

# Nextra トラブルシューティングガイド

---

Version 6

1<sup>st</sup> Edition



# 目次

<b>第 1 章 はじめに</b> .....	<b>2</b>
本書の利用方法.....	2
表記規則 .....	3
<b>第 2 章 エラーリカバリ</b> .....	<b>5</b>
エラー関数の使用方法 .....	5
ログファイルの使用法 .....	6
致命的エラー .....	6
エラーの変換 .....	7
SQL エラー .....	7
Nextra のエラーメッセージ .....	14
<b>第 3 章 Nextra ユーティリティのトラブルシューティング</b> .....	<b>33</b>
broker.....	33
RPC Developer .....	35
RPCMake.....	35
SQLMake, SQL ファイル .....	36
<b>第 4 章 クライアントのトラブルシューティング</b> .....	<b>37</b>
全てのクライアントについて.....	37
Perl クライアント.....	44
PowerBuilder クライアント (Windows) .....	44
Visual Basic クライアント (Windows) .....	45
<b>第 5 章 サーバのトラブルシューティング</b> .....	<b>47</b>
一般的なトラブルシューティング .....	47

# 第 1 章 はじめに

---

このマニュアルは、Nextra を使用しているときに遭遇する問題、エラーメッセージおよびその対処法をまとめています。また、Nextra オフィシャルページ (<http://www.nextra.jp>) の「Q&A」も参考になるでしょう。

## 本書の利用方法

---

本書『トラブルシューティングガイド』では、Nextra/DB アクセス・モジュール、および Nextra/RPC ライブラリのエラーコードを説明します。また、本書では、Nextra ユーティリティを使用したり、クライアントとサーバのコーディングを行ったりする場合に発生する可能性のある問題、およびエラーメッセージとその対処方法についてもまとめています。まず、対象読者に該当し、本書が必要に応じた適切なマニュアルであることを確認してください。

## 対象読者

---

本書は、3 層分散アプリケーションを扱う全ての開発者と管理者を対象にしており、初心者 of 学習用途には不向きでしょう。特定のエラーコードや状態の問題についての情報を検索する必要のある、3 層分散アプリケーション開発者およびシステム管理者を対象としています。

## 前提知識

---

読者は、Nextra を使用したアプリケーション開発について熟知していることが必要です。本書は、読者が Nextra ユーティリティの使用方法についての概要を理解していることを前提にしています。

## 本書の使用方法

---

『トラブルシューティングガイド』は、次のような情報を検索するために使用します。

- Nextra/DB アクセス・モジュール、Nextra/RPC ライブラリから返された特定のエラーコード
- Nextra ユーティリティ使用中に発生する問題
- クライアントおよびサーバの開発および管理中に発生する問題

本書は、Nextra ドキュメントセットのユーザ向けガイドの補助に使用してください。

## 項目

本書では、次の項目について説明しています。

- エラーリカバリ
- Nextra ユーティリティのトラブルシューティング
- クライアントのトラブルシューティング
- サーバのトラブルシューティング

## 表記規則

### 文中の表記規則

本書で使用する規則を理解しておくこと、ユーティリティの使用方法などを容易に理解できます。

形式	説明	例
terminal	OS やサードパーティのユーティリティ、ファイル名、変数の定数値を示します。	telnet cust.def
sub-text	ユーザが指定する必要がある値を示します。	server_c.pl -e environment_file
bold	本文中では Nextra ユーティリティを示します。サンプル中では、強調される部分を示します。	<b>broker</b> <b>RPCMake</b>
[brackets]	がない場合、オプションテキストを示します。  がある場合、いずれか 1 つを選択することを示します。	[-d def_file] [NONE   ERROR   WARN   DEBUG]

次の形式で区別されているパラグラフはコード例です。

```
#include <stdio.h>

main( ) {
    int i ;
    printf("the number is %d\n",i);
}
```

## 本書で使用するシンボル

---

本書では、次のようなシンボルを使用しています。

	<p><b>警告メッセージ</b></p> <p>このシンボルに続くメッセージに、特別な注意を払う必要があることを示しています。このメッセージには重要な情報が含まれており、この情報を正しく理解してから先に進んでください。</p>
	<p><b>ヒントメッセージ</b></p> <p>このシンボルに続く本文は、必須ではありませんが状況に応じて役立つ手順であることを示しています。</p>
	<p><b>オプションメッセージ</b></p> <p>このシンボルに続く本文はオプションであることを示しています。内容は、追加機能または代替手法の概要、ある概念を理解するために役立つプロセスステップの詳細などです。</p>
	<p><b>デバッグ方法</b></p> <p>このシンボルに続く本文は、プロジェクトの現在のステップをデバッグする手順が含まれていることを示しています。この方法はあくまで参考であり、別の有効なデバッグ方法の使用を妨げるものではありません。</p>

## 第 2 章 エラーリカバリ

---

Nextra/RPC ライブラリは、Nextra ユーティリティ、サーバ、クライアント実行中に発生するエラー条件のトラップ、表示、ロギングを行うための関数を提供します。この章では、その関数の詳細について説明します。

RPC メカニズムのエラートラップについては、RPC の呼び出しの直後にステータスをチェックできます。Nextra を使用する開発者は、自ら設計するよりも、この方式を利用した方が便利であることに気づかれることでしょう。さらに、エラーリカバリ機構を使用する際に発生する問題については、弊社カスタマーサポート担当がより適切に支援させていただきます。

本章では、Nextra エラー処理スキームの機能について説明し、章末には Nextra/RPC ライブラリが生成する可能性のあるエラーのリストを示します。

### エラー関数の使用方法

---

RPC の正常な機能を妨げるエラーは、RPC ライブラリ関数によってトラップされます。一方、RPC ライブラリ関数は、ライブラリ変数に値を割り当てます。RPC を発行する度にこれらの関数を呼び出して、変数の値をチェックするようしておく必要があります。このようにすれば、関数呼び出しの間に発生したどのような異常な条件でも把握することができます。また、このようにすることが、RPC が正常に実行されることを保証するただひとつの方法でもあります。

RPC 実行中にエラーが発生したかどうかを判断するために使う関数は、`dce_errnum()`、`dce_errstr()`、`dce_error()`です。

- `dce_errnum()`は、カレントエラー番号を示す整数を返します。正常時の値は 0 です。
- `dce_errstr()`は、カレントエラーに関連するメッセージへのポインタを返します。関数内部で割り当てを行うため、開発者がメモリを割り当てる必要はありません。正常時のメッセージは、「No error detected」です。
- `dce_error()`は、メモリへのポインタの引数を 1 つとり、カレントエラー番号を示す整数と、エラーに対応するメッセージを返します。ポインタのためにメモリを割り当てる必要があります。たとえば、Visual Basic では、250 文字分のスペースを割り当てなければなりません。正常時に返されるエラー番号は 0 で、メッセージは「No error detected」です。

開発者は、エラー条件を適切に処理するために、エラー関数によって得られる情報を使用してください。dce\_error()実行中に「Out of Memory」が発生した場合、正しいメッセージが返されないこともあります。しかし、エラーの値は正しいので、開発者は実際に発生したエラーを判断することができます。

## ログファイルの使用方法

Nextra ランタイムエンティティ(ブローカ、サーバ、クライアント)のログファイルが環境ファイル属性 DCE\_LOG で指定されている場合には、エラーはこのログファイルにも書き込まれます。実際には、このログファイルにはさらに多くの状態情報を書き込むことができ、詳細レベルは、この環境ファイル属性 DCE\_DEBUGLEVEL の設定によって制御されます。詳細レベルが最も低い場合には、エラーがログに書き込まれるだけです。詳細レベルを高く設定することにより、エラーが発生した場合に限って、そのエラーをデバッグするためのより多くの情報をログに書き込むようにすることができます。

エラーのレポートの詳細については、『リファレンス』の「ログファイル」の節を参照してください。

## 致命的エラー

Nextra ランタイムエンティティが生成するエラーには、致命的とみなされるものがあります。致命的エラーが発生したら、実行しながら解決することはできないので、アプリケーションを終了しなくてはなりません。それ以外のエラーの場合は、RPC を一定回数再試行すると成功することがあります。

dce\_err\_is\_fatal()に致命的エラーを表す 1 がセットされた場合は、DCE\_ERRNUM には値 1 が返されます。次のエラーコードは致命的とみなされます。(エラー番号順に示します。)

