MANUAL DE INTEGRACIÓN DEL SERVIDOR MATCHONCARD

PARA

Ministerio Secretaría General de la Presidencia

Gobierno de Chile

Interacción servidor matchoncard web

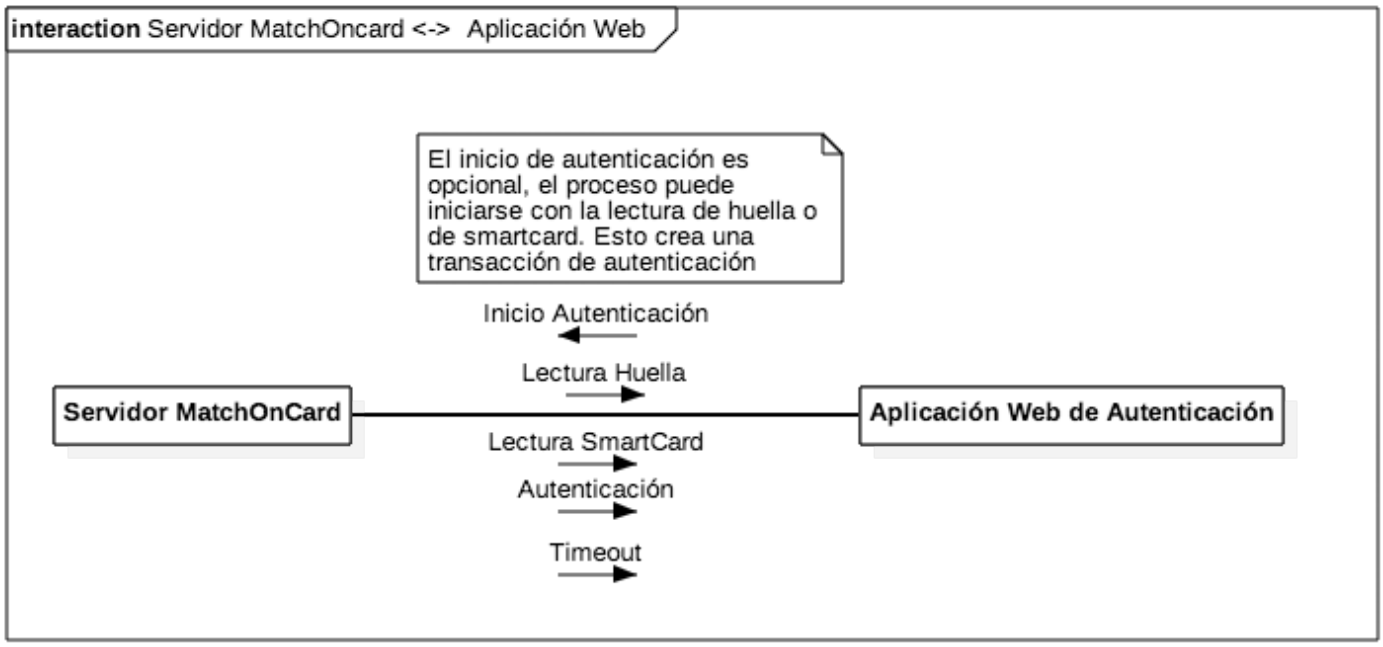
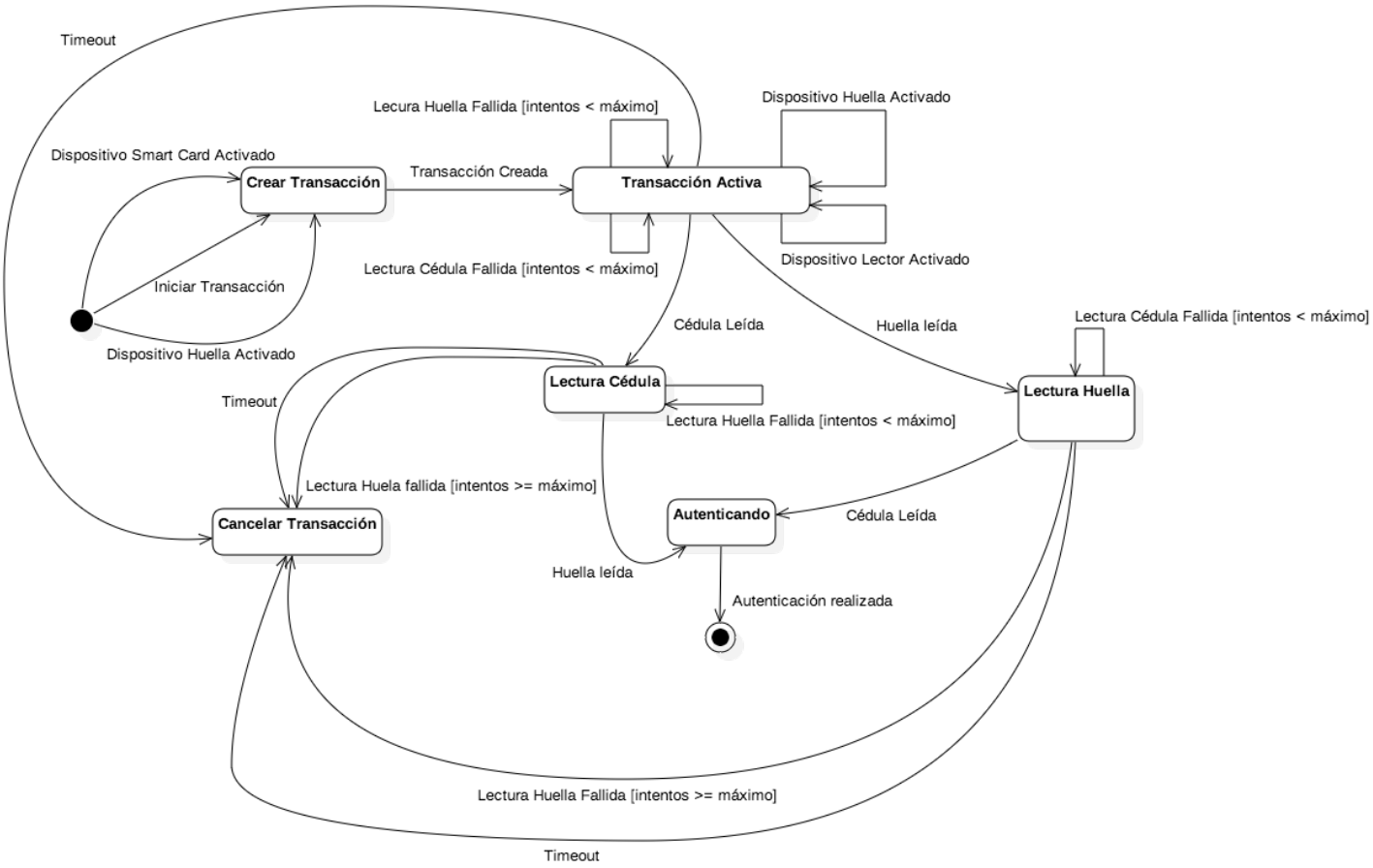


