

Présentation générale

Les informations contenues dans ce document sont susceptibles d'être modifiées sans préavis dans le cadre de l'évolution du produit. Le logiciel décrit dans ce document est cédé dans le cadre d'un accord de licence et ne peut être utilisé ou copié que selon les stipulations de ce contrat. A moins d'être spécifiquement autorisée aux termes de l'accord, la reproduction du logiciel sur quelque support que ce soit est illégale. Toute copie ou transmission de ce manuel sous quelque forme, à quelque fin ou par quelque procédé électronique ou mécanique que ce soit, photocopie et enregistrement compris, sans le consentement écrit exprès de Nat System est interdite par la loi.

© 2006 Nat System. Tous droits réservés.

Le nom et le logo Nat System, NatRcs, NatStar et NatWeb sont des marques déposées de Nat System, Inc. Toutes les autres marques citées dans cet ouvrage sont déposées par leur auteur.

Table des Matières

| | |
|---|-----|
| A propos de ce manuel..... | iv |
| Organisation du manuel | iv |
| Conventions | iv |
| Chapitre 1 Présentation de NatRcs | |
| Qu'est-ce que NatRcs ? | 1-3 |
| Limitations | 1-4 |
| Appels de fonctions à travers NatRcs | 1-5 |
| Chapitre 2 Acteurs de l'architecture NatRcs | |
| Présentation des acteurs..... | 2-3 |
| Process client NatStar | 2-3 |
| Process serveur HTTP ou Web | 2-3 |
| Process listener serveur applicatif NSRpcl..... | 2-3 |
| Process superviseur NWSERVER | 2-4 |
| Process Base de données..... | 2-5 |
| Modules et concepts..... | 2-6 |
| Module de communication NSW2WWTP | 2-6 |
| Protocole Fast CGI | 2-7 |
| Plug-in FastCGI | 2-7 |
| Process NSRpcl et connexion à une base de données | 2-7 |
| Mode Scalabilité..... | 2-9 |
| Contexte..... | 2-9 |

A propos de ce manuel

Ce manuel de Présentation Générale vous aidera à prendre connaissance des notions fondamentales de NatRcs.

Lorsque vous démarrez avec NatRcs, nous vous conseillons de commencer par ce manuel. Une fois que vous aurez acquis une familiarité suffisante avec les concepts et procédures généraux de NatRcs, vous aurez les connaissances nécessaires pour lire le "Guide de Développement" qui documente la phase de développement d'une application NatRcs et le manuel de "Mise en Production" qui documente la phase d'exploitation d'une application NatRcs.

Organisation du manuel

Ce manuel est organisé en deux chapitres.

| | |
|-------------------|---|
| Chapitre 1 | Présentation de NatRcs |
| | Ce chapitre présente succinctement NatRcs. |
| Chapitre 2 | Acteurs de l'architecture NatRcs |
| | Ce chapitre présente les composants de l'architecture d'une transaction NatRcs. |

Conventions

Conventions typographiques

| | |
|---------------------------------|--|
| Terme important | Les termes importants sont imprimés en gras . |
| <i>Composant de l'interface</i> | Les noms de fenêtres, boîtes de dialogue, contrôles, boutons, menus et articles sont imprimés en <i>italique</i> . |
| [F9] | Les noms de touche à utiliser sont indiqués entre crochets. |
| NOM DE FICHER | Les noms de fichier sont imprimés en MAJUSCULES. |
| exemple de syntaxe | Les exemples de syntaxe sont imprimés dans une police de caractères à chasse fixe. |

Conventions de notation

- Le rond est utilisé pour les énumérations.
- ◆ Le losange est utilisé pour les alternatives.
- 1. Les numéros sont utilisés pour marquer les étapes d'une procédure à exécuter séquentiellement.

définition

Une **définition** apparaît avec une présentation spéciale. Elle exprime dans un seul paragraphe la signification d'un terme au singulier. Ce terme apparaît en première colonne, et une seule fois en gras dans la définition.

