Specification of the Alternative Interdiction Query Service (CAI-REST)

Version 2.2

July 2026

Contents

1 Objectives

Royal Decree 1613/2011, of 14 November, which implements Law 13/2011, of 27 May, on the regulation of gambling, with regard to the technical requirements of gambling activities, establishes that gambling operators must verify the identity and age of participants and must also check, in real time, the General Registry of Gambling Access Interdictions (RGIAJ) in order to activate gambling accounts.

In addition, the DGOJ provides each operator, every hour, with any changes that may have occurred in the RGIAJ, so that it can incorporate them into its customer database and carry out the necessary checks before paying out prizes.

The DGOJ provides this service through web services, as well as the participant identity verification service.

In this latter case, should the systems of the DGOJ, of the Intermediation Services of the Secretariat of State for Public Administrations, or of the Directorate-General of the Police (which ultimately provides the identity verification service) go down, gambling regulations allow such identification to be delayed for three days, although the technical rules recommend retrying after 30 minutes.

However, and having regard to the Eleventh and Twelfth sections of the Resolution of 12 July 2012 of the Directorate-General for the Regulation of Gambling, approving the provision that implements Articles 26 and 27 of Royal Decree 1613/2011, of 14 November, in relation to the identification of participants in games and the control of subjective prohibitions on participation, the check in the RGIAJ must be carried out in real time.

Therefore, and in order not to harm the interests of operators or the right of participants to take part in gambling activities, an alternative system for querying gambling access interdictions (CAI) is required to complement the web services provided by the DGOJ. Should these systems go down, it is still possible to maintain at least the verification of the participant's status in the RGIAJ, within the Contingency Plan established for this circumstance.

This Alternative Interdiction Query system will be activated by the DGOJ in the event of a prolonged outage of its systems, following the procedure indicated in section 3.4.

It should be stressed that, since the DGOJ's main system will be down when the CAIREST service is activated, no change can occur in the RGIAJ status of the users the operator had previously queried. This means that, as there are no RGIAJ status changes, the last information downloaded through the VerificarCambiosRGIAJ operation will remain valid for the entire time the CAIREST service is active. Therefore, the CAIREST service should only be used to check the RGIAJ status of those users who were never previously checked by the operator in question.

On the other hand, it should be recalled that the identity verification service (being a service whose use is optional for operators, and given that the legislation allows the DGOJ to defer received identification requests for up to three days) will not have an alternative query service in the event of a serious outage of the DGOJ's systems. When this occurs, operators may activate whatever documentary identity verification services they deem appropriate.

2 Document change control

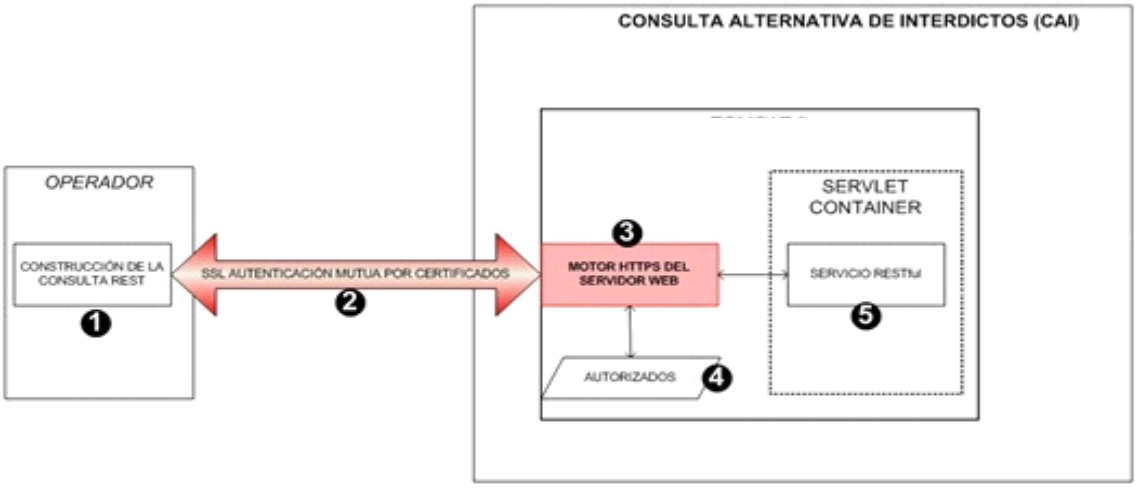
Version	Date	Description
0.1	SEP 2012	Initial version.
0.2	OCT 2012	Error corrections. Where the HMAC MD5 encryption algorithm is indicated, it should read HMAC SHA1
0.3	JAN 2013	Clarification on the use of certificates and the provision of the key file during the testing period.
1.0	FEB 2013	Activation of the CAIREST testing environment
1.1	FEB 2nd 2013	Actions to be taken once the conventional RGIAJ service is restored after RGIAJ activation (section 4.5)
1.2	MAR 2013	Information on when the CAIREST service should be used (section 1) The DNIs/NIEs used in the encryption process must always have 9 characters, so it will be necessary to pad with 0 until that number of

