Facultad de Estudios Estadísticos
Universidad Complutense de Madrid

ntic master
revolucionamos la comunicación

VERSION FINAL 15-09-2020

Blockchain & BigData Canino

ANEXO III: Chaincodes y funcionalidades del sistema

Autores:

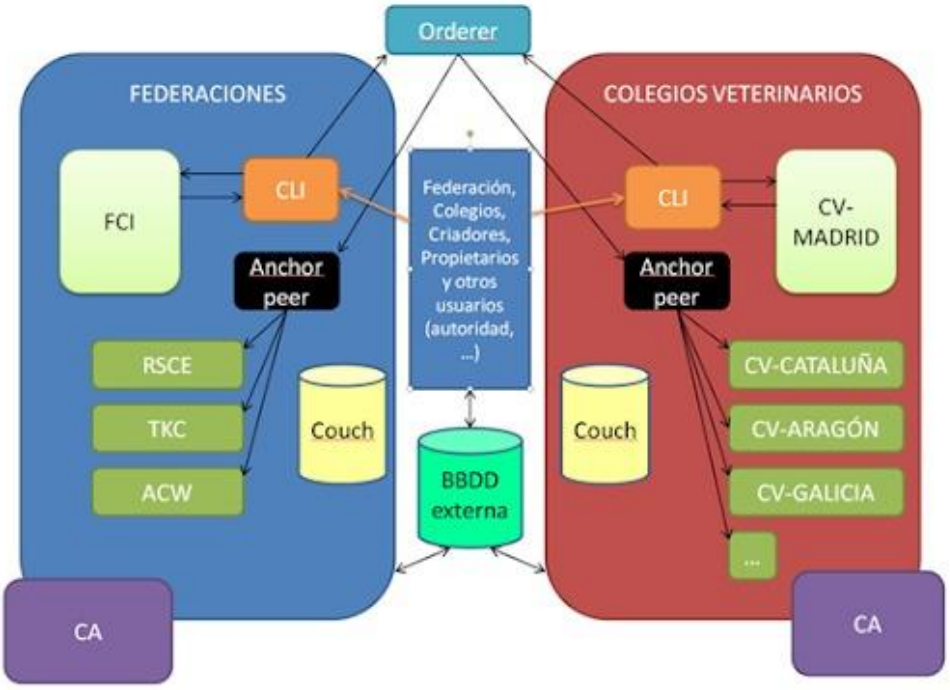
Cristina Rodríguez Chamorro
Daniel Lanzas Pellico
Helena García Fernández
José Bennani Pareja
Juan José Lucas de la Fuente
Unai Ares Icaran

Tutor:
Sergio Torres Palomino

Documentación del TFM
Máster Blockchain y Big Data. *Curso 2019-2020*

ÍNDICE

Contratos divididos por Organizaciones



Los contratos que utiliza cada organización son los siguientes

Federaciones Caninas

- Personas
- Perfil
- Perros
- Afijos
- Solicitudes
- Veterinarios
- Microchip
- Razas
- Exposiciones y concursos
- Títulos

Colegios de Veterinarios

- Personas
- Perfil
- Perros
- Veterinarios
- Vacunas
- Microchips
- Razas

Contratos inteligentes

Parámetros de entrada/salida de las funciones

Las funciones definidas en los contratos disponen de parameros de entrada específicos para su funcionamiento.

Como regla general dispones 2 argumentos:

```
args[0] : de tipo JSON con datos específicos de a función

args[1]: de tipo JSON con los datos de seguridad donde se identifica al usuario
que invoca la función.
```

Por el contrario, todas las funciones definidas en los contratos tendrán un parámetro de salida de tipo Response:

```
type Response struct {
    // A status code that should follow the HTTP status codes.
    Status int32 `protobuf:"varint,1,opt,name=status,proto3"
json:"status,omitempty"`
    // A message associated with the response code.
    Message string `protobuf:"bytes,2,opt,name=message,proto3"
json:"message,omitempty"`
    // A payload that can be used to include metadata with this response.
    Payload              []byte   `protobuf:"bytes,3,opt,name=payload,proto3"
json:"payload,omitempty"`
    XXX_NoUnkeyedLiteral struct{} `json:"-"`
    XXX_unrecognized     []byte   `json:"-"`
    XXX_sizecache        int32    `json:"-"`
}
```

Funciones comunes de los contratos

Existen un conjunto defunciones que se definen dentro de los contratos:

ejecutarConsulta

Permite realizar consultas al sistema a través del lenguaje de consulta de Mango, que se expresa como un objeto JSON que describe los filtros de los documentos de interés que se desean consultar.

asignarEstado

Asigna un valor de estado al sistema.

borrarEstado

Elimina un valor de estado del sistema.

consultarEstado

Consulta un valor de estado del sistema.

consultarRangoEstados

Consulta los valores de estado del sistema comprendidos entre dos valores.

getQueryResultForQueryString

Devuelve en el siguiente formato JSON el resultado de una consulta, donde el TipoEstado tendrá la definición especifica del estado consultado.

```
type TipoQueryEstado struct {
    Key    string             `json:"Key"`
    Record TipoEstado `json:"Record"`
}
```

PERSONAS

Este contrato inteligente es el encargado de gestionar los datos de las personas que intervienen en las solicitudes de los usuarios del sistema.

Actualmente gestiona los datos identificativos de:

- Criadores (propietarios de afijos)
- Propietarios de Perros
- Veterinarios
- Miembros de Federaciones

El presente contrato inteligente esta diseñado para admitir en el futuro ampliaciones de la blockchain, dando la posibilidad de albergar los datos de identificación de otro tipo de usuarios o datos asociados.

Por ejemplo, incorporar:

- Jueces de certámenes caninos
- Miembros de Asociaciones caninas
- Miembros de Club de raza caninas y de grupos
- Miembros de laboratorios (para registros de ADN)
- ...

Funciones que incorpora el contrato inteligente:

registrarPersona

Introduce en el sistema los datos necesarios para la identificación de una nueva persona.

- **Argumentos entrada:**

```
type registrarPersonas struct {
    Nombre                string `json:"Nombre"`
    Apellido1             string `json:"Apellido1"`
    Apellido2             string `json:"Apellido2"`
    TipoDocumento         string `json:"TipoDocumento"`
    IdentificadorDocumento string `json:"IdentificadorDocumento"`
    PaisEmisor            string `json:"PaisEmisor"`
    Certificado           string `json:"Certificado"`
}
```

El contrato realiza las siguientes validaciones:

- El número de argumentos de entrada sea 1
- El formato JSON de args[0] sea valido
- Los argumentos Nombre, Apellido1, Apellido2, TipoDocumento, IdentificadorDocumento y PaisEmisor tenga un valor.
- No exista otra persona registrada con eses mismo TipoDocumento, IdentificadorDocumento y PaisEmisor.

- **Argumentos salida:**

Devuelve el registro insertado:

```
type Personas struct {
        ObjectType            string `json:"docType"`
        IDPersona             int    `json:"IDPersona"`
        Nombre                string `json:"Nombre"`
        Apellido1             string `json:"Apellido1"`
        Apellido2             string `json:"Apellido2"`
        TipoDocumento         string `json:"TipoDocumento"`
        IdentificadorDocumento string `json:"IdentificadorDocumento"`
        PaisEmisor            string `json:"PaisEmisor"`
        Certificado           string `json:"Certificado"`
        FechaAlta             string `json:"FechaAlta"`
        FechaBaja             string `json:"FechaBaja"`
}
```

registrarDocumentoIdentidad

Introduce en el sistema los datos otros documentos de identidad de la persona, válidos para identificar a una persona registra.

Por ejemplo: las mujeres rumanas cuando se casa cambian el documento de identidad por el de su marido añadiendo un digito adicional o en España cuando un persona adquiere la nacionalidad se le cambia el NIE por el NIF.

modificarNombreApellidos

Modifica en el sistema el nombre y/o apellido de una persona registra.

Por ejemplo: en caso de cambio de nombre como Ignacio por Iñaki o cambio de orden de los apellidos.

cargarDatosIniciales

Registrar en el sistema los datos de las propietarios, veterinarios y personal de las federaciones definidos en un fichero de texto con el siguiente formato JSON:

```
type Personas struct {
        ObjectType             string `json:"docType"`
        IDPersona              int    `json:"IDPersona"`
        Nombre                 string `json:"Nombre"`
        Apellido1              string `json:"Apellido1"`
        Apellido2              string `json:"Apellido2"`
        TipoDocumento          string `json:"TipoDocumento"`
        IdentificadorDocumento string `json:"IdentificadorDocumento"`
        PaisEmisor             string `json:"PaisEmisor"`
        Certificado            string `json:"Certificado"`
        FechaAlta              string `json:"FechaAlta"`
        FechaBaja              string `json:"FechaBaja"`
}
```

- **Argumentos entrada:**

```
NombreFichero string
```

- **Argumentos salida:**

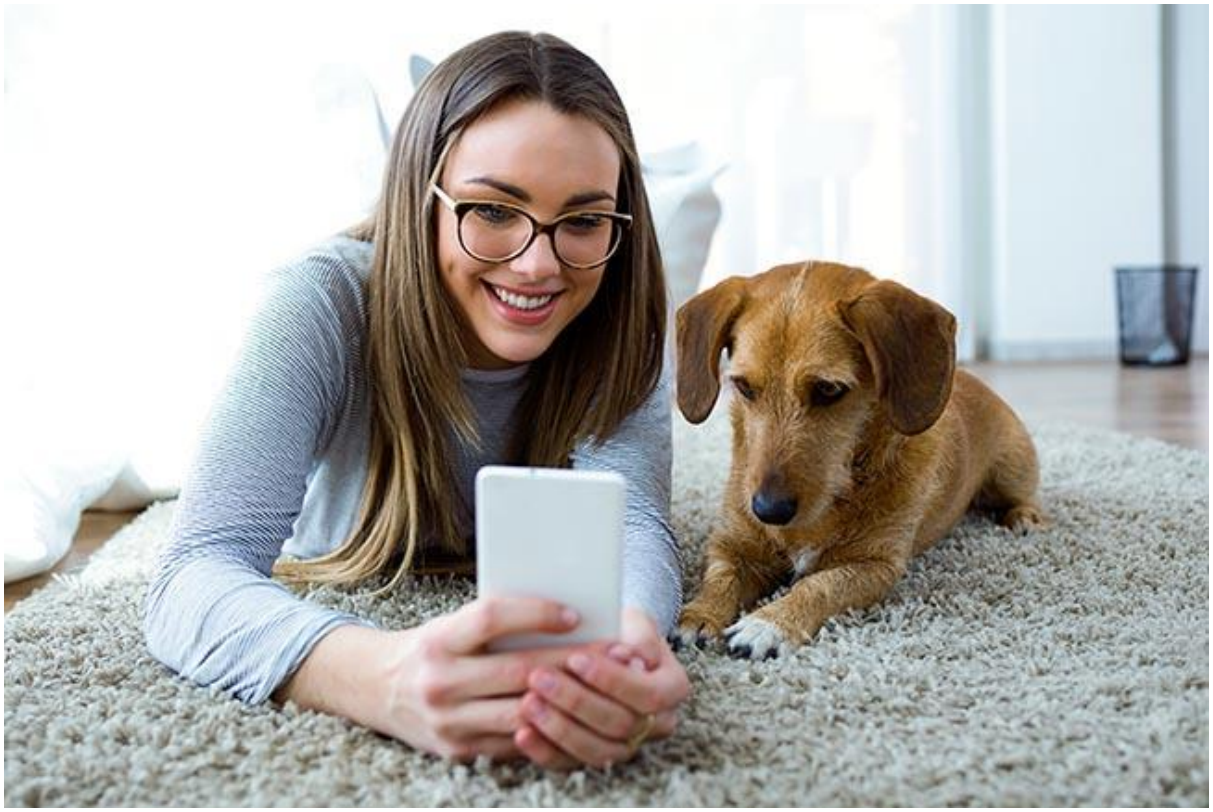
Devuelve el número de registros insertados:

```
NumeroRegistros string
```

Funciones comunes

Funciones de los contratos del sistema incorporadas a este contrato:

- **getQueryResultForQueryString**
- **asignarEstado**
- **borrarEstado**
- **consultarEstado**
- **consultarRangoEstados**
- **ejecutarConsulta**



Plan de pruebas

A continuación, se especifican una serie de comando que permite al lector realizar una batería de pruebas sobre el contrato inteligente

```
export CHANNEL_NAME=netcanchannel

export
CA_FILE=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/ordererOrganizat
ions/netcan.com/orderers/orderer.netcan.com/msp/tlscacerts/tlsca.netcan.com-
cert.pem

export ORDENER_URL=orderer.netcan.com:7050


# PERSONAS

peer chaincode invoke -n personas -c
'{"function":"registrarPersona","Args":["{\"Nombre\":\"Unai\",\"Apellido1\":\"Ares
\",\"Apellido2\":\"Icaran\",\"TipoDocumento\":\"NIF\",\"IdentificadorDocumento\":\
"30624315E\",\"PaisEmisor\":\"Spain\"}"]}' -o $ORDENER_URL --tls --cafile $CA_FILE
-C $CHANNEL_NAME


# ERRORES PERSONAS
```

```
peer chaincode invoke -n personas -c '{"function":"registrarPersona","Args":[""]}'
-o $ORDENER_URL --tls --cafile $CA_FILE -C $CHANNEL_NAME


peer chaincode invoke -n personas -c
'{"function":"registrarPersona","Args":["{Nombre\":\"Unai\",\"Apellido1\":\"Ares\"
,\"Apellido2\":\"Icaran\",\"TipoDocumento\":\"NIF\",\"IdentificadorDocumento\":\"3
0624315E\",\"PaisEmisor\":\"Spain\"}"]}' -o $ORDENER_URL --tls --cafile $CA_FILE -
C $CHANNEL_NAME

peer chaincode invoke -n personas -c
'{"function":"registrarPersona","Args":["{\"Nombre\":\"Unai\",\"Apellido1\":\"Ares
\",\"Apellido2\":\"Icaran\",\"TipoDocumento\":\"NIF\",\"IdentificadorDocumento\":\
"30624315E\",\"PaisEmisor\":\"Spain\"}"]}' -o $ORDENER_URL --tls --cafile $CA_FILE
-C $CHANNEL_NAME

peer chaincode invoke -n personas -c
'{"function":"registrarPersona","Args":["{\"Apellido1\":\"Ares\",\"Apellido2\":\"I
caran\",\"TipoDocumento\":\"NIF\",\"IdentificadorDocumento\":\"30624315E\",\"PaisE
misor\":\"Spain\"}"]}' -o $ORDENER_URL --tls --cafile $CA_FILE -C $CHANNEL_NAME


# CARGAS INICIALES

peer chaincode invoke -n personas -c
'{"function":"cargarDatosIniciales","Args":["./json/personas_001.json"]}' -o
$ORDENER_URL --tls --cafile $CA_FILE -C $CHANNEL_NAME

sleep 5

peer chaincode invoke -n personas -c
'{"function":"asignarEstado","Args":["PERSONAS",
"{\"docType\":\"CONTADOR\",\"IDMaximo\":1000}"]}' -o $ORDENER_URL --tls --cafile
$CA_FILE -C $CHANNEL_NAME
```

AFIJOS

Este contrato inteligente es el encargado de gestionar los datos de los afijos de los criadores del sistema.

Un **afijo de un criador** es una denominación o marca personal registrado a través de una sociedad o federación canina que le autoriza a su titular o titulares a utilizarlo en la inscripción de camadas.

Real Sociedad Canina de España

Declarada de Utilidad Pública por Real Orden del Ministerio de Fomento de 27-02-1918.

COMISION DEL «LIBRO DE ORIGENES ESPAÑOL»

Cédula de concesión del afijo DE ALMAYATE Núm afijo: 005513

Fecha de concesión: 9 de Mayo de 2005

(Reconocido por la Fédération Cynologique Internationale con el núm. 0239/94)

V.º B.º EL PRESIDENTE

EL SECRETARIO

TITULAR/ES: MANZANO HERGUIDO, ARTURO

Funciones que incorpora el contrato inteligente:

registrarAfijo

Introduce en el sistema los datos necesarios para registrar y autorizar el uso de un nuevo Afijo de Criador por parte de uno o más usuarios cumpliendo los siguientes requisitos:

- Ningún solicitante puede ser propietario o copropietario de otro afijo
- El nombre afijo no debe existir

- **Argumentos entrada:**

```
type tipoRegistrarAfijoPropietario struct {
        IDPersona int `json:"IDPersona"`
}

type tipoRegistrarAfijo struct {
        Nombre       string                          `json:"Nombre"`
```

```
        Propietarios []tipoRegistrarAfijoPropietario `json:"Propietarios"`
}
```

```
type TipoSeguridad struct {
    IDPersona int `json:"IDPersona"`
}
```

El contrato realiza las siguientes validaciones:

- El número de argumentos de entrada sea 2
- El formato JSON de args[0] y args[1] sea valido
- Los argumentos Nombre y Propietarios tenga un valor.
- No existe otro Afijo con el mismo nombre
- Los propietarios están registrados
- Alguno de los propietarios no tiene actualmente otro afijo asignado como propietario o copropietario
- La persona que invoca la acción esta registrada y dispone de los permisos para realizar dicha acción

- **Argumentos salida:**

Devuelve los registros insertados:

```
type Afijos struct {
        ObjectType string `json:"docType"`
        IDAfijo    int    `json:"IDAfijo"`
        Nombre     string `json:"Nombre"`
        FechaAlta  string `json:"FechaAlta"`
        FechaBaja  string `json:"FechaBaja"`
}

type AfijosPropietarios struct {
        ObjectType         string `json:"docType"`
        IDAfijoPropietario int    `json:"IDAfijoPropietario"`
        IDAfijo            int    `json:"IDAfijo"`
        IDPersona          int    `json:"IDPersona"`
        FechaAlta          string `json:"FechaAlta"`
        FechaBaja          string `json:"FechaBaja"`
}
```

registrarCambioPropietario

Registrar en el sistema los datos de los nuevos propietarios de un afijo, dando de baja el registro de los actuales propietarios.