Conventions de manipulation

Activez la commande Signifie que vous devez ouvrir le menu XXX, puis activer la

| | |
|------------------------------------|---|
| XXX \ YYY ... | commande (le choix) YYY de ce menu. Vous pouvez effectuer cette manipulation à la souris ou au clavier, grâce aux caractères mnémoniques. |
| Activez le bouton XXX \ YYY ... | Signifie que vous devez afficher la barre d'outils de nom XXX, puis cliquer sur le bouton YYY de cette barre d'outils (le nom d'un bouton est indiqué par son info- bulle). Vous ne pouvez effectuer cette manipulation qu'avec la souris. |
| Activez le bouton XXX ... | Dans une boîte de dialogue, signifie que vous devez activer le bouton XXX. Vous pouvez effectuer cette manipulation à la souris ou au clavier, grâce aux caractères mnémoniques. |

Codes d'icônes



Remarque, note...



Renvoi à un autre emplacement dans la documentation.



Indication de **danger** : précaution à prendre, manipulation irréversible, ...



Indication de **conseil** : astuce...



Pour aller plus loin : niveau de détail ou d'expertise supérieur au niveau standard du document.

Chapitre 1

Présentation de NatRcs

NatRcs (Nat System – Rich Client Solution) est une extension payante de NatStar.

Préalablement à toute manipulation, vous devez installer NatStar (l'AGL sur le poste client et le runtime serveur sur le poste serveur).

Vous trouverez dans ce chapitre

- Une présentation générale de NatRcs.
- Comment appeler des fonctions/instructions avec NatRcs.

Table des matières

| | |
|---|-----|
| Qu'est-ce que NatRcs ?..... | 1-3 |
| Limitations | 1-4 |
| Appels de fonctions à travers NatRcs..... | 1-5 |

Qu'est-ce que NatRcs ?

NatRcs (Nat System – Rich Client Solution) est une extension payante de NatStar qui permet de communiquer entre un client NatStar et un serveur à travers les protocoles HTTP ou HTTPS.

NatRcs est constitué de trois outils essentiels :

- le plug-in ADE_RPCL pour faciliter le développement,
- un driver de communication client,

Le module NSW2WWTP est un driver de communication client/serveur qui utilise le protocole http.

- Un process listener RPC, NSRPCL, qui s'exécute sur le serveur. C'est dans ce process que s'exécute les fonctions/instructions Remote développées dans NatStar.

NatRcs permet de :

- déployer automatiquement le code binaire client généré par NatStar en utilisant le protocole de communication HTTP ou HTTPS.
 - assurer les communications entre clients et serveurs en utilisant le protocole de communication HTTP ou HTTPS pour les fonctions Remote NatStar.
 - conserver un cadre de développement NatStar standard en mode N-tiers pour la réalisation.
 - permettre une consultation de statistiques des communications NatRcs réalisées.
-

Limitations

Les fonctions/instructions Remote appelées avec NatRcs ne peuvent pas prendre des paramètres ayant les types suivants : POINTER, INTEGER ou Control.

