Rapport au sujet du caractère exportable du système espagnol d'e-APP

Ministère de la Justice

Mai 2011

TABLE DES MATIERES

1. INTRODUCTION ET OBJECTIFS

Le système d'apostille électronique (désormais e-APP) du ministère de la Justice de l'Espagne a été mis au point sous la prémisse de la possibilité d'utilisation, outre par les autorités émettrices d'apostilles relevant du cadre des compétences du ministère lui-même, par d'autres organismes ayant des compétences d'émission d'apostilles pour des actes publics dans d'autres domaines. Pour cela, il a été conçu en modules et en utilisant les paradigmes de " injection de dépendances " (" connecteurs ") et de " orientation services " qui permettent de remplacer en toute simplicité certains de ses composants par d'autres conformes aux nécessités et aux exigences technologiques de l'organisme d'implantation, ainsi que de proposer une architecture ouverte et fortement interopérable.

L'objectif du présent document est d'informer au sujet des possibilités d'exportation du système e-APP mis au point par le ministère de la Justice de l'Espagne vers d'autres organismes ayant des compétences d'émission d'apostilles dans d'autres domaines, tant à l'échelle nationale qu'internationale.

2. DESCRIPTION DU SYSTÈME D'e-APP

2.1. Description générale du système

- Le système e-APP est constitué comme une application en architecture web J2EE résidant dans des serveurs centralisés, à laquelle les utilisateurs finals accèdent par l'intermédiaire d'un navigateur. On peut estimer que le système est constitué par deux sous-systèmes :
  - **Sous-système du traitement d'apostilles électroniques**. Ce sous-système est à usage interne des utilisateurs chargés d'effectuer les démarches concernant les apostilles. Les utilisateurs ont accès à ce sous-système par l'intermédiaire d'un simple navigateur.
  - **Sous-système de publication sur Internet**. Ce sous-système permet de réaliser les actions suivantes à travers Internet et au moyen d'un navigateur :
    - le téléchargement d'une apostille électronique émise (par le requérant, avec authentification)
    - l'interrogation du Registre électronique des apostilles (par les autorités étrangères, sans authentification)
- Le système repose sur des services qui lui procurent des fonctions de base nécessaires, telles que la Plate-forme de signature électronique ou le Service de génération de codes sécurisés de vérification, et il expose des services à son tour pour pouvoir être intégré à d'autres applications à travers la dénommée " couche d'intégration " (architecture SOA)
- Les produits et technologies employés pour l'implémentation au ministère de la Justice de l'Espagne sont les suivants :
  - Application en langage Java sur WAS 6.1
  - SGBDR: Oracle 10g
  - Plate-forme de e-Signature : ASF de TB Solutions
  - Format des documents : Adobe PDF
  - Standards de signature électronique :
    - PAdES pour la signature des e-apostilles
    - Les formats de signature d'actes publics XAdES, CAdES et PAdES sont admis
- Les exigences technologiques des postes clients depuis lesquels le système e-APP va être utilisé sont les suivantes :
  - **Sous-système du traitement**: Ce sous-système est destiné au personnel administratif des organisations concernées par les processus du traitement et d'émission d'apostilles électroniques. Dans ces cas, les exigences de logiciel des stations clientes seraient :
    - Système d'exploitation : Windows 2000, XP, Vista ou 7
    - Navigateurs : Microsoft Internet Explorer 7 ou 8
    - Lecteur de format PDF : Adobe Reader X
    - Logiciel pour des dispositifs de lecture de cartes cryptographiques X509 v3

- **Sous-système de publication sur Internet**. Ce sous-système est destiné aux requérants d'apostilles électroniques, en vue de leur téléchargement en ligne, ou bien aux autorités étrangères destinataires des apostilles électroniques, pour leur vérification en ligne. Dans ces cas-là, les exigences de logiciel des stations clientes seraient les suivantes :
  - Système d'exploitation : Windows 2000, XP, Vista, 7 ou Linux
  - Navigateurs : Microsoft Internet Explorer 7 ou 8, Mozilla Firefox 2.0 ou supérieur, Google Chrome 10
  - Lecteur de format PDF : Adobe Reader X

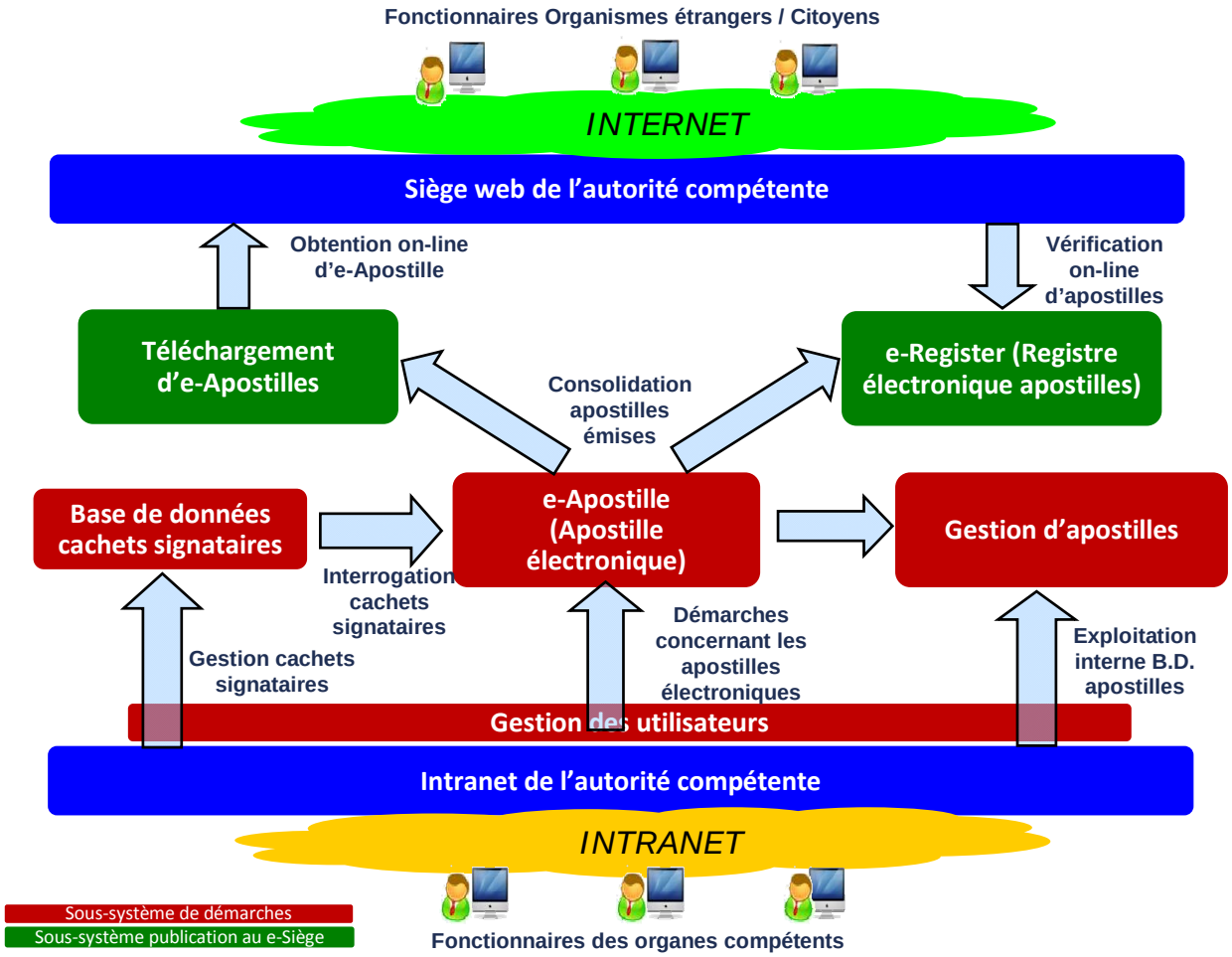
2.2. Décomposition fonctionnelle

Les deux sous-systèmes qui composent le système e-APP (Sous-système du traitement d'apostilles et Sous-système de publication sur Internet) se décomposent, à des fins fonctionnelles, en divers modules :

- **Sous-système du traitement** :
  - Module d'émission d'apostilles électroniques (e-Apostille)
    - Génération et signature électronique des apostilles électroniques.
  - Module de base de données de signatures
    - Inclusion, radiation et modification de signatures manuscrites numérisées et/ou de signatures électroniques d'autorités dont la signature doit être reconnue et vérifiée par l'autorité émettant l'apostille
  - Module de gestion des apostilles
    - Consultations internes et maintenance de données de demandes d'apostilles
  - Module de gestion des utilisateurs
    - Inclusion, radiation, modification et attribution de profils des utilisateurs des différents modules du Sous-système du traitement.

- **Sous-système de publication sur Internet** :
  - Module de Registre électronique des apostilles (e-Register).
    - Registre électronique où demeurent enregistrées et classifiées toutes les apostilles émises tant en format papier qu'électronique et consultable sur Internet
  - Module de téléchargement des apostilles électroniques
    - Il permet au requérant de télécharger l'apostille électronique par l'intermédiaire d'Internet

Les modules cités et leurs interactions peuvent être représentés au moyen du diagramme suivant :



Le système d'e-APP dispose en outre d'une couche d'intégration sous forme de Services Web, qui permet l'accès aux fonctions du système depuis d'autres systèmes d'intégration. Dans cette couche d'intégration, deux Services Web peuvent être différenciés :

- Service Web d'intégration avec e-Apostille (création de flux du traitement d'apostilles). Prévu pour l'intégration avec des applications externes génératrices d'actes publics électroniques

- Service Web d'intégration avec e-Register : inclusion d'apostilles dans e-Register. Il permet d'utiliser e-Register indépendamment d'e-Apostille

2.3. Description fonctionnelle

La présente rubrique décrit la fonctionnalité de base assurée par le Système e-APP.

