

## 目次

### Unify Vision系

<a href="#">ACCELL/SQL から UnifyVISION Release2へのコンバージョン</a>	… 1
<a href="#">ACCELL/SQL から UnifyVISION Release2へのコンバージョン(C-Hooks)</a>	… 4
<a href="#">UnifyVISION Release2 から Release3以降の上位リリースへのバージョンアップ</a>	… 5
<a href="#">ACCELL/IDS からUnifyVISION Release3以降のリリースへのコンバージョン</a>	… 9
<a href="#">ACCELL/SQL からUnifyVISION Release3以降のリリースへのコンバージョン</a>	… 10
<a href="#">UnifyVISION Release3 以降で上位リリースへのバージョンアップ</a>	… 11



## ACCELL/SQL からUnifyVISION Release2へのコンバージョン

ACCELL/SQL インストールディレクトリ /home/ASQL  
UnifyVISION Release2インストールディレクトリ /home/VISION2

### 1. UnifyVISIONアプリケーションの変換

#### Step1 UnifyVISIONとDataServer の環境変数の設定

```
PATH=/home/VISION2/bin:$PATH
VISION_HOME=/home/VISION2
GALAXYHOME=/home/VISION2/gui
LANG=japanese
LANG_DIR=jpn_jae
DISPLAY=jupiter
UNIFY=/home/ASQL/lib
CLIENTINFO=/home/DS_DB
DCMFILE=/export/home/ohmura/vdev/tutorial/tutorial.dcm
DBCONN=TUTORIAL._U2K
```

#### 【tutorial.dcmの内容】

```
[TUTORIAL_U2K]
DBTYPE=U2000
DBHOST=jupiter
DBUSER=ohmura
DBPASSWORD=BeEsXRO
DBNAME=file.db
DBPATH=/home/DS_DB
DBSCHEMA=PUBLIC
```

注) これらの設定は、使用する環境により変更する。

注) \*.az, \*.fzファイルがない場合にはStep1の前に作成しておく。

#### Step2 makefileの作成と編集

ACCELLアプリケーションのあるディレクトリに移り、acl2uvユーティリティを実行する。

```
$ acl2uv
```

makefileのvcpl部分を以下のように変更する。

#### 【変更前】

```
-vcpl -warm all -log $*.log -script $?
```

#### 【変更後】

```
-vcpl -warm all -log $*.log -script $? -dcmfile $(DCMFILE)
      -dbconn $(DBCONN)
```

#### Step3 makeの実行

```
$ make -f vision.mak
```

#### Step4 visionを起動しStep3で作成したアプリケーションを取り込む。

プロファイルのデータベース・プリファレンスを設定して、接続を確認する。

**Step5 Step3**で作成されたログ(filename.log)を参照して4GLの編集を行う。  
 以下の変更なしでもアプリケーションのコンパイルは問題なくできるが、将来にわたってUnifyVISIONを使用することを前提に変更すべき箇所を以下に示す。  
 ここに示したものは、使用頻度が高いと思われるもの。

UnifyVISIONで変更された4GL文(左側:ACCELL 右側:UnifyVISION)

●APPLCAITION master_form	FORM CLASS master_form
●AFTER APPLICATION	ON DESTROY
●BEFORE FORM	ON CREATE
◎ENABLE ZOOM TO	ENABLE ZOOM TO CHILD FORM form_name OF CLASS form_name
●CHOOSE NEXT FORM	ON CHOOSE NEXT FORM
●ON PREVIOUS FORM	ON DISMISS FORM
-----	
●PREVIOUS_FORM	DISMISS_FORM
-----	
●NEXT_FIELD	NEXT_FIELD_NAME
●CUR_NEXT_FIELD	CURRENT_NEXT_FIELD_NAME

UnifyVISIONで追加されたもの

◎FIELD SECTIONにBEGIN ENDを入れる

UnifyVISIONで削除されたもの

- ◎DISPLAY TRIM
- ◎REFRESH SCREEN
- ◎REQUIRED FORMS
- 
- ◎BLINK
- ◎LOW\_INTENSITY
- ◎REVERSE
- ◎UNDERLINE

### UnifyVISIONで設定が細かくなったもの

AUTO\_COMMITについては、追加・削除・更新などについて別々に指定できるようになった。

◎ AUTO\_COMMIT TX\_MODE\_ADD\_RECORD  
TX\_MODE\_DELETE\_RECORD  
TX\_MODE\_UPDATE\_RECORD

SET COMMANDはなくなり、アトリビュート(属性)で指定するようになった。例えば、

◎SET COMMAND ‘NEXT\_FORM’:ACTION TO ‘DISABLED’

