



RAPPORT DE PROJET CORBA

Sécurité et Confidentialité du numérique personnel

Cyril CRISTOFOL – Farid EL JAMMAL – Manavai TEIKITUHAAHAA

Table des matières

1.1 INTRODUCTION	3
1.2 DIAGRAMME DE CAS D'UTILISATION	4
1.3 DIAGRAMMES DE SEQUENCE	5
1.3.1 LOUER UN NOM DE DOMAINE	5
1.3.2 CONFIGURER LES PARAMETRES D'UN DOMAINE	6
1.3.3 TRANSFERER UN NOM DE DOMAINE A UN NOUVEAU PROPRIETAIRE	7
1.3.4 CHANGER DE REGISTRAR	8
1.3.5 RENOUELER LA LOCATION D'UN NOM DE DOMAINE	9
1.3.6 RESOLUTION DNS/WHOIS	10
1.4 DIAGRAMME DE COMPOSANTS	11
1.5 CONTRAT IDL	11
1.5.1 CONTENU	11
1.5.2 COMPILATION AVEC IDLJ.EXE	14
1.6 ANNEXES NUMERIQUES	15

Introduction

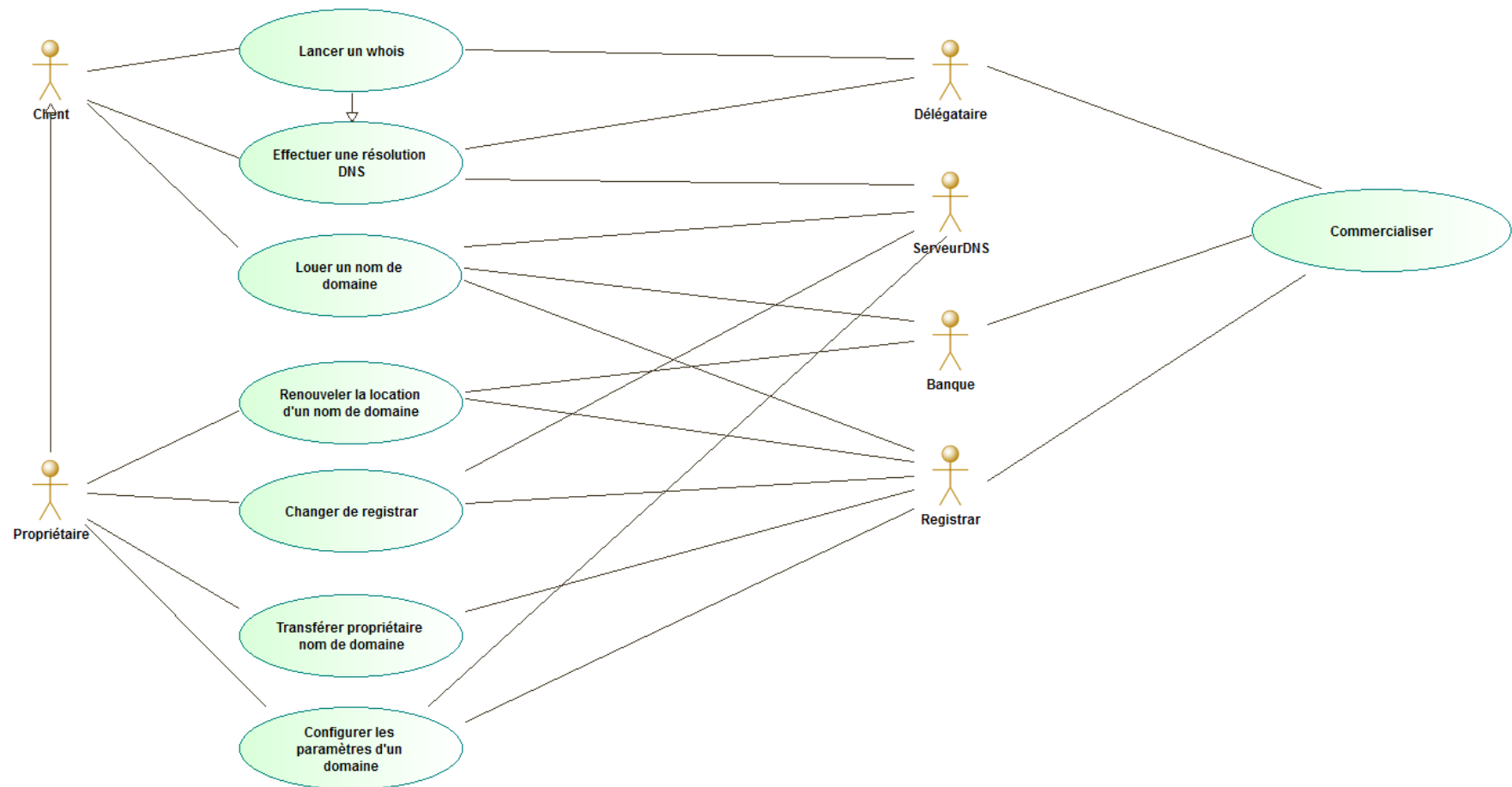
Dans le cadre du module « traitements répartis », il nous est proposé un projet de conception CORBA qui consiste à simuler un système permettant une gestion des noms de domaines sur Internet. Ce dossier, comporte en plus du contrat IDL les éléments suivants :

- les diagrammes UML exprimant le résultat de l'analyse de cette application ;
- les différentes entités logicielles ;
- les interactions pouvant survenir entre ces types d'entités.