エラーコード	番号
DCE_NOPORT	2
DCE_NOHOST	3
DCE_BADHOST	6
DCE_NOSOCKCREATE	8
DCE_NOSOCKBIND	9
DCE_NOSOCKTMOUT	12
DCE_NOSOCKLISTEN	15
DCE_NOMEMORY	17
DCE_CANTOPENLOG	18
DCE_BADSYNTAX	20
DCE_CANTOPENENV	21

DCE_NULLENVFILE	22
DCE_BADTCPINIT	26
DCE_IPCINITBAD	27
DCE_IPCCANTCLOSE	28
DCE_CANTFORK	31
DCE_NOINTERFACE	33
DCE_SIGINT	36
DCE_WRITABLE	42
DCE_MAXCAPACITY	43
DCE_NOSUBB	44
DCE_FSERROR	45
DCE_NODYNINTERFACE	48

その他のエラーコードについては、\$ODEDIR/include/dceerr.hを参照してください。

## エラーの変換

Nextra クライアントは、サーバから DCE\_SERVERFAILED を返されることがあります。これは、クライアントでは修正不可能なエラーがサーバにおいて発生したことを意味します。

DCE\_SERVERFAILED ではなく、クライアントライブラリに返されるサーバエラーコードの値は次の通りです。

DCE_BADINTERFACE	DCE_BADLOGIN	DCE_BADTICKET
DCE_MAX_CAPACITY	DCE_NOAUTH	DCE_NOSEC
DCE_NOSUBB	DCE_NOSUCHFUNC	DCE_SECREQ


## SQL エラー

RDBMS を呼び出す関数は、負の値を返します。このエラー番号は、RDBMS が生成するエラーコードに対応しています。

戻り値 -1 は、DB 内で発生したのではないエラーを示します。SQL ステートメントの実行が正常終了した場合は、戻り値は 0 以上の値になります。0 より大きい値は、SQL ステートメントによって影響を受けた行数を示します。これらのコードは DB 起動ユーティリティによって返されます。

-1 が返された場合は、発生したエラーを調べるために、Nextra エラー関数を呼び出してください。

エラー番号がわかっても対応するエラーメッセージがわからない場合があります。表 2.1 は、エラー番号とメッセージをリストしたものです。

	<p><b>コードの中でエラー番号を使用しないでください</b></p> <p>これらのエラーメッセージ番号は、変更される可能性があります。必ずコードの中ではマクロを使用してください。生成されたログファイルが番号だけを示している場合には、エラーメッセージを調べることが必要になるので、これらのエラーメッセージが提供されています。しかし、ログファイルに頼るよりは、各 RPC の後で <code>dce_errnum()</code> をチェックして、<code>dce_errstr()</code> を使用してユーザに対してメッセージを表示することをお勧めします。なお、他のエラーコードについては、<code>\$ODEDIR/include/dbcommon.h</code> を参照してください。</p>
---	---

次に、`sql_set_max_rowq_interface()` または `sql_prepare_interface()` の呼び出しで返される可能性のあるエラー、エラーの説明および可能な対処方法を示します。

## [エラーコード] DBNOERROR

---

[メッセージ]

Success

[説明]

結果が正常終了の場合のデフォルト設定値。

[対処方法]

必要な処理はありません。

## [エラーコード] DBALREADYCONN

---

[メッセージ]

Already Connected

[説明]

RDBMS への接続が既に確立しているところへ、再び接続を試みた。たとえば、デディケイテッド・サーバに `sql_prepare_interface()` 呼び出しを 2 回実行しようとした。

[対処方法]

クライアントコードを編集して、2 回目の `sql_prepare_interface()` 呼び出しを削除してください。

## [エラーコード] DBCOMMITPEND

---

[メッセージ]

Commit Pending

[説明]

最初のトランザクションをコミットもアボートもせずに、2 回目の `begin_work()` を発行して、2 回目のトランザクションを実行しようとした。

[対処方法]

最初のトランザクションに対して `commit_work()` または `abort_work()` 呼び出しを発行するように、トランザクション・サーバ関数を編集してください。

## [エラーコード] DBCONNDOWN

---

[メッセージ]

Connection Down

[説明]

RDBMS への接続が失われた。通常、このエラーは RDBMS のエラーである。

[対処方法]

DB が動作していることを確認してください。次に、インタラクティブ SQL (Interactive SQL) を使用して、DB に接続を試みてください。ISQL で DB に接続できれば、RPC でのアクセスも可能なはずです。

## [エラーコード] DBCONNREFUSED

---

[メッセージ]

Connection Refused

[説明]

RDBMS がログインを拒否した。

[対処方法]

RDBMS のセキュリティユーティリティをチェックして、この RDBMS にアクセスしようとしているユーザがアクセスを認可されていることを確認してください。

## [エラーコード] DBINVALIDPARAM

---

[メッセージ]

Invalid Parameter

[説明]

関数に無効なパラメータを渡そうとした。たとえば、`sql_set_max_rows_interface()` 関数には 0、または 0 より大きい入力値が必要である。負の数を入力すると、このエラーが発生する。

[対処方法]

関数のパラメータの必要条件をチェックして、有効な値を指定するように関数コードを編集してください。

## [エラーコード] DBLASTREC

---

[メッセージ]

DB Last Record

[説明]

DB 照会の最後のレコードが検索された。Nextra は、このエラーコードを使用して DB カーソルを管理する。

[対処方法]

必要な処理はありません。

## **[エラーコード] DBLOGERROR**

---

[メッセージ]

Log File Error

[説明]

Nextra がログファイルに書き込もうとしていたときに、おそらくメモリ割り当てエラーと思われるエラーが発生した。

[対処方法]

サーバのログファイルで情報をチェックしてください。さらに、ディレクトリの許可が、ログファイルへ書き込めるように設定されていることも確認してください。

## **[エラーコード] DBMISC**

---

[メッセージ]

Miscellaneous Error

[説明]

その他のエラーが発生した。

[対処方法]

サーバのログファイルで情報をチェックしてください。

## **[エラーコード] DBNOCURSORS**

---

[メッセージ]

No Cursors Allowed

[説明]

非デディケイテッド・サーバで `sql_set_max_rows_interface()` 呼び出しを発行して、DB カーソルを使用しようとした。カーソルがクライアントからの複数 RPC を受け付ける可能性があり、他のクライアントから RPC に割り込みがあると、カーソルの状態を変更してしまうことになるため、この処理は実行できない。

[対処方法]

Nextra サーバをデディケイテッド・サーバにするには、環境ファイル属性で `DCE_DEDICATED=N` を設定してください。

## [エラーコード] DBNOMEMORY

---

[メッセージ]

No Memory

[説明]

メモリ割り当てに失敗した。マシンで実行しているプロセスが多すぎるのが原因と考えられる。

[対処方法]

サーバログファイルで情報をチェックしてください。システム管理者とも相談してください。

## [エラーコード] DBNOTRANSAC

---

[メッセージ]

No Transaction

[説明]

コミット、またはアボートしてトランザクションを終了しようとしたが、`begin_work()` 呼び出しでトランザクションを開始していなかった。

[対処方法]

トランザクション・サーバ関数を編集して、`commit_work()`または`abort_work()`を呼び出す前に`begin_work()`を呼び出すようにしてください。

## [エラーコード] DBQUERYNOTFOUND

---

[メッセージ]

Query Not Found

[説明]

DBには理解できないSQLコマンドが入力されたため実行できない。このエラーは、異なるRDBMSで使用された固有のSQLシンタックスを使用すると発生する可能性がある。

[対処方法]

SQLシンタックスをチェックして、現在実行しているRDBMSで使用可能であることを確認してください。

## [エラーコード] DBUSELOGINRPC

---

[メッセージ]

Use Login RPC

[説明]