Sous-système de traitement des e-Apostilles

Module e-Apostille :

Traitement des apostilles pour d'actes publics sur papier

- Le requérant dépose l'acte public sur papier.
- Un utilisateur chargé du traitement (dont le rôle est de " enregistrer les demandes ") crée une nouvelle demande d'apostille et remplit les données de celle-ci.
- De façon facultative, il numérise l'acte au moyen d'un scanner. Dans ce cas, l'image numérisée de l'acte public est incluse en annexe de l'apostille électronique.
- L'utilisateur chargé du traitement enregistre la demande. Celle-ci passe au stade de " Validation de signature de l'acte public ". Un justificatif de demande est émis par l'application que le chargé du traitement imprime et remet au requérant.
- Un utilisateur ayant le rôle de " Validateur " (il peut s'agir de la même personne ayant enregistré la demande) valide la signature manuscrite et/ou le sceau de l'acte public. Pour ce faire, il peut afficher depuis l'application l'acte public (s'il a été numérisé) et également les images de la signature et du sceau du signataire. Si tout est correct, l'utilisateur marque la demande comme " Valide ". La demande passe à l'état " Validée ".
- Un utilisateur ayant le rôle de " Signataire " (il peut s'agir de la même personne ayant validé la demande) émet et signe le document d'apostille électronique. Ce document est un fichier PDF avec le format trilingue de l'apostille proposé par la Conférence de La Haye (espagnol – anglais – français) qui contient, s'il a été numérisé, en annexe incrustée, le fichier d'image de l'acte public numérisé, qui est signé électroniquement avec la signature PAdES. Une fois signée, la demande d'apostille passe à l'état " Émise ".
- Un utilisateur chargé du traitement (dont le rôle est de " notifier les demandes ") récupère l'apostille et la notifie au requérant selon la voie retenue :
    - Sur place : la personne chargée du traitement doit contacter (par e-mail ou par téléphone) le requérant pour l'informer qu'il peut se rendre au siège pour récupérer l'apostille électronique. Celle-ci peut être remise en format électronique (sur une *clé USB* ou un autre support similaire) ou en format papier. Lorsque le requérant a récupéré l'apostille, la personne chargée du traitement marque sur l'application que la demande a été " Notifiée ".
    - Par courrier postal : ce cas n'est applicable que si l'apostille va être remise en format papier avec l'acte public apostillé. L'envoi postal une fois réalisé, la personne chargée du traitement marque sur l'application que la demande a été " Notifiée ".
    - Par « téléchargement de l'Apostille électronique » en ligne (en cas de demande expresse de l'utilisateur). Dans ce cas-là, la personne chargée du traitement n'a qu'à appuyer sur le bouton " Notifier " et l'application rend dès lors disponible l'apostille depuis le module de " Téléchargement d'apostilles " du sous-système de publication sur Internet.

Traitement des apostilles d'actes publics électroniques

- Le requérant dépose l'acte public en format électronique (au moyen d'une *clé USB* ou similaire).
- Un utilisateur chargé du traitement (dont le rôle est de " enregistrer les demandes ") crée une nouvelle demande d'apostille et remplit les données de celle-ci, puis lui joint le fichier de l'acte public électronique.
- L'utilisateur chargé du traitement enregistre la demande. Celle-ci passe au stade de " Validation de signature de l'acte public ". Un justificatif de demande est émis depuis l'application que le chargé du traitement imprimera et remettra au requérant.
- Le système, de façon automatique et sans que personne ne s'en occupe, valide la signature électronique de l'acte public et, si elle est correcte, cherche le certificat avec lequel a été signé l'acte public dans la Base de données des signatures. S'il trouve un signataire en vigueur pour ce certificat, il considère la demande correcte ; s'il ne parvient pas à valider la signature ou s'il ne trouve pas le certificat parmi les signatures, la demande passe à l'état " Refusée ".
- Un utilisateur ayant le rôle de " Signataire " émet et signe le document d'apostille électronique. Ce document est un fichier PDF avec le modèle d'Apostille trilingue proposé par la Conférence de La Haye (espagnol – anglais – français) qui contient en annexe incrustée le fichier de l'acte public électronique, et qui est signé électroniquement avec une signature PAdES. Une fois signée, la demande d'apostille passe à l'état " Émise ".
- Un utilisateur chargé du traitement (dont le rôle est de " notifier les demandes ") récupère l'apostille et la notifie au requérant selon la voie retenue :
    - Sur place : la personne chargée du traitement doit contacter (par e-mail ou par téléphone) le requérant pour l'informer du fait qu'il doit se rendre au siège pour récupérer l'apostille électronique. Celle-ci ne peut être remise qu'en format électronique (sur une clé USB ou un autre support similaire). Lorsque le requérant a récupéré l'apostille, la personne chargée du traitement marque sur l'application que la demande est " Notifiée ".
    - Par « téléchargement de l'Apostille électronique » en ligne (lorsque l'utilisateur en aura fait la demande expresse). Dans ce cas, la personne chargée du traitement n'a qu'à appuyer sur le bouton " Notifier " et l'application rend disponible dès lors l'apostille depuis le module de " Téléchargement d'apostilles " du sous-système de publication sur Internet.

Module de gestion des signatures :

Il permet de réaliser une maintenance de la Base de données des signatures, y compris les opérations suivantes :

- Inclusion d'une nouvelle signature, avec ses données de nom, fonction, entité et dates d'effet. Il faut indiquer également le type de signature (manuscrite ou numérique) ; en fonction de ce type, un ou deux fichiers doivent être annexés au registre :
    - Si la signature est de type manuscrite, un fichier avec la signature manuscrite numérisée doit être annexé, ou bien un fichier avec la signature numérisée, ou les deux. Ces fichiers d'image doivent être sous format JPEG.
    - Si la signature est de type numérique, un fichier au format DER doit être annexé avec les références électroniques du signataire (exportation au format DER de la partie publique du certificat utilisé pour signer).

- Recherche et consultation des signatures, avec possibilité de divers critères de recherche
- Modification d'une signature. Au préalable, il est nécessaire d'avoir réalisé une consultation pour sélectionner la signature à modifier ; ensuite, la façon de procéder est similaire à celle de l'inclusion.
- Élimination d'une signature. Au préalable, il est nécessaire d'avoir réalisé une consultation pour sélectionner la signature à supprimer ; une signature ne peut être éliminé que s'il n'est pas en cours de référencement par une demande d'apostille.

Module de gestion d'apostilles :

Il permet de réaliser une exploitation de la Base de données des demandes d'apostille, y compris les opérations suivantes :

- Recherche et consultation des demandes, avec possibilité de divers critères de recherche.
- Modification d'une demande. Au préalable, il est nécessaire d'avoir réalisé une consultation pour sélectionner la demande à modifier ; ne peuvent être modifiées que les demandes qui ne sont pas arrivées à l'état de " Validée ".
- Élimination d'une demande. Au préalable, il est nécessaire d'avoir réalisé une consultation pour sélectionner la demande à supprimer ; une demande ne peut être supprimée que si elle n'est pas arrivée à l'état de " Émise ".
- Émission d'une liste de demandes, qui permet de remplacer le " livre registre des apostilles " sur papier traditionnel. Au préalable, il est nécessaire d'avoir réalisé une consultation pour sélectionner l'ensemble de demandes à inclure dans la liste.

Module de gestion des utilisateurs :

Il permet de réaliser une maintenance des utilisateurs du système, y compris les opérations suivantes :

- Inclusion d'un utilisateur, avec indication de ses données personnelles, mot de passe initial et autorisations.
- Recherche et consultation des utilisateurs, avec possibilité de divers critères de recherche.
- Modification d'un utilisateur. Au préalable, il est nécessaire d'avoir réalisé une consultation pour sélectionner l'utilisateur à modifier ; ensuite, la façon de procéder est similaire à celle de l'inclusion.
- Radiation d'un utilisateur. Au préalable, il est nécessaire d'avoir réalisé une consultation pour sélectionner l'utilisateur à supprimer.

Sous-système de publication sur Internet :

Module de registre électronique des apostilles (e-Register) :

Il permet aux autorités étrangères recevant les apostilles (ou à toute personne intéressée) de réaliser plusieurs tâches de vérification de celles-ci. Pour l'une quelconque de ces tâches il est nécessaire que l'intéressé introduise trois données qui identifient l'apostille (et qui y sont consignées) :

- Numéro d'apostille
- Date d'émission
- Code Sécurisé de Vérification (CSV)

Le " Code Sécurisé de Vérification " est un code alphanumérique qui identifie de façon indubitable le document d'apostille électronique, qui en établit le lien avec l'organe de l'administration publique signataire de celle-ci, et qui permet la vérification de son intégrité et authenticité au moyen de l'accès au site électronique de cet organisme. Celle-ci est une figure prévue par la législation espagnole qui permet d'accorder aux copies imprimées de l'apostille électronique la valeur juridique d'un document signé électroniquement. Au regard de l'utilisation du système d'apostille électronique dans d'autres États n'ayant pas envisagé cette figure légale, il suffirait d'associer à chaque apostille un code alphanumérique assurant qu'elle identifie de façon unique une apostille émise à une date donnée (c'est-à-dire que le code ne devrait pas se répéter pour une même date d'émission).