Dans la première partie, nous avons le diagramme de cas d'utilisation. Ensuite, nous avons les différents diagrammes de séquences, ainsi que le diagramme de composants. A la fin, nous présentons notre contrat IDL commenté.

Diagramme de Cas d'utilisation

Tous les diagrammes ont été créés à partir du logiciel **Modelio 3.6**, dans le workspace disponible en annexe numérique. Notre diagramme de cas d'utilisation composé de 6 acteurs : le client, le propriétaire, le serveurDNS, le délégataire, le registrar, et la banque. L'interaction qui peut se produire entre les 3 derniers acteurs est principalement la commercialisation des noms de domaine. Elle regroupe plusieurs sous-fonctions qui seront détaillées dans les diagrammes de séquences. De même pour les autres interactions visibles sur le diagramme ci-dessous :

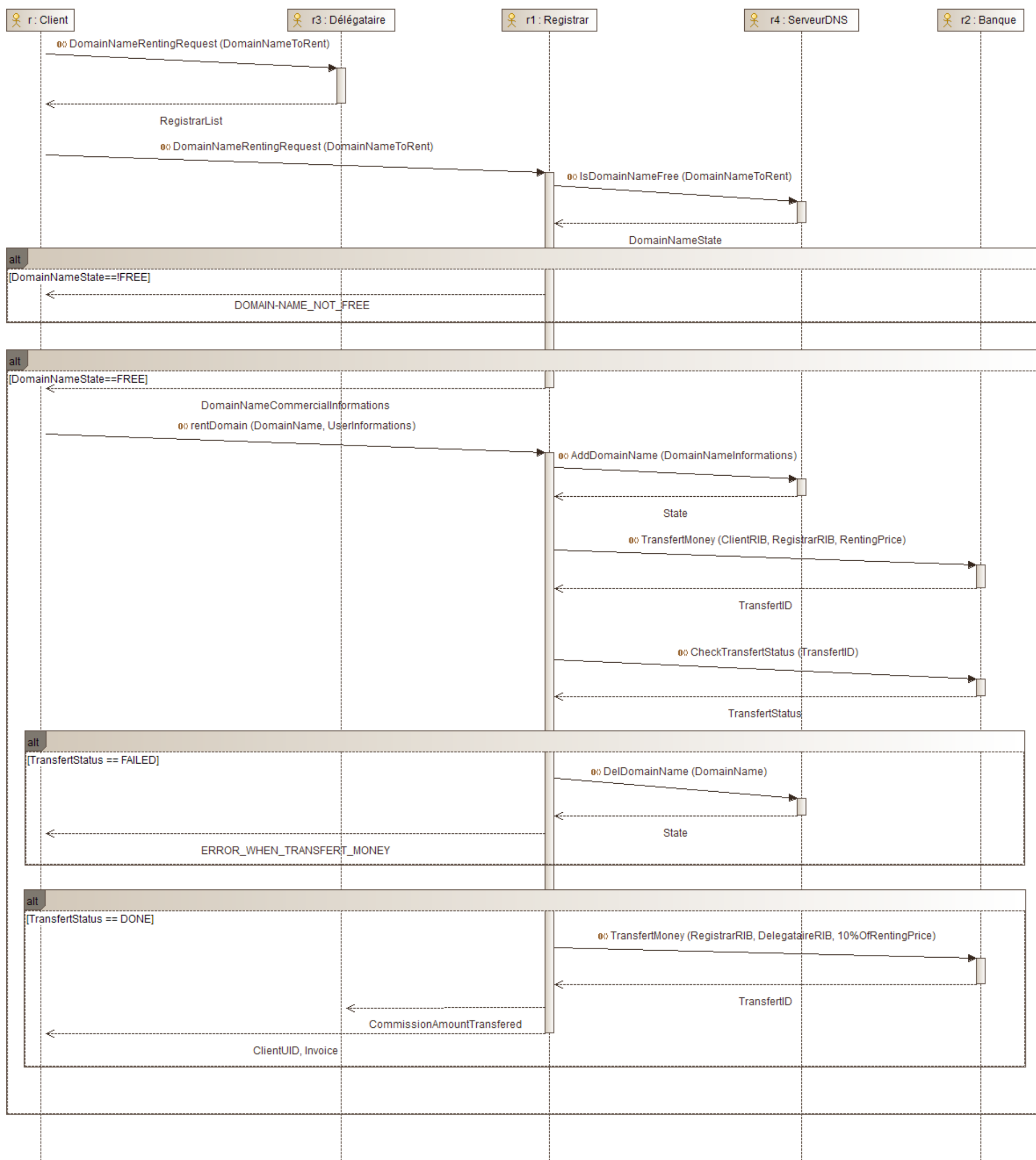


Diagrammes de séquence

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

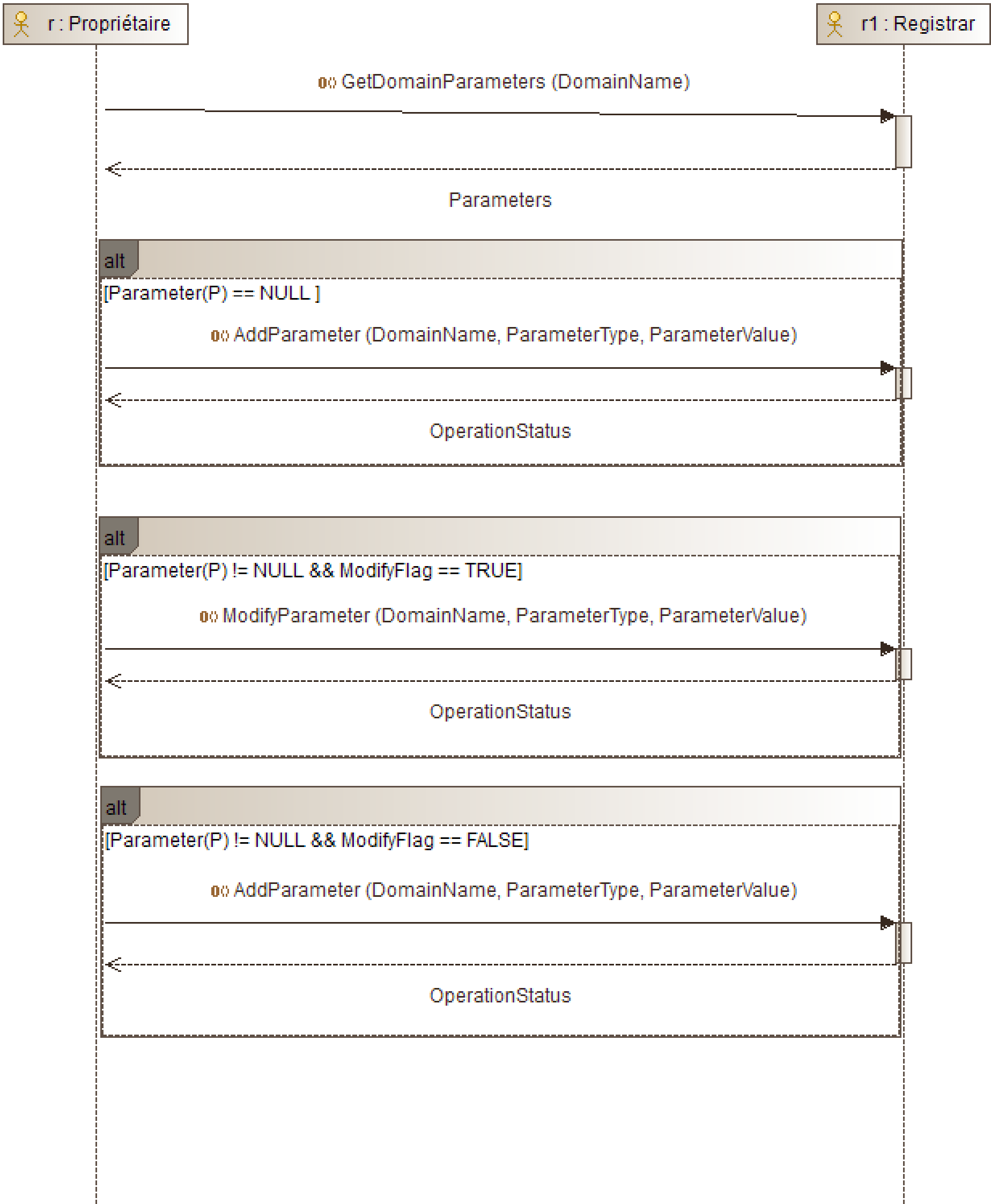
1.3.1 Louer un nom de domaine

Cette action consiste à louer un nom de domaine par un client. Les acteurs impliqués sont donc le client, le délégataire, le registrar, le serveurDNS et la banque. Une fois l'opération réussite, le client devient propriétaire d'un nom de domaine et aura donc son identificateur UID et une facture, comme le précise le sujet.



