

RAPPORT DE PROJET CORBA

Sécurité et Confidentialité du numérique personnel

Cyril CRISTOFOL – Farid EL JAMMAL – Manavai TEIKITUHAAHAA



Table des matières

* 1. [Diagramme de Cas d’utilisation 3](#_Toc469679614)
  2. [Diagrammes de séquence 4](#_Toc469679615)

[Louer un nom de domaine 4](#_Toc469679616)

[Configurer les paramètres d’un domaine 5](#_Toc469679617)

[Transférer un nom de domaine à un nouveau propriétaire 6](#_Toc469679618)

[Changer de Registrar 7](#_Toc469679619)

[Renouveler la location d’un nom de domaine 8](#_Toc469679620)

[Résolution DNS/WhoIs 9](#_Toc469679621)

* 1. [Diagramme de composants 10](#_Toc469679622)
  2. [Contrat IDL 10](#_Toc469679623)

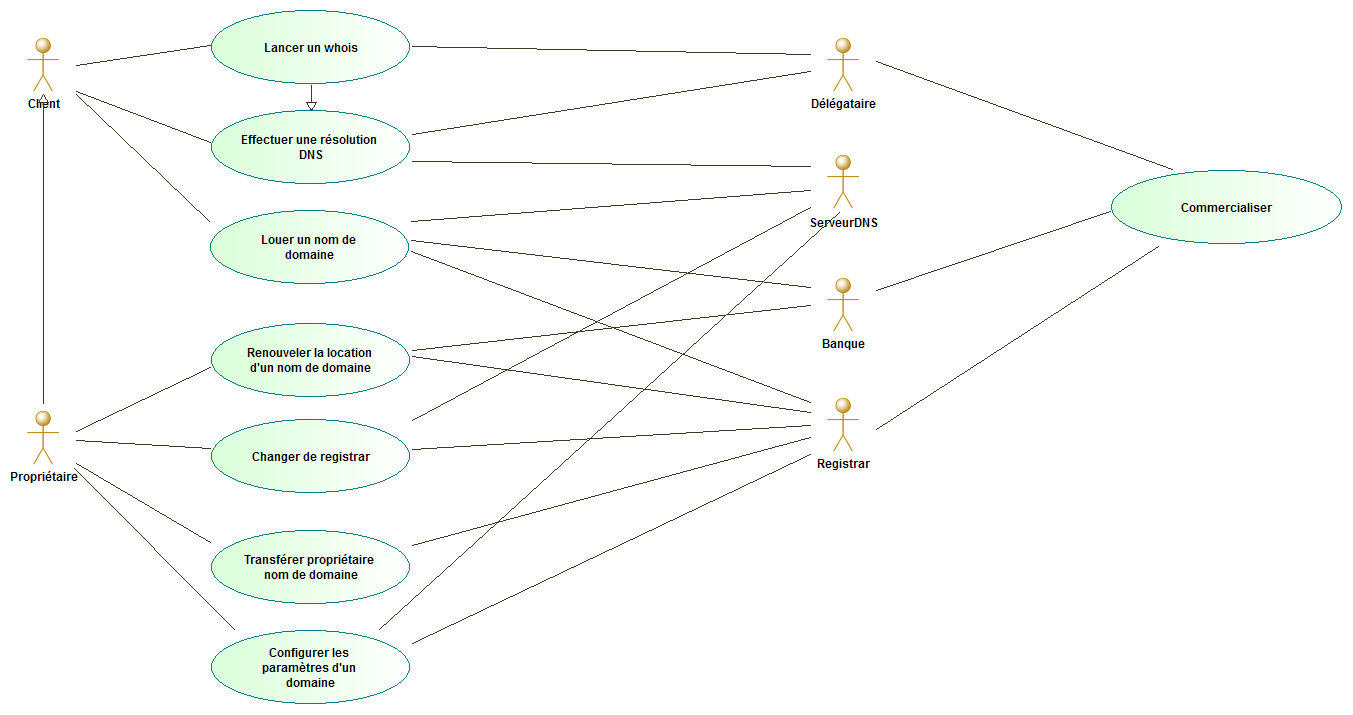
[Contenu 10](#_Toc469679624)

[Compilation avec IDLJ.EXE 13](#_Toc469679625)

* 1. [Annexes Numériques 14](#_Toc469679626)