Appels de fonctions à travers NatRcs

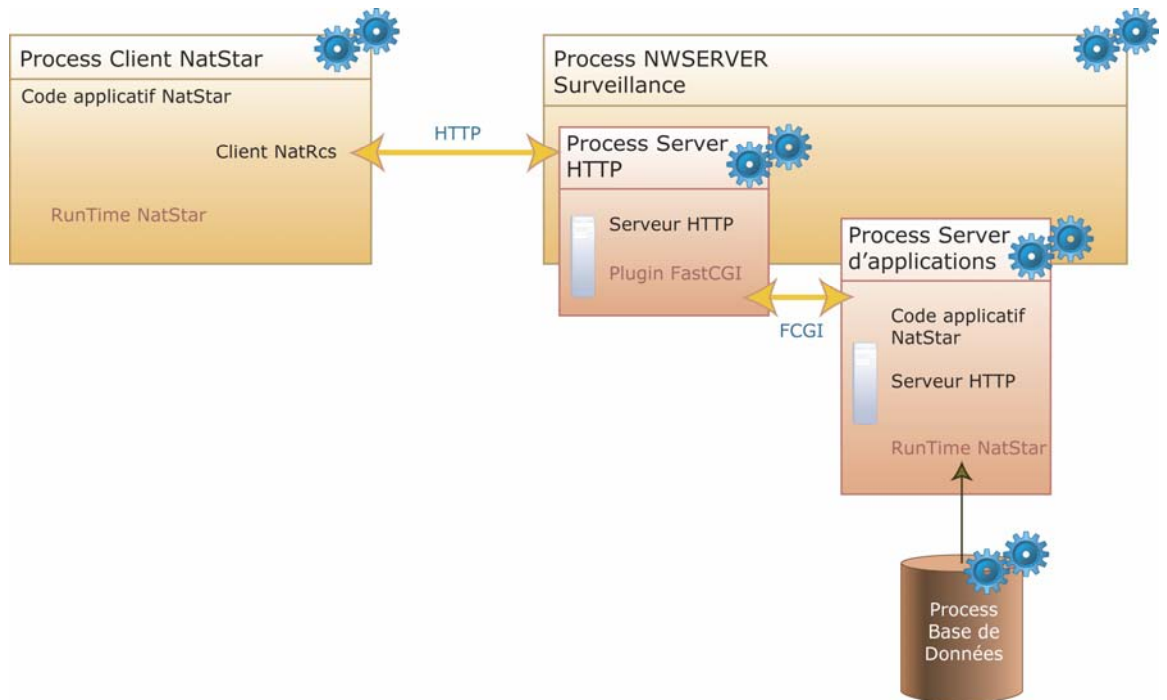


Figure - Appel d'une application NatStar à partir de NatRcs

L'appel distribué passe par les étapes suivantes :

1. Pliage des paramètres en entrée ; seuls les paramètres nécessaires à l'exécution de la procédure (fonction/instruction ou méthode) sont véhiculés,
2. Appel de la couche transport qui se charge de résoudre l'adressage logique en adressage physique,
3. Dépliage des paramètres en entrée dans le processus appelé,
4. Exécution de la fonction distribuée,
5. Pliage des paramètres en sortie (et résultat de la procédure),
6. Retour par la couche transport,
7. Dépliage des paramètres en sortie correspondant au résultat de l'appel de la procédure.

On remarque la définition de paramètres en entrée et en sortie. Les premiers doivent être transportés jusqu'à la procédure Remote, tandis que les secondes doivent revenir à l'appelant après que la procédure Remote ait fixé leur valeur. Un paramètre peut être à la fois en entrée et en sortie.

Chapitre 2

Acteurs de l'architecture NatRcs



Ce chapitre présente les différents acteurs de l'architecture NatRcs.

***Vous trouverez
dans ce chapitre***

- Une présentation des différents acteurs de l'architecture NatRcs.

Table des matières

| | |
|---|-----|
| Présentation des acteurs..... | 2-3 |
| Process client NatStar | 2-3 |
| Process serveur HTTP ou Web | 2-3 |
| Process listener serveur applicatif NSRpcl | 2-3 |
| Process superviseur NWSERVER | 2-4 |
| Process Base de données | 2-5 |
| Modules et concepts..... | 2-6 |
| Module de communication NSW2WWTP | 2-6 |
| Protocole Fast CGI | 2-7 |
| Plug-in FastCGI | 2-7 |
| Process NSRpcl et connexion à une base de données | 2-7 |
| Mode Scalabilité | 2-9 |
| Contexte | 2-9 |

Présentation des acteurs

Ce paragraphe vous propose les définitions des principaux acteurs utilisés dans une architecture NatRcs.