Diagrama de estado de la interacción matchoncard web



API WEBSOCKET PARA LA APLICACIÓN

Sobre Websockets

WebSocket es una tecnología que proporciona un canal de comunicación bidireccional y full-duplex sobre un único socket TCP. Está diseñada para ser implementada en navegadores y servidores web, pero puede utilizarse por cualquier aplicación cliente/servidor.
En este caso el servidor de matchoncard es el servidor y la aplicación actúa como cliente. La información que se intercambia entre el servidor y el cliente puede ser en formato texto o en formato binario. En este caso el formato utilizado es texto. En concreto son mensajes que contienen un JSON.

El servidor funciona vía websockets en la url: http://localhost:8082/api/canal

El servidor mantiene el websocket abierto mediante un heartbeat (Comando Eco). Este comando no se documenta como parte de la integración.

COMANDOS

La manera en que el client interactúa con el servidor de matchoncard es vía comandos. Los comandos que se le pueden dar al servidor son los siguientes:

- **iniciar_transaccion**:Inicia una transacción de autenticación
- **finalizar_transaccion**:Finaliza una transacción de autenticación
- **cambiar_dedo**:Cambia el dedo seleccionado la autenticación por huella dactilar

La estructura JSON utilizada para enviar un comando al servidor de matchoncard es la siguiente:

```
{
    token:"fvdzWLeXqST8JaOLxjERrA==",
    comando:"iniciar_trasanccion"
}
```

El token corresponde a un texto codificado en base 64 y encriptado con AES. El comando es uno de los tres comandos posibles, mencionados anteriormente.

iniciar_transaccion

```
{
  "token": "fvdzWLeXqST8JaOLxjERrA==",
  "comando": "iniciar_transaccion"
}
```

finalizar_transaccion

```
{
  "token": "fvdzWLeXqST8JaOLxjERrA==",
  "comando": "finalizar_transaccion"
}
```

cambiar_dedo

Si falla la lectura de huella, una de las posibilidades de falla, es que la huella del usuario presente algún problema y por ello es factible solicitar un cambio de dedo, al autenticar.
En el caso del comando "cambiar_dedo", se agrega un dato opcional que es el dedo a utilizar.

```
{
    token:"fvdzWLeXqST8JaOLxjERrA==",
    comando:"cambiar_dedo",
    dedo:1
}
```

La numeración de los dedos está dada por un byte (8 bits), con valores que se asignan a los 5 bits menos significativos: La numeración de los dedos es la siguiente:

Bit8	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Subtipo Biométrico
0	0	0	0	0	0	0	0	Sin Información
						0	1	Mano Derecha
						1	0	Mano Izquierda
			0	0	0			Sin Significado
			0	0	1			Dedo Pulgar
			0	1	0			Dedo índice
			0	1	1			Dedo Medio
			1	0	0			Dedo Anular
			0	1	1			Dedo Menor o Meñique

Ejemplos, el dedo índice de la mano derecha corresponde : al valor entero 9 (Hexadecimal 09), el dedo índice la mano izquierda al 10 (Hexadecimal 0A).

EVENTOS

El servidor despacha al cliente evento. Un evento puede ser despachado como respuesta a un comando o de manera automática por algo que ocurrió en el servidor.

Los tipos de eventos son los siguientes:

- **transaccion_iniciada**: Transacción de autenticación iniciada
- **transaccion_finalizada**: Transacción de autenticación finalizada
- **timeout**: Se produce un timeout al procesar la transacción.
- **huella_leida**: Se ha intentado leer una huella. Si la lectura falló estó lleva un asociado un error. Si por el

contrario la lectura fue exitosa, en la data se incorpora el dedo y la minucia.
- **cedula_leida**: Se ha intentado leer la cédula. Si la lectura falló estó lleva un asociado un error. Si por el contrario la lectura fue exitosa, en la data se incorpora el rut y el nombre que figura en la cédula.
- **dedo_cambiado**: Se ha cambiado el dedo asociado a la lectura.
- **huella_activada**: Se ha activado el dispositivo de lectura de huella.
- **cedula_activada**: Se ha activado el dispositivo de lectura de cédula
- **reintentos_superado**: El número de reintentos se ha superado. El error indica el código y en el mensaje se detalla los reintentos.
- **autenticacion**: Resultado de la autenticación, si falló el proceso de matchoncard el evento contiene un error. Si se logró invocar el proceso de autenticación matchoncard, el resultado contiene si la persona fue autenticada (true) o false en caso contrario. También incluye la minucia del dedo utilizada para la autenticación.

La estructura JSON utilizada por el servidor de matchoncard para enviar un evento al cliente es la siguiente:

```
{
    id : 1234,
    tipo:"transaccion_iniciada",
    fecha:1232323,
    idTransaccion:123233,
    idTotem :12313,
        ....Datos dependiendo del tipo de evento....

    codigoError : 1001,
    mensajeError :"Glosa del error",
    firma : "fvdzWLeXqST8JaOLxjERrA=="
}
```

En este caso el id es un identificador único del evento. El tipo corresponde a uno de los 10 tipos de eventos listados anteriormente. Luego viene la fecha en que fue generado el evento de acuerdo al servidor, el identificador único de transacción. el identificador único del tótem. Si el evento detalla un suceso exitoso la data asociada viene en el campo data. Si por el contrario es un error el detalle de este figura en los campos codigoError (Numérico entero) y mensajeError que corresponde a al mensaje detallado del error. campo error. **Estos campos son opcionales por lo que pueden ir o no en el JSON. Si estos campos no se incluyen eso refleja que no hubo problemas asociados.**
Finalmente la firma corresponde a la firma digital del mensaje que son bytes codificados en Base64.

