Micro Focus Server Express 5.1 J for SPARC Solaris WebOTX Application Server V8.4 動作検証結果報告書

2011 年 10 月 24 日

マイクロフォーカス株式会社

1 検証概要、目的及びテスト方法

1.1 検証概要

Micro Focus Server Express 5.1 J の Enterprise Server が提供する Java EE Connector 機能は、JCA仕様準拠のコンテナとして多くの Java EE準拠アプリケーションサーバーについて動作検証がなされています。本報告書は、NECのWebOTX Application Server(以降WebOTX ASと略記)との JavaEE Connector の接続性を検証し、報告するものです。

1.2 目的及びテスト方法

Micro Focus Server Express 5.1 J の Enterprise Server が提供する JavaEE Connectorは、現在 WebSphere, WebLogic, JBoss などとの連携が動作保証されています。しかしEnterprise Serverは、JCA仕様準拠のコンテナとして、設計上は JCA仕様に準拠したすべてのアプリケーションサーバーとの連携が可能です。
WebOTX ASはJCAの仕様に準拠しており、理論的には Micro Focus Enterprise ServerのEISとも連携するはずです。今回、以下のテストプログラムを実行することによって、このことを実際に検証しました。

(1) 渡された2つの数字パラメタを加算してその結果を返すCOBOLサブルーチンを使用
(2) Interface Mapping Toolkit が自動生成した EJB と ServletクライアントをWebOTX AS上で運用し、COBOLを呼び出す

2 使用ハードウェア及びソフトウェア一覧

Sun Fire V240 US3i-1.5GHz * 2 Memory 8Gb
Solaris 10
Oracle Java DK 1.6.0_21
Micro Focus Server Express 5.1 WrapPack 6
WebOTX AS Express V8.41 (8.41.00.00 (build 20110714))

作業用環境として Windows XP PC を使用し、Internet Explorer 8及び CygWin X Serverを利用

3 テスト内容

以下に実施したテストの概要を述べます。詳細な手順については補足に記載します。

(1) 使用した COBOLロジック
渡された2つの数字パラメタを加算してその結果を返す簡単なCOBOLサブルーチンを使用
(2) 使用したリソースアダプタ
$COBOL/ lib/javaee5/oracleweblogic10/mfcobol-notx.rar
WebLogic 10.x にディプロイするのに適した形式でパッケージされたものであり、JavaEE仕様に照らして最も標準的な提供形態です。
(3) 使用した Enterprise Server
既定義の ESDEMO をそのまま使用。
(4) 使用したWebOTX ASのドメイン
既定義の domain1 をそのまま使用。
(5) 使用した JavaEEアプリケーション
Server Express の Interface Mapping Toolkit がデプロイ時に自動生成する EJB と、自動生成される Webモジュールクライアントを使用。

4 結果

上記のテストを実行した結果、正常に実行されることを確認しました。詳細な結果については補足に記載します。

5 テスト結果及び考察

最新の JavaEE標準をサポートする WebOTX ASで、既存の Micro Focus Server Express 5.1 Jの

JavaEE Connector 接続を問題なく使用できることが検証できました。

以上

補足. 検証の手順

1. 前提条件

本検証では、各ソフトウェアはデフォルトでインストールされたままの状態になっていることを仮定しています。Server Express はデフォルトのインストール先に Enterprise Server も含めてインストールされており、 出荷時設定のサーバー ESDEMO がそのままの状態で利用可能になっているものとします。 検証を始める前に ESDEMO を開始状態にしておきます。

WebOTX AS もデフォルトでインストールされており、管理者ユーザ admin/adminadmin で、出荷時設定のドメイン domain1 が利用可能になっているものとします。

ここでは、以下の簡単な COBOL 例題プログラム CALCU.cbl を使用します。第一、第二の引数を加算し、結果を RESULT に返すというだけのプログラムです:

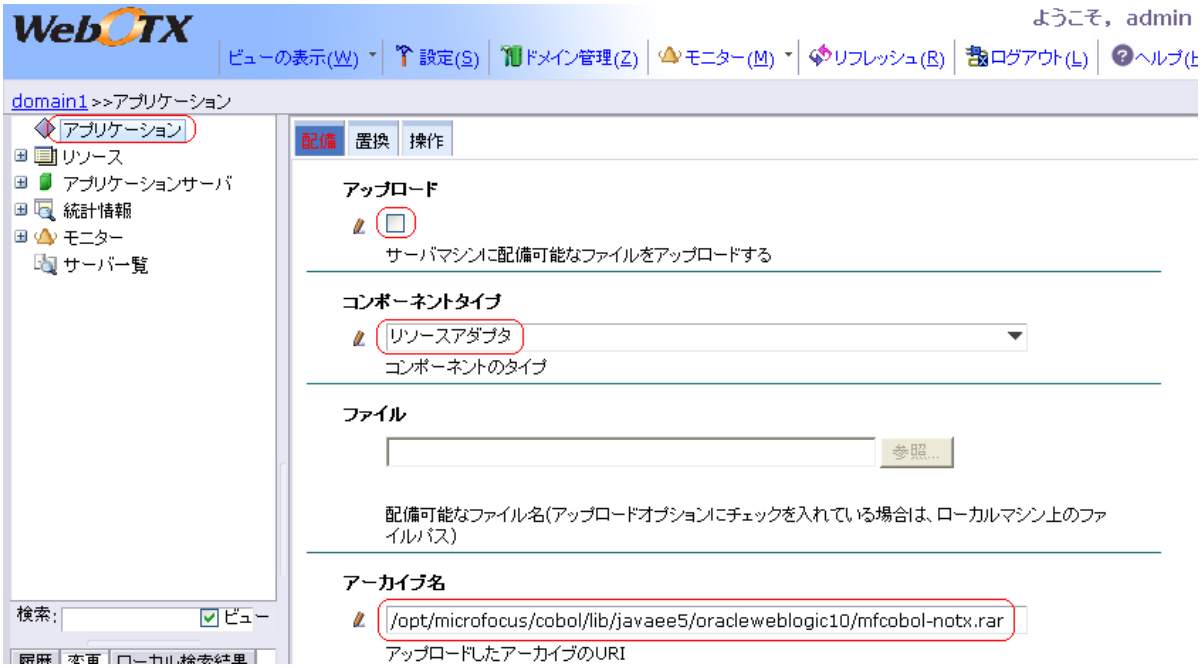
```
      LINKAGE SECTION.
      01 CALCULATOR.
         05 ARG1       pic 9(5) comp-3.
         05 ARG2       pic 9(5) comp-3.
         05 RESULT     pic 9(5) comp-3.
      procedure division using CALCULATOR.
          move ARG1 to RESULT
          add  ARG2 to RESULT
          exit program.
```

2. リソースアダプタの設定

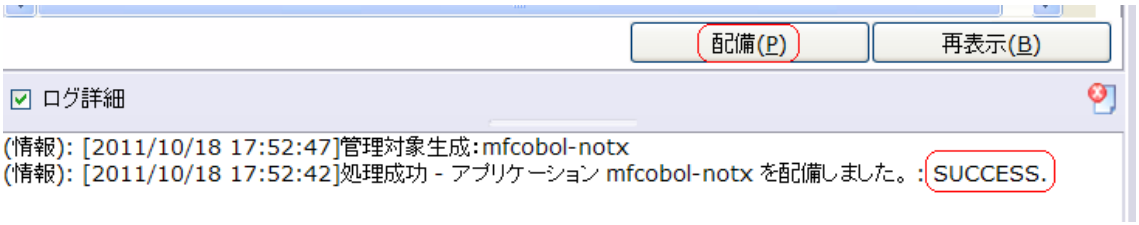
Server Express5.1JのEnterprise ServerへのJCA接続は、WebLogic などのいくつかのJava EEアプリケーションサーバーで動作保証されており、それらのそれぞれに対応したリソースアダプタが個別に製品に添付されています。これらは基本的に同じ物ですが、アプリケーションサーバーの種類によって必要となるマニフェストやデプロイメントディスクリプタが個別にパッケージ化されています。
今回の検証対象である WebOTX に対応したものは用意されていませんので、ここでは比較的標準的な内容を持っている WebLogic 向けのリソースアダプタを使用しました。

