Micro Focus メインフレームソリューション

スターターズキット

7. Enterprise Developer データベース連携

7.1 チュートリアルの準備

本チュートリアルで使用する例題プログラムは、キットに添付されている DBtutorial.zip に圧縮されて います。これを C:¥ の直下に解凍しておきます。

また、作業用に C:¥work というフォルダを作成しておきます。

ここでは JCL バッチからの使用方法として紹介していますが、CICS や IMS プログラムで Oracle, DB2, SQL Server などとのデータベース連携を XA リソース方式で行う場合は同様の設定となります。

なお、対象のデータベースクライアントは同じ環境にインストール済みで、プリコンパイラも含め接続が 確認された環境であることが前提です。

本書作成時に使用したデータベース環境は下記の通りです。

Oracle

Oracle クライアント 11g 11.2.0.4.0 64bit DB: リモート環境(Oracle 10g 10.2.0.1.0 64bit) (Oracle Pro*COBOL プリコンパイラ使用)

DB2

DB2 クライアント 9.7.900.250 64bit DB: ローカル環境(DB2 9.7.900.250 64bit) (DB2 PREP プリコンパイラ使用)

SQL Server

ODBCドライバー: SQL Server Native Client 11.0 DB: ローカル環境(SQL Server 2012 R1 64bit) (ODBC 使用)

7.2 Enterprise Developer の起動

まず、Enterprise Developer を起動し、新たなワークスペースを作成します。

1) メニューから Enterprise Developer for Eclipse を選択して起動します。

_					
	サンプル	+		Micro Focus Enterprise Developer	
	ツール	+		Micro Focus License Manager	►
	データ ツール	•		Micro Focus Rumba	►
	構成	•		Microsoft Silverlight	►
I MF	Enterprise Developer fo	r Eclipse		Microsoft Silverlight 4 SDK - 日本語	►
MF	ドキュメント - Eclipse			Microsoft_Silverlight 5.SDK - 日本語	►
1	Souther State Marshold	Enterprise [Dev	eloper for Eclipse の起動 008	•

2) 以下のダイアログでは C:¥work を指定し [OK] をクリックします。

The second se	ワークスペース・ランチャー	×
ワークスペースの選択 Eclipse は、ワークスペースと呼 このセッションに使用するワークス	flばれるフォルダーにプロジェクトを保管します。 スペース・フォルダーを選択してください。	
ワークスペース(<u>W</u>): C:¥worl	k v 参	照(<u>B</u>)
□この選択をデフォルトとして使	を用し、今後この質問を表示しない(U) OK キ	ヤンセル

3) 「ようこそ」タブの右の X をクリックしてバナーを閉じます。以下のように Eclipseの COBOLパー スペクティブが開きます。

197 第6回 3-25700 3-2570 3-2570 3-2570 3-2570 3-2570 3-2570 3-2570 3-2570 3-2570 3-2570 3-2570 3-2570 3-2570 3-25 第一日	2 (5 .000)
# • £)•∅•∞•∞•	ा (<mark>फ</mark> ुरवक) -
22 問題 23 タスク ロプロパティー	04 QI 21 Q - 13 - 13
	11 <u>8</u> NB 8939 [27/94-]

7.3 プロジェクトの新規作成

作成されたワークスペースに新たなプロジェクトを作成します。

1) [ファイル] > [新規] > [メインフレーム COBOL プロジェクト] を選択します。

COBOL - I	IL - Eclipse										
ファイル(F)	編集(E)	ナビゲート(N)	Search	プロジョ	ロクト	(P)	実行(R)	ウィンドウ(\	N)	へルプ(H)	
新規(N)			Alt+S	nift+N ▸	19	COB	OL JVM	プロジェクト			
ファイル	、を開く(.)				1	XT	ンフレー	ム COBOL プロ	コジ	ェクト	

 以下のダイアログでプロジェクト名を指定します。ここでは "DBDEMO" と命名します。[完了] をク リックします。

メインフレーム COBOL プロジェクト	
CICS®、JCL または IMS® アプリケーションを作成するためのプロジェクトで	g. 🗾
フロジェクト名(P): DBDEMO	
プロジェクト テンプレートを選択	
🚰 Micro Focus テンプレート	
<u></u>	ンプレートの設定を構成
□テンプレートの参照	
場所:	参照
 デフォルト・ロケーションの使用(<u>D</u>) 	
ロケーション(上): C:¥work¥DBDEMO	参照(<u>R</u>)
ファイル・システムを選択(Y): デフォルト 🗸	
? 完了(E)) キャンセル