		characters is reached (section 4.3)
1.3	MAR 2nd 2013	The DNIs/NIEs used in the encryption process must have their non-numeric characters in uppercase (section 4.3)
2.0	NOV 2024	The libraries of the API are updated (section 4.11) and source code example (section 4.10)
2.1	MAY 2025	The encryption algorithm is parameterized to SHA256 The service access URL is changed for both PRE and PRO
2.2	JUL 2026	Update of the service registration (4.1) and the description of the environments (4.2)

3 Functional specification

The system provides a service for querying the General Registry of Gambling Access Interdictions file maintained by the DGOJ, in order to inform operators of the status of players when they register in the online gambling systems.

3.1 System description



3.2 Secure connection

- The operator will connect through an SSL channel. All accesses will be audited by the DGOJ.
- Requests must use the SSL channel (HTTPS), so they must be built with the operator's certificate so that the web server correctly establishes communication; the server, in turn, also identifies itself with a certificate. Operators must use the same certificate (depending on the environment) that they use to connect to the player verification service

3.3 Query requests

- Once the connection is established, the operator must build a request for the resource; this request is simply a GET request to the resource over the HTTPS protocol, that is, a request similar to the following:

  https://cairest.ordenacionjuego.gob.es/CAIREST/rest/getStatus?id=qfVHE0iVt5DvNIGCMtxjs87cyfHZmBaVFY3bmDdwQjA=

  The id value is the BASE64 representation of the result of applying the search-key composition algorithm, which is built with a HASH by means of an HMAC SHA256 algorithm with a password. http://en.wikipedia.org/wiki/HMAC.

- The web server directs the request to the servlet engine (since a static resource is not being requested). This servlet will use JAX-RS and the reference implementation (Jersey) for RESTful services.

3.4 Responses of the Alternative Interdiction Query service (CAI-REST).

The service receives the search key built by the operator, checks the operator's identity, and if the operator is valid it performs the search in the interdictions file. It builds a response in JSON format. The possible responses from the system are the following:

Response code	Meaning
{"Code":"COD001"}	The user is registered in the RGIAJ
{"Code":"COD002"}	The user is not registered in the RGIAJ
{"Code":"ERR001"}	Technical error (internal error)
{"Code":"ERR002"}	Operator not authorized
{"Code":"COD006"}	The request has no certificate