Ces données une fois introduites, l'intéressé peut :

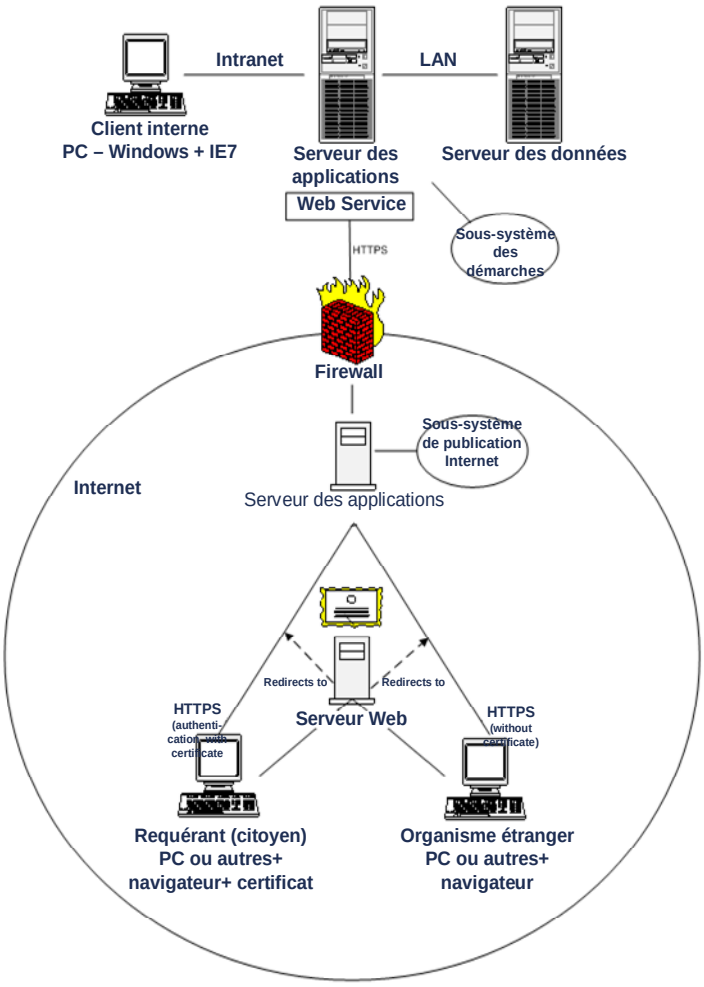
- **Valider l'apostille**. Cette option informe simplement l'intéressé qu'il existe en effet une apostille dans le système comportant les données fournies, et lui permet d'afficher sa " version imprimable " (version non signée ne contenant pas l'acte public comme annexe) à des fins de confrontation visuelle.
- **Valider l'acte public**. Si l'intéressé a reçu l'acte public sous format électronique, outre l'apostille elle-même, il peut vérifier au moyen de cette option que cet acte public coïncide exactement avec celui qui a été apostillé. Pour ce faire, l'intéressé doit " télétransmettre " le fichier de l'acte public ; le module e-Register prendra une " empreinte numérique électronique " (code *hash*) de ce fichier et le comparera à l'empreinte stockée associée à l'apostille. Si les deux empreintes coïncident, il informera l'intéressé que l'acte public télétransmis est exactement le même que celui qui a été apostillé.
- **Vérifier la signature de l'apostille**. Si l'intéressé qui reçoit une apostille électronique ne dispose pas d'un outil ayant la capacité d'ouvrir et de vérifier la signature de documents PAdES (par exemple, Adobe Reader version X), ou si pour toute autre raison il n'est pas à même de vérifier la signature de l'apostille avec son propre *software*, il peut procéder à " télétransmettre " le fichier d'apostille électronique et, au moyen de cette option, la signature de l'apostille est vérifiée (y compris l'éventuelle révocation du certificat avec lequel elle a été signée) sur le serveur du ministère de la Justice.

Module de téléchargement d'apostilles :

Il permet à un requérant de télécharger sur son ordinateur l'apostille électronique par l'intermédiaire d'Internet celle-ci une fois émise. Pour cela, il doit s'authentifier avec un nom d'utilisateur et un mot de passe. Ces données figurent sur le justificatif de demande que la personne chargée du traitement aura remis au requérant. Une fois authentifié, le requérant peut procéder à télécharger sur son ordinateur depuis le navigateur le fichier d'apostille électronique.

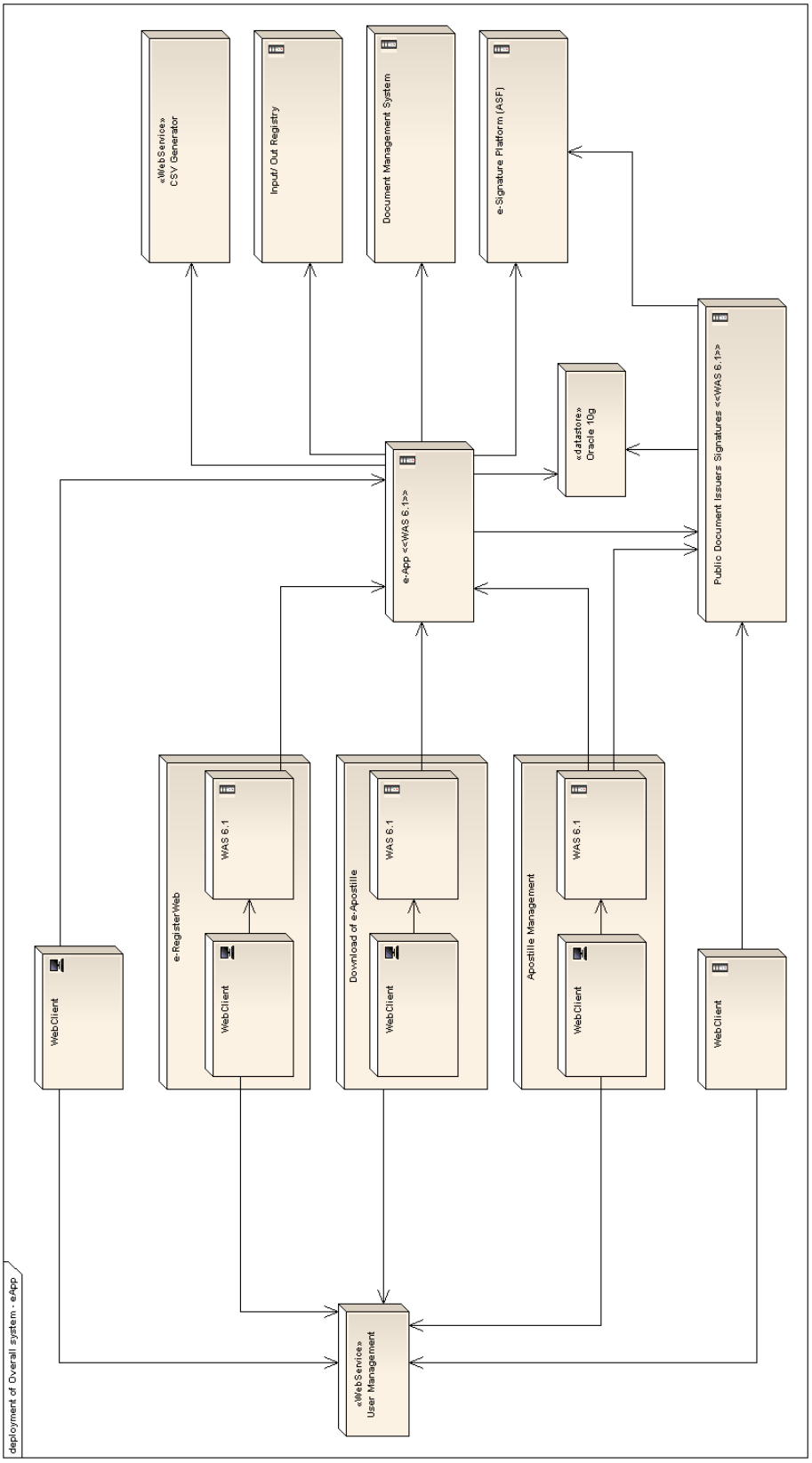
2.4. Architecture de déploiement du système

L'architecture de déploiement du système d'e-APP peut être représenté au moyen du diagramme suivant :



2.4. Composants bas niveau

De façon plus détaillée, nous trouvons la disposition suivante des composants du système d'e-APP



3. EXPORTATION DU SYSTÈME D'e-APP

3.1. Remplacement des connecteurs

Grâce à sa conception modulaire basée sur des " connecteurs ", le système du traitement d'apostilles électroniques permet de pouvoir remplacer certains des modules employés dans sa construction et qui assurent la communication entre le système e-APP et d'autres systèmes corporatifs par d'autres conformes aux nécessités de l'organisme qui va l'utiliser.
Un " connecteur " est, essentiellement, une classe implémentant une interface qui agit comme " contrat " entre l'application utilisatrice du connecteur et le connecteur lui-même, en établissant ainsi une forme standardisée de communication entre les deux. Ceci permet que l'on puisse écrire une nouvelle implémentation du connecteur sans qu'il soit nécessaire d'altérer le code de l'application, à la seule condition que le connecteur assure l'interface " convenue ".