# Diagramme de Cas d’utilisation

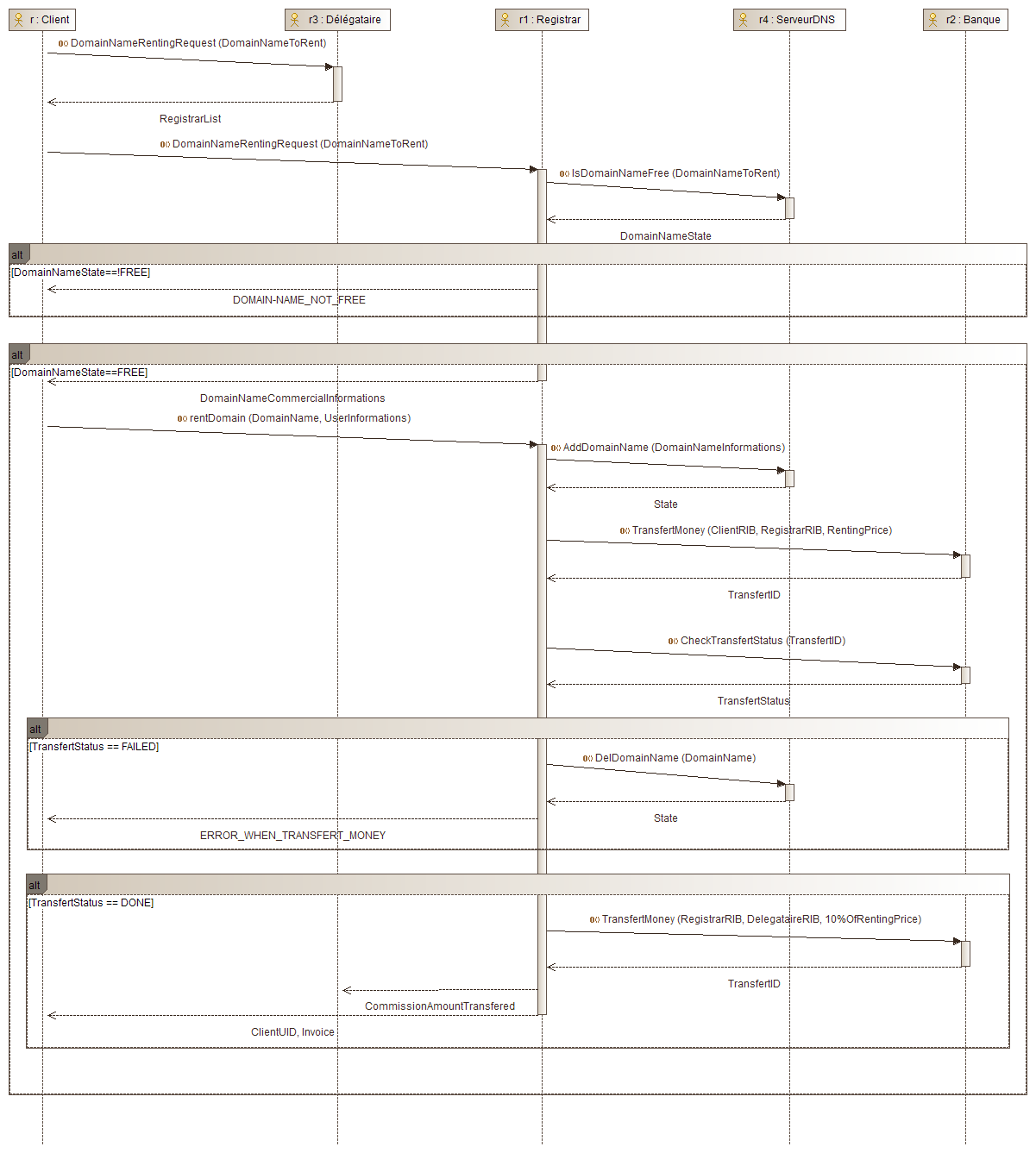
Tous les diagrammes ont été créé à partir du logiciel ***Modelio 3.6***, dans le workspace en dispnible en annexe numérique.



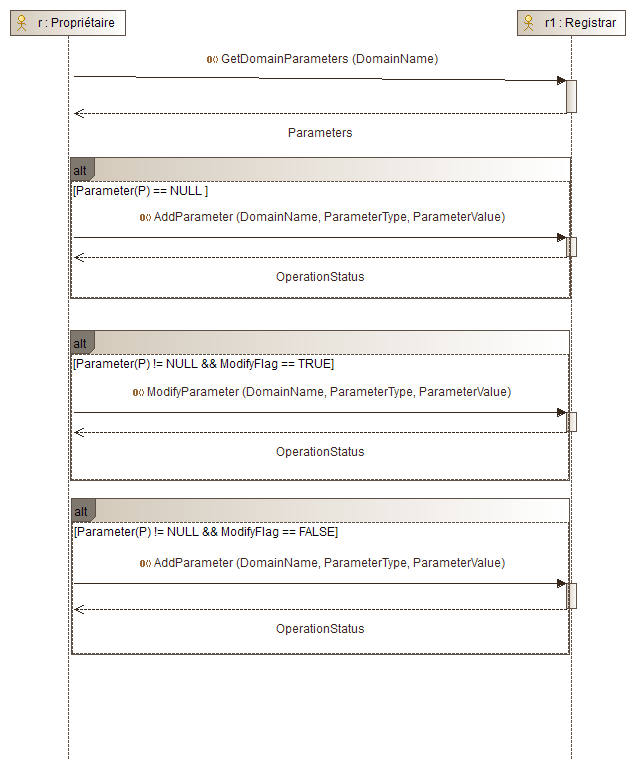
# Diagrammes de séquence

A partir de chaque cas d’utilisation, nous avons créé un diagramme de séquence associé.