3) 空のメインフレームプロジェクトが作成されます。



4) [プロジェクト] > [自動ビルド] をオフにしておきます。



 6) 作成されたプロジェクトに必要なプロパティの設定を行います。COBOL エクスプローラ内で DBDEMO を右クリックして [プロパティー] を選択します。



 以下のようにプロジェクトのプロパティダイアログが開きます。ここでメインフレームプロジェクトの 各種設定を行うことができます。左側ペインのツリービューにて [Micro Focus] > [Project 設定] > [COBOL] を開き、以下のように設定してください。本チュートリアルで使用する DB 例題プログラ ムは IBM Enterprise COBOL の方言を使用しています。

 > リワイス Micro Focus Project 設定 BMS > [COBOL] > IMS > アセンブラコンパイラ アセンブラコンパイラ アセンブラコンパイラ アセンブラコンパイラ アセンブラコンパイラ アセンブラコンパイラ アビンブラコンパイラ アビンブラコンパイラ アビンブラコンパイラ アビンブラコンパイラ アビンブラコンパイラ アビンブラコンパイジアイルの生成 > ビルド構成 > 受けい構成 > 受けい構成 > ジース フォーマット: 協定 > UP(ムのコピー処理: ○ COPY > 「日本マット ○ COPY > UP(ムのコピー処理: ○ COPY > UP(ムのコピーベリン ○ GNT にコンパイル(D) ○ GNT にコンパイル ※ UP(ムのコピーベリン ※ COPY ○ GNT にコンパイル ※ UP(ムのコピーベリン ※ COPY ○ GNT にコンパイル ※ UP(ムのコピーベリン ※ COPY ○ GNT にコンパイル ※ UP(ムのコピーベリン ※ GNT にコンパイル ※ UP(ムのコピーベリン ※ GNT にコンパイル ※ UP(ムのコピーベリン ※ GNT にコンパイル ※ GNT にコンパイル ※ GD(ロンビーベリン ※ GNT にコンパイル ※ GNT にコンパイル ※ GNT にコンパイル ※ GD(ロンビーベリン ※ GNT にコンパイル ※ GNT にコンパール ※ GNT (ロンパール ※ GNT	フィルター入力	COBOL	⇔ ◄ ⇔ ▼ ▼
 Project 設定 BMS BMS COBOL DCBOL DCCBOL DCBOL DCBOL	▷ リソース ▲ Micro Focus	☆ 字 住 企・	
	▲ Project 設定 BMS	天于来口,	ASCII
 ▶ IMS ▶ Pt2>75 コンパイ5 Pt2>75 コンパイ5 Pt2>75 コンパイ5 Pt2>75 コンパイ5 Pt2>75 コンパイ5 Pt2>75 コンパイ5 Pt2>75 コンパイ5 Pt2>75 コンパイ5 Pt2>76 コンパイカの生成 □はていないまし、 □はたり、 UlxFlow ▶ EULW構成 □コンパイルを生成 □コンパイルレ(D) □コンパイルレ □コンパイルレ □コンパイルレ □コンパイルレ □コンパイルレ □ロ道可能なエラーを含める(レベル E) ▼ ■加増令: □加増令: □加増令: □コンパイルレの進売(D) □加増令: □コンパイルレの進売(D) □加増令: □コンパイルレの進売(D) □加増令: □コンパイルレの進売(D) □コンパイルレの進売(D) □加増へ(D) □コンパイルレの進売(D) □加増へ(D) □コンパイルレの進売(D) □加増へ(D) □コンパイルレの進売(D) □100 □コンパイルレの進売(D) □100 □コンパイルレの進売(D) □100 □コンパイルレの進売(D) □1000 □1000 □1000 □10	▷ COBOL	8:0/2.	Enterprise COBOL for z/OS
 ▶ アセンブラ コンパイラ アセンブラ リンカ ビルド パス ▶ ビルド構成 ▶ ジロバ構成 ▶ マゴバサ構成 ▶ マゴバサ用にコンパイル(D) Project Facets □ 出力の表示 ♥ Alidation ビルジー プロジェクト参照 実行/デバッグ設定 警告レベル: 回復可能なエラーを含める(レベル E) ・ ・	⊳ IMS	ソース フォーマット:	固定 >
ビルド パス 当ちっイルの生成 シビルド構成 リストライルを生成 シ 実行時構成 リストライルを生成 Project Facets コカの表示 Task Tags J. GNT にコンパイル(D) ジレダー プロジェクト参照 支行/デバッグ設定 警告レベル: 置加指令: 回復可能なエラーを含める(レベル E) 道加指令: 「ゴロロ ブフォレトの復元(D) 適用(A)	アセンブラ コンパイラ アセンブラ リンカ	メインフレームのコピー処理:	COPY Y
 ▶ ビルド構成 ▶ 実行時構成 ▶ 実行時構成 ♥ 写/いが用にコンパイル(D) □ 出力の表示 □ ふの表示 ♥ Alidation ビノダー プロジェクト参照 要行/デバッグ設定 警告レベル: 回復可能なエラーを含める(レベル E) 最大エラー数: □ 100 追加指令: (アンホレトの復元(D) (アンホレトの復元(D) (アンホレトの復元(D) (アンホレトの復元(D) 	ビルドパス	□ 指令ファイルの生成	
 ▶ 実行時構成 ▶ マラバッグ用にコンパイル(D) □ 出力の表示 ▼ Aildation ビノダー プロジェクト参照 実行/デバッグ設定 警告レベル: 回復可能なエラーを含める(レベル E) 最大エラー数: 100 追加指令: 	▷ ビルド構成	□リストファイルを生成	
HoleCraces Task Tags Validation ビルダー プロジェクト参照 実行/デバッグ設定 警告レベル: 回復可能なエラーを含める(レベル E) 最大エラー数: 100 追加指令: 「フォルトの復元(D) 適用(A)	▷ 実行時構成 Project Eacots	✓ デバッグ用にコンパイル(D)	
Validation ビルダー プロジェクト参照 実行/デバッグ設定 ・ ・	Task Tags		
ビルダー プロジェクト参照 実行/デバッグ設定 警告レベル: 回復可能なエラーを含める(レベル E) ▼ 最大エラー数: 100 100 追加指令: 「ジェルトの復元(D)」 運用(A)	Validation	✓ .GNT (CL2/(1))	
第二人小・ 回復可能なエラーを含める(レベル E) 最大エラー数: 100 追加指令: 「ジオルトの復元(D)」 第二人小・ 第二人小・ 第二人小・ 「ジオルトの復元(D)」	ビルダー プロミックト券昭		
最大エラー数: 100 追加指令:	実行/デバッグ設定	警告レベル:	回復可能なエラーを含める(レベル E) V
追加指令:		最大エラー数:	100
		追加指令:	デフォルトの復元(D) 適用(A)
	3		
	\odot		