別のRPCを発行する前に、ログインRPCである`sql_prepare_interface()`を発行して、デディケイテッド・サーバにログインしなかった。たとえば、デディケイテッド・サーバへの最初のRPCが`sql_prepare_interface()`でないと、サーバは環境変数`DB_LOGIN`および`DB_PWD`を使用してRDBMSにログインしようとする。これらの変数でログインできない場合、この後発行されるRPCへの`sql_prepare_interface()`以外の呼び出しには、全てこのエラーが返される。

[対処方法]

クライアントコードを編集して、`sql_prepare_interface()`呼び出しが含まれるようにしてください。

## Nextra のエラーメッセージ

---

ここでは、RPC ライブラリ変数 `DCE_ERRNUM` の値に関連したエラー名とエラーメッセージを示します。RPC ライブラリ関数 `dce_errnum()` はエラー番号を、`dce_errstr()` はエラーメッセージを、`dce_error()` はこの両方を返します。ここでは、エラーそれぞれについての説明、および可能な対処方法も示しています。

### [エラーコード: 34] DCE\_BADARG

---

[メッセージ]

Bad function argument passed

[説明]

関数に 1 つ以上の引数を誤って指定した。指定した引き数が多すぎるか、少なすぎるか、またはフォーマットに誤りがあった。あるいは、領域を割り当てられた文字列、または配列を必要とする関数に `NULL` ポインタを渡した可能性がある。

[対処方法]

関数の正しいシンタックスをチェックし、コードを編集して修正してください。

### [エラーコード: 6] DCE\_BADHOST

---

[メッセージ]

Host name not found in hosts file

[説明]

接続しようとしているマシン、またはブローカを実行しようとしているマシンのホスト名を解決できなかった。

[対処方法]

環境ファイルに `DCE_BROKER` 属性で指定されているホスト名が有効であることを、そのホスト名でホストを ping して確認してください。ホストマシンを ping できない場合には、環境ファイルを編集して `DCE_BROKER` 属性で指定されているホスト名を有効なホスト名に置き換えるか、またはネットワーク管理者に問い合わせてください。

## [エラーコード: 38] DCE\_BADINTERFACE

---

[メッセージ]

Wrong type of server/bad interface

[説明]

クライアントが、誤ったインタフェースでサーバに接続した。このエラーは、サーバがシャットダウンした後、別のサーバが同じポートで起動し、さらにクライアントがそのポートのオーナーとして、以前に最初のサーバをキャッシュしていた場合に限って発生する。

[対処方法]

RPC を再実行してください。クライアントは最新のサーバ位置情報をブローカより取得し、その情報を元にサーバにアクセスすることによってこのエラーを解決します。

## [エラーコード: 18] DCE\_BADLOGIN

---

[メッセージ]

Bad login/pwd combination

[説明]

クライアントプログラムは環境ファイルをロードするために `dce_setenv()` を呼び出したが、*login*、または *pwd* パラメータに誤った引数を与えた。

[対処方法]

*login* パラメータで指定したユーザ名と *pwd* パラメータで指定したパスワードが、NULL であることを確認してください。(言語によっては NULL ではなく、空ストリング("") です。)

## [エラーコード: 32] DCE\_BADPORT

---

[メッセージ]

Server port is in use or invalid

[説明]

サーバまたはブローカにポート番号を入力しなかったか、または無効なポート番号を指定した。

[対処方法]

`netstat` ユーティリティを使用して、そのポートが使用されているかどうかを判断します。次のコマンドを入力してください。

```
netstat -an | grep port_num
```

指定したポートを含む行が画面に表示された場合には、そのポートは使用されています。応答がない場合には、そのポートは使用されていません。必要に応じてブローカの環境ファイルを編集し、`DCE_BROKER` 属性で指定されたポート番号を変更するか、またはサーバ環境ファイルを編集して、`DCE_SERVERPORT` 属性で指定されたポート番号を変更してください。未使用のポート番号の 2048 から 65535 までが有効です。

## [エラーコード: 20] DCE\_BADSYNTAX

---

[メッセージ]

Syntax error in env file

[説明]

指定した環境ファイルに無効なシンタックスが含まれている。

[対処方法]

環境ファイルを編集して、エラーメッセージで指定されているエラーを修正してください。環境ファイルキーワードの綴りの誤り、句読点および `DCE_BROKER` 行でのポート番号の欠落のような、キーワードへの引数の欠落などの誤りを探します。

## [エラーコード: 26] DCE\_BADTCPINIT

---

[メッセージ]

TCP/IP task could not initialize

[説明]

おそらく TCP/IP が正しく設定されていないことが原因で、net\_taskInit()呼び出しに失敗した。

[対処方法]

設定情報については、TCP/IP のドキュメントを参照してください。

## [エラーコード: 35] DCE\_BADTICKET

---

[メッセージ]

Security ticket rejected

[説明]

サーバがブローカの登録に成功すると、ブローカはサーバにチケットを割り当てる。クライアントがブローカにコンタクトしてサーバへのアクセスを要求すると、ブローカはサーバのチケットをクライアントに与える。クライアントはこのチケットをサーバへの呼び出しに含め、サーバはクライアントの要求に応答する前にチケットをチェックする。クライアントが渡したチケットがサーバのチケットと一致しない場合に、Nextra ではこのエラーが発生する。

[対処方法]

クライアントを停止して、再起動してください。

## [エラーコード: 47] DCE\_BADVERSION

---

[メッセージ]

Stub and transport versions conflict

[説明]

Nextra ランタイムライブラリのバージョンと互換性のない **RPCMake** ユーティリティのバージョンで、クライアントまたはサーバのスタブ・スケルトンを生成した。

[対処方法]

**RPCMake** ユーティリティ、および Nextra ランタイムライブラリのバージョンをチェックしてください。必要に応じて、**RPCMake** のバージョンと互換性のあるスタブ・スケルトンを再生成してください。

## [エラーコード: 31] DCE\_CANTFORK

---

[メッセージ]

Failure in fork() when spawning

[説明]

システム制限のため、Nextra ランタイムライブラリは必要なプロセスを生成(フォーク)できなかった。

[対処方法]

システム管理者に相談してください。

## [エラーコード: 21] DCE\_CANTOPENENV

---

[メッセージ]

Could not open env file

[説明]

クライアント、サーバまたはブローカが環境ファイルをオープンできなかった。

[対処方法]

環境ファイルおよび環境ファイルが格納されているディレクトリについて、読み出しアクセスが許可されていることを確認してください。また、正しい環境ファイル名を指定していることを確認してください。

## [エラーコード: 19] DCE\_CANTOPENLOG

---

[メッセージ]

Could not open debug file

[説明]

ブローカ、サーバまたはクライアントが、デバッグ情報をログファイルに書き込めなかった。

[対処方法]

ブローカ、サーバまたはクライアントのログファイルの許可をチェックして、ログファイルには書き込みアクセスが許されていることを確認してください。また、ログファイルが格納されているディレクトリの許可もチェックしてください。

## **[エラーコード: 16] DCE\_CLIENTBUSY**

---

[メッセージ]

Client already processing request

[説明]

Windows クライアントが、最初の RPC が完了する前に次の RPC を実行しようとした。

[対処方法]

クライアントコードを編集して、1 つの RPC が終了するのを待って、次の RPC を発行するようにしてください。

## **[エラーコード: 45] DCE\_FSERROR**

---

[メッセージ]

File system error (possibly full)

[説明]

オペレーティングシステムがデバイスに書き込みもうとしたところ、エラーが返された。「disk full」エラーが返されたことを意味する。

[対処方法]

システム管理者に確認してください。

## **[エラーコード: 57] DCE\_LASTERR**

---

[メッセージ]

unrecognized error

[説明]

このエラーは、Nextra の内部エラー検証に使用される。このエラーがランタイムに返されることはなく、開発者はこれを修正してはならない。

[対処方法]

処理は不要です。

## **[エラーコード: 14] DCE\_LOCALHOSTUNKN**

---

[メッセージ]

Could not find local host name

[説明]

ARPA はローカルホスト名を見つけられなかった。

[対処方法]

hosts ファイル、与えられたホスト名およびネットワーク接続をチェックしてください。

## **[エラーコード: 25] DCE\_LOSTSERVER**

---

[メッセージ]

Lost connection to dedicated server

[説明]

クライアントとデディケイテッド・サーバの接続が困難になった。

[対処方法]