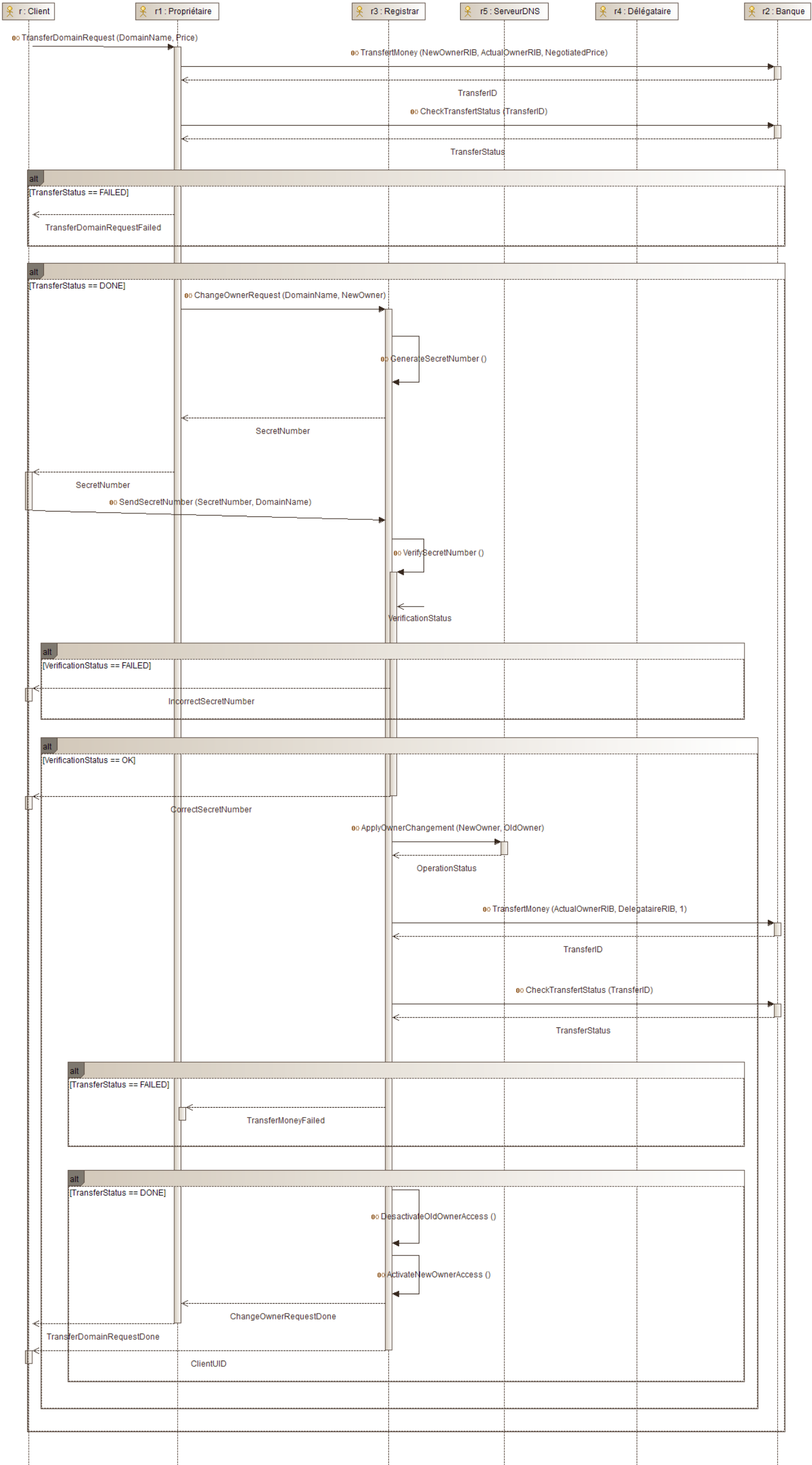
1.3.2 Configurer les paramètres d'un domaine

Cette opération permet à un propriétaire de configurer les paramètres de son nom de domaine, notamment les correspondances nom symboliques-@IP (types AA, AAAA, CNAME, MX...). Il pourra également ajouter ou changer son adresse mail, son mot de passe ou tout autre paramètre. En prenant l'exemple des nom symboliques, un client souhaitant ajouter un CNAME devra d'abord demander à son registrar la liste des CNAME existants. S'il n'y en a pas, le client pourra l'ajouter par la méthode « AddParameter ». S'il en a mais qu'il souhaite le modifier, il pourra le faire par la méthode « modifyParameter ». Dans le cas où un CNAME existe déjà mais le client souhaite en ajouter un autre, il pourra l'indiquer à son registrar et procéder par la méthode « AddParameter ».