7) 使用するデータベース製品に合わせ、[追加指令]欄へ埋め込み SQL 対応のプリプロセッサの設 定を追加します。

<Oracle の例>

COBSQL プリプロセッサを使用しますので、以下の指令を追加します。

入力值)P(COBSQL) ENDP

追加指令:

P(COBSQL) ENDP

<DB2 の例>

DB2 ECM プリプロセッサを使用しますので、以下の指令を追加します。

入力值)DB2(DB=SAMPLE,VALIDATE=RUN)

追加指令:

DB2(DB=SAMPLE,VALIDATE=RUN)

<SQL Server の例>

OPENESQL プリプロセッサを使用しますので、以下の指令を追加します。

入力值)SQL(DBMAN=ODBC,BEHAVIOR=JCL,TARGETDB=MSSQLSERVER)

追加指令:

SQL(DBMAN=ODBC,BEHAVIOR=JCL,TARGETDB=MSSQLSERVER)

 8) 既に前項で埋め込み SQL 対応のプリプロセッサを使用するために追加指令を行っていますので、 左側ペインのツリービューにて [Micro Focus] > [Project 設定] > [COBOL] > [SQL プリプロセッ サ] を開き、以下のように [SQL プリプロセッサに使用] のチェックをオフにして、[OK] をクリック します。(必要に応じて使い分け可能です)

> リソース	
Micro Focus	
⊿ Project 設定	
BMS	プリプロセッサの種類: OpenESQL
COBOL	指合:
CICS プリプロセッサ	18 13.
SOL プリプロセッサ	フィルタテキストを入力
追加のプリプロセッサ	
> IMS	設定
アセンブラ コンパイラ	ACCESS
アセンブラ リンカ	ALLOWNULLCHAR
ビルド パス	ANSI92ENTRY
⊿ ビルド構成	AUTOCOMMIT
BMS	AUTOFETCH

 2) 左側ペインのツリービューにて [Micro Focus] > [ビルド構成] > [COBOL] を開き、以下のように [ターゲットの種類]として[すべて INT/GNT ファイル]を選択かつ、[プラットフォーム ターゲット]は [64 ビット]を選択し、一旦、[OK] をクリックし、プロパティーダイアログを閉じます。

フィルター入力	COBOL
 > リソース Micro Focus Project 設定 BMS > COBOL > IMS > アセンブラ コンパイラ アセンブラ リンカ ビルド パス レド構成 	New Configuration [使用中] 出力パス: New Configuration.bin エントリポイント:
BMS ▷ COBOL ▷ アセンブラ コンパイラ アセンブラ リンカ イペント ▷ リンク	ターゲット設定 ターゲットの種類 すべて INT/GNT ファイル ・ プロジェクトの COBOL の設定の上書き