- **Argumentos entrada:**

```
type tiporegistrarCancelacionAfijo struct {
        IDAfijo int `json:"IDAfijo"`
    }
```

```
type TipoSeguridad struct {
    IDPersona int `json:"IDPersona"`
}
```

El contrato realiza las siguientes validaciones:

- El número de argumentos de entrada sea 2
- El formato JSON de args[0] y args[1] sea valido
- El IDAfijo tenga un valor y corresponda a un afijo activo.
- Los nuevos propietarios están registrados y no tiene ninguno de ellos otro afijo asignado como propietario o copropietario
- La persona que invoca la acción está registrada y dispone de los permisos para realizar dicha acción

- **Argumentos salida:**

Devuelve los registros insertados:

```
type Afijos struct {
        ObjectType string `json:"docType"`
        IDAfijo    int    `json:"IDAfijo"`
        Nombre     string `json:"Nombre"`
        FechaAlta  string `json:"FechaAlta"`
        FechaBaja  string `json:"FechaBaja"`
}

type AfijosPropietariosBaja  struct {
        ObjectType         string `json:"docType"`
        IDAfijoPropietario int    `json:"IDAfijoPropietario"`
        IDAfijo            int    `json:"IDAfijo"`
        IDPersona          int    `json:"IDPersona"`
        FechaAlta          string `json:"FechaAlta"`
        FechaBaja          string `json:"FechaBaja"`
}

type AfijosPropietariosAlta  struct {
        ObjectType         string `json:"docType"`
        IDAfijoPropietario int    `json:"IDAfijoPropietario"`
        IDAfijo            int    `json:"IDAfijo"`
        IDPersona          int    `json:"IDPersona"`
        FechaAlta          string `json:"FechaAlta"`
        FechaBaja          string `json:"FechaBaja"`
}
```

registrarCancelacionAfijo

Registrar en el sistema la cancelación del Afijo de Criador, cuyo nombre no podrá ser utilizado por ninguna otra persona.

- **Argumentos entrada:**

```
type tiporegistrarCancelacionAfijo struct {
    IDAfijo int `json:"IDAfijo"`
}
```

```
type TipoSeguridad struct {
    IDPersona int `json:"IDPersona"`
}
```

El contrato realiza las siguientes validaciones:

- El número de argumentos de entrada sea 2
- El formato JSON de args[0] y args[1] sea valido
- El IDAfijo tenga un valor y corresponda a un afijo activo.
- La persona que invoca la acción está registrada y dispone de los permisos para realizar dicha acción

- **Argumentos salida:**

Devuelve los registros insertados:

```
type Afijos struct {
        ObjectType string `json:"docType"`
        IDAfijo    int    `json:"IDAfijo"`
        Nombre     string `json:"Nombre"`
        FechaAlta  string `json:"FechaAlta"`
        FechaBaja  string `json:"FechaBaja"`
}
type AfijosPropietarios  struct {
        ObjectType         string `json:"docType"`
        IDAfijoPropietario int    `json:"IDAfijoPropietario"`
        IDAfijo            int    `json:"IDAfijo"`
        IDPersona          int    `json:"IDPersona"`
        FechaAlta          string `json:"FechaAlta"`
        FechaBaja          string `json:"FechaBaja"`
}
```

cargarDatosIniciales

Insertar en el sistema los datos los distintos Afijos registrados por las federaciones en un fichero de texto con el siguiente formato JSON:

```
type Afijos struct {
        ObjectType string `json:"docType"`
        IDAfijo    int    `json:"IDAfijo"`
        Nombre     string `json:"Nombre"`
        FechaAlta  string `json:"FechaAlta"`
        FechaBaja  string `json:"FechaBaja"`
}
```

- **Argumentos entrada:**

```
NombreFichero string
```

- **Argumentos salida:**

Devuelve el número de registros insertados:

```
NumeroRegistros string
```

cargarDatosIniciales_Propietarios

Insertar en el sistema los datos los Propietarios de los distintos Afijos registrados por las federaciones en un fichero de texto con el siguiente formato JSON:

```
type AfijosPropietarios struct {
        ObjectType         string `json:"docType"`
        IDAfijoPropietario int    `json:"IDAfijoPropietario"`
        IDAfijo            int    `json:"IDAfijo"`
        IDPersona          int    `json:"IDPersona"`
        FechaAlta          string `json:"FechaAlta"`
        FechaBaja          string `json:"FechaBaja"`
}
```

- **Argumentos entrada:**

```
NombreFichero string
```

- **Argumentos salida:**

Devuelve el número de registros insertados:

```
NumeroRegistros string
```

consultarDatosAfijo

Devuelve al solicitante los datos que sean públicos del afijo consultado.

obtenerCertificadoAfijo

Proporciona al propietario un fichero que certifica la propiedad de este.

Funciones comunes

Funciones de los contratos del sistema incorporadas a este contrato:

- **getQueryResultForQueryString**
- **asignarEstado**
- **borrarEstado**
- **consultarEstado**
- **consultarRangoEstados**
- **ejecutarConsulta**

Plan de pruebas

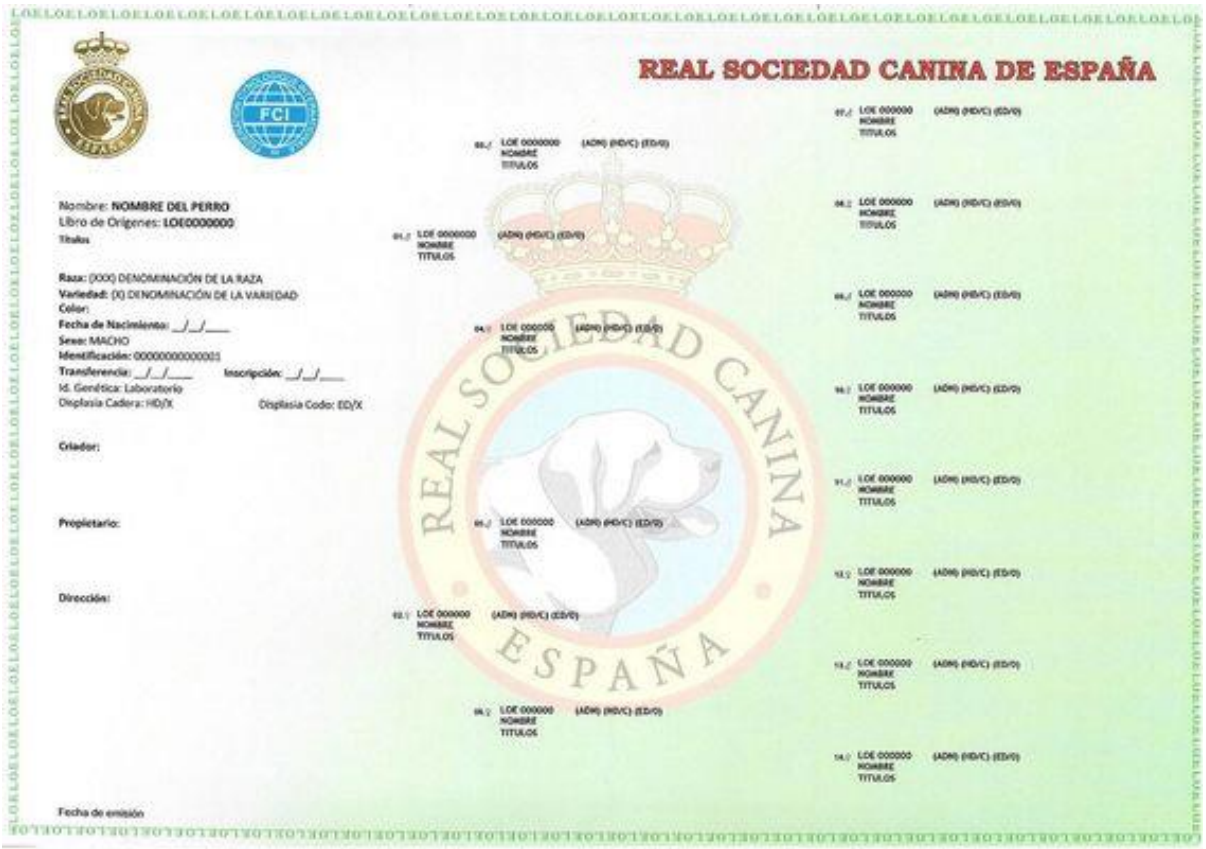
A continuación, se especifican una serie de comando que permite al lector realizar una batería de pruebas sobre el contrato inteligente

```
export CHANNEL_NAME=netcanchannel

export
CA_FILE=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/ordererOrganizat
ions/netcan.com/orderers/orderer.netcan.com/msp/tlscacerts/tlsca.netcan.com-
cert.pem

export ORDENER_URL=orderer.netcan.com:7050


# AFIJOS

peer chaincode invoke -n afijos -c
'{"function":"registrarAfijo","Args":["{\"Nombre\":\"NUEVO
AFIJO\",\"Propietarios\":[{\"IDPersona\":700},{\"IDPersona\":701},{\"IDPersona\":7
02}]}", "{\"IDPersona\":6}"]}' -o $ORDENER_URL --tls --cafile $CA_FILE -C
$CHANNEL_NAME

peer chaincode invoke -n afijos -c
'{"function":"registrarCambioPropietario","Args":["{\"IDAfijo\":51,\"Propietarios\
":[{\"IDPersona\":800},{\"IDPersona\":801},{\"IDPersona\":802},{\"IDPersona\":803}
]}", "{\"IDPersona\":6}"]}' -o $ORDENER_URL --tls --cafile $CA_FILE -C
$CHANNEL_NAME

peer chaincode invoke -n afijos -c
'{"function":"registrarCancelacionAfijo","Args":["{\"IDAfijo\":51}",
"{\"IDPersona\":6}"]}' -o $ORDENER_URL --tls --cafile $CA_FILE -C $CHANNEL_NAME



# ERRORES AFIJOS

peer chaincode invoke -n afijos -c
'{"function":"registrarAfijo","Args":["{\"Nombre\":\"NUEVO
AFIJO\",\"Propietarios\":[{\"IDPersona\":700},{\"IDPersona\":701},{\"IDPersona\":7
02}]}", "{\"IDPersona\":6}"]}' -o $ORDENER_URL --tls --cafile $CA_FILE -C
$CHANNEL_NAME

peer chaincode invoke -n afijos -c
'{"function":"registrarAfijo","Args":["{\"Nombre\":\"OTRO
AFIJO\",\"Propietarios\":[{\"IDPersona\":700},{\"IDPersona\":801},{\"IDPersona\":8
02}]}", "{\"IDPersona\":6}"]}' -o $ORDENER_URL --tls --cafile $CA_FILE -C
$CHANNEL_NAME

peer chaincode invoke -n afijos -c
'{"function":"registrarCambioPropietario","Args":["{\"IDAfijo\":51,\"Propietarios\
":[{\"IDPersona\":1},{\"IDPersona\":2},{\"IDPersona\":3},{\"IDPersona\":4}]}",
"{\"IDPersona\":6}"]}' -o $ORDENER_URL --tls --cafile $CA_FILE -C $CHANNEL_NAME

peer chaincode invoke -n afijos -c
'{"function":"registrarCambioPropietario","Args":["{\"IDAfijo\":1051,\"Propietario
s\":[{\"IDPersona\":800},{\"IDPersona\":801},{\"IDPersona\":802},{\"IDPersona\":80
3}]}", "{\"IDPersona\":6}"]}' -o $ORDENER_URL --tls --cafile $CA_FILE -C
$CHANNEL_NAME
```

```
peer chaincode invoke -n afijos -c
'{"function":"registrarCambioPropietario","Args":["{\"IDAfijo\":51,\"Propietarios\
":[{\"IDPersona\":88888888888888},{\"IDPersona\":2},{\"IDPersona\":3},{\"IDPersona
\":4}]}", "{\"IDPersona\":6}"]}' -o $ORDENER_URL --tls --cafile $CA_FILE -C
$CHANNEL_NAME

peer chaincode invoke -n afijos -c
'{"function":"registrarCancelacionAfijo","Args":["{\"IDAfijo\":51}",
"{\"IDPersona\":6}"]}' -o $ORDENER_URL --tls --cafile $CA_FILE -C $CHANNEL_NAME

peer chaincode invoke -n afijos -c
'{"function":"registrarCambioPropietario","Args":["{\"IDAfijo\":555555551,\"Propie
tarios\":[{\"IDPersona\":800},{\"IDPersona\":801},{\"IDPersona\":802},{\"IDPersona
\":803}]}", "{\"IDPersona\":6}"]}' -o $ORDENER_URL --tls --cafile $CA_FILE -C
$CHANNEL_NAME


# CARGAS INICIALES

peer chaincode invoke -n $CC_NOMBRE_AFIJOS -c
'{"function":"cargarDatosIniciales","Args":["./json/afijos.json"]}' -o
$ORDENER_URL --tls --cafile $CA_FILE -C $CHANNEL_NAME

peer chaincode invoke -n $CC_NOMBRE_AFIJOS -c
'{"function":"asignarEstado","Args":["AFIJOS",
"{\"docType\":\"CONTADOR\",\"IDMaximo\":50}"]}' -o $ORDENER_URL --tls --cafile
$CA_FILE -C $CHANNEL_NAME

peer chaincode invoke -n $CC_NOMBRE_AFIJOS -c
'{"function":"cargarDatosIniciales_Propietarios","Args":["./json/afijos_propietari
os.json"]}' -o $ORDENER_URL --tls --cafile $CA_FILE -C $CHANNEL_NAME

peer chaincode invoke -n $CC_NOMBRE_AFIJOS -c
'{"function":"asignarEstado","Args":["AFIJOS_PROPIETARIOS",
"{\"docType\":\"CONTADOR\",\"IDMaximo\":50}"]}' -o $ORDENER_URL --tls --cafile
$CA_FILE -C $CHANNEL_NAME
```

PERROS

Este contrato inteligente es el encargado de gestionar el registro de ejemplares canino, identificando su origen, pureza de raza, mencionando sus ascendentes y descendientes.



Permite registrar a los propietarios de ejemplar hembra registrar el nacimiento de una nueva camada en un periodo no superior a 1 mes desde su nacimiento (identificando a los progenitores yel nombre y sexo de cada cachorro

También permite el registro de propiedad de perros mestizos en los cuales algún de los progenitores es desconocido o es un mestizo, y los cambios de propietarios, así como la defunción de los ejemplares.

Funciones que incorpora el contrato inteligente:

registrarPerro

Introduce en el sistema los datos necesarios para identificar el origen genealógico de uno o varios perros y registrar sus propietarios.

Permite registrar una camada o un ejemplar, determinando la pureza de raza y el afijo de criador según los criterios de validación detallados más adelante.

- **Argumentos entrada:**

```
type tipoRegistrarCamadaPerro struct {
        Nombre string `json:"Nombre"`
        IDSexo int    `json:"IDSexo"`
}

type tipoRegistrarCamadaPropietario struct {
        IDPersona int `json:"IDPersona"`
}

type tipoRegistrarCamada struct {
        Perros          []tipoRegistrarCamadaPerro       `json:"Perros"`
        IDPerroMadre    int                              `json:"IDPerroMadre"`
        IDPerroPadre    int                              `json:"IDPerroPadre"`
        IDAfijo         int                              `json:"IDAfijo"`
        IDRaza          int                              `json:"IDRaza"`
        FechaNacimiento string                           `json:"FechaNacimiento"`
        Propietarios    []tipoRegistrarCamadaPropietario `json:"Propietarios"`
}
```

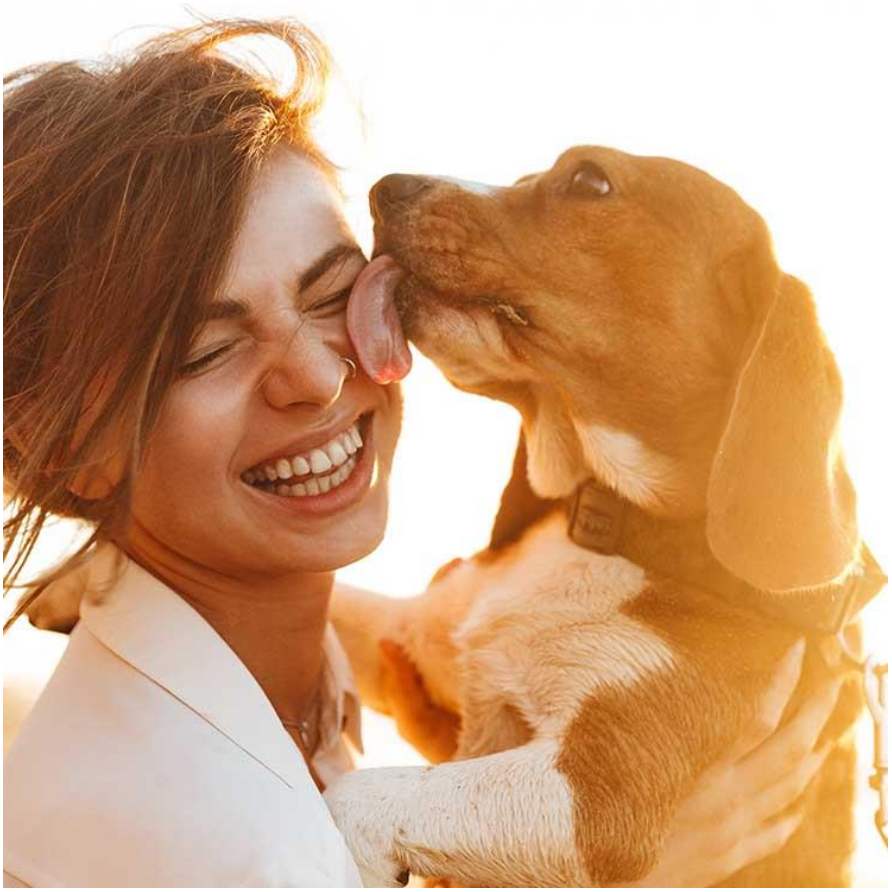
```
type TipoSeguridad struct {
    IDPersona int `json:"IDPersona"`
}
```

