

NEC CLUSTERPRO による Micro Focus Enterprise Server クラスター化の検証報告

平成 18 年 4 月 26 日

マイクロフォーカス株式会社

# 目次

はじめに・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・	• • •	1
使用環境・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・	•••	2
使用したハードウェア・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・	• • •	2
使用したソフトウェア・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・	• • •	2
概要 · · · · · · · · · · · · · · · · · · ·	• • •	3
Enterprise Server とテストシステム ・・・・・・・・・・・・・・・・・・・・・	• •	3
クラスタリング・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・	• • •	4
環境設定・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・	•••	5
手順・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・	• • •	5
環境設定モデル・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・	• • •	5
1. 事前準備・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・	• • •	5
2. 製品のインストール・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・	• • •	6
3. CLUSTERPRO の設定・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・	• • •	6
4. COBOL コンポーネントの準備・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・	• • •	11
5. Enterprise Server の設定・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・	• • •	11
6. クライアントの作成・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・	• • •	13
検証・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・		14
テストパターン・・・・・		14
パターン① 手動切り替え・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・		15
パターン② インタコネクト障害・・・・・・・・・・・・・・・・・		17
パターン③ 業務障害・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・		20
パターン④ OS のみで待機(シングルスタンバイ)・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・		24
パターン(5) MFDS 起動で待機(高速スタンバイ1)・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・		28
パターン⑥ FS サーバ起動で待機(高速スタンバイ 2)・・・・・・・・・		32
パターン⑦別の業務を提供しながら待機(相互スタンバイ1)・・・・		34
まとめ		39
(1) (1) (1) (1) (1) (1) (1) (1) (1) (1)		40
サーバアプリケーションソースサンプル・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・		40
クライアントアプリケーションソースサンプル・・・・・・・・・・・		41
		•••

# はじめに

本書では、Micro Focus Enterprise Server (以下、Enterprise Server)で提供するアプ リケーションサービスを NEC CLUSTERPRO SE (以下、CLUSTERPRO)により共有ディスクを 利用したクラスタ上で動作させた検証を報告します。

Enterprise Server は Windows、Unix、Linux 環境で COBOL サービスをトランザクショ ナルに管理・実行するアプリケーション サーバです。Enterprise Server は Micro Focus Net Express および Server Express COBOL コンパイラ ツール群の運用部分です。 Enterprise Server はメインフレームのトランザクション環境と同等の 0LTP 機能を、 J2EE アプリケーションサーバと協調実行できるよう設計されています。ユーザにとって は、ハイエンドのトランザクション システムで、機能およびパフォーマンスと同様に重 要なことは、その環境がきわめて可用性、信頼性、保守性に優れていることです。

CLUSTERPROはHAクラスタを構築するためのクラスタリングソフトウェアです。 障害発生時には、業務再起動や、冗長化されたサーバへの業務引継ぎ(フェイルオー バ)を行い、継続したサービスを提供します。

次ページ以降、CLUSTERPROとEnterprise Serverを組み合わせ、如何に信頼性・可用性の 高いCOBOLサービスシステムを提供していくか、その方法について検証します。

# 使用したハードウェア

(サーバ)	
Express580	0/120Lh 2 台
CPU	Intel(R) Xeon(TM) 3.00GHz×4
MEM	約4.45GB(4669152KB)
HD	36GB × 5
OS	Red Hat Enterprise Linux AS release 4
(ストレージ)	
ST12000	1 台
HD	8GB × 4
(クライアン	F)
Express580	0/58Ta 1 台
CPU	Celeron (mendcio) 465.594MHz
MEM	約192MB(191276KB)
HD	36GB × 5
OS	Red Hat Enterprise Linux AS release 4

# 使用したソフトウェア

(アプリケーションサーバ) Micro Focus Enterprise Server 4.0J Service Pack 2 for Intel x86 Linux

(クラスタリング) CLUSTERPRO SE for Linux Ver3.1-7

# (言語)

Micro Focus Server Express 4.0J Service Pack 2 for Intel x86 Linux Java(TM) 2 SDK, Standard Edition 1.4.2\_8

# Enterprise Server とテストシステム

本検証では Enterprise Server に実装するサンプルモデルを用意しました。



クライアント

サーバ(クラスタ)

Java クライアントから要求されたリクエストは Enterprise Server 上にディプロイされ たサービスの COBOL アプリケーションに引き渡されます。この COBOL アプリケーション は索引ファイルであるアプリケーションデータから連番を取得し、この連番をカウント アップして書き込みします。そして日時、プログラム名、実行ホスト名、連番を返しま す。Java クライアントはこれを表示して終了します。

以下に関連する Enterprise Server の構成要素について説明します。

(プロセス関連)	
• MFDS	Enterprise Server の基本サービスの一つ。1ノードに1つだけ 起動が必要
・ES サーバ	Enterprise Server の管理単位。1 つのノードに複数定義でき、 複数のサービスを含むことができる。
・サービス	Enterprise Server で提供する COBOL サービスであり、ディプロ イの単位。
・ COBOL アプリケーション	COBOL プログラムが提供する業務プロセス。
・MF 提供クラス	クライアントから Enterprise Server に接続するために利用
<ul> <li>Java クライアント</li> </ul>	コマンドライン起動で Enterprise Server にリクエストを出し 受け取った結果を表示する。
(ファイル関連)	
・ES システム	Enterprise Server のシステムコマンド、ライブラリなどのファ イルを含む。\$COBDIR 環境変数がポイントするディレクトリパ ス。共有可。
• ライセンスシステム	Enterprise Server のライセンス管理の製品のライセンスが投入 されるデータベースを保管する。このディレクトリをディスク コピーなどで移動・復元することはでない。共有不可。
・ログファイル	Enterprise Server のログ。ES サーバ単位で管理され、再起動 によって自動的に新しいファイルに書き換えられる。共有可。
・構成リポジトリ	Enterprise Server の設定情報を格納したファイル群。共有可。

- ・ディプロイパッケージ car ファイル。ディプロイに必要なファイル群。実装する際のディプロイパッケージは Interfase Mapping Tool Kit(IMTK)により COBOL ソースから容易に生成することが出来る。Server Express によってコンパイルされた実行形式のファイルを含む。(ソースサンプルは巻末に記載)共有可。
- ・アプリケーションデータ COBOL プログラムが利用するユーザファイル。共有可。
- ・Java クライアント Java プログラム。(ソースサンプルは巻末に記載)
- ・MF 提供クラス クライアントプログラムが利用する Enterprise Server 接続用の Java クラス。

## クラスタリング



CLUSTERPRO ではフェイルオーバする一連の操作をフェイルオーバグループの単位で管理 します。

フェイルオーバグループには、フローティングIPアドレス(仮想IPアドレス)や共有 ディスクパーティションの有効化、アプリケーションの起動・停止・監視の処理を設定 します。

フローティングIPアドレスや共有ディスクパーティションはフェイルオーバグループ が立ち上がると自動的に有効化され停止すると無効化します。

MFDSやESサーバなどの起動・停止はそれぞれのシェルにコマンドで処理を記述し ます。本運用ではこの起動・停止のシェルには障害時の状態回復・状態保存を行なう処 理なども組み込む必要があります。

またアプリケーションの監視は、障害とする事象が発生するまで監視を繰り返す業務監 視用のシェルをユーザが準備しこれを組み込む事で行ないます。 下図のモデルを想定した場合の環境の CLUSTERPRO、MF 製品の設定例を記述します。

## 手順

- 1. 事前準備
- 2. 製品インストールと動作環境の設定
- 3. CLUSTERPROの設定
- 4. COBOL コンポーネントの準備
- 5. Enterprise Server の設定
- 6. クライアントの作成

環境設定モデル



## 1. 事前準備

各ノードに搭載されたNICの設定、共有ファイルの接続に関する事前の設定、その他の OS レベルの設定を済ませておきます。

# 2. 製品のインストール

(クラスタ環境)

各ノード個別に下記2つの製品をインストールします。

- Micro Focus Enterprise Server 4.0J Service pack 2 for Intel x86 Linux
- CLUSTERPRO SE for Linux Ver3.1-7

#### (開発環境)

実装する COBOL コンポーネントの準備に COBOL・Java 開発環境が必要になります。こ のノード上もしくは別の同一 OS のマシン上に下記の製品をインストールします。 ・Micro Focus Server Express 4.0J Service pack 2 for Intel x86 Linux

- Mildro Focus Server Express 4.00 Service pack 2 for filler xoo L
- Java(TM) 2 SDK, Standard Edition 1.4.2\_8
- j2ee. jar

### (クライアント環境)

本検証ではクライアントは Java プログラムからリクエストしますので、クライアント に Java 環境が必要になります。本検証では

- Java(TM) 2 SDK, Standard Edition 1.4.2\_8
- j2ee. jar

本検証での Enterprise Server のインストールでは、下記の様に配置しました。 ES システム : /opt/mf/ES40SP2 ライセンスシステム : /var/mfasImf(固定) ログライル : /var/mfcobol/es 構成リポジトリ : /opt/mf/ES40SP2/etc/mfds/ /opt と/var は、いずれもローカルのディスクです。

またあらかじめ各製品毎の動作に必要な環境変数等の設定をしておきます。

## 3. CLUSTERPROの設定

本検証においては CLUSTERPRO の設定は製品に付属するトレッキングツールを使用しました。本ツールを使用するとリモート端末上の GUI 画面でクラスタ全体の構成ファイルを 定義・配布が可能となります。