10) 再度、COBOL エクスプローラ内で DBDEMO を右クリックして [プロパティー] を選択します。左 側ペインのツリービューにて [Micro Focus] > [Project 設定] > [COBOL] を開き、以下のように [.GNT にコンパイル] のチェックをオンにし、[OK] をクリックします。

7ィルター入力	COBOL			← • → • •
▷ リソース				
▲ Micro Focus ▲ Project 設定	文字集合:	ASCII	~	
BMS	言語方言:	Enterprise COBOL for z/OS	~	
> IMS	ソース フォーマット:	固定	~	
▷ アセンブラ コンパイラ アセンブラ リンカ	メインフレームのコピー処理:	COPY	~	
ビルドパス	□ 指令ファイルの生成			
▶ ビルド構成	□リストファイルを生成			
▷ 実行時構成	 デバッグ用にコンパイル(D) 			
Project Facets	□出力の表示			
Server	✓.GNT にコンパイル			
Task Tags				
Validation				

7.4 埋め込み SQL 付き COBOL ソースのインポート

作成されたプロジェクトに例題プログラムをインポートします。

オリジナルの埋め込み SQL 付き COBOL ソースは予め Micro Focus 形式の COBOL ソースにプリコ ンパイルしてから使用することも可能ですが、製品機能であるプリプロセッサ機能からプリコンパイラを 呼び出して内部的にプリコンパイルすることにより、オリジナルのソースのイメージのままでデバッグが 可能となり管理もし易くなります。ここでは後者の方法を紹介します。

11) COBO エクスプローラ内で DBDEMO を右クリックして [インポート] > [インポート] を選択します。



12) 以下のダイアログで [一般] > [ファイルシステム] を選択し、[次へ] をクリックします。

選択 ローカル・ファイル・システムから既存のプロジェクトへリソースをインポートします。	Ľ
インボート・ソースの道訳(<u>S</u>): フルタースカ ○ ファイル・システム ○ 酸疗力シェクトをワークスペースへ □ 設定 > CVS > CVS > EB > D BE > Micro Focus インターフェイスマッパー > Micro Focus インターフェイスマッパー > Micro Focus インターフェイスマッパー > Micro Focus インターフェイスマッパー > Web Services > Web Services > CVS > Micro Focus インターフェイスマッパー > Focus インターフェイスマッパー > Micro Focus インターフェイスマッパー > Focus インターフェイスマッパー > CVS > Micro Focus インターフェイスマッパー > Focus インター > Focus インター	*
? < 戻る(B) 次へ(N) > 完了(E)	キャンセル

13) 参照ボタンをクリックし、以下のダイアログで C:¥DBturotial を選択し、[OK] をクリックします。

インポート元のディレクトリーを選択します。
▲ 💼 ローカル ディスク (C:) ヘ
▶] app
▷ 🌗 DB2
DBtutorial
Inetpub
📔 PerfLogs
🛛 🎍 Program Files 🗸 🗸 🗸
フォルダー(<u>E</u>): DBtutorial
新しいフォルダーの作成(N) OK キャンセル

14) C:¥DBturotial 直下に解凍されているファイルをすべて選択し、[完了] をクリックします。

ファイル・システム				
コーカル・ファイル・システムからリソースを	インポートします。			/
Rのディレクトリーから(Y): C:¥DBtutorial			•	参照(R)
🗷 🗁 DBtutorial		✓ ● dbdemo1.jd ✓ ● TBLCRTE.cbl ✓ ● TBLDROP.cbl ✓ ● TBLFETC.cbl ✓ ● TBLISRT.cbl		
タイプのフィルター(T)) すべて 洗フォルダー(L): MSSDEMO ゼジョン	選択(S) 選択をすべて解除(D)		参照(W)
警告を出さずに既存リソースを上書き(O) Create top-level folder(C) 拡張(A) >>				

7.5 プログラムのコンパイル

1) [自動的にビルド] をオンにします。



2) 自動ビルドが実行され、以下のように再コンパイルが完了します。なお、警告が出力される場合が ありますが、この例題では無視して構いません。



7.6 XA スイッチモジュールの生成

この環境で実行するプログラム内では、XA スイッチモジュール経由でデータベースと接続することになります。ここでは使用するデータベース製品に合わせた XA スイッチモジュールを作成します。作成手順は以下の通りです。

- 1) プリコンパイルを行うため、下記フォルダを書き込み権限があるフォルダ配下へコピーします。当チュートリアルでは C:¥直下へコピーします。
 - 【理由1】 Oracle のプリコンパイラはパスに英数字とアンダースコア以外は許容しない
 - 【理由2】 製品関連フォルダの書き込み権限によるトラブルを避ける
 - 【コピー元フォルダ例】: Micro Focus 製品があるパス
 - C:¥Program Files (x86)¥Micro Focus¥Enterprise Developer¥src¥enterpriseserver¥xa
 - 【コピー先フォルダの例】: 書き込み権限があるパス