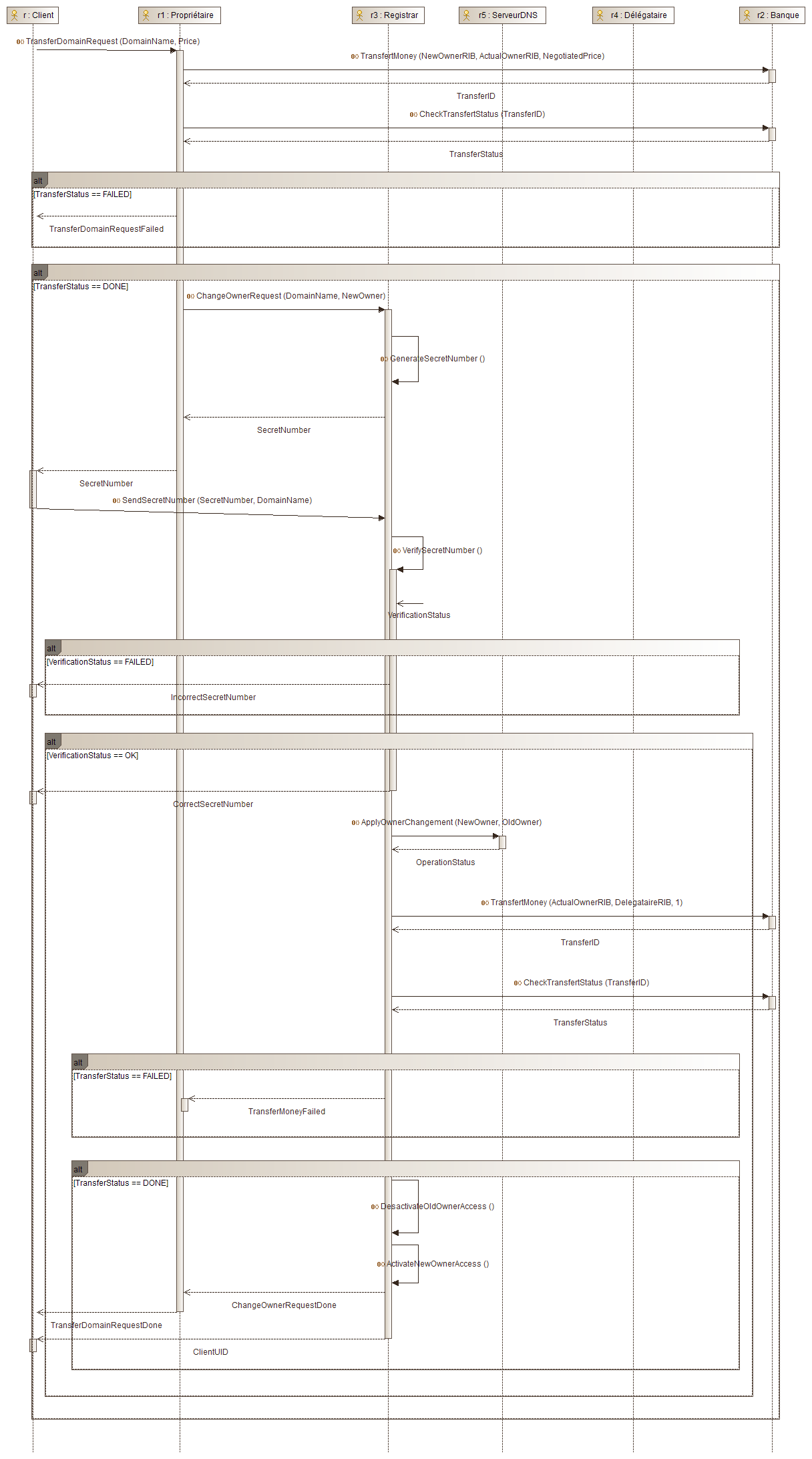
## Louer un nom de domaine