Les connecteurs qui peuvent être remplacés sont les suivants :

- **Connecteur avec le service de génération de Codes Sécurisés de Vérification (CSV)**. Ce connecteur est chargé et invoqué par l'application e-Apostille lorsqu'il est nécessaire de générer le code de vérification qui identifie l'apostille. L'implémentation utilisée par le ministère de la Justice de l'Espagne invoque un service web horizontal du ministère lui-même qui se charge de générer ce code. On peut substituer à ce connecteur un autre générant un code de vérification avec toute autre procédure ; la seule restriction est que le code de vérification doit être une chaîne alphanumérique de 50 caractères maximum qui doit assurer l'unicité du triplet " nº d'apostille " / " date d'émission " / " code de vérification ".
  L'interface que doit présenter ce connecteur est la suivante :

```
public interface IConectorCSV extends IConector {
       public String obtenerCSV(String aplicacion, String tipoServicio)
                     throws GenericException;
}
```

  C'est-à-dire que le connecteur doit être une classe Java qui implémente une méthode **obtenerCSV** qui recevra deux arguments : **aplicacion** et **tipoServicio**. L'application d'apostille invoquera cette méthode en passant comme valeur pour les deux arguments le string " APOSTILLE ". Ces arguments sont une condition requise du service utilisé par le ministère de la Justice, mais dans toute autre implémentation du connecteur ils peuvent être ignorés. La seule chose que devrait faire cette méthode est de générer un code alphanumérique de 50 caractères maximum assurant qu'il ne se répète pas pour une même date et pour un même numéro d'apostille. Comme proposition, l'on pourrait générer un code incluant la date-heure du système (an-mois-jour-heure-minute-seconde-millisecondes) + un numéro aléatoire, tel que montré dans l'exemple de code suivant :

```
public String obtenerCSV(String aplicacion, String tipoServicio) {
   String result=null;
   DateFormat dateFormat = new SimpleDateFormat("yyyyMMddHHmmssSSS");
   result = dateFormat.format(Calendar.getInstance().getTime()) +
Integer.toString((int)(Math.random()*9999));
   return result;
   }
}
```

- **Connecteur avec le gestionnaire documentaire.** Ce connecteur est chargé et invoqué par l'application e-Apostille en vue de stocker des documents dans le système, et par la suite pour les récupérer. Il est utilisé tant pour stocker des documents de façon temporaire (documents publics électroniques ou numérisés allant être apostillés, apostille électronique) que pour le stockage à long terme (version imprimable ou " couverture " de l'apostille). Ce connecteur est également utilisé par l'application e-Register pour récupérer ladite version imprimable et l'afficher. L'implémentation actuelle stocke les documents dans une colonne BLOB d'une table Oracle, mais le modèle permettrait de réaliser une implémentation archivant les documents dans un dépôt informatique d'un gestionnaire documentaire quelconque. La seule condition requise est que le gestionnaire documentaire attribue à chaque document un identifiant ou localisateur unique pouvant être stocké dans un string à 50 caractères ; le gestionnaire documentaire devra rendre ce localisateur lors de l'opération d'archivage du document, puis permettre par la suite sa récupération en utilisant comme paramètre uniquement ce localisateur.

  L'interface que doit présenter ce connecteur est la suivante :

```
public interface IConectorGestorDocumental extends IConector {
       public String altaDocumento(InputStream inFile, String descripcion,
                     String extension) throws GenericException;
       public boolean eliminarDocumento(String uid) throws GenericException;
       public DatosDocumento leerDocumento(String uid) throws GenericException;
}
```

C'est-à-dire que le connecteur doit être une classe Java implémentant les méthodes **altaDocumento**, **eliminarDocumento** et **leerDocumento** qui recevront comme argument :

inFile : stream avec le contenu binaire du fichier à archiver

descripcion : description textuelle du fichier

extension : extension du nom de fichier (pdf, doc, txt, etc.)

uid : identifiant unique du fichier dans le dépôt informatique documentaire

La méthode **altaDocumento** doit retourner l'identifiant unique du document, celui-ci une fois archivé dans le dépôt informatique documentaire.

La méthode **eliminarDocumento** doit retourner **true** si elle a réussi à supprimer correctement le document ou à lever une exception dans le cas contraire.

La méthode **leerDocumento** doit retourner un objet de classe **DatosDocumento** contenant toute l'information sur le document, y compris le contenu binaire de celui-ci. La définition de cette classe peut être consultée à l'Annexe I du présent document.

Le ministère de la Justice de l'Espagne peut procurer aux organismes intéressés le connecteur de gestionnaire documentaire qui archive les documents dans des champs BLOB de base de données. Dans le cas précis du ministère de la Justice, une base de données Oracle est employée dans ce but, mais ce même connecteur pourrait fonctionner pour d'autres moteurs de base de données, tels que MySQL. Les adaptations à réaliser sur le connecteur seraient les mêmes qui sont décrites ci-après au point 3.3. " Changement de gestionnaire de base de données ".

- **Connecteur avec plate-forme de signature électronique.** Ce connecteur est chargé et invoqué par les applications e-Apostille et e-Register à chaque fois qu'il est nécessaire de vérifier une signature électronique. Dans le cas d'e-Apostille, il est utilisé pour vérifier la signature des documents publics électroniques et pour vérifier la signature de l'e-apostille elle-même celle-ci une fois réalisée. Dans le cas d'e-Register, il est utilisé pour l'option de " vérification de signature d'e-apostilles ". Il est également utilisé par le module de gestion des signatures afin d'extraire les références (émetteur et nº de série) des certificats électroniques au format DER qui sont chargés associés à des signataires. Par conséquent, ce connecteur peut être remplacé par un autre utilisant une plate-forme de services de signature différente, pourvu que cette plate-forme assure les services suivants :
  - Vérification de signature électronique d'un document
  - Extraction de données (concrètement émetteur et nº de série) d'un certificat fourni au format DER

  Il est prévu qu'à l'avenir le système d'e-apostille permette de signer avec les dénommés " signatures d'entité " ; celui-ci est un type de signature réalisé sur un serveur ; par conséquent, lorsque ce modèle de signature sera implémenté, la plate-forme de services de signature sera également employée pour signer les apostilles. À ce moment-là, le connecteur sera adapté pour habiliter cette fonction. L'interface que doit présenter à cette date ce connecteur est la suivante :

```
public interface IConectorPlataformaFirma extends IConector {
       public ValidacionResponse validarCertificado(InputStream certificadoDER,
                     String datosAdicionales) throws GenericException;

       public DatosValidarDocumentoFirmado[] validarDocumentoFirmado(
                     InputStream contenido, String datosAdicionales)
                     throws GenericException;
}
```

  C'est-à-dire que le connecteur doit être une classe Java implémentant les méthodes **validarCertificado** et **validarDocumentoFirmado**. Ces méthodes reçoivent les arguments suivants :

  certificadoDER : Contenu binaire de l'exportation de la partie publique d'un certificat numérique au format DER

  datosAdicionales : String avec des valeurs possibles "XADES", "CADES" ou "PADES" qui indique le type de document à valider. Dans la méthode **validarCertificado** réellement cet argument n'est pas utilisé.

  contenido : Contenu binaire du document signé à valider.

La méthode **validarCertificado** rend une instance de classe **ValidacionResponse** avec information au sujet du certificat fourni comme argument. La méthode **validarDocumentoFirmado** rend une array d'objets de classe **DatosValidarDocumentoFirmado** avec information au sujet de la/des signature(s) du document. La définition de ces deux classes peut être consultée à l'Annexe I du présent document.

3.2. Remplacement du mécanisme de signature côté client

L'application e-Apostille utilise actuellement un *applet* Java dénommé *WebSigner*, de TB Solutions, pour réaliser la signature des apostilles depuis le client (navigateur). Une interface *javascript* est employée pour l'interaction avec cet *applet*. Il serait possible de substituer à cet *applet* une autre solution de signature côté client, bien que dans ce cas ceci ne se borne pas à remplacer un connecteur, mais il faudrait modifier le code *javascript* utilisé pour avoir accès à l'*applet*. Du moment que la solution de signature retenue aura une interface de programmation *javascript* similaire à celle proposée par WebSigner, il est possible de remplacer ce composant, or ce changement est plus " intrusif " puisqu'il implique la modification du code javascript de l'application web e-Apostille elle-même.