は、以下のように変える必要がある。

SET ‘NEXT\_FORM’:AUD\_ACTION TO ‘DISABLED’

また、ACTIONについてもAUD\_ACTIONとFIND\_ACTIONの2つを指定する必要がある。

### UnifyVISIONで条件が厳しくなったもの

FINDなど、QUEUE COMMANDに置き換えが必要なものがある。

◎NEXT ACTION IS QUEUE COMMAND

以上の変更は、ac12uvを実行する前にシェルスクリプトなどで行っても良い。ただし、ログにはACCELLでは未知のキーワードであるとのメッセージが出力される。

NULLの設定は、NULLを指定して関数を使って変換する。例えば、DATEタイプでは、以下のように行う。

◎SET TODAY\_DATE TO str\_to\_date\$('\*/\*/\*')

は、以下のように変更する。

SET TODAY\_DATE TO to\_date\$(NULL)

### UnifyVISIONデザイナーで修正が必要なもの

◎リターンを押しても次フィールドにいかない

各フィールドの“タブストップ”属性をTRUEに設定する。

(フォーム・プロパティパネルのフィールド順で“タブストップ”をクリックする)

◎マウスでクリックしてフィールド間を移動できない

フォーム属性のフィールド・クリックをTRUEに設定する。

(フォーム・プロパティパネルの対話型操作で“フィールド・クリック”をクリックする)



## ACCELL/SQL からUnifyVISION Release2へのコンバージョン(C-Hooks)

ACCELL/SQL インストールディレクトリ /home/ASQL  
UnifyVISION Release2インストールディレクトリ /home/VISION2

### 1. UnifyVISIONユーザ関数(C-Hooks)の相違点

ACCELL/IDSからのコンバージョンの場合には、“ACCELL/IDSからACCELL/SQL へのコンバージョン(C-Hooks)”の“1. ACCELLユーザ関数(C-Hooks)の相違点”を参照のこと。ACCELL/SQLとUnifyVISIONは同一仕様。

ただし、インクルードしている`chookincl.h`を`chookinc.h`に変更すること。

### 2. カスタムマネージャの再リンク

#### Step1 ユーザ関数の変更

上記の対応表をもとに該当箇所をACCELL/SQL (VISION)のコードに置き換える。

#### Step2 UnifyVISIONカスタムマネージャの再リンク

通常の方法でカスタムマネージャを再リンクする。

例えば、`namechk.c`というファイルをコンパイルし、カスタムマネージャにリンクするには、以下のようにする。リンク前には`HDATYPE`を設定する必要がある。`PATH`の設定やデータベース固有の設定(例えば`ORACLE_HOME`など)もしておく必要がある。

```
$ cc -c -I$UNIFY/.. namechk.c
$ cc -c -I$UNIFY/.. chooktb.c

$ HDATYPE="U2000 ORACLE"; export HDATYPE
$ vision.ld custvisn namechk.o chooktb.o
```

メモリを大量に消費するため、失敗した場合には、以下のように`swap`を追加して再度行う。

```
Solaris2.xの例
# mkfile 30M /export/home0/swap
# swap -a /export/home0/swap
```



## UnifyVISION Release2からRelease3以降の上位リリースへのバージョンアップ

UnifyVISION Release2インストールディレクトリ c:\¥vision2  
UnifyVISION 新リリースインストールディレクトリ c:\¥vision\_new

注意：UnifyVISION 新リリースがインストールされている必要があります。  
UnifyVISION Release2はコンバージョン作業には必要ありません。

### 1. UnifyVISION Releaseアプリケーションの変換

Unify VISION Releaseの開発環境には、Release2のアプリケーションを新リリースフォーマットに自動的にアップグレードする機能がある。（パーティションに分割されたアプリケーションを除く）  
ここでは、その手順について説明する。