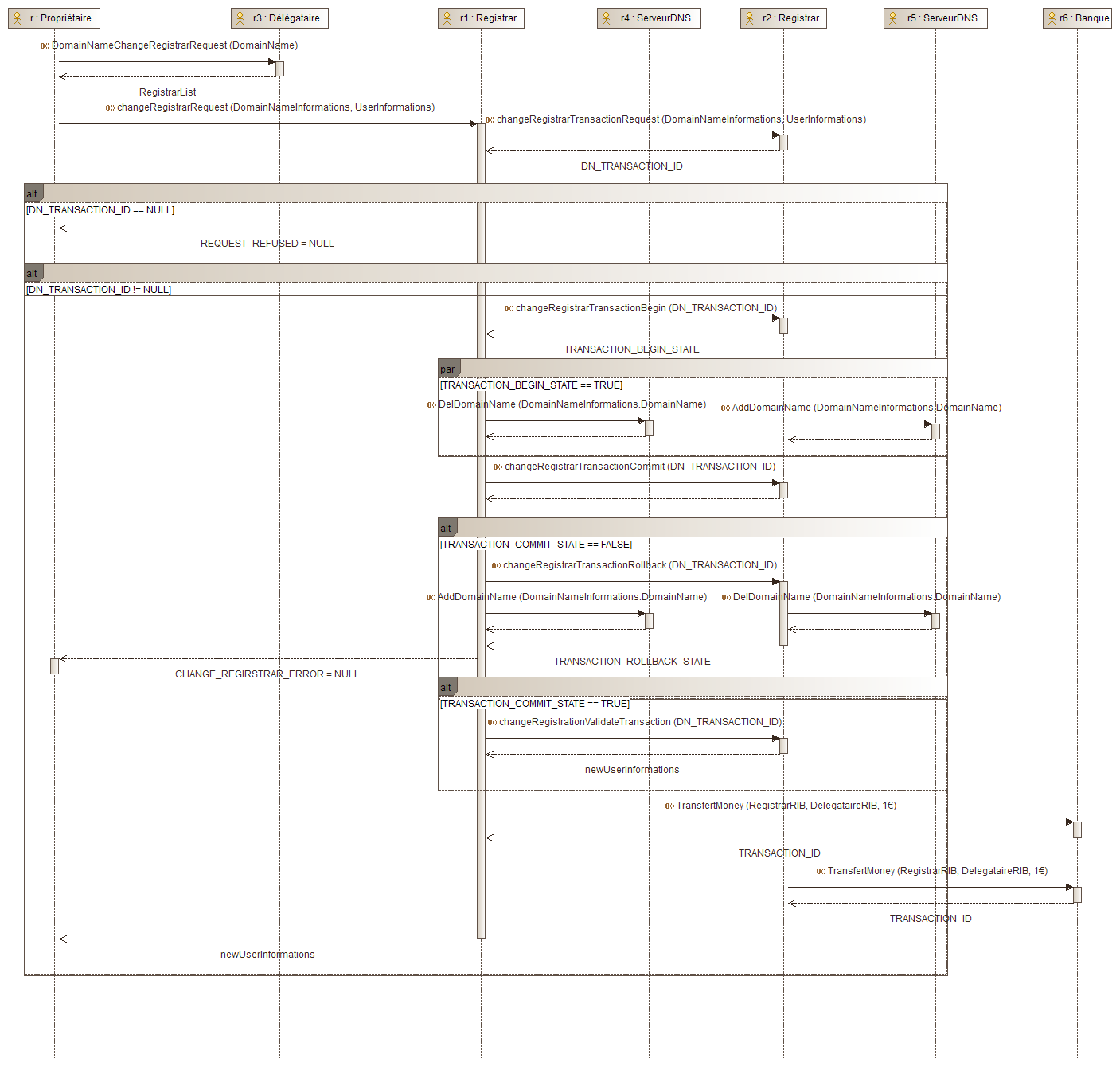
## Configurer les paramètres d’un domaine