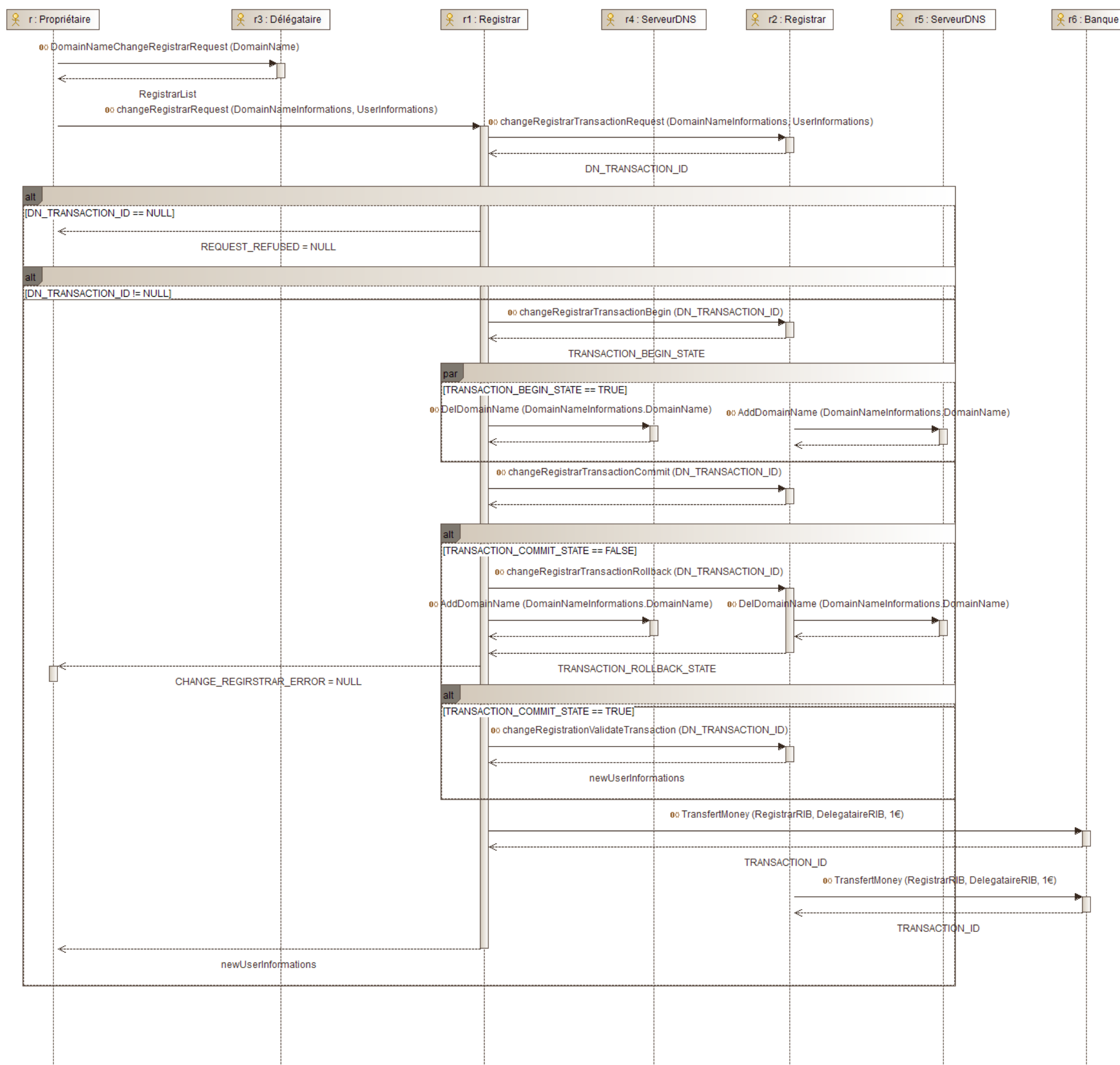
1.3.3 Transférer un nom de domaine à un nouveau propriétaire

Cette opération consiste à transférer un nom de domaine à un nouveau propriétaire. Tous les acteurs de notre système seront impliqués selon le digramme ci-dessous :