Data por evento

Para efectos de completitud, en algunos casos se incluye en el JSON el campo error, pero como ya se mencionó es opcional.

transaccion_iniciada

Al iniciarse una transacción de autenticación de manera exitosa el evento asociado es:

```
{
  "idTransaccion": 1,
  "fecha": 11851200000,
  "tipo": "transaccion_iniciada",
  "id": 1,
  "idTotem": 1,
  "firma": "fvdzWLeXqST8JaOLxjERrA=="
}
```

Si sucede un error al iniciar la transacción, el código de error asociado es el 1000 y el mensaje base es "La transacción no se pudo iniciar".

```
{
  "idTransaccion": 1,
  "fecha": 11851200000,
  "tipo": "transaccion_iniciada",
  "id": 1,
  "idTotem": 1,
  "codigoError": 1000,
  "mensajeError": "La transacción no se puedo iniciar"
  "firma": "fvdzWLeXqST8JaOLxjERrA=="
}
```

transaccion_finalizada

Al finalizar una transacción de autenticación de manera exitosa el evento asociado es:

```
{
  "idTransaccion": 1,
  "fecha": 11851200000,
  "tipo": "transaccion_finalizada",
  "id": 1,
  "idTotem": 1,
  "firma": "fvdzWLeXqST8JaOLxjERrA=="
}
```

Al ocurrir un error al finalizar la transacción, el código de error asociado es 1001 y el mensaje base "La transacción no se pudo finalizar".

```
{
  "idTransaccion": 1,
  "fecha": 11851200000,
  "tipo": "transaccion_finalizada",
  "id": 1,
  "idTotem": 1,
  "codigoError" : 1001,
  "mensajeError": "La transacción no se pudo finalizar"
  "firma": "fvdzWLeXqST8JaOLxjERrA=="
}
```

timeout

En la data devuelve el número de segundos transcurridos desde que se inició la tansacción hasta que se gatilló el timeout. El timeout se puede gatillar, al leer la huella, al leer la cédula, al autenticar o en el total de tiempo asociado a la transacción de autenticación. En este evento siempre está presente el campo error con código 1020, glosa base "La operación no se pudo completar en el tiempo establecido"

```
{
  "idTransaccion": 1,
  "fecha": 11851200000,
  "tipo": "timeout",
  "id": 1,
  "idTotem": 1,
  "codigoError": 1020,
  "mensajeError": "La operación no se pudo completar en el tiempo establecido",
  "tiempo": 300
  "firma": "fvdzWLeXqST8JaOLxjERrA==",

}
```

huella_leida

Lectura exitosa

Si fue exitosa, la lectura, en la data devuelve el dedo y la minucia como bytes codificados en base64.

```
{
  "minucia": "hG6n/gb2FUASmtOK6fN1aQ==",
  "idTransaccion": 1,
  "fecha": 11851200000,
  "tipo": "huella_leida",
  "id": 1,
  "idTotem": 1,
  "dedo": 9,
  "firma": "fvdzWLeXqST8JaOLxjERrA=="
}
```


Lectura no exitosa

Si la lectura no fue exitosa, devuelve el error con el código 1041 y la glosa "No se pudo leer la huella dactilar".


```
{
  "idTransaccion": 1,
  "fecha": 11851200000,
  "tipo": "huella_leida",
  "id": 1,
  "idTotem": 1,
  "codigoError": 1041,
  "mensajeError": "No se pudo leer la huella dactilar"
  "dedo": 9,
  "firma": "fvdzWLeXqST8JaOLxjERrA=="
}
```


cedula_leida

Lectura exitosa

Si fue exitosa, la lectura de la cédula, en la data devuelve el rut y el nombre de la persona

```
{
  "rut": "1-9",
  "idTransaccion": 1,
  "nombre": "Alfredo Ruiz Arteaga",
  "fecha": 11851200000,
  "tipo": "cedula_leida",
  "id": 1,
  "idTotem": 1,
  "firma": "fvdzWLeXqST8JaOLxjERrA=="
}
```