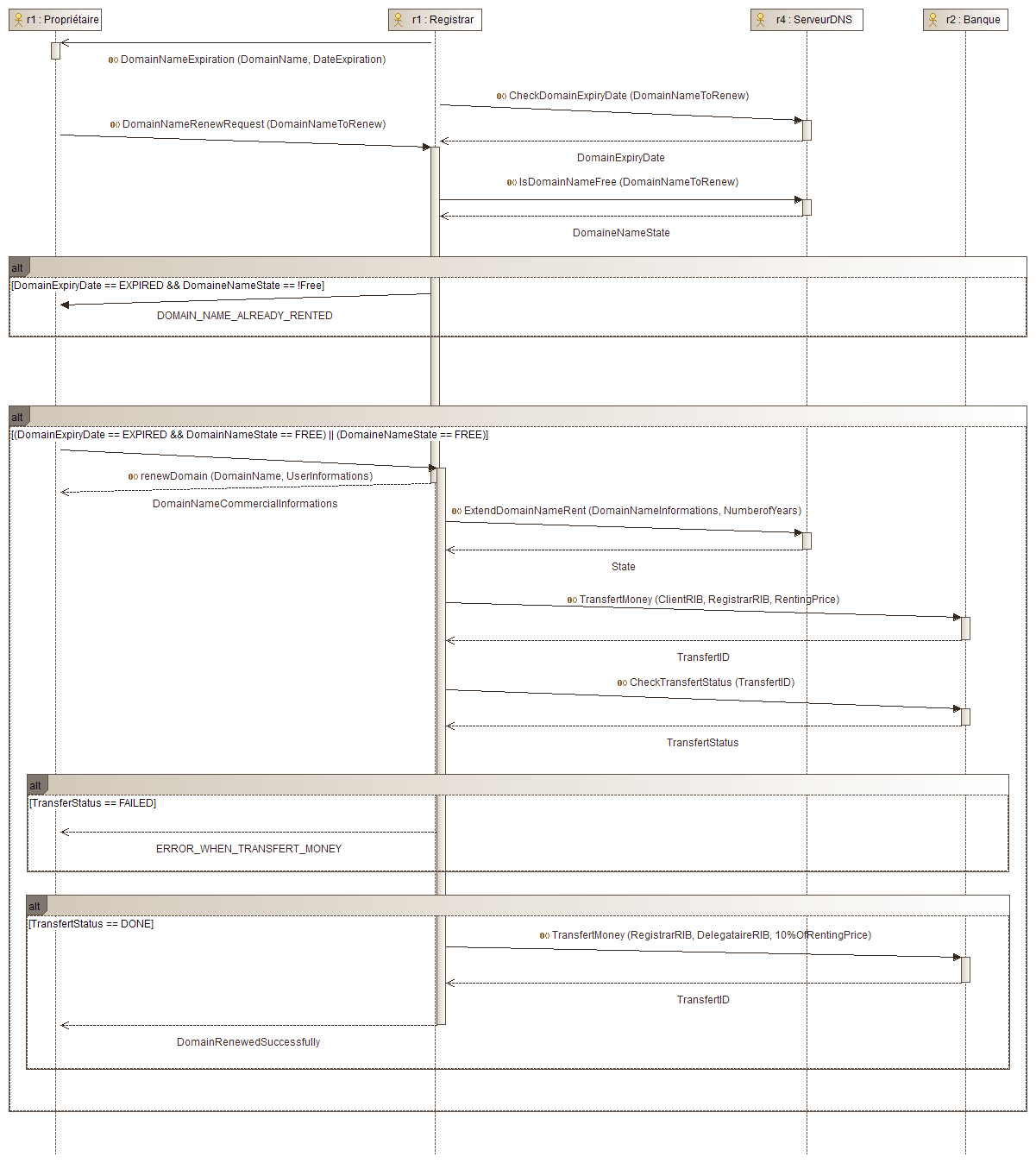
## Transférer un nom de domaine à un nouveau propriétaire