C:¥xa



 スタートメニューで、[Micro Focus Enterprise Developer] > [ツール] > [Enterprise Developer コ マンドプロンプト(64-bit)] を管理者として実行します。この例題では 64 ビットですが、DB に合わせ て 32 ビットもしくは 64 ビットを選択してください。
 C:¥xa ディレクトリに移動します。

C:¥Users¥Administrator¥Documents>cd	c:¥×a
c:¥xa>	

3) 使用するデータベース製品に合わせた XA スイッチモジュールを build コマンドで作成します。

<Oracle の例>

build < ora11 または ora12 > を実行します。ここでは下記コマンドを実行します。

build ora11

C:¥Users¥tarot¥Documents>cd c:¥xa c:¥xa>build orall Building 64-bit switch module... Micro Focus COBOL Yersion 2.2.02170 Copyright (C) Micro Focus 1984-2014. All rights reserved. * Cobsql Integrated Preprocessor * CSOL-1-018: Oracle ブリコンパイラトランスレータを起動します。 * CSOL-1-020: Oracle ブリコンパイラの出力を処理中。 * CSOL-1-020: Oracle ブリコンパイラの出力を処理中。 * CSOL-1-0101: COBSOL: チェッカへの引き渡しを完了しました。 * チェック終了: エラー(はありません-コード生成を開始します * Generating ESORAXA * Data: 8704 Code: 34336 Literals: 2224 Micro Focus COBOL - CBLLINK utility Version 2.2.2.44 Copyright (C) Micro Focus 1984-2014. All rights reserved. Microsoft (R) Incremental Linker Version 10.00.40219.01 Copyright (C) Microsoft Corporation. All rights reserved. ESORAXA.obj cbl1ds00001130.obj Creating library ESORA11XA.lib and object ESORA11XA.exp Microsoft (R) Manifest Tool version 5.2.3790.2076 Copyright (c) Microsoft Corporation 2005. All rights reserved. c:¥xa>_

<DB2 の例>

ここでは下記コマンドを実行します。

build db2

C:¥xa>build db2 Building 64-bit switch module Micro Focus COBOL - CBLLINK utility Version 2.2.2.39 Copyright (C) Micro Focus 1984-2014. All rights reserved.
Micro Focus COBOL Version 2.2.02161 Copyright (C) Micro Focus 1984-2014. All rights reserved. * チェック終了:エラーはありません-コード生成を開始します * Generating FSDR2XA
* Data: 3792 Code: 8051 Literals: 992 Microsoft (R) Incremental Linker Version 10.00.40219.01 Copyright (C) Microsoft Corporation. All rights reserved.
ESDB2XA.obj cbllds0000D640.obj Creating library ESDB2XA.lib and object ESDB2XA.exp Microsoft (R) Manifest Tool version 5.2.3790.2076 Copyright (c) Microsoft Corporation 2005. All rights reserved.
C:¥xa>

JCL 機能で IKJEFT01 ユーティリティを使用する場合は、xxx1pc.dll サブモジュール生成も必要なた め、build db21pc < DB のエイリアス > を実行します。ここでは下記コマンドを実行します。 build db21pc sample

DB2 ヘアクセスするためのログイン画面が表示されますので、ユーザ ID とパスワードを入力します。 "build db21pc sample ユーザ ID パスワード"を実行するとログイン画面は出力されません。

Micro Focus SQL Logon
Connect failed with SQL code -30082. SQL database SAMPLE
Enter user name and password:
User name: db2inst1
Password: *****
□ <u>S</u> ave
Save to INI file
OK Cancel

C:¥xa>build db21pc sample Building 64-bit switch module... Micro Focus COBOL Version 2.2.02161 Copyright (C) Micro Focus 1984-2014. All rights reserved. * チェック終了: エラー(ありません- コード生成を開始します * Generating ESDB20PC * Data: 5992 Code: 11078 Literals: 976 Micro Focus COBOL - CBLLINK utility Version 2.2.2.39 Copyright (C) Micro Focus 1984-2014. All rights reserved. Microsoft (R) Incremental Linker Version 10.00.40219.01 Copyright (C) Microsoft Corporation. All rights reserved. ESDB20PC.obj cload state Linker Version 5.2.3790.2076 Copyright (c) Microsoft Corporation 2005. All rights reserved. If you have upgraded your installation from an earlier version of Net Express or Studio, then you may need to additionally rebuild the main switch module, that is, execute : build db2 Unable to locate the COBOL bin64 directory. If you intend to execute JES-initiated transactions under Enterprise Server you must copy ESDB20PC.DLL to a directory on your PATH, such as your COBOL bin64 directory. If you do not do so, then such transactions will not be able to communicate with the database server. C:¥xa>