クライアントがデディケイテッド・サーバへの接続を失うと、サーバにある状態情報も失われます。たとえば、デディケイテッド・サーバを使用して、DB テーブルから一度に 10 行

のデータを検索するとします。30 行分のデータを検索した後で、このデディケイテッド・サーバへの接続が失われたとすると、状態情報が失われるため、次の RPC では 1 行目から 10 行目までが返されます。したがって、このエラーをチェックし、RPC を再開始しなくてはならない場合を処理するロジックを、あらかじめ含めておくようにしてください。

## [エラーコード: 43] DCE\_MAXCAPACITY

---

### [メッセージ]

A dedicated master has reached its capacity

### [説明]

デディケイテッド・サーバが、環境ファイル属性 DCE\_DEDICATED を設定して指定した最大数のデディケイテッド・サーバの子プロセスを起動した。

### [対処方法]

クライアントはサーバへの呼び出しを再試行できます。クライアントが終了すると、対応するデディケイテッド・サーバの子プロセスが終了し、ようやくサーバは新しいクライアントに対して新しいデディケイテッド・サーバの子プロセスを起動できます。クライアントで頻繁にこのエラーが発生する場合には、デディケイテッド・サーバの環境ファイルにある DCE\_DEDICATED 属性の設定を変更して、許可されるデディケイテッド・サーバの子プロセスの最大数を増やすことを検討してください。

## [エラーコード: 53] DCE\_NOACL

---

### [メッセージ]

Configuration error - No ACL specified for server

### [説明]

セキュアサーバがセキュアブローカを登録しようとしたが、セキュアサーバのアクセス制御リスト (ACL) にエントリがなかったため登録できなかった。

### [対処方法]

ACL 内にこのサーバのエントリを作成します。ブローカを再起動するか、ブローカで **oecupdate** を実行してください。サーバを再起動してください。

## [エラーコード: 39] DCE\_NOAUTH

---

[メッセージ]

Broker key incorrect

[説明]

ブローカ起動時には、ブローカが認証および認可を実行するために使用するブローカキーを入力する。サーバを起動するときも、同じブローカキーを使用する。このエラーは、サーバがブローカに登録しようとして、そのブローカとは異なるブローカキーを使用した場合に発生する。

[対処方法]

正しいブローカキーを使用して、サーバを再起動してください。

## [エラーコード: 4] DCE\_NOBROKER

---

[メッセージ]

Could not connect to broker

[説明]

クライアントまたはサーバがブローカに接続しようとして失敗した。

[対処方法]

ブローカが実行されていることを確認してください。**AppMinder** を使用して、ブローカを ping することができます。さらに、ブローカを実行しているマシンを自分のマシンから ping できることも確認してください。

## [エラーコード: 48] DCE\_NODYNINTERFACE

---

[メッセージ]

Dynamic servers must specify interface name at start-up

[説明]

バリアブル・ネームド・サーバを起動するときには、環境ファイル属性 `DCE_SERVERNAME` を設定するか、または `-s` コマンド行オプションを使用して、バリアブル・ネームド・インタフェース名にユニークな値を指定しなければならない。

[対処方法]

サーバの環境ファイルを編集して `DCE_SERVERNAME` 属性を含むようにするか、または `-s` コマンド行オプションを使ってサーバを再起動してください。

## [エラーコード: 1] DCE\_NOENVSET

---

[メッセージ]

Environment not set

[説明]

`NULL` の環境ファイルを入力したか、またはクライアントが他の `RPC` ライブラリを呼び出す前に、`dce_setenv()` 呼び出しを発行して環境を初期化しなかった。

[対処方法]

環境ファイルをチェックして、必要に応じて編集してください。クライアントプログラムを編集して、`dce_setenv()` が最初の `RPC` ライブラリ呼び出しとなるように修正してください。

## [エラーコード: 0] DCE\_NOERROR

---

[メッセージ]

No error detected

[説明]

呼び出しは正常終了した。

[対処方法]

処理は不要です。

## [エラーコード: 3] DCE\_NOHOST

---

[メッセージ]

DCE\_BROKERHOST not set

[説明]

ブローカ、クライアントまたはサーバの環境ファイルにブローカホスト、つまりブローカが実行されるマシン名が含まれていない。

[対処方法]

適切な環境ファイルを編集して、ブローカが実行されるマシンのホスト名、または IP アドレスを DCE\_BROKER 属性で指定してください。

## [エラーコード: 17] DCE\_NOMEMORY

---

[メッセージ]

Out of memory

[説明]

メモリ割り当て呼び出しに失敗した。大量のデータを渡そうとしたが、システムがそのデータを十分格納できる大きさのバッファを割り当てられない可能性がある。または、システムのメモリが少なくなっている可能性もある。

[対処方法]

他のアプリケーションやユーティリティが、システムの仮想メモリ全てを使用していないことを確認してください。サーバコードをチェックして、可変長データ型の値を終了するために十分なメモリを割り当てていることを確認してください。たとえば、可変長文字列の最後に、終端文字のための予備バイトを追加する領域を割り当てるようにします。

## [エラーコード: 49] DCE\_NONAMECHANGE

---

[メッセージ]

The interface name for fixed-name servers cannot be changed

[説明]

固定名サーバを起動しようとして、`-s` コマンドラインフラグを指定したか、または環境ファイル属性 `DCE_SERVERNAME` を設定した。この 2 つのオプションは、バリアブル・ネームド・サーバでだけ許可されており、固定名サーバでは使用できない。

[対処方法]

環境ファイルをチェックして、`DCE_SERVERNAME` 属性が含まれていないことを確認してください。`-s` コマンドラインフラグを指定せずにサーバを再起動してください。

## [エラーコード: 2] DCE\_NOPORT

---

[メッセージ]

DCE\_BROKERPORT not set

[説明]

ブローカ、クライアント、またはサーバの環境ファイルにブローカのポート番号が含まれていない。

[対処方法]

環境ファイルを編集して、ポート番号を `DCE_BROKER` 属性の 2 番目の引数として指定してください。2048 から 65535 の未使用のポート番号が有効です。次のコマンドを入力して、ポートが使用されているかどうかを判断してください。

```
netstat -an | grep port_num
```

指定したポートを含む行が画面に表示された場合は、そのポートは使用されています。応答がない場合は、そのポートは使用されていません。

## [エラーコード: 41] DCE\_NOSEC

---

[メッセージ]

Illegal secure connect to insecure environment

[説明]

セキュアクライアントが、セキュリティのかかっていないサーバまたはブローカに接続しようとした。環境ファイルに DCE\_SECURE 行が含まれていないことを確認し、`dce_set_env()`へのログインおよびパスワード引数に NULL、または 0 を渡すことによって、クライアントのセキュリティをオフにできる。

[対処方法]

サーバがセキュアになるようにするには、サーバの環境ファイルを編集して、属性 DCE\_SECURE=1 が含まれるようにしてください。

## [エラーコード: 5] DCE\_NOSERVER

---

[メッセージ]

Could not connect to server

[説明]

要求されたタイプのサーバは使用できない。RPC を使用するためのセキュリティ許可がないか、または起動されていないアプリケーションサーバがある。

[対処方法]

サーバが実行されていることを確認し、必要に応じて再起動してください。RPC の初期設定関数 `dce_setenv()` の引数が正しいことを確認してください。

## [エラーコード: 11] DCE\_NOSOCKACCEPT

---

[メッセージ]

Could not accept

[説明]

TCP/IP レベルでの問題のため、サーバソケットがクライアントの RPC を受け付けるように設定されていない。

[対処方法]

別のマシンに telnet 接続をして、そのマシンから自分のマシンに ping してネットワークをテストしてください。システム管理者にも相談してください。

## [エラーコード: 9] DCE\_NOSOCKBIND

---

[メッセージ]

Could not bind socket

[説明]

サーバまたはブローカを起動しようとしたが、環境ファイルに指定されたポートが無効であるか、別のサーバまたはブローカが使用している。

[対処方法]

netstat ユーティリティを使用して、ポートが使用中かどうかを判断してください。次のコマンドを入力します。

```
netstat -an | grep port_num
```

