



## Master Informatique

Programmation distribuée  
M1 / 178UD02

# C2 – Introduction à RPC

Thierry Lemeunier

[thierry.lemeunier@univ-lemans.fr](mailto:thierry.lemeunier@univ-lemans.fr)

---

# Plan du cours

- Principes de RPC
- ONC RPC
  - Caractéristiques
  - Les APIs
  - RPC Language et XDR
  - Exemples

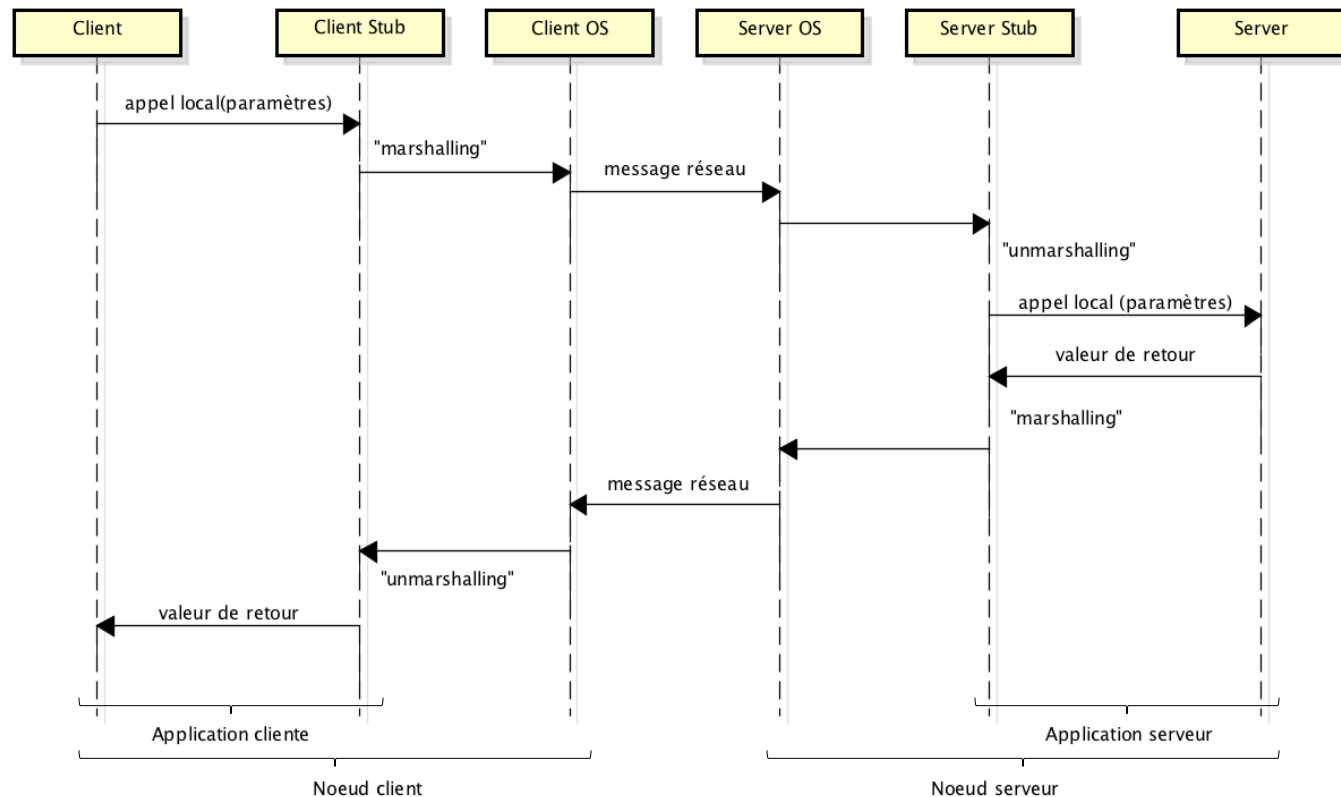
# Principes de RPC (1/2)

## ■ Philosophie de RPC

- ❑ RPC : Remote Procedure Call
- ❑ Appeler une procédure se trouvant dans un espace d'adressage différent de celui de l'appelant (2 nœuds ou 2 processus sur le même nœud !)
- ❑ Transparence de l'appel distant  $\approx$  appel local
- ❑ Procédure *idempotent* (sans état) = retourne toujours le même résultat quelque soit le client

## ■ Historique (très très bref)

- ❑ Propositions théoriques années 1970
- ❑ Premières mises en œuvre commerciales : début des années 1980 (sous systèmes Unix avec Sun RPC)



