempo",
  "form": {
    "version": "0001",
    "containers": [
      {
        "name": "Container 1",
        "rows": [
          {
            "items": [
              {
                "key": "KEY_01",
                "type": "text",
                "label": "KEY_01"
              }
            ]
          },
          {
            "items": [
              {
                "key": "KEY_02",
                "type": "text",
                "label": "KEY_02"
              }
            ]
          },
          {
            "items": [
              {
                "key": "KEY_03",
                "type": "text",
                "label": "KEY_03"
              }
            ]
          },
          {
            "items": [
              {
                "key": "KEY_04",
                "type": "text",
                "label": "KEY_04"
              }
            ]
          }
        ]
      }
    ],
    "settings": {
      "policies": [
        {
          "evidences": [
            {
              "type": "SIGNATURE",
              "helpText": "Firma del usuario",
              "typeFormatSign": "XADES_B"
            }
          ],
          "signatures": [
            {
              "type": "SERVER",
              "helpText": "Firma del servicio",
              "typeFormatSign": "PADES_B"
            }
          ]
        }
      ]
    }
  },
  "version": "1",
```

```
  "type": "docx"
}
```

De los cuales incluiremos el fragmento de política en el cuerpo de la petición para generar el documento, y configuraremos el atributo *checklist* para indicar que requerimos la acción de aprobar. Un ejemplo del cuerpo de la petición, con la política modificada, sería:

```
{
  "groupCode": "my_group_code",
  "notification" : {
    "text" : "Cuerpo del correo",
    "sharedLink" : {
      "appCode" : "com.viafirma.app",
      "email" : "enduserCode@example.com",
      "subject" : "Asunto del correo"
    }
  },
   "document" : {
    "templateCode" : "301_example",
    "items" : [ {
      "key" : "KEY_01",
      "value" : "Test value"
    }, {
      "key" : "KEY_02",
      "value" : "Test value"
    }, {
      "key" : "KEY_03",
      "value" : "Test value"
    }, {
      "key" : "KEY_04",
      "value" : "Test value"
    } ]
  },
  "policies": [
        {
          "evidences": [
            {
              "type": "SIGNATURE",
              "helpText": "Firma del usuario",
              "typeFormatSign": "XADES_B"
            }
          ],
          "signatures": [
            {
              "type": "SERVER",
              "helpText": "Firma del servicio",
              "typeFormatSign": "PADES_B"
            }
          ],
          "checklist": [
            {
                "signature": {
                    "type": "SERVER",
                    "helpText": "Firma del servicio",
                    "typeFormatSign": "PADES_B"
                }
            }
          ]
        }
      ]
}
```

Como cualquier otra solicitud de generación de documento, el CRM indicará los datos necesarios acordes a la plantilla utilizada dentro del parámetro items, mediante el sistema *clave/valor*.

Respuesta:

Los datos obtenidos consistirán en:

- el cuerpo de la *Respuesta*, que indicará el código único del documento generado:

```
1493028491652R956
```

- el código HTTP del estado de la operación:

```
200
```

Rechazar el documento firmado

Los datos para realizar la petición de rechazo son:

URL de la Solicitud (rechazar documento):

La URL de la solicitud será:

```
https://sandbox.viafirma.com/documents/api/v3/check/reject/{messageCode}/{checkCode}
```

Además, indicaremos que la petición HTTP es de tipo PUT.

Parámetros:

Los parámetros aceptados son:

- messageCode *(requerido)*: código único del documento (ejemplo: 1493028491652R956).
- checkCode *(requerido)*: código de la acción que vamos a rechazar (ejemplo: 1493028491652R956P001C001).
- comment *(opcional)*: texto con el motivo de rechazo (ejemplo: datos incorrectos).
- validateCode *(opcional)*: código de validación para proteger la acción a realizar (ejemplo: k3W28)

Respuesta:

Los datos obtenidos consistirán en:

- el cuerpo de la *Respuesta*, indicará el estado del documento, así como los datos de la acción realizada:

```
{
  "status": "REJECTED",
  "code": "1493028491652R956P001C001",
  "signature": {
    "type": "SERVER",
    "helpText": "Firma del servicio",
    "typeFormatSign": "PADES_B"
  }
}
```

- el código HTTP del estado de la operación:

```
200
```

Ejemplo de obtención de documentos por estado

A continuación se detallarán los datos necesarios para realizar la integración que permita obtener el listado de documentos en un estado concreto. Podemos obtener los documentos asociados a un usuario existente, o bien los documentos visibles a todos los usuarios pertenecientes a un determinado grupo.

Documentos asignados a un usuario

Para obtener un listado de los documentos en un determinado estado y asignados a un usuario existente tendremos que realizar la siguiente petición:

URL de la Solicitud:

La URL de la solicitud será:

```
https://sandbox.viafirma.com/documents/api/v3/messages/{status}/user/{userCode}
```

Además, indicaremos que la petición HTTP es de tipo GET.

Parámetros:

Los parámetros requeridos son:

- status: estado en el que se encuentran los documentos (ejemplo: WAITING).
- userCode: código del usuario al que están asignados los documentos (ejemplo: user1234).

Respuesta:

Los datos obtenidos consistirán en:

- el cuerpo de la *Respuesta*, que contendrá el listado con los documentos obtenidos:

```
[
    {
        "messageCode": "1493028491652R956",
        "status": "WAITING",
        "userCode" : "user1234",
        "groupCode" : "groupA",
        "templateCode": "template_example",
        "creationDate": 1495176157551
    },
    {
        "messageCode": "1472093891409R34",
        "status": "WAITING",
        "userCode" : "user1234",
```

```
        "templateCode": "template_example",
        "creationDate": 1495176157802
    },
    {
        "messageCode": "147739024511R993",
        "status": "WAITING",
        "userCode" : "user1234",
        "groupCode" : "groupB",
        "templateCode": "template_example",
        "creationDate": 1495176157223
    }
]
```

- el código HTTP del estado de la operación:

```
200
```

Documentos asignados a un grupo

Para obtener un listado de los documentos en un determinado estado y asignados a un grupo existente tendremos que realizar la siguiente petición:

URL de la Solicitud:

La URL de la solicitud será:

```
https://sandbox.viafirma.com/documents/api/v3/messages/{status}/group/{groupCode}
```

Además, indicaremos que la petición HTTP es de tipo GET.

Parámetros:

Los parámetros requeridos son:

- status: estado en el que se encuentran los documentos (ejemplo: WAITING).
- groupCode: código del grupo al que están asignados los documentos (ejemplo: groupA).

Respuesta:

Los datos obtenidos consistirán en:

- el cuerpo de la *Respuesta*, que contendrá el listado con los documentos obtenidos:

```
[
```

```
    {
        "messageCode": "14930284916 52R956",
        "status": "WAITING",
        "userCode" : "user1234",
        "groupCode" : "groupA",
        "templateCode": "template_example",
        "creationDate": 1495176157551
    },
    {
        "messageCode": "1472093891409R34",
        "status": "WAITING",
        "userCode" : "user5678",
        "groupCode" : "groupA",
        "templateCode": "template_example",
        "creationDate": 1495176157802
    },
    {
        "messageCode": "147739024511R993",
        "status": "WAITING",
        "groupCode" : "groupA",
        "templateCode": "template_example",
        "creationDate": 1495176157223
    }
]
```


- el código HTTP del estado de la operación:

```
200
```

Firma Web desde link

Este flujo está pensado para poder firmar una petición desde un navegador web y sin necesidad de abrir sesión.

Para su uso, tenemos dos opciones:

- Enviar por correo electrónico un enlace que redirecciona a la página web pública donde se realizará la firma web.
- Recuperar el mensaje a través del servicio message y localizar el link de la firma web en el json obtenido.

En caso de que se requiera aprobación, el usuario deberá confirmar o rechazar dicha aprobación (*Ver la sección Servicios para aprobación más abajo*).

Por tanto, el ciclo de vida de este flujo se resumen en los siguientes pasos:

- se recibe nuevo mensaje,
- se envía el enlace por correo electrónico o recuperamos el link a través del servicio message,
- se firma la petición desde el navegador web,
- si la petición requiere aprobación, desde el backend o el backoffice integrado se confirma o rechaza la aprobación,
- si la petición no requiere ninguna aprobación, el flujo finaliza.

Mensaje de Ejemplo

nota: puedes parametrizar el siguiente JSON de ejemplo y usarlo tal y como se explica en el capítulo 2 "Testing" a través de la UI habilitada en el backend de viafirma documents.

Envío de enlace por correo electrónico

```
{
  "workflow" : {
    "code" : "EX006",
    "type" : "WEB"
  },
  "notification" : {
    "text" : "Firma biométricas simple con check en backend",
    "detail" : "Check en backend requerido",
    "sharedLink" : {
      "appCode" : "com.viafirma.documents",
      "email" : "enduser@yopmail.com",
      "subject" : "Asunto del correo"
    }
  },
  "document" : {
    "templateCode" : "firma_web_test",
    "items" : [ {
      "key" : "KEY_01",
      "value" : "Test value"
    }, {
      "key" : "KEY_02",
      "value" : "Test value"
    }, {
```

```
      "key" : "KEY_03",
      "value" : "Test value"
    }, {
      "key" : "KEY_04",
      "value" : "Test value"
    } ]
  },
  "policies": [
      {
        "evidences": [
          {
            "type": "SIGNATURE",
            "helpText": "Firma del usuario",
            "typeFormatSign": "XADES_B"
          }
        ],
        "signatures": [
          {
            "type": "SERVER",
            "helpText": "Firma del servicio",
            "typeFormatSign": "PADES_B"
          }
        ]
      }
    ]
}
```

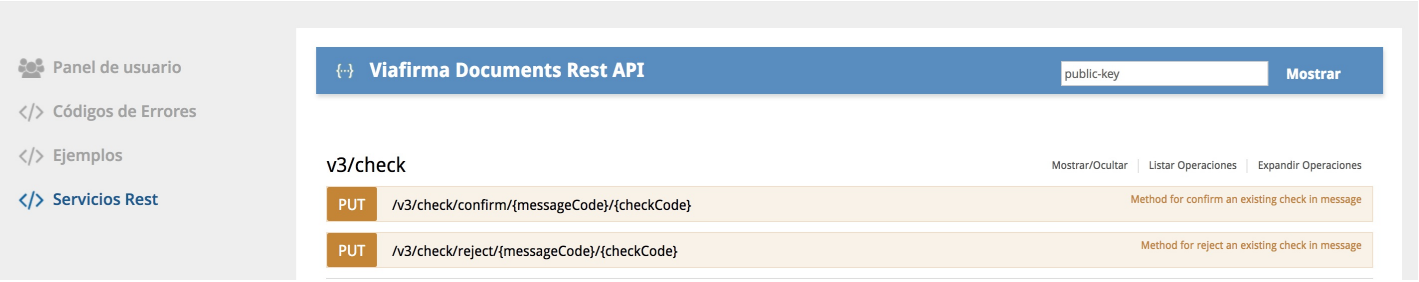
Generación de enlace sin correo electrónico

```
{
  "workflow" : {
    "code" : "EX006",
    "type" : "WEB"
  },
  "notification" : {
    "text" : "Firma biométricas simple con check en backend",
    "detail" : "Check en backend requerido",
    "sharedLink" : {
      "appCode" : "com.viafirma.documents",
    }
  },
  "document" : {
    "templateCode" : "firma_web_test",
    "items" : [ {
      "key" : "KEY_01",
      "value" : "Test value"
    }, {
      "key" : "KEY_02",
      "value" : "Test value"
    }, {
      "key" : "KEY_03",
      "value" : "Test value"
    }, {
      "key" : "KEY_04",
      "value" : "Test value"
    } ]
  },
  "policies": [
      {
        "evidences": [
          {
            "type": "SIGNATURE",
            "helpText": "Firma del usuario",
```

```
            "typeFormatSign": "XADES_B"
          }
        ],
        "signatures": [
          {
            "type": "SERVER",
            "helpText": "Firma del servicio",
            "typeFormatSign": "PADES_B"
          }
        ]
      }
    ]
}
```

Servicios para aprobación

Los servicios para confirmar o rechazar una aprobación son públicos, de forma que son accesibles desde la aplicación con usuario logado, en la parte de documentación de los servicios REST, introduciendo la clave public-key.



Confirmar

Para confirmar la aprobación, será necesario indicar el código de mensaje, así como el código de la aprobación. Además, se podrá requerir un código de validación configurable desde la política de la petición.

Rechazar

Para rechazar la aprobación, será necesario indicar el código de mensaje, así como el código de la aprobación. De forma opcional, el usuario podrá indicar un comentario con el motivo de rechazo. Además, se podrá requerir un código de validación, configurable desde la política de la petición.

Flujo de ejemplo para refirmar un documento

1: hacemos la primera firma

POST:

```
{
  "notification" : {
    "text" : "Ejemplo refirmado PDF",
    "detail" : "PDF firmado en dos flujos distintos",
    "devices" : [ {
      "appCode" : "com.viafirma.mobile.ios.documents",
      "code" : "benito",
      "type" : "IOS",
      "userCode" : "benito"
    } ]
  },
  "document" : {
    "templateCode" : "sample_refirmado",
    "items" : [ {
      "key" : "fecha",
      "value" : "21/02/2017",
      "type" : "todayText",
      "size" : "",
      "update" : ""
    } ]
  }
}
```

RESPONSE:

```
Code 200
1487689635773R368
```

Recuperamos el PDF

(damos por hecho de que ya tenemos callback de firma OK y recuperamos el PDF firmado con un simple GET.

GET:

```
https://sandbox.viafirma.com/documents/api/v3/documents/download/signed/1487689635773R368
```

RESPUESTA:

```
200
```

con el siguiente body:

```
{
  "link": "https://sandbox.viafirma.com/documents/download?id=1487689635773R368.pdf/f9017f60-e186-43f9-b87c
  "md5": "dd2ac57be733535154746538026bedb8",
  "fileName": "1487689635773R368.pdf",
  "expires": 1487690474552
}
```

Con esta respuesta ya tendremos a nuestra disposicin el PDF que contiene la primera firma.

2A: refirmamos el PDF: alternativa Type B64

Convertimos el PDF a B64

Para poder pasar el PDF que incluye la primera firma al servicio, debemos antes codificarlo a B64. Para el ejemplo, se puede usar el siguiente encoder online:

http://www.motobit.com/util/base64-decoder-encoder.asp

Segunda firma del PDF

En este caso, haremos un POST con una variación en el servicio, especificando que el PDF lo vamos a pasar en B64, tal y como se muestra en el siguiente detalle:

```
{
  "document" : {
    "templateReference" : "JVBERi0xLjQKJeLjz9MKMSAwIG9iago8PC9UeXBlL1hPY...(acortado)...",
    "templateType" : "base64"
  }
}
```

POST:

nota: este JSON de ejmplo tiene acortado el B64 del PDF para facilitar su lectura. Debes usar el B64 completo para poder seguir el ejemplo. En el siguiente enlace puedes descargar uno completo: post_sample_b64_reference

```
{
  "notification" : {
    "text" : "Demo 001",
    "detail" : "Ejemplo de firma de un documento disponible en una url",
    "devices" : [ {
      "appCode" : "com.viafirma.mobile.ios.documents",
      "code" : "benito",
```

```
        "type" : "IOS",
        "userCode" : "benito"
      } ]
    },
    "document" : {
      "templateReference" : "JVBERi0xLjQKJeLjz9MKMSAwIG9iago8PC9UeXBlL1hPY...(acortado)...",
      "templateType" : "base64"
    },
    "policies" : [ {
      "evidences" : [ {
        "type" : "SIGNATURE",
        "helpText" : "Segunda Firma",
        "typeFormatSign" : "XADES_B"
      } ],
      "signatures" : [ {
        "type" : "SERVER",
        "helpText" : "Server signature",
        "typeFormatSign" : "PADES_LTA"
      } ]
    } ]
  }
```

RESPUESTA:

```
200 - 1487691331472R594
```

A partir de este momento, el PDF ya ha llegado a un nuevo dispositivo para su firma, y cuando ésta se produzca, podremos recuperar el PDF que ya incluirá 2 firmas, y para ello repetiremos el GET para obtener el documento firmado usando el nuevo código de mensaje recibido en nuestro último POST:

GET:

```
https://sandbox.viafirma.com/documents/api/v3/documents/download/signed/1487691331472R594
```

2B: refirmamos el PDF: alternativa Template Type URL

Con *Type URL* pasaremos una URI al servicio, donde estará el PDF que deseamos refirmar. A diferencia del *Type b64* aquí no será necesario trabajar con el PDF inicial ni convertirlo a B64. Esta modalidad de firma sólo es compatible si la configuración de seguridad en la descarga de documentos firmados está activada y debidamente configurada, ya que el servicio para obtener documentos firmados incluye un ciclo de vida basado en tiempo y/o número de descargas.

Para ello partimos de la base en el que el integrador YA tiene el PDF con la primera firma. Este PDF lo podría publicar en una URI gestionado por su propio backoffice, o bien usar la URI facilitada por el backend de documents teniendo en cuenta la restricción de seguridad descrtia en el párrafo anterior.

Para facilitar las pruebas, se pueden usar URIs generadas por repositorios públicos, por ejemplo dropbox.

```
{
```

```
  "document" : {
    "templateReference" : "https://sandbox.viafirma.com/documents/download?id=1487689635773R368.pdf/f9017f6(
    "templateType" : "url"
  }
}
```

El post quedaría como sigue:

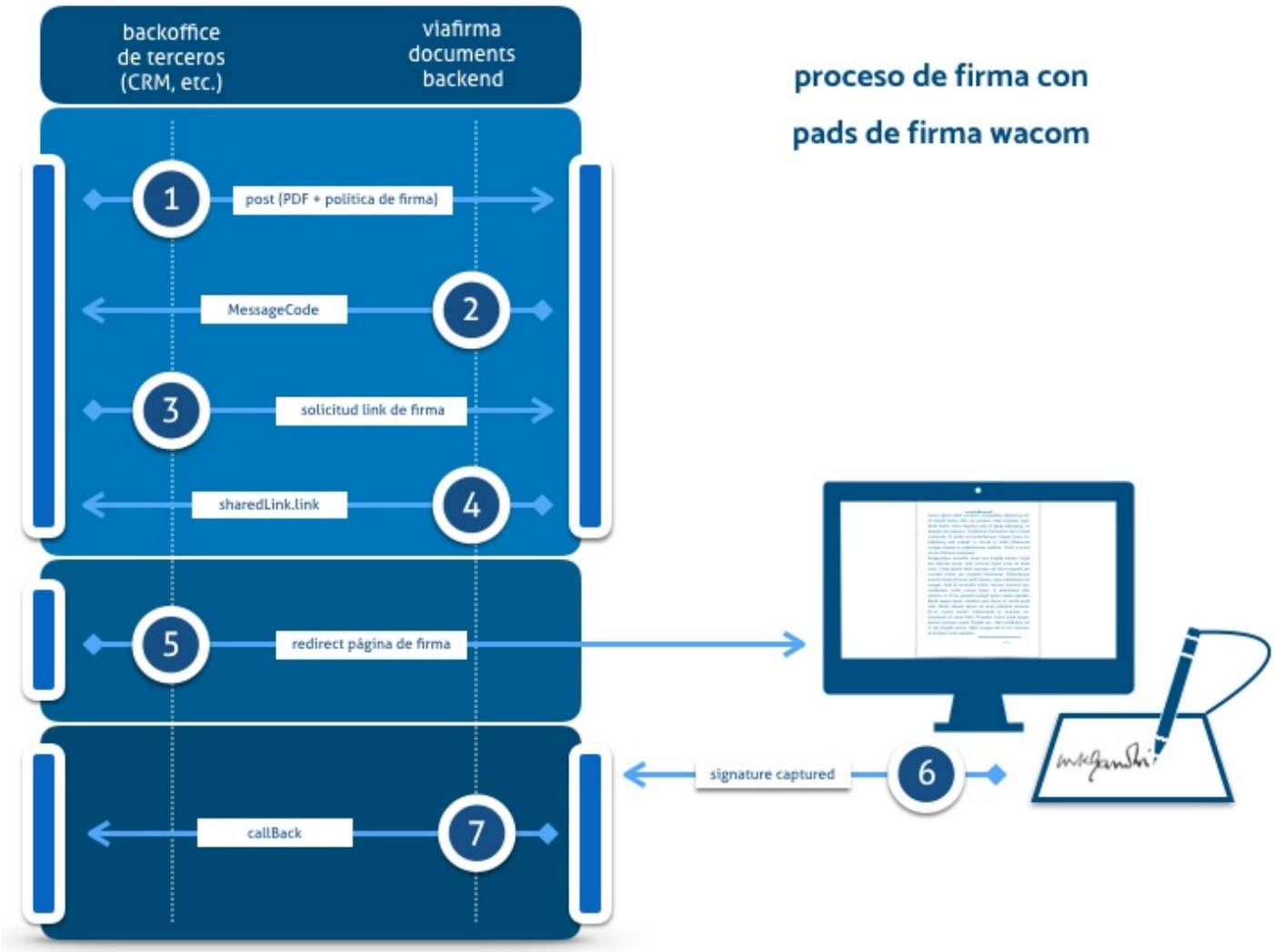
POST:

```
{
  "notification" : {
    "text" : "Demo 001",
    "detail" : "Ejemplo de firma de un documento disponible en una url",
    "devices" : [ {
      "appCode" : "com.viafirma.mobile.ios.documents",
      "code" : "benito",
      "type" : "IOS",
      "userCode" : "benito"
    } ]
  },
  "document" : {
    "templateReference" : "https://sandbox.viafirma.com/documents/download?id=1487689635773R368.pdf/f9017f6(
    "templateType" : "url"
  },
  "policies" : [ {
    "evidences" : [ {
      "type" : "SIGNATURE",
      "helpText" : "Segunda Firma",
      "typeFormatSign" : "XADES_B"
    } ],
    "signatures" : [ {
      "type" : "SERVER",
      "helpText" : "Server signature",
      "typeFormatSign" : "PADES_LTA"
    } ]
  } ]
}
```

Firmar con WACOM: paso a paso

En esta guía se explicará el paso a paso para la integración de un sistema de terceros, por ejemplo un CRM, con viafirma documents y con un caso de uso "Firmar desde Wacom".

Secuencia de trabajo



Secuencia 1: solicitamos servicio de firma

Hacemos POST del servicio MESSAGE, enviando nuestro PDF y el identificador de políticas de firma que deseemos.

- Servicio: `https://sandbox.viafirma.com/documents/api/v3/messages`
- Método: POST
- Seguridad: OAuth v1.0

Resuesta esperada:

- Código de respuesta: 200
- Body de la respuesta: {messageCode}

Ejemplo JSON para el POST:

```
{
  "groupCode" : "myGroupCode",
  "workflow" : {
    "code" : "EX006"
  },
  "notification" : {
    "text" : "card title printed on signature request",
    "detail" : "card detail printed on signature request",
    "sharedLink" : {
      "appCode" : "com.viafirma.documents"
    }
  },
  "document" : {
    "policyCode" : "myPolicy_id",
    "templateType" : "base64",
    "templateReference" : "****HERE YOUR PDF IN BASE64 FORMAT*****"
  },
  "callbackMails" : "client@viafirma.com,sales@viafirma.com,others@viafirma.com"
}
```

descripción de atributos

- groupCode: es opcional, y al indicarlo permitirá que cualquier usuario que pertenezca al grupo indicado podrá hacer seguimiento de la solicitud. Además nos permitirá asignar un diseño a la página de firma; para ello, habrá que crear un estilo llamado igual al grupo; los estilos se gestionan en el backend de viafirma documents, debiendo contar con permisos de administrador.
- workflow.code: para la firma con WACOM usaremmos el código "EX006".

- notification.text: es opcional, y se usa para indicar un título a la solicitud de firma.

- notification.detail: es opcional, y se usa para la descripción extendida a la solicitud de firma.
- notification.sharedLink.appCode: aquí indicaremos el identificador de la app que se usará para la firma; podrá ser personalizada para cada cliente, pero por defecto se debe usar la aplicación escritorio de viafirma documents, con soporte a wacom. Para ello, indicar el valor "com.viafirma.documents".

- document.policyCode: aquí indicaremos el identificador de la política de firma que se aplicará al PDF remitido. Las políticas se diseñan y gestionan en el backend de viafirma documents. En caso de no contar con políticas gestionadas por viafirma, consultar en la documentación cómo incluirlas manualmente en el POST del servicio.

- document.templateType: se permiten hasta tres valores: "base64", "template" y "url"; si el PDF es generado por un sistema externo, por ejemplo un CRM, indicaremos "base64"; si el PDF estuviera disponible en un URL como un recurso directo (URI), entonces usaremos el tipo "URL"; en caso de gestionar las plantillas, en formatos "docx, odt o pdf", desde el propio backend de viafirma documents, con la ayuda del diseñador de plantillas de viafirma, entonces usaremos el valor "template".

- document.templateReference: dependiendo del templateType usado, la referencia esperada para cada caso será

distintas: un base64, una URL en formato http o https o bien el código de la plantilla gestionada en el propio backend de viafirma.

- callbackMail: es opcional, y podremos indicar una o varias cuentas de email a las que se enviará el documento firmado una vez finalzada la operación.

- callbackResponse: es opcional, y podremos indicar una URL a la que se hará POST de forma automática, enviando un objeto MESSAGE con toda la información necesaria para cualquier integrador que desee automatizar acciones de negocio.

Secuencia 2: montar página de firma

A partir del messageCode obtenido tras hacer el POST con la solicitud de firma vamos a realizar un GET para obtener la siguiente información:

- Servicio: `https://sandbox.viafirma.com/documents/api/v3/messages/{messageCode}`
- Método: GET
- Seguridad: OAuth v1.0

Resuesta esperada:

- Código de respuesta: 200
- Body de la respuesta: objeto MESSAGE (en formato JSON)

del objeto MESSAGE obtenido, nos interesan dos datos: link de firma y estado, y los obtendremos en los valores de los siguientes atributos:

- notification.sharedLink.link
- workflow.current

Con el valor del link autogenerado por viafirma, se montará una vista web, con un permalink único, y que incluirá todos los componentes de firma necesarios según se hayan definidos en la política de firma.

En nuestro caso, el usuario verá un enlace para abrir o descargar e instalar por primera vez la app de escritorio que interacturará con el pad de firma WACOM.

Una vez se haya procedido a la firma del documento, y en caso de que se haya definido en el POST del servicio una URL para hacer callback e informar de la finalización del proceso, el estado pasará a ser "RESPONSED", estado que ya permitirá consumir el servicio que nos devolverá el documento firmado.

Secuencia 3: obtener el documento firmado

En caso de no haber definido una URL de respuesta automática, se deberá consultar el estado del mensaje hasta comprobar que su estado ya es "RESPONSED" para poder consumir el siguiente servicio:

- Servicio: `https://sandbox.viafirma.com/documents/api/v3/documents/download/signed/{messageCode}`
- Método: GET
- Seguridad: OAuth v1.0

Resuesta esperada:

- Código de respuesta: 200
- Body de la respuesta: objeto DOWNLOAD (en formato JSON), donde dispondremos de la siguiente información:

```
{
  "link": "temporal link for downloading",
  "md5": "signed-file hash",
  "fileName": "filename",
  "expires": "timestamp in miliseconds format"
}
```

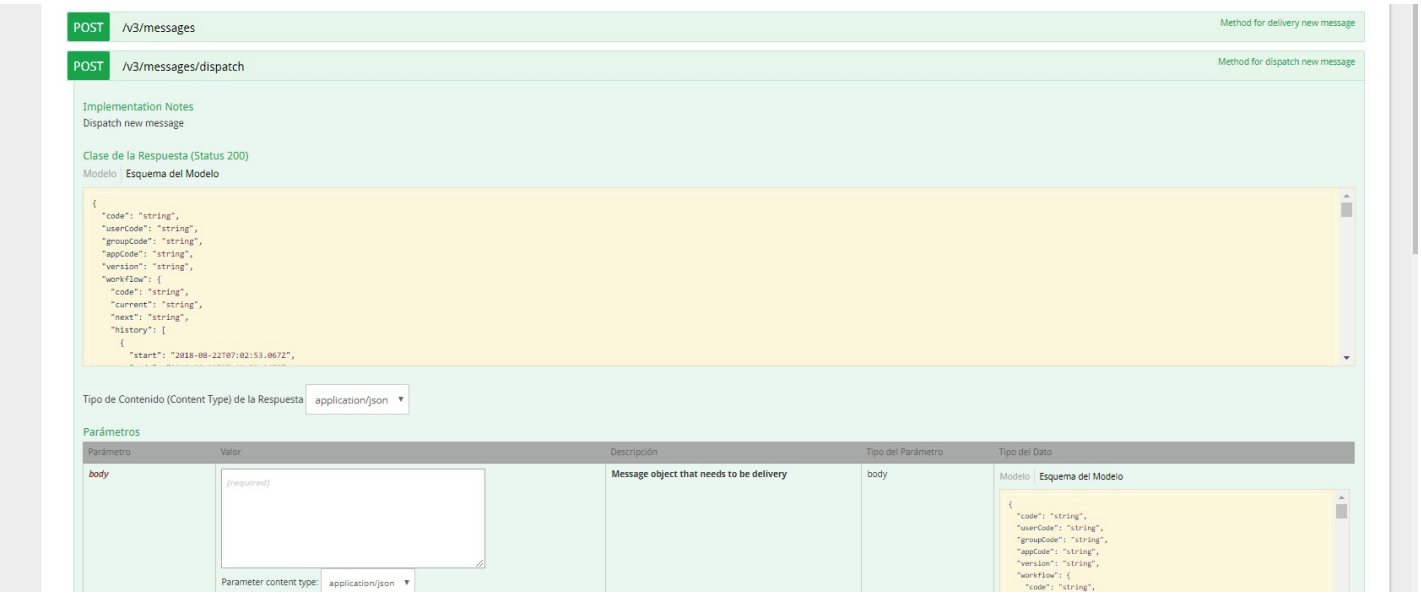
Firmar con documents desktop por protocolo

A continuación se explica los pasos a seguir para realizar una firma con un dispositivo wacom compatible con documents o certificado digital local desde una aplicación externa mediante la llamada por protocolo a viafirma documents desktop.

Vídeo - Demostración de firma con wacom por protocolo

1. Envío de petición de firma

Desde la aplicación cliente se llamará al servicio rest dispatch para crear una petición de firma:



2. Recuperación del valor del atributo scheme

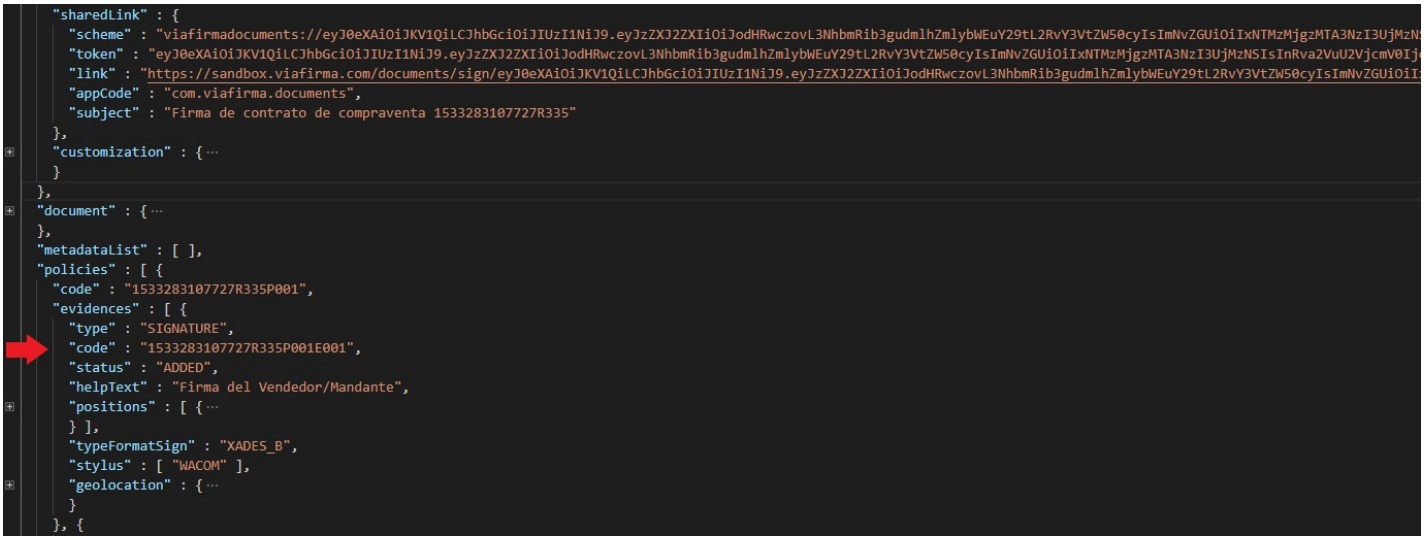
El servicio anterior devuelve el json del modelo message donde localizamos el atributo scheme dentro el objeto sharedLink que a su vez está dentro de notification.

El valor de scheme es el que utlizaremos para componer la llamada a viafirma documents desktop por protocolo. En este caso:

viafirmadocuments://eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJzZXJ2ZXIiOiJodHRwczovL3NhbmRib3gudmlhZmlybWEuY29tL2RvY3VtZW50cyIsImNvZGUiOiIxNTMzMjgzMTA3NzI3UjMzNSIsInRva2VuU2VjcmV0IjoiN2U4ZTk5MDA4MmFmNDE3NzgzYmEyMDBmNjNiYjkyMTgiLCJ0b2tlbiI6IlRLMTUzMzI4MzEwNzcyN1lzMzUifQ.sbwGmK4cdS_OI29FMFRsJd3UTz-CKeE785nyJbBuIG0

3. Composición de la llamada

Para componer la llamada final a la apliación viafirma documents desktop por protocolo se debe concatenar al valor scheme el código de la evidencia tipo wacom o certificado obtenidos dentro del objeto policies del json:



Para este ejemplo, obtenemos los siguientes atributos del json:

- scheme : viafirmadocuments://eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJzZXJ2ZXIiOiJodHRwczovL3NhbmRib3gudmlhZmlybWEuY29tL2RvY3VtZW50cyIsImNvZGUiOiIxNTMzMjgzMTA3NzI3UjMzNSIsInRva2VuU2VjcmV0IjoiN2U4ZTk5MDA4MmFmNDE3NzgzYmEyMDBmNjNiYjkyMTgiLCJ0b2tlbiI6IlRLMTUzMzI4MzEwNzcyN1lzMzUifQ.sbwGmK4cdS_OI29FMFRsJd3UTz-CKeE785nyJbBuIG0,
- code : 1533283107727R335P001E001 (obtenido del objeto evidencies (para firma wacom) o signatures (para firma con certificado digital))

  Componemos la llamada scheme:code que tendrá que ejecutarse al pulsar sobre la acción "firmar" desde la aplicación cliente: viafirmadocuments://eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJzZXJ2ZXIiOiJodHRwczovL3NhbmRib3gudmlhZmlybWEuY29tL2RvY3VtZW50cyIsImNvZGUiOiIxNTMzMjgzMTA3NzI3UjMzNSIsInRva2VuU2VjcmV0IjoiN2U4ZTk5MDA4MmFmNDE3NzgzYmEyMDBmNjNiYjkyMTgiLCJ0b2tlbiI6IlRLMTUzMzI4MzEwNzcyN1lzMzUifQ.sbwGmK4cdS_OI29FMFRsJd3UTz-CKeE785nyJbBuIG0:1533283107727R335P001E001

4. Firma en documents desktop

La llamada por protocolo anterior abrirá la aplicación viafirma documents desktop, donde se procederá a la firma con el dispositivo wacom o seleccionando el certificado digital según el caso.

Cuando se complete la firma (o la última en caso de que haya varias), se finaliza la petición en documents y se ejecuta

el callback (si lo hubira) definido en el POST del paso 1.

Ejemplos de integración en PHP

Para poder utilizar estos ejemplo necesitamos disponer de credenciales para acceder a la capa de servicios rest.

En estos ejemplos hacemos uso de la librería oauth-php para gestionar la seguridad de acceso a los servicios con OAuth 1.0, puedes acceder a esta librería en https://code.google.com/archive/p/oauth-php/

```
require_once dirname(__FILE__) . '/library/OAuthRequestSigner.php';

define("DOCUMENTS_API_URL", "https://sandbox.viafirma.com/documents/api/v3");
define("DOCUMENTS_CONSUMER_KEY", "com.viafirma.documents.XXXXXXX");
define("DOCUMENTS_CONSUMER_SECRET", "XXXXXXXXXXXX");
```

Comprobar acceso a los servicios rest

```
function system_alive ()
{
    error_reporting(E_ALL);

    header('Content-Type: text/plain; charset=utf-8');

    echo "Documents OAuth 1.0a Client\n\n";
    echo "See also: http://doc.viafirma.com/documents\n\n";

    $url=DOCUMENTS_API_URL."/system/alive";
    echo "URL: ".$url."\n";

    //  Initiate curl
    $ch = curl_init();

    // Disable SSL verification
    curl_setopt($ch, CURLOPT_SSL_VERIFYPEER, false);

    // Will return the response, if false it print the response
    curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);

    // Set the url
    curl_setopt($ch, CURLOPT_URL,$url);

    // Execute
    $result=curl_exec($ch);
    echo prettyPrint($result);

    // Closing
    curl_close($ch);
}
```

Enviar una nueva petición

```
function send_message ()
```

```
{
    error_reporting(E_ALL);

    header('Content-Type: text/plain; charset=utf-8');

    $url=DOCUMENTS_API_URL."/messages";
    echo "URL: ".$url."\n";

    OAuthStore::instance('MySQL', array('conn'=>false));
    $req = new OAuthRequestSigner($url, 'POST');
    $fecha = new DateTime();
    $secrets = array(
                'consumer_key'      => DOCUMENTS_CONSUMER_KEY,
                'consumer_secret'   => DOCUMENTS_CONSUMER_SECRET,
                'token'             => '',
                'token_secret'      => '',
                'signature_methods' => array('HMAC-SHA1'),
                'nonce'             => '3jd834jd9',
                'timestamp'         => $fecha->getTimestamp(),
                );
    $req->sign(0, $secrets);

    // POST
    $string_json = file_get_contents("../message.json");
    $ch = curl_init($url);
    curl_setopt($ch, CURLOPT_SSL_VERIFYPEER, false);
    curl_setopt($ch, CURLOPT_CUSTOMREQUEST, "POST");
    curl_setopt($ch, CURLOPT_POSTFIELDS, $string_json);
    curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);

    // OAuth Header
    $headr = array();
    $headr[] = 'Content-Length: ' . strlen($string_json);
    $headr[] = 'Content-type: application/json';
    $headr[] = ''.$req->getAuthorizationHeader();
    curl_setopt($ch, CURLOPT_HTTPHEADER,$headr);

    $result = curl_exec($ch);
    echo "MessageCode: ".$result;

    // Closing
    curl_close($ch);
}
```

Recuperar información de una petición

```
function get_message ($messageCode = '')
{
    error_reporting(E_ALL);

    header('Content-Type: text/plain; charset=utf-8');

    $url=DOCUMENTS_API_URL."/messages/".$messageCode;
    echo "URL: ".$url."\n";

    OAuthStore::instance('MySQL', array('conn'=>false));
    $req = new OAuthRequestSigner($url, 'GET');
    $fecha = new DateTime();
    $secrets = array(
                'consumer_key'      => DOCUMENTS_CONSUMER_KEY,
                'consumer_secret'   => DOCUMENTS_CONSUMER_SECRET,
                'token'             => '',
                'token_secret'      => '',
```