ここで設定するのはサーバの構成、フェイルオーバグループの構成、モニタリングの構 成などです。

#### サーバの構成

構成するノード2つを登録します。ここでは server1 をマスタに設定しました。 ファイル(E) 編集(E) ヘルプ(E)

👇 🙋 cluster	名前	タイプ	コメント
🔶 🔚 Groups	server1	マスタ	
🗆 💻 failover1	server2		
— 🚍 Monitors			
P □ Servers			
— 🔳 server1			
🗕 🔳 server2			

ノード毎に各インタフェースの設定を行	ないます。
[ server1 ] サーバのプロパティ 🛛 🗙	[ server1 ] サーバのプロパティ 🛛 🗙
「情報 インタコネクトLAN IF」 パプリックLAN IF COM IF ディスク IF	情報 インタコネクトLAN IF パプリックLAN IF COM IF ディスク IF
「福祉」インタコネクトLANUF     バグリックLANUF     「COM UF     ディスク UF       利用する成績を登録してください。       インタコネクトLANUF 一覧(L)       UF 番号     IP 7 F L Z       1     192158 0.1       2     10.34.69.35   第時(B) 編集(E)	「情報」インタコネクトLANIF     バブリックLANIF     COM IF     ディスク IF       利用する候補を登録してください。     パブリックLANIF     10.3469.35     追加(D)       丁 作 香号     IP 7 F レス     IB 10.3469.35     IP 7 F レス       1     10.3469.35     IP 7 F レス     IB 10 10.3469.35
To(0)	T <sub>C</sub> (0)
Ok         キャンセル         適用(A)	Ok         キャンセル         適用(A)
[server1] サーバのプロパティ 🔀	J
「 诸報 」インタコネクトLAN IF パプリックLAN IF COM IF ディスク IF	
利用する()()浦を登録してください。 ディスク IF 一覧(L)	
UF 番号 デバイス Rawデバイス 1 //dew/sdf1 //dew/raw/raw1 調修(B) 編集(E)	
上へ切 下への の k キャンセル 途用(A)	

## フェイルオーバグループの構成

フェイルオーバグループを必要数定義します。本検証では、1 つのフェイルオーバグル ープに、フローティング IP、共有ディスク、フェイルオーバするリソースを設定し、テ ストパターンによっては監視プロセスなどを追加登録しました。

ファイル(日) 編集(日)	ヘルフ団
---------------	------

リソース一覧	全体の依存関係		
名前	タイプ	リソース情報	コメント
disk1	disk resource	/dew/sdf2	Sec. Sec.
exect	execute resource	start sh, stop.sh	
fip1	floating ip resource	10.34.69.38	
erver1 erver2		п	

フェイルオーバグループのリソースタイプ execute resource ではアプリケーションの起動・停止をシェルで管理できます。Enterprise Server のコマンドは下記の様に記述します。

処理	コマンド例
mfds の初期化処理	killall -9 mfds32
(必要に応じて)	
mfds の起動	mfds &
	Sleep 15 注)sleep コマンド値は環境に依存します
mfds の停止	killall -9 mfds32
ES サーバの初期化処理	para= <es サーバ名=""></es>
(必要に応じて)	ps -ef   grep \$para   grep -E "cassi cascd casjcp casmgr
	mfcs " > \$\$tmp.txt 2> /dev/null

	tmp=`cat \$\$tmp.txt   grep -v "grep -E "  awk '{print \$2}'`
	if [ ".\$tmp" != "." ] ; then
	kill -9 \$tmp
	fi
	rm -f \$\$tmp.txt
ES サーバの起動	casstart /r <esサーバ名></esサーバ名>
ES サーバの停止	casstop /r <es サーバ名=""></es>

フェイルオーバグループの起動シェルサンプル(start.sh)

#! /bin,	/sh	ta bala da ba
#****** #*	********** S	**************************************
#*****	• ********	*****
if [ "\$0	CLP_EVENT"	= "START" ]
then	Γ.Γ.Γ. ″.Φ.ΟΙ	
	IT L \$C	LP_DISK = SUCCESS 」
	Luen	
		source /root/ES40SP2
		mfds &
		sleep 15
		casstart /rESSERV1
		echo "NORMAL 1"
		if [ "\$CLP_SERVER" = "HOME" ]
		then
		echo "NORMAL2"
		else
		echo "UN_UTHER1"
	وادو	
	0100	echo "ERROR DISK from START"
	fi	-
elif ['	"\$CLP_EVEN	T" = "FAILOVER" ]
then	:ድር ፖለል	
	IT [ \$U then	LP_DISK = SUCCESS 」
	LIIGII	
		source /root/ES40SP2
		mfds &
		sleep 15 <i>追記部分</i>
		casstart /rESSERV1
		echo "FALLOVER1"
		if [ "\$CLP SERVER" = "HOME" ]
		then
		echo "FAILOVER2"
		else
		echo UN_UIHEKZ
	وادو	11
	6196	echo "ERROR DISK from FAILOVER"
	fi	
else		
	echo "NO	_CLP″
fi , "		
echo "E)	XII	
EXIL U		

```
フェイルオーバグループの停止シェルサンプル
#! /bin/sh
#*****
#*
               stop. sh
                                  *
#*****
if [ "$CLP_EVENT" = "START" ]
then
        if [ "$CLP_DISK" = "SUCCESS" ]
                                              環境変数の設定
        then
                source /root/ES40SP2
                casstop /rESSERV1
                                                追記部分
                killall -9 mfds32
                echo "NORMAL1"
                if [ "$CLP_SERVER" = "HOME" ]
                then
                        echo "NORMAL2"
                else
                       echo "ON_OTHER1"
                fi
        else
                echo "ERROR_DISK from START"
        fi
elif [ "$CLP_EVENT" = "FAILOVER" ]
then
        if [ "$CLP_DISK" = "SUCCESS" ]
                                               環境変数の設定
        then
                source /root/ES40SP2 
                casstop /rESSERV1
                                                追記部分
                killall -9 mfds32
                echo "FAILOVER1"
                if [ "$CLP_SERVER" = "HOME" ]
                then
                       echo "FAILOVER2"
                else
                        echo "ON_OTHER2"
                fi
        else
                echo "ERROR_DISK from FAILOVER"
        fi
else
        echo "NO_CLP"
fi
echo "EXIT"
exit O
```

これらのシェルは execute resource のプロパティ画面からエディタを起動して編集できます。

またフェイルオーバグループ毎に起動するサーバを優先順位をつけて設定します。

情報 起	副サーバ 属性	
起動可能な	なサーバ( <u>S</u> )	利用可能なサーバ(
順位	名前	< i6加(D) 名前
1	server1	
2	server2	<b>削除(R) &gt;</b>
		- Logo
		<u>لامم</u>
		上へ(U) 下へ(D)

### ハートビートの構成

ハートビートの構成では検知する箇所と優先順位を登録します。 例えばインタコネクト LAN で検知を行なうには、まず各サーバの設定で IP アドレスを指 定してインタコネクト LAN を登録して I/F 番号を取得します。

[ server1 ] サーバのプロパティ	×	[server]	2]サーバのブロパティ			×
「请報 インタコネクトLAN IF パプリックLAN IF COL	MIF ディスクIF	情報	インタコネクトLANIF	パブリックLAN IÆ	COMIÆ	ディスクル
利用する候補を登録してください。			利用する候補を	登録してください。		
インタコネクトLAN IF 一覧(L)		インタ	! コネクトLAN IF 一覧(L)			
レF番号 レアドレス 1 10216801	追加(D)	I/F	番号 IPアト 19216802	ドレス		追加(D)
2 10.34.69.35	削除( <u>R</u> )	2	10.34.69.36			削除( <u>R</u> )
	編集(E)					編集(E)
	上へ(山)					上へ(山)
	下へ(0)					下へ(0)
Ok ‡+:	ンセル 適用(A)			Ok	キャンセル	, 適用( <u>A</u> )

次にクラスタの設定で、この I/F 番号と LAN タイプでハートビートの登録を行ないます。

E	cluster ] クラスタのプロパラ	Ē-{			
7	監視 メール通報 1	Webマネ		アラートログ	遅延警告
ł	情報 ハートビート	VF 7	・- ・ マスタサー	バ タイムア	ウト ポート番号
f				1	1
	ハートビートIFの優先II	順位一覧(	P)		利用可能なIF(V)
	順位	タイプ	I/F 番号	< 追加(D)	タイブ IF番号
	1	LAN	1		KLAN 1
	3	DISK	2	削除(B) >	KLAN Z
	-	DIDIK			
				Lo (I)	
				T.(i)	
				下へ(0)	
	I				
	📃 サーバダウン通知(S	0			
					No. 1
				Ok ++	ンセル 通用(A)

### モニタリングの構成

モニタリングの構成では障害を検知するモニタの登録を行ないます。

ファイル(E) 編集(E) ヘルブ(E)

	1	89				
9	ê	cluster	名前	タイプ	監視先	コメント
	9	Groups	diskw1	disk monitor	/dev/sdf2	
	1	- A failover1	miw1	NIC Link Up/Down	eth0	
	13	Manitors	userw	user mode monitor	softdog	user mode monitor
	9	Servers		- VI		