# Principes de RPC (2/2)

## ■ Problématiques de RPC

- ❑ Protocole de type requête-réponse synchrone (= modèle du client / serveur)
- ❑ La procédure distante ne peut pas modifier la zone mémoire du nœud client
  - => envoie d'une copie des valeurs pointées pour les paramètres de type pointeur
  - => retour d'une copie de la valeur pointée pour le cas d'un retour de type pointeur
- ❑ Utilise les réseaux :
  - Réseau informatique : latence ; congestion ; pertes de paquets ; coupures
  - LAN ok ; WAN ko ?
- ❑ Formalisme de codage des données : quelle procédure ? ; les paramètres ; la valeur de retour
- ❑ Spécification programmatique : utilise un IDL (Interface Description Language)
- ❑ Beaucoup d'implémentations (souvent incompatibles)

Echanges  
entre *stubs*

## ■ Variantes de RPC

- ❑ XML-RPC : appel encodé en XML encapsulé dans une requête HTTP (1998)
- ❑ JSON-RPC : appel encodé en JSON encapsulé dans une requête HTTP (2005)
- ❑ gRPC : concept étendue par Google utilisant HTTP/2 (2015)

**Même une vieille technologie peut être utilisée  
aujourd'hui par des millions de machines**

# ONC RPC – Caractéristiques – API

## ■ Standard normalisé par IETF

- ❑ ONC RPC = Open Network Computing RPC
- ❑ RFC 5531 (màj 2009) + RFC 2695 et 7861 pour les aspects sécurité
- ❑ Anciennement : Sun RPC mise au point et utilisé pour NFS
- ❑ Standard basé sur Unix et le langage C (portage Windows)
- ❑ Multithreading possible côté serveur  
= un thread créé et exécuté par requête du client
- ❑ Serveur RPC frontal commun écoute sur le port 111 (UDP et TCP) : *rpcbind*
  - Le client requête le serveur *rpcbind*
  - *rpcbind* peut retransmettre au client l'adresse du service RPC adéquat enregistré auparavant ou retransmettre l'appel au service RPC
- ❑ Utilise l'IDL nommé RPCL (RPC Language)

## ■ Les APIs

- ❑ Plusieurs APIs sont disponibles permettant plusieurs niveaux de contrôle et de configuration
- ❑ Elles permettent :
  - d'enregistrer une procédure distante (côté serveur)
  - d'appeler une procédure distante (côté client)
  - créer une connexion au serveur
  - créer un serveur RPC
  - etc.

# ONC RPC – RPC Language et XDR

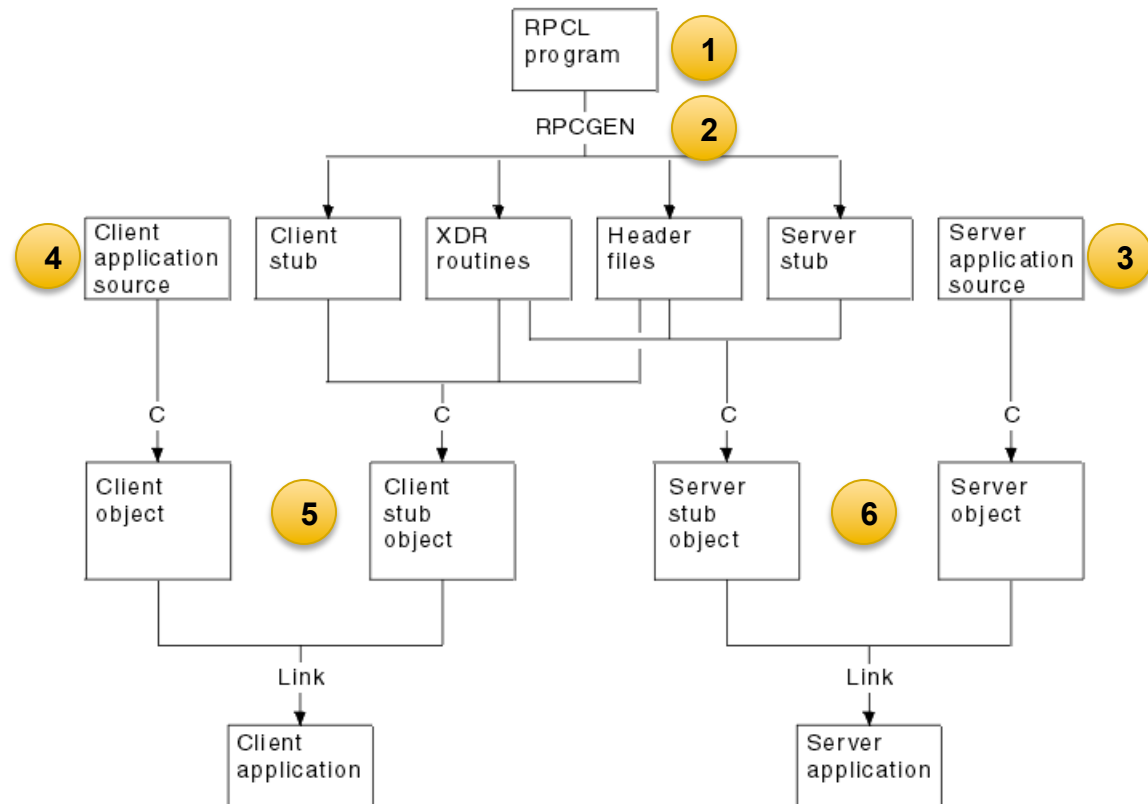
- RPCL est une extension de XDR ; la syntaxe est proche du langage C
  - RPCL est utilisé pour spécifier :
    - le nom des procédures distantes
    - le type du paramètre
    - le type de la valeur de retour
- Historiquement  
ONC RPC autorise  
1 paramètre et  
1 valeur de retour  
uniquement**
- XDR (External Data Representation)
    - Norme IETF 4506 (màj 2006)
    - Format pour encoder des données indépendamment de la machine et de l'OS
    - Encodage en série de 4 octets minimum
    - Données encodables (en multiple de 4 octets) :
      - booléen, entiers (32 ou 64 bits), float, double et quadruple
      - structure, string
      - tableau de taille fixe et variable
      - opaque (= série d'octets non interprétés) de taille fixe ou variable
      - etc.
  - Le fichier de spécification permet d'identifier de manière unique une procédure distante par un triplet :
    - Numéro du service RCP (≠ serveur RPC frontal)  
Ce numéro doit être dans la plage réservée [0x20000000, 0x3FFFFFFF]
    - Numéro de version (=> on peut implémenter simultanément plusieurs versions)
    - Numéro de la procédure distante