1. Windows PC上の Internet Explorer で http://<Solarisサーバーアドレス>:5858/ を指定し、WebOTX 統合運用管理コンソールを開きます。

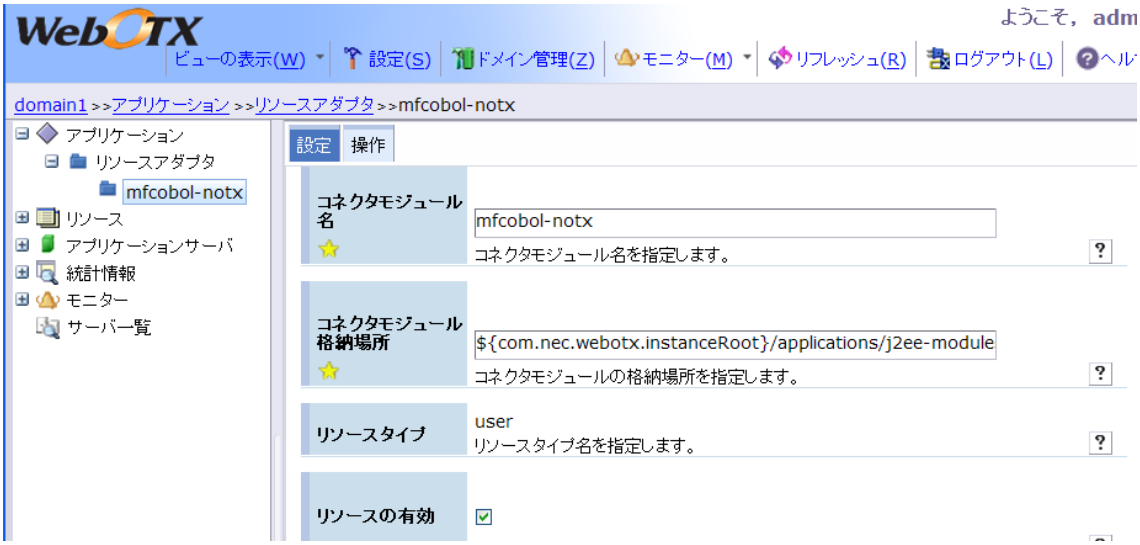
2. 左ペインのツリービューで [アプリケーション] をクリックします。以下のように、[アップロード] のチェックをオフにし、[コンポーネントタイプ] のプルダウンで [リソースアダプタ] を選択し、[アーカイブ名] に $COBOL/ lib/javaee5/oracleweblogic10/mfcobol-notx.rar を指定します。($COBDIR の部分は Server Express のインストール先パス名に置き換えます)



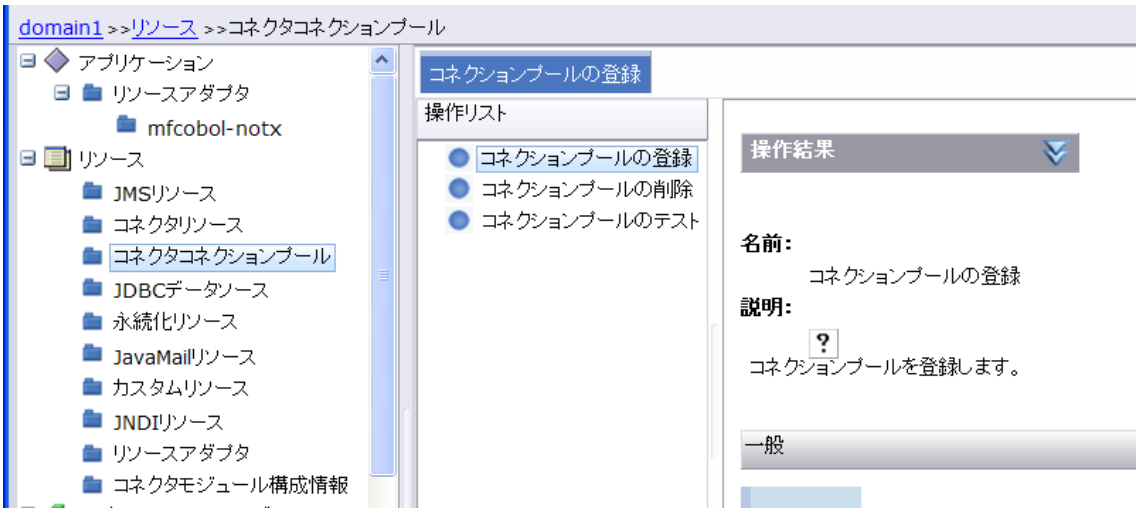
3. [配備] をクリックします。 以下のようにリソースアダプタの配備が SUCCESS になることを確認します。