正常に作成されました。メッセージにあるように "ESDB2OPC.DLL" ファイルを製品の bin64 フォルダ

ヘコピーします。32 ビットの場合は bin フォルダヘコピーします。

例) C:¥Program Files (x86)¥Micro Focus¥Enterprise Developer¥bin64

<SQL Server の例>

ここでは下記コマンドを実行します。

build mssql



正常に作成されました。メッセージにあるように "ESMSSQL.DLL" ファイルを製品の bin64 フォルダ

ヘコピーします。32 ビットの場合は bin フォルダヘコピーします。

例) C:¥Program Files (x86)¥Micro Focus¥Enterprise Developer¥bin64

xaswitch.lib の読み込み失敗でリンクエラーとなる場合は、環境変数「LIB」へ左記ファイルパスを 追加し、コマンドプロンプトを再起動後に再ビルドしてください。

32 ビット例: C:¥Program Files (x86)¥Windows Kits¥8.0¥Lib¥win8¥um¥x86

64 ビット例: C:¥Program Files (x86)¥Windows Kits¥8.0¥Lib¥win8¥um¥x64

🛛 « Windows Kits	> 8.0 > Lib > win8 > um > x86	v C
^	名前 翻 Xāudio2.lip	更新日時 2012/0//25 16:0/
	🏙 xaswitch.lib	2012/07/25 16:12

- 4) 使用するビット数に合わせた ODBC データソースを [コントロールパネル]>[管理ツール]>
 [ODBC データソース] から追加します。
 注意)ODBC データソース名称が Enterprise Server へ設定する XA リソース定義の OPEN 文字
 列で使用する DSN 名称となります。
- 5) XA スイッチモジュールを有効にするため下記設定を確認します。

[コントロールパネル]>[管理ツール]>[コンポーネントサービス]>[コンピューター]>[マイコンピュ ータ]>[Distributed Transaction Coordinator]>[ローカル DTC]まで展開します。



[ローカル DTC]を右クリックして[プロパティ]を選択し、[セキュリティ]タブへ移動します。

[XA トランザクションを有効にする]にチェックされていることを確認、もしくはチェックして有効にします。

ローカル DTCのプロパティ ? ×				
トレース中 ログ セキュリティ				
セキュリティの設定 マネットワーク DTC アクセス(D) クライアントおよび管理				
✓ リモート クライアントを許可する(<u>R</u>) ✓ リモート管理を許可する(<u>M</u>)				
- トランザクション マネージャー通信 ■ 受信を許可する(<u>I</u>) ■ 送信を許可する(<u>W</u>)				
 ● 相互認証を必要とする(L) ○ 着信呼び出し/側には認証を必要とする(G) ○ 認証を必要としない(<u>T</u>) 				
✓ XA トランザクションを有効にする(X) ✓ SNA LU 6.2 トランザクションを有効にする	5 <u>(E</u>)			

7.7 リージョンの作成とXAリソースの設定

本チュートリアルでは例題 JCL を Enterprise Server にサブミットして実行します。これには Enterprise Developerに内蔵されているテスト用のメインフレームランタイム環境を使用します。これは Enterprise Server と呼ばれるミドルウェアであり、Enterprise Developer には開発用の Enterprise Server が内蔵されています。これがメインフレームアプリケーションのテスト・デバッグのために使用さ れます。またマイグレーションにおいては本番実行用の Enterprise Server 製品を使用します。

 Enterprise Developer 内で開発用の Enterprise Server を操作するにはサーバーエクスプローラ を使用します。サーバーエクスプローラは COBOL エクスプローラの後ろに隠れていますので、こ のタブをクリックします。[ローカル] を右クリックして [Administration ページを開く] を選択します。



Enterprise Server の管理コンソールが開きます。メインフレームアプリケーションの実行のためには新たなサーバー (リージョン)を定義する必要があります。画面下部の [追加] ボタンをクリックします。



3) 以下の画面に遷移します。新規に作成するサーバー名として DBDEMO を入力し、[動作モード] は[64-bit]を選択後、[次へ] をクリックします。

	ステータス MDS0000I OK Wed Oct 15 11:40:04 201	4
Home	[mfuser] [Page id: a00	1]
アクション アドレス更新 エクスポート インポート すべて削除	サーバー追加 (Page 1 of 3): サーバー名: DBDEMO	
構成 オブション セキュリティ 表示	動作モード: ③ 32-bit ● 64-bit You cannot change your choice of working mode once a server is created, although you can change it when copying or importing a server.	
ティレクトリ 統計 セッション	キャンセル 次へ >>	