この検証では、ディスクモニタ、NIC Link Up/Down モニタ、ユーザ空間モニタ(デフォ ルト)を設定しました。テストパターンによっては pid モニタを追加登録しました。

### 4. COBOL コンポーネントの準備

Enterprise Server に実装するコンポーネントの準備を行ないます。 作業は開発環境にて行ないます。本検証では Server Express が提供する Interface Mapping Tool Kit を使って一連の作業を簡略化しています。 初めに COBOL ソースの元にサービスに対するデフォルトのマッピングを作成します。サ ービス名は任意に命名します。

imtkmake -defmap src=cblapp01.cbl service=srv01 type=ejb これで"srv01.xml"の名称でデフォルトのマッピング情報が生成されます。 デフォルトではこのサービスのパラメータはプログラム中のパラメータ名で全 て"io"(入出力)として生成されています。本サンプルでは出力パラメータのみである ため実態に合わせ全て"out"に手修正しました。

更にここで同じ COBOL ソースを cob コマンドでコンパイルし、実行形式ファイル(本検証 では gnt 形式を選択)を生成しておきます。

そして次のコマンドでディプロイパッケージを生成します。 imtkmake -generate service=srv01 type=ejb cblapp01.gnt これで"srv01.car"ファイルが生成されます。

## 5. Enterprise Server の設定

まず/etc/services に次の2行を追加します。 mfcobol 86/tcp mfcobol 86/udp 次にmfds コマンドで Enterprise Server を起動します。 Web ブラウザから http://server1:86 でアクセスすると Enterprise Server Administration 画面が開きます。ここで多くの操作を実行できる様になります。

初めにデフォルトで生成されている ES サーバ"ESDEMO"を削除します。 次に新規に必要な ES サーバを作成します。名称以外はデフォルトの設定としました。

次にディプロイパッケージを使ってサービスの登録を行ないます。 <Enterprise Server インストールディレクトリ>/deploy の下に任意のディプロイ用ディレクトリを 作成し、その下に4で作成した car ファイルと実行形式ファイルをコピーします。 <Enterprise Server インストールディレクトリ>/deploy/.mfdeploy を以下の様に書き換えます。 MFDS=localhost MFUS\_SERVER=<u>ESSERV1</u> MFUS\_LISTENER=Web Services <Enterprise Server インストールディレクトリ>/deploy の下にのディプロイ用ディレクトリに移動 して下記のコマンドを実行します。

mfdepinst srv01.car

すると.mfdeploy に記載した ES サーバ上に car ファイル作成時に指定した名称でサービ スが作成されます。登録された ES サーバは Start (開始)ボタンで手動で開始できます。

8	1 - 1 of 1 Servers	Show 10 servers at a tir	ne << Previous Next >>
	Type Status Processes	Status Licenses Log	Objects Description
Edit	MFEE ESSERVI 3)arted Details 1 top:10.34.69.35*3: (server1 +) 4 2 Listeners Det	3224 <sup>#</sup> * / 10 MD \$3800 E Server started successfully 15:21:28	3 Services Details 2 Handlers Details 5 in 2 Handlers Details
Add	Stop	"Started " state since 15:2	28 1 Packages Details

Server ESSERV1 [Started]									
Server Listeners (2) Services (3) Handlers (2) Packages (1) Refresh									
1 - 1 of 1 packages Show 10 packages at a time << Previous Next >>>									
Name	Current Status	Status Log	Package Module	ют	Package Path	Custom Configuration	Description		
Edit (srv01.CBLAPP01	Available	ок		/opt/mf/ES40SP2/deploy/svr01/srv01.idt	/opt/mf/ES40SP2/deploy/svr01		oreated 15??⊡ \$ 2005-06-25 from svr01/srv01.car		
Add									

4	Server	ESSERV	1 [ Starte	d ]									
	Server ] Listeners (2) <mark>Services (3)</mark> Handlers (2) Packages (1)												
	Service Display Filters Namespace: Operation: Class: All 💌 Handler: All 💌									Y Pac			
	1 - 3 of 3 displayable namespaces from a total of 3 Show 10 service namespaces at a time										<< Pre		
	Service Search Req Namespace Operation Class Order Listener(s) Har						Request Handler	Implementation Package	Current Status	Status Log	Custom Configuration		
		Deployer	Edit	MF deployment	1	1 CP 1 Web top:127.0.0.1*:33548* (server2 +)			Available	ОК	[MF client] s URL=/ogi/m1 accept=appli listener=Web	cheme=http (deploy.exe/uploads cation/x-zip-compres ) Services	sed [destination
		ES	ES Edit	MF ES	1	1 CP 1 Web Services top:10.34.69.38:9003 O			Available	ОК			
Delete (srv01) 1 of 1 operations shown													
			.CBLAPP01 Edit		1	1 CP 1 Web Services top:10.34.69.38:9003 0	MFRHBINP	srv01.CBLAPP01	Available	ок			
	Add												

### 次にリスナーの設定をします。

ES サーバを停止後、リスナーの画面の Edit(編集)ボタンで編集画面を開 き、"WebServices"の中に"\*:\*"で設定されている IP アドレス:ポート番号の IP アド レスを固定で設定します。

ここで設定する IP アドレスはフェイルオーバグループでフェイルオーバする場合にはフ ローティング IP (10.34.69.38) を、グループ化せずに単体でテストする場合には固定 IP (10.34.69.35) を設定すると便利です。

Server E	Server ESSERV1 [Started]											
Server	Server Listeners (2) Services (3) Handlers (2) Packages (1) Refresh Auto-refresh interval (seconds) 30											
Listener Display Filters Process: All V Conversation Type: All V Status: All V												
武 🛄 Con	Communications Process 1 V Auto-start Statistics Stop Copy Delete											
L	isteners	Process	ID Control Channel Ad	ldress	St	tatus	Last Status Change Status Log			Version		
Edit	2 Add	proces	s top:127.0.0.1*:3354 (server2 +)	16×		Started	02/14/06-15:43:54 Received admin request 6 respon			nin request 6 response OK		1.6.1
		Name	Address	Status	Last Stat	tus Change	Status Log	Conve	rsation Type	Custom Configuration	Descri	ption
	Edit	Web Services	top:10.34.69.38:9003 9	Started	02/14/06	-15:43:54	ок	Web S J2EE	ervices and		Web S HTTP/	ervices via SOAP
	Edit Web top:127.0.0.1*:33548* Started 02/14/		02/14/06	-15:43:54	ок	Web		[virtual paths] cgi= <es>/bin uploads=<es>/deploy docs=<es>/docs/ntml</es></es></es>	Basic server	HTTP web		

# 6. クライアントの作成

本検証で Enterprise Server 上にディプロイしたサービスにリクエストを出すのは Java のクライアントプログラムです。(ソースサンプルは巻末に記載)

このプログラムを利用するには\$CLASSPATHに下記の3ファイルを指定する必要があります。

j2ee.jar

mfconnector.jar (<Enterprise Server インストールディレクトリ>/libに存在) mfcobolpure.jar (<Enterprise Server インストールディレクトリ>/libに存在) 検証

# テストパターン

本検証では以下の7パターンの確認をしました。

パターン	タイトル	説明
1	手動切り替え	CLUSTERPRO への操作による切り替えを確認
2	インタコネクト障害	インタコネクト LAN のケーブルを引き抜いて切り替えを確認
3	業務障害	業務監視用シェルの異常検出による切り替えを確認
4	OS のみで待機	待機状態のノードでは MFDS 起動せず切り替えを確認
	(シングルスタンバイ)	
5	MFDS 起動で待機	待機状態のノードでは MFDS 起動して切り替えを確認
	(高速スタンバイ 1)	
6	DS サーバ起動で待機	待機状態のノードでは MFDS、DS サーバを起動状態で切
	(高速スタンバイ 2)	り替えを確認
$\overline{\mathcal{O}}$	別の業務を提供しなが	待機状態のノードでは MFDS 起動、切り替える DS サー
	ら待機	バとは別の DS サーバを動作させせながら切り替えを確
	(相互スタンバイ)	認。ノード1⇔ノード2の双方向で確認

①~③はフェイルオーバのトリガーのテストです。代表的なトリガー3種類のパターンで確認します。

フェイルオーバをステータスで確認します。

④~⑦は待機側のノードの状態を変えるテストです。本検証では Enterprise Server の プロセスの実行状態を変えて確認します。

Java クライアントから繰り返しリクエストを出し続け、フェイルオーバ後もサービスが 提供され続ける事を確認します。

以降、パターン毎の設定と検証結果を記述します。

# パターン① 手動切り替え

<目的>

通常の運用でも行なわれる動作の確認。

CLUSTERPRO のコマンドを実行し、フェイルオーバグループの移動が行なわれるか検証する。

<構成>



<設定のポイント>

failover1の開始シェルに組み込む起動コマンド

mfds &	
sleep 15	
casstart /rESSERV1	

failover1の終了シェルに組み込む停止コマンド

casstop /rESSERV1 killall -9 mfds32 <検証> CLUSTERPRO コマンドにてフェイルオーバを行なう。

CLUSTERPRO が立ち上がると server1 でフェイルオーバグループ failover1 が開始する。

ES サーバが起動されている事を確認。

MICRO Focus	Enterprise Server Administration     Version 1.04.00     server1 (10.34.69.35)
Home	Status MDS0000I OK
Actions Save Restore	Refresh Auto-refresh interval (seconds) 10
Delete All	1 - 1 of 1 Servers         Show 10
Shut Down	Type Name Current Communications Processes Licenses
Configure Options	Edit MFES ESSERV Started 1 top:10.34.69.36*.33224* */10
Users Add	2 Listeners Details
Update	Add
Display	