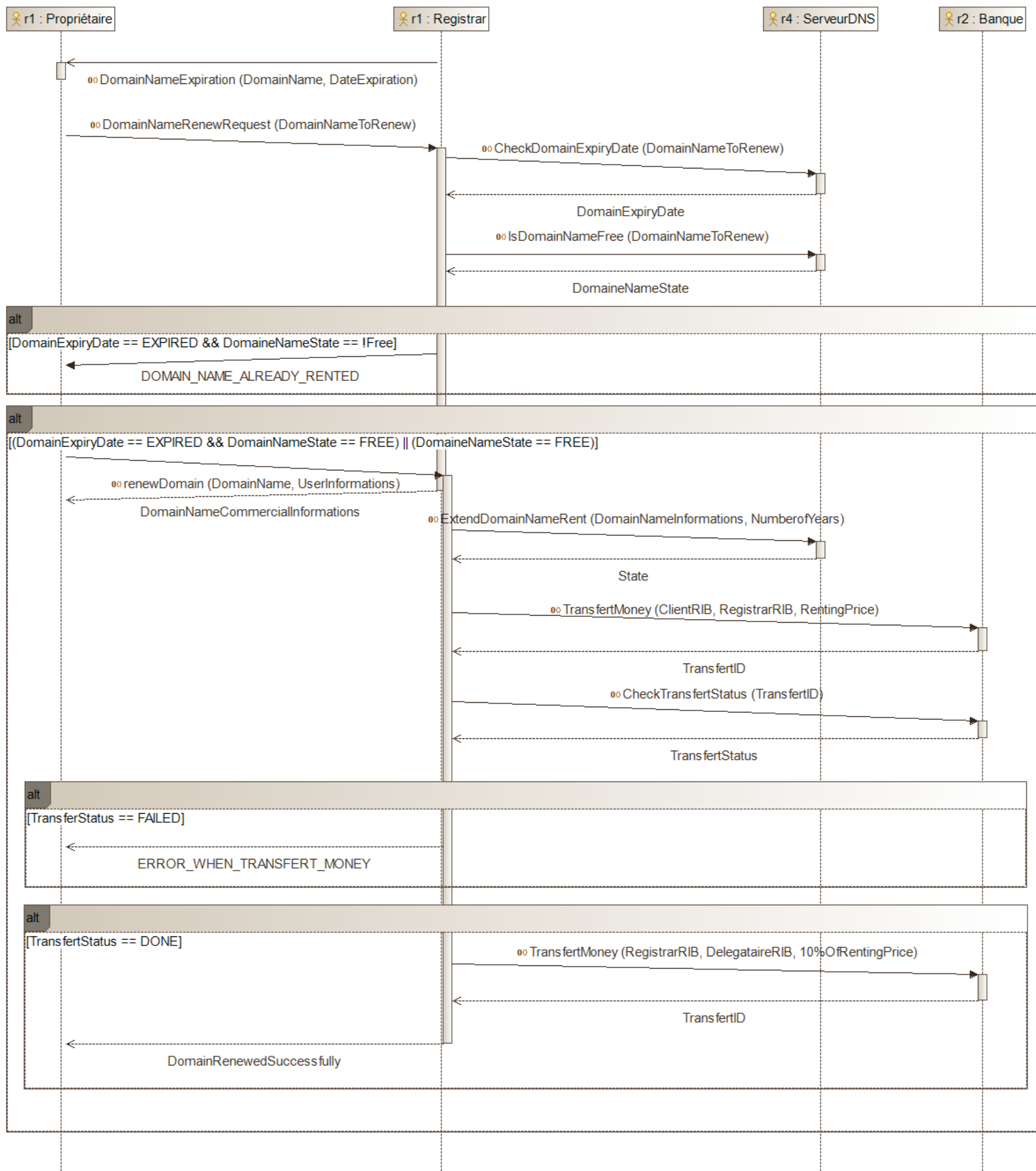
1.3.4 Changer de Registrar

Cette opération consiste à changer de registrar par un propriétaire d'un nom de domaine. Les interactions se font entre ce dernier et le délégataire de son domaine ainsi que son registrar, le nouveau registrar, le serveurDNS associé à chacun des registrar et aussi la banque pour le paiement des frais qui en résultent.



1.3.5 Renouveler la location d'un nom de domaine

Cette opération consiste à renouveler la location d'un nom de domaine par une demande au registrar concerné. Comme ce dernier devra notifier le propriétaire du nom de domaine de l'imminence de l'échéance de son abonnement, le diagramme commence par la fonction `DomaineNameExpiration`. Ensuite, le propriétaire devra réagir par l'issue d'une demande de renouvellement de location auprès de son registrar. A l'issus de cette demande, le registrar, revérifiera la date d'expiration du domaine en question et sa disponibilité. Si le propriétaire a mis du temps pour réagir, le domaine risque d'être récemment loué par un autre client et dans ce cas l'opération échoue et le propriétaire sera informé. Dans tous les autres cas, le registrar complètera la demande comme le montre le diagramme ci-dessous



1.3.6 Résolution DNS/WhoIs

La résolution DNN ou WhoIs se déroule comme spécifié dans le sujet. Cependant, elle n'est pas fidèle au vrai système de résolution du DNS.