Process client NatStar

Le process client NatStar correspond à l'application développée qui s'exécute sur le poste client.

Process serveur HTTP ou Web

Un serveur HTTP communique avec un client HTTP qui en ce qui nous concerne est une application NatStar. A chaque requête HTTP, le serveur HTTP renvoie une réponse. Les serveurs HTTP les plus couramment utilisés sont Apache et IIS.

Le serveur HTTP sert d'intermédiaire entre le client et le listener applicatif.

Apache 2.0 supporte un mode de compression du contenu HTTP de sa réponse vers le client. Ce mode standard est mis en œuvre par le module mod_deflate.

Dans le cadre d'une communication sur un réseau à faible débit, son activation permet des gains importants à la fois sur la consommation de bande passante mais également en termes de temps de réponse. Ce mode de compression est supporté par NatRcs.




Pour le problème de communication entre le client et le serveur HTTP, reportez-vous à l'erreur NSWWTP-1002 documentée dans le chapitre 7 du manuel "Mise en Production".

Process listener serveur applicatif NSRpcl

NSRpcl est un serveur applicatif qui est en écoute sur un port ou sur un PIPE. Ce serveur reçoit ainsi les requêtes provenant des composants FastCGI, puis déclenche le code utilisateur correspondant à la requête (exécution d'une fonction, instruction ou méthode).

En d'autres termes, le serveur applicatif NSRpcl communique, via l'intermédiaire du serveur HTTP, avec le client NatStar et les serveurs de données. L'application accède aux transactions et données originelles sur la plupart des serveurs (propriétaires et ouverts), exécute votre code NatStar sur le serveur puis renvoie la réponse au client NatStar.

 Pour plus d'informations sur l'absence de réponse de NSRpcl à FCGI, reportez-vous à l'erreur NSWWTP-1007 documentée dans le chapitre 7 du manuel "Mise en Production".

Process superviseur NWSERVER

Le superviseur NWSERVER régule le nombre de serveurs applicatifs NSRpcl qui s'exécute. Il s'assure que les serveurs clones tournent en permanence et se charge de relancer les serveurs applicatifs NSRpcl si ceux-ci sont interrompus. Une mémoire est utilisée pour partager le nombre de sessions en cours (c'est-à-dire le nombre d'instance de l'application NSRpcl) entre NWSERVER et le plugin Fast CGI.

Pour lancer les serveurs applicatifs NSRpcl, exécutez la commande suivante :

```
nwserver -start nwserver=<nom de l'application>
```

Pour arrêter les serveurs applicatifs NSRpcl, exécutez la commande suivante :

```
nwserver -stop nwserver=<nom de l'application>
```



Il y a autant de superviseurs NWSERVER que d'applications NSRpcl.

La syntaxe générale correspond à :

```
NWSERVER NWSERVER=RPCServer [-ENV=Envfile] [-start | -stop | -i | -u]
```

Où RPCServer est une section [FCGI.RPCServer] du fichier NATWEB.INI dont le répertoire est pointé par la variable NS-INI ou l'entrée NS-INI du fichier d'environnement Envfile.

[-ENV=EnvFile] est un paramètre facultatif disponible exclusivement sous Windows qui indique à NWSERVER de lire les paramètres d'environnement à partir d'un fichier d'environnement ou EnvFile dont le nom suit le signe =.

Le fichier d'environnement EnvFile est un fichier texte format DOS 8.3 dans lequel sont consignées des variables d'environnement.

Avec le paramètre -ENV, il est possible d'exécuter plusieurs serveurs RPCL sur une même machine avec des environnements différents comme, par exemple, un premier serveur utilisant une base Oracle avec un runtime NatStar 5.00 et un fichier NATWEB.INI sous D:\INI, un deuxième serveur avec une base DB2 et un runtime NatStar 5.00 et un NATWEB.INI sous C:\NEW\INI et pour terminer, un troisième serveur avec un Middleware Tuxedo, NatStar 5.00 et un fichier NATWEB.INI sous C:\NEW\INI.