# ONC RPC – Exemples (1/3)

- La programmation RPC est simplifiée par l'outil **rpcgen** qui permet :
  - ❑ la génération automatique des fichiers stubs
  - ❑ la génération du fichier d'entête et de filtre communs au client et au serveur
  - ❑ la génération possible des squelettes du client et du serveur
  - ❑ la génération possible d'un fichier Makefile
  - ❑ etc.

- Démarche de développement

- 1) Ecrire le fichier RPCL de spécification
- 2) Générer les stubs et les fichiers communs [ + les squelettes client et serveur ]
- 3) Ecrire le serveur
- 4) Ecrire le client
- 5) Compiler le client
- 6) Compiler le serveur



# ONC RPC – Exemples (2/3)

- Exemple d'une procédure distante (simple) du max de deux entiers

Spécification du paramètre

Spécification de la valeur de retour

Spécification du serveur et des procédures distances

```
struct data {  
    unsigned int arg1;  
    unsigned int arg2;  
};  
typedef struct data data;  
  
typedef unsigned int reponse; /* typedef obligatoire */  
  
program CALCUL { /* Noms en MAJUSCULE par convention */  
    version VERSION_UN {  
        void CALCUL_NULL(void) = 0; /* Test par convention */  
        reponse CALCUL_MAX(data) = 1; /* N° de la procédure */  
    } = 1; /* N° de version du serveur */  
} = 0x20000001; /* N° du serveur */
```

Fichier de spécification  
(extension .x par convention)  
*calcul.x*

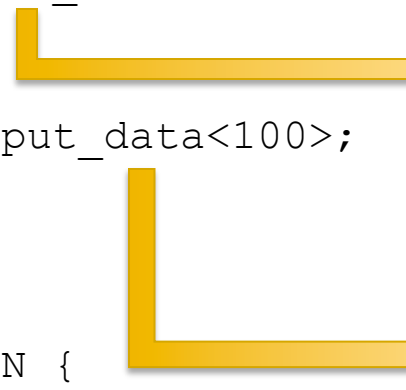


# ONC RPC – Exemples (3/3)

## ■ Autre exemple : les tableaux de taille variable

### Spécification RPCL

```
typedef double input_data<100>;  
  
typedef double output_data<100>;  
  
program MAP {  
  version VERSION_UN {  
    output_data MAP(input_data) = 1;  
  } = 1;  
} = 0x20000001;
```



### Génération en langage C par *rpcgen*

```
typedef struct {  
  u_int input_data_len;  
  double *input_data_val;  
} input_data;
```

```
typedef struct {  
  u_int output_data_len;  
  double *output_data_val;  
} output_data;
```