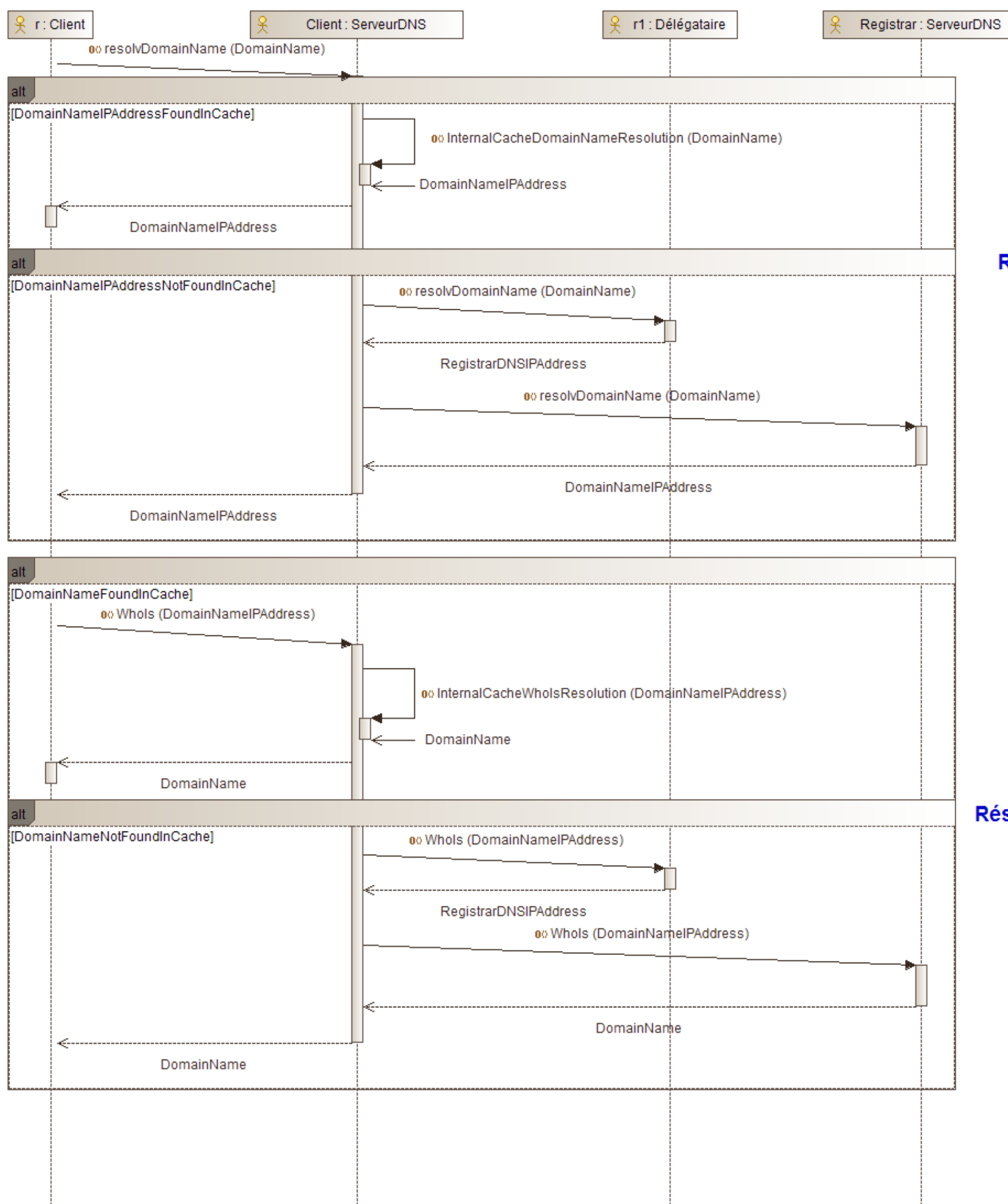
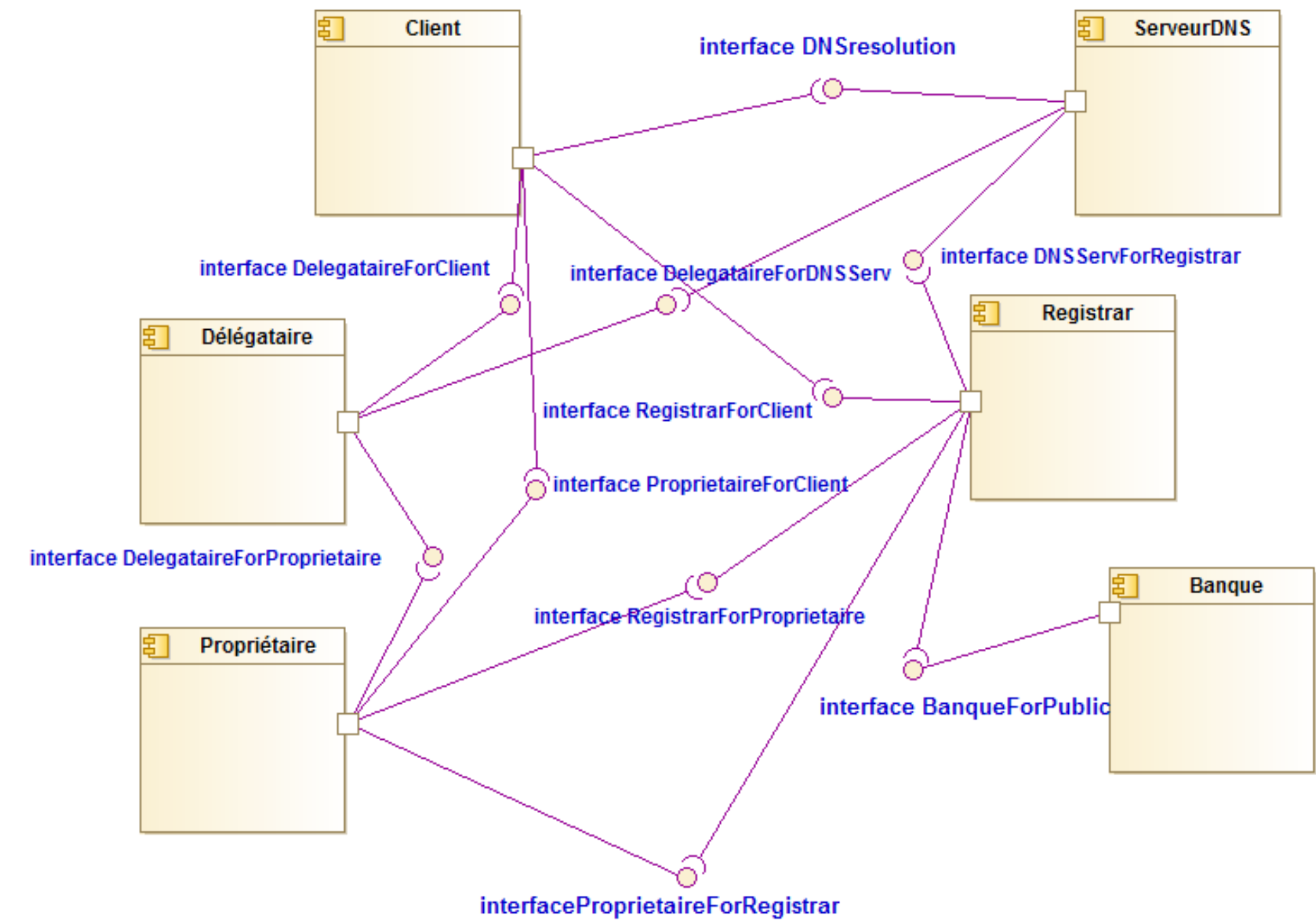