指定したポートを含む行が画面に表示された場合は、そのポートは使用されています。応答がない場合は、そのポートは使用されていません。必要に応じて、ブローカの環境ファイルを編集して、DCE\_BROKER 属性で指定されたポート番号を変更するか、またはサーバ環境ファイルを編集して、DCE\_SERVERPORT 属性で指定されたポート番号を変更してください。2048 から 65535 までの未使用のポート番号が有効です。

## [エラーコード: 10] DCE\_NOSOCKCONN

---

[メッセージ]

Could not connect to socket

[説明]

TCP/IP レベルでの問題のため、サーバソケットがクライアントの RPC を受け付けるように設定されていない。

[対処方法]

別のマシンに telnet 接続をし、そのマシンから自分のマシンに ping してネットワークをテストしてください。システム管理者にも相談してください。

## [エラーコード: 8] DCE\_NOSOCKCREATE

---

[メッセージ]

Could not create a socket

[説明]

TCP/IP レベルでの問題のため、サーバソケットがクライアントの RPC を受け付けるように設定されていない。

[対処方法]

別のマシンに telnet 接続をし、そのマシンから自分のマシンに ping してネットワークをテストしてください。システム管理者にも相談してください。

## [エラーコード: 15] DCE\_NOSOCKLISTEN

---

[メッセージ]

Could not listen on socket

[説明]

TCP/IP レベルでの問題のため、サーバソケットがクライアントの RPC を受け付けるように設定されていない。

[対処方法]

別のマシンに telnet 接続をし、そのマシンから自分のマシンに ping してネットワークをテストしてください。システム管理者にも相談してください。

## [エラーコード: 13] DCE\_NOSUCHFUNC

---

[メッセージ]

Could not find function

[説明]

クライアントが、現在そのクライアントが接続しているサーバでは提供されていないサーバ関数にアクセスしようとした。このエラーは、通常、IDL ファイルを変更することによって発生する。クライアント・スタブは、サーバ・スケルトンからではなく、IDL ファイルの新しいバージョンから生成されるためと考えられる。

[対処方法]

クライアントコードを編集して、その関数呼び出しを削除するか、またはサーバコードを更新して、インタフェースに追加された関数をサポートするようにしてください。

## [エラーコード: 22] DCE\_NULLENVFILE

---

[メッセージ]

Env file not set

[説明]

dce\_setenv() の最初の引数に 0、または NULL を渡した。

[対処方法]

dce\_setenv() の最初の引数として、有効な環境ファイル名を渡してください。

## [エラーコード: 24] DCE\_PEERERROR

---

[メッセージ]

Problems with peer during transaction

[説明]

RPC 完了前に、クライアントとサーバの接続が切断された。サーバまたはネットワークのいずれかがダウンした。

[対処方法]

サーバのホストマシンを ping することによって、ネットワークをテストしてください。ネットワーク接続が確立できていても RPC が完了できない場合には、サーバプロセスがダウンしているものと考えられるので、サーバの再起動が必要になります。RPC を繰り返し呼び出しても問題がない場合には、クライアントコードを編集して、ユーザが RPC を再実行できるようにしてください。

## [エラーコード: 29] DCE\_READFAIL

---

[メッセージ]

Low-level network read failure

[説明]

RPC 完了前に、クライアントとサーバの接続が切断された。サーバまたはネットワークのいずれかがダウンした。

[対処方法]

サーバのホストマシンを ping することによって、ネットワークをテストしてください。ネットワーク接続が確立できていても RPC が完了できない場合には、サーバプロセスがダウンしているものと考えられるので、サーバプロセスの再起動が必要になります。RPC を繰り返し呼び出しても問題がない場合には、クライアントコードを編集して、ユーザが RPC を再実行できるようにしてください。

## [エラーコード: 46] DCE\_RPCTIMEOUT

---

[メッセージ]

RPC timed out

[説明]

環境ファイル属性 DCE\_RECEIVETIMEOUT に指定された時間が経過したため、サーバ、またはブローカは RPC を発行したクライアントとの接続を切断した。

[対処方法]

必要な処理はありません。サーバはタイムアウト後、別のソケットをオープンします。このエラーが頻繁に発生する場合には、DCE\_RECEIVETIMEOUT の値を増やすことを検討してください。

## **[エラーコード: 37] DCE\_SERVERFAILED**

---

[メッセージ]

Server encountered an error

[説明]

クライアントに渡される一般的なメッセージであり、サーバで、クライアントに影響しないエラーが発生したことを示す。

[対処方法]

エラーの詳細について、サーバのログファイルをチェックしてください。

## **[エラーコード: 36] DCE\_SIGINT**

---

[メッセージ]

Received SIGINT, terminating

[説明]

サーバが割り込み信号を受信した。

[対処方法]

処理は不要です。

## **[エラーコード: 52] DCE\_UNAVAILABLE**

---

[メッセージ]

Unable to confirm server status

[説明]

接続したサーバの状態が確認できません。

[対処方法]

通信タイムアウトなど、何らかの理由によりこのエラーが発生した可能性があります。再度実行してみてください。

## **[エラーコード: 30] DCE\_WRITEFAIL**

---

[メッセージ]

Low-level network write failure

[説明]

TCP/IP は、確立されたソケットから書き込みエラーを返した。

[対処方法]

エラーの詳細について、ログファイルをチェックしてください。

## 第 3 章 Nextra ユーティリティのトラブルシューティング

---

---

この章では、Nextra ユーティリティ使用時に発生する潜在的な問題をリストして、原因の特定と解決策を説明します。

参照を容易にするために、トラブルシューティングの項目を、UNIX 環境のみで発生する問題、および PC 環境のみで発生する問題について注釈をつけています。このように分類されていない問題については、どのプラットフォームにも当てはまります。

なお、ここでの分類は絶対的なものではありません。ここでは、説明の重複を避けるようにしているため、見出しから探して該当する問題が見つからない場合は、別の見出しの部分も参照してください。

### broker

---

#### [エラー] Environment not set

---

[考えられる原因]

環境ファイルが存在しない。

[対処方法]

環境ファイルを作成してください。

環境ファイルのプロンプトに対して、フルパス名を入力してみてください。

#### [エラー] Can't bind socket <###>

---

[考えられる原因]

デフォルトポートが既に使用されているか、使用不可となっている。

[対処方法]

環境ファイル(たとえば `broker.env`)を編集して、ポート番号を変更してください。

ポート番号は、`DCE_BROKER` 属性に設定される 2 番目の値になります。

次の例では、ホストマシンは `UNIX9` で、ポート番号は `21011` です。

```
DCE_BROKER= unix9,21011
```

指定できるポート番号は `2048~65535` の値です。以下のコマンドを入力して、ポートが使用中かどうか判断することができます。

```
netstat -an | grep port_num
```

指定したポートを含むラインが画面に表示されれば、そのポートは使用中です。何も応答が返されなければ、そのポートは使用可能です。

## **[エラー] Can't open log file**

---

[考えられる原因]

存在しないディレクトリに `DCE_LOG` 属性が設定されている。

書き込み許可のないディレクトリに `DCE_LOG` 属性が設定されている。

[対処方法]

`DCE_LOG` 属性に指定したパス名の綴りを確認してください。

指定したディレクトリに書き込み許可があるかどうかを確認してください。書き込み許可がない場合は、別のディレクトリを選択してください。

## **[問題] The Broker's log file is missing**

---

[考えられる原因]

UNIX 環境でブローカを実行しているとき、どの時点かでログファイルが削除されると、たとえブローカ実行中にログファイルを再作成してもエラーは生成されず、ログ情報は失われる。UNIX では、削除したファイルに書き込みができるように見えても、実際には、そのファイルに書き込まれた内容は全て捨てられるためである。たとえ、新しいコピーがすぐに作成されても同じである。

[対処方法]

ブローカ実行中は、ログファイルを削除しないでください。

## RPC Developer

---

### [エラー] Can't open display (UNIX)

---

[考えられる原因]

UNIX 環境で **RPC Developer** の GUI を起動しようとしたが、DISPLAY 環境変数が正しく設定されていない。

[対処方法]

DISPLAY が次のように設定されているかどうか確認してください。

```
DISPLAY=`hostname`:0.0
```

### [エラー] RPC Developer not found

---

[考えられる原因]