 以下の画面では "Micro Focus Enterprise Server with Mainframe Subsystem Support" のラジ オボタンを選択し、[次へ] をクリックします。

	ステータス MDS0000I OK Wed Oct 15 11:42:28 2014
Home	[mfuser] [Page id: a101
アクション アドレス更新 エクスポート インボート すべて削除	サーバー追加 (Page 2 of 3): サーバー名: DBDEM0
構成 オプション セキュリティ 表示	MICRO Focus Enterprise Server An enterprise server that provides an execution environment for COBOL application programs running as services in a service orientated architecture.
ディレクトリ 統計 セッション ジャーナル	MFES Micro Focus Enterprise Server with Mainframe Subsystem Support An enterprise server that also provides an execution environment for CICS applications that have been migrated from the mainframe.
ヘルブ このページ Support Feedback	You can change your choice of server type later.
237444444444444444444444444444444444444	(<戻る) 次へ >>

5) 以下の画面では、[TN3270 リスナーの作成] のチェックをオフにし、[追加] をクリックします。

生成オプション:
TN3270リスナーの作成 🗌 using port
構成情報
記明
Micro Focus Enterprise Server
キャンセル << 戻る 追加

 以下のように64ビット対応の DBDEMO が新規に追加されました。作成された DBDEMO のプロ パティを設定します。左端の [編集...] ボタンをクリックします。



7) [サーバー] > [プロパティ] > [MSS...] > [JES] > [General] のタブを開き、以下のように各プロパテ ィを設定し、[Apply] をクリックします。

なお、これらのフィールドでは改行を入れないように注意してください。

① [ジョブ入力サブシステム有効] のチェックをオンにします。

② [JES プログラムパス] はジョブステップで実行される COBOL アプリケーションの探索先パス ですので、開発プロジェクトの bin ディレクトリを指定しています。

③ [システムカタログ] は、JES リージョンで仮定されるマスターカタログの置き場所です。

④ [データセットの省略時ロケーション] はジョブの実行ともに生成されるスプールデータやカタロ グされるデータセットの置き場所です。

⑤ [システムプロシージャライブラリ] は、ジョブの実行時に使用されるプロシージャライブラリの 名前です。

<mark>プロバティ…</mark> 構成 診断… 過去の統計			
一般】 XAリソース (1) ┃ MSS (✔) MQ 】.	スクリプト) アクセス権) セキュリティ		
メインフレーム サブシステム サポート有効: 🗹			
CICS (✔) JES (✔) IMS PL/I			
General Initiators (1) Printers (0)			
ジョブ入力サブシステム有効: 🗹			
JESプログラムパス:			
C:\Work\DBDEMO\New_Configuration.bin	\bigcirc		
C: WORK DBDEMO (DATAFILE (CATALOG. DAT	\bigcirc		
データセットの省略時ロケーション:			
C:\work\DBDEMO\DATAFILE			
システムプロシージャライブラリ:			
Fileshare Configuration Location:	🔀 要事項		
	パスは全て半角英数字で指定してください。		
Apply	以降も同様です。		

8) 「JES...」> [Initiators] タブを開き、[追加] をクリックします。

CICS (∢) Ì	JES (✔))]	IMS	PL/I
一般 Initiators (0)		F	Printers (0		
追加					

9) 以下のように入力し [追加] ボタンをクリックします。

一般〕	Initiators (0)	Printers (0))
Add	d Initiator		
名前:			
INITAE	3C		
Class:			
ABC			
説明:			
クラス	\BC のイニシエータ	_	
++)	ノセル 追加		

10) 以下のようにジョブクラス A, B, C に対する JES イニシエータが定義されます。

一般	Initiators (1)		Printers (0)
	名前	クラス	説明
編集	INITABC	ABC	クラスABCのイニシェーター
追加			

- 11) リージョンで指定したデータセットを格納する DATAFILE フォルダを C:¥work¥DBDEMO 配下に 作成しておきます。
- 12) JES リージョンで指定したシステムプロシジャライブラリのフォルダを C:¥work¥DBDEMO¥DATAFILE 配下に作成しておきます。
- 13) XA リソースを使用可能とするための準備を行います。[サーバー] > [プロパティ] > [XA リソース] を開き、[追加] ボタンをクリックします。

サーバー リスナー(3) サービス (4) ハンドラ (3)	バッケージ (0)
プロパティ 構成	診断 過去の統計	
一般 XAリソース (0) MSS (✔) スクリプト	アクセス権 セキュリティ
通加		

14) 使用するデータベース製品に合わせた XA リソースを設定します。

[ID]: IKJEFT ユーティリティに渡す DSN TSO コマンドの SYSTEM パラメタで渡すサブシステム名 (このチュートリアルでは XADB とします)