次に CLUSTERPRO コマンドにてフェイルオーバグループを移動する > clpgrp -m failover1

すると、フェイルオーバグループ failover1の停止処理により mfds が終了するため server1の管理画面が表示できなくなる。

<b>i</b> ~	ージを表示できません
検索中の~	ページは現在、利用できません。Web サイトに技術的な問題が発
生している	か、ブラウザの設定を調整する必要があります。
次のことを	試してください
• 🖗	匣新] ボタンをクリックするか、後でやり直してください。
• 7	ドレス バーにページ アドレスを入力した場合は、ページ アドレスを
正	しく入力したかどうかを確認してください。
• 接	続の設定を確認するには、「シール]メニューの「インターネットオ
ン:	ンヨンフ をクリックしょう。 (Fak) タフ ビ (タイマルアッフの)数定) フ
ル	ーブの [設定] ボタン、または [LAN の設定] グルーブの [LAN の
設	定] ボタンをクリックしてください。 設定情報頃 、LAN ローカル エ
01	アネット ワーク) の管理者か、ISP (インターネット サービス ブロバ
12	ダ) が提供する情報と一致する必要があります。
• ネ	ットワーク管理者がネットワークの接続の設定を使用可能にしてい
れ	ば、Microsoft Windows を使用して、ネットワークの接続試験を行
っ)	とり、自動的にネットワークの接続の設定を見つけることができま。
す.	。

そして server2 でフェイルオーバグループ failover1 が開始される。

MICRO Focus	Versi Serv	terprise on 1.04.00 rer2 (10.34.69	<b>Serve</b> 9.36)	r Administrat	ion
Home	Status MDS	0000I OK			
Actions Save Restore	Refresh	Auto-refres	h interval	(seconds) 10	
Delete All Shut Down	8	1 - 1 of 1 Sen	/ers		Sh
Graden		Type Name	Current Status	Communications Processes	Licenses
Options	Edit	MFES ESSERV	Started Decans	1 top:10.34.69.36*:33224* (server2) ✔	· / 10
Users Add		3	Stop	2 Listeners Details	
Update Display	Add				

## <結果>

CLUSTERPRO のコマンド clpgrp によるフェイルオーバグループの移動を確認。

# パターン② インタコネクト障害

### <目的>

CLUSTERPROの機能のみで検知する障害パターンの確認 インタコネクトLANの障害を起こし、フェイルオーバグループの移動が行なわれるか検 証する。

## <構成>



<設定のポイント>

failover1の開始シェルに組み込む起動コマンド

mfds &	
sleep 15	
casstart /rESSERV1	

failover1の終了シェルに組み込む停止コマンド

casstop /rESSERV1 killall -9 mfds32

クラスタの設定にインタコネクト LAN の登録をする。





インタコネクト LAN のケーブルを引き抜く。

すると、フェイルオーバグループ failover1 の停止処理により mfds が止まるため server1 の管理画面が表示できなくなる。

<b>i</b> ベージを表示できません
検索中のページは現在、利用できません。Web サイトに技術的な問題が発 生しているか、 ブラウザの設定を調整する必要があります。
<ul> <li>次のことを訪してください。</li> <li>(アレス) 「モダンをクリックするか、後でやり直してください。</li> <li>アドレス バーにページ アドレスを入力した場合は、ページ アドレスを 正く入力したかどうかを確認してください。</li> <li>接続の設定を確認するには、 (ソール) メニューの ビインターネット オ ブション) をクリックします。 (接続) タブで (ダイヤルアップの設定) グ ルーブの (設定) ボタン、または、(LAN の設定) グルーブの (LAN の 設定) ボタンをクリックしてください。 設定情報は、LAN (ローカル エ リアネット ワーク) の管理者か、ISP (インターネット サービス ブロバ イダ) が提供する情報とつ取する必要があります。</li> <li>キットワークで毎年がネットワークの接続の設定を使用するにしてい れば、Microsoft Windows を使用して、ネットワークの接続は決を行 ったり、自動的にネットワークの接続の設定を見つけることができま す。</li> </ul>

そして server2 でフェイルオーバグループ failover1 が開始される。

MICRO FOCUS Server Administration Server2 (10.34.69.36)					
Home	Status MDS00001 OK				
Actions Save Restore	Refresh Auto-refresh interval (seconds) 10				
Delete All	1 - 1 of 1 Servers Sh				
Shar Down	Type Name Current Communications Licenses Licenses				
Options	Edit MFES ESSERVI Started 1 top:10.34.69.36*33224* -/10				
Users Add	2 Listeners Details				
Update	Add				
Display					

この時の CLUST	TERPR0 の状態	態(clpstat コマンドの結果)
Every 2.0s: clpstat		
	==== CLUSTER STAT	US =======
Cluster : cluster		
<server></server>		
server1	: Caution	
lanhb1	: Normal	LAN Heartbeat
l anhb2	: Caution	LAN Heartbeat
diskhb1	: Normal	DISK Heartbeat
*server2	: Caution	
lanhb1	: Normal	LAN Heartbeat
l anhb2	: Caution	LAN Heartbeat
diskhb1	: Normal	DISK Heartbeat
<gr oup=""></gr>		
failover1	: Online	
current	: server2	
disk1	: Online	
exec1	: Online	
fip1	: Online	
<monitor></monitor>		
diskw1	: Normal	
miiw1	: Caution	
userw	: Normal	user mode monitor

<結果>

インタコネクトの障害によるフェイルオーバグループの移動を確認。

# パターン③ 業務障害

### <目的>

ユーザ作成の監視シェルにより検知する障害パターンの確認 本検証では Enterprise Server のプロセス異常を想定し、主要サービスである MFDS を停 止しすることでフェイルオーバグループの移動が行なわれるか検証する。

### <構成>



<設定のポイント>

failover1の開始シェルに組み込む起動コマンド

mfds & sleep 15 casstart /rESSERV1

failover1の終了シェルに組み込む停止コマンド

```
casstop /rESSERV1
killall -9 mfds32
```

ユーザ作成シェルで状態監視を行なうには、監視用のシェルを新たな execute resource として登録・起動し、このシェルの実行状態を CLUSTERPRO で監視して障害検知を行なう。

ユーザ作成シェルは自動的に監視対象をチェックし続け、異常を検知した時に終了する ように作成する。

# failover1に監視用の execute resource、execo1 を追加

ノアイル(ロー油来(ロ) ヘルノ(ロ)						
a 8 9						
ዮ 🥡 cluster	リソース一覧 全	体の依存関係				
P Groups	名前	タイプ	リソース情報	コメント		
a falloveri	disk1	disk resource	/dev/sdf2			
- Monitors	exec1	execute resource	start.sh, stop.sh			
🕈 🧰 Servers	execo1	execute resource	start.sh, stop.sh			
- 🛛 server1	fip1	floating ip resource	10.34.69.38			
🗕 🗌 server2						
1000						
			III			

execolの起動シェル start. sh に監視処理を組み込む

```
sleep 10
mdump_func()
{
mdump localhost ESSERV1 > $$tmp.txt 2> /dev/null
web_srv_flg=0
cat $$tmp.txt | while read line
do
         tmp=`echo $line | fgrep "SERVER_DOWN"`
         if [ ".$tmp" != "." ] ; then
                 return 3
         fi
         tmp=`echo $line | fgrep "CN:"`
         if [ ".$tmp" != "." ] ; then
                 tmp=`echo $line | fgrep "Web Services"`
if [ ".$tmp" != "." ] ; then
                          web_srv_flg=1
                 else
                          web_srv_flg=0
                 fi
        fi
         if [ $web_srv_flg -eq 1 ] ; then
                 lisnr=`echo $line | fgrep "mfListenerStatus:"`
if [ ".$lisnr" != "." ] ; then
                          lisnr=`echo $line | fgrep "Started"`
                          if [ ".$lisnr" != "." ] ; then
                                   return O
                          else
                                   return 1
                          fi
                 fi
        fi
done
}
if [ ".$host_name" = "." ] || [ ".$es_name" = "." ] ; then
     host_name="localhost"
     es_name=""
fi
while : ; do
         # echo "monitoring"
        mdump_func
        res=$?
         rm $$tmp.txt
         if [ "$res" = "1" ]; then
                 exit O
         fi
         if [ "$res" = "3" ]; then
                 exit O
        fi
        sleep 5
done
;;
```

ここに記載した監視処理はマイクロフォーカスが提供する mdump ユーティリティの出力 を利用して異常を検出する。(MFDS が正常に動作しておらず) mdump がステータス情報を 正常に取得できない場合、及び、Web Service リスナーのステータスが" Started"以 外にて異常とする検知する方式とした。

さ	らに execol	を監視対象とす	るpidモニタ	を追加登録する。
---	-----------	---------	---------	----------