**RPC Developer** がファイルパスに存在しない。

[対処方法]

UNIX の場合は、ODEDIR 環境変数が正しく設定されて PATH に含まれているかどうか確認してください。

Windows の場合は、ODEDIR ディレクトリが PATH に含まれているかどうかを確認してください。

## RPCMake

---

### [エラー] Server or client language is illegal

---

[考えられる原因]

存在しない言語オプションを入力した。

[対処方法]

言語オプションの文字列の綴りが正しいかどうか、また、サポートされている言語を指定したかどうか確認してください。

## SQLMake, SQL ファイル

---

### [エラー] Bad interface definition: [each line of .def file]

---

[考えられる原因]

サーバ名にドット(.)が含まれている。

**SQLMake** はこの形式を受け付けるのに問題はないが、**RPCMake** では問題になる。

スタブ生成のために、**SQLMake** の GUI が **RPCMake** を呼び出すので、この問題が発生する。

[対処方法]

ドットを含まないように、サーバ名をリネームしてください。

### [問題] 2つの Nextra/DB アクセス・モジュールが同じ関数を持っている。 複製された照会ファイルを使わないようにできるか？

---

[状況]

同じ動作を実行する 2 つの Nextra/DB アクセス・モジュールがあるが、データは 2 つの異なる DB に格納されている。照会ファイルを異なる関数名で複製するのではなく、クライアントプログラムで 2 つのサーバ/DB を交互に使用したい。

[対処方法]

バリアブル・ネームド・サーバを複数使用してください。詳細については、『運用／設定ガイド』の「バリアブル・ネームド・サーバの概要」を参照してください。

## 第 4 章 クライアントのトラブルシューティング

---

---

この章では、さまざまな状況で発生しうるクライアントの問題を示し、原因の特定と解決について説明します。

ここでは、UNIX、Windows で発生する問題については注釈を付けています。

なお、ここでの分類は絶対的なものではありません。ここでは説明の重複を避けるようにしているため、見出しから探して該当する問題が見つからない場合、別の見出しの部分も参照してください。また、Nextra オフィシャルページ(<http://www.nextra.jp>)の「Q&A」も参考になるでしょう。

### 全てのクライアントについて

---

**[問題]** 多数の RPC 呼び出しを短い時間に行うと、時折 "Cannot connect to Broker/server" というエラーを生成する。

---

[考えられる原因]

クライアントが呼び出しを行うたびに、PC は新しいソケットを必要とする。呼び出しが完了すると、ソケットは解放されるが、ただちに解放されるわけではない。このため、多数の呼び出しを短い間に行うと、ソケットを使い切ってしまうこともある。この状態では一時的だが、どのマシンにも接続できない。

[対処方法]

一定の間隔で RPC を発行するように変更する。または、1 つの RPC でより多くの処理を行うようにサーバ側のプログラムの見直しを行ってください。

**[問題]** 同時に RPC を発行すると "busy" メッセージを引き起こす。

---

[考えられる原因]

RPC 間で `pause` 文がインプリメントされている必要がある。

[対処方法]

各 GUI インプリメンテーションの詳細については、『クライアント開発者ガイド』を参照してください。

---

**[問題] PC クライアントから、サーバまたはブローカに接続できない。**

[考えられる原因]

環境ファイルに指定されたホスト名を PC が認識していないか、またはネットワークに異常がある可能性がある。クライアントが通信しようとしているサーバプロセスが実行されていない可能性もある。

[対処方法]

ホスト名が PC の C:\WINNT\system32\drivers\etc\に格納されている hosts ファイルにエントリがあることを確認してください。

また、ping のようなユーティリティを使用して、ネットワークをチェックしておくとういでしょう。

---

**[問題] PC が hosts ファイルを読み込もうとしたときにエラーが発生する。**

[考えられる原因]

hosts ファイルの最後の行の終わりに改行文字がないとパーサーが正しく動作しない。

[対処方法]

hosts ファイルの最後の行の終わりに改行文字があるかどうか確認してください。

---

**[Find server エラー] <ERROR: Could not connect to Broker <#>>**

[考えられる原因]

ブローカが実行されていない。

環境ファイルが誤ったブローカを指している。

ネットワーク接続に問題がある。

ソケットの最大数を越えた。

[対処方法]

ブローカを手動で起動するか、**AppMinder** を使ってアプリケーションを起動してください。

PC 上の環境ファイルを編集し、正しいブローカホストとポートを指すようにしてください。

ping ユーティリティを使って、ホスト間の接続を確認してください。ping が正常終了した場合は、アプリケーションを再実行してください。ping でエラーとなった場合は、物理的な接続を確認してください。

PC を再設定し、より多くのソケットを扱えるようにしてください。再設定をしてもうまくいかない場合は、PC をリブートしてください。

## **[Find server エラー] <ERROR: Could not connect to server <#>>**

---

[考えられる原因]

サーバが実行されていない。

環境ファイルが誤ったブローカを指している。

サーバがブローカ階層内に登録されていない。

[対処方法]

サーバが起動されているかどうか確認してください。サーバが起動されていない場合は、サーバを手動で起動するか、または必要なサーバを監視するように **AppMinder** を再設定してください。

ブローカの正しいホストとポートを反映するように、PC 上の環境ファイルを編集してください。

サーバが起動時に参照する環境ファイルをチェックしてください。ブローカ、ブローカに接続されているサブブローカ、またはブローカのマスタブローカにサーバを登録しなければなりません。サーバがこの階層構造にない場合は、環境ファイルを編集して、サーバがブローカの管理下に含まれるようにしなければなりません。

## [エラー] Unrecoverable Application Error (Windows)

---

### [考えられる原因]

予約語がクライアント・スタブの中で使用されている。

DLL 中の関数が、誤ったフォーマットの引数を受け取った。たとえば、パーサー関数が複数の引数の間にある“|”に困惑してしまった場合など。

シンタックスまたは文字入力のエラーである。

### [対処方法]

IDL ファイルで、予約語を非予約語に変更してスタブを再生成してください。

クライアントから DLL に送られるフォーマットが、DLL に認識されるものであることを確認してください。RPC のシンタックスをチェックしてください。

文字入力のエラーによって **Unrecoverable Application Error** が発生する場合があります。また、文字列に存在しない変数を関数が探す場合があります。クライアントから DLL に送る引数をチェックして、正しい綴りで入力されていることを確認してください。

## [エラー] Out of Memory

---

### [考えられる原因]

クライアントコードで `dce_release()` を使用していない。

### [対処方法]

クライアント・スタブは、`dce_malloc()`、`dce_calloc()`、`dce_realloc()` を使用してメモリを割り当てます。スタブからデータを取得した後、クライアントはそれまで使用していたメモリを解放しなければなりません。`dce_release` 関数は、前回の `dce_release` 関数以降、`Nextra` 関数が割り当てた全てのメモリを解放します。ユーザ関数内部でメモリ解放ができる場合には、`malloc()` または `calloc()` を使用することも可能です。

**[問題] クライアントからサーバに BLOB (Binary Large Objects)を送るとき、最初の NULL キャラクタ以降を切り捨てる。**

---

[考えられる原因]

BLOB は、*void*の配列として渡さなければならない。

[対処方法]

コードをチェックして、BLOB が *void*の配列としてクライアントからサーバへ渡されていることを確認してください。

### **[問題] サーバが「誤った」RPC を実行する。**

---

[考えられる原因]

クライアントおよびサーバのスタブ・スケルトンに互換性がない。

[対処方法]

クライアントとサーバの両方についてスタブ・スケルトンを再生成して再リンクしてください。

### **[問題] 同じ機能を持つ 2 つの Nextra/DB アクセス・モジュールがある。複製された関数名を使わない方法はあるか？**

---

[状況]

同じ動作を実行する 2 つの DB があるが、データは 2 つの異なる DB に格納されている。照会ファイルを異なる関数名で複製するのではなく、クライアントプログラムで 2 つのサーバ/DB を交互に使用したい。

[対処方法]