[名前]: XA リソース名として任意の名前を指定しますが、Oracle は "ORACLE_XA" 固定です。

[モジュール]: XA スイッチメインモジュールのパスとDLL ファイル名 <<指定 DLL 名>> Oracle:ESORA11XA.DLL、DB2:ESDB2XA.DLL、SQLServer:ESMSSQL.DLL

[OPEN 文字列]: DB オープン用の指定(設定済みの DB 環境設定に依存)

<Oracle の例>

例えば以下のように設定し、[追加] ボタンをクリックします。

一舟	βÌ	XAリソープ	z (3)	MSS (✔)	MQ	スクリプト	アク
П	D:						
	XAD)B					
4	名前	:					
	OR/	ACLE_XA					
Ŧ	Eジ:	ո−ル։					
F	C:\xa\ESORA11XA.DLL						
Ċ	DPE	N文字列:					
	ORACLE_XA+SesTm=100+SqlNet=tok-par+Acc=P/scott/tiger						
C	CLO	SE文字列:					
Γ							
	兑明	:					
Γ							
有	言効						

追加キャンセル

<DB2の例>

例えば以下のように設定し、[追加] ボタンをクリックします。

一般 XAUソース (0) MSS (✔) MQ 】スクリプト 】アク
ID:
名前:
DB2_XA モジュール:
C:\xa\ESDB2XA.DLL
OPEN文字列: DB=SAMPLE.uid=db2inst1.pwd=ibmdb2.AXLIB=casaxlib
CLOSE文字列:
有効: 🗹
追加キャンセル

<SQL Server の例>

例えば以下のように設定し、[追加] ボタンをクリックします。 前項で作成した ODBC データソース名称を OPEN 文字列に指定します。

・DSN= ODBC データソース名称

一般 【 XAYソース (2) MSS (✔) 】 MQ 】 スクリナ
ID: XADB
名前:
SQLSVR_XA
モジュール:
C:\xa\ESMSSQL.DLL
OPEN文字列:
DSN=SQLSVR
CLOSE文字列:
說明:
有効: 🗹
追加キャンセル

15) DBDEMO にプロジェクトを関連付けます。



16) DBDEMO を開始します。

🕏 ታビ 😫 CO 🛛	<u>- 17 - 18 - 19</u>	
⊿ 💻 ローカル [lo	calhost:86]	
DBDE	10	
ESDEI	新規(N)	+
🖻 🛓 IMSDI	開始	

17) XA スイッチモジュールが Enterprise Server に正しくロードできると、Enterprise Server Administration のコンソールログには以下のメッセージが出力されます。(DB2 の場合、 NAME(DB2 for WINDOWS) となり、SQL Server の場合、NAME(Micro Focus SQL Server XA struct) となります。)



7.8 データベースアクセスを含む COBOL バッチプログラムの実行

1) COBOL エクスプローラ内で dbdemo1.jcl をダブルクリックしエディタで開きます。



このジョブは4つのステップから構成され下記の様に連携されています。

• STEP01:

DB にテーブルを新規作成

STEP02:

JCL に記述された SYSIN データをテーブルに追加

- STEP03:
- テーブルからデータを全件読み出し SYSOUT へ出力
- STEP04:
- テーブルを削除
- COBOL エクスプローラ内で dbdemo1.jcl を右クリックして [Enterprise Server へのサブミット] を選択します。



3) 以下のように DBDEMO1 ジョブが実行された旨のメッセージがコンソールに表示されます。

🖳 א-עכב 🖳

Enterprise Server: ローカル/DBDEMO

サーバー: DBDEMO 正常に停止しました 開始 サーバー: DBDEMO CASCD0167I ES Daemon successfully auto-started 13:18:33 CASCD0050I ES "DBDEMO" initiation is starting 13:18:33

<u>サーバー: DBDEMO 正常に起動されました</u>	
JCLCM0187I JOB01000 DBDEM01 JOB SUBMITTED (JOBNAME=DBDEM01,JOBNUM=01000) 1	13:18:59
JCLCM0180I JOB01000 DBDEM01 Job ready for execution. 13:18:59	
Processed "C:¥work¥DBDEMO¥dbdemo1.jcl"	

ハイパーリンクをクリックすると自動的にスプールが表示されますので、以降は、JCL チュートリアルを 参考にして、スプールビューの Out Hold キューおよび Output キューを開き、実行された DBDEMO1 ジョブの結果を確認するなど、バッチプログラムのデバッグも同様に実行してみてください。また、各 DB 付属の管理ツールを使用すると、各ステップにおける DB テーブルの状況が確認できます。

以上、

<