Lectura no exitosa

Si la lectura no fue exitosa, devuelve el error con el código 1051 y la glosa base "No se pudo leer la cédula"


```
{
  "idTransaccion": 1,
  "fecha": 11851200000,
  "tipo": "cedula_leida",
  "id": 1,
  "idTotem": 1,
  "codigoError":1051,
  "mensajeError": "No se pudo leer la cédula",
  "firma" : "fvdzWLeXqST8JaOLxjERrA=="
}
```


dedo_cambiado

Se gatilla al enviar el comando de cambio de dedo utilizado para la lectura de la huella. El campo dedo, indica cual es el nuevo dedo a utilizar para la lectura de huella.


```
{
  "idTransaccion": 1,
  "fecha": 11851200000,
  "tipo": "dedo_cambiado",
  "id": 1,
  "idTotem": 1,
  "dedo": 10,
  "firma": "fvdzWLeXqST8JaOLxjERrA=="
}
```


Se se produce un error al cambiar el dedo, se devuelve el código 1042, y como menaje base "El dedo no se

pudo cambiar".

```
{
  "idTransaccion": 1,
  "fecha": 11851200000,
  "tipo": "dedo_cambiado",
  "id": 1,
  "idTotem": 1,
  "dedo": 10,
  "codigoError": 1042,
  "mensajeError": "El dedo no se pudo cambiar",
  "firma": "fvdzWLeXqST8JaOLxjERrA=="
}
```

huella_activada

Este evento se gatilla al activarse el dispositivo de lectura de huella.

```
{
  "idTransaccion": 1,
  "fecha": 11851200000,
  "tipo": "huella_activada",
  "id": 1,
  "idTotem": 1,
  "tipo_dispositivo": "lector_huella",
  "firma": "fvdzWLeXqST8JaOLxjERrA=="
}
```

Cuando se produce un error al activar el lector de huella el código de error corresponde al 1040 y el mensaje base a "No se pudo activar el lector de huella dactilar".

```
{
  "idTransaccion": 1,
  "fecha": 11851200000,
  "tipo": "huella_activada",
  "id": 1,
  "idTotem": 1,
  "tipo_dispositivo": "lector_huella",
  "codigoError": 1041,
  "mensajeError": "No se pudo activar el lector de huella dactilar"
  "firma": "fvdzWLeXqST8JaOLxjERrA=="
}
```

cedula_activada

Este evento se gatilla al activarse el dispositivo de lectura de cédula.

```
{
  "idTransaccion": 1,
  "fecha": 11851200000,
  "tipo": "cedula_activada",
  "id": 1,
  "idTotem": 1,
  "tipo_dispositivo": "lector_cedula",
  "firma": "fvdzWLeXqST8JaOLxjERrA=="
}
```

Al gatillarse un error al activar el léctor de cédula, el código asociado es el 1051, y el mensaje base "No se pudo activar el lector de cédula".

```
{
  "idTransaccion": 1,
  "fecha": 11851200000,
  "tipo": "cedula_activada",
  "id": 1,
  "idTotem": 1,
  "tipo_dispositivo": "lector_cedula",
  "codigoError": 1051,
  "mensajeError": "No se pudo activar el lector de cédula",
  "firma": "fvdzWLeXqST8JaOLxjERrA=="
}
```

reintentos_superado

Al superarse el número de reintentos para lectura de huella, lectura de cédula o de autenticación para una transacción se gatilla este evento. El campo error siempre está presente en este evento con código 1030 y glosa base "La cantidad de reintentos para la operación se ha superado".

```
{
  "idTransaccion": 1,
  "fecha": 11851200000,
  "tipo": "reintentos_superado",
  "intentos": 3,
  "id": 1,
  "idTotem": 1,
  "codigoError": 1030,
  "mensajeError": "La cantidad de reintentos para la operación se ha superado",
  "firma": "fvdzWLeXqST8JaOLxjERrA=="
}
```


autenticacion

Este evento se gatilla al invocarse la autenticación en la cédula (Match on card).