バリアブル・ネームド・サーバを使用してください。詳細については『運用／設定ガイド』の「バリアブル・ネームド・サーバの概要」を参照してください。

### **[問題] Windows 用 C クライアントをどのように作成するか？**

---

[対処方法]

C コンパイラの `implib` ユーティリティを使って、ファイル `librpc.lib` を作成して、ソースコードに `Nextra DLL` をリンクしてください。( `implib` については、コンパイラのマニュアルを参照してください。)

プロジェクトファイル(Borland コンパイラでは、`.prj` ファイル)に、`.lib` ファイルを取り込み、ソースにリンクしてください。

## **[問題] クライアントログファイルが失われる。**

---

### [考えられる原因]

クライアントが UNIX 環境で実行されている場合、ログファイルが削除されると、クライアント実行中に再度ログファイルを作成しても、エラーは生成されず、ログ情報は失われる。UNIX では、削除したファイルに対して書き込みを続けられるように見えても、実際には書き込んだ情報は全て失われるためである。たとえ、直ぐに新しいファイルを作成し直しても同じである。

### [対処方法]

クライアント実行中は、ログファイルを削除しないでください。

## **[問題] クライアントがサーバを認識した後、ネットワークの不調により通信が切断される。**

---

### [対処方法]

このとき、クライアントがタイムアウトを判定するまで待ち時間があるので、これを短くする。クライアント環境ファイル(`client.env`)において以下の環境変数を設定します。例えばタイムアウトまでの判定を 10 秒としたい場合、

```
DCE_CLN_TIMEOUT=10
```

のように設定します。

`DCE_CLN_TIMEOUT` は、クライアントがサーバに送った RPC の応答を待つ時間(単位:秒)を設定します。サーバからのレスポンスが設定した時間(上の例の場合は 10 秒)までに得られない場合、クライアント側から接続を切ります。

- ・上記現象発生時に再度接続を試みる方法

C 言語のプログラムでの例を示します。

```
#define RETRY 3 /* 3回接続を試みる場合 */

for(i=0;i<RETRY;i++){
    result=RPC_function(argument); /* RPC の結果を result に入れる */

    if(dce_errnum()==0) break; /* 正常に終了したら再接続のループから抜ける */

    /* RPC のエラーのうちサーバに接続できたが応答がないというエラー以外のときは
    エラーメッセージを出力して終了 */
    else if(dce_errnum() !=DCE_RPC_TIMEOUT){
        printf("Error: %s\n", dce_errstr());
        exit(1);
    }
    if(i==RETRY-1){ /* RETRY 回接続を試みてもだめだった場合 */
        printf("Error: %s\n", dce_errstr());
        exit(1); /* エラーメッセージを出して終了 */
    }
    sleep(20); /* 20 秒待つて再接続。秒数はサーバの処理時間を考慮して調整してくだ
    さい。 */
}
```

\* 上記プログラム中の「DCE\_RPC\_TIMEOUT」は サーバに接続はしたが、応答が帰ってこない場合の RPC エラーのエラー番号を示します。

## Perl クライアント

---

### **[問題]RPC 上の呼び出しが予測しない結果を返す。**

---

[考えられる原因]

Perl は常に変数属性リストを使用する。たとえば、パラメータが 10 個しかない RPC に引数を 20 個指定すると、後の 10 個の引数は無視される。

[対処方法]

Perl のクライアントコードを編集して、RPC に対する呼び出しで正しい数の引数が含まれるようにしてください。

## PowerBuilder クライアント (Windows)

---

### **[問題] PowerBuilder フロントエンドが実行されない。**

---

[考えられる原因]

Nextra 関数を取り込まれていない。

[対処方法]

アプリケーションに Nextra PowerBuilder 関数を取り込んでください。詳細については、『クライアント開発者ガイド』の「必要なファイルのロード」を参照してください。

### **[エラー] Out of bounds error when retrieving rows from a database**

---

[考えられる原因]

行が返されなかったため、RDBMS が NULL を返した。クライアントコードが、配列の下限をチェックせずに、値のあるものとしてスタブに渡している可能性がある。代わりに、クライアントはリストボックス中の配列要素をただちに置くため、配列境界条件が発生する。

[対処方法]

クライアントが配列に対して LowerBound を実行するようにしてください。この結果が有効であれば、リストボックスに要素を置いてください。そうでなければ、その配列を処理しないでください。

## Visual Basic クライアント (Windows)

---

### [エラー] Could not set environment

---

[考えられる原因]

dce\_setenv() の呼び出しに誤りがある。

[対処方法]

dce\_setenv() の呼び出しが含まれているフォーム、またはイベントを探し、コードのシンタックスをチェックしてください。

全てのプログラムで、dce\_setenv() の呼び出しは、RPC の前に呼び出されなければなりません。

正しいパスと環境ファイル名が、dce\_setenv() の呼び出しに含まれていることを確認してください。パス名が誤っている場合には、呼び出し引数または環境ファイルの位置のいずれか適切な方を変更することができます。

### [問題] VB スタブが正しくロードされないが、エラーメッセージは現れない。

---

[考えられる原因]

IDL ファイル中の関数パラメータに「name」のような Visual Basic キーワードが含まれている。

[対処方法]

IDL 中の関数パラメータに、Visual Basic キーワードが含まれないようにしてください。

### [エラー] Parameter type mismatch

---

[考えられる原因]

シンタックスエラー。通常はクライアント・スタブのシンタックスエラーである。

[対処方法]

引数の型と対応するパラメータの型が一致しているかどうかを確認してください。一致していない場合は、状況に応じて、引数かパラメータを変更してください。パラメータの型を変更する場合は、対応する IDL ファイルも変更しなければなりません。これは、**RPCMake** を使って、クライアント・スタブを再生成しなくてはならないことを意味します。そして、新しいクライアント・スタブをフロントエンドに取り込みます。

Visual Basic では、クライアント・スタブ内で、関数名、またはパラメータ名は予約語になります。関数名かパラメータを変更したら、プロシージャファイル、または IDL ファイル中で対応する名前も変更しなくてはなりません。そして、**RPCMake** を使ってスタブを再生成して、新しいクライアント・スタブをフロントエンドに取り込みます。スタブを取り込むときに、Visual Basic がプロシージャごとにスタブを扱うのではなく、1 つのエントリとしてスタブを扱う場合、予約語は正しく使われません。

**[問題] 文字列を配列から直接出力する場合、出力される 2 番目の文字列が 1 番目の文字列より短い場合、1 番目の文字列から文字が追加される。**

---

[考えられる原因]

Visual Basic は、文字列の最後を意味する NULL 文字を認識しません。

[対処方法]

出力時、および連結時の問題を回避するために、NULL 文字を含む文字列を明示的にチェックして、NULL 文字を見つけたら残りの文字列を捨てる関数を作成してください。

**[問題] Visual Basic クライアントでログファイルが空である。**

---

[考えられる原因]

Visual Basic 実行中、ログファイルは開かれたままになっている。Visual Basic を実行している途中でログファイルを読み取ろうとすると、ログファイルは開かれたまま読み取られることになるが、これは DOS では適切ではない。

[対処方法]

ログファイルを読み取ろうとする前に、Visual Basic を終了してください。

## 第 5 章 サーバのトラブルシューティング

---

この章では、さまざまな状況で発生しうるサーバの問題を示し、原因の特定と解決について説明します。

ここでは、UNIX、Windows でのみ発生する問題については注釈を付けています。このように分類されていない問題については、どのプラットフォームにも当てはまります。

なお、ここでの分類は絶対的なものではありません。ここでは、説明の重複を避けるようにしているため、見出しから探して該当する問題が見つからない場合は、別の見出しの部分も参照してください。また、Nextra オフィシャルページ (<http://www.nextra.jp>) の「Q&A」も参考になるでしょう。

### 一般的なトラブルシューティング

---

#### [問題] COBOL サーバランタイムエラー:

---

```
Execution error : file 'curtdb_c'  
error code: 114, pc=0, call=5, seg=0  
114 Attempt to access item beyond bounds of memory (Signal 11)
```

[考えられる原因]

IDL ファイルおよび対応する COBOL サイズファイルの両方で、配列サイズが冗長にハードコードされている。

[対処方法]