Exemple de fichier d'environnement EnvFile :

```
NS-INI=C:\MYLIBS\LIB1  
REM Setting path  
PATH=C:\MYLIBS\LIB1;%PATH%
```



Ne pas mettre de SET devant les variables du fichier d'environnement comme vous le faites pour un fichier bat ou cmd.

La dernière partie des paramètres [-start |-stop|-i|-u] détermine la façon avec laquelle le moniteur NWSERVER et les listeners RPC sont exécutés.

Sans aucun paramètre NWSERVER et le(s) listener(s) RPC sont démarrés comme de simples exécutable, on peut voir leur trace dans la console d'où ils ont été relancés.

-start permet d'installer et de démarrer NSRpcl comme service NT.

-stop permet d'arrêter NSRpcl sans le désinstaller.

-i permet d'installer NSRpcl comme service NT sans le démarrer.

-u permet de désinstaller NSRpcl comme service NT.

Exemple

```
nserver nserver=nsrpcl -env=c:\mylibs\lib1\lib1.env -start
```

Process Base de données

Le serveur applicatif NSRpcl est généralement connecté à une base de données afin d'exécuter les traitements applicatifs.

Modules et concepts

Ce paragraphe présente certains concepts et modules utilisés avec NatRcs.

Module de communication NSW2WWTP

Le module NSW2WWTP est un driver de communication client/serveur qui utilise le protocole HTTP.

Le module NSW2WWTP joue un rôle équivalent au browser. Ainsi, il remplit le champ Browser-Agent de l'en-tête HTTP avec la valeur NatRcs.

Le choix du module de communication par une application client est réalisé avec l'instruction THINGS_INITIALIZE et le paramètre TRANSAC.

Exemple :

```
Things_Initialize( "TRANSAC=NS02WWTP" )
```

Par ailleurs, le module NSW2WWTP utilise la variable d'environnement NATRCS_SERVER_URL pour récupérer les paramètres de connexion à la base de données. Habituellement, cette variable indique une URL où récupérer le fichier HTTP. Mais, il est également possible d'utiliser une syntaxe File au lieu de HTTP.

La variable NATRCS_SERVER_URL pointe vers un fichier .conf définissant la corrélation entre les services NStanza (utilisé par le client NatStar) et l'URL des services NatRcs défini dans le fichier NATWEB.INI.

Exemple d'initialisation de la variable d'environnement NATRCS_SERVER_URL :

```
Set NATRCS_SERVER_URL=http://monserv:8080/alias/natrcs-application.conf
```

Exemple de fichier .conf :

```
<?xml version="1.0" encoding="utf-8" ?>
<natrcs>
  <services>
    <service name="client">
      <url>http://capella:12000/cgi-bin/natrcs.nw/nstest/</url>
    </service>
    <service name="ns_entry">
      <url>http://capella:12000/cgi-bin/natrcs.nw/NS_ENTRY/</url>
    </service>
  </services>
  <client>
    <use-compression>>false</use-compression>
    <httptimeout>120</httptimeout>
  </client>
</natrcs>
```

Il peut également utiliser la variable d'environnement NATRCS_ENVFILE qui indique un fichier texte contenant l'URL. Ce mécanisme peut être utilisé dans le cas d'une URL contextuelle.



Pour plus d'informations sur la variable d'environnement NATRCS_ENVFILE, reportez-vous au manuel "Mise en Production".



Reportez-vous aux erreurs NSWWTP-1015 et NSWWTP-1016 documentées dans le chapitre 7 du manuel "Mise en Production".