Invocación de autenticación exitosa

Si se logró invocar al proceso de manera exitosa devuelve en la data el resultado de la autenticación, la minucia y el dedo asociado. En este caso el campo resultado devuelve true en caso de que el matchoncard haya autenticado al usuario, false en caso contrario. Es importante, entender que no es un error que el usuario no haya sido autenticado. El resultado indica si la identidad fue verificada o no, pero en ambos casos el proceso de autenticación es exitoso.


```
{
  "minucia": "hG6n/gb2FUASmtOK6fN1aQ==",
  "rut": "1-9",
  "idTransaccion": 1,
  "nombre": "Jhon Foo",
  "resultado": true,
  "fecha": 11851200000,
  "tipo": "autenticacion",
  "id": 1,
  "idTotem": 1,
  "dedo": 9,
  "firma": "fvdzWLeXqST8JaOLxjERrA=="
}
```


Invocación de autenticación no exitosa

Si se produce un error al realizar la autenticación, devuelve el error con el código 1052 y la glosa base "No se pudo autenticar utilizando la cédula".

```
{
  "idTransaccion": 1,
  "fecha": 11851200000,
  "tipo": "autenticacion",
  "id": 1,
  "idTotem": 1,
  "codigoError": 1052,
  "mensajeError": "No se pudo autenticar utilizando la cédula",
  "firma": "fvdzWLeXqST8JaOLxjERrA=="
}
```


Resumen de los códigos de error

Todos los errores posee un código (codigoError) y un mensaje de error (mensajeError). Los mensajes poseen un glosa o mensaje básico, que es complementado con el detalle del error ocurrido.

Errores asociados a la transacción

- Error al iniciar la transacción. Código 1000, mensaje "La transacción no se pudo iniciar".
- Error al finalizar la transacción. Código 1001, mensaje "La transacción no se pudo finalizar".

Errores asociados a cualquier operación

- Error de timeout. Código 1020, mensaje "La operación no se pudo completar en el tiempo establecido".
- Error cantidad de reintentos superada. Código 1030, mensaje "La cantidad de reintentos para la operación se ha superado".

Errores asociados a la operación de lectura de huella

- Error al activar el lector de huella. Código 1040, mensaje "No se pudo activar el lector de huella dactilar".
- Error al leer la huella. Código 1041, mensaje "No se pudo leer la huella dactilar".
- Error al cambiar el dedo. Código 1042, menaje "El dedo no se pudo cambiar".

Errores asociados a la operación de la cédula

- Error al activar el léctor de cédula. Código 1050, mensaje "No se pudo activar el lector de cédula".
- Error al leer la Cédula. Código 1051, mensaje "No se pudo leer la cédula".
- Error al autenticar utilizando la cédula. Código 1052, mensaje "No se pudo autenticar utilizando la cédula".

Errores asociados al estado de un dispositivo

- Error de estado de un dispositivo. Código 1060, mensaje "Error del dispositivo".

API JAVASCRIPT

WSSocketConstructor

La API javascript provee de una función constructor que crea el canal de comunicación vía websocket. Dicha función es WSSocketConstructor, que recibe dos parámetros un token de autorización (llave simétrica conocida por el servidor remoto y el servidor matchoncard) y un funcion de tipo callback donde se envían los eventos recibidos desde el servidor matchoncard.

Ejemplo:

```
wssocket = new WSSocketConstructor("fvdzWLeXqST8JaOLxjERrA==", recibir);
```

En este caso la función recibir se define como

```
function recibir(data){
            /* Acá se procesa data que correspnde a un objeto JSON de tipo evento*/
            ...
            ...
        }
```

Métodos del objeto wssocket

El objeto obtenido a partir del constructor provee de los siguientes métodos

- **conectar**:Abre el websocket.
- **enviar**: Envía un comando al servidor de matchoncard. Los comandos son los descritos anteriormente.
- **cerrar**: Cierra la conexión

  Y de las siguentes propiedades

- **debug**: Propiedad booleana que habilita la traza por consola javascript. Por omisión su valor es false.
- **reintentos**: Número máximo de errores antes de cerrar la conexión. Su valor por omisión es 3
- **tiempoeco**: Tiempo en milisegundos del heartbeat. Valor por omsión 5000.