3.5 Description of the key XML file format

The key will be provided in an xml file with the following structure:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
 <key>
 <value>loiJQ8TmtmOomM+A+KkbbXakomJ0/NR1RQWa3YxM5OU=</value>
 </key>
```

Its root element is the key element, with a value element whose value indicates the key that has been used to generate the HMAC SHA256.

The key indicated above may be used for testing in the Pre-production environment.

4 Annexes

4.1 Service registration.

All operators that have requested registration for the production web services will automatically have access to the CAI-REST service.

Before being able to use the service, they must report, through the fee-based communications procedure their technical contacts responsible for CAIREST. It should preferably be a specific email account for this function and not a personal account.

In the event that the Plan is activated, the DGOJ will send to that email account the activation notification and an attached file with the encryption key for the HASH.

The reply email will communicate the key to be used in both environments. In the pre-production environment the key will not change. It is the one indicated in section 3.5. In the production environment the key will be changed when the actual contingency is activated, communicating the new key by an email sent to the address notified in the registration request. As long as there is no contingency, this environment can be tested with the same data as in pre-production.

4.2 Description of the environments

There are two environments for accessing the CAIREST service:

- **pre-production**: invocation through the URL https://cairest-pre.ordenacionjuego.gob.es/CAIREST/rest/getStatus?id=qfVHE0iVt5DvNIGCMtxjs87cyfHZmBaVFY3bmDdwQjA=

- **production**: invocation through the URL https://cairest.ordenacionjuego.gob.es/CAIREST/rest/getStatus?id=qfVHE0iVt5DvNIGCMtxjs87cyfHZmBaVFY3bmDdwQjA=

In the pre-production environment, it will be possible to operate using those certificates reported to the DGOJ for interoperating with the pre-production player verification environment.

In the production environment, it will be possible to operate using those certificates reported to the DGOJ for interoperating with the production player verification environment.

At any time, operators may query the following DNIs/NIEs in the pre-production environment. The system must respond with COD0001 for the following DNIs/NIEs:

00000048W
X0000019L
00000024R

and COD002 for the DNIs/NIEs:

00000034B
00000057B
X0000052Y
00000001R

To avoid errors, when the contingency plan is activated in production, the pre-production environment will be stopped.

NOTE: when carrying out RGIAJ presence checks based on a hash of the DNI/NIE, it is important to stress that:

- note that the documents used are determined by a 9-character identifier, padding with zeros on the left until that number is reached
- note that non-numeric characters are used in uppercase format, since the hash calculation operation yields a different result for a DNI

The two previous conditions are due to the fact that the result of a hash calculation is different for each of the following cases, with only the first example of each block being valid:

- 00000034B
- 00000034b
- 034B
- 034b

- X0000052Y
- X0000052y
- x0000052Y
- X52Y

Once the contingency plan is activated, the production environment will respond to operators' real requests, taking into account their actual status

in the RGIAJ database. When the plan is deactivated, the test data will be reloaded into the preproduction environment.

4.3 Calculating the hash of a DNI/NIE

The DNI of the participant to be queried will be hashed using an HMAC SHA256 encryption algorithm with a key. This key will be provided in an xml file attached to the email notifying the activation of the Contingency Plan.

The DNI/NIE must always have 9 characters, and must be padded with as many 0s on the left as necessary to reach that figure.

The non-numeric characters used in the DNI/NIE must always be in uppercase.