```
                'signature_methods' => array('HMAC-SHA1'),
                'nonce'             => '3jd834jd9',
                'timestamp'         => $fecha->getTimestamp(),
                );
    $req->sign(0, $secrets);

    //  Initiate curl
    $ch = curl_init();

    // Disable SSL verification
    curl_setopt($ch, CURLOPT_SSL_VERIFYPEER, false);

    // Will return the response, if false it print the response
    curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);

    // Set the url
    curl_setopt($ch, CURLOPT_URL,$url);

    // OAuth Header
    $headr = array();
    $headr[] = 'Content-length: 0';
    $headr[] = 'Content-type: application/json';
    $headr[] = ''.$req->getAuthorizationHeader();
    curl_setopt($ch, CURLOPT_HTTPHEADER,$headr);

    // Execute
    $result=curl_exec($ch);
    echo prettyPrint($result);

    // Closing
    curl_close($ch);
}
```

Recuperar un documento firmado

```
function download_signed ($messageCode = '')
{
    error_reporting(E_ALL);

    header('Content-Type: text/plain; charset=utf-8');

    $url=DOCUMENTS_API_URL."/documents/download/signed/".$messageCode;
    echo "URL: ".$url."\n";

    OAuthStore::instance('MySQL', array('conn'=>false));
    $req = new OAuthRequestSigner($url, 'GET');
    $fecha = new DateTime();
    $secrets = array(
                'consumer_key'      => DOCUMENTS_CONSUMER_KEY,
                'consumer_secret'   => DOCUMENTS_CONSUMER_SECRET,
                'token'             => '',
                'token_secret'      => '',
                'signature_methods' => array('HMAC-SHA1'),
                'nonce'             => '3jd834jd9',
                'timestamp'         => $fecha->getTimestamp(),
                );
    $req->sign(0, $secrets);

    //  Initiate curl
    $ch = curl_init();

    // Disable SSL verification
    curl_setopt($ch, CURLOPT_SSL_VERIFYPEER, false);
```

```
    // Will return the response, if false it print the response
    curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);

    // Set the url
    curl_setopt($ch, CURLOPT_URL,$url);

    // OAuth Header
    $headr = array();
    $headr[] = 'Content-length: 0';
    $headr[] = 'Content-type: application/json';
    $headr[] = ''.$req->getAuthorizationHeader();
    curl_setopt($ch, CURLOPT_HTTPHEADER,$headr);

    // Execute
    $result=curl_exec($ch);
    echo prettyPrint($result);

    // Closing
    curl_close($ch);
}
```

Recuperar la lista de dispositivos de un usuario

```
function get_user_devices ($userCode = '')
{
    error_reporting(E_ALL);

    header('Content-Type: text/plain; charset=utf-8');

    $url=DOCUMENTS_API_URL."/devices/user/".$userCode;
    echo "URL: ".$url."\n";

    OAuthStore::instance('MySQL', array('conn'=>false));
    $req = new OAuthRequestSigner($url, 'GET');
    $fecha = new DateTime();
    $secrets = array(
                'consumer_key'      => DOCUMENTS_CONSUMER_KEY,
                'consumer_secret'   => DOCUMENTS_CONSUMER_SECRET,
                'token'             => '',
                'token_secret'      => '',
                'signature_methods' => array('HMAC-SHA1'),
                'nonce'             => '3jd834jd9',
                'timestamp'         => $fecha->getTimestamp(),
                );
    $req->sign(0, $secrets);

    //  Initiate curl
    $ch = curl_init();

    // Disable SSL verification
    curl_setopt($ch, CURLOPT_SSL_VERIFYPEER, false);

    // Will return the response, if false it print the response
    curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);

    // Set the url
    curl_setopt($ch, CURLOPT_URL,$url);

    // OAuth Header
    $headr = array();
    $headr[] = 'Content-length: 0';
    $headr[] = 'Content-type: application/json';
```

```
    $headr[] = ''.$req->getAuthorizationHeader();
    curl_setopt($ch, CURLOPT_HTTPHEADER,$headr);

    // Execute
    $result=curl_exec($ch);
    echo prettyPrint($result);

    // Closing
    curl_close($ch);
}
```

Rechazar una petición

```
function reject_message ($messageCode = '', $comment = '')
{
    error_reporting(E_ALL);

    header('Content-Type: text/plain; charset=utf-8');

    $url=DOCUMENTS_API_URL."/messages/reject/".$messageCode;
    echo "URL: ".$url."\n";

    $data = array(
        'comment' => $comment,
    );
    $params=http_build_query($data);
    OAuthStore::instance('MySQL', array('conn'=>false));
    echo "URL: ".$url."?comment=".rawurlencode($comment);
    $req = new OAuthRequestSigner($url."?comment=".rawurlencode($comment), 'PUT');
    $fecha = new DateTime();
    $secrets = array(
                'consumer_key'      => DOCUMENTS_CONSUMER_KEY,
                'consumer_secret'   => DOCUMENTS_CONSUMER_SECRET,
                'token'             => '',
                'token_secret'      => '',
                'signature_methods' => array('HMAC-SHA1'),
                'nonce'             => '3jd834jd9',
                'timestamp'         => $fecha->getTimestamp(),
                );
    $req->sign(0, $secrets);

    // PUT
    $ch = curl_init($url);
    curl_setopt($ch, CURLOPT_SSL_VERIFYPEER, false);
    curl_setopt($ch, CURLOPT_CUSTOMREQUEST, "PUT");
    curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);
    curl_setopt($ch, CURLOPT_POSTFIELDS, http_build_query($data));

    // OAuth Header
    $headr = array();
    $headr[] = 'Content-type: application/x-www-form-urlencoded';
    $headr[] = ''.$req->getAuthorizationHeader();
    curl_setopt($ch, CURLOPT_HTTPHEADER,$headr);

    $result=curl_exec($ch);
    echo prettyPrint($result);

    // Closing
    curl_close($ch);
}
```

Funciones de utilidad

```php
function prettyPrint( $json )
{
    $result = '';
    $level = 0;
    $in_quotes = false;
    $in_escape = false;
    $ends_line_level = NULL;
    $json_length = strlen( $json );

    for( $i = 0; $i < $json_length; $i++ ) {
        $char = $json[$i];
        $new_line_level = NULL;
        $post = "";
        if( $ends_line_level !== NULL ) {
            $new_line_level = $ends_line_level;
            $ends_line_level = NULL;
        }
        if ( $in_escape ) {
            $in_escape = false;
        } else if( $char === '"' ) {
            $in_quotes = !$in_quotes;
        } else if( ! $in_quotes ) {
            switch( $char ) {
                case '}': case ']':
                    $level--;
                    $ends_line_level = NULL;
                    $new_line_level = $level;
                    break;

                case '{': case '[':
                    $level++;
                case ',':
                    $ends_line_level = $level;
                    break;

                case ':':
                    $post = " ";
                    break;

                case " ": case "\t": case "\n": case "\r":
                    $char = "";
                    $ends_line_level = $new_line_level;
                    $new_line_level = NULL;
                    break;
            }
        } else if ( $char === '\\' ) {
            $in_escape = true;
        }
        if( $new_line_level !== NULL ) {
            $result .= "\n".str_repeat( "\t", $new_line_level );
        }
        $result .= $char.$post;
    }

    return $result;
}
```

Integración del visor de formularios

Para poder integrar la visualización de formularios generados desde el diseñador de formularios de viafirma Documents dentro de aplicaciones externas se puede usar este kit de integración Javascript.

Los requisitos para usarlo son, únicamente, la librería Javascript jQuery.

Aquí se puede encontrar un ejemplo básico de integración en el que se lee un fichero de formulario generado con el diseñador y se muestra en pantalla.

Paso a paso

Lo primero es generar un formulario usando el diseñador de formularios de viafirma Documents y exportarlo para guardarlo como un fichero `.json`.

Ese fichero JSON será leído posteriormente por el SDK del visor de formularios para pintarlo.

Esto se puede conseguir inyectando las dependencias del visor en cualquier página HTML.

Estas dependencias son, en cuanto a Javascript:

```
<script src="jquery.js"></script>
<script src="jfb.js"></script>
```

Y en cuanto a CSS:

```
<link rel="stylesheet" href="styles.css">
```

Una vez hecho esto, debemos definir un elemento HTML en el que se inyectará el formulario una vez procesado. Este elemento debe tener un `class` concreto:

```
<div class="jfb-form"></div>
```

Una vez completados estos requisitos, podemos inicializar el visor de formularios pasándole el contenido del fichero JSON del formulario que se necesite mostrar. La carga de este fichero queda completamente en manos del desarrollador, pudiendola realizar de la manera que más le convenga. Aquí un ejemplo cargando el JSON mediante AJAX:

```
$(function() {
    $.getJSON("form/test.json", function( data ) {
        jfb.init(data);
    });
});
```

La llamada a `jfb.init()` es la que inicia la visualización del formulario. Recibe el objeto JSON con la definición del formulario. Buscará el elemento HTML con `class="jfb-form"` y pintará dentro del formulario.

Obtener los valores introducidos en el formulario

El visor de formularios expone un método Javascript para obtener el contenido del formulario (y realizar las validaciones pertinentes).

```
var json_formulario_con_valores = JSON.parse(jfb.getStringifiedData());
```

Si la validación del formulario encuentra algún problema, el método `jfb.getStringifiedData()` devolverá la cadena `"error"`, y actualizará el formulario para mostrar los errores pertinentes.

Guía rápida de intregración

En esta guía veremos ejemplos básicos de integración con viafirma documents.

Cómo enviar un PDF a viafirma documents

Cuando un sistema externo necesita enviar un PDF para ser firmado a viafirma documents podrá hacerlo mediante dos mecanismos:

- enviando el PDF en base64.
- enviando una URL en la que el PDF está disponible para su descarga directa (URI).

Este comportamiento se define en las propiedades del objeto Documents, tal y como se detalla a continuación:

```
Document {
  templateReference (string, optional),
  templateType (string) = ['docx', 'odt', 'url', 'pdf', 'cache', 'base64'],
  policyCode (string, optional)
}
```

- Document.templateReference: aquí indicaremos el base64 del PDF o la URL en la que poder descargar el PDF.
- Document.templateType: usaremos los valores "base64" o "url" según corresponda.
- Document.policyCode: identificador de la política elegida para cada caso. Este identificador podrá ser gestionado desde el backend de viafirma documents; ver gestión de modelos > políticas. Este atributo es opcional, y si lo omitimos, tendremos que informar de forma explícita la política de firma deseada; para este caso consultar el uso de Políticas.

Ejemplo:

```
Document {
  templateReference:"https://www.viafirma.com/myPDF_sample.pdf",
  templateType:"url",
  policyCode:"myPolicyCode_sample_001"
}
```

El objeto Document admite otro tipo de configuración que podrá ser consultada en la definición del servicio. En este artículo sólo se han mencionado los atributos necesarios para poder enviar un PDF a viafirma documents mediante servicio.

¿Dónde podremos enviar el PDF para su firma?

Podrás optar por varios mecanismos para enviar el PDF al destinatario deseado en función de cada caso de negocio y en función de los dos modalidades principales de firma que permite viafirma documents:

- firmas presenciales
- firmas a distancia

enviar un PDF para firmas presenciales

Nos ayudaremos de dos mecanismos principalmente:

- apps iOS, Android y Windows UWP
- pad de firma Wacom

enviar un PDF para firmas a distancia

El PDF se mostrará embebido en un visor web disponible en una página web disponible en un link seguro y único. El link de acceso a esta página web podrá notificarse automáticamente al usuario final mediante tres mecanismos:

- SMS
- Email
- o bien, obtener el link de firma para montar la página de forma embebida en un sistema externo, por ejemplo, una web-wiew o iframe.

Aquí te explicamos cómo hace uso de ambos mecanismos:

- Enviar un PDF para firmas presenciales
- Enviar un PDF para firmas a distancia

Enviar PDF para firmas presenciales

Para enviar un PDF para firmas presenciales nos ayudaremos de dos mecanismos principalmente:

- apps iOS, Android y Windows UWP
- pad de firma Wacom

Para hacer uso de cualquiera de los mecanismos previstos para la FIRMA a DISTANCIA, tendremos que indicar el tipo WEB en el flujo de trabajo, o en su defecto para versiones anteriores a 3.6 el código EX06.

```
{
  "workflow" : {
    "code" : "EX001",
    "type" : "APP"
  }
}
```

- workflow.code: en versiones anteriores a 3.6 se usa para definir el tipo de flujo de trabajo, y en los mecanismos previstos para firmas presenciales, se debe usar el código EX001. Esto habilitará el envío de una notificación push al usuario y app seleccionados.
- workflow.type: en versiones 3.6 sustituye al atributo CODE, y en los mecanismos previstos para firmas a distancia, se debe usar el tipo APP. Ambos atributos se podrán usar conjuntamente para mantener compatibilidad hacia atrás en caso de usar modelos diseñados con versiones anteriores.

Una vez definido el tipo de flujo de trabajo que vamos a habilitar para la firma, se debe definir el DOCUMENTO que deseamos firmar. Para ello usaremos el ojbeto DOCUMENT, tal y como se detalla en capítulo "Cómo enviar un PDF".

Para este ejemplo usaremos una configuración básica de documento:

```
{
  "document": {
    templateReference:"https://www.viafirma.com/myPDF_sample.pdf",
    templateType:"url",
    policyCode:"myPolicyCode_sample_001"
  }
}
```

Con el flujo de trabajo y documento definido, ya sólo falta definir cómo vamos a comunicarnos con la app móvil que recibirá el documento, y para ello vamos a explicar la estructura del objeto NOTIFICATION.