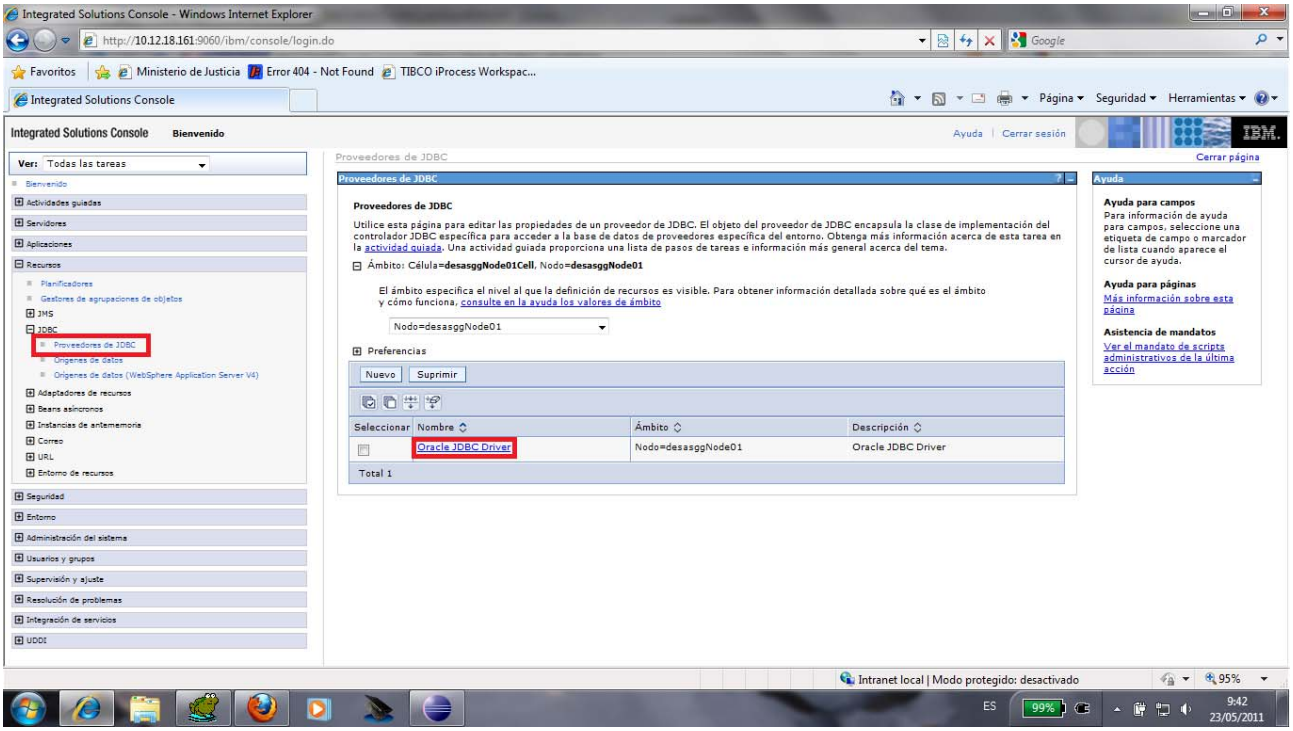
Lorsque, à l'avenir, il existera en outre la possibilité de la signature serveur d'apostilles à travers la plate-forme de services signature, l'opération de signature pourra être réalisée à travers le connecteur avec cette plate-forme. Dans ce cas, la substitution du connecteur avec la plate-forme de signature permettra l'emploi de différentes plateformes de signature pour les opérations de signature numérique d'apostille sur serveur.

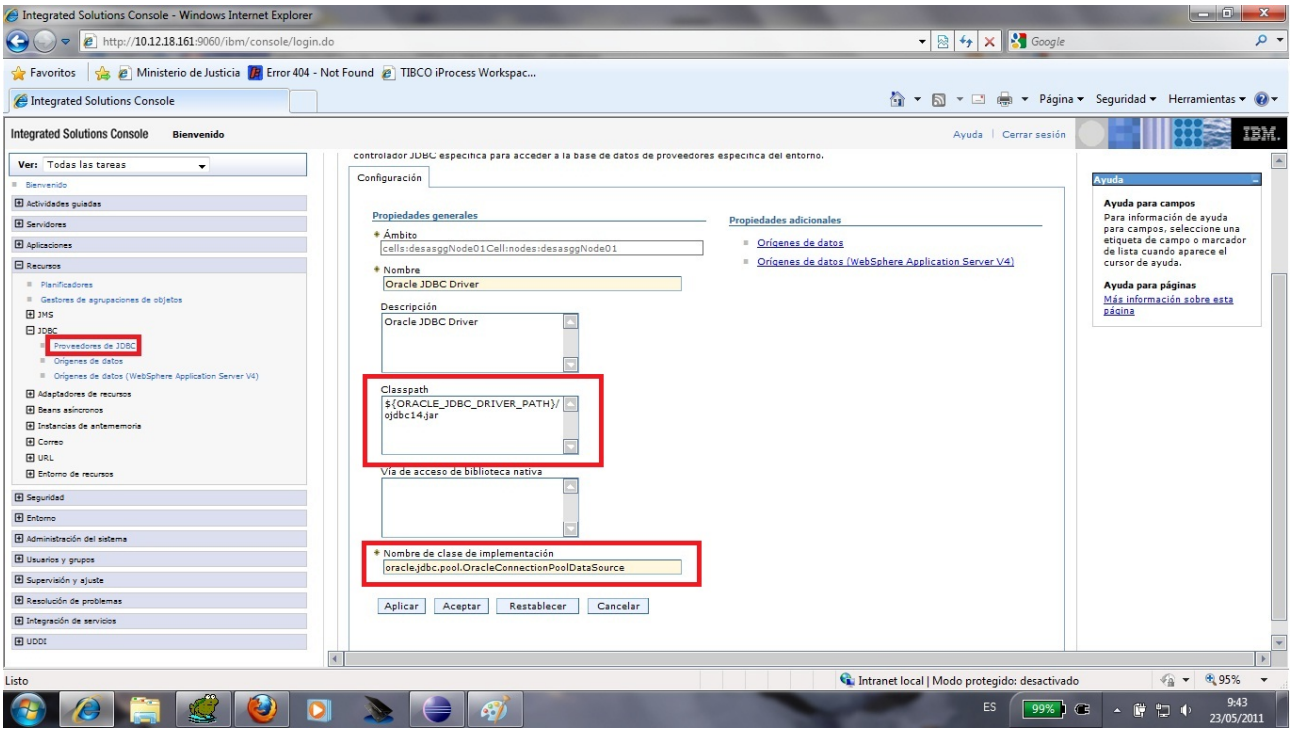
3.3. Changement de gestionnaire de base de données

Tous les modules qui composent le système d'e-APP utilisent le Framework de mappage objet-relationnel Hibernate. Cette couche de *software* isole les applications des spécificités du moteur de base de données relationnel utilisé. Par conséquent, on pourrait utiliser tout autre gestionnaire de base de données supporté par Hibernate ; les seuls changements à réaliser seraient :

- Configurer un *driver JDBC* et créer un *datasource* visant la base de données à utiliser
- Établir le *nom JNDI* du *datasource* dans les fichiers de propriétés
- Modifier dans les fichiers de propriétés le " dialecte " de SQL correspondant au gestionnaire de base de données allant être utilisé

La configuration du driver JDBC est réalisée dans l'application de configuration de WebSphere Application Server :





Dans le cas, par exemple, de **MySQL**, il faudrait introduire dans *Classpath* le chemin au module jar qui contient les librairies de MySQL : **`.../mysql.jar`** et dans *Nombre de clase de implementación* (Nom de classe d'implémentation) : **`com.mysql.jdbc.jdbc2.optional.MysqlConnectionPoolDataSource`**

Ensuite on doit créer un *datasource* pour ce *driver JDBC* et lui attribuer un nom JNDI (*nombre JNDI)*.

Tant ce nom JNDI que le " dialecte " SQL doivent être configurés dans les fichiers **applicationContext-configuration.xml** qui se trouvent dans le chemin relatif : **services/src/main/resources/common/hibernate** à l'intérieur de chaque projet parmi ceux qui composent le système d'apostille électronique. Par exemple, pour l'application e-Apostille, il faudrait modifier le fichier :
/eApp/eApp-services/src/main/resources/common/hibernate/applicationContext-configuration.xml

Les entrées à modifier sont celles marquées en jaune :

```
<bean id="sessionFactory" class="org.springframework.orm.hibernate3.LocalSessionFactoryBean">
   <property name="dataSource" ref="dataSource" />
   <property name="mappingResources">
      <ref bean="mappingResources" />
   </property>
   <property name="hibernateProperties">
      <props>
         <prop key="hibernate.hbm2ddl.auto">none</prop>
         <prop key="hibernate.dialect">org.hibernate.dialect.Oracle10gDialect</prop>
         <prop key="hibernate.show_sql">false</prop>
      </props>
   </property>
</bean>
<bean id="dataSource" class="org.springframework.jndi.JndiObjectFactoryBean">
   <property name="resourceRef"><value>false</value></property>
   <property name="jndiName"><value>jdbc/eapp</value></property>
</bean>
```

Le *nombre JNDI* (nom JNDI) de l'exemple (`jdbc/eapp`) doit être remplacé par celui qui aura été attribué au *datasource JDBC* allant être utilisé.
En ce qui concerne le dialecte, pour utiliser, par exemple, **MySQL 5.x**, il faudrait utiliser l'entrée suivante :
`<prop key="hibernate.dialect">org.hibernate.dialect.MySQL5Dialect</prop>`

La liste de dialectes supportés peut être consultée dans la documentation d'Hibernate :
http://www.163jsp.com/help/hibernate32api/org/hibernate/dialect/Dialect.html

3.4. Intégration à travers Web Services

La couche d'intégration du système e-APP permet l'intégration d'applications externes avec des modules complets du système. Concrètement, est permise l'intégration avec le module e-Apostille et avec le module e-Register, tel que décrit ci-après.