El contrato realiza las siguientes validaciones:

- El número de argumentos de entrada sea 2
- El formato JSON de args[0] y args[1] sea valido
- En caso de que la madre no sea desconocida (IDMadre≠0):
    - El IDMadre está registrado y sigue viva.
    - El IDMadre este registrada como ejemplar Hembra (IDSexo = 0)
    - Su edad esta comprendida entre 1 y 10 años (sino perderá la pureza de raza y se asignará IDRaza = 0)
    - Si todos los propietarios de la Madre son propietarios del mismo Afijo de Criador (sino perderá la denominación del afijo, y se asignara IDAfijo = 0) que serán los propietarios del ejemplar

- En caso de que la madre sea desconocida (IDMadre = 0):
    - Los nuevos propietarios están registrados

- En caso de que el padra no sea desconocida (IDPadre≠0):
    - El IDPadre está registrado y sigue vivo.
    - El IDPadre este registrado como ejemplar Macho (IDSexo = 1)
    - Su edad está comprendida entre 1 y 12 años (sino perderá la pureza de raza y se asignará IDRaza = 0)
- Si la raza de los dos progenitores es igual (sino perderá la pureza de raza y se asignará IDRaza=0)
- Los argumentos Nombre e IDSexo tenga un valor valido:
    - En el caso de que IDAfijo ≠ 0 no exista para ese afijo de criador otro ejemplar con el mismo nombre
    - IDSexo contiene o 0 (HEMBRA) o 1 (MACHO)
- El argumento FechaNacimiento tenga un valor q los últimos 30 días (sino perderá la pureza de raza y se asignará IDRaza = 0)
- La persona que invoca la acción está registrada y dispone de los permisos para realizar dicha acción



- **Argumentos salida:**

Devuelve los registros insertados:

```
type Perros struct {
        ObjectType      string `json:"docType"`
        IDPerro         int    `json:"IDPerro"`
        Nombre          string `json:"Nombre"`
        IDAfijo         int    `json:"IDAfijo"`
        IDSexo          int    `json:"IDSexo"`
        IDPerroMadre    int    `json:"IDPerroMadre"`
        IDPerroPadre    int    `json:"IDPerroPadre"`
        IDRaza          int    `json:"IDRaza"`
        FechaNacimiento string `json:"FechaNacimiento"`
        FechaDefuncion  string `json:"FechaDefuncion"`
        FechaAlta       string `json:"FechaAlta"`
        FechaBaja       string `json:"FechaBaja"`
}

type PerrosPropietarios struct {
        ObjectType         string `json:"docType"`
        IDPerroPropietario int    `json:"IDPerroPropietario"`
        IDPerro            int    `json:"IDPerro"`
        IDPersona          int    `json:"IDPersona"`
        FechaAlta          string `json:"FechaAlta"`
        FechaBaja          string `json:"FechaBaja"`
}
```

registrarCambioPropietario

Registrar en el sistema los datos de los nuevos propietarios de un ejemplar, dando de baja el registro de los actuales propietarios.

- **Argumentos entrada:**

```
type TipoRegistrarCambioPropietarioPropietario struct {
    IDPersona int `json:"IDPersona"`
}

type TipoRegistrarCambioPropietario struct {
    IDPerro      int `json:"IDPerro"`
    Propietarios []TipoRegistrarCambioPropietarioPropietario `json:"Propietarios"`
}
```

```
type TipoSeguridad struct {
    IDPersona int `json:"IDPersona"`
}
```

El contrato realiza las siguientes validaciones:

- El número de argumentos de entrada sea 2
- El formato JSON de args[0] y args[1] sea valido
- El IDAfijo tenga un valor y corresponda a un afijo activo.
- Los nuevos propietarios están registrados
- Alguno de los nuevos propietarios no tiene actualmente otro afijo asignado como propietario o copropietario
- La persona que invoca la acción está registrada y dispone de los permisos para realizar dicha acción

- **Argumentos salida:**

Devuelve los registros insertados:

```
type Perros struct {
        ObjectType      string `json:"docType"`
        IDPerro         int    `json:"IDPerro"`
        Nombre          string `json:"Nombre"`
```

```
        IDAfijo         int    `json:"IDAfijo"`
        IDSexo          int    `json:"IDSexo"`
        IDPerroMadre    int    `json:"IDPerroMadre"`
        IDPerroPadre    int    `json:"IDPerroPadre"`
        IDRaza          int    `json:"IDRaza"`
        FechaNacimiento string `json:"FechaNacimiento"`
        FechaDefuncion  string `json:"FechaDefuncion"`
        FechaAlta       string `json:"FechaAlta"`
        FechaBaja       string `json:"FechaBaja"`
}

type PerrosPropietariosBaja struct {
        ObjectType         string `json:"docType"`
        IDPerroPropietario int    `json:"IDPerroPropietario"`
        IDPerro            int    `json:"IDPerro"`
        IDPersona          int    `json:"IDPersona"`
        FechaAlta          string `json:"FechaAlta"`
        FechaBaja          string `json:"FechaBaja"`
}

type PerrosPropietariosAlta struct {
        ObjectType         string `json:"docType"`
        IDPerroPropietario int    `json:"IDPerroPropietario"`
        IDPerro            int    `json:"IDPerro"`
        IDPersona          int    `json:"IDPersona"`
        FechaAlta          string `json:"FechaAlta"`
        FechaBaja          string `json:"FechaBaja"`
}
```

registrarDefuncionPerro

Registrar en el sistema la baja de un ejemplar por defunción.

- **Argumentos entrada:**

```
type tiporegistrarCancelacionAfijo struct {
       IDAfijo int `json:"IDAfijo"`
}
```

```
type TipoSeguridad struct {
        IDPersona int `json:"IDPersona"`
}
```

El contrato realiza las siguientes validaciones:

- El número de argumentos de entrada sea 2
- El formato JSON de args[0] y args[1] sea valido
- El IDPerro tenga un valor y corresponda a un ejemplar vivo.

- La persona que invoca la acción está registrada y dispone de los permisos para realizar dicha acción

- **Argumentos salida:**

Devuelve los registros insertados:

```
type Perros struct {
        ObjectType      string `json:"docType"`
        IDPerro         int    `json:"IDPerro"`
        Nombre          string `json:"Nombre"`
        IDAfijo         int    `json:"IDAfijo"`
        IDSexo          int    `json:"IDSexo"`
        IDPerroMadre    int    `json:"IDPerroMadre"`
        IDPerroPadre    int    `json:"IDPerroPadre"`
        IDRaza          int    `json:"IDRaza"`
        FechaNacimiento string `json:"FechaNacimiento"`
        FechaDefuncion  string `json:"FechaDefuncion"`
        FechaAlta       string `json:"FechaAlta"`
        FechaBaja       string `json:"FechaBaja"`
}

type PerrosPropietarios struct {
        ObjectType         string `json:"docType"`
        IDPerroPropietario int    `json:"IDPerroPropietario"`
        IDPerro            int    `json:"IDPerro"`
        IDPersona          int    `json:"IDPersona"`
        FechaAlta          string `json:"FechaAlta"`
        FechaBaja          string `json:"FechaBaja"`
}
```

registrarReconocimientoRaza

Registrar en el sistema la certificación de la Pueeza de raza de un ejemplar cuyos ascendentes no están determinados o registrados en un libro de origen genealógico.

Este hecho se realiza de manera excepcional por una de las siguientes causas:

- Un juez especialista de Raza certifica en un evento organizado por las federaciones que el ejemplar cumple con todas las características del estándar de la raza.
- El propietario presenta una Certificado de un Libro de Origen Genealógico reconocido y la federación verifica la autenticación del documento con el organismo que gestiona el libro.

- **Argumentos entrada:**

```
type PerroReconocimientoRaza struct {
        IDPerro                 int    `json:"IDPerro"`
        IDRaza                  int    `json:"IDRaza"`
        Justificacion           string `json:"Justificacion"`
}
```

```
type TipoSeguridad struct {
        IDPersona int `json:"IDPersona"`
}
```

El contrato realiza las siguientes validaciones:

- El número de argumentos de entrada sea 2
- El formato JSON de args[0] y args[1] sea valido
- El IDPerro tenga un valor y corresponda a un ejemplar vivo.
- El IDRaza tenga un valor y corresponda a un afijo activo.
- La Justificación tenga un valor
- La persona que invoca la acción está registrada y dispone de los permisos para realizar dicha acción

- **Argumentos salida:**

Devuelve los registros insertados:

```
type Perros struct {
        ObjectType      string `json:"docType"`
        IDPerro         int    `json:"IDPerro"`
        Nombre          string `json:"Nombre"`
        IDAfijo         int    `json:"IDAfijo"`
        IDSexo          int    `json:"IDSexo"`
        IDPerroMadre    int    `json:"IDPerroMadre"`
        IDPerroPadre    int    `json:"IDPerroPadre"`
        IDRaza          int    `json:"IDRaza"`
        FechaNacimiento string `json:"FechaNacimiento"`
        FechaDefuncion  string `json:"FechaDefuncion"`
        FechaAlta       string `json:"FechaAlta"`
        FechaBaja       string `json:"FechaBaja"`
}

type PerrosPropietarios struct {
        ObjectType         string `json:"docType"`
        IDPerroPropietario int    `json:"IDPerroPropietario"`
        IDPerro            int    `json:"IDPerro"`
        IDPersona          int    `json:"IDPersona"`
        FechaAlta          string `json:"FechaAlta"`
        FechaBaja          string `json:"FechaBaja"`
}
```

cargarDatosIniciales

Registrar en el sistema los ejemplares definidos en un fichero de texto con el siguiente formato JSON:

```
type Perros struct {
        ObjectType      string `json:"docType"`
        IDPerro         int    `json:"IDPerro"`
```

```
        Nombre          string `json:"Nombre"`
        IDAfijo         int    `json:"IDAfijo"`
        IDSexo          int    `json:"IDSexo"`
        IDPerroMadre    int    `json:"IDPerroMadre"`
        IDPerroPadre    int    `json:"IDPerroPadre"`
        IDRaza          int    `json:"IDRaza"`
        FechaNacimiento string `json:"FechaNacimiento"`
        FechaDefuncion  string `json:"FechaDefuncion"`
        FechaAlta       string `json:"FechaAlta"`
        FechaBaja       string `json:"FechaBaja"`
}
```

- **Argumentos entrada:**

```
NombreFichero string
```

- **Argumentos salida:**

Devuelve el número de registros insertados:

```
NumeroRegistros string
```

cargarDatosIniciales_Propietarios

Registrar en el sistema los propietarios de los ejemplares definidos en un fichero de texto con el siguiente formato JSON:

```
type PerrosPropietarios struct {
        ObjectType         string `json:"docType"`
        IDPerroPropietario int    `json:"IDPerroPropietario"`
        IDPerro            int    `json:"IDPerro"`
        IDPersona          int    `json:"IDPersona"`
        FechaAlta          string `json:"FechaAlta"`
        FechaBaja          string `json:"FechaBaja"`
}
```

- **Argumentos entrada:**

```
NombreFichero string
```

- **Argumentos salida:**

Devuelve el número de registros insertados:

```
NumeroRegistros string
```

consultarDatosEjemplar

Consultar los datos del ejemplar. La información presentada variará dependiendo de la persona que realiza la consulta.

obtenerCertificadoRegistro

Obtener un certificado con los datos del registro que certifica su inscripción en el libro genealógico.

obtenerPedigri

Obtener un fichero digital que contiene los datos del ejemplar y su línea genealógica que certifica su pureza de raza.

registrarCesionTemporal

Solicitar el registro de cesión temporal de la propiedad a uno o varios criadores por un tiempo limitado.

Funciones comunes

Funciones de los contratos del sistema incorporadas a este contrato:

- **getQueryResultForQueryString**
- **asignarEstado**
- **borrarEstado**
- **consultarEstado**
- **consultarRangoEstados**
- **ejecutarConsulta**

Plan de pruebas

A continuación, se especifican una serie de comando que permite al lector realizar una batería de pruebas sobre el contrato inteligente

```
export CHANNEL_NAME=netcanchannel

export
CA_FILE=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/ordererOrganizat
ions/netcan.com/orderers/orderer.netcan.com/msp/tlscacerts/tlsca.netcan.com-
cert.pem

export ORDENER_URL=orderer.netcan.com:7050


# PERROS

peer chaincode invoke -n perros -c
'{"function":"registrarPerro","Args":["{\"Perros\":[{\"Nombre\":\"PERRO_0001\",\"I
DSexo\":1},{\"Nombre\":\"PERRA_0001\",\"IDSexo\":0}],\"IDPerroMadre\":181,\"IDPerr
oPadre\":193,\"FechaNacimiento\":\"2020-08-23\"}", "{\"IDPersona\":6}"]}' -o
$ORDENER_URL --tls --cafile $CA_FILE -C $CHANNEL_NAME

peer chaincode invoke -n perros -c
'{"function":"registrarPerro","Args":["{\"Perros\":[{\"Nombre\":\"PERRO_0002\",\"I
DSexo\":1},{\"Nombre\":\"PERRA_0002\",\"IDSexo\":0}],\"IDPerroMadre\":0,\"IDPerroP
adre\":0,\"FechaNacimiento\":\"2020-08-
23\",\"Propietarios\":[{\"IDPersona\":600},{\"IDPersona\":601}]}",
"{\"IDPersona\":6}"]}' -o $ORDENER_URL --tls --cafile $CA_FILE -C $CHANNEL_NAME

peer chaincode invoke -n perros -c
'{"function":"registrarPerro","Args":["{\"Perros\":[{\"Nombre\":\"PERRO_0001\",\"I
DSexo\":1},{\"Nombre\":\"PERRA_0001\",\"IDSexo\":0}],\"IDPerroMadre\":0,\"IDPerroP
adre\":193,\"FechaNacimiento\":\"2020-08-
23\",\"Propietarios\":[{\"IDPersona\":600},{\"IDPersona\":601}]}",
"{\"IDPersona\":6}"]}' -o $ORDENER_URL --tls --cafile $CA_FILE -C $CHANNEL_NAME

peer chaincode invoke -n $CC_NOMBRE_PERROS -c
'{"function":"registrarCambioPropietario","Args":["{\"IDPerro\":5760,\"Propietario
s\":[{\"IDPersona\":401},{\"IDPersona\":402},{\"IDPersona\":403},{\"IDPersona\":40
4}]}", "{\"IDPersona\":6}"]}' -o $ORDENER_URL --tls --cafile $CA_FILE -C
$CHANNEL_NAME

peer chaincode invoke -n $CC_NOMBRE_PERROS -c
'{"function":"registrarDefuncionPerro","Args":["{\"IDPerro\":100,\"FechaDefuncion\
":\"2020-08-08\"}", "{\"IDPersona\":6}"]}' -o $ORDENER_URL --tls --cafile $CA_FILE
-C $CHANNEL_NAME

peer chaincode invoke -n perros -c
'{"function":"registrarReconocimientoRaza","Args":["{\"IDPerro\":37,\"IDRaza\":126
,\"Justificacion\":\"Ha sido reconocida su raza por el Juez PEPE GUTIERREZ
DOSSANTOS\"}", "{\"IDPersona\":6}"]}' -o $ORDENER_URL --tls --cafile $CA_FILE -C
$CHANNEL_NAME
```

```
# ERRORES PERROS

peer chaincode invoke -n perros -c
'{"function":"registrarPerro","Args":["{\"Perros\":[{\"Nombre\":\"PERRO_0001\",\"I
DSexo\":1},{\"Nombre\":\"PERRA_0001\",\"IDSexo\":0}],\"IDPerroMadre\":181,\"IDPerr
oPadre\":3,\"FechaNacimiento\":\"2020-08-23\"}", "{\"IDPersona\":6}"]}' -o
$ORDENER_URL --tls --cafile $CA_FILE -C $CHANNEL_NAME

peer chaincode invoke -n perros -c
'{"function":"registrarPerro","Args":["{\"Perros\":[{\"Nombre\":\"PERRO_0001\",\"I
DSexo\":1},{\"Nombre\":\"PERRA_0001\",\"IDSexo\":0}],\"IDPerroMadre\":181,\"IDPerr
oPadre\":10,\"FechaNacimiento\":\"2020-08-23\"}", "{\"IDPersona\":6}"]}' -o
$ORDENER_URL --tls --cafile $CA_FILE -C $CHANNEL_NAME

peer chaincode invoke -n perros -c
'{"function":"registrarPerro","Args":["{\"Perros\":[{\"Nombre\":\"PERRO_0001\",\"I
DSexo\":1},{\"Nombre\":\"PERRA_0001\",\"IDSexo\":0}],\"IDPerroMadre\":0,\"IDPerroP
adre\":193,\"FechaNacimiento\":\"2020-08-23\"}", "{\"IDPersona\":6}"]}' -o
$ORDENER_URL --tls --cafile $CA_FILE -C $CHANNEL_NAME

peer chaincode invoke -n $CC_NOMBRE_PERROS -c
'{"function":"registrarCambioPropietario","Args":["{\"IDPerro\":577777777760,\"Pro
pietarios\":[{\"IDPersona\":401},{\"IDPersona\":402},{\"IDPersona\":403},{\"IDPers
ona\":404}]}", "{\"IDPersona\":6}"]}' -o $ORDENER_URL --tls --cafile $CA_FILE -C
$CHANNEL_NAME

peer chaincode invoke -n $CC_NOMBRE_PERROS -c
'{"function":"registrarCambioPropietario","Args":["{\"IDPerro\":5760,\"Propietario
s\":[{\"IDPersona\":401},{\"IDPersona\":402},{\"IDPersona\":444444444403},{\"IDPer
sona\":404}]}", "{\"IDPersona\":6}"]}' -o $ORDENER_URL --tls --cafile $CA_FILE -C
$CHANNEL_NAME
```

```
Peer chaincode invoke -n $CC_NOMBRE_PERROS -c
'{"function":"registrarDefuncionPerro","Args":["{\"IDPerro\":100,\"FechaDefuncion\
":\"2020-08-08\"}", "{\"IDPersona\":6}"]}' -o $ORDENER_URL --tls --cafile $CA_FILE
-C $CHANNEL_NAME

peer chaincode invoke -n $CC_NOMBRE_PERROS -c
'{"function":"registrarDefuncionPerro","Args":["{\"IDPerro\":1000000000,\"FechaDef
uncion\":\"2020-08-08\"}", "{\"IDPersona\":6}"]}' -o $ORDENER_URL --tls --cafile
$CA_FILE -C $CHANNEL_NAME

peer chaincode invoke -n $CC_NOMBRE_PERROS -c
'{"function":"registrarDefuncionPerro","Args":["{\"IDPerro\":100\"FechaDefuncion\"
:\"2020-08-08\"}", "{\"IDPersona\":6}"]}' -o $ORDENER_URL --tls --cafile $CA_FILE
-C $CHANNEL_NAME

peer chaincode invoke -n perros -c
'{"function":"registrarReconocimientoRaza","Args":["{\"IDPerro\":12,\"IDRaza\":126
,\"Justificacion\":\"Ha sido reconocida su raza por el Juez PEPE GUTIERREZ
DOSSANTOS\"}", "{\"IDPersona\":6}"]}' -o $ORDENER_URL --tls --cafile $CA_FILE -C
$CHANNEL_NAME



# CARGAS INICIALES

peer chaincode invoke -n perros -c
'{"function":"cargarDatosIniciales","Args":["./json/perros_001.json"]}' -o
$ORDENER_URL --tls --cafile $CA_FILE -C $CHANNEL_NAME

sleep 7s

peer chaincode invoke -n perros -c
'{"function":"cargarDatosIniciales","Args":["./json/perros_002.json"]}' -o
$ORDENER_URL --tls --cafile $CA_FILE -C $CHANNEL_NAME

sleep 7s

peer chaincode invoke -n perros -c
'{"function":"cargarDatosIniciales","Args":["./json/perros_003.json"]}' -o
$ORDENER_URL --tls --cafile $CA_FILE -C $CHANNEL_NAME

sleep 7s

peer chaincode invoke -n perros -c
'{"function":"cargarDatosIniciales","Args":["./json/perros_004.json"]}' -o
$ORDENER_URL --tls --cafile $CA_FILE -C $CHANNEL_NAME

sleep 7s

peer chaincode invoke -n perros -c
'{"function":"cargarDatosIniciales","Args":["./json/perros_005.json"]}' -o
$ORDENER_URL --tls --cafile $CA_FILE -C $CHANNEL_NAME

sleep 7s

peer chaincode invoke -n perros -c '{"function":"asignarEstado","Args":["PERROS",
"{\"docType\":\"CONTADOR\",\"IDMaximo\":5759}"]}' -o $ORDENER_URL --tls --cafile
$CA_FILE -C $CHANNEL_NAME

peer chaincode invoke -n perros -c
'{"function":"cargarDatosIniciales_Propietarios","Args":["./json/perros_propietari
os_001.json"]}' -o $ORDENER_URL --tls --cafile $CA_FILE -C $CHANNEL_NAME

sleep 7s
```

```
peer chaincode invoke -n perros -c
'{"function":"cargarDatosIniciales_Propietarios","Args":["./json/perros_propietari
os_002.json"]}' -o $ORDENER_URL --tls --cafile $CA_FILE -C $CHANNEL_NAME

sleep 7s

peer chaincode invoke -n perros -c
'{"function":"cargarDatosIniciales_Propietarios","Args":["./json/perros_propietari
os_003.json"]}' -o $ORDENER_URL --tls --cafile $CA_FILE -C $CHANNEL_NAME

sleep 7s

peer chaincode invoke -n perros -c
'{"function":"cargarDatosIniciales_Propietarios","Args":["./json/perros_propietari
os_004.json"]}' -o $ORDENER_URL --tls --cafile $CA_FILE -C $CHANNEL_NAME

sleep 7s

peer chaincode invoke -n perros -c
'{"function":"cargarDatosIniciales_Propietarios","Args":["./json/perros_propietari
os_005.json"]}' -o $ORDENER_URL --tls --cafile $CA_FILE -C $CHANNEL_NAME

sleep 7s

peer chaincode invoke -n perros -c
'{"function":"asignarEstado","Args":["PERROS_PROPIETARIOS",
"{\"docType\":\"CONTADOR\",\"IDMaximo\":5759}"]}' -o $ORDENER_URL --tls --cafile
$CA_FILE -C $CHANNEL_NAME
```