```
Notification {
  text (string),
  detail (string),
  notificationType (string, optional) = ['PUSH_IOS', 'PUSH_ANDROID', 'CALLBACK', 'MAIL', 'SMS'],
  devices (Array[Device], optional)
}
```

objeto Notification:

- Notification.text: título de la notificación.
- Notification.detail: descripción de la notificación.
- Notification.notificationType: tipo de notificación elegida: 'PUSH_IOS', 'PUSH_ANDROID', 'CALLBACK', 'MAIL', 'SMS'
- Notification.devices: sólo en caso de notificaciones tipo iOS y Android.

objeto Device:

Para las notificaciones en procedimientos de firmas presenciales, donde usaremos las apps de viafirma documents, usaremos el objeto Device:

```
{
  "devices" : [ {
      "appCode" : "com.viafirma.documents",
      "code" : "customer_care_ipad01",
      "type" : "ANDROID",
      "userCode" : "jhon.doe@viafirma.com"
    } ]
}
```

- Device.appCode: código de la app a la que queremos hacer llegar el PDF. Por ejemplo, com.viafirma.documents.
- Device.code: código del dispositivo; este código es elegido por el usuario, o bien auto-asignado por el sistema, dependiendo de la configuración del servicio, por ejemplo "customer_care_ipad01".
- Device.type: tipo de dispositivos; soportados "IOS", "ANDROID" y "WINDOWS".
- Device.userCode: username del usuario de la app, por ejemplo "jhon.doe@viafirma.com".

Resumen de la estructura del mensaje

En resumen, el mensaje que tendremos que intercambiar con viafirma tendrá la siguiente estructura:

Ejemplo de uso para firmas presenciales:

```
{
  "notification" : {
    "text" : "Nuevo contrato pendiente de firma",
    "detail" : "Cliente con ref. A1882",
    "notificationType" : "PUSH_IOS",
    {
     "devices" : [ {
       "appCode" : "com.viafirma.documents",
       "code" : "customer_care_ipad01",
       "type" : "IOS",
       "userCode" : "jhon.doe@viafirma.com"
```

```
      } ]
    }
  },
  "document" : {
    templateReference:"https://www.viafirma.com/myPDF_sample.pdf",
    templateType:"url",
    policyCode:"myPolicyCode_sample_001"
  },
  "workflow" : {
    "code" : "EX001",
    "type" : "APP"
  }
}
```

Enviar PDF para firma a distancia

El PDF se mostrará embebido en un visor web disponible en una página web disponible en un link seguro y único. El link de acceso a esta página web podrá notificarse automáticamente al usuario final mediante tres mecanismos:

- SMS
- Email
- o bien, obtener el link de firma para montar la página de forma embebida en un sistema externo, por ejemplo, una web-wiew o iframe.

Para hacer uso de cualquiera de los mecanismos previstos para la FIRMA a DISTANCIA, tendremos que indicar el tipo WEB en el flujo de trabajo, o en su defecto para versiones anteriores a 3.6 el código EX06.

```
{
  "workflow" : {
    "code" : "EX006",
    "type" : "WEB"
  }
}
```

- workflow.code: en versiones anteriores a 3.6 se usa para definir el tipo de flujo de trabajo, y en los mecanismos previstos para firmas a distancia, se debe usar el código EX006. Esto habilitará la generación automática del link de firma con el que permitiremos al usuario final participar en el proceso de firma.
- workflow.type: en versiones 3.6 sustituye al atributo CODE, y en los mecanismos previstos para firmas a distancia, se debe usar el tipo WEB. Ambos atributos se podrán usar conjuntamente para mantener compatibilidad hacia atrás en caso de usar modelos diseñados con versiones anteriores.

Una vez definido el tipo de flujo de trabajo que vamos a habilitar para la firma, se debe definir el DOCUMENTO que deseamos firmar. Para ello usaremos el ojbeto DOCUMENT, tal y como se detalla en capítulo "Cómo enviar un PDF".

Para este ejemplo usaremos una configuración básica de documento:

```
{
  "document": {
    templateReference:"https://www.viafirma.com/myPDF_sample.pdf",
    templateType:"url",
    policyCode:"myPolicyCode_sample_001"
  }
}
```

Con el flujo de trabajo y documento definido, ya sólo falta definir cómo vamos a comunicarnos con el usuario final, y para ello vamos a explicar la estructura del objeto NOTIFICATION.

```
Notification {
```

```
  text (string),
  detail (string),
  notificationType (string, optional) = ['PUSH_IOS', 'PUSH_ANDROID', 'CALLBACK', 'MAIL', 'SMS'],
  sharedLink (SharedLink, optional),
  devices (Array[Device], optional)
}
```

objeto Notification:

- Notification.text: título de la notificación.
- Notification.detail: descripción de la notificación.
- Notification.notificationType: tipo de notificación elegida: 'PUSH_IOS', 'PUSH_ANDROID', 'CALLBACK', 'MAIL', 'SMS'
- Notification.sharedLink: sólo en caso de notificaciones tipo mail, SML o generación de links de firma.
- Notification.devices: sólo en caso de notificaciones tipo iOS y Android.

objeto Device:

Para las notificaciones en procedimientos de firmas presenciales, donde usaremos las apps de viafirma documents, usaremos el objeto Device:

```
Device {
  appCode (string),
  code (string),
  userCode (string)
}
```

- Device.appCode: código de la app a la que queremos hacer llegar el PDF. Por ejemplo, com.viafirma.documents.
- Device.code: código del dispositivo; este código es elegido por el usuario, o bien auto-asignado por el sistema, dependiendo de la configuración del servicio, por ejemplo "customer_care_ipad01".
- Device.userCode: username del usuario de la app, por ejemplo "jhon.doe@viafirma.com".

objeto SharedLink:

Cuando usemos notificaciones vía SMS o Email, o usemos el link de firma para embeberlo en un sistema externo, usaremos el objeto SharedLink:

```
SharedLink {
  email (string, optional),
  subject (string, optional),
  phone (string, optional)
}
```

- SharedLink.email: email del destinatario.
- SharedLink.subject: asunto del email remitido al destinatario.
- SharedLink.phone: número de teléfono móvil, incluyendo prefijo internacional, al que se le notificará vía SMS.

Resumen de la estructura del mensaje

En resumen, el mensaje que tendremos que intercambiar con viafirma tendrá la siguiente estructura:

Ejemplo de uso para firma a distancia:

```
{
  "notification" : {
    "text" : "Nuevo contrato pendiente de firma",
    "detail" : "Cliente con ref. A1882",
    "notificationType" : "MAIL",
```

```
    "sharedLink" : {
      "email" : "jhon.doe@viafirma.com",
      "subject" : "viafirma: testing with ref. number [MESSAGE_CODE]"
    }
  },
  "document" : {
    templateReference:"https://www.viafirma.com/myPDF_sample.pdf",
    templateType:"url",
    policyCode:"myPolicyCode_sample_001"
  },
  "workflow" : {
    "code" : "EX006",
    "type" : "WEB"
  }
}
```

Consulta del estado de una petición (Síncrono)

Una vez que disponemos del código de un mensaje podemos consultar el estado mismo.

Para ello se debe realizar una petición de tipo GET a la url *url-viafirma-documents*/api/v3/messages/messageCode

Por ejemplo en el entorno de sandbox sería:
`https://sandbox.viafirma.com/documents/api/v3/messages/1234567890123R123`

Debe sustituirse messageCode por el código de mensaje a obtener.

El resultado de esta petición será un json con toda la información del mensaje. Así por ejemplo se podrá comprobar el estado en el nodo workflow > current. Siendo RESPONSED el estado cuando la petición ha sido completada correctamente.

Puede descargar un ejemplo de respuesta aquí

Callback de respuesta automática

Te lo explicamos en esta guía:

https://doc.viafirma.com/documents/developer/callback.html

Información de un proceso

```
Revisión: 23-octubre-2018
```

El servicio disponible en el API para conocer el estado y detalle de un proceso es siguiente:

```
GET api/v3/message/{messageCode}
```

Obteniendo una respuesta en formato Application/JSON como la siguiente:

```
{
  "code": "string",
  "userCode": "string",
  "groupCode": "string",
  "appCode": "string",
  "workflow": {
    "current": "string",
    "history": [
      {
        "start": "timestamp",
        "ends": "timestamp",
        "taskName": "string"
      }
    ],
    "initiate": "timestamp",
    "lastUpdated": "timestamp",
    "expires": "timestamp",
    "type": "string"
  },
  "notification": {
    "code": "string",
    "text": "string",
    "detail": "string",
    "sharedLink": {
      "scheme": "string",
      "token": "string",
      "link": "string",
      "appCode": "string",
      "subject": "string"
    }
  },
  "document": {
    "templateCode": "string",
    "templateVersion": "number",
    "draftedCode": "string",
    "signedCode": "string",
    "signedID": "string",
    "templateType": "string",
    "formRequired": "boolean",
    "items": [
      {
        "key": "string",
        "value": "string"
      }
```

```
      ],
      "numPages": 1
    },
    "policies": [
      {
        "code": "string",
        "evidences": [
          {
            "type": "string",
            "code": "string",
            "status": "string",
            "helpText": "string",
            "positions": [
              {
                "rectangle": {
                  "x": "number",
                  "y": "number",
                  "width": "number",
                  "height": "number"
                },
                "page": "number"
              }
            ],
            "metadataList": [
              {
                "key": "string",
                "value": "string"
              }
            ],
            "typeFormatSign": "string",
            "geolocation": {
              "accuracy": "number",
              "latitude": "number",
              "longitude": "number"
            }
          }
        ],
        "signatures": [
          {
            "type": "string",
            "code": "string",
            "status": "string",
            "helpText": "string",
            "certificateAlias": "string",
            "typeFormatSign": "string",
            "idSign": "string",
            "stampers": [
              {
                "type": "string",
                "rotation": "string",
                "width": "number",
                "height": "number",
                "xAxis": "number",
                "yAxis": "number",
                "page": "number"
              }
            ]
          }
        ]
      }
    ],
    "server": "string",
    "auditory": [
      {
        "date": "timestamp",
        "ip": "string",
        "action": "string",
        "detail": ""
```

```
    }
  ]
}
```

A continuación la explicación de los distintos atributos y objetos contenidos en la respuesta:

Message

```
{
  "code": "string",
  "userCode": "string",
  "groupCode": "string",
  "appCode": "string"
}
```

donde:

Param	Desc
code	código del proceso con el que podrás identificar al proceso para el resto de servicios disponibles en el API.
userCode	código del usuario propietario del proceso
groupCode	código del grupo desde el que el propietario generó el proceso; el grupo podrá determinar comportamiento específico de negocio.
appCode	aplicación desde la que se generó el proceso; cuando el proceso se generó desde el backend de viafirma, el código será el asociado la aplicación `Documents Backend`, normalmente identificada por defecto con el código `com.viafirma.mobile.services`.

Workflow

Objeto con la información sobre el estado y detalle de las distintas tareas asociadas al ciclo de vida del proceso. Este ciclo de vida variará de un proceso a otro en función de múltiples factores: tipo de proceso de firma (Firma remota, firma presencial), tipo de transferencias, tipo de validaciones, etc.

```
{
"workflow": {
    "current": "string",
    "history": [
      {
        "start": "timestamp",
        "ends": "timestamp",
        "taskName": "string"
      }
    ],
    "initiate": "timestamp",
    "lastUpdated": "timestamp",
    "expires": "timestamp",
    "type": "string"
  }
}
```

donde:

Param	Desc
current	estado actual del proceso; ver `lista de estados disponibles`
history	lista de estados por los que el proceso ha pasado, incluyendo para cada uno la fecha y hora expresada en `milisegundos` para el inicio y fin y el nombre de la tarea.
initiate	fecha y hora en la que se inició el proceso, expreado en `milisegundos`
lastUpdated	fecha y hora de la última actualización del proceso, expreado en `milisegundos`
expires	fecha y hora en la que el proceso caducará, expresado en `milisegundos`. La caducidad del proceso se configura por defecto en la propiedades de la aplicación propietaria del proceso (ver fecha y hora en la que se inició el proceso, expreado en `appCode`), o bien la fecha de caducidad explícita indicada por el integrador a la hora de consumir el servicio mediante API
type	tipo de proceso; ver `lista de procesos disponibles`

Lista de estados disponibles:

- DELETED
- EXPIRED
- ERROR
- REJECTED
- TRANSFERRED
- RESPONSED
- SERVER_SIGN
- SENT
- SIGNED
- WAITING
- COMMITTED
- RECEIVED
- TEMPORAL_STORED
- STAND_BY
- WAITING_CHECK
- APPROVED
- FINISHED
- WAITING_OCR
- WAITING_CLIENT_SIGNATURE
- MAX_ERROR_REACHED

Lista de procesos disponibles:

WORKFLOW TYPE	DESC
APP	Proceso donde el documento es enviado a la app de viafirma en un dispositivo móvil, notificando al destinatario mediante una notificación push.
WEB	Proceso donde el documento es procesado desde una página web autogenerada por viafirma y notificada al destintario de la notificación vía EMAIL y/o SMS.

PRESENTIAL	Mismo procedimiento utilizado para WEB pero no hay notificación a ningún destinatario, sino que el link autogenerado por viafirma es consumido vía API por otra aplicación, por ejemplo para embeber la página de firma en una web-view de otra app o de una web-app. Este tipo de proceso es el indicado para el uso de firma biométrica basada en el uso de pads de firma WACOM. También es el proceso utilizado para procesos de firma con certificado digital a través de la app viafirma documents desktop.

Notification