[pidw1 ]監視リソースのプロパティ	<u>×</u>
请報 監視 異常検出	
インターバル(1)	5 秒
タイムアウト(1)	30 秒
リトライ回数(13)	0
監視開始待ち時間(S)	0 秒
監視タイミング ○ 常時(L) ● 活性時(C)	
対象リソース(G) execo1	参照(W)
nice(@(E)	
監視を行うサーバを選択する	サーバ( <u>R</u> )
	Ok キャンセル 適用(A)

く検証>

server1 でフェイルオーバグループ failover1 を起動している初期状態にする。

MICRO Focus	Enterprise Server Administration     Version 1.04.00     server1 (10.34.69.35)
Home	Status MDS0000I OK
Actions Save Restore Import	Refresh Auto-refresh interval (seconds) 10
Delete All Shut Down	L I-Tori Servers Show 10
Configure	Type Name Status Processes Licenses
Options	Edit MFES ESSERV Started 1 top:10.34.69.35*.33224* (server1 +) *
Users	2 Listeners Details
Update	hth
Display	

次にMFDSのプロセスをkill する。 > killall -9 mfds32

すると、mfdsのプロセス異常により execol の監視シェルが障害を検知しファイルオーバが開始する。

フェイルオーバグループ failover1 の停止処理により mfds が止まるため server1 の管理 画面が表示できなくなる。

<u>そして server2 でフェイルオーバグループ failover1 が開始される。</u>

MICRO Focus	Enterprise Server Administration     Version 1.04.00     server2 (10.34.69.36)					
Home	Status MDS00001 OK					
Actions Save Restore Import	Refresh Auto-refresh interval (seconds) 10					
Delete All Shut Down	1 - 1 of 1 Servers					
Configure	Type Name Current Communications Processes Licenses					
Users Add	(gerver2) V 2 Listeners Details					
Update Display	Add					

<結果>

ユーザ作成の監視シェルにより検知した障害によるフェイルオーバグループの移動を確認。

<目的>

待機側のノードで Enterprise Server の関連プロセスは起動しない状態からのフェイル オーバを確認する。

フェイルオーバ前と後で同じサービスが継続される事も確認する。

<構成>



<設定のポイント>

(パターン1)~3の構成と同じ)

failover1の開始シェルに組み込む起動コマンド

```
mfds &
sleep 15
casstart /rESSERV1
```

failover1の終了シェルに組み込む停止コマンド

casstop /rESSERV1 killall -9 mfds32

クラスタの設定にインタコネクト LAN の登録をする。

NEC CLUSTERPRO による Micro Focus Enterprise Server クラスター化の検証報告

```
<検証>
```

server1でフェイルオーバグループ failover1を起動している初期状態にする。

MICRO Focus	Enterprise Server Administration     Version 1.04.00     server1 (10.34.69.35)					
Home	Status MDS0000I OK					
Actions Save Restore Import Delete All	Refresh Auto-refresh interval (seconds)					
Shut Down	Tyrne Name Statue Processer Licenses					
Configure Options	Edit MFES ESSERV Started 1 top:10.34.60.35*:33224 <sup>4</sup> (server1 +) √ 10					
Users Add	2 Listeners Details					
Opdate Display	Add					

クライアントマシンで Java クライアントプログラムを定周期で実行を繰り返す。 シェル例 (Run java Test. sh)



インターコネクト LAN のケーブルを引き抜く。

すると、フェイルオーバグループ failover1の停止処理により mfds が止まるため server1の管理画面が表示できなくなる。

<b>i</b> ページる 検索中のページス	を表示できません 現在、利用できません。Web サイトに技術的な問題が発	
主じているがく ク.	ノブリ いあれたとき時か 9 〜2013年(1 1007)ま 9 。 	
<ul> <li>・ (注) 便新)</li> <li>・ アドレス)</li> <li>・ アドレス)</li> <li>・ アドレス)</li> <li>・ アドレス)</li> <li>・ 投稿の設 プション)</li> <li>ルーブの 設定する。</li> <li>・ クリー・ のまし、単 ったり、自 す。</li> </ul>	If 考少をクリックするか、後でやり直してください。     If インページ アドレスを入力した場合は、ページ アドレスを     Tuchッジ カアドレスを     Tuchッジ かアドレスを     Tuchッジ かない     Tuch・     Tuch・	プfailover1が開始される。
FOCUS	Version 1.04.00 Server2 (10.34.69.36)	
Home	Status MDS00001 OK	
Actions Save Restore Import	Refresh Auto-refresh interval (seconds) 10	
Delete All Shut Down	1 - 1 of 1 Servers	Sh
Configure Options	Edit	icenses - / 10
<b>Users</b> Add Update	Stop 2 Listeners Details	
Display	Aub	

クライアントプログラム実行結果 [root@gw test]# ./RunjavaTest.sh 06/03/14 15:42:18.07 CBLAPP01 server1 128 06/03/14 15:42:20.54 CBLAPP01 server1 129 06/03/14 15:42:22.87 CBLAPP01 server1 130 06/03/14 15:42:25.21 CBLAPP01 server1 131 06/03/14 15:42:27.53 CBLAPP01 server1 132 06/03/14 15:42:29.82 CBLAPP01 server1 133 server1 でサービスが稼動 06/03/14 15:42:32.12 CBLAPP01 server1 134 06/03/14 15:42:34.43 CBLAPP01 server1 135 06/03/14 15:42:36.72 CBLAPP01 server1 136 06/03/14 15:42:39.02 CBLAPP01 server1 137 06/03/14 15:42:41.35 CBLAPP01 server1 138 06/03/14 15:42:43.64 CBLAPP01 server1 139 iava.net.ConnectException: Connection refused at java.net.PlainSocketImpl.socketConnect(Native Method) at java.net.PlainSocketImpl.doConnect(PlainSocketImpl.java:305) at java.net.PlainSocketImpl.connectToAddress(PlainSocketImpl.java:171) サービス中断中 at java.net.PlainSocketImpl.connect(PlainSocketImpl.java:158) at java.net.Socket.connect(Socket.java:452) at java. net. Socket. connect (Socket. java: 402) at java.net.Socket.<init>(Socket.java:309) at java.net.Socket.<init>(Socket.java:153) at com. microfocus. cobol. connector. transport. BINPStream. createSocketWithTimeout(BINPStream. java: 894) at com.microfocus.cobol.connector.transport.BINPStream.createSocketWithTimeout(BINPStream.java:825) at com.microfocus.cobol.connector.transport.BINPStream.</br/>
(BINPStream.java:97) at com. microfocus. cobol. connector. transport. BINPRemoteCall. open (BINPRemoteCall. java: 176) at com.microfocus.cobol.connector.transport.BINPRemoteCall.connect(BINPRemoteCall.java:73) at com. microfocus. cobol. connector. transport. BINPRemoteCall. reconnect(BINPRemoteCall. java:145) at com.microfocus.cobol.connector.PureCobolBean.cobcall(PureCobolBean.java:285) at com. microfocus. cobol. connector. cci. CobolInteraction. exec (CobolInteraction. java:247) at com. microfocus. cobol. connector. cci. CobolInteraction. execute (CobolInteraction. java: 174) at clptest.main(clptest.java:47) ( 繰り返し・中略 ) iavax, resource, spi, EISSystemException: CobolException Exception throw during open (see getCause() for more information) (elapsed time=45103.0) for 10.34.69.38:9003 (cause class: java.net.ConnectException, cause:Connection refused) executing srv01.CBLAPP01 iava.net.ConnectException: Connection refused at java.net.PlainSocketImpl.socketConnect(Native Method) at java.net.PlainSocketImpl.doConnect(PlainSocketImpl.java:305) at java.net.PlainSocketImpl.connectToAddress(PlainSocketImpl.java:171) at java.net.PlainSocketImpl.connect(PlainSocketImpl.java:158) at java.net.Socket.connect(Socket.java:452) at java. net. Socket. connect (Socket. java: 402) at java.net.Socket.<init>(Socket.java:309) at java.net.Socket.<init>(Socket.java:153) at com. microfocus. cobol. connector. transport. BINPStream. createSocketWithTimeout(BINPStream. java:894) at com. microfocus. cobol. connector. transport. BINPStream. createSocketWithTimeout(BINPStream. java:825) at com.microfocus.cobol.connector.transport.BINPStream.<br/>
(init)(BINPStream.java:97) at com. microfocus. cobol. connector. transport. BINPRemoteCall. open (BINPRemoteCall. java: 176) at com. microfocus. cobol. connector. transport. BINPRemoteCall. connect(BINPRemoteCall. java:73) at com.microfocus.cobol.connector.transport.BINPRemoteCall.reconnect(BINPRemoteCall.java:145) at com.microfocus.cobol.connector.PureCobolBean.cobcall(PureCobolBean.java:285) at com. microfocus. cobol. connector. cci. CobolInteraction. exec (CobolInteraction. java:247) at com. microfocus. cobol. connector. cci. CobolInteraction. execute (CobolInteraction. java: 174) at clptest.main(clptest.java:47) javax.resource.spi.ElSSystemException: CobolException Exception throw during open (see getCause() for more information) (elapsed time=268.0) for 10.34.69.38:9003 (cause class: java.net.ConnectException, cause:Connection refused) executing srv01.CBLAPP01 06/03/14 15:43:55.76 CBLAPP01 server2 140 06/03/14 15:43:58.07 CBLAPP01 server2 141 06/03/14 15:44:00.37 CBLAPP01 server2 142 06/03/14 15:44:02 67 CBI APP01 server2 143 06/03/14 15:44:04.98 CBLAPP01 server2 144 06/03/14 15:44:07.28 CBLAPP01 server2 145 server2でサービスを引継ぎ。データファイルの連番も継続 06/03/14 15:44:09.59 CBLAPP01 server2 146 06/03/14 15:44:11.91 CBLAPP01 server2 147 06/03/14 15:44:14.19 CBLAPP01 server2 148 06/03/14 15:44:16.48 CBLAPP01 server2 149 06/03/14 15:44:37.26 CBLAPP01 server2 157 [root@gw test]#