3.4.1. Intégration avec e-Apostille

Le Service Web d'intégration avec e-Apostille publie des méthodes qui permettent à une application externe de :

- Créer une nouvelle demande d'e-apostille, en recommençant son flux de réalisation de démarches (méthode **`solicitarApostilla`**)
- Connaître l'état de réalisation du traitement d'une e-apostille, et plus précisément, savoir si elle a déjà été émise et signée (méthode **`consultarApostilla`**)
- Récupérer une e-apostille déjà signée (méthode **`obtenerApostilla`**)
- Annuler une demande d'apostille (préalablement réalisée à travers Web Service), si celle-ci n'a pas encore été traitée (méthode **`anularApostilla`**)

À travers ce service, une application externe générant des documents publics électroniques pourrait proposer à ses utilisateurs l'option de " marquer " sur son interface d'utilisateur s'ils souhaitent que l'acte public demandé soit apostillé par la même occasion. Dans ce cas, l'application, après avoir généré et signé l'acte public, créerait une demande d'apostille électronique sur l'acte public électronique, qui serait traitée par le flux normal de l'application e-Apostille. L'application externe pourrait, régulièrement, demander au système d'e-APP si sa demande a été complètement traitée et, dans l'affirmative, récupérer l'apostille électronique et la servir à l'utilisateur final.

À l'Annexe II du présent document vous trouverez la définition en WSDL de ce service.

3.4.2. Intégration avec e-Register

Le Service Web d'intégration avec e-Apostille publie une méthode qui permet à une application externe générant des apostilles électroniques de les inclure dans e-Register. Pour cela, l'application externe devra fournir au moins :

- Les données d'identification de l'apostille électronique : numéro, date d'émission et code de vérification
- Autres données minimum exigées par la Convention de la Haye pour un registre d'apostilles :
    - nom du signataire de l'acte public
    - qualité en laquelle il a agi
    - autorité ayant apposé le sceau ou timbre
- Un fichier PDF avec la " version imprimable " de l'apostille électronique.

Si l'on souhaite que la fonction de " vérifier hash d'acte public " de e-Register soit disponible, il faudra fournir de façon supplémentaire la méthode d'intégration :

- Le hash de l'acte public, généré suivant l'un des algorithmes supportés
- Le nom d'identification de l'algorithme employé

Les algorithmes de *hash* supportés sont :

- SHA-1
- MD5

Nous devons noter que pour que l'option de e-Register de " vérifier signature d'e-Apostille " soit opérationnelle, il faut disposer d'une plate-forme de services et de son connecteur correspondant.

La définition de ce service en WSDL ne se trouve pas encore disponible à la date du présent document, mais elle pourra être fournie aux personnes intéressées dès qu'elle sera disponible.

3.5. Adaptation des tables maîtres

Outre les adaptations de software pertinentes, pour exporter le système d'e-APP du ministère de la Justice de l'Espagne à d'autres organismes, il serait nécessaire d'adapter le contenu de certaines tables " maîtres " de données qui contiennent de l'information de configuration du système. Ce sont les suivantes :

- Table des Sièges (TC_SEDE). Elle doit être remplie avec les données des différents sièges des autorités émettrices d'apostilles dont la compétence revient à l'organisme allant implanter la solution. Sa définition peut être consultée à l'Annexe III du présent document.

3.6. Localisation

Dans le cas où le système serait implanté dans un pays non hispanophone, tant les textes statiques des formulaires que les contenus des tables " maîtres " devraient être traduits dans la langue du pays d'implantation (processus de " localisation ").

Pour la localisation des textes statiques il faudrait modifier les fichiers de propriétés (fichiers de texte) dans lesquels ils résident et qui se déploient avec l'application dans le serveur WAS. Ces fichiers se trouvent dans le chemin relatif `src/main/resources/locale` de chaque module qui compose le système d'e-APP. Par exemple, pour le module e-Apostille, le fichier de propriétés de chaque langue se trouverait sur le chemin : `eApp/eApp-webapp/src/main/resources/locale`

Les tables maîtres qu'il faudrait adapter seraient :

- Table des pays (TC_PAIS)
- Table des états (TC_ESTADO)

NOTE : Les tables des " Provinces " (TC_PROVINCIA) et des " Communes " (TC_MUNICIPIO) ne s'emploient que si le pays sélectionné pour le domicile est " l'Espagne " ; par conséquent, pour l'implantation dans d'autres pays, il ne faudrait pas adapter ces tables ni les alimenter avec les informations du territoire propres au pays d'implantation de la solution. La ville et/ou la province ou similaire à laquelle appartient le domicile pourraient être saisies en les tapant manuellement dans ce cas-là.
Alternativement, si le modèle territorial du pays d'implantation de la solution est similaire à celui de l'Espagne (division en Provinces et Communes, ou bien des concepts similaires), un petit réglage pourrait être effectué dans le code de l'application afin qu'il envisage ces champs lors de la sélection du pays en question. Dans ce cas, les tables des Provinces et des Communes devraient être remplies avec celles du pays en question.
La définition de toutes ces tables peut être consultée à l'Annexe III du présent document.

ANNEXE I. DÉFINITION DES CLASSES

Classe DatosDocumento

```
public class DatosDocumento implements java.io.Serializable {
	private static final long serialVersionUID = -604263192269590408L;
	private String doUidDocumento;
	private String doDescripcion;
	private OutputStream doArchivo;
	private String doExtension;

	public DatosDocumento() {
	}

	public String getDoUidDocumento() {
		return this.doUidDocumento;
	}

	public void setDoUidDocumento(String doUidDocumento) {
		this.doUidDocumento = doUidDocumento;
	}

	public String getDoDescripcion() {
		return this.doDescripcion;
	}

	public void setDoDescripcion(String doDescripcion) {
		this.doDescripcion = doDescripcion;
	}

	public String getDoExtension() {
		return this.doExtension;
	}

	public void setDoExtension(String doExtension) {
		this.doExtension = doExtension;
	}

	public OutputStream getDoArchivo() {
		return doArchivo;
	}

	public void setDoArchivo(OutputStream doArchivo) {
		this.doArchivo = doArchivo;
	}
}
```

Classe ValidacionResponse

```
public class ValidacionResponse implements IDescribeError {
	private static final long serialVersionUID = 3567193432574376046L;
	private String codError;
	private String descError;
	private String issuer;
	private String subject;
	private String serialNumber;
	private Calendar validFrom;
	private Calendar validUntil;
	private int certState;

	public String getIssuer() {
		return issuer;
	}

	public void setIssuer(String issuer) {
		this.issuer = issuer;
	}

	public String getSerialNumber() {
		return serialNumber;
	}

	public void setSerialNumber(String serialNumber) {
		this.serialNumber = serialNumber;
	}

	public Calendar getValidFrom() {
		return validFrom;
	}

	public void setValidFrom(Calendar validFrom) {
		this.validFrom = validFrom;
	}

	public Calendar getValidUntil() {
		return validUntil;
	}

	public void setValidUntil(Calendar validUntil) {
		this.validUntil = validUntil;
	}

	public int getCertState() {
		return certState;
	}

	public void setCertState(int certState) {
		this.certState = certState;
	}
```

```
	public String getCodError() {
		return this.codError;
	}

	public String getDescError() {
		return this.descError;
	}

	public void setCodError(String codigo) {
		this.codError = codigo;
	}

	public void setDescError(String descripcion) {
		this.descError = descripcion;
	}

	public void setSubject(String subject) {
		this.subject = subject;
	}

	public String getSubject() {
		return subject;
	}
}
```

Classe DatosValidarDocumentoFirmado

```
public class DatosValidarDocumentoFirmado implements IDescribeError {
	private static final long serialVersionUID = 3567193432574376046L;
	private String codError;
	private String descError;
	private String issuer;
	private String subject;
	private Calendar validFrom;
	private Calendar validUntil;
	private Calendar signDate;
	private String serialNumber;
	private int resultCode;

	public String getIssuer() {
		return issuer;
	}

	public void setIssuer(String issuer) {
		this.issuer = issuer;
	}

	public Calendar getValidFrom() {
		return validFrom;
	}

	public void setValidFrom(Calendar validFrom) {
		this.validFrom = validFrom;
	}

	public Calendar getValidUntil() {
		return validUntil;
	}

	public void setValidUntil(Calendar validUntil) {
		this.validUntil = validUntil;
	}

	public String getCodError() {
		return this.codError;
	}

	public String getDescError() {
		return this.descError;
	}

	public void setCodError(String codigo) {
		this.codError = codigo;
	}

	public void setDescError(String descripcion) {
		this.descError = descripcion;
	}

	public void setSubject(String subject) {
		this.subject = subject;
```

```
	}

	public String getSubject() {
		return subject;
	}

	public void setSignDate(Calendar signDate) {
		this.signDate = signDate;
	}

	public Calendar getSignDate() {
		return signDate;
	}

	public void setSerialNumber(String serialNumber) {
		this.serialNumber = serialNumber;
	}

	public String getSerialNumber() {
		return serialNumber;
	}

	public void setResultCode(int resultCode) {
		this.resultCode = resultCode;
	}

	public int getResultCode() {
		return resultCode;
	}
}
```