## Changer de Registrar

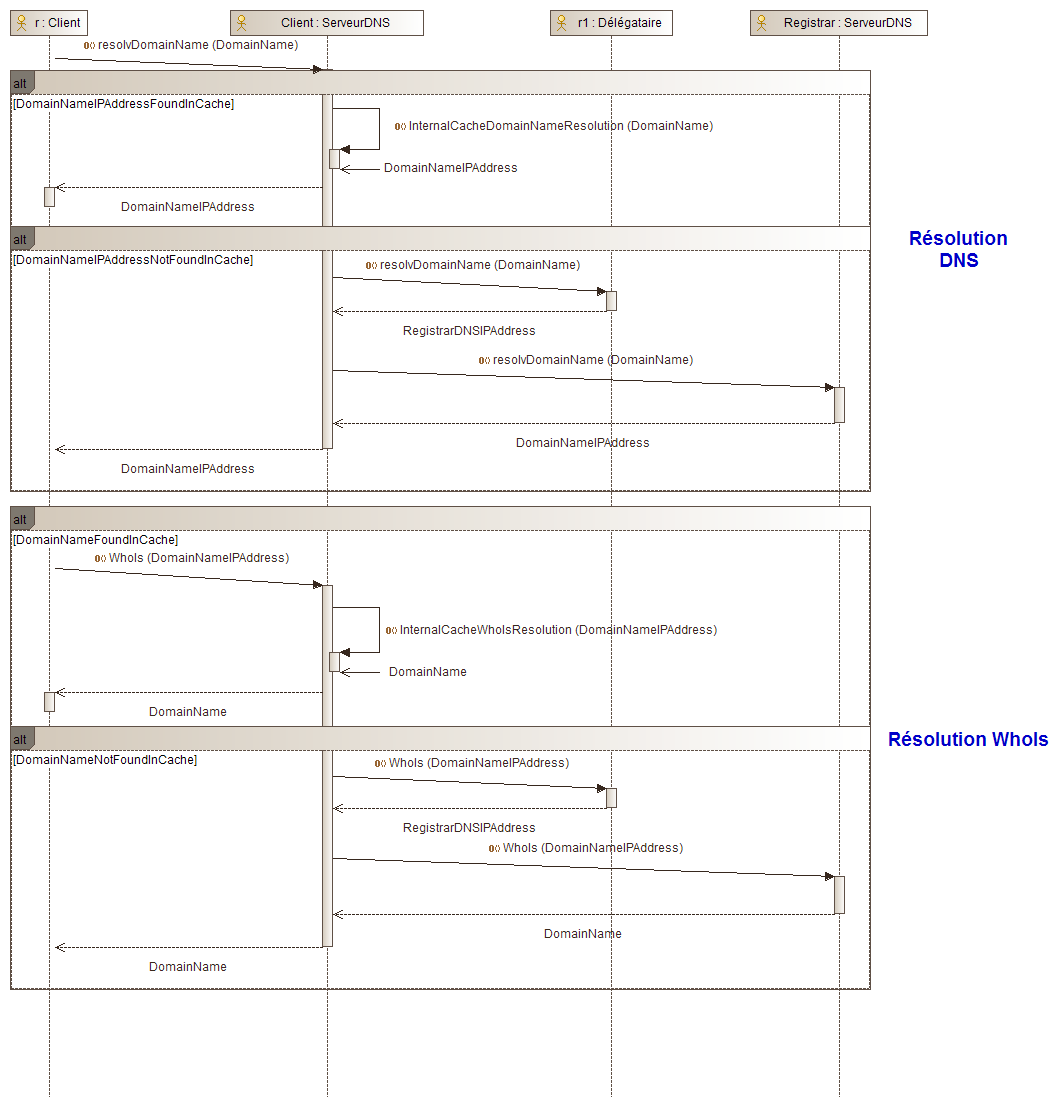


## Renouveler la location d’un nom de domaine

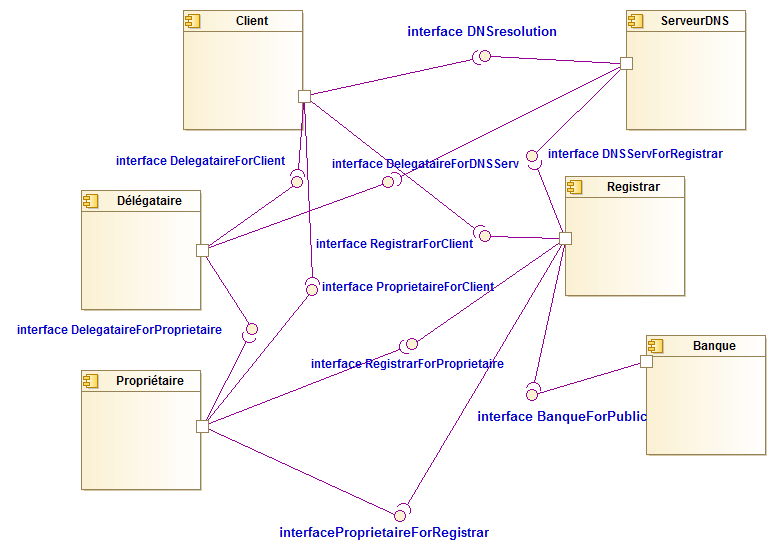


## Résolution DNS/WhoIs

a résolution DNN ou WhoIS se déroule comme spécifié dans le sujet. Il n’est pas fidèle au vrai système de résolution du DNS.



# Diagramme de composants



# Contrat IDL

Le contrat IDL est rédigé selon la méthode suivante :

* Créer un module pour chaque Acteur
* Créer une interface pour chacune présente dans le digramme de composants, en les plaçant dans le module fournissant le service
* placer les méthodes créées lors de la création du diagramme de séquence dans les bonnes interfaces
* Créer des variables redéfinis (typedef) au fur et à mesure pour faire correspondre avec le nom des variables des diagrammes de séquence.

Le contrat IDL est disponible en Annexe numérique.