```
{
"notification": {
    "code": "string",
    "text": "string",
    "detail": "string",
    "sharedLink": {
      "scheme": "string",
      "token": "string",
      "link": "string",
      "appCode": "string",
      "subject": "string"
    }
  }
}
```

Descarga de un documento firmado

```
Revisión: 23-octubre-2018
```

El servicio disponible en el API para descargar el documento firmado es el siguiente:

```
GET api/v3/documents/download/signed/{messageCode}
```

Obteniendo una respuesta en formato Application/JSON como la siguiente:

```
{
  "link": "string",
  "md5": "string",
  "fileName": "string",
  "expires": "string"
}
```

donde:

Param	Desc
link	Link autogenerado para la descarga del documento firmado; este link tiene una validez de 10 minutos; superado ese tiempo es necesario volver a consumir el servicio.
md5	hash del contenido del documento firmado para facilitar procesos de verificación y autenticidad.
fileName	nombre del documento firmado; por defecto el nombre de los documentos firmados será igual el messageCode del proceso más la extensión .pdf.
expires	fecha y hora en formato milisegundos que indica el vencimiento del LINK de descarga. Por defecto son 10 minutos.

¿Cuándo debo consumir el servicio?

Existen múltiples estados en función del workflow implementado, y por ello no todos los estados estarán asociados a un documento ya firmado. Un documento firmado podrá descargarse si está en alguno de los siguientes estados:

Status	Desc
FINISHED	proceso finalizado correctamente, donde el usuario NO ha rechazado la solicitud y el documento ya ha sido firmado, pudiendo incluir evidencias adicionales según la política de firma utilizada.
RESPONSED	Tras finalizar el proceso se ha hecho algún tipo de callback, por ejemplo callbackURL o callbackMail. Este estado también permite consumir el servicio para obtener el documento firmado.

TRANSFERRED	Además del callback realizado, el proceso o el grupo al que pertenece están asociados a una transferencia automática del documento firmado a un repositorio externo.
MAX_ERROR_REACHED	La trasnferencia al repositorio externo ha fallado. Durante ese período de tiempo el documento firmado sigue estando en viafirma, por lo que en este estado también es posible consumir el servicio para descargar el documento firmado.

Recuerda para que para conocer el estado de un proceso podrás usar el siguiente servicio disponible en el API:

```
GET api/v3/messages/{messageCode}
```

y que te explicamos en esta otra guía: detalle de un proceso

Posibles estados de una petición

Estado	¿Estado final?	Explicación
FINISHED	SÍ	Petición firmada y transeferida al sistema del cliente. Se ejecuta callback si se ha configurado
MAX_ERROR_REACHED	SÍ	Petición firmada y no transferida al cliente debido a algún tipo de error descripto en el detalle de la petición
RESPONSED	SÍ	Petición firmada. Se ejecuta callback si se ha configurado
ERROR	NO	Este estado indica que ha habido un error durante la vida de la petición. Comprobar error en el detalle de la petición
EXPIRED	SÍ	Petición caducada. Se ejecuta callback si se ha configurado
RECEIVED	NO	Petición recibida por el backend y a la espera de su procesado
REJECTED	SÍ	Petición rechazada por cliente. Se ejecuta callback si se ha configurado
SENT	NO	Petición enviada al backend para ser firmada una vez cumplimentadas todas las evidencias por el usuario
SIGNED	NO	Estado en el que se realiza la firma en servidor del documento una vez añadidas todas las evidencias
STAND_BY	NO	Petición en estado borrador pendiente de ser enviada por el usuario
WAITING	NO	Petición con documento generado a la espera de ser cumplimentado
WAITING_CHECK	NO	Petición a la espera de aprobación manual
WAITING_CLIENT_SIGNATURE	NO	Petición a la espera de una firma cliente
WAITING_OCR	NO	Petición a la espera del tratamiento OCR de una imagen

Uso de Políticas en viafirma documents

```
Últ. revisión: 06 ago 2018
```

¿Qué es una política en viafirma documents?

Las políticas son un conjunto de evidencias, firmas, aprobaciones y otro tipo de elementos de seguridad que van a determinar cómo se debe firmar el PDF que remitiremos al usuario.

Los elementos que intervienen en una política son:

Firmas electrónicas (con certificado digital)

- desatendida (en servidor)
- en cliente

Evidencias Electrónicas

- firma biométrica
- OTP/SMS
- OTP/Mail
- imágenes
- huellas
- cuestionarios de seguridad
- checks avanzados

Aprobaciones

- aprobación simple (check)
- aprobación con código (PIN)
- aprobación con certificado digital en cliente
- aprobación con certificado digital en servidor

Otras

- geolocalización (GPS)
- código de bloqueo (PIN)
- lectura obligada del documento

Objeto Policy

Los mensajes intercambiados con viafirma documents podrán tener una o varias políticas que serán aplicadas al documento que se desea firmar. Este objeto cuenta con las siguientes características y estructura básica.

```
Policy {
  evidences (Array[Evidence], optional),
  signatures (Array[Signature], optional)
}
```

- evidences: conjunto de evidencias electrónicas: imágenes, huellas, firmas biométricas, OTP/SMS, OTP/Mail o Checks avanzados.
- signatures: conjunto de firmas electrónicas (con certificado digital) que se realizarán sobre el PDF tras haber capturado las evidencias definidas.

Evidencias

Este objeto cuenta con las siguientes características y estructura básica.

```
Evidence {
  type (string) = ['SIGNATURE', 'FINGERPRINT', 'IMAGE', 'ANNOTATION', 'FINGER_PRINT', 'OTP_SMS', 'GENERIC'],
  helpText (string, optional),
  helpDetail (string, optional),
  positions (Array[Position], optional),
  typeFormatSign (string, optional),
  optional (boolean, optional),
  positionsKey (string, optional)
}
```

Este objeto permite mayor número de atributos y configuración que puedes consultar en detalle en la documentación avanzada de políticas.

Ejemplo de evidencia

```
Evidence {
  type: "SIGNATURE",
  helpText: "Firma biométrica del cliente",
  helpDetail: "Nombre y apellidos del cliente",
  typeFormatSign: "XADES_T",
  optional: "false",
  positionsKey: "signature_box"
}
```

Signatures

Este objeto cuenta con las siguientes características y estructura básica.

```
Signature {
  type (string) = ['CLIENT', 'SERVER'],
  helpText (string, optional),
  typeFormatSign (string, optional) = ['PADES_B', 'PADES_T', 'PADES_LT', 'PADES_LTA']
}
```

Este objeto permite mayor número de atributos y configuración que puedes consultar en detalle en la documentación avanzada de políticas.

Ejemplo de firma

```
Signature {
  type: "SERVER",
  helpText: "Sello electrónico",
  typeFormatSign: "PADES_LTA"
}
```

Cómo usar las políticas en viafirma documents

```
Últ. revisión: 06 ago 2018
```

Como integrador tienes 2 alternativas para usar políticas:

- uso implícito de políticas
- uso explícito de políticas

Uso implícito de políticas

Como integrador podrás facilitar tu trabajo apuntando a un código de políticas que previamente ha sido diseñado y gestionado en el backend de viafirma documents. De esta forma, sólo tendrás que definir qué documento enviar y a quién.

```
"document" : {
    templateType:"url",
    templateReference:"https://www.viafirma.com/myPDF_sample.pdf",
    policyCode:"myPolicyCode_sample_001"
  }
```

Ejemplo de uso implícito de políticas

```
{
  "notification" : {
    "text" : "Nuevo contrato pendiente de firma",
    "detail" : "Cliente con ref. A1882",
    "notificationType" : "PUSH_IOS",
    {
     "devices" : [ {
       "appCode" : "com.viafirma.documents",
       "code" : "customer_care_ipad01",
       "type" : "IOS",
       "userCode" : "jhon.doe@viafirma.com"
      } ]
    }
  },
  "document" : {
    templateReference:"https://www.viafirma.com/myPDF_sample.pdf",
    templateType:"url",
    policyCode:"myPolicyCode_sample_001"
  },
  "workflow" : {
    "type" : "APP"
  }
}
```

En el ejemplo anterior, el documento está referenciado en una URL, y se aplicará la política de firma con código "myPolicyCode_sample_001", política que previamente ha debido ser creada y diseñada en el backend de viafirma documents.

Uso explícito de políticas

Si no puedes referenciar a una política existente tendrás que definirla explícitamente en el servicio. Para ello tendrás que gestionar el objeto POLICY.

Ejemplo de uso implícito de políticas

Política de ejemplo que informaremos de forma explícita en nuestro mensaje:

```
"policies" : [ {
      "evidences" : [ {
        "type" : "SIGNATURE",
        "helpText" : "Firma de {{cliente}}",
        "typeFormatSign" : "XADES_B",
        "positionsKey" : "firma_place"
      } ],
      "signatures" : [ {
        "type" : "SERVER",
        "helpText" : "",
        "typeFormatSign" : "PADES_B",
        "stampers" : [ {
          "type" : "QR_BARCODE128",
          "rotation" : "ROTATE_90",
          "positionsKey" : "stamper_place"
        } ]
      } ]
    } ]
```

El mensaje con la política usada de forma explícita quedaría de la siguiente forma:

```
{
  "notification" : {
    "text" : "Nuevo contrato pendiente de firma",
    "detail" : "Cliente con ref. A1882",
    "notificationType" : "PUSH_IOS"
    {
     "devices" : [ {
       "appCode" : "com.viafirma.documents",
       "code" : "customer_care_ipad01",
       "type" : "IOS",
       "userCode" : "jhon.doe@viafirma.com"
      } ]
    }
  },
  "document" : {
    "templateReference" : "https://www.viafirma.com/myPDF_sample.pdf",
    "templateType" : "url"
  },
  "policies" : [ {
      "evidences" : [ {
        "type" : "SIGNATURE",
```