## <結果>

待機側のノードで Enterprise Server の関連プロセスは起動しない状態(シングルスタン バイ)からのフェイルオーバグループの移動を確認。

また、一定時間のサービス中断後、同じサービスを待機側で提供できていることも確認。

<目的>

待機側のノードで Enterprise Server の基本プロセス MFDS のみ起動した状態からのフェ イルオーバを確認する。

フェイルオーバ前と後で同じサービスが継続される事も確認する。

<構成>



<設定のポイント>

それぞれ各ノードであらかじめ MFDS を立ち上げておく。 MFDS の制御はフェイルオーバグループに含めない。

failover1の開始シェルに組み込む起動コマンド

casstart /rESSERV1

failover1の終了シェルに組み込む停止コマンド

casstop /rESSERV1

クラスタの設定にインタコネクト LAN の登録をする。

NEC CLUSTERPRO による Micro Focus Enterprise Server クラスター化の検証報告

# く検証>

<u>server1 でフェイルオーバグループ failover</u>1 を起動している初期状態にする。

MICRO Focus	Enterprise Server Administration     Warsion 1.04.00     server1 (10.34.69.35)					
Home	Status MD	S00001	OK			
Actions Save Restore Import	Refresh	Aut	o-refresh	interval	(seconds) 10	
Shut Down		1-10	JI I Gerve	ers	1	Show 10
Configure Options	Edit	Type MFES	Name ESSERV1	Status Status Started	Communications Processes 1 top:10.34.69.35*:332 (server1 +)	Licenses
Users Add				Stop	2 Listeners Detai	IS
Opdate Display	Add					

server2 ではフェ・	イルオーバグルーフ	গfailover1(ES サー	バ ESSERV1)は停止している。

MICRO Focus	Enterprise Server Administration     Version 1.04.00     server2 (10.34.89.36)	ı
Home	Status MDS0000I OK	
Actions Save Restore Import	Refresh Auto-refresh interval (seconds) 10	
Delete All Shut Down	1 - 1 of 1 Servers	Sh
Configure	Type Name Current Communications Processes Licen	nses
Options	Edit MFES ESSERVI (Stopped) 1 top:10.34.89.36*.36736* -/	10
Users Add	2 Listeners Details	
Display	Add	

クライアントマシンで Java クライアントプログラムを定周期で実行を繰り返す。 シェル例(RunjavaTest.sh)

#!/bin/sh
while [ "1" == "1" ]; do
java -
${\tt classpath} :: /{\tt home/clinux2/test/mfcobol.jar:/{\tt home/clinux2/test/mfconnector.jar:/{\tt home/clinux2/test/mfconnector.jar:/{{\tt home/clinux2/test/mfconnector.jar:/}{{\tt home/clinux2/test/mfconnector.jar:/}{{\tt home/clinux2/test/mfconnector.jar:/}{{\tt home/clinux2/test/mfconnector.jar:/}{{\tt home/clinux2/test/mfconnector.jar:/}{{\tt home/clinux2/test/mfconnector.jar:/}{{\tt home/clinux2/test/mfconnector.jar:/}{{\tt home/clinux2/test/mfconnector.jar:/}{{\tt home/clinux2/test/mfconnector.jar}}}}}}}}}}}}}}}}}}}}}}}$
est/j2ee.jar clptest
#sleep 10
done

インターコネクト LAN のケーブルを引き抜く。

MICRO Focus	Enterprise Server Administration     Version 1.04.00     server1 (10.34.69.35)
Home	Status MDS0000I OK
Actions Save Restore	Refresh Auto-refresh interval (seconds) 10
Delete All	1 - 1 of 1 Servers
Shut Down	Type Name Current Communications Processes Licenses
Options	Edit Mres ESSERVI Stopped 1 top:10.34.69.36*34940* */ 10 (server1 +) */
Users Add	2 Listeners Details
Update	Add
Display	

すると、フェイルオーバグループ failover1の停止処理により ES サーバ ESSERV1 が停止

そして server2 でフェイルオーバグループ failover1(ES サーバ ESSERV1)が開始される。



### クライアントプログラム実行結果



NEC CLUSTERPRO による Micro Focus Enterprise Server クラスター化の検証報告

at com.microfocus.cobol.connector.cci.CobolInteraction.exec(CobolInteraction.java:247) at com.microfocus.cobol.connector.cci.CobolInteraction.execute(CobolInteraction.java:174)				
at clptest.main(clptest.java:47)				
javax.resource.spi.ElSSystemException: CobolException Exception throw during open (see getCause() for more				
information) (elapsed time=268.0) for 10.34.69.38:9003 (cause class:java.net.ConnectException,				
cause:Connection refused) executing srv01.CBLAPP01				
06/03/14 15:43:55.76 CBLAPP01 server2 140				
06/03/14 15:43:58.07 CBLAPP01 server2 141				
06/03/14 15:44:00.37 CBLAPP01 server2 142				
06/03/14 15:44:02.67 CBLAPP01 server2 143				
06/03/14 15:44:04.98 CBLAPP01 server2 144				
06/03/14 15:44:07.28 CBLAPP01 server2 145 <i>server2 でサービスを引継ぎ。データファイルの連番も継続</i>				
06/03/14 15:44:09.59 CBLAPP01 server2 146				
06/03/14 15:44:11.91 CBLAPP01 server2 147				
06/03/14 15:44:14.19 CBLAPP01 server2 148				
06/03/14 15:44:16.48 CBLAPP01 server2 149				
06/03/14 15:44:37.26 CBLAPP01 server2 157 J				
[root@gw test]#				

<結果>

待機側のノードで Enterprise Server の基本プロセス MFDS のみ起動した状態からのフェ イルオーバグループの移動を確認。

また、一定時間のサービス中断後、同じサービスを待機側で提供できていることも確認。

<目的>

待機側のノードで Enterprise Server の基本プロセス MFDS と ES サーバを起動した状態 からのフェイルオーバを確認する。

フェイルオーバ前と後で同じサービスが継続される事も確認する。

<構成>



<設定のポイント>

それぞれ各ノードであらかじめ MFDS を立ち上げておく。

フェイルオーバグループではフローティング IP アドレスと共有ディスクの切り替えのみ を行なう。

クラスタの設定にインタコネクト LAN の登録をする。

<検証> server1 で手動で ESSERV1 を起動する。 FICUS Enterprise Server Administration Status MDS0000I OK Home Actions Refresh Auto-refresh interval (seconds) 10 Save Restore Import Delete All 1 - 1 of 1 Servers Show 10 Shut Down Current Communications Status Processes Type Name License Configure Edit... MFES ESSERVE Started 1 top:10.34.69.35\*33224\* \*/10 (server1 +) \*/ 2 Listeners Details Options 2 Listeners Details Users Stop... hhA Update Add... Display

server2 で手動で ESSERV1 を起動する。

<u>ところがserver2 で手動でESSERV1 を立ち上げようとするとリスナーが立ち上がらない。</u>



理由: Enterprise Server のリスナーは指定した IP アドレスが有効になっている必要がある。

<結果>

Enterprise Server の制限事項により、待機側の ES サーバを正常に立ち上がらせること ができないため、本パターンの前提条件を満たせなかった。 よって本パターンでの動作検証は不可。 <目的>

待機側のノードで Enterprise Server の MFDS を起動し、別の ES サーバによるサービス を起動した状態からのフェイルオーバを確認する。

フェイルオーバ前と後で同じサービスが継続される事も確認する。

<構成>



<設定のポイント>

それぞれ各ノードであらかじめ MFDS を立ち上げておく。 MFDS の制御はフェイルオーバグループに含めない。

failover1の開始シェルに組み込む起動コマンド

 $\texttt{casstart} \ / \texttt{rESSERV1}$ 

failover1の終了シェルに組み込む停止コマンド

casstop /rESSERV1

failover2の開始シェルに組み込む起動コマンド

casstart /rESSERV2

### failover1の終了シェルに組み込む停止コマンド

casstop /rESSERV2

failover1 に監視用の execute resource、execo1 と execo2 を追加 ファイルE 編集 ヘルプ曲

a 8 9

ዮ 🕻	cluster	名前	タイプ	監視先	コメント
9	- 🗂 Groups	diskw1	disk monitor	/dev/sdf2	
	🗕 💻 failover1	miiw1	NIC Link Up/Down	eth0	
	🛛 🔲 failover2	pidw1	pid monitor	execo1	
	- 🗂 Monitors	pidw2	pid monitor	execo2	
	Servers	userw	user mode monitor	softdog	user mode monitor
1	server1				
	server2				

execol に登録するシェルはパターン③と同様。

execo2のシェルは mdump コマンドの箇所の ESSERV1 を ESSERV2 に変更したもの。

### さらに execo1、execo2 を監視対象とする pid モニタを追加登録する。

[pidw1]監視リソースのプロパティ	×	[pidw2]監視リソースのプロパティ	×
请報 監視 異常検出		情報 監視 異常核出	
インターバル()	5 秒	インターバル①	5 秒
タイムアウト(①	30 秒	タイムアウト(1)	30 秒
リトライ回数(R)	0	リトライ回数(R)	0
監視開始待ち時間( <u>S</u> )	0 秒	監視開始待ち時間( <u>S</u> )	0 秒
監視タイミング ○ 常時(L) ● 活性時(C)		<ul> <li>監視タイミング</li> <li>○ 常時(L)</li> <li>● 活性時(C)</li> </ul>	
対象リソース(G) execo1	参照(W)	対象リソース(G) execo2	参照(W)
nice值(E)		nice(@(E)	
監視を行うサーバを選択する	サーバ(R)	監視を行うサーバを選択する	サーバ(R)
	Ok キャンセル 適用(A)	Ok	キャンセル 適用(A)