4.4 Activating the Contingency Plan

When it is not possible to verify players' status in the RGIAJ through the web services for a prolonged period due to technical problems, the DGOJ will activate the contingency plan to allow the alternative query of the Interdictions file (CAI). To do so, it will bring up the corresponding services on the emergency server in the production environment, it will upload the Hash identifiers of all those registered in the RGIAJ up to the previous hour, and it will notify operators by email of the service activation and the encryption key for the identifiers (DNI/NIE). This Plan will not be activated in the event that the web services allow querying the status in the RGIAJ, even if the identity verification query is not working.

4.5 Operation with the Contingency Plan activated

When the CAIREST service is active, only those DNIs/NIEs for which, prior to the activation of the CAIREST Service, the RGIAJ status had never been queried (conventional RGIAJ status query services) should be consulted.

Those DNIs/NIEs whose RGIAJ status was previously queried will not have changed their status, and the information obtained at the time of the query remains valid, as do the corresponding updates available through the VerificarCambiosRGIAJ operation.

To avoid errors, when the contingency plan is activated in production, the pre-production environment will be stopped.

4.6 Deactivating the Contingency Plan

Once the ability to query the RGIAJ through the web services is restored, the DGOJ will notify its deactivation by email, proceeding to disable the querying of real data through the contingency system.

To ensure that those DNIs/NIEs that were queried through the CAIREST service become incorporated into the DGOJ's information system so that they are taken into account in the RGIAJ status-change alert mechanism (the VerificarCambiosRGIAJ operation of the conventional web services), it is necessary that operators, once the conventional service has been restored, query again the RGIAJ status of all those DNIs/NIEs they queried while the CAIREST service was active. Failing to do so, possible status changes of those customers (and especially those that returned a negative RGIAJ presence response (COD002)) will not be taken into account in the VerificarCambiosRGIAJ process.

4.7 Contingency Plan drills

The DGOJ will carry out, with the frequency deemed appropriate in its security plan, a drill of Contingency Plan activation that will include the activation of this service, as well as any other procedure that may be incorporated in the future. Such activation will not affect the conventional RGIAJ or identity query service, affecting only the contingency service. The aim is therefore to respond with real data through this system. Operators will be notified of the start and end of the drill so that they can test their systems.

4.8 Example of using the CAIREST service

The development of the client to query the CAI service is the responsibility of the gambling operators.

However, the DGOJ provides an API, with its dependencies, so that the operator can develop a client in Java technology, available at https://www.ordenacionjuego.es/operadores-juego/informacion-operadores/supervision-control/servicios-web-verificacion section Alternative Query to the General Registry of Gambling Access Interdictions.

The DGOJ does not provide support for this software, nor does it assume any responsibility for the operation thereof. Likewise, no support will be provided regarding the content thereof, nor is there

any responsibility to solve the problems that may be detected in its operation.

This client is designed to be integrated within a Java development.

The JAR provided (API) offers functionality both for encoding the searched id in HMAC, and for making the SSL call to the CAIREST service.

We must place the JAR provided on the application's CLASSPATH.

To perform the conversion to HMAC format, we carry out the following steps.

The HmacUtils class is imported