#### Step1 UnifyVISION 新リリースの環境変数の設定を行う

```
PATH=c:\¥vision_new¥bin;%PATH%  
VISION_HOME=c:\¥vision_new  
LANGDIR=jpn_sjis  
DISPLAY=hostname:0.0  
HOME=c:\¥vis_home
```

（HOMEはRelease2の開発環境のHOMEを指定する）

注）これらの設定は、使用する環境により変更する。

#### Step2 新リリースの開発環境を起動する。

#### Step3 新リリースの開発環境で、Release2のプロジェクトまたはフォルダをオープンすると自動的にアップグレードを行うためのアップグレード・ダイアログが表示される。

#### Step4 アップグレード・ダイアログの使用方法

ダイアログでは、個々のアプリケーション、ライブラリ、またはリポジトリを段階的にアップグレードするか、1度に全てをアップグレードするかを選択できる。  
以下のオプションの中から選択しアップグレードを実行する。  
実行後、新リリースのクラス・ライブラリが自動的に作成される。

- |               |   |
|---------------|---|
| アップグレード (U)   | 指示されたオブジェクトだけを新リリースのクラス・ライブラリにアップグレードする。<br>アップグレードされる項目の名前を表示するには、ドロップダウン・リストを使用する。                          |
| 全てアップグレード (A) | 全ての項目（アプリケーション、リポジトリ、オブジェクト・ライブラリ）を新リリースのクラス・ライブラリにアップグレードする。<br>アップグレードの後でクラス・ライブラリのアイコンがFolder ウィンドウに表示される。 |

スキップ (S) 指示されたオブジェクトをアップグレードしない。

完了 (D) アップグレード・プロセスを終了する。

#### Step5 プロファイルの確認

開発環境で、アップグレード後のクラス・ライブラリをオープンし、プロファイルの設定を確認する必要がある。  
正しく設定されたプロファイルを選択してから、「ファイルメニュー」→「カレントプロファイルにする」を選択しプロファイルをカレントにする。正しいプロファイルがカレントになっていない場合には、次のコンパイルに失敗する可能性がある。

#### Step6 アップグレード後の全コンパイル

クラス・ライブラリ内の全てのオブジェクトをコンパイルし直す必要がある。  
オープンしたクラス・ライブラリに対してウィンドウの「ファイルメニュー」→「構築」→「全て」を実行し、全てのオブジェクトをコンパイルする。

#### Step7 アップグレード後のRelease2アプリケーションについて

新リリースのクラス・ライブラリへのアップグレードの際、Release2のアプリケーション(\*.uva, \*.uvl)はそのまま残る。

#### Step8 アップグレードの再実行

アップグレードが期待通りに行われなかった場合は、Release2のアプリケーションを修正し、アップグレードをやり直すことが可能。2度目のアップグレードを開始するには、アップグレード済みのクラス・ライブラリをフォルダから削除する必要がある。  
クラス・ライブラリを削除するには、クラス・ライブラリのアイコンを選択し、「編集」→「削除」を選択する。

## 2. パーティショニング・アプリケーションの変換

Release2でパーティションに分割されたアプリケーションを新リリースにアップグレードするためには、手作業でアップグレードを実行する必要がある。  
ここでは、その手順について説明する。

#### Step1 UnifyVISION 新リリースの環境変数の設定を行う

```
PATH=c:¥vision_new¥bin:$PATH
VISION_HOME=c:¥vision_new
LANGDIR=jpn_sjis
DISPLAY=hostname:0.0
HOME=c:¥vis_home
```

( HOMEはRelease2の開発環境のHOMEを指定する)

注) これらの設定は、使用する環境により変更する。

#### Step2 新リリースの開発環境を起動する。

#### Step3 パーティショニング・アプリケーションに必要な新しい構成要素を追加

1.アプリケーションの各パーティションのプロファイルで、[分散] パネルの [分

散機能を有効にする] ダイアログオプションを選択する。

2. [分散] パネルでパーティション・グループ・プロファイルの名前を指定する。  
パーティション・グループはランタイムにuorouterユーティリティ・プロセスによって管理される。
3. アプリケーション・ネットワークの全てのホストに対して、ホスト構成ファイルを使用してパーティションの起動コマンドを指定する。  
(デフォルト:\$VISION\_HOME¥lib¥uohost.ini)
4. uohostdユーティリティを使用してグローバル・ネームサービスを起動。  
(uvnatedユーティリティは使用できない)