配列サイズは、COBOL サイズファイルだけで設定してください。

#### [問題] AppMinder でサーバを起動できない。

---

[考えられる原因]

AmViewer でサーバの起動コマンドが正しく記載されていない可能性があります。また、起動するサーバプログラムのパス設定が正しくない可能性があります。

AmViewer は、コマンドラインの認識をスペースで行っています。その為、コマンドの始まりから最初のスペースまでをサーバ名と判断し、その後、スペースで区切りを判断し、コマンドラインを認識します。

たとえば、下記のようにサーバ名の前に逆引用符などを追加した場合、AmViewer はサーバ名が ``server_name`` であると判断し、環境ファイル名を `server.env`` と判断します。その為、このコマンドラインではサーバを起動できません。

```
`server_name -e server.env`
```

[対処方法]

AmViewer のサーバインスタンス設定編集の画面で、サーバの起動コマンドが正しく記載されているか、サーバプログラムまでのパスが正しく設定されているかどうかの確認を行ってください。

## **[問題] AppMinder でバリアブル・ネームド・サーバが起動できない。**

---

[考えられる原因]

AmViewer で正しくサーバの設定ができていない可能性があります。

[対処方法]

AmViewer でバリアブル・ネームド・サーバを起動する為の設定は下記の 2 通りがあります。

\* **Interface Name:** の記入 (必須) と **Variable Named Server:** のチェックボックスをチェックする。

\* **Interface Name:** の記入 (必須) とサーバ起動 コマンドに `-s` オプションを追加する。

```
server_name -e server.env -s interface_name
```

前者の場合、環境ファイル中に `DCE_SERVERNAME` の値が設定されます。

後者の場合、環境ファイル中に `DCE_SERVERNAME` の値は設定されません。

通常のサーバのインタフェース名は、IDL ファイル中に設定した値を使用しますが、バリアブル・ネームド・サーバでは、インタフェース名をサーバ起動時に決定できます。

## **[問題] 環境が設定されない。**

---

[考えられる原因]

環境ファイルが作成されていない。

環境ファイルに誤りがある。

[対処方法]

環境ファイルを作成して、必要な属性を設定してください。

環境ファイルの形式とシンタックスが正しいかどうか確認してください。DCE\_BROKER、DCE\_LOG および DCE\_DEBUGLEVEL 属性が正しく割り当てられているかどうか確認してください。

## **[エラー] Out of memory**

---

[考えられる原因]

サーバでメモリを割り当てするのに `dce_malloc()` を使っていない。

[対処方法]

可変サイズの出力には `calloc()` または `malloc()` ではなく、`dce_malloc()` または `dce_calloc()` を使用してください。サーバ・スケルトンがメモリからデータを読み込み、前の `dce_release()` 以降割り当てられた全てのメモリを解放するために `dce_release()` を実行します。

## **[問題] サーバがゴミを返す。**

---

[考えられる原因]

データをスケルトンに渡す前に、割り当てられたメモリを解放している。

[対処方法]

`calloc()` / `malloc()` ではなく、`dce_malloc()` / `dce_calloc()` を使用してください。サーバコード中ではメモリを解放しないでください。

サーバ・スケルトンがメモリからデータを読み込み、前の `dce_release()` 以降割り当てられた全てのメモリを解放するために `dce_release()` を実行します。

---

**[問題] Perl サーバが"\n"で終わる文字列のリストを返す。**

---

[考えられる原因]

出力に C の `new line` 文字が含まれている。

[対処方法]

`chop` 文を使って、`\n` をスペースに置き換えてください。

---

**[問題] 2次元配列を C サーバに戻すとき、時折、以前の呼び出しでのデータが、現在の関数呼び出しに追加されてしまう。**

---

[考えられる原因]

引数のために固定サイズのメモリを再割り当てする際に、以前に値が入っていたスペースをサーバが割り当てている。新しい関数呼び出しが、前の関数呼び出しより少ないデータを返す場合、メモリ中で古いデータの全てが上書きされない。

[対処方法]

関数の返り値が、返される配列要素数になるようにコーディングしてください。これで、クライアントは配列要素数だけを表すようになります。

---

**[問題] サーバがクライアントでもある場合、どこに `dce_malloc()` と `dce_release()` を置けばよいのか？**

---

[状況]

クライアント A がサーバ B に対して呼び出しを行う。一方、サーバ B は、サーバ C に対してはクライアントとして動作するものとする。

[対処方法]

この場合、サーバ C は、応答を作成する機能を持つので、メモリスペースについて `dce_malloc()` を実行しなくてはなりません。

クライアント A は、応答の最終的な受け取り手なので、メモリについて `dce_release()` を実行しなくてはなりません。

サーバクライアント B は、このプロセスの中間に位置するので、`dce_malloc()` / `dce_release()` を使用する必要はありません。

サーバクライアント B は、サーバ C からの応答を保管して、クライアント A に直接渡します。

## **[問題] C サーバに対して呼び出しを行うとき、メモリエラーとなる。**

---

[考えられる原因]

呼び出されているサブルーチンが、終了するまで実行していないか、またはサブルーチンは終了するまで実行しているが適切でない (つまり、IDL ファイルとは異なった) 形式で値を返す。

[対処方法]

デバッグ文を C コードに挿入して、終了するまで実行するようにしてください。挿入する文は次のとおりです。

```
dce_log(ERROR, "This is a debugging string\n");
```

引用符に囲まれている文は、デバッグログに出力されます。これらの文を必要なだけ挿入して、デバッグ文字列がディスクリプタになるようにします。また、この文がリターン文の直前に挿入されていることを確認してください。これは、エラーが発生した場所を特定するのに役立ちます。ルーチンが終了するまで実行している場合には、1 つまたは複数の値が誤って返されます。出力引数をいくつか削除してみて、どの引数が原因になっているかを調べてください。

## **[問題] 同じ機能を持つ 2 つの Nextra/DB アクセス・モジュールがある。複製された関数名を使わない方法はあるか？**

---

[状況]

同じ機能を持つ 2 つの Nextra/DB アクセス・モジュールがあるが、2 つの異なる DB にデータが置かれている。別の関数名の照会ファイルを重複して持つのではなく、クライアントプログラムから DB を切り替えるようにしたい。

### [対処方法]

バリアブル・ネームド・サーバを使用してください。詳細については『運用／設定ガイド』の「バリアブル・ネームド・サーバの概要」を参照してください。

## **[問題] サーバのログファイルが失われる。**

---

### [考えられる原因]

サーバが UNIX 環境で実行されている場合、ログファイルが削除されると、サーバ実行中に再度ログファイルを作成してもエラーは生成されずログ情報は失われる。UNIX では、削除したファイルに対して書き込みを続けられるように見えても、実際は書き込んだ情報は全て失われる。たとえ、すぐに新しいファイルを作成し直しても同じである。

### [対処方法]

サーバ実行中は、ログファイルを削除しないでください。

## ご注意

### 商標権に関する注意

Nextra 製品は、全て Inspire International Inc. の商標または登録商標です。その他記載のブランドおよび製品名は、該当する会社の商標または登録商標です。

### 著作権に関する注意

インスパイア インターナショナル株式会社の書面による許可なく、このマニュアルの内容の全部、もしくは一部を複写、複製、写真によるコピー、製本、翻訳、もしくは電子メディア化ないしは機械読み取りが可能な形態に変換することは固く禁じます

なお、本マニュアルの内容、連絡先などについては、弊社の都合により予告なく変更することがございます。あらかじめご了承ください。

特に記載がない限り、この製品に含まれるソフトウェアおよびドキュメントの著作権は Inspire International Inc. が所有しています。

## Nextra トラブルシューティングガイド

---

2011年 9月15日 v6 1<sup>st</sup> Edition  
2010年 9月9日 Reviewed  
2008年 10月15日 v5 2<sup>nd</sup> Edition  
2007年 4月17日 第3版発行  
2006年 11月17日 DCE\_UNAVAILABLE の追加  
2004年 8月24日 <[問題] クライアントがサーバを認識した後…> 追加  
2004年 7月19日 第2版発行  
2003年 4月18日 初版発行

著者 Inspire International Inc.

---

Copyright © 1998-2011 Inspire International Inc.  
Printed in Japan