### く検証>

server1 でフェイルオーバグループ failover1 を、server2 でフェイルオーバグループ failover2 を開始しておく。



クライアントマシンで各サービスに対する Java クライアントプログラムを定周期で実行 を繰り返す。(2 サービス同時に呼び出す)

シェル例(RunjavaTest.sh)

```
#!/bin/sh
while [ "1" == "1" ]: do
java -
classpath .:/home/clinux2/test/mfcobol.jar:/home/clinux2/test/mfconnector.jar:/home/clinux2/t
est/j2ee.jar clptest
#sleep 10
done
```

次に server1 で MFDS を直接手動で停止する。 > killall -9 mfds32

すると、mfds のプロセス異常により execol の監視シェルが障害を検知しファイルオー バが開始する。 server1 ではサービスが停止し、server2 でフェイルオーバグループ failover1 が開始さ れ ESSERV1 が起動する。

・ 、 、 、 、 、 、 、 、 、 、 、 、 、 、 、 、 、 、					
生しているか、ブ	うウザの設定を調整する必要があります。				
次のことを試して	<te>t</te>				
<ul> <li>(更新) ボタンをクリックするか、後でやり直してください。</li> <li>アドレス バーにページ アドレスを入力した場合は、ページ アドレスを 正レく入力したかどうかを確認してください。</li> <li>「務核の設定を確認する口は、[ソール]メニューの [インターネットオ ヴィコン] をクリックします。【務核] タブで [ダイヤルアップの設定] グ ループの [設定] ボタン、または [LAN の設定] グループの [LAN の 設定] ボタンをフリックしてください。設定情報頃、LAN ローカル エ リアネット ワーク のの管理者か、思 F (インターネット サービス ブロバ イタ) が提供する情報と一致する必要があります。</li> <li>ネットワークの管理者がネットワークの接続の設定を使用可能につてい れば、Microsoft Windows を使用して、ネットワークの接続の設定を見つけることができま す。</li> </ul>					
,.	🔏 🗷 Enterprise Server Administra	ation			
MICRO Focus	Version 1.0400 server2 (10.34.69.36)	ation			
9. MICRO FOCUS Home	Enterprise Server Administration version 1.04.00 server2 (10.34.69.36) Status MD S000001 OK	ation			
9 • MICRO FOCUS Home Actions Save Restore Import	Contempose Server Administration (Seconds)  Contempose Server Administration (Seconds)  Status MDS00001 OK  Refresh Auto-refresh interval (Seconds)  10	ation			
Home Actions Save Restore Import Delete All Shut Down	Charles Later Prise Server Administration (Server 2 (10.34.69.36))  Status MDS00001 OK  Refresh Auto-refresh interval (seconds) 10  1 - 2 of 2 Servers	ation			
Home Actions Save Restore Import Delete All Shut Down Configure Options	Image: Server Administration           Version 104:00 server2 (10.34.69.36)           Status MDS00001 OK           Refresh         Auto-refresh interval (seconds)         10           Image: Type         Status         More Status         Formulations           Type         Status         Status         Image: Status         Image: Status         Image: Status           Edit         Mare         Status         Image: Status <tht< th=""><th>ation Sh Ucenses 067 - / 10</th></tht<>	ation Sh Ucenses 067 - / 10			
Home Actions Save Restore Import Delete All Shut Down Configure Options Users Add Update	Enterprise Server Administration     Warsin 104:00     server2 (10.34.69.36)      Status MDS00001 OK      Refresh Auto-refresh interval (seconds) 10      1 - 2 of 2 Servers      Type Name Surver Communications     Type Name Surver Communications     Type Name Surver Communications     Edit MFS ESSERVI Stated     I (pri10.34.69.36)      Listeners Details     I details	ation Sh Ucenses -/10 s			
Home Actions Save Restore Import Delete All Shut Down Configure Options Users Add Update Display Directories Statistics	Enterprise Server Administration     Warsin 104.00     server2 (10.34.69.36)     Status MDSD0001 OK      Refresh Auto-refresh interval (seconds) 10      1 - 2 of 2 Servers      Type Name Current Communications     Type Name Current Communications     Edit MFES ESSERVI Started 11 (top:10.34.69.36*)271     (cerver2)      2 Listeners Detail     Sop 2 Listeners Detail	ation Sh Ucenses eff -/ 10 a			

### クライアントプログラム実行結果(CBLAPP01)

[root@gw test]# ./RunjavaTest.sh 06/03/14 17:57:25.73 CBLAPP01 server1 205 06/03/14 17:57:29.69 CBLAPP01 server1 206 06/03/14 17:57:34.30 CBLAPP01 server1 207 06/03/14 17:57:39.29 CBLAPP01 server1 208 server1 でサービスが稼動 06/03/14 17:57:43.50 CBLAPP01 server1 209 06/03/14 17:57:48.31 CBLAPP01 server1 210 06/03/14 17:57:53.05 CBLAPP01 server1 211 java.net.ConnectException: Connection refused at java.net.PlainSocketImpl.socketConnect(Native Method) at java.net.PlainSocketImpl.doConnect(PlainSocketImpl.java:305) at java.net.PlainSocketImpl.connectToAddress(PlainSocketImpl.java:171) サービス中断中 **▼** at java.net.PlainSocketImpl.connect(PlainSocketImpl.java:158) at java.net.Socket.connect(Socket.java:452) at java.net.Socket.connect(Socket.java:402) at java.net.Socket.<init>(Socket.java:309) at java.net.Socket.<init>(Socket.java:153) at com. microfocus. cobol. connector. transport. BINPStream. createSocketWithTimeout(BINPStream. java:894) at com. microfocus. cobol. connector. transport. BINPStream. createSocketWithTimeout(BINPStream. java:825) at com.microfocus.cobol.connector.transport.BINPStream.<br/>
(init)(BINPStream.java:97) at com. microfocus. cobol. connector. transport. BINPRemoteCall. open (BINPRemoteCall. java: 176) at com. microfocus. cobol. connector. transport. BINPRemoteCall. connect(BINPRemoteCall. java:73) at com.microfocus.cobol.connector.transport.BINPRemoteCall.reconnect(BINPRemoteCall.java:145) at com.microfocus.cobol.connector.PureCobolBean.cobcall(PureCobolBean.java:285) at com.microfocus.cobol.connector.cci.CobolInteraction.exec(CobolInteraction.iava:247) at com.microfocus.cobol.connector.cci.CobolInteraction.execute(CobolInteraction.java:174) at clotest main(clotest java:47) javax.resource.spi.EISSystemException: CobolException Exception throw during open (see getCause() for more information) (elapsed time=449.0) for 10.34.69.38:9003 (cause class: java.net.ConnectException, cause:Connection refused) executing srv01.CBLAPP01 ( 繰り返し・中略 ) java.net.ConnectException: Connection refused at java.net.PlainSocketImpl.socketConnect(Native Method) at java.net.PlainSocketImpl.doConnect(PlainSocketImpl.java:305) at java.net.PlainSocketImpl.connectToAddress(PlainSocketImpl.java:171) at java.net.PlainSocketImpl.connect(PlainSocketImpl.java:158) at java.net.Socket.connect(Socket.java:452) at java. net. Socket. connect (Socket. java: 402) at java.net.Socket.<init>(Socket.java:309) at java.net.Socket.<init>(Socket.java:153) at com.microfocus.cobol.connector.transport.BINPStream.createSocketWithTimeout(BINPStream.java:894) at com. microfocus, cobol, connector, transport, BINPStream, createSocketWithTimeout(BINPStream, java: 825) at com.microfocus.cobol.connector.transport.BINPStream. at com. microfocus. cobol. connector. transport. BINPRemoteCall. open (BINPRemoteCall. java: 176) at com. microfocus. cobol. connector. transport. BINPRemoteCall. connect(BINPRemoteCall. java:73) at com.microfocus.cobol.connector.transport.BINPRemoteCall.reconnect(BINPRemoteCall.java:145) at com.microfocus.cobol.connector.PureCobolBean.cobcall(PureCobolBean.java:285) at com. microfocus. cobol. connector. cci. CobolInteraction. exec (CobolInteraction. java:247) at com. microfocus. cobol. connector. cci. CobolInteraction. execute (CobolInteraction. java: 174) at clptest.main(clptest.java:47) javax.resource.spi.ElSSystemException: CobolException Exception throw during open (see getCause() for more information) (elapsed time=513.0) for 10.34.69.38:9003 (cause class: java.net.ConnectException, cause:Connection refused) executing srv01.CBLAPPO♪ 06/03/14 17:58:58.36 CBLAPP01 server2 212 06/03/14 17:59:03.00 CBLAPP01 server2 213 06/03/14 17:59:07.97 CBLAPP01 server2 214 06/03/14 17:59:12.90 CBLAPP01 server2 215 06/03/14 17:59:17.31 CBLAPP01 server2 216 06/03/14 17:59:21.95 CBLAPP01 server2 217 server2でサービスを引継ぎ。データファイルの連番も継続 06/03/14 17:59:26.79 CBLAPP01 server2 218 06/03/14 17:59:31.15 CBLAPP01 server2 219 06/03/14 17:59:35.30 CBLAPP01 server2 220 06/03/14 17:59:37.80 CBLAPP01 server2 221 [root@gw test]#