#### Step4 手作業によるサービス・クラスのアップグレードの手順

1. 新リリースの関連するクラス・ライブラリのVISIONクラス・ブラウザ・ウィンドウでServiceクラスのサブクラスを作成する。
2. クラスのスクリプトを編集する。
3. サービスにPRIVATE変数があれば、PRIVATEセクションを追加し、\_CREATEグローバル関数の各変数をPRIVATEセクションにコピーする。
4. Release2のサービスにコンストラクタ \_CREATEがあれば、ON CREATEメソッドを追加し、\_CREATEグローバル関数に入っている文をON CREATEメソッドにコピーする。
5. Release2のサービスにデストラクタ\_DESTROYがあれば、ON DESTROYメソッドを追加し、\_DESTROYグローバル関数に入っている文をON DESTROYメソッドにコピーする。
6. Release2のサービス・インタフェース内で宣言されている関数ごとに同じ名前でもETHODとパラメータ宣言を作成する。メソッドのグローバル関数内の文を新しいサービス・クラスのMETHODにコピーする。
7. Release2サービス内のグローバル関数ごとに（サービス・インタフェースにはない、サービスのローカル関数）新しいサービスクラス内にLOCAL FUNCTIONセクションを作成し、Release2のグローバル関数の内容に対応するローカル関数にコピーする。

### 3. Release2アプリケーションをランタイムでそのまま使用する

(パーティションに分割しているアプリケーションを除く)

パーティションに分割していないRelease2のUnify VISIONアプリケーションは、新リリースのVISIONランタイム・マネージャでそのまま実行することができる。  
アプリケーションを再コンパイルする必要もない。

#### Step1 UnifyVISION 新リリースの環境変数の設定を行う

```
PATH=c:¥vision_new¥bin:$PATH
VISION_HOME=c:¥vision_new
LANGDIR=jpn_sjis
DISPLAY=hostname:0.0
HOME=c:¥vis_home
```

(HOMEはRelease2の開発環境のHOMEを指定する)

注) これらの設定は、使用する環境により変更する。

#### Step2 新リリースの開発環境を起動する。



**Step3** Release2のアプリケーション、オブジェクト・ライブラリ、リポジトリ・コンテナは、Release2のアイコンのまま 新リリースの開発環境で表示される。これらに対して実行できる操作は、カット、コピー、ペースト、削除、エクスポート、実行、デバッグに限られる。



## ACCELL/IDSからUnifyVISION Release3以降のリリースへのコンバージョン

ACCELL/SQLインストールディレクトリ

/home/AIDS

UnifyVISION Release インストールディレクトリ

c:\¥vision

ACCELL/IDSからUnifyVISION Release3以降のリリースへのコンバージョンは、1度に行うことができない。

ACCELL/IDS → ACCELL/SQL → UnifyVISION Release2 → UnifyVISION 新リリースと3段階のコンバージョン作業を行う必要がある。

### 1. ACCELL/IDSからACCELL/SQL へのコンバージョン

コンバージョン作業の詳細は、「ACCELL/IDSからACCELL/SQL へのコンバージョン」に関するドキュメントを参照する。

### 2. ACCELL/SQL からUnify VISION Release2へのコンバージョン

コンバージョン作業の詳細は、「ACCELL/SQL から UnifyVISION Release2へのコンバージョン」に関するドキュメントを参照する。

### 3. Unify VISION Release2からRelease3以降の上位リリースへのバージョンアップ

コンバージョン作業の詳細は、「UnifyVISION Release2からRelease3以降の上位リリースへのバージョンアップ」に関するドキュメントを参照する。



## ACCELL/SQL からUnifyVISION Release3以降のリリースへのコンバージョン

ACCELL/SQLインストールディレクトリ /home/ASQL  
UnifyVISION リリースインストールディレクトリ c:\¥vision

ACCELL/SQL からUnifyVISION Release3以降のリリースへのコンバージョンは、1度に行うことができない。

ACCELL/SQL → UnifyVISION Release2 → UnifyVISION 新リリースと2段階のコンバージョン作業を行う必要がある。

### 1. ACCELL/SQL からUnify VISION Release2へのコンバージョン

コンバージョン作業の詳細は、「ACCELL/SQL から UnifyVISION Release2へのコンバージョン」に関するドキュメントを参照する。

### 2. Unify VISION Release2からRelease3以降の上位リリースへのバージョンアップ

コンバージョン作業の詳細は、「UnifyVISION Release2からRelease3以降の上位リリースへのバージョンアップ」に関するドキュメントを参照する。