EJEMPLO DE USO DE LA API JAVASCRIPT

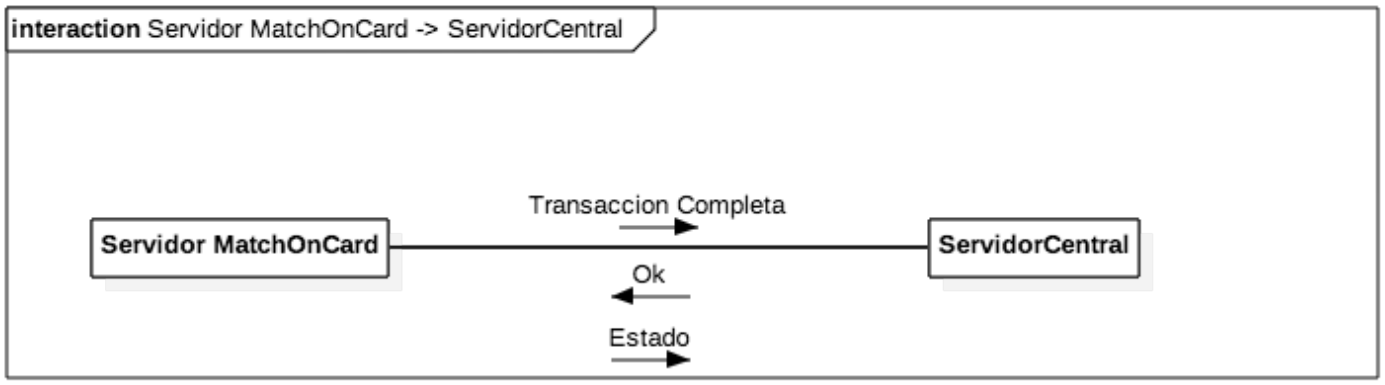
```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Title</title>
    <script src="/js/servidor.js"></script>
    <script>
        function recibir(data){
            console.log("Recibiendo");
            console.log(data);
        }

        wssocket = new WSSocketConstructor("123456", recibir);
    </script>

</head>
<body>
Main
<input type="button" onclick="wssocket.debug = !wssocket.debug" value="Debug!"/>
<input type="button" onclick="wssocket.conectar()" value="Conectar!"/>
<input type="button" onclick='wssocket.enviar({"token": wssocket.token,"comando": "in
iciar_transaccion"})' value="Enviar!"/>
<input type="button" onclick='wssocket.cerrar()' value="Cerrar!"/>
</body>
</html>
```

SERVICIOS A DISPONIBILIZAR EN EL SERVIDOR CENTRAL

Interacción servidor matchoncard servidor central



En el servidor central se deben habilitar dos servicios para reportar las transacciones y el estado de los dispositivos. Ambos servicios, son de tipo REST vía el método HTTP POST. Ambos servicios deben ser idempotentes

En caso de éxito el POST responderá el estadó HTTP (Status Code) 200 (si el recurso se actualizó), 201 (si el recurso se creó), y 202 (si se aceptó el requerimiento pero no ha sido procesado totalmente). Cualquiera de estos tres estados se considerará como exitoso y evitará el reenvío de la información.

Transacción

Las transacciones procesadas son enviadas a una URL del tipo

http://ip:puerto/api/transaccion

No es necesario que la ruta relativa sea api/transaccion, pero se recomienda. El JSON enviado a esa URL vía POST es el siguiente:

```
{
  "estado": "COMPLETADA",
  "minucia": "hG6n/gb2FUASmtOK6fN1aQ==",
  "rut": "1-9",
  "nombre": "Nombre",
  "resultado": true,
  "fecha": 11851200000,
  "id": 1,
  "idTotem": 1,
  "codigoError":1,
  "mensajeError": "Glosa de Error",
  "dedo": 9,
  "firma": "fvdzWLeXqST8JaOLxjERrA=="
}
```

Los campos error, minucia, resultado y dedo son opcionales. Los estados posibles son `COMPLETADA`, `CANCELADA` y `FINALIZADA_CON_ERROR`. En el caso de que el estado sea `COMPLETADA`, se incluyen los campos, minucia, resultado y dedo. En el caso de `FINALIZADA_CON_ERROR` se incluye el campo error.

Estado

El estado de los dispositivos es enviado a una URL del tipo:

http://ip:puerto/api/estado

No es necesario que la ruta relativa sea api/estado, pero se recomienda. El JSON enviado a esa URL vía POST es el siguiente:

```
{
  "dispositivo": "lector_huella",
  "estado": "KO",
  "fecha": 11851200000,
  "id": 1,
  "idTotem": 1,
  "codigoError":1060,
  "mensajeError": "Error del dispositivo"
  }
}
```

El campo error es opcional y solo se incluye en el caso de que el estado sea `KO`. Si el estado es `OK`, el dispositivo está funcionando adecuadamente. Los valores posibles para el campo dispositivo son `lector_huella` y `lector_cedula`.