SOLICITUDES

Este contrato inteligente es el encargado de gestionar las solicitudes de los usuarios cuando es necesario realizar la validación o autorización de la acción por más de una persona.

Por ejemplo:

- Cuando se desea realizar un cambio de propietario de un ejemplar y este pertenece a varias personas, requiere la autorización de ambas personas.
- Cuando se desea registrar el nacimiento de una nueva camada y los progenitores pertenecen a dos criadores o propietarios distintos, no basta con que se registre la solicitud por parte de uno de ellos sino de ambos.

La solicitud es registrada en el sistema por cualquiera de los usuarios implicados, quedando pendiente su ejecución hasta que todos los implicados validan o rechazan el contenido de la solicitud o caduca por transcurrir un tiempo desde su registro.

Una vez que todos los implicados han validado la solicitud esta se ejecutara automáticamente.

En el caso en el que solo sea necesaria la intervención de un usuario y sea este mismo usuario el que registrara la solicitud, esta se ejecutar a también automáticamente.

Con estas certificaciones se evitan fraudes que son comunes, como por ejemplo vender un perro a un nuevo propietario falsificando su origen genealógico (por ejemplo, es hijo del campeón del España)

Las funciones que incorpora el contrato inteligente son similares a las vistas anteriormente en otros contratos:

- **Argumentos entrada:**

Serán los mismos parámetros de entrada que tendría la llamada a la función si realizara la acción directamente.

Por ejemplo:

solicitarRegistrarPerro tendrá los mismos parámetros de entrada la función **registrarPerro** del contrato **PERROS:**

```
type tipoRegistrarCamadaPerro struct {
        Nombre string `json:"Nombre"`
        IDSexo int    `json:"IDSexo"`
}

type tipoRegistrarCamadaPropietario struct {
        IDPersona int `json:"IDPersona"`
}

type tipoRegistrarCamada struct {
        Perros          []tipoRegistrarCamadaPerro       `json:"Perros"`
        IDPerroMadre    int                              `json:"IDPerroMadre"`
        IDPerroPadre    int                              `json:"IDPerroPadre"`
        IDAfijo         int                              `json:"IDAfijo"`
        IDRaza          int                              `json:"IDRaza"`
        FechaNacimiento string                           `json:"FechaNacimiento"`
        Propietarios    []tipoRegistrarCamadaPropietario `json:"Propietarios"`
}
```

```
type TipoSeguridad struct {
    IDPersona int `json:"IDPersona"`
}
```

- **Argumentos salida:**

Si la solicitud se puede ejecutar automáticamente devolverá los mismos parámetros de salida que la función **registrarPerro** del contrato **PERROS:**

```
type Perros struct {
        ObjectType      string `json:"docType"`
        IDPerro         int    `json:"IDPerro"`
        Nombre          string `json:"Nombre"`
        IDAfijo         int    `json:"IDAfijo"`
        IDSexo          int    `json:"IDSexo"`
```

```
        IDPerroMadre    int    `json:"IDPerroMadre"`
        IDPerroPadre    int    `json:"IDPerroPadre"`
        IDRaza          int    `json:"IDRaza"`
        FechaNacimiento string `json:"FechaNacimiento"`
        FechaDefuncion  string `json:"FechaDefuncion"`
        FechaAlta       string `json:"FechaAlta"`
        FechaBaja       string `json:"FechaBaja"`
}

type PerrosPropietarios struct {
        ObjectType         string `json:"docType"`
        IDPerroPropietario int    `json:"IDPerroPropietario"`
        IDPerro            int    `json:"IDPerro"`
        IDPersona          int    `json:"IDPersona"`
        FechaAlta          string `json:"FechaAlta"`
        FechaBaja          string `json:"FechaBaja"`
}
```

En caso contrario devolverá los registros de la solicitud insertados:

```
type Solicitudes struct {
    ObjectType           string `json:"docType"`
    IDSolicitud          int    `json:"IDSolicitud"`
    TipoSolicitud        string `json:"TipoSolicitud"`
    JSONSolicitud        string `json:"JSONSolicitud"`
    IDPersonaSolicitante int    `json:"IDPersonaSolicitante"`
    EstadoSolicitud      string `json:"EstadoSolicitud"`
    FechaEjecucion       string `json:"FechaEjecucion"`
    FechaAlta            string `json:"FechaAlta"`
    FechaBaja            string `json:"FechaBaja"`
}

type SolicitudesAutorizaciones struct {
    ObjectType      string `json:"docType"`
    IDSolicitud     int    `json:"IDSolicitud"`
    IDPersona       int    `json:"IDPersona"`
    EstadoSolicitud string `json:"EstadoSolicitud"`
```

```
    FechaEjecucion  string `json:"FechaEjecucion"`
    FechaAlta       string `json:"FechaAlta"`
    FechaBaja       string `json:"FechaBaja"`
}
```

donde la autorización del usuario que realiza la solicitud estará registrada y aprobada automáticamente, por lo que no necesitará validar esta acción.

El contrato valida los mismos parámetros que si realizara la acción y no permitirá que se registren una solicitud si existen otra activa, referente a la misma acción.

De esta forma existe la siguiente correlación de funciones y parámetros entre contratos:

PERROS

solicitarRegistrarCamada

- Función **registrarPerro** del contrato **PERROS**

solicitarRegistrarPerro

- Función **registrarPerro** del contrato **PERROS**

solicitarRegistrarCambioPropietarioPerro

- Función **registrarCambioPropietario** del contrato **PERROS**

AFIJOS

solicitarRegistrarCambioPropietarioAfijo

- Función **registrarCambioPropietario** del contrato **AFIJOS**

solicitarRegistrarCancelacionAfijo

- Función **registrarCancelacionAfijo** del contrato **AFIJOS**

El contrato inteligente también incorpora las siguientes funciones:

validarSolicitud

Registrar las validaciones que los usuarios implicados en una solicitud realizan sobre una petición. en el sistema el número o código de microchip insertado subcutáneamente al ejemplar por un veterinario.

En el caso de que todos los usuarios implicados en la solicitud den el visto bueno a solicitud, esta se procederá a ejecutarse automáticamente.

- **Argumentos entrada:**

```
type tipoValidarSolicitud struct {
                IDSolicitud     int    `json:"IDSolicitud"`
                EstadoSolicitud string `json:"EstadoSolicitud"`
        }
```

```
type TipoSeguridad struct {
    IDPersona int `json:"IDPersona"`
}
```

El contrato realiza las siguientes validaciones:

- El número de argumentos de entrada sea 2
- El formato JSON de args[0] y args[1] sea valido

- El IDSolicitud tenga un valor y corresponda a una solicitud que se encuentre en estado pendiente (no ejecutada y no caducada)
- El EstadoSolicitud tenga un valor valido (APROBADO / RECHAZADO).
- La persona que invoca la acción está registrada y dispone de los permisos para realizar dicha acción

- **Argumentos salida:**

En el caso de que el usuario valide la solicitud y no quede ningún otro usuario por validar el contrato ejecutara automáticamente la solicitud devolviendo los parámetros de la ejecución de la acción solicitada.

En caso contrario devolverá el registros actualizado:

```
type SolicitudesAutorizaciones struct {
        ObjectType      string `json:"docType"`
        IDSolicitud     int    `json:"IDSolicitud"`
        IDPersona       int    `json:"IDPersona"`
        EstadoSolicitud string `json:"EstadoSolicitud"`
        FechaEjecucion  string `json:"FechaEjecucion"`
        FechaAlta       string `json:"FechaAlta"`
        FechaBaja       string `json:"FechaBaja"`
}
```

cargarDatosIniciales

Introduce en el sistema el historial de las solicitudes que han realizado los diferentes usuarios, definidos en un fichero de texto con el siguiente formato JSON:

```
type Solicitudes struct {
        ObjectType           string `json:"docType"`
        IDSolicitud          int    `json:"IDSolicitud"`
        TipoSolicitud        string `json:"TipoSolicitud"`
        JSONSolicitud        string `json:"JSONSolicitud"`
        IDPersonaSolicitante int    `json:"IDPersonaSolicitante"`
        EstadoSolicitud      string `json:"EstadoSolicitud"`
        FechaEjecucion       string `json:"FechaEjecucion"`
        FechaAlta            string `json:"FechaAlta"`
        FechaBaja            string `json:"FechaBaja"`
}
```

- **Argumentos entrada:**

```
NombreFichero string
```

- **Argumentos salida:**

Devuelve el número de registros insertados:

```
NumeroRegistros string
```

cargarDatosIniciales_Autorizaciones

Introduce en el sistema el historial de las autorizaciones de las solicitudes que han realizado los diferentes usuarios, definidos en un fichero de texto con el siguiente formato JSON:

```
type SolicitudesAutorizaciones struct {
        ObjectType      string `json:"docType"`
        IDSolicitud     int    `json:"IDSolicitud"`
        IDPersona       int    `json:"IDPersona"`
        EstadoSolicitud string `json:"EstadoSolicitud"`
        FechaEjecucion  string `json:"FechaEjecucion"`
        FechaAlta       string `json:"FechaAlta"`
        FechaBaja       string `json:"FechaBaja"`
}
```

- **Argumentos entrada:**

```
NombreFichero string
```

- **Argumentos salida:**

Devuelve el número de registros insertados:

```
NumeroRegistros string
```

Funciones complementarias

- **querySolicitudes**

  Permite a un usuario consultar las solicitudes de un IDPersona y sus estados

Funciones comunes

Funciones de los contratos del sistema incorporadas a este contrato:

- **getQueryResultForQueryString**
- **asignarEstado**
- **borrarEstado**
- **consultarEstado**
- **consultarRangoEstados**
- **ejecutarConsulta**