## UnifyVISION Release3 以降で上位リリースへのバージョンアップ

UnifyVISION 現リリースインストールディレクトリ c:¥vision  
UnifyVISION 新リリースインストールディレクトリ c:¥vision\_new

注意 : UnifyVISION 新リリースがインストールされている必要があります。  
UnifyVISION 現リリースはコンバージョン作業には必要ありません。

### 1. UnifyVISION 現リリースクラス・ライブラリをそのまま使用する

UnifyVISION Release3以降で開発したクラスライブラリは、アップグレードを行わずに Unify VISION 新リリースの開発環境、及びランタイム・マネージャでそのまま使用できる。

(日本語名オブジェクトを持つクラス・ライブラリを除く)

### 2. 日本語名オブジェクトを持つクラス・ライブラリの変換

UnifyVISION Release3以降で開発したクラス・ライブラリに日本語名のオブジェクト (テーブル名を含む) がある場合は、新リリースでそのまま使用できない。

uvcheck -fixユーティリティを使用して日本語名オブジェクトのタグを修正する必要がある。ここでは、その手順について説明する。

**Step1** UnifyVISION 新リリースの環境変数の設定を行う。

```
PATH=c:¥vision_new¥bin:$PATH  
VISION_HOME=c:¥vision_new  
LANGDIR=jpn_sjis  
DISPLAY=hostname:0.0  
HOME=c:¥vis_home
```

( HOMEは現リリースの開発環境のHOMEを指定する)

注) これらの設定は、使用する環境により変更する。

**Step2** uvcheck -fixユーティリティで日本語オブジェクトのタグを修正する。

コマンドラインから該当するクラス・ライブラリに対して以下を実行する。

```
$ uvcheck -fix *.ucl (日本語名のオブジェクトを持つクラス・ライブラリ名)
```

### 3. Unify VISION/Webアプリケーションの変換

現リリースのUnify VISION/Webのアプリケーションを新リリースにアップグレードするためには、新リリースの Unify VISION/Web開発環境で、全てのアプリケーションを再コンパイルする必要がある。

ここでは、その手順について説明する。

**Step1** UnifyVISION 新リリースの環境変数の設定を行う。

```
PATH=c:¥vision_new¥bin:$PATH  
VISION_HOME=c:¥vision_new  
LANGDIR=jpn_sjis  
DISPLAY=hostname:0.0
```



HOME=c:\%vis\_home

( HOMEは現リリースの開発環境のHOMEを指定する)

注) これらの設定は、使用する環境により変更する。

**Step2** 新リリースの開発環境を起動する。

**Step3** 新リリースの開発環境で、現リリースで作成したクラス・ライブラリをオープンする。

**Step4** Unify VISION/Webのコンパイラ・オプションの指定

クラス・ライブラリ・ウィンドウの「ファイルメニュー」→「コンパイラ・オプション」を実行すると、ライブラリ・コンパイラ・オプション・エディタが表示される。

クラス・ライブラリによってターゲット環境の指定を変更する。

Webクライアント・クラス・ライブラリ: 「VISION/Web」を選択する。

「標準Unify VISION」の選択も可能

Webサーバ・クラス・ライブラリ : 「標準Unify VISION」のみを選択する。

**Step5** プロファイルの確認

オープンしたクラス・ライブラリのプロファイルの設定を確認する必要がある。正しく設定されたプロファイルを選択してから、「ファイルメニュー」→「カレントプロファイルにする」を選択しプロファイルをカレントにする。正しいプロファイルがカレントになっていない場合には、次のコンパイルに失敗する可能性がある。

**Step6** クラス・ライブラリの全コンパイル

クラス・ライブラリ内の全てのオブジェクトをコンパイルする。

ウィンドウの「ファイルメニュー」→「構築」→「全て」を実行し、全てのオブジェクトをコンパイルする。