Diagramme de composants

Dans cette partie, nous représentons notre diagramme de composants qui décrit l'organisation du système du point de vue des éléments logiciels :



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](#).

1.5.1 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 suppression de nom de domaine
        boolean AddDomainName(in DomainNameInformation DomainNameInformation);
        boolean DelDomainName(in DomainNameInformation DomainNameInformation);
        // Vérification de la date d'expiration
        Date CheckDomainExpiryDate(in DomainName DomainNameToRenew);
        // Renouvellement d'un nom de domaine
        boolean ExtendDomainNameRent(in DomainNameInformation DomainNameInformation, in long
NumberOfYears);
        // Changement de propriétaire d'un nom de domaine
        boolean ApplyOwnerChangement(in UserInformation NewOwner, in UserInformation 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);
    };
};
};
};

```

1.5.2 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 :

 Banque	16/12/2016 18:25	Dossier de fichiers	
 Delegataire	16/12/2016 18:25	Dossier de fichiers	
 Proprietaire	16/12/2016 18:25	Dossier de fichiers	
 Registrar	16/12/2016 18:25	Dossier de fichiers	
 ServeurDNS	16/12/2016 18:25	Dossier de fichiers	
 ClientID.java	16/12/2016 18:25	JCreator.java	1 Ko
 ClientIDHelper.java	16/12/2016 18:25	JCreator.java	3 Ko
 ClientIDHolder.java	16/12/2016 18:25	JCreator.java	1 Ko
 ClientUIDHelper.java	16/12/2016 18:25	JCreator.java	2 Ko
 DateHelper.java	16/12/2016 18:25	JCreator.java	2 Ko
 DNTransactionIDHelper.java	16/12/2016 18:25	JCreator.java	2 Ko
 DomainNameHelper.java	16/12/2016 18:25	JCreator.java	2 Ko
 DomainNameInformations.java	16/12/2016 18:25	JCreator.java	1 Ko
 DomainNameInformationsHelper.java	16/12/2016 18:25	JCreator.java	4 Ko
 DomainNameInformationsHolder.java	16/12/2016 18:25	JCreator.java	1 Ko
 DomainNameParameters.java	16/12/2016 18:25	JCreator.java	1 Ko
 DomainNameParametersHelper.java	16/12/2016 18:25	JCreator.java	3 Ko
 DomainNameParametersHolder.java	16/12/2016 18:25	JCreator.java	1 Ko
 IPAddressHelper.java	16/12/2016 18:25	JCreator.java	2 Ko
 ParameterTypeHelper.java	16/12/2016 18:25	JCreator.java	2 Ko
 ParameterValueHelper.java	16/12/2016 18:25	JCreator.java	2 Ko
 RegistrarClass.java	16/12/2016 18:25	JCreator.java	1 Ko
 RegistrarClassHelper.java	16/12/2016 18:25	JCreator.java	3 Ko
 RegistrarClassHolder.java	16/12/2016 18:25	JCreator.java	1 Ko
 RegistrarListHelper.java	16/12/2016 18:25	JCreator.java	2 Ko
 RegistrarListHolder.java	16/12/2016 18:25	JCreator.java	1 Ko
 TransfertIDHelper.java	16/12/2016 18:25	JCreator.java	2 Ko
 TransfertStatusHelper.java	16/12/2016 18:25	JCreator.java	2 Ko
 UserInformations.java	16/12/2016 18:25	JCreator.java	1 Ko
 UserInformationsHelper.java	16/12/2016 18:25	JCreator.java	5 Ko
 UserInformationsHolder.java	16/12/2016 18:25	JCreator.java	1 Ko

Pour chaque module, il y a donc eu un package 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 ;
- Contrat 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/>