Plan de pruebas

A continuación, se especifican una serie de comando que permite al lector realizar una batería de pruebas sobre el contrato inteligente

```
export CHANNEL_NAME=netcanchannel

export
CA_FILE=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/ordererOrganizat
ions/netcan.com/orderers/orderer.netcan.com/msp/tlscacerts/tlsca.netcan.com-
cert.pem

export ORDENER_URL=orderer.netcan.com:7050


# SOLICITUDES


# solicitarRegistrarCamada / Perro

peer chaincode invoke -n solicitudes -c
'{"function":"solicitarRegistrarCamada","Args":["{\"Perros\":[{\"nombre\":\"PERRO_
uno\",\"IDSexo\":1},{\"nombre\":\"PERRA_uno\",\"IDSexo\":0}],\"IDPerroMadre\":0,\"
```

```
IDPerroPadre\":0,\"FechaNacimiento\":\"2020-08-
23\",\"Propietarios\":[{\"IDPersona\":666},{\"IDPersona\":999}]}",
"{\"IDPersona\":6}"]}' -o $ORDENER_URL --tls --cafile $CA_FILE -C $CHANNEL_NAME

peer chaincode invoke -n solicitudes -c
'{"function":"solicitarRegistrarCamada","Args":["{\"Perros\":[{\"nombre\":\"PERRO_
dos\",\"IDSexo\":1},{\"nombre\":\"PERRA_dos\",\"IDSexo\":1}],\"IDPerroMadre\":1,\"
IDPerroPadre\":50,\"FechaNacimiento\":\"2020-08-23\"}", "{\"IDPersona\":6}"]}' -o
$ORDENER_URL --tls --cafile $CA_FILE -C $CHANNEL_NAME

peer chaincode invoke -n solicitudes -c
'{"function":"ejecutarConsulta","Args":["{\"selector\": {\"docType\":
\"SOLICITUDES_AUTORIZACIONES\", \"EstadoSolicitud\": \"PENDIENTE\", \"FechaBaja\":
{ \"$lt\": \"2020-09-15\"}}} ", "{\"IDPersona\":5}"]}' -o $ORDENER_URL --tls --
cafile $CA_FILE -C $CHANNEL_NAME

peer chaincode invoke -n solicitudes -c
'{"function":"validarSolicitud","Args":["{\"IDSolicitud\":2,\"EstadoSolicitud\":\"
APROBADO\"}", "{\"IDPersona\":5}"]}' -o $ORDENER_URL --tls --cafile $CA_FILE -C
$CHANNEL_NAME

peer chaincode invoke -n solicitudes -c
'{"function":"solicitarRegistrarCamada","Args":["{\"Perros\":[{\"nombre\":\"PADRE_
tres\",\"IDSexo\":1},{\"nombre\":\"MADRE_tres\",\"IDSexo\":1}],\"IDPerroMadre\":1,
\"IDPerroPadre\":50,\"FechaNacimiento\":\"2021-05-22\"}", "{\"IDPersona\":6}"]}' -
o $ORDENER_URL --tls --cafile $CA_FILE -C $CHANNEL_NAME

peer chaincode invoke -n solicitudes -c
'{"function":"validarSolicitud","Args":["{\"IDSolicitud\":3,\"EstadoSolicitud\":\"
RECHAZADO\"}", "{\"IDPersona\":5}"]}' -o $ORDENER_URL --tls --cafile $CA_FILE -C
$CHANNEL_NAME
```

```
# solicitarRegistrarCambioPropietarioPerro

peer chaincode invoke -n $CC_NOMBRE_SOLICITUDES -c
'{"function":"solicitarRegistrarCambioPropietarioPerro","Args":["{\"IDPerro\":1,\"
Propietarios\":[{\"IDPersona\":1},{\"IDPersona\":2}]}", "{\"IDPersona\":1}"]}' -o
$ORDENER_URL --tls --cafile $CA_FILE -C $CHANNEL_NAME

peer chaincode invoke -n solicitudes -c
'{"function":"validarSolicitud","Args":["{\"IDSolicitud\":4,\"EstadoSolicitud\":\"
APROBADO\"}", "{\"IDPersona\":6}"]}' -o $ORDENER_URL --tls --cafile $CA_FILE -C
$CHANNEL_NAME


# solicitarRegistrarCambioPropietarioAfijo

peer chaincode invoke -n solicitudes -c
'{"function":"solicitarRegistrarCambioPropietarioAfijo","Args":["{\"IDAfijo\":2,\"
Propietarios\":[{\"IDPersona\":201},{\"IDPersona\":202},{\"IDPersona\":203},{\"IDP
ersona\":204}]}", "{\"IDPersona\":5}"]}' -o $ORDENER_URL --tls --cafile $CA_FILE -
C $CHANNEL_NAME

peer chaincode invoke -n solicitudes -c
'{"function":"solicitarRegistrarCambioPropietarioAfijo","Args":["{\"IDAfijo\":2,\"
Propietarios\":[{\"IDPersona\":5}]}", "{\"IDPersona\":201}"]}' -o $ORDENER_URL --
tls --cafile $CA_FILE -C $CHANNEL_NAME

peer chaincode invoke -n solicitudes -c
'{"function":"validarSolicitud","Args":["{\"IDSolicitud\":6,\"EstadoSolicitud\":\"
APROBADO\"}", "{\"IDPersona\":202}"]}' -o $ORDENER_URL --tls --cafile $CA_FILE -C
$CHANNEL_NAME

peer chaincode invoke -n solicitudes -c
'{"function":"validarSolicitud","Args":["{\"IDSolicitud\":6,\"EstadoSolicitud\":\"
APROBADO\"}", "{\"IDPersona\":203}"]}' -o $ORDENER_URL --tls --cafile $CA_FILE -C
$CHANNEL_NAME

peer chaincode invoke -n solicitudes -c
'{"function":"validarSolicitud","Args":["{\"IDSolicitud\":6,\"EstadoSolicitud\":\"
APROBADO\"}", "{\"IDPersona\":204}"]}' -o $ORDENER_URL --tls --cafile $CA_FILE -C
$CHANNEL_NAME


# solicitarRegistrarCancelacionAfijo"

peer chaincode invoke -n solicitudes -c
'{"function":"solicitarRegistrarCancelacionAfijo","Args":["{\"IDAfijo\":50}",
"{\"IDPersona\":6}"]}' -o $ORDENER_URL --tls --cafile $CA_FILE -C $CHANNEL_NAME

peer chaincode invoke -n solicitudes -c
'{"function":"solicitarRegistrarCambioPropietarioAfijo","Args":["{\"IDAfijo\":3,\"
Propietarios\":[{\"IDPersona\":301},{\"IDPersona\":302}]}",
"{\"IDPersona\":123}"]}' -o $ORDENER_URL --tls --cafile $CA_FILE -C $CHANNEL_NAME


# ERRORES SOLICITUDES


peer chaincode invoke -n solicitudes -c
'{"function":"validarSolicitud","Args":["{\"IDSolicitud\":2,\"EstadoSolicitud\":\"
APROBADO\"}", "{\"IDPersona\":555}"]}' -o $ORDENER_URL --tls --cafile $CA_FILE -C
$CHANNEL_NAME
```

```
peer chaincode invoke -n solicitudes -c
'{"function":"validarSolicitud","Args":["{\"IDSolicitud\":2,\"EstadoSolicitud\":\"
APROBADO\"}", "{\"IDPersona\":5}"]}' -o $ORDENER_URL --tls --cafile $CA_FILE -C
$CHANNEL_NAME

peer chaincode invoke -n $CC_NOMBRE_SOLICITUDES -c
'{"function":"solicitarRegistrarCambioPropietarioPerro","Args":["{\"IDPerro\":1234
56,\"Propietarios\":[{\"IDPersona\":1},{\"IDPersona\":2}]}",
"{\"IDPersona\":1}"]}' -o $ORDENER_URL --tls --cafile $CA_FILE -C $CHANNEL_NAME

peer chaincode invoke -n $CC_NOMBRE_SOLICITUDES -c
'{"function":"solicitarRegistrarCambioPropietarioPerro","Args":["{\"IDPerro\":1234
56,\"Propietarios\":[{\"IDPersona\":1765},{\"IDPersona\":23456}]}",
"{\"IDPersona\":1234}"]}' -o $ORDENER_URL --tls --cafile $CA_FILE -C $CHANNEL_NAME

peer chaincode invoke -n solicitudes -c
'{"function":"validarSolicitud","Args":["{\"IDSolicitud\":7,\"EstadoSolicitud\":\"
APROBADO\"}", "{\"IDPersona\":259}"]}' -o $ORDENER_URL --tls --cafile $CA_FILE -C
$CHANNEL_NAME


# CARGAS INICIALES


peer chaincode invoke -n solicitudes-c
'{"function":"cargarDatosIniciales_Propietarios","Args":["./json/solicitudes.json"
]}' -o $ORDENER_URL --tls --cafile $CA_FILE -C $CHANNEL_NAME

peer chaincode invoke -n solicitudes-c
'{"function":"cargarDatosIniciales_Propietarios","Args":["./json/autorizaciones.js
on"]}' -o $ORDENER_URL --tls --cafile $CA_FILE -C $CHANNEL_NAME
```

MICROCHIP

Registro de identificación de microchip subcutáneo insertado por los veterinarios que contiene un transpondedor con un código único que permite identificar de manera univoca al ejemplar.

Funciones que incorpora el contrato inteligente:

registrarMicrochips

Introduce en el sistema el número o código de microchip insertado subcutáneamente al ejemplar por un veterinario

- **Argumentos entrada:**

```
type MicrochipsPerros struct {
        IDPerro              int    `json:"IDPerro"`
        IDPersonaVeterinario int    `json:"IDPersonaVeterinario"`
        CODMicrochip         string `json:"CODMicrochip"`
}
```

```
type TipoSeguridad struct {
    IDPersona int `json:"IDPersona"`
}
```

El contrato realiza las siguientes validaciones:

- El número de argumentos de entrada sea 2
- El formato JSON de args[0] y args[1] sea valido
- Los argumentos Nombre y Propietarios tenga un valor.
- El IDPerro tenga un valor y corresponda a un ejemplar vivo.
- El IDPersonaVeterinario tenga un valor y corresponda a una persona con un perfil de Veterinario activo.
- El CODMicrochip tenga un valor
- La persona que invoca la acción está registrada y dispone de los permisos para realizar dicha acción

- **Argumentos salida:**

Devuelve los registros insertados:

```
type MicrochipsPerros struct {
        ObjectType           string `json:"docType"`
        IDMicrochipPerro     int    `json:"IDMicrochipPerro"`
        IDPerro              int    `json:"IDPerro"`
        IDPersonaVeterinario int    `json:"IDPersonaVeterinario"`
        CODMicrochip         string `json:"CODMicrochip"`
        FechaAlta            string `json:"FechaAlta"`
        FechaBaja            string `json:"FechaBaja"`
}
```

consultarMicrochip

Consultar los datos del microchip y del ejemplar. La información presentada variara dependiendo de la persona que realiza la consulta.

cargarDatosIniciales

Introduce en el sistema el historial de inserciones de microchips que han tenido los diferentes ejemplares, definidos en un fichero de texto con el siguiente formato JSON:

```
type MicrochipsPerros struct {
        ObjectType           string `json:"docType"`
        IDMicrochipPerro     int    `json:"IDMicrochipPerro"`
        IDPerro              int    `json:"IDPerro"`
        IDPersonaVeterinario int    `json:"IDPersonaVeterinario"`
```

```
        CODMicrochip        string `json:"CODMicrochip"`
        FechaAlta           string `json:"FechaAlta"`
        FechaBaja           string `json:"FechaBaja"`
}
```

- **Argumentos entrada:**

```
NombreFichero string
```

- **Argumentos salida:**

Devuelve el número de registros insertados:

```
NumeroRegistros string
```

Funciones comunes

Funciones de los contratos del sistema incorporadas a este contrato:

- **getQueryResultForQueryString**
- **asignarEstado**
- **borrarEstado**
- **consultarEstado**
- **consultarRangoEstados**
- **ejecutarConsulta**

Plan de pruebas

A continuación, se especifican una serie de comando que permite al lector realizar una batería de pruebas sobre el contrato inteligente

```
export CHANNEL_NAME=netcanchannel

export
CA_FILE=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/ordererOrganizat
ions/netcan.com/orderers/orderer.netcan.com/msp/tlscacerts/tlsca.netcan.com-
cert.pem

export ORDENER_URL=orderer.netcan.com:7050


# MICROCHIPS

peer chaincode invoke -n $CC_NOMBRE_MICROCHIPS -c
'{"function":"registrarMicrochipPerro","Args":["{\"IDPerro\":200,\"IDPersonaVeteri
nario\":100,\"CODMicrochip\":\"123456789012345\"}", "{\"IDPersona\":6}"]}' -o
$ORDENER_URL --tls --cafile $CA_FILE -C $CHANNEL_NAME


# ERRORES MICROCHIPS

peer chaincode invoke -n $CC_NOMBRE_MICROCHIPS -c
'{"function":"registrarMicrochipPerro","Args":["{\"IDPerro\":200,\"IDPersonaVeteri
nario\":100000000000,\"CODMicrochip\":\"123456789012345\"}",
"{\"IDPersona\":6}"]}' -o $ORDENER_URL --tls --cafile $CA_FILE -C $CHANNEL_NAME

peer chaincode invoke -n $CC_NOMBRE_MICROCHIPS -c
'{"function":"registrarMicrochipPerro","Args":["{\"IDPerro\":200,\"IDPersonaVeteri
nario\":100,\"CODMicrochip\":\"123456789012345\"}", "{\"IDPersona\":6}"]}' -o
$ORDENER_URL --tls --cafile $CA_FILE -C $CHANNEL_NAME


# CARGAS INICIALES

peer chaincode invoke -n microchips -c
'{"function":"cargarDatosIniciales","Args":["./json/microchips_perros_001.json"]}'
-o $ORDENER_URL --tls --cafile $CA_FILE -C $CHANNEL_NAME

peer chaincode invoke -n microchips -c
'{"function":"cargarDatosIniciales","Args":["./json/microchips_perros_002.json"]}'
-o $ORDENER_URL --tls --cafile $CA_FILE -C $CHANNEL_NAME

peer chaincode invoke -n microchips -c
'{"function":"cargarDatosIniciales","Args":["./json/microchips_perros_003.json"]}'
-o $ORDENER_URL --tls --cafile $CA_FILE -C $CHANNEL_NAME

peer chaincode invoke -n microchips -c
'{"function":"cargarDatosIniciales","Args":["./json/microchips_perros_004.json"]}'
-o $ORDENER_URL --tls --cafile $CA_FILE -C $CHANNEL_NAME

peer chaincode invoke -n microchips -c
'{"function":"cargarDatosIniciales","Args":["./json/microchips_perros_005.json"]}'
-o $ORDENER_URL --tls --cafile $CA_FILE -C $CHANNEL_NAME

peer chaincode invoke -n microchips -c
'{"function":"asignarEstado","Args":["MICROCHIPS_PERROS",
"{\"docType\":\"CONTADOR\",\"IDMaximo\":5759}"]}' -o $ORDENER_URL --tls --cafile
$CA_FILE -C $CHANNEL_NAME
```

VACUNAS

Registro de vacunas administradas por veterinarios.

Los nombres de las vacunas suelen variar según el fabricante y aunque los nombres puedan ser similares (Pentavalente, Hexavalente u Octovalente) su contenido puede variar su contenido con vacunas contra diversas enfermedades tales como, por ejemplo:

- el moquillo canino,
- hepatitis infecciosa
- la leptospirosis
- el parvovirus
- el coronaviru
- la rabia
- la parainfluenza
- la Bordetella bronchisepticala
- la borreliosis o enfermedad de Lyme
- la herpesvirus canino
- la leishmaniasis

Funciones que incorpora el contrato inteligente:

registrarVacuna

Registrar en el sistema el tipo de vacuna administrada al ejemplar, su identificador medicinal, el identificador del colegiado del veterinario, la fecha de administración y la duración de la dosis.

Introduce en el sistema el número o código de microchip insertado subcutáneamente al ejemplar por un veterinario

- **Argumentos entrada:**

```
type tipoRegistrarVacunaPerroPropietario struct {
       IDVacunaProteccion        int    `json:"IDProteccion"`
        FechaBaja                string `json:"FechaBaja"`
    }

    type tipoRegistrarVacunaPerro struct {
        IDPerro             int                      `json:"IDPerro"`
        IDPersonaVeterinario int                     `json:"IDPersonaVeterinario"`
        CODVacuna       string                   `json:"CODVacuna"`
        FechaAlta       string                   `json:"FechaAlta"`
        FechaBaja       string                   `json:"FechaBaja"`
        Protecciones    []tipoRegistrarVacunaPerroPropietario `json:"Protecciones"
`
    }
```

```
type TipoSeguridad struct {
    IDPersona int `json:"IDPersona"`
}
```

El contrato realiza las siguientes validaciones:

- El número de argumentos de entrada sea 2
- El formato JSON de args[0] y args[1] sea valido
- Los argumentos CODVacuna, FechaAlta y Protecciones tenga un valor.
- El IDPerro tenga un valor y corresponda a un ejemplar vivo.
- El IDPersonaVeterinario tenga un valor y corresponda a una persona con un perfil de Veterinario activo.
- La persona que invoca la acción está registrada y dispone de los permisos para realizar dicha acción

- **Argumentos salida:**

Devuelve los registros insertados:

```
type MicrochipsPerros struct {
        ObjectType           string `json:"docType"`
        IDMicrochipPerro     int    `json:"IDMicrochipPerro"`
        IDPerro              int    `json:"IDPerro"`
        IDPersonaVeterinario int    `json:"IDPersonaVeterinario"`
        CODMicrochip         string `json:"CODMicrochip"`
        FechaAlta            string `json:"FechaAlta"`
        FechaBaja            string `json:"FechaBaja"`
}
```

obtenerCertificadoVacunaciones (pasaporte)

Obtener un fichero digital que contiene los datos del ejemplar y su cartilla de vacunación.

consultarVacunaciones

Consultar la cartilla de vacunación del ejemplar. La información presentada variara dependiendo de la persona que realiza la consulta.

cargarDatosIniciales

Introduce en el sistema el historial de vacunaciones que han tenido los diferentes ejemplares, definidos en un fichero de texto con el siguiente formato JSON:

```
type VacunasPerros struct {
        ObjectType           string `json:"docType"`
        IDVacunaPerro        int    `json:"IDVacunaPerro"`
        IDPerro              int    `json:"IDPerro"`
        IDPersonaVeterinario int    `json:"IDPersonaVeterinario"`
        CODVacuna            string `json:"CODVacuna"`
        FechaAlta            string `json:"FechaAlta"`
        FechaBaja            string `json:"FechaBaja"`
}
```

- **Argumentos entrada:**

```
NombreFichero string
```

- **Argumentos salida:**

Devuelve el número de registros insertados:

```
NumeroRegistros string
```

cargarDatosIniciales_VacunasPerrosProteccion

Introduce en el sistema el historial de vacunaciones que han tenido los diferentes ejemplares, definidos en un fichero de texto con el siguiente formato JSON:

```
type VacunasPerrosProteccion struct {
        ObjectType              string `json:"docType"`
        IDVacunaPerroProteccion int    `json:"IDVacunaPerroProteccion"`
        IDVacunaPerro           int    `json:"IDVacunaPerro"`
        IDVacunaProteccion      int    `json:"IDVacunaProteccion"`
        FechaAlta               string `json:"FechaAlta"`
        FechaBaja               string `json:"FechaBaja"`
}
```

- **Argumentos entrada:**

```
NombreFichero string
```

- **Argumentos salida:**

Devuelve el número de registros insertados:

```
NumeroRegistros string
```

cargarDatosIniciales_VacunasProteccion

Introduce en el sistema el repositorio de tipos de vacunas más comunes, que determinan el tipo de protección que contiene cada vacuna, que han tenido los diferentes ejemplares, definidos en un fichero de texto con el siguiente formato JSON:

```
type VacunasProteccion struct {
        ObjectType         string `json:"docType"`
        IDVacunaProteccion int    `json:"IDVacunaProteccion"`
        Nombre             string `json:"Nombre"`
        FechaAlta          string `json:"FechaAlta"`
        FechaBaja          string `json:"FechaBaja"`
}
```

- **Argumentos entrada:**

```
NombreFichero string
```

- **Argumentos salida:**

Devuelve el número de registros insertados:

```
NumeroRegistros string
```

Funciones comunes

Funciones de los contratos del sistema incorporadas a este contrato:

- **getQueryResultForQueryString**
- **asignarEstado**
- **borrarEstado**
- **consultarEstado**
- **consultarRangoEstados**
- **ejecutarConsulta**