クライアントプログラム実行結果(CBLAPP02)

06/03/14	17:57:35 12	CRI APP02	server2	7
06/03/14	17:57:30 78		sarvar?	2 8
06/03/14	17:57:44 45		sarvar?	) Q
06/03/14	17:57:44.40		sorver?	2 10
06/03/14	17:57:53 01		sorver?	) 11
06/03/14	17.57.58 50		sorver?	) 19
06/03/14	17:58:02.86		sorver?	) 12
06/02/14	17:50:02.00		server2	. 13
06/03/14	17.50.07.94		Server2	) 15
06/03/14	17.50.12.02		Server2	) 16
06/03/14	17.50.10.00		Server2	) 17
06/03/14	17.50.21.00		Server2	) 10
06/03/14	17.50.20.20		Server2	2 10
00/03/14	17.50.30.91		serverz	19
06/03/14	17.58.30.09		server2	20
06/03/14	17.50.40.01		server2	2 21
06/03/14	17.50.40.39		server2	22
00/03/14	17.58.50.10		serverz	23
00/03/14	17.50.54.73		serverz	24
00/03/14	17.50.04.01		serverz	20
00/03/14	17.59.04.21		serverz	20
00/03/14	17.59.08.97		serverz	27
06/03/14	17.59.13.52		serverz	28
06/03/14	17.59.18.41	CBLAPPUZ	serverz	29
06/03/14	17.59:23.06	UBLAPP02	server2	2 3U
06/03/14	1/:59:27.71	CBLAPP02	server2	2 31

同様に同じ状態から server2の MFDS を直接手動で停止し、server2の ESSERV2 が server1 にフェイルオーバすることが確認する。

<結果>

待機側のノードで Enterprise Server の MFDS を起動し、別の ES サーバによるサービス を起動した状態からのフェイルオーバグループの移動を双方向で確認。 また、一定時間のサービス中断後、同じサービスを待機側で提供できていることも確認。

# まとめ

本検証では代表的な異常検出例3パターンと待機状態例4パターンについて動作を確認 しました。

パターン	タイトル	テスト結果	説明
1	手動切り替え	0	CLUSTERPRO 標準機能でフェイルオーバ確認
2	インターコネクト障害	0	CLUSTERPRO 標準機能でフェイルオーバ確認
3	業務障害	0	MDUMP ユーティリティの出力を解析するシェ
			ルを自作してフェイルオーバ確認
4	OS のみで待機	0	MFDS 制御、ES サーバ制御をフェイルオーバ
	(シングルスタンバイ)		グループに含めフェイルオーバ確認
5	MFDS 起動で待機	0	ES サーバ制御をフェイルオーバグループに
	(高速スタンバイ 1)		含めフェイルオーバ確認
6	DS サーバ起動で待機	×	Enterprise Server のリスナーは定義する
	(高速スタンバイ 2)		IP が立ち上げ時に実在しなければ立ち上が
			らないため、このパターンは実装不可
$\overline{\mathcal{O}}$	別の業務を提供しなが	0	フェイルオーバグループを分けて別のサー
	ら待機		ビスを継続的に提供しながらフェイルオー
	(相互スタンバイ)		バ確認

異常検出のパターンでは、今回マイクロフォーカス提供の mdump ユーティリティにより を使用し、ステータスを監視する事で異常を検出することが出来ました。 但し、本運用においては、ユーザプログラム又は関連ソフトウェアの監視なども含め、 監視方法や監視対象を再検討する必要があります。

待機状態のパターンでは、待機系のノードでは ES サーバをあらかじめ立ち上げておく事ができない事が分かりました。

また MFDS は開始状態でも停止状態からもフェイルオーバは可能でした。

但し、複数のフェイルオーバグループで ES サーバを立ち上げるケース等では MFDS は ES サーバの起動前にあらかじめ起動しておく必要があるため、基本的には OS の起動時に MFDS を自動起動するように設定することをお奨めします。

サーバアプリケーションソースサンプル(cblapp01.cbl)

FILE-CONTROL.	
SELECT APPFILE ASSIGN TO	EXTERNAL APPFILE01
ORGANIZATION INDE	XED RECORD KEY RKEY
ACCESS MODE RANDO	M
FILE STATUS IS AP	-STATUS.
DATA DIVISION	
FILE SECTION.	
FD APPFILE.	
01 AF-REC.	
05 RKEY PIC X(6)	
05 AF-COUNTER PIC 9(10	).
WORKING-STORAGE SECTION.	
01 WK-DATE PIC X(6)	
01 WK-TIME PIC X(8)	
01 AP-STATUS pic x(02	).
LINKAGE SECTION.	
01 LK-DATETIME PIC X (20	).
01 LK-PGMNAME PIC X (10	).
01 LK-HOSTNAME PIC X (20	).
01 LK-COUNTER PIC 9(10	).
PROCEDURE DIVISION	
USING LK-DATETIME LK-P	GMNAME LK-HOSTNAME LK-COUNTER.
1.	
PERFORM GEI-DATETIME.	
MOVE CBLAPP01	10 LK-PGMNAME.
CALL 'gethostname'	USING LK-HOSTNAME BY VALUE 20.
OPEN I-O	APPFILE.
MOVE SPACE	10 RKEY.
READ APPFILE	INVALID KEY CONTINUE.
MOVE AF-COUNTER	IU LK-CUUNIER.
	INVALID KEY CONTINUE.
GLUSE APPFILE.	
EXII PROGRAM.	
GET_DATET I ME	
	те
	IL. ME
STRING WK_DATE (1.2) '/'	₩L. ₩K_DATE(2·2) '/' ₩K_DATE(5·2) ' '
WK-TIME(1:2) ' '	WK_TIME(3:2) '.' WK_TIME(5:2) '.'
WK-TIME(7:2) .	IMITED BY SIZE INTO IK-DATETIME
	THILE DI GILL HAIV LA DAILITHE.
	FILE-CONIROL. SELECT APPFILE ASSIGN TO ORGANIZATION INDE ACCESS MODE RANDO FILE STATUS IS AP DATA DIVISION FILE SECTION. FD APPFILE. O1 AF-REC. O5 RKEY PIC X(6) O5 AF-COUNTER PIC 9(10) WORKING-STORAGE SECTION. O1 WK-DATE PIC X(6) O1 WK-TIME PIC X(6) O1 WK-TIME PIC X(8) O1 AP-STATUS pic x(02) LINKAGE SECTION. O1 LK-DATETIME PIC X(20) O1 LK-OATETIME PIC X(20) O1 LK-OATETIME PIC X(20) O1 LK-OATETIME PIC X(20) O1 LK-OATETIME PIC X(20) O1 LK-OUNTER PIC 9(10) PROCEDURE DIVISION USING LK-DATETIME LK-P 1. PERFORM GET-DATETIME LK-P 1. PERFORM GET-DATETIME LK-P 1. PERFORM GET-DATETIME. MOVE 'CBLAPPO1' CALL 'gethostname' OPEN I-O MOVE SPACE READ APPFILE MOVE AF-COUNTER ADD 1 REWRITE AF-REC CLOSE APPFILE. EXIT PROGRAM. GET-DATETIME. ACCEPT WK-DATE FROM DA ACCEPT WK-DATE FROM TI STRING WK-DATE (1:2) '/' WK-TIME (1:2) ':' WK-TIME (7:2) DEL

このドキュメントの内容は予告なしに変更される場合があります。このドキュメントの 内容の保証や、商品性又は特定目的への適合性の黙示的な保証や条件を含め明示的又は 黙示的な保証や条件は一切無いものとします。マイクロフォーカス株式会社および日本 電気株式会社は、このドキュメントについていかなる責任も負いません。

また、このドキュメントによって直接又は間接にいかなる契約上の義務も負うものでは ありません。

このドキュメントを形式、手段(電子的又は機械的)、目的に関係なく、マイクロフォ ーカス社および日本電気株式会社の書面による事前の承諾なく、複製又は転載すること はできません。

Copyright© 2006 Micro Focus. All Rights Reserved.

Micro Focus は Micro Focus 社の登録商標、Enterprise Server, Server Express は同 社の商標です。

CLUSTERPRO ®は日本電気株式会社の登録商標です。

その他記載の会社名、製品名は、各社の商標または登録商標です。