## Contenu

/\*

M2 STRI 2016 - PROJET OMG CORBA

CRISTOFOL - EL JAMMAL - TEIKITUHAAHAA

\*/

module ProjetCorbaDNS{

//typedef simple

typedef string DomainName;

typedef string IPAddress;

typedef string Date;

typedef string ClientUID;

typedef string DNTransactionID;

typedef string TransfertStatus;

typedef string TransfertID;

typedef string ParameterType;

typedef string ParameterValue;

//struct

//

struct UserInformations{

string nom;

string prenom;

long age; // int en java

string adresse;

string mail;

string password;

string RIB;

};

//

struct DomainNameInformations{

DomainName domainName;

IPAddress IPAddr;

ClientUID clientUID;

Date ExpirationDate;

};

//

struct ClientID{

UserInformations userInformations;

ClientUID clientUID;

};

//

struct RegistrarClass{

string Nom;

string Contact;

string RIB;

};

//

struct DomainNameParameters{

// non-défini

string ParameterType;

string ParameterValue;

};

//

typedef sequence<RegistrarClass> RegistrarList;

//

module Registrar {

interface RegistrarForClient {

// Louer un nom de domaine

// demander à un registrar si un nom de domaine est libre pour location

boolean DomainNameRentingRequest(in DomainName DomainNameToRent);

// demander à un registrar de louer un nom de domaine

ClientID rentDomain(in DomainName DomainNameToRent, in UserInformations userInformations);

};

interface RegistrarForProprietaire {

// Configurer les paramètres d'un nom de domaine

boolean AddParameter(in ParameterType ParameterType, in DomainName DomainName, in ParameterValue ParameterValue);

boolean ModifyParameter(in DomainName DomainName, in ParameterType ParameterType, in ParameterValue ParameterValue);

DomainNameParameters GetDomainParameters(in DomainName DomainName);

// transferer un nom de domaine vers un nouveau propriétaire

boolean DomainNameRenewRequest(in DomainName DomainNameToRenew);

boolean renewDomain(in DomainName DomainName, in UserInformations UserInformations);

boolean ChangeOwnerRequest(in DomainName DomainName, in UserInformations NewOwner);

// changer de registrar

UserInformations changeRegistrarRequest(in DomainNameInformations DomainNameInformations, in UserInformations UserInformations);

DNTransactionID changeRegistrarTransactionRequest(in DomainNameInformations DomainNameInformations);

boolean changeRegistrarTransactionBegin(in DNTransactionID DNTransactionID);

boolean changeRegistrarTransactionCommit(in DNTransactionID DNTransactionID);

boolean changeRegistrarTransactionRollback(in DNTransactionID DNTransactionID);

UserInformations changeRegistrarValidateTransaction(in DNTransactionID DNTransactionID);

};

};

//

module Delegataire {

//

interface DelegataireForProprietaire {

// méthode pour obtenir la liste des registrar pour un nom de domaine donné pour une location

RegistrarList DomainNameRentingRequest(in DomainName DomainName);

};

//

interface DelegataireForClient {

// méthode pour obtenir la liste des registrar pour un nom de domaine donné pour un changement de registrar

RegistrarList DomainNameChangeRegistrarRequest(in DomainName DomainName);

};

//

interface DelegataireForDNSServ{

// résolution DNS

IPAddress resolVDomainName(in DomainName DomainName);

// résolution WhoIS

DomainName WhoIs(in IPAddress IPAddress);

};

};

//

module ServeurDNS{

//

interface DNSresolution {

//

IPAddress resolVDomainName(in DomainName DomainName);

//

DomainName WhoIs(in IPAddress IPAddress);

};

//

interface DNSServForRegistrar {

// vérifier l'état d'un nom de domaine (libre ou non)

boolean IsDomainNameFree(in DomainName DomainName);

// Ajout et supression de nom de domaine

boolean AddDomainName(in DomainNameInformations DomainNameInformations);

boolean DelDomainName(in DomainNameInformations DomainNameInformations);

// Vérification de la date d'expiration

Date CheckDomainExpiryDate(in DomainName DomainNameToRenew);

// Renouvellement d'un nom de domaine

boolean ExtendDomainNameRent(in DomainNameInformations DomainNameInformations, in long NumberOfYears);

// Changement de propriétaire d'un nom de domaine

boolean ApplyOwnerChangement(in UserInformations NewOwner, in UserInformations OldOwner);

};

};

//

module Banque{

/\* Interface utilisé par les registrar pour demander un transfert d'argent

lors d'un location, d'un renouvellement de domaine, ou bien d'un changement

de Registrar par un propriétaire d'un DN.

\*/

interface BanqueForPublic {

// méthode permettant d'ordonner un Transfert d'argent

TransfertID TransferMoney(in string TransfertFromRIB, in string TransfertToRIB, in long SUM);

// méthode permettant de vérifier le status d'un transfert

TransfertStatus CheckTransfertStatus(in TransfertID transfertID);

};

};

//

module Proprietaire{

interface ProprietaireToClient{

// demande de transfert d'un nom de domaine

boolean TransfertDomainRequest(in DomainName DomainName, in long Price);

};

interface ProprietaireforRegistrar {

// notifie le client qu'un nom de domaine est sur le point d'expirer

oneway void DomainNameExpiration(in DomainName DomainName, in Date DateExpiration);