Plan de pruebas

A continuación, se especifican una serie de comando que permite al lector realizar una batería de pruebas sobre el contrato inteligente

```
export CHANNEL_NAME=netcanchannel

export
CA_FILE=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/ordererOrganizat
ions/netcan.com/orderers/orderer.netcan.com/msp/tlscacerts/tlsca.netcan.com-
cert.pem

export ORDENER_URL=orderer.netcan.com:7050


# VACUNAS

peer chaincode invoke -n vacunas -c '{"function":"registrarVacunaPerro", "Args":["
{\"IDPerro\":6, \"IDPersonaVeterinario\":106, \"CODVacuna\":\"123456789012345\",
\"FechaBaja\":\"2022-02-02\", \"Protecciones\":[
{\"IDVacunaProteccion\":1,\"FechaBaja\":\"2022-02-
02\"},{\"IDVacunaProteccion\":2,\"FechaBaja\":\"2022-02-02\"} ]}",
"{\"IDPersona\":6}"]}' -o $ORDENER_URL --tls --cafile $CA_FILE -C $CHANNEL_NAME


# ERRORES VACUNAS

peer chaincode invoke -n vacunas -c '{"function":"registrarVacunaPerro", "Args":["
{\"IDPerro\":6, \"IDPersonaVeterinario\":106, \"CODVacuna\":\"123456789012345\",
\"FechaBaja\":\"2022-02-02\", \"Protecciones\":[
{\"IDVacunaProteccion\":1,\"FechaBaja\":\"2022-02-
02\"},{\"IDVacunaProteccion\":2,\"FechaBaja\":\"2022-02-02\"} ]}",
"{\"IDPersona\":6}"]}' -o $ORDENER_URL --tls --cafile $CA_FILE -C $CHANNEL_NAME

peer chaincode invoke -n vacunas -c '{"function":"registrarVacunaPerro", "Args":["
{\"IDPerro\":6, \"IDPersonaVeterinario\":106, \"CODVacuna\":\"987654321012345\",
\"FechaBaja\":\"2022-02-02\", \"Protecciones\":[
{\"IDVacunaProteccion\":1,\"FechaBaja\":\"2022-02-
02\"},{\"IDVacunaProteccion\":230,\"FechaBaja\":\"2022-02-02\"} ]}",
"{\"IDPersona\":6}"]}' -o $ORDENER_URL --tls --cafile $CA_FILE -C $CHANNEL_NAME
```

```
# CARGAS INICIALES

peer chaincode invoke -n vacunas -c
'{"function":"cargarDatosIniciales_VacunasProteccion","Args":["./json/vacunas_prot
eccion_001.json"]}' -o $ORDENER_URL --tls --cafile $CA_FILE -C $CHANNEL_NAME

sleep 7s

peer chaincode invoke -n vacunas -c
'{"function":"asignarEstado","Args":["VACUNAS_PERROS_PROTECCION",
"{\"docType\":\"CONTADOR\",\"IDMaximo\":512}"]}' -o $ORDENER_URL --tls --cafile
$CA_FILE -C $CHANNEL_NAME

peer chaincode invoke -n vacunas -c
'{"function":"cargarDatosIniciales","Args":["./json/vacunas_perros_001.json"]}' -o
$ORDENER_URL --tls --cafile $CA_FILE -C $CHANNEL_NAME

sleep 7s

peer chaincode invoke -n vacunas -c
'{"function":"asignarEstado","Args":["VACUNAS_PROTECCION",
"{\"docType\":\"CONTADOR\",\"IDMaximo\":11}"]}' -o $ORDENER_URL --tls --cafile
$CA_FILE -C $CHANNEL_NAME

peer chaincode invoke -n vacunas -c
'{"function":"cargarDatosIniciales_VacunasPerrosProteccion","Args":["./json/vacuna
s_perros_proteccion_001.json"]}' -o $ORDENER_URL --tls --cafile $CA_FILE -C
$CHANNEL_NAME

sleep 7s

peer chaincode invoke -n vacunas -c
'{"function":"asignarEstado","Args":["VACUNAS_PERROS",
"{\"docType\":\"CONTADOR\",\"IDMaximo\":89}"]}' -o $ORDENER_URL --tls --cafile
$CA_FILE -C $CHANNEL_NAME
```

PERFILES

Contrato que gestiona el mantenimiento de los diferentes roles especiales de usuario:

- Administradores
- Veterinarios
- Trabajadores de las federaciones caninas

La asignación de los perfiles puede ser consultado desde cualquier contrato, pero solo podrán ser modificados por administradores.

Funciones que incorpora el contrato inteligente:

registrarPerfilPersona

Inserta en el sistema la definición de un nuevo perfil para un usuario.

- **Argumentos entrada:**

```
type PerfilesPersonasInsertar struct {
    IDPersona       int    `json:"IDPersona"`
    CODPerfil       string `json:"CODPerfil"`
}
```

```
type TipoSeguridad struct {
    IDPersona int `json:"IDPersona"`
}
```

El contrato realiza las siguientes validaciones:

- El número de argumentos de entrada sea 2
- El formato JSON de args[0] y args[1] sea valido
- El IDPersona este registrado.
- El CODPerfil tenga un valor
- La persona que invoca la acción está registrada y dispone de los permisos para realizar dicha acción (Perfil administrador)

- **Argumentos salida:**

Devuelve los registros insertados:

```
type PerfilesPersonas struct {
        ObjectType      string `json:"docType"`
        IDPerfilPersona int    `json:"IDPerfilPersona"`
        IDPersona       int    `json:"IDPersona"`
        CODPerfil       string `json:"CODPerfil"`
        FechaAlta       string `json:"FechaAlta"`
        FechaBaja       string `json:"FechaBaja"`
}
```

cancelarPerfilPersona

Cancela en el sistema la definición de un perfil existente para un usuario.

- **Argumentos entrada:**

```
type PerfilesPersonasCancelar struct {
    IDPersona       int    `json:"IDPersona"`
    CODPerfil       string `json:"CODPerfil"`
}
```

```
type TipoSeguridad struct {
    IDPersona int `json:"IDPersona"`
}
```

El contrato realiza las siguientes validaciones:

- El número de argumentos de entrada sea 2
- El formato JSON de args[0] y args[1] sea valido
- El IDPersona este registrado.
- El CODPerfil tenga un valor
- El IDPersona y CODPerfil tenga definido un perfil activo
- La persona que invoca la acción está registrada y dispone de los permisos para realizar dicha acción (Perfil administrador)

- **Argumentos salida:**

Devuelve los registros insertados:

```
type PerfilesPersonas struct {
        ObjectType      string `json:"docType"`
        IDPerfilPersona int    `json:"IDPerfilPersona"`
        IDPersona       int    `json:"IDPersona"`
        CODPerfil       string `json:"CODPerfil"`
        FechaAlta       string `json:"FechaAlta"`
        FechaBaja       string `json:"FechaBaja"`
}
```

cargarDatosIniciales

Introduce en el sistema el historial de perfiles que han tenido los diferentes usuarios, definidos en un fichero de texto con el siguiente formato JSON:

```
type PerfilesPersonas struct {
        ObjectType      string `json:"docType"`
        IDPerfilPersona int    `json:"IDPerfilPersona"`
        IDPersona       int    `json:"IDPersona"`
        CODPerfil       string `json:"CODPerfil"`
        FechaAlta       string `json:"FechaAlta"`
        FechaBaja       string `json:"FechaBaja"`
}
```

- **Argumentos entrada:**

```
NombreFichero string
```

```
type TipoSeguridad struct {
    IDPersona int `json:"IDPersona"`
}
```

El contrato realiza las siguientes validaciones:

- El número de argumentos de entrada sea 2
- El formato JSON args[1] sea valido
- La persona que invoca la acción está registrada y dispone de los permisos para realizar dicha acción (Perfil administrador)

- **Argumentos salida:**

Devuelve el número de registros insertados:

```
NumeroRegistros string
```

Funciones complementarias

- **asignarEstado_OnlyAdmin**

  Permite a un usuario administrador modificar un estado de la cadena de bloques.

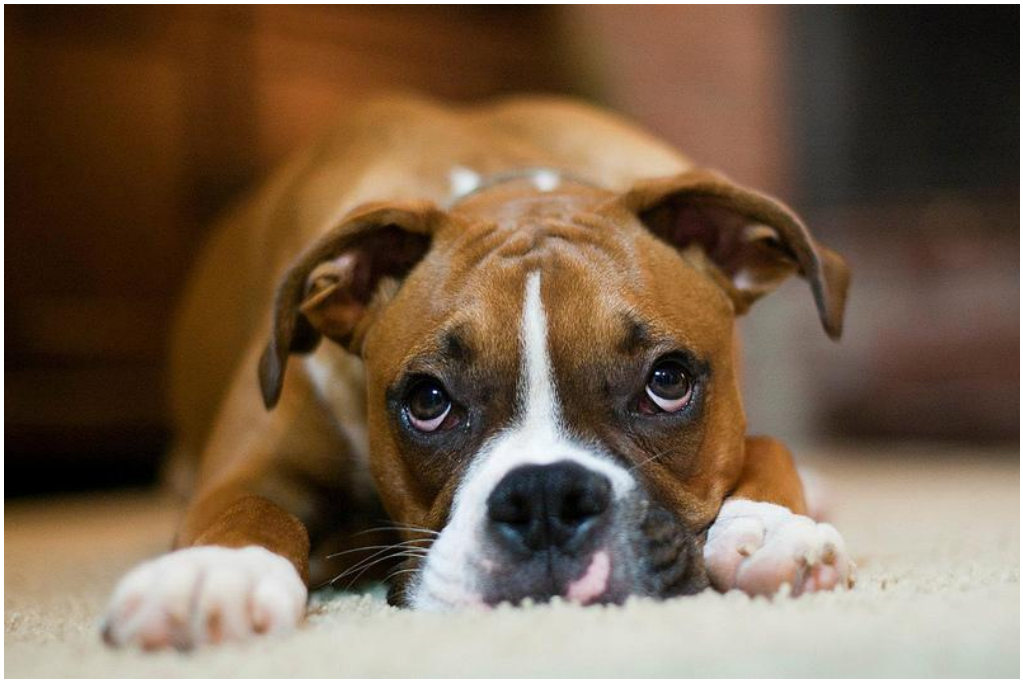
- **borrarEstado_OnlyAdmin**
  Permite a un usuario administrador modificar un estado de la cadena de bloques

- **esAdministrador**
  Indica si la persona consultada dispone de un perfil de Administrador activo.

- **esMiembroFederacion**
  Indica si la persona consultada dispone de un perfil de miembro en una Federación Canina activo.

- **esVeterinario**
  Indica si la persona consultada dispone de un perfil de Veterinario activo.

- **esPerfilPersona**
  Indica si la persona consultada dispone de un perfil especifico activo.



Funciones comunes

Funciones de los contratos del sistema incorporadas a este contrato:

- **getQueryResultForQueryString**
- **consultarEstado**
- **consultarRangoEstados**
- **ejecutarConsulta**

Plan de pruebas

A continuación, se especifican una serie de comando que permite al lector realizar una batería de pruebas sobre el contrato inteligente

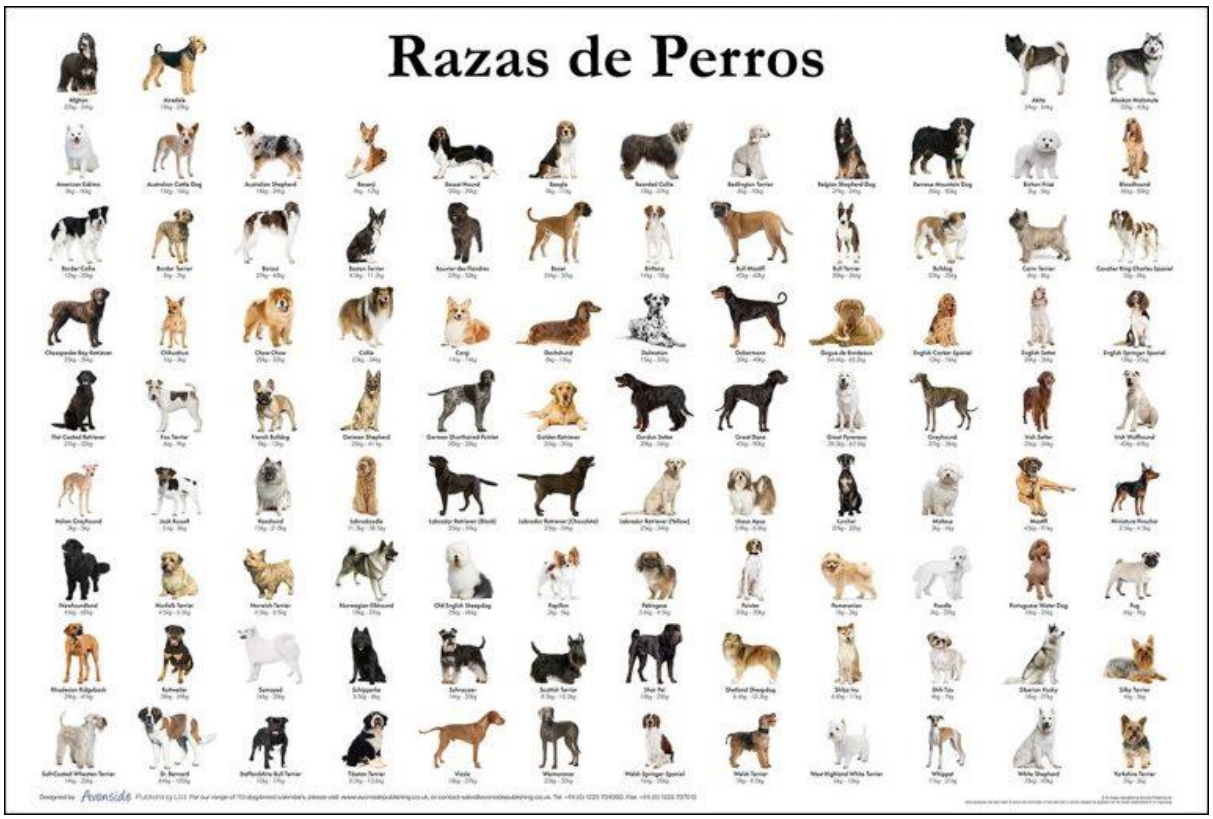
```
export CHANNEL_NAME=netcanchannel

export
CA_FILE=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/ordererOrganizat
ions/netcan.com/orderers/orderer.netcan.com/msp/tlscacerts/tlsca.netcan.com-
cert.pem

export ORDENER_URL=orderer.netcan.com:7050


# PERFILES

peer chaincode invoke -n $CC_NOMBRE_PERFILES -c
'{"function":"registrarPerfilPersona","Args":["{\"IDPersona\":100,\"CODPerfil\":\"
ADMINISTRADOR\"}", "{\"IDPersona\":6}"]}' -o $ORDENER_URL --tls --cafile $CA_FILE
-C $CHANNEL_NAME

peer chaincode invoke -n $CC_NOMBRE_PERFILES -c
'{"function":"cancelarPerfilPersona","Args":["{\"IDPersona\":100,\"CODPerfil\":\"A
DMINISTRADOR\"}", "{\"IDPersona\":100}"]}' -o $ORDENER_URL --tls --cafile $CA_FILE
-C $CHANNEL_NAME

# ERRORES PERFILES

peer chaincode invoke -n perfiles -c
'{"function":"cargarDatosIniciales","Args":["./json/perfiles.json",
"{\"IDPersona\":123456}"]}' -o $ORDENER_URL --tls --cafile $CA_FILE -C
$CHANNEL_NAME

peer chaincode invoke -n $CC_NOMBRE_PERFILES -c
'{"function":"cancelarPerfilPersona","Args":["{\"IDPersona\":100,\"CODPerfil\":\"A
DMINISTRADOR\"}", "{\"IDPersona\":6}"]}' -o $ORDENER_URL --tls --cafile $CA_FILE -
C $CHANNEL_NAME

peer chaincode invoke -n $CC_NOMBRE_PERFILES -c
'{"function":"cancelarPerfilPersona","Args":["{\"IDPersona\":666,\"CODPerfil\":\"A
DMINISTRADOR\"}", "{\"IDPersona\":0}"]}' -o $ORDENER_URL --tls --cafile $CA_FILE -
C $CHANNEL_NAME

# CARGAS INICIALES

peer chaincode invoke -n perfiles -c
'{"function":"cargarDatosIniciales","Args":["./json/perfiles.json",
"{\"IDPersona\":0}"]}' -o $ORDENER_URL --tls --cafile $CA_FILE -C $CHANNEL_NAME

peer chaincode invoke -n perfiles -c
'{"function":"asignarEstado_OnlyAdmin","Args":["PERFILES_PERSONAS",
"{\"docType\":\"CONTADOR\",\"IDMaximo\":48}", "{\"IDPersona\":0}"]}' -o
$ORDENER_URL --tls --cafile $CA_FILE -C $CHANNEL_NAME

peer chaincode invoke -n perfiles -c
'{"function":"cancelarPerfilPersona","Args":["{\"IDPersona\":0,\"CODPerfil\":\"ADM
INISTRADOR\"}", "{\"IDPersona\":0}"]}' -o $ORDENER_URL --tls --cafile $CA_FILE -C
$CHANNEL_NAME
```

RAZAS

Contrato que gestiona el mantenimiento de los diferentes estándares de pureza de raza reconocidos, actualmente cerca de 346 razas.