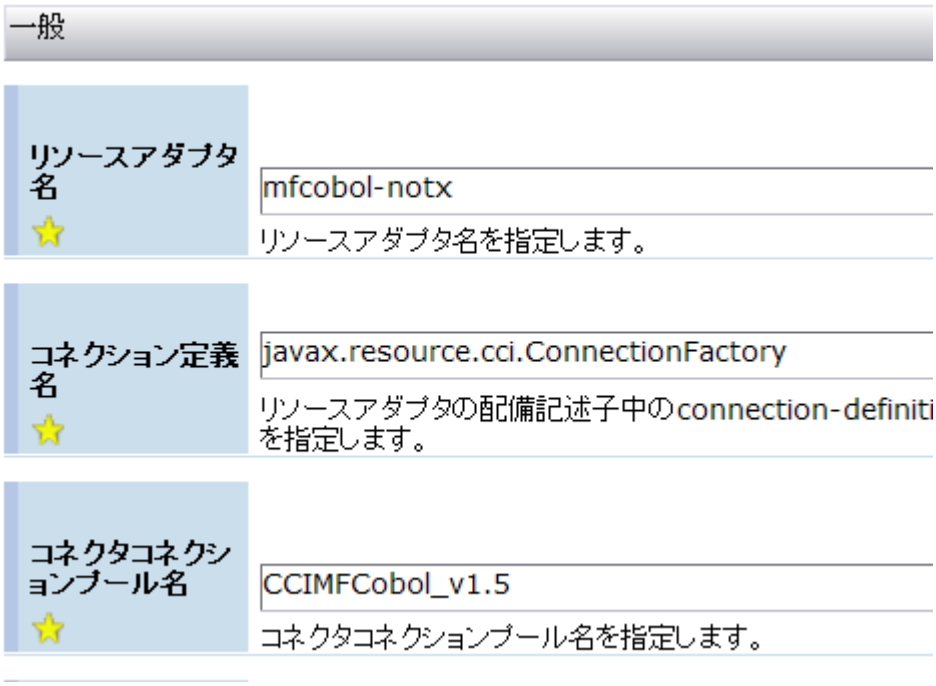
4. 統合運用管理コンソールのツリービューの [アプリケーション] の下に以下のように mfcobol-notx が作成されていることを確認します。これをクリックすると右ペインに以下のように配備されたリソースアダプタの情報が表示されます。



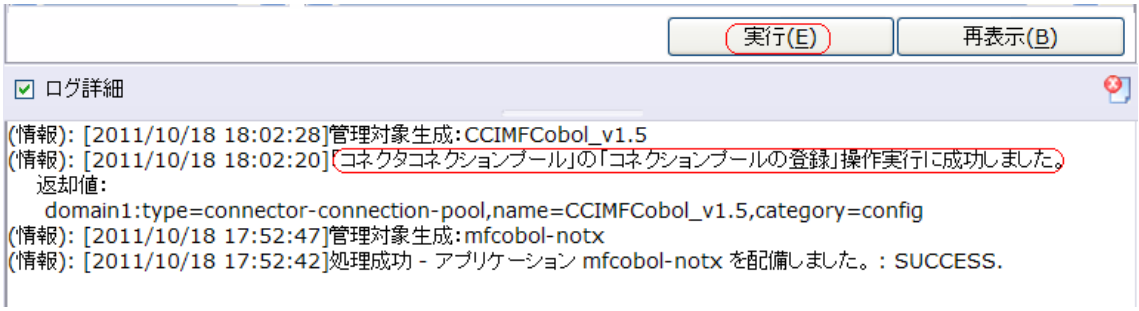
5. 配備されたリソースアダプタのコネクションプールを登録します。統合運用管理コンソールのツリービューで [リソース] を展開し、[コネクションプール] を選択します。右ペインに現れる [操作リスト] の中から [コネクションプールの登録] をクリックします。



6. 以下の通りに登録するコネクションプールの情報を入力します。

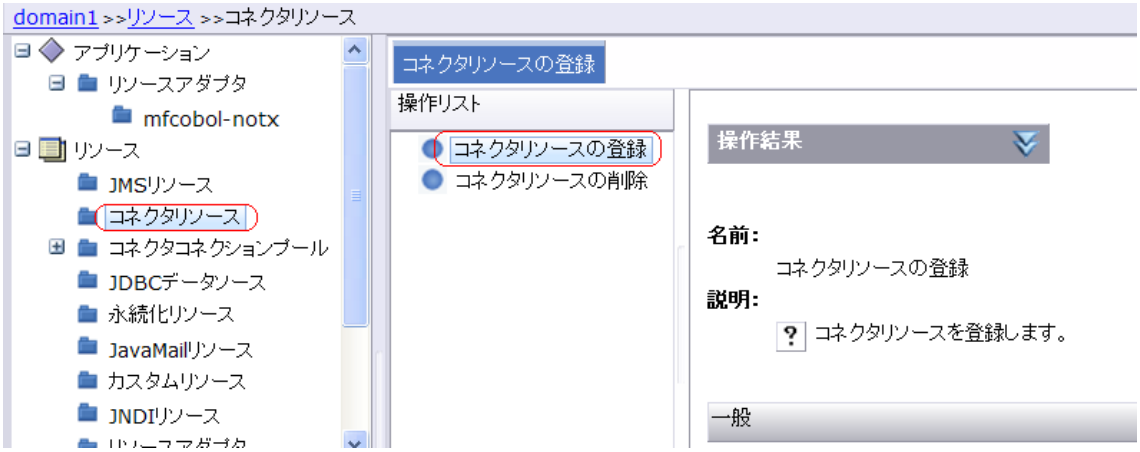


7. [実行] ボタンをクリックし、以下のように成功のログが表示されることを確認します。