ANNEXE II. DÉFINITIONS EN WSDL DES SERVICES WEB

WSDL du service d'intégration avec e-Apostille

```
<?xml version="1.0" encoding="UTF-8"?>
<wsdl:definitions targetNamespace="http://impl.ws.eApostille.eApp.apostilla.majus.ntj.mju"
xmlns:apachesoap="http://xml.apache.org/xml-soap"
xmlns:impl="http://impl.ws.eApostille.eApp.apostilla.majus.ntj.mju"
xmlns:intf="http://impl.ws.eApostille.eApp.apostilla.majus.ntj.mju"
xmlns:tns1="http://utils.ws.eApostille.eApp.apostilla.majus.ntj.mju"
xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
xmlns:wsdlsoap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
<!--WSDL created by Apache Axis version: 1.4
Built on Apr 22, 2006 (06:55:48 PDT)-->
 <wsdl:types>
  <schema elementFormDefault="qualified"
targetNamespace="http://utils.ws.eApostille.eApp.apostilla.majus.ntj.mju"
xmlns="http://www.w3.org/2001/XMLSchema">
   <complexType name="InfConsulApostilla">
    <sequence>
     <element name="csv" nillable="true" type="xsd:string"/>
     <element name="numApostilla" nillable="true" type="xsd:string"/>
    </sequence>
   </complexType>
   <complexType name="DatosConsulApostilla">
    <sequence>
     <element name="autoridadApostillante" nillable="true" type="xsd:string"/>
     <element name="autoridadFirmante" nillable="true" type="xsd:string"/>
     <element name="calidadFirmante" nillable="true" type="xsd:string"/>
     <element name="codError" nillable="true" type="xsd:string"/>
     <element name="comparecenciaElectronica" type="xsd:boolean"/>
     <element name="cp" nillable="true" type="xsd:string"/>
     <element name="csv" nillable="true" type="xsd:string"/>
     <element name="descError" nillable="true" type="xsd:string"/>
     <element name="direccionSolicitante" nillable="true" type="xsd:string"/>
     <element name="emailSolicitante" nillable="true" type="xsd:string"/>
     <element name="estado" nillable="true" type="xsd:string"/>
     <element name="fechaAnulacion" nillable="true" type="xsd:dateTime"/>
     <element name="fechaCreacion" nillable="true" type="xsd:dateTime"/>
     <element name="fechaDescarga" nillable="true" type="xsd:dateTime"/>
     <element name="fechaEmision" nillable="true" type="xsd:dateTime"/>
     <element name="fechaFirma" nillable="true" type="xsd:dateTime"/>
     <element name="fechaNotificacion" nillable="true" type="xsd:dateTime"/>
     <element name="fechaValidacion" nillable="true" type="xsd:dateTime"/>
     <element name="fechaVigencia" nillable="true" type="xsd:dateTime"/>
     <element name="identificadorSolicitante" nillable="true" type="xsd:string"/>
     <element name="localizador" nillable="true" type="xsd:string"/>
     <element name="movRechazo" nillable="true" type="xsd:string"/>
     <element name="municipioSolicitante" nillable="true" type="xsd:string"/>
     <element name="nombreSolicitante" nillable="true" type="xsd:string"/>
     <element name="numApostilla" nillable="true" type="xsd:string"/>
     <element name="numRegEntrada" nillable="true" type="xsd:string"/>
     <element name="numRegSalida" nillable="true" type="xsd:string"/>
     <element name="organismo" nillable="true" type="xsd:string"/>
     <element name="paisDestino" type="xsd:int"/>
```

```
  <element name="paisDestinoDesc" nillable="true" type="xsd:string"/>
  <element name="paisSolicitante" type="xsd:int"/>
  <element name="paisSolicitanteDesc" nillable="true" type="xsd:string"/>
  <element name="provinciaSolicitante" type="xsd:int"/>
  <element name="provinciaSolicitanteDesc" nillable="true" type="xsd:string"/>
  <element name="telefonoSolicitante" nillable="true" type="xsd:string"/>
  <element name="tipoDocumento" type="xsd:boolean"/>
  <element name="url" nillable="true" type="xsd:string"/>
 </sequence>
</complexType>
<complexType name="InfSolApostilla">
 <sequence>
  <element name="DNI_solicitante" nillable="true" type="xsd:string"/>
  <element name="apAutoridadApostillante" nillable="true" type="xsd:string"/>
  <element name="apAutoridadFirmante" nillable="true" type="xsd:string"/>
  <element name="apCalidadFirmante" nillable="true" type="xsd:string"/>
  <element name="apComparecenciaElect" type="xsd:boolean"/>
  <element name="apCp" nillable="true" type="xsd:string"/>
  <element name="apDireccion" nillable="true" type="xsd:string"/>
  <element name="apEmail" nillable="true" type="xsd:string"/>
  <element name="apLocalizadorDpElect" nillable="true" type="xsd:string"/>
  <element name="apMunicipio" nillable="true" type="xsd:string"/>
  <element name="apOrganismo" nillable="true" type="xsd:string"/>
  <element name="apTelefono" nillable="true" type="xsd:string"/>
  <element name="apTipoFormatoDp" type="xsd:int"/>
  <element name="apUrlDpElect" nillable="true" type="xsd:string"/>
  <element name="apellidos_solicitante" nillable="true" type="xsd:string"/>
  <element name="descripcionDP" nillable="true" type="xsd:string"/>
  <element name="docPublico" nillable="true" type="xsd:base64Binary"/>
  <element name="extensionDP" nillable="true" type="xsd:string"/>
  <element name="idPais" type="xsd:int"/>
  <element name="idPaisDest" type="xsd:int"/>
  <element name="idProv" type="xsd:int"/>
  <element name="idSede" type="xsd:long"/>
  <element name="nombre_solicitante" nillable="true" type="xsd:string"/>
  <element name="usr_solicitante" nillable="true" type="xsd:string"/>
 </sequence>
</complexType>
<complexType name="DatosSolApostilla">
 <sequence>
  <element name="DNI_solicitante" nillable="true" type="xsd:string"/>
  <element name="apAutoridadApostillante" nillable="true" type="xsd:string"/>
  <element name="apAutoridadFirmante" nillable="true" type="xsd:string"/>
  <element name="apCalidadFirmante" nillable="true" type="xsd:string"/>
  <element name="apComparecenciaElect" type="xsd:boolean"/>
  <element name="apCp" nillable="true" type="xsd:string"/>
  <element name="apCsv" nillable="true" type="xsd:string"/>
  <element name="apDescripcionDp" nillable="true" type="xsd:string"/>
  <element name="apDireccion" nillable="true" type="xsd:string"/>
  <element name="apEmail" nillable="true" type="xsd:string"/>
  <element name="apHash" nillable="true" type="xsd:string"/>
  <element name="apLocalizadorDpElect" nillable="true" type="xsd:string"/>
  <element name="apMotivoRechazo" nillable="true" type="xsd:string"/>
  <element name="apMunicipio" type="xsd:long"/>
  <element name="apNumRegEntrada" nillable="true" type="xsd:string"/>
  <element name="apNumRegSalida" nillable="true" type="xsd:string"/>
  <element name="apNumeroApostilla" nillable="true" type="xsd:string"/>
```

```
      <element name="apOrganismo" nillable="true" type="xsd:string"/>
      <element name="apTelefono" nillable="true" type="xsd:string"/>
      <element name="apTipoFormatoDp" type="xsd:int"/>
      <element name="apUrlDpElect" nillable="true" type="xsd:string"/>
      <element name="apellidos_solicitante" nillable="true" type="xsd:string"/>
      <element name="codError" nillable="true" type="xsd:string"/>
      <element name="descError" nillable="true" type="xsd:string"/>
      <element name="descPais" nillable="true" type="xsd:string"/>
      <element name="descPaisDest" nillable="true" type="xsd:string"/>
      <element name="descProv" nillable="true" type="xsd:string"/>
      <element name="docPublico" nillable="true" type="xsd:base64Binary"/>
      <element name="extensionDP" nillable="true" type="xsd:string"/>
      <element name="idPais" type="xsd:int"/>
      <element name="idPaisDest" nillable="true" type="xsd:string"/>
      <element name="idProv" type="xsd:int"/>
      <element name="idSede" type="xsd:int"/>
      <element name="nombre_solicitante" nillable="true" type="xsd:string"/>
     </sequence>
    </complexType>
    <complexType name="InfAnulApostilla">
     <sequence>
      <element name="csv" nillable="true" type="xsd:string"/>
      <element name="numApostilla" nillable="true" type="xsd:string"/>
     </sequence>
    </complexType>
    <complexType name="DatosAnulApostilla">
     <sequence>
      <element name="codError" nillable="true" type="xsd:string"/>
      <element name="descError" nillable="true" type="xsd:string"/>
      <element name="resultadoAnulacion" type="xsd:boolean"/>
     </sequence>
    </complexType>
    <complexType name="InfObtenerApostilla">
     <sequence>
      <element name="csv" nillable="true" type="xsd:string"/>
      <element name="numApostilla" nillable="true" type="xsd:string"/>
     </sequence>
    </complexType>
    <complexType name="DatosObtenerApostilla">
     <sequence>
      <element name="codError" nillable="true" type="xsd:string"/>
      <element name="descError" nillable="true" type="xsd:string"/>
      <element name="fichero" nillable="true" type="xsd:string"/>
     </sequence>
    </complexType>
  </schema>
  <schema elementFormDefault="qualified"
targetNamespace="http://impl.ws.eApostille.eApp.apostilla.majus.ntj.mju"
xmlns="http://www.w3.org/2001/XMLSchema">
    <import namespace="http://utils.ws.eApostille.eApp.apostilla.majus.ntj.mju"/>
    <element name="consulta" type="tns1:InfConsulApostilla"/>
    <element name="consultarApostillaReturn" type="tns1:DatosConsulApostilla"/>
    <element name="solicitud" type="tns1:InfSolApostilla"/>
    <element name="solicitarApostillaReturn" type="tns1:DatosSolApostilla"/>
    <element name="anulacion" type="tns1:InfAnulApostilla"/>
    <element name="anularApostillaReturn" type="tns1:DatosAnulApostilla"/>
    <element name="apostilla" type="tns1:InfObtenerApostilla"/>
```

```
  <element name="obtenerApostillaReturn" type="tns1:DatosObtenerApostilla"/>
 </schema>
</wsdl:types>

  <wsdl:message name="anularApostillaResponse">
     <wsdl:part element="impl:anularApostillaReturn" name="anularApostillaReturn">
     </wsdl:part>
  </wsdl:message>

  <wsdl:message name="anularApostillaRequest">
     <wsdl:part element="impl:anulacion" name="anulacion">
     </wsdl:part>
  </wsdl:message>

  <wsdl:message name="solicitarApostillaRequest">
     <wsdl:part element="impl:solicitud" name="solicitud">
     </wsdl:part>
  </wsdl:message>

  <wsdl:message name="consultarApostillaRequest">
     <wsdl:part element="impl:consulta" name="consulta">
     </wsdl:part>
  </wsdl:message>

  <wsdl:message name="solicitarApostillaResponse">
     <wsdl:part element="impl:solicitarApostillaReturn" name="solicitarApostillaReturn">
     </wsdl:part>
  </wsdl:message>

  <wsdl:message name="consultarApostillaResponse">
     <wsdl:part element="impl:consultarApostillaReturn" name="consultarApostillaReturn">
     </wsdl:part>
  </wsdl:message>

  <wsdl:message name="obtenerApostillaRequest">
     <wsdl:part element="impl:apostilla" name="apostilla">
     </wsdl:part>
  </wsdl:message>

  <wsdl:message name="obtenerApostillaResponse">
     <wsdl:part element="impl:obtenerApostillaReturn" name="obtenerApostillaReturn">
     </wsdl:part>
  </wsdl:message>
```

```
   <wsdl:portType name="ServicioSolicitarApostillas">
      <wsdl:operation name="consultarApostilla" parameterOrder="consulta">
         <wsdl:input message="impl:consultarApostillaRequest"
name="consultarApostillaRequest">
       </wsdl:input>

         <wsdl:output message="impl:consultarApostillaResponse"
name="consultarApostillaResponse">
       </wsdl:output>
      </wsdl:operation>
      <wsdl:operation name="solicitarApostilla" parameterOrder="solicitud">

         <wsdl:input message="impl:solicitarApostillaRequest"
name="solicitarApostillaRequest">
       </wsdl:input>
         <wsdl:output message="impl:solicitarApostillaResponse"
name="solicitarApostillaResponse">
       </wsdl:output>
      </wsdl:operation>

      <wsdl:operation name="anularApostilla" parameterOrder="anulacion">
         <wsdl:input message="impl:anularApostillaRequest" name="anularApostillaRequest">
       </wsdl:input>
         <wsdl:output message="impl:anularApostillaResponse" name="anularApostillaResponse">
       </wsdl:output>
      </wsdl:operation>

      <wsdl:operation name="obtenerApostilla" parameterOrder="apostilla">
         <wsdl:input message="impl:obtenerApostillaRequest" name="obtenerApostillaRequest">
       </wsdl:input>
         <wsdl:output message="impl:obtenerApostillaResponse"
name="obtenerApostillaResponse">
       </wsdl:output>
      </wsdl:operation>

   </wsdl:portType>

   <wsdl:binding name="ServicioSolicitarApostillasSoapBinding"
type="impl:ServicioSolicitarApostillas">
      <wsdlsoap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http"/>
      <wsdl:operation name="consultarApostilla">
         <wsdlsoap:operation soapAction=""/>
         <wsdl:input name="consultarApostillaRequest">
            <wsdlsoap:body use="literal"/>
         </wsdl:input>
         <wsdl:output name="consultarApostillaResponse">
            <wsdlsoap:body use="literal"/>
         </wsdl:output>
```

```
      </wsdl:operation>
      <wsdl:operation name="solicitarApostilla">
         <wsdlsoap:operation soapAction=""/>
         <wsdl:input name="solicitarApostillaRequest">
            <wsdlsoap:body use="literal"/>
         </wsdl:input>
         <wsdl:output name="solicitarApostillaResponse">
            <wsdlsoap:body use="literal"/>
         </wsdl:output>
      </wsdl:operation>

      <wsdl:operation name="anularApostilla">
         <wsdlsoap:operation soapAction=""/>
         <wsdl:input name="anularApostillaRequest">
            <wsdlsoap:body use="literal"/>
         </wsdl:input>
         <wsdl:output name="anularApostillaResponse">
            <wsdlsoap:body use="literal"/>
         </wsdl:output>
      </wsdl:operation>

      <wsdl:operation name="obtenerApostilla">
         <wsdlsoap:operation soapAction=""/>
         <wsdl:input name="obtenerApostillaRequest">
            <wsdlsoap:body use="literal"/>
         </wsdl:input>
         <wsdl:output name="obtenerApostillaResponse">
            <wsdlsoap:body use="literal"/>
         </wsdl:output>
      </wsdl:operation>

   </wsdl:binding>

   <wsdl:service name="ServicioSolicitarApostillasService">
      <wsdl:port binding="impl:ServicioSolicitarApostillasSoapBinding"
name="ServicioSolicitarApostillas">
         <wsdlsoap:address
location="http://localhost:8080/ServicioSolicitarApostillasWS/services/ServicioSolicitarApost
illas"/>
      </wsdl:port>
   </wsdl:service>

</wsdl:definitions>
```