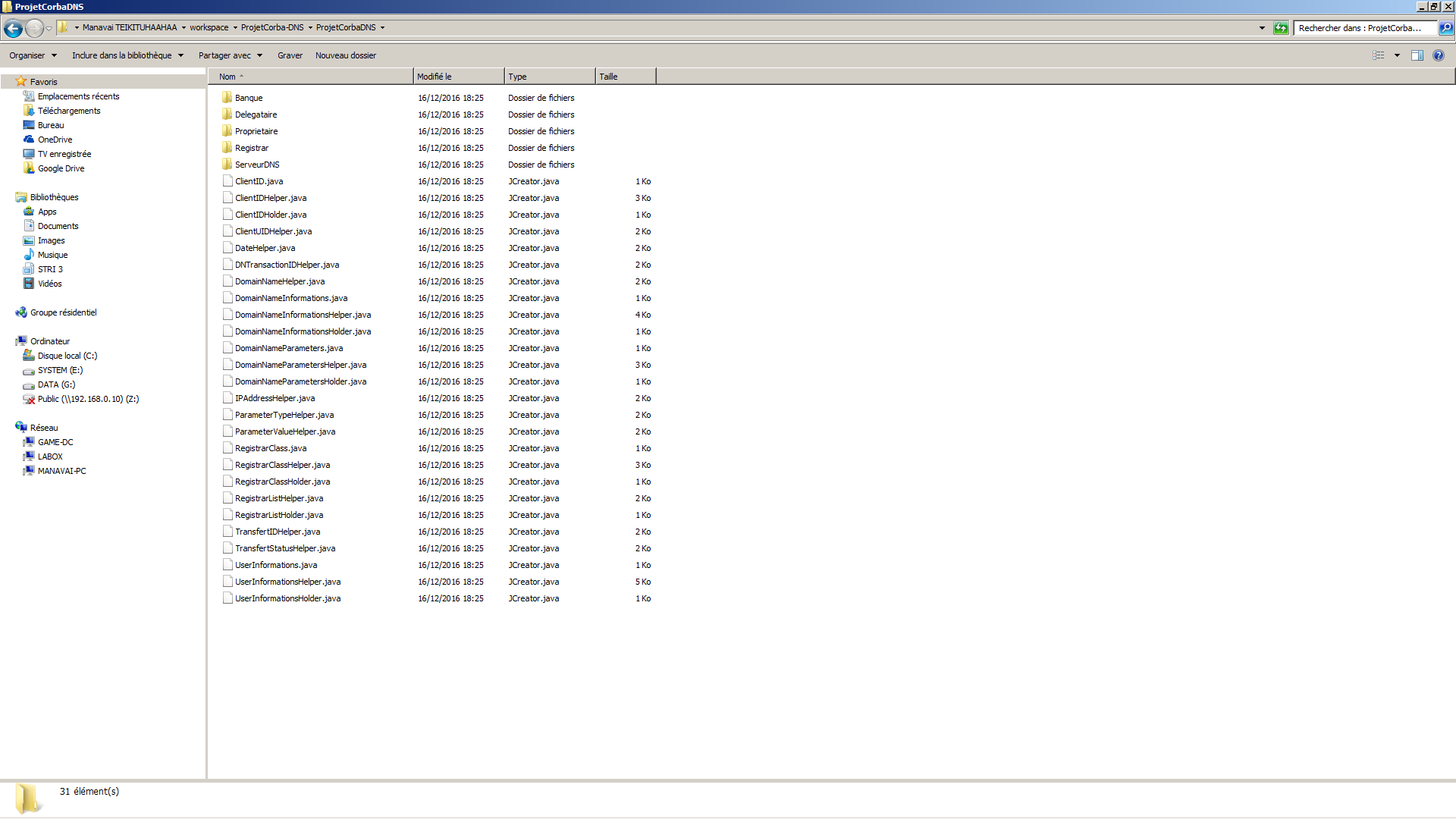
};

};

};

## Compilation avec IDLJ.EXE

Pour compiler le contrat IDL et ainsi vérifier s’il y avait des fautes de syntaxes, nous avons utilisé l’application IDLJ.EXE disponible dans la JDK (version utilisé 1.8.40). Voici ce qu’a donné le résultat de la compilation :



Pour chaque module, il y a donc eu un package de créé. On peut donc conclure qu’il n’y pas d’erreur de syntaxe IDL.

# Annexes Numériques

Voici la liste des éléments disponible en annexes numériques :

* Projet modélio 3.6 pour la création des diagrammes ;
* Contrait IDL : GestionNomsDomaine.idl ;
* Le paquetage créé à la suite de la compilation du contrat IDL (dossier ProjetCorbaDNS) ;

Le Github du projet est disponible au lien suivant : https://github.com/cyrilcristofol/ProjetCorba-DNS/