```
import com.dgoj.crypto.utils.HmacUtils;
```

We make a call to the static method HmacUtils.generateHMAC with the appropriate parameters.

```
HmacUtils.generateHMAC(idToConvert, new FileInputStream(keyFilePath));
```

Or

```
HmacUtils.generateHMAC(idToConvert, key);
```

The parameters are described below:

idToConvert: This is a String whose value is the ID that we want to convert to HMAC format.

keyFilePath: This is a String whose value is the PATH of the file that contains the KEY (provided by the DGOJ).

or

key: This is a String whose value is the secret key to generate the HMAC.

The call to the REST service is made as follows.

The classes CAIRESTService and ConnectionParameters are imported.

```
import com.dgoj.toolkit.net.CAIRESTService;
import com.dgoj.toolkit.params.ConnectionParameters;
```

We create an instance of said classes and call the searchRestService method with the appropriate parameters.

```
ConnectionParameters conParams = new ConnectionParameters(restHost, restHostSSLPort, hmacId, keyStorePath, keyStorePass, trustStorePath, trustStorePass);


CAIRESTService caiService = new CAIRESTService();
String caiResponse = caiService.searchRestService(conParams);
```

We describe the parameters below.

restHost: Address (IP or DNS name) of the server where the CAIREST Web application is installed.

restHostSSLPort: SSL port enabled on the server, 443 (production) or 1443 (pre-production).

hmacId: Identifier to search for in the interdictions file; it must be in HMAC SHA256.

keyStorePath: Path of the keyStore.

keyStorePass: Password of the keyStore.

trustStorePath: Path of the trustStore.

trustStorePass: Password of the trustStore

4.9 Requirements.

A certificate for the client is required within a JKS-type keystore.

A JKS-type trusted certificate store is required, in which we will keep the public part of the server's certificate or the certificate of the CA that signed it.

4.10 Examples.

Example of a call to a client that uses the API. (available at https://www.ordenacionjuego.es/operadores-juego/informacion-operadores/supervision-control/servicios-web-verificacion section Alternative Query to the General Registry of Gambling Access Interdictions)

Production.

java -jar ClienteCairest.jar cairest.dgojuego.es 443 00000015S Mikeystore.jks miclavekeystore MItrustore.jks miclavetrustore key.xml

Pre-production.

java -jar ClienteCairest.jar cairest.dgojuego.es 1443 00000015S Mikeystore.jks miclavekeystore MItrustore.jks miclavetrustore key.xml

Source code example of using the API.

```
import java.io.FileInputStream;
import java.io.IOException;
import jakarta.xml.bind.JAXBException;
import com.dgoj.crypto.utils.HmacUtils;
import com.dgoj.toolkit.error.SSLUtilsError;
import com.dgoj.toolkit.net.CAIRESTService;
import com.dgoj.toolkit.params.ConnectionParameters;

public class Cliente {

  public static void main(String[] args) throws SSLUtilsError, JAXBException, IOException
{

        if (args == null || args.length < 8) {
                System.out.println("\n*************** ERROR ***************");
                System.out.println("* Faltan parametros: <HostName> <Port>
                <idToSearch> <KeyStorePath> <KeyStorePass> <TrustStorePath>
                <TrustStorePass> <KeyFilePath>");
                System.out.println("*************************************");
                return;
        }
        String restHost = args[0];
        int restHostSSLPort = Integer.valueOf(args[1]).intValue();
        String idToSearch = args[2];
        String keyStorePath = args[3];
        String keyStorePass = args[4];
        String trustStorePath = args[5];
        String trustStorePass = args[6];
        String keyFilePath = args[7];

        String hmacId = HmacUtils.generateHMAC(idToSearch, new
```

```
                    FileInputStream(keyFilePath));
            ConnectionParameters conParams = new ConnectionParameters(restHost,
            restHostSSLPort, hmacId, keyStorePath, keyStorePass, trustStorePath,
            trustStorePass);
            CAIRESTService caiService = new CAIRESTService();
            String caiResponse = caiService.searchRestService(conParams);

            System.out.println("\n************* PARAMETROS *************");
            System.out.println("*        Server: " + conParams.getRestHost());
            System.out.println("*          Port: " + conParams.getRestHostSSLPort());
            System.out.println("*        hmacId: " + conParams.getIdToSearch());
            System.out.println("*   KeyStorePath: " + conParams.getKeyStorePath());
            System.out.println("* TrustStorePath: " + conParams.getTrustStorePath());
            System.out.println("*    KeyFilePath: " + keyFilePath);
            System.out.println("****************************************");
            System.out.println("\n" +caiResponse +"\n");

      }
    }
```

4.11 Dependencies

The libraries to include in the project are the following

- log4j-core-2.24.1.jar
- log4j-api-2.24.1.jar
- commons-codec-1.17.1.jar
- commons-logging-1.3.4.jar
- jakarta.xml.bind-api-4.0.2.jar
- jakarta.activation-api-2.1.3.jar
- jaxb-core-4.0.5.jar
- jaxb-runtime-4.0.5.jar
- istack-commons-runtime-4.2.0.jar
- httpclient-4.5.14.jar
- httpcore-4.4.16.jar

These libraries can be obtained by the operator over the Internet or downloaded from a .zip file provided by the DGOJ at https://www.ordenacionjuego.es/operadores-juego/informacion-operadores/supervision-control/servicios-web-verificacion
section Alternative Query to the General Registry of Gambling Access Interdictions.