Protocole Fast CGI

Fast CGI est un protocole de communication très efficace permettant à deux processus de communiquer. Contrairement au vieux mécanisme CGI, le processus traitant la demande n'est pas démarré à chaque demande ce qui améliore grandement les temps de réponse.

Plug-in FastCGI

Module qui assure la communication entre le serveur HTTP et les serveurs applicatifs NSRpcl.

Ce module tourne dans le processus du serveur Web et utilise le protocole FastCGI pour communiquer avec le serveur NatRcs.

Quand le serveur Web reçoit une requête pour une URL qui correspond à une adresse dont le type correspond à celle du plug-in (l'attribution type d'adresse/plug-in ce fait dans la configuration du serveur HTTP), la requête est passée par le serveur HTTP au plug-in.

Celui-ci analyse la requête afin de la passer après transformation en protocole FCGI au listener du serveur applicatif.

Le plug-in attend la réponse du serveur qu'il recompose au format HTTP afin de la renvoyer au serveur HTTP.

Process NSRpcl et connexion à une base de données

Le serveur applicatif NSRpcl permet de connecter une base de données.

Cette connexion peut être réalisée suivant deux mécanismes différents grâce aux paramètres acceptés par l'exécutable nsrpcl.exe.

Les paramètres **init** et **term** permettent d'indiquer deux instructions NatStar qui seront exécutées à l'initialisation du processus et à sa terminaison.

Les syntaxes sont les suivantes :

init= *lib_name.instr_init* : correspond à une instruction NatStar d'initialisation qui doit se trouver dans la librairie correspondant à **lib_name**. Elle sert à réaliser l'ouverture de la base de données.

term= *lib_name.instr_stop* : correspond à une instruction NatStar de terminaison qui doit se trouver dans la librairie correspondant à **lib_name**. Elle sert à réaliser la fermeture de la base de données.

Il est donc possible d'établir dans ces fonctions une connexion à la base de données.

Les paramètres **dbdll**, **dbbase** et **dbuser** du fichier NATWEB.INI permettent également une définition simple d'une connexion à la base.

- **dbdll** est le nom du driver base de données que l'on souhaite utilisé (paramètre du `sql_init`).
- **dbbase** et **dbuser** sont les mêmes paramètres que ceux utilisés par `sql_open`.
- **dbuser** est donc la chaîne de connexion complète à la base de données (celle-ci dépend de la base de données retenue).

```
nsrpcl dbdll=ns02or102 dbbase=nat dbuser=scott/tiger@service_ora92.world
```

Le paramétrage du lancement des serveurs s'effectue dans le fichier de paramétrage NATWEB.INI.

```
NSRPCL.EXE INIT=librairie.fonct_ini term=libstop.fonct_stop
```

Le module `nsrpcl.exe` accepte également comme paramètre **server** qui permet d'attribuer un nom au process serveur différent du défaut NATSTAR. Ce paramètre indique également au process qu'il doit charger des librairies serveurs (préfixe `t_` sur Unix).

Il est donc indispensable dans un environnement Unix de spécifier ce paramètre.



Il est recommandé d'utiliser comme nom de serveur le nom de la rubrique `fcgi`.

Mode Scalabilité

Plus le nombre de connexions à votre application Web augmente, plus vous observez des problèmes de performance. Ces problèmes de performance sont liés au fait qu'un serveur ne peut traiter qu'un nombre limité de clients en même temps.

Deux solutions sont possibles :

- soit acheter un serveur plus puissant et rapide,
- soit partager la charge de travail entre plusieurs serveurs quand le nombre de clients dépasse la limite d'un seul serveur. Cette deuxième solution s'appelle scalabilité.

Contexte

Le contexte est un jeton permettant d'identifier un client afin de réaliser des statistiques sur le nombre d'utilisateurs.

Quand un utilisateur se connecte, lors de la première transaction un jeton (context) lui est attribué.

Il utilise ce même jeton pour toutes les transactions suivantes avec le même serveur.