Se clasifican en los siguientes grupos:

- **Grupo 1**: Perros de pastor y perros boyeros (excepto perros boyeros suizos)
- **Grupo 2**: Perros tipo pinscher y schnauzer, molosoides y perros tipo montaña y boyeros suizos
- **Grupo 3**: Terriers
- **Grupo 4**: Teckels
- **Grupo 5**: Tipo spitz y tipo primitivo
- **Grupo 6**: Perros tipo sabueso, perros de rastro (exceptuando lebreles) y razas semejantes.
- **Grupo 7**: Perros de muestra
- **Grupo 8**: Perros cobradores de caza - perros levantadores de caza - perros de agua
- **Grupo 9**: Perros de compañía
- **Grupo 10**: Lebreles



- **Razas Provisionales:** Razas provisionalmente aceptadas
- **Mestizos:** de padres desconocidos, diferente raza o cuyo origen de especie no esté validado.

La definición de las razas y grupos puede ser consultado desde cualquier contrato, pero solo podrán ser modificados por miembros de federaciones caninas.

Funciones comunes

Funciones de los contratos del sistema incorporadas a este contrato:

- **getQueryResultForQueryString**
- **asignarEstado**
- **borrarEstado**
- **consultarEstado**
- **consultarRangoEstados**
- **ejecutarConsulta**

Plan de pruebas

A continuación, se especifican una serie de comando que permite al lector realizar una batería de pruebas sobre el contrato inteligente

```
export CHANNEL_NAME=netcanchannel
```

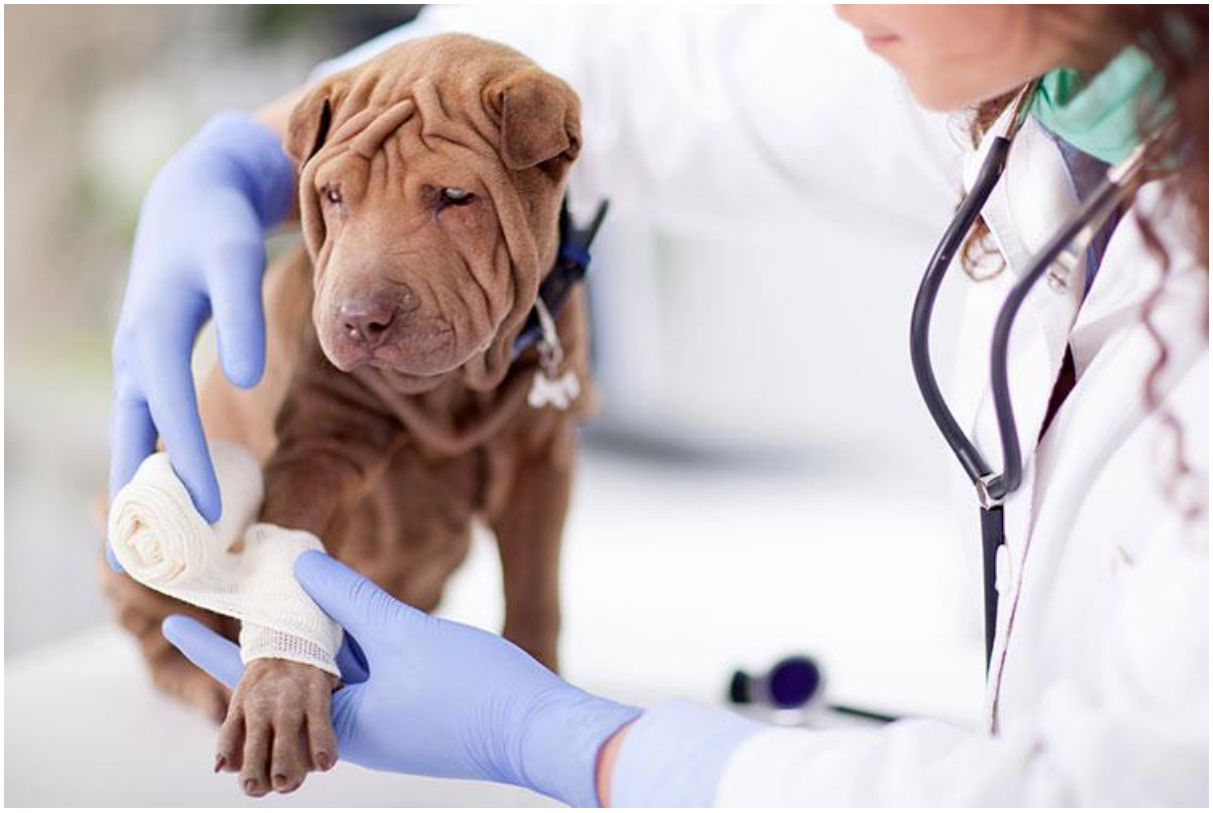
```
export
CA_FILE=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/ordererOrganizat
ions/netcan.com/orderers/orderer.netcan.com/msp/tlscacerts/tlsca.netcan.com-
cert.pem

export ORDENER_URL=orderer.netcan.com:7050


# CARGAS INICIALES


peer chaincode invoke -n razas -c
'{"function":"cargarDatosIniciales_Grupos","Args":["./json/grupos.json"]}' -o
$ORDENER_URL --tls --cafile $CA_FILE -C $CHANNEL_NAME


peer chaincode invoke -n razas -c '{"function":"asignarEstado","Args":["GRUPOS",
"{\"docType\":\"CONTADOR\",\"IDMaximo\":12}"]}' -o $ORDENER_URL --tls --cafile
$CA_FILE -C $CHANNEL_NAME


peer chaincode invoke -n razas -c
'{"function":"cargarDatosIniciales","Args":["./json/razas.json"]}' -o $ORDENER_URL
--tls --cafile $CA_FILE -C $CHANNEL_NAME


peer chaincode invoke -n razas -c '{"function":"asignarEstado","Args":["RAZAS",
"{\"docType\":\"CONTADOR\",\"IDMaximo\":346}"]}' -o $ORDENER_URL --tls --cafile
$CA_FILE -C $CHANNEL_NAME
```

VETERINARIOS

Contrato que gestiona los datos específicos de un veterinario.

Para la prueba de concepto se ha simplificado los datos a gestionar reduciéndolos al número de colegiado.

En un caso real este contrato podrá contener toda la información que sea necesaria en relación con los veterinarios.



Funciones que incorpora el contrato inteligente:

registrarColegiaturaPersona

Inserta en el sistema el registro de una colegiatura de un usuario.

- **Argumentos entrada:**

```
type ColegiaturasPersonasRegistro struct {
    IDPersona            int    `json:"IDPersona"`
    CODColegiatura       string `json:"CODColegiatura"`
}
```

```
type TipoSeguridad struct {
    IDPersona int `json:"IDPersona"`
}
```

El contrato realiza las siguientes validaciones:

- El número de argumentos de entrada sea 2
- El formato JSON de args[0] y args[1] sea valido
- El IDPersona este registrado.
- El CODColegiatura tenga un valor y no este registrado ya en el sistema
- La persona que invoca la acción está registrada y dispone de los permisos para realizar dicha acción

- **Argumentos salida:**

Devuelve los registros insertados:

```
type ColegiaturasPersonas struct {
    ObjectType           string `json:"docType"`
    IDColegiaturaPersona int    `json:"IDColegiaturaPersona"`
    IDPersona            int    `json:"IDPersona"`
    CODColegiatura       string `json:"CODColegiatura"`
    FechaAlta            string `json:"FechaAlta"`
    FechaBaja            string `json:"FechaBaja"`
}
```

cancelarPerfilPersona

Cancela en el sistema la definición de una colegiatura existente para un usuario.

- **Argumentos entrada:**

```
type PerfilesPersonasCancelar struct {
    IDPersona       int    `json:"IDPersona"`
    CODPerfil       string `json:"CODPerfil"`
}
```

```
type TipoSeguridad struct {
    IDPersona int `json:"IDPersona"`
```

```
}
```

El contrato realiza las siguientes validaciones:

- El número de argumentos de entrada sea 2
- El formato JSON de args[0] y args[1] sea valido
- El IDPersona este registrado.
- El CODColegiatura tenga un valor
- El IDPersona y CODColegiatura tenga definido un perfil activo
- La persona que invoca la acción está registrada y dispone de los permisos para realizar dicha acción

- **Argumentos salida:**

Devuelve los registros insertados:

```
type ColegiaturasPersonas struct {
    ObjectType           string `json:"docType"`
    IDColegiaturaPersona int    `json:"IDColegiaturaPersona"`
    IDPersona            int    `json:"IDPersona"`
    CODColegiatura       string `json:"CODColegiatura"`
    FechaAlta            string `json:"FechaAlta"`
    FechaBaja            string `json:"FechaBaja"`
}
```

cargarDatosIniciales

Introduce en el sistema el historial de colegiaturas que han tenido los diferentes usuarios, definidos en un fichero de texto con el siguiente formato JSON:

```
type ColegiaturasPersonas struct {
    ObjectType           string `json:"docType"`
    IDColegiaturaPersona int    `json:"IDColegiaturaPersona"`
    IDPersona            int    `json:"IDPersona"`
    CODColegiatura       string `json:"CODColegiatura"`
    FechaAlta            string `json:"FechaAlta"`
    FechaBaja            string `json:"FechaBaja"`
}
```

- **Argumentos entrada:**

```
NombreFichero string
```

- **Argumentos salida:**

Devuelve el número de registros insertados:

```
NumeroRegistros string
```

Funciones comunes

Funciones de los contratos del sistema incorporadas a este contrato:

- **getQueryResultForQueryString**
- **asignarEstado**
- **borrarEstado**
- **consultarEstado**
- **consultarRangoEstados**
- **ejecutarConsulta**

Plan de pruebas

A continuación, se especifican una serie de comando que permite al lector realizar una batería de pruebas sobre el contrato inteligente

```
export CHANNEL_NAME=netcanchannel

export
CA_FILE=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/ordererOrganizat
```

```
ions/netcan.com/orderers/orderer.netcan.com/msp/tlscacerts/tlsca.netcan.com-
cert.pem

export ORDENER_URL=orderer.netcan.com:7050


# VETERINARIOS

peer chaincode invoke -n veterinarios -c
'{"function":"registrarColegiaturaPersona","Args":["{\"IDPersona\":1,\"CODColegiat
ura\":\"COVM-1234567890\"}", "{\"IDPersona\":6}"]}' -o $ORDENER_URL --tls --cafile
$CA_FILE -C $CHANNEL_NAME

peer chaincode invoke -n veterinarios -c
'{"function":"cancelarColegiaturaPersona","Args":["{\"IDPersona\":1,\"CODColegiatu
ra\":\"COVM-1234567890\"}", "{\"IDPersona\":6}"]}' -o $ORDENER_URL --tls --cafile
$CA_FILE -C $CHANNEL_NAME


# ERRORES VETERINARIOS

peer chaincode invoke -n veterinarios -c
'{"function":"registrarColegiaturaPersona","Args":["{\"IDPersona\":1,\"CODColegiat
ura\":\"COVM-1234567890\"}", "{\"IDPersona\":123}"]}' -o $ORDENER_URL --tls --
cafile $CA_FILE -C $CHANNEL_NAME

peer chaincode invoke -n veterinarios -c
'{"function":"registrarColegiaturaPersona","Args":["{\"IDPersCODColegiatura\":\"CO
VM-1234567890\"}", "{\"IDPersona\":6}"]}' -o $ORDENER_URL --tls --cafile $CA_FILE
-C $CHANNEL_NAME

peer chaincode invoke -n veterinarios -c
'{"function":"registrarColegiaturaPersona","Args":["{\"IDPersona\":10000000,\"CODC
olegiatura\":\"COVM-1234567890\"}", "{\"IDPersona\":6}"]}' -o $ORDENER_URL --tls -
-cafile $CA_FILE -C $CHANNEL_NAME

peer chaincode invoke -n veterinarios -c
'{"function":"cancelarColegiaturaPersona","Args":["{\"IDPersona\":1,\"CODColegiatu
ra\":\"COVM-NO-EXISTE\"}", "{\"IDPersona\":6}"]}' -o $ORDENER_URL --tls --cafile
$CA_FILE -C $CHANNEL_NAME

peer chaincode invoke -n veterinarios -c
'{"function":"registrarColegiaturaPersona","Args":["{\"IDPersona\":126,\"CODColegi
atura\":\"1234567890\"}", "{\"IDPersona\":6}"]}' -o $ORDENER_URL --tls --cafile
$CA_FILE -C $CHANNEL_NAME


# CARGAS INICIALES

peer chaincode invoke -n veterinarios -c
'{"function":"cargarDatosIniciales","Args":["./json/veterinarios.json"]}' -o
$ORDENER_URL --tls --cafile $CA_FILE -C $CHANNEL_NAME

leep 7s

peer chaincode invoke -n veterinarios -c
'{"function":"asignarEstado","Args":["VETERINARIOS_PERSONAS",
"{\"docType\":\"CONTADOR\",\"IDMaximo\":21}"]}' -o $ORDENER_URL --tls --cafile
$CA_FILE -C $CHANNEL_NAME
```

Otros posibles contratos inteligentes

A continuación, se especifican algunos contratos más que podrían incorporarse en el fututo a la Blockchain, a modo de ejemplo:

TITULOS

Registra los títulos concedidos por federaciones, asociaciones y clubes por los méritos obtenidos por los ejemplares en explosiones y/o concursos

solicitarTitulo

Registra la solicitud de un título a una organización en base a los resultados obtenidos en exposiciones y concursos. La validación se realiza de manera automática por el sistema.

obtenerTitulo

Obtener un fichero digital que certifica el título obtenido por el ejemplar.

consultarTitulos

Obtiene la relación de títulos que un ejemplar ha obtenido hasta ese momento.

EXPOSICIONES Y CONCURSOS

Registra los resultados de los concursos y exposiciones de Morfología, Trabajo y disciplina, Rastreo y Agility que las diferentes federaciones, asociaciones y club realizan.

registarShow

Registrar los datos y características de una exposición o concurso *(entidad organizadora, tipo de concurso, ámbito, fecha, lugar, ...)*

registarResultadoShow

Registrar el resultado que un ejemplar ha obtenido al participar en una exposición o concurso, tanto a nivel de calificación como de puesto.

cagarResultadosShow

Realizar la carga masiva de todos los resultados de una exposición, con el fin de registrar las calificaciones y puestos obtenidos por cada ejemplar.

consultarResultadosShow

Obtiene información de los resultados y calificaciones de exposiciones y concursos.

El ámbito del proyecto puede expandirse a multitud de gestiones del mundo canino, que hoy día no han podido ser abordadas por el momento de manera global.

Por ejemplo:

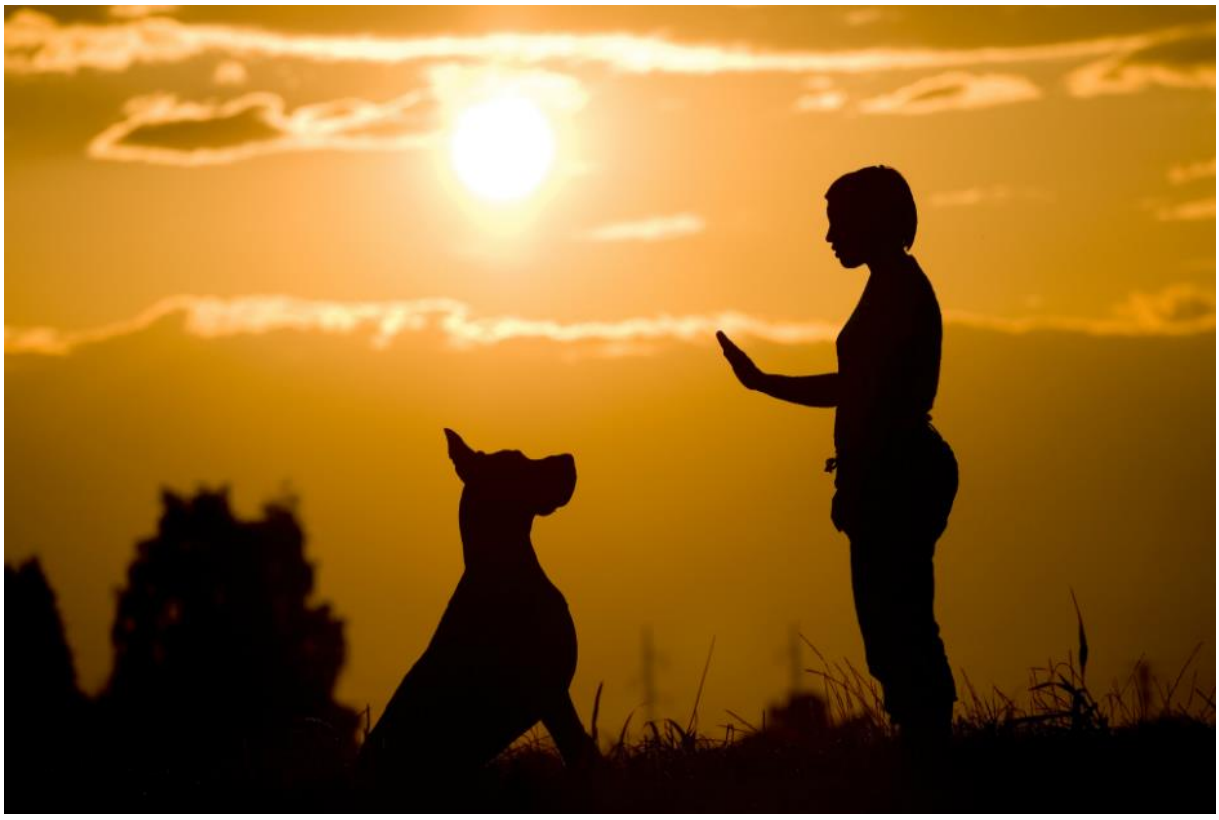
ADN

Contrato que gestiona el registro de ADN.

Facilitaría la certificación de autenticad de la linera genealógica y permitiría el estudio genético a partir de los datos obtenidos.