```
        "helpText" : "Firma del cliente",
        "typeFormatSign" : "XADES_B",
        "positionsKey" : "signature_box"
      } ],
      "signatures" : [ {
        "type" : "SERVER",
        "helpText" : "",
        "typeFormatSign" : "PADES_B",
        "stampers" : [ {
          "type" : "QR_BARCODE128",
          "rotation" : "ROTATE_90",
          "positionsKey" : "stamper_box"
        } ]
      } ]
    } ],
  "workflow" : {
    "type" : "APP"
  }
}
```

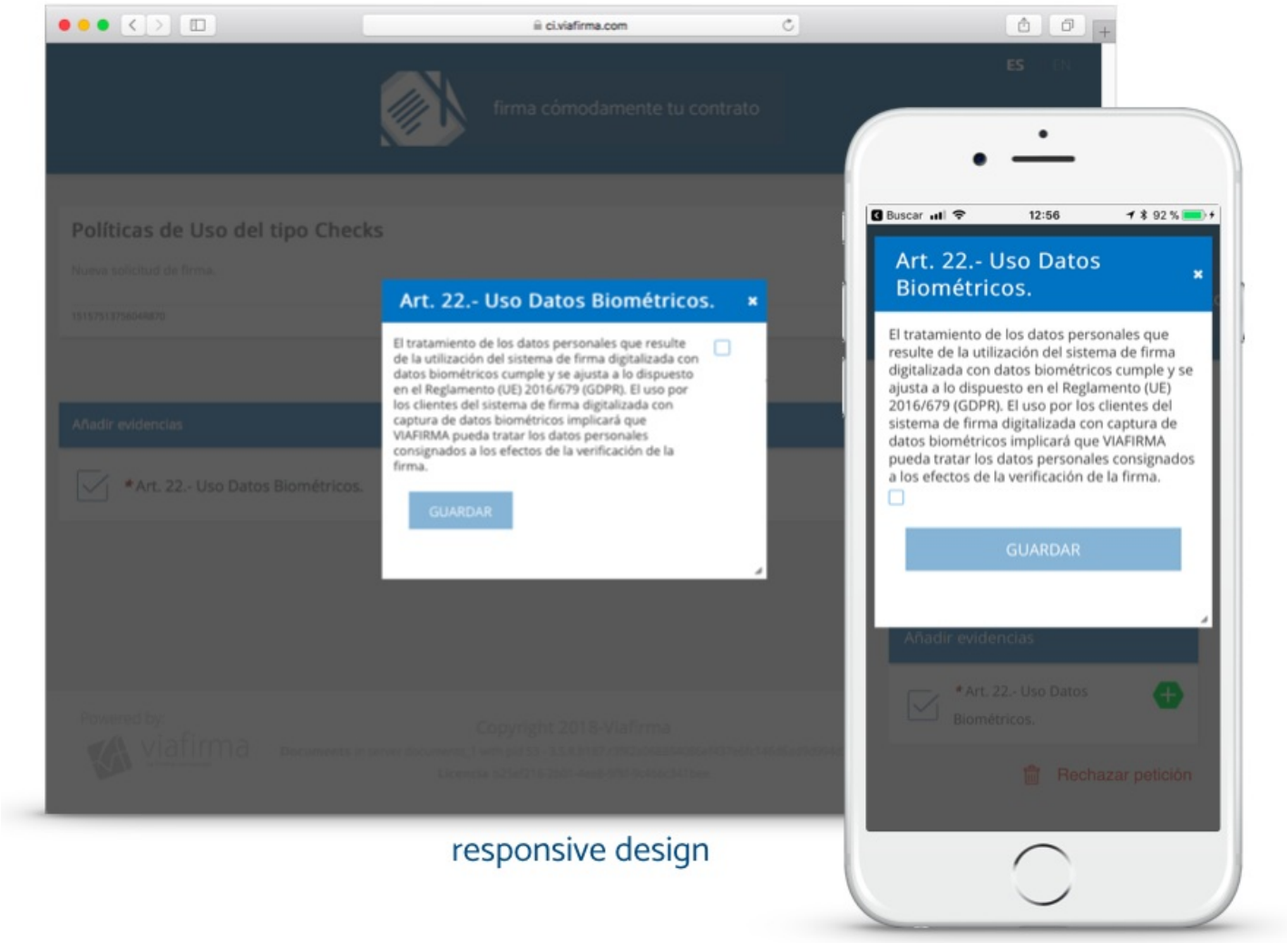
Políticas basadas en Checks Avanzados

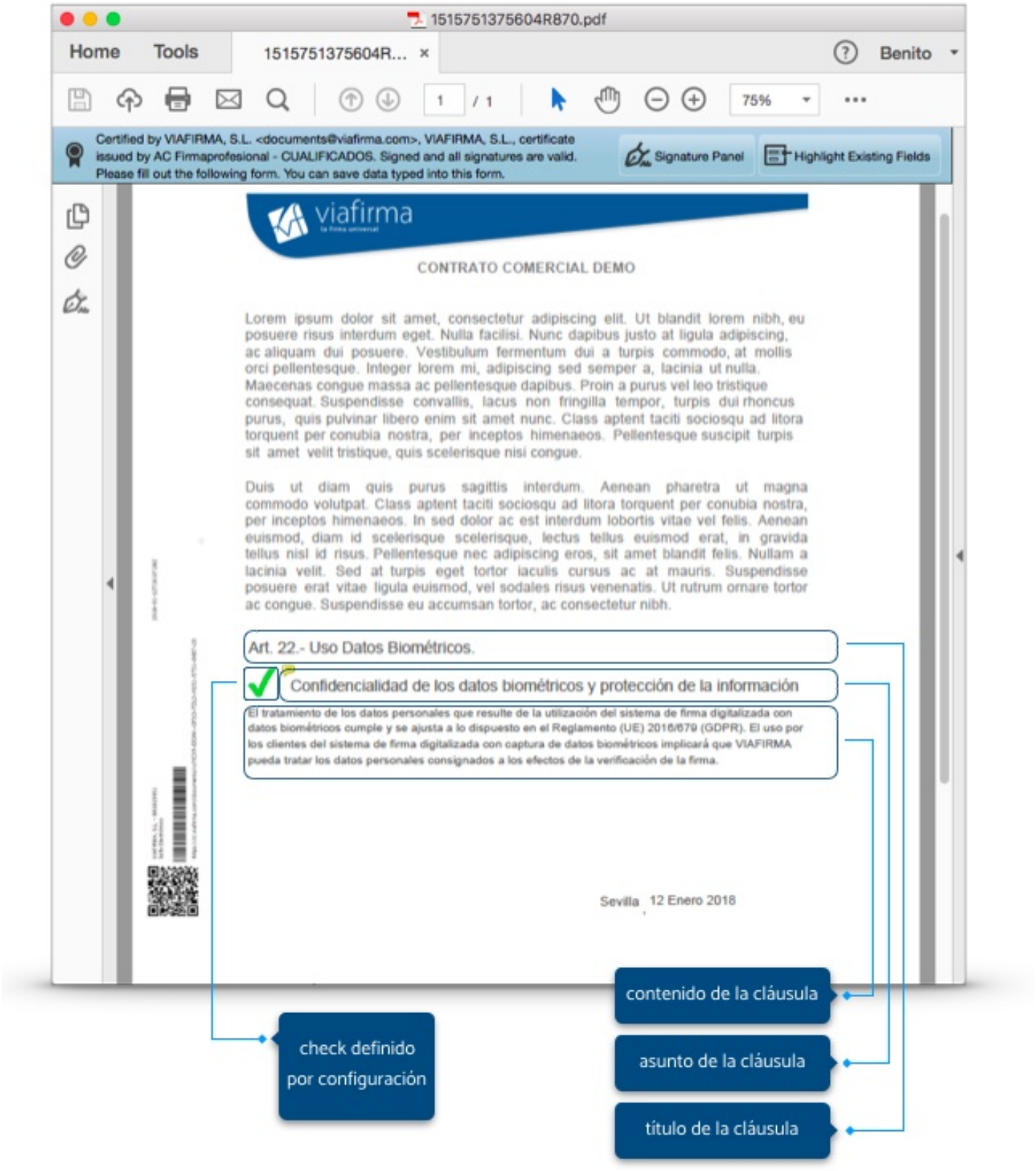
```
Últ. revisión: 06 ago 2018
```

Viafirma documents permite agregar políticas basadas en el uso de evidencias electrónicas del tipo Check Avanzado. Resultarán útiles para capturar aceptaciones voluntarias del usuario destacando clásulas o condiciones críticas o de especial importancia, por ejemplo, aquellas cláusulas recomendadas por GDPR, MIFID II, etc.

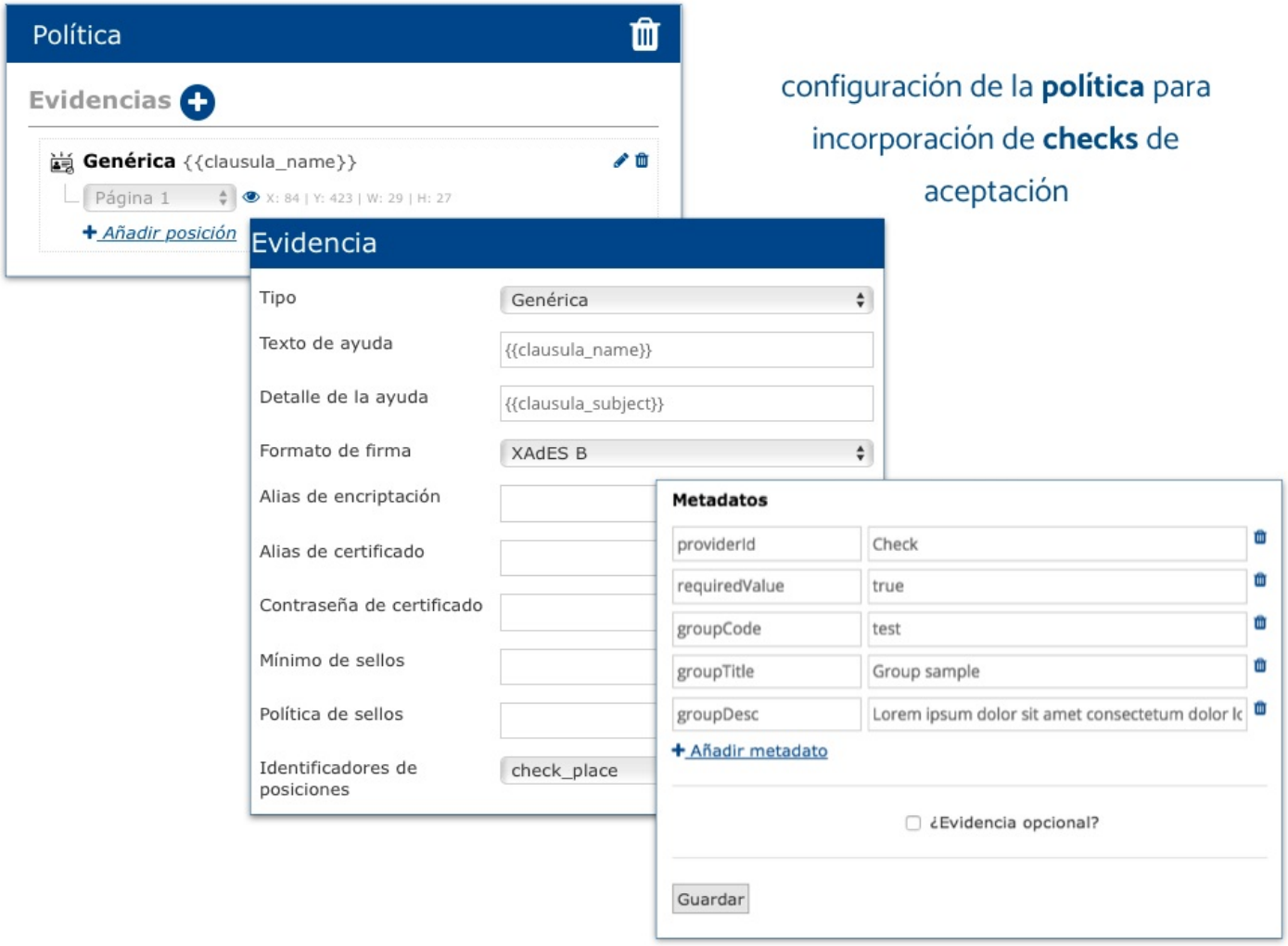
Podrás agregarlas directamente desde el diseñador visual de politicas para referenciarlas a partir de un identificador único de política, o bien podrás definirla al vuelo en tu servicio consumido vía API.

Uso de Checks Avanzados





Configuración



Consumo vía API

Entre la lista de evidencias disponibles en las políticas debes agregar una evidencia del tipo GENERIC y formatearla a las necesidades de cada caso.

Atributo	Descripción
evidences.type	hasta la fecha están soportadas tres tipos de evidencias: "IMAGE","SIGNATURE","FINGERPRINT y "GENERIC". Para las evidencias del tipo CHECK que nos ocupa en esta documentación usaremos el tipo "GENERIC"
evidences.helpText	título de la cláusula o propiedad que queremos definir; se usará en las cabeceras de las ventanas emergentes
evidences.helpDetail	subtítulo de la cláusula o propiedad que queremos definir, que podremos usar a modo de "asunto"; se usará en la segunda línea de las cabeceras de las ventanas emergentes
evidences.metadalist.providerId	Check
evidences.metadalist.requiredValue	Opcional; sólo lo usaremos si queremos forzar un valor en concreto: true o false. En caso de no usar este atributo el usuario podrá marcar o no marcar el check, a su criterio.
evidences.metadalist.groupCode	Opcional; se usa para agrupar un conjunto de Checks avanzados dentro de la misma política, mostrándose todos ellos en un mismo pop-

	up facilitando la marcación múltiple por parte del usuario.
evidences.metadalist.groupTitle	Opcional; en caso de usar agrupaciones de ckecks avanzados, y se usa para el título del grupo.
evidences.positions	Opcional si se usa "positionsKey"; lista de posiciones en la que imprimiremos la marca "checked" sobre el contenido del PDF. Podemos evitar el uso de posiciones si el PDF cuenta con marcas del tipo Acrofields, en cuyo caso usaremos el atributo positionsKey explicado más abajo. Cada posición debe informar lo siguiente:
evidences.positions.rectangle	define el área de la marca a incrustar, indicando posición (x,y) y su tamaño (widht,height)
evidences.positions.page	podrás definir hasta tres valores: "0" para imprimirla en todas las páginas, "1" para imprirmirla sólo en la primera página, "-1" para imprimirla sólo en la última y "-2" para imprimir la marca en una página en blanco insertada al final del documento original
evidences.typeFormatSign	la evidencia se construye y envuelve en un formato XML, el cual es firmado con certificado digital. Con typeFormatSign definimos el formato de la firma, pudiendo usar los distintos valores: "XADES_B" y "XADES_LTA", este último consume sello de tiempo, el cual debería estar previamente configurado en las propiedades generales del servicio
evidences.positionsKey	Opcional si se usan posiciones manuales (ver atributo "positions"); podemos evitar el uso de posiciones si el PDF cuenta con marcas del tipo Acrofields, en cuyo caso usaremos este atributo para indicar el nombre del acroField que usaremos para imprimir la marca "checked"
evidences.base64Image	Es opcional, y se usará para indicar la imagen que usaremos como marca "checked", en formato base64. Si no informamos este atributo, la imagen utilizada para la marca checked será la asignada en la configuración del estilo asignado al grupo propietario de la petición.

Ejemplo de EVIDENCIA PARA CHECKS AVANZADOS

```
{
  "type" : "GENERIC",
  "helpText" : "Art. 22.- Uso Datos Biométricos.",
  "helpDetail" : "Confidencialidad de los datos biométricos y protección de la información conforme al GDPR
  "metadataList" : [ {
    "key" : "providerId",
    "value" : "Check"
  }, {
    "key" : "text",
    "value" : "El tratamiento de los datos personales que resulte de la utilización del sistema de firma dig
  }, {
    "key" : "requiredValue",
    "value" : "true"
  } ],
  "typeFormatSign" : "XADES_B",
  "positionsKey" : "check_place"
}
```

nota: el positionsKey "ckeck_place" hace referencia a un Acrofield insertado en el PDF utilizado; en caso de no utilizar acrofields en tus plantillas, la posición del check avanzado se fijará de forma absoluta, sustituyendo el atributo "positionsKey" por el objeto "positions", tal y como se muestra en el siguiente ejemplo:

```
"positions" : [ {
  "rectangle" : {
    "x" : 68,
    "y" : 415,
    "width" : 51,
    "height" : 38
  },
  "page" : 1
} ]
```

Monta tu propia plantilla

Aquí te dejamos los recursos para que puedas crear tu nueva plantilla basada en esta evidencia y puedas probarla.

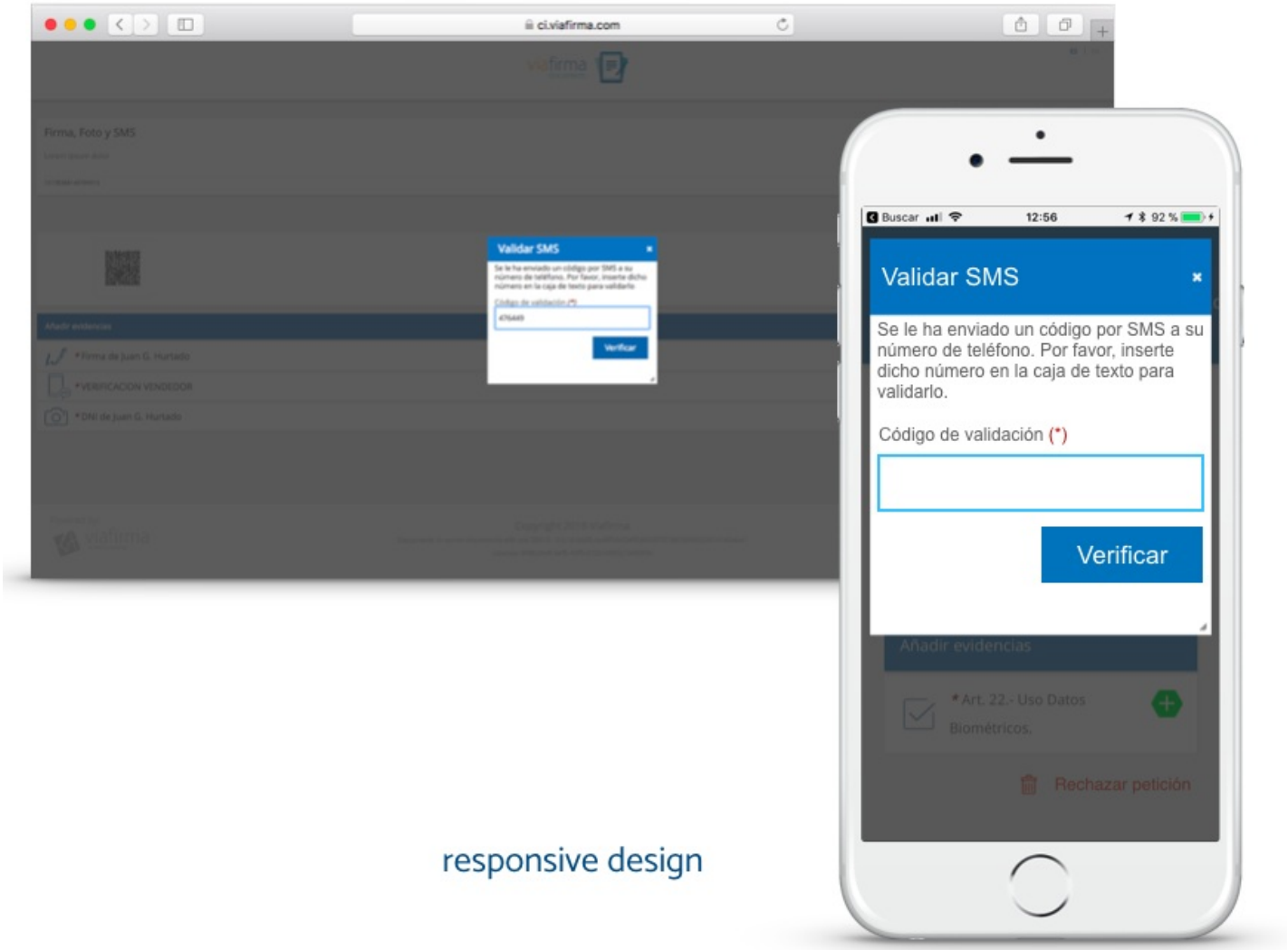
- Plantilla 364_example.pdf
- Configuración 364_example.json

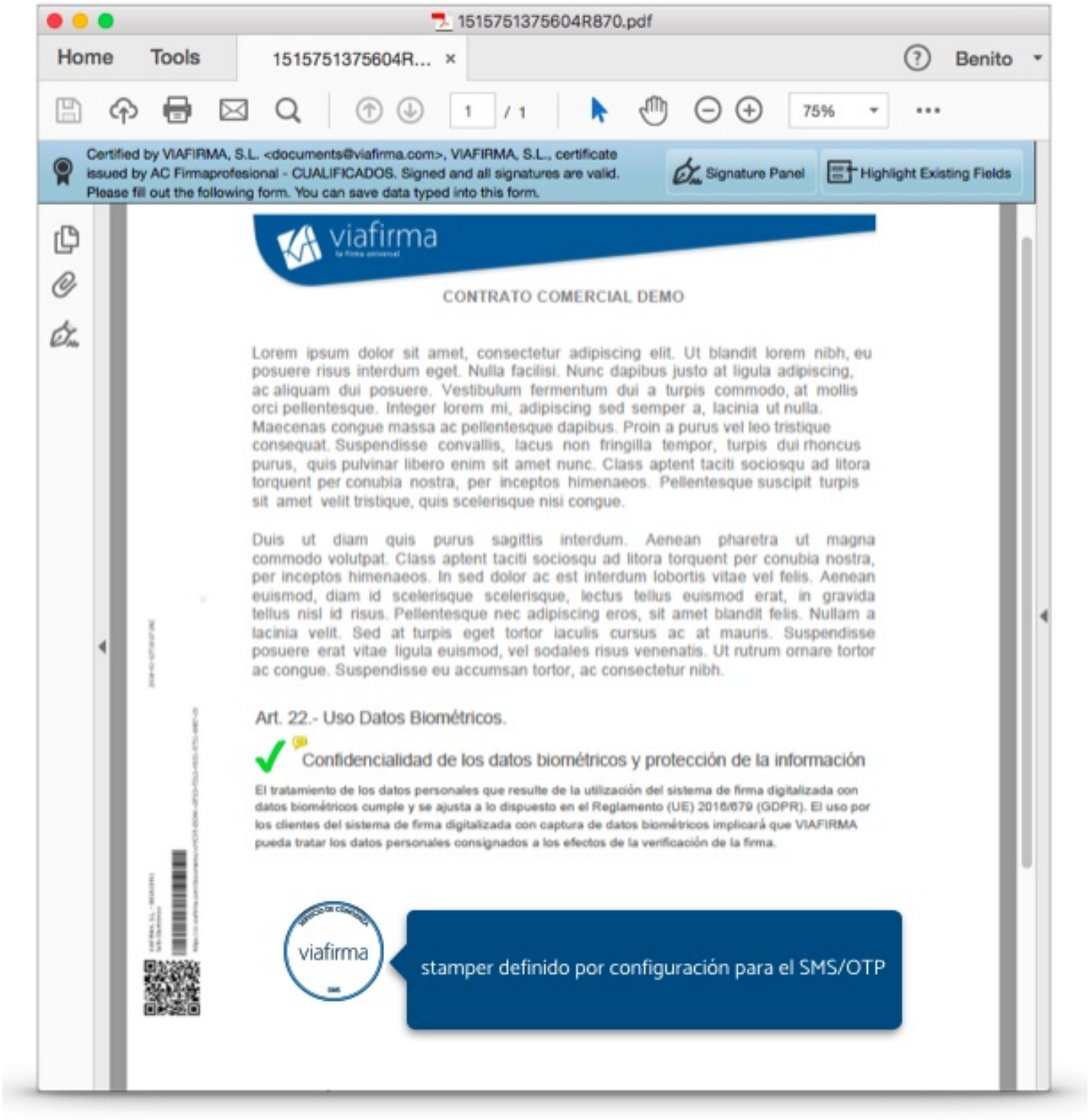
Políticas basadas en OTP/SMS

Viafirma documents permite agregar políticas basadas en el uso de evidencias electrónicas del tipo OTP por SMS. Permiten añadir una evidencia basándose en el envío de una contraseña de tipo OTP (One time password) por SMS al número de teléfono del destinatario, garantizando que sólo él puede ver esa contraseña e introducirla a la hora de firmar el documento de la petición.

Podrás agregarlas directamente desde el diseñador visual de politicas para referenciarlas a partir de un identificador único de política, o bien podrás definirla al vuelo en tu servicio consumido vía API.

Uso de OTP/SMS





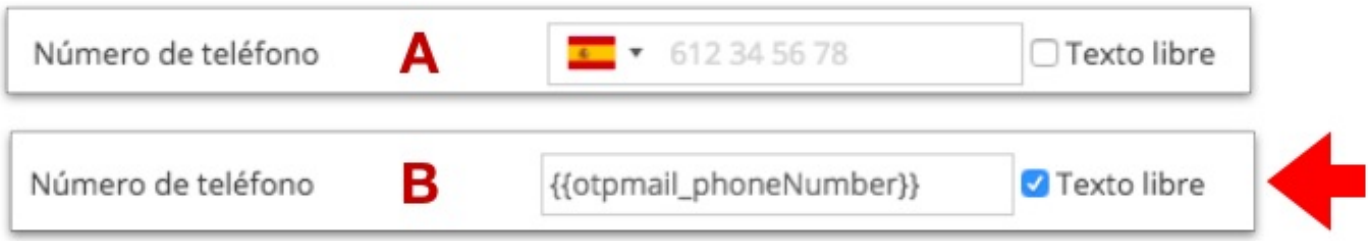
Configuración

Podrás definir las propiedades básicas para el uso de este tipo de evidencia.

También podrás incorporar en la configuración de tu política valores dinámicos, inyectándolos de distintas formas, por ejemplo:

- solicitándolo en un formulario
- pasándolo como metadato mediante API

Para hacer uso de un valor dinámico en el número de teléfono solo tendrás que activar la casilla 'Texto Libre'. De esta forma desactivaremos la validación del formato de número de teléfono, y podrás incluir el nombre de la variable que desees, haciendo uso de los limitadores doble llave, tal y como te mostramos a continuación:



Para el resto de configuración podrás hacer uso de variables directamente, por ejemplo:

```
Detalle de la ayuda: Introduce aquí el código que hemos enviado al número {{otpmail_phoneNumber}}
Texto del SMS: Usa este valor para validar y firmar tu nuevo contrato {{contract_name}}
```

Consumo vía API

Entre la lista de evidencias disponibles en las políticas debes agregar una evidencia del tipo OTP_SMS y formatearla a las necesidades de cada caso.

Atributo	Descripción

evidences.type	esta evidencia es de tipo "OTP_SMS"
evidences.helpText	título de la cláusula o propiedad que queremos definir; se usará en las cabeceras de las ventanas emergentes
evidences.helpDetail	subtítulo de la cláusula o propiedad que queremos definir, que podremos usar a modo de "asunto"; se usará en la segunda línea de las cabeceras de las ventanas emergentes
evidences.positions	lista de posiciones en la que imprimiremos el sello de verificación por OTP/SMS sobre el contenido del PDF. Podemos evitar el uso de posiciones si el PDF cuenta con marcas del tipo Acrofields, en cuyo caso usaremos el atributo positionsKey explicado más abajo. Cada posición debe informar lo siguiente:
evidences.positions.rectangle	define el área de la marca a incrustar, indicando posición (x,y) y su tamaño (widht,height)
evidences.positions.page	podrás definir hasta tres valores: "0" para imprimirla en todas las páginas, "1" para imprirmirla sólo en la primera página, "-1" para imprimirla sólo en la última y "-2" para imprimir la marca en una página en blanco insertada al final del documento original
evidences.typeFormatSign	la evidencia se construye y envuelve en un formato XML, el cual es firmado con certificado digital. Con typeFormatSign definimos el formato de la firma, pudiendo usar los distintos valores: "XADES_B" y "XADES_LTA", este último consume sello de tiempo, el cual debería estar previamente configurado en las propiedades generales del servicio
evidences.positionsKey	Podemos evitar el uso de posiciones si el PDF cuenta con marcas del tipo Acrofields, en cuyo caso usaremos este atributo para indicar el nombre del acroField que usaremos para imprimir el sello de verificación por OTP/SMS
evidences.base64Image	Es opcional, y se usará para indicar la imagen que usaremos como sello de la evidencia, en formato base64. Si no informamos este atributo, la imagen utilizada para el sello será la asignada en la configuración del grupo propietario de la petición
metadataList	En esta lista de pares "key"/"value" se deberá introducir el número de teléfono del destinatario del SMS en un objeto cuya "key" debe ser "phoneNumber" y cuyo "value" será el número de teléfono (con prefijo internacional) (ver ejemplo).

Ejemplo de política del tipo OTP/SMS:

```
{
  "policies" : [ {
    "evidences" : [ {
      "type" : "OTP_SMS",
      "helpText" : "Código SMS",
      "helpDetail" : "Código de verificación",
      "positions" : [ {
        "rectangle" : {
          "x" : 84,
          "y" : 423,
          "width" : 29,
          "height" : 27
        },
        "page" : 1
      } ],
      "typeFormatSign" : "XADES_B",
      "positionsKey" : "sms_place",
      "base64Image" : "iVBORw0KGgoAAAANSUhEU[...]ORK5CYII=",
      "metadataList": [
        {
```

```
            "key": "phoneNumber",
            "value": "+34666777888"
          }
        ]
      } ]
    } ]
  }
```