ANNEXE III. DÉFINITIONS DES TABLES RELATIONNELLES

Table des sièges

TC_SEDE				
Y sont stockées les données des "Sièges" des Autorités émettrices des apostilles				
Colonne	**Type Donnée**	**Description**	**Obligatoire**	**Valeurs**
SE_CLAVE	NUMBER (12)	Identifiant unique d'un Siège	OUI	
SE_NOMBRE	VARCHAR2 (50)	Code du Siège (obtenu de la page du test de compatibilité fourni par le ministère de la Justice)	OUI	
SE_DESCRIPCION	VARCHAR2 (150)	Description du Siège	NON	
SE_MUNICIPIO	VARCHAR2 (100)	Commune du Siège	OUI	
SE_TELEFONO	VARCHAR2 (35)	Téléphone du Siège	NON	
SE_FAX	VARCHAR2 (35)	Fax du Siège	NON	
SE_DIRECCION	VARCHAR2 (250)	Adresse postale du Siège	NON	
SE_CP	VARCHAR2 (5)	Code postal du Siège	NON	
SE_EMAIL	VARCHAR2 (150)	Adresse courriel du Siège	NON	

Table des états

TC_ESTADO				
Y sont stockés les différents états dans lesquels peut se trouver une demande d'apostille				
Colonne	**Type Donnée**	**Description**	**Obligatoire**	**Valeurs**
ES_CLAVE	NUMBER (2)	Identifiant unique d'un état d'apostille	OUI	Valeurs : - 1 : MISE EN ROUTE - 2: EN ATTENTE DE VALIDATION - 3: VALIDÉE - 4: ÉMISE - 5: NOTIFIÉE - 6: REFUSÉE - 7: ANNULÉE - 8: FIN TÉLÉ-CHARGEMENT
ES_NOMBRE	VARCHAR2 (50)	Nom de l'état	OUI	
ES_DESCRIPCION	VARCHAR2 (150)	Description de l'état de l'apostille	NON	

Table des pays

TC_PAIS				
Elle contient les données d'un pays, que l'application a besoin de connaître à un moment donné du flux d'apposition de l'apostille				
Colonne	**Type Donnée**	**Description**	**Obligatoire**	**Valeurs**
PA_CLAVE	NUMBER (3)	Identifiant unique d'un pays	OUI	
PA_NOMBRE	VARCHAR2 (100)	Nom du pays	OUI	

Table des provinces

TC_PROVINCIA				
Elle contient les données d'une province que l'application a besoin de connaître à un moment donné du flux d'apposition de l'apostille				
Colonne	**Type Donnée**	**Description**	**Obligatoire**	**Valeurs**
PR_CLAVE	NUMBER (2)	Identifiant unique d'une province	OUI	
PR_NOMBRE	VARCHAR2 (100)	Nom de la province	OUI	

Table des communes

TC_MUNICIPIO				
Elle contient les données d'une commune que l'application a besoin de connaître à un moment donné du flux d'apposition de l'apostille				
Colonne	**Type Donnée**	**Description**	**Obligatoire**	**Valeurs**
MU_CLAVE	NUMBER (4)	Identifiant unique d'une commune	OUI	
MU_NOMBRE	VARCHAR2 (100)	Nom de la commune	OUI	
MU_PR_CF_PROV	NUMBER (2)	Identifiant unique de la province à laquelle appartient la commune	OUI	