ENFERMEDADES / TRATAMIENTOS

Contrato que gestiona el registro de las enfermedades y tratamientos que los veterinarios realizan sobre los perros, pudiéndose disponerse de un historial clínico para que cualquier veterinario pudiera consultarlo cuando llegue un ejemplar a su consulta.



Carga inicial de datos en la Blockchain

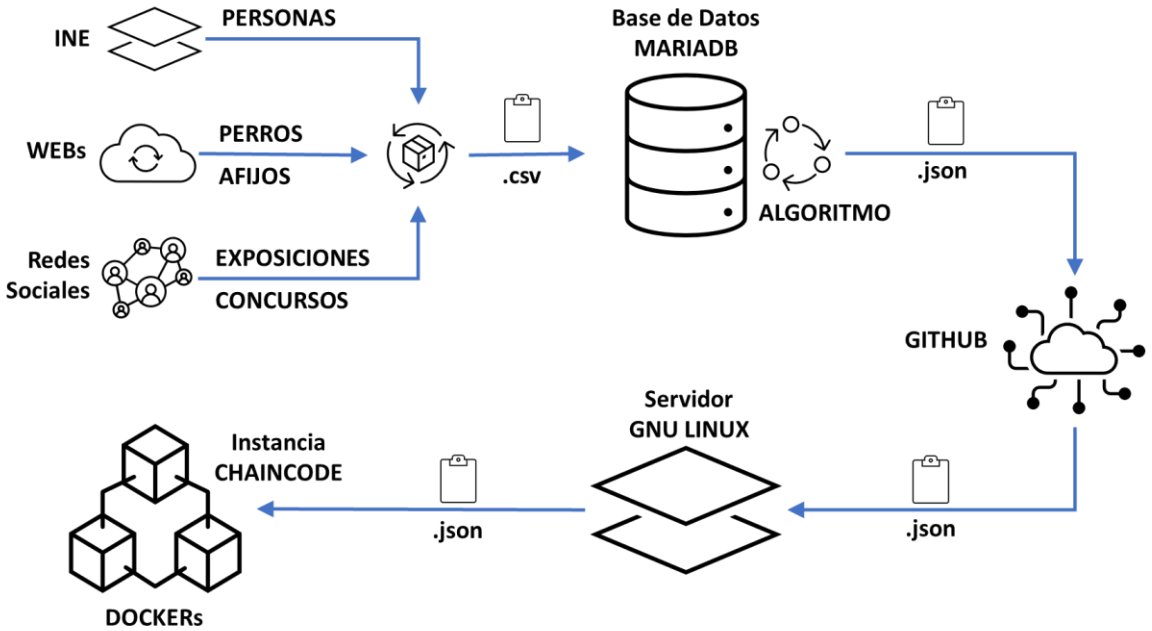
Introducción

A pesar de que hemos intentado que federaciones , asociaciones y club colaboren en el desarrollo del proyecto no hemos conseguido que nos proporcionan datos que sirvieran de base para cargar la cadena de bloques, más allá de lo que estas organizaciones presentaban en sus plataformas digitales, ya que consideran que los datos eran solo de uso interno.

Por tanto, ha sido necesaria la generación de una base de datos a partir de información que hemos podido extraer de organismos oficiales como el INE, y el depositado en paginas web y redes sociales.

Proceso de carga

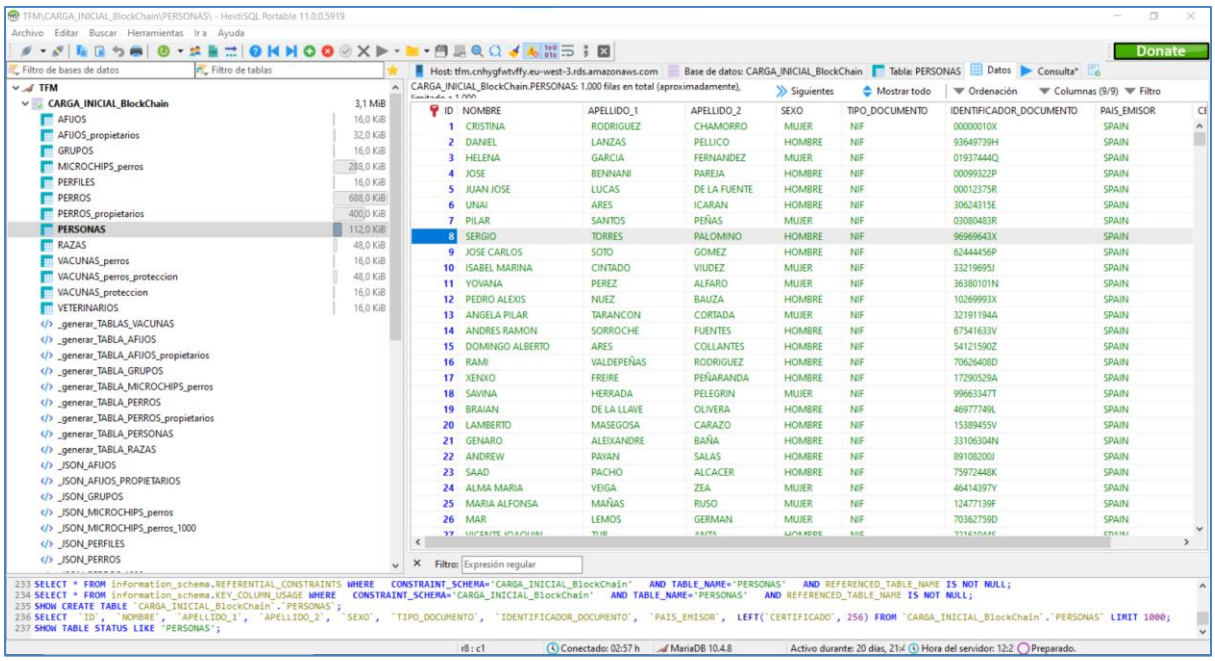
El proceso de carga sigue el siguiente esquema:



A través de las consultas al INE se obtiene los 5000 apellidos más comunes en España, así como los 5000 nombres de mujeres y 5000 nombres de hombres.

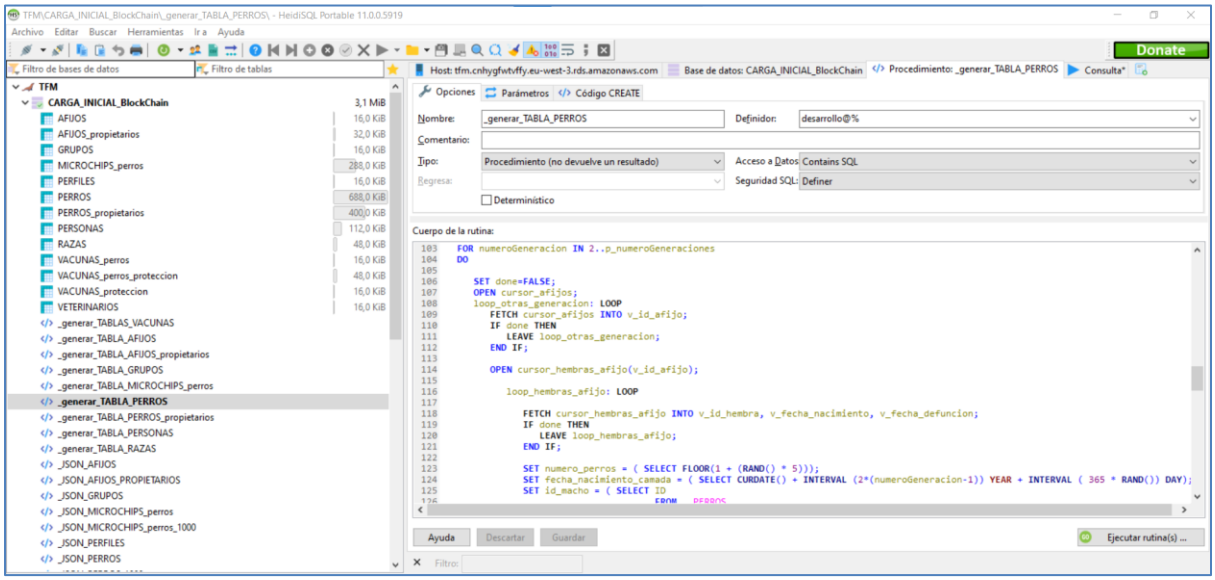
A través de un `proceso automático que los combina automáticamente, junto con un generador de NIF se obtiene la primera tabla con las PERSONAS que formaran parte de la organización en una Base de datos **MariaDB**.

En las pruebas iniciales conseguimos generar sin problema 100.000 usuarios, pero para la prueba de concepto reducimos a 1000 para no sobrecargar el tiempo de carga, aunque comprobamos que la blockchain lo podía soportar con los servidores que había cedido la universidad, siempre y cuando cargáramos los datos de 1000 en 1000.



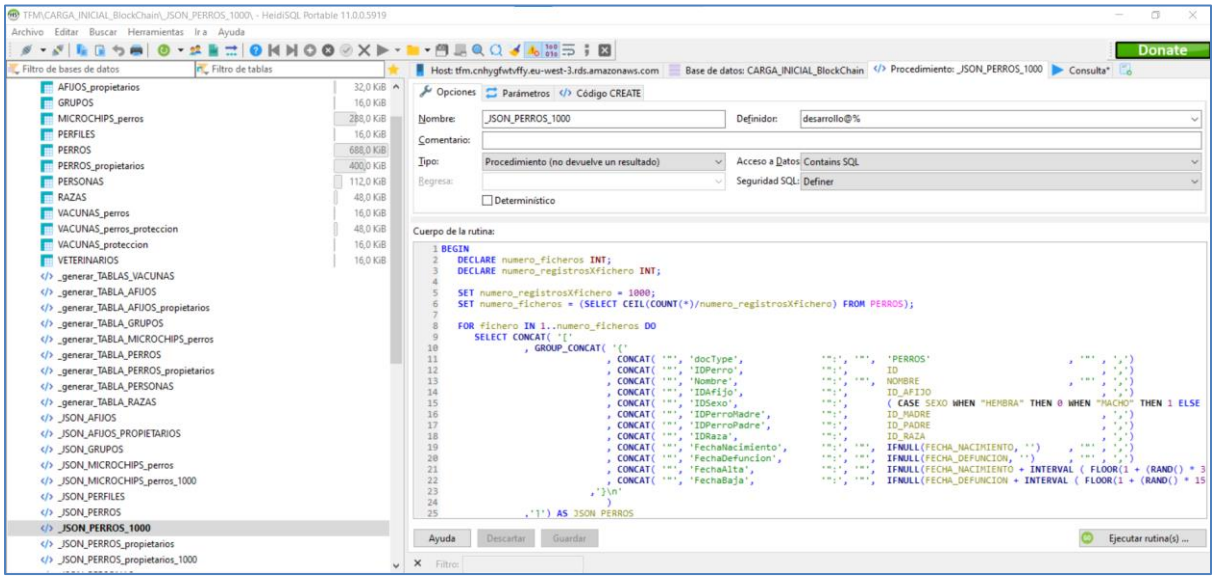
A través de algoritmos aleatorios se genera las restantes tablas:

- Los nombres de los Perros
- El afijo de criador al que pertenecen
- Los propietarios y los cambios de titularidad de los perros y de los afijos de criador
- La fecha de nacimiento y defunción
- La línea genealógica ascendente y descendente donde las fechas deben coincidir con la edad mínima y máxima de procreación de las progenitoras mientras estén vivos (la línea genealógica se puede configurar según se dese tomando como base para la prueba de concepto **cargar a los perros con 20 generaciones anteriores**)
- Los perfiles de los diferentes usuarios
- Los datos relativos a las colegiaturas de los veterinarios
- El historial de inserción de microchips y vacunación de los perros
- ...



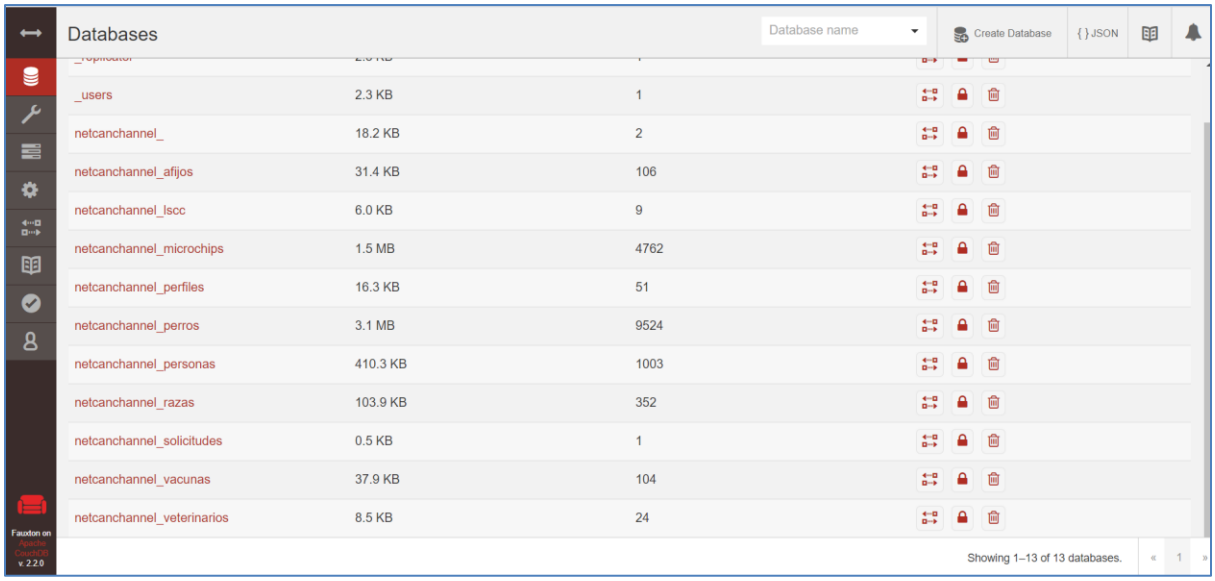
Una vez generada las tablas en la base de datos **MariaDB** se exportan a unos ficheros JSON que serán almacenados en GitHub en la siguiente dirección:

https://github.com/DFLBB/TFM_archs/tree/master/json

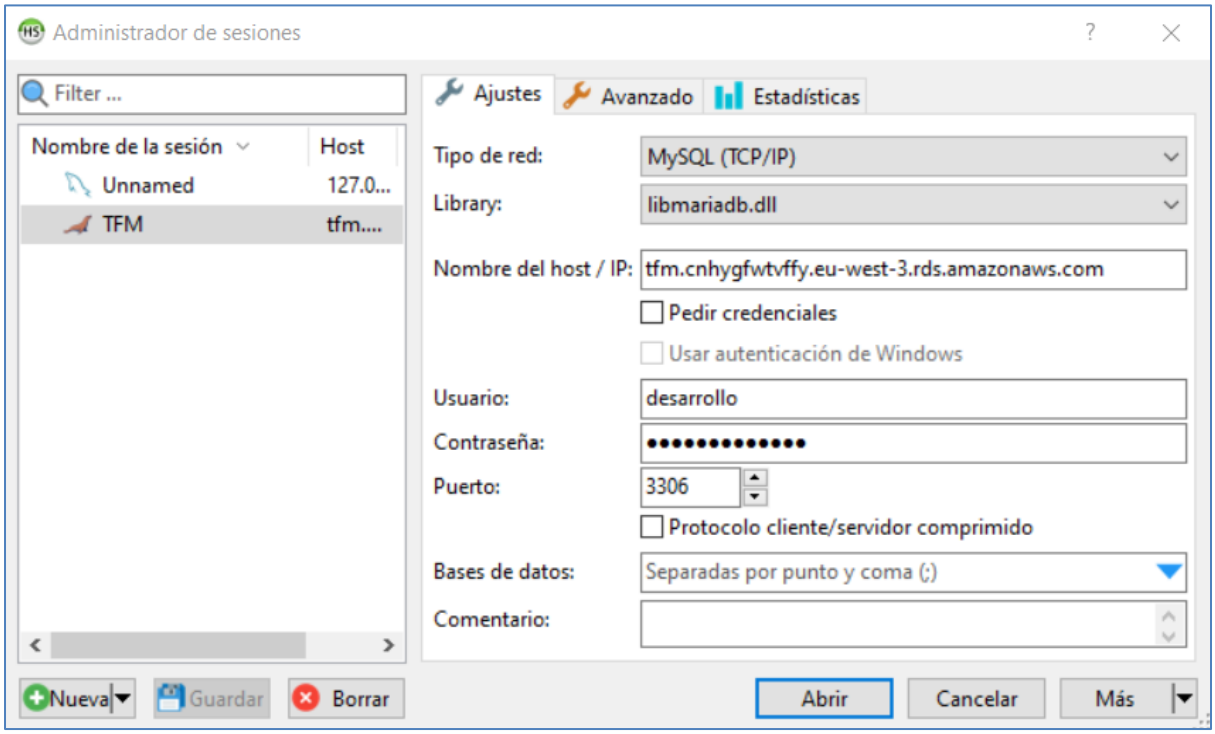


Cuando se arranca la red de blockchain los ficheros contenidos en esta carpeta son copiado junto con el resto de los scripts y código fuente de los contratos inteligente s en el servidor GNU Linux.

Al instalar e instanciar los contratos se copia los ficheros JSON dentro del Docker que lo contiene para su ejecución a través de las funciones de carga.



Para acceder a la Base de Datos de MariaDB lo puede hacer con la siguiente configuración:



- HOST: tfm.cnhygfwtvffy.eu-west-3.rds.amazonaws.com
- USUARIO: desarrollo
- CLAVE: tfmDesarrollo
- BASE DATOS: CARGA_INICIAL_Blockchain

También puede revisar estos procesos a través del script de SQL definido en:

https://github.com/DFLBB/TFM_archs/tree/master/script_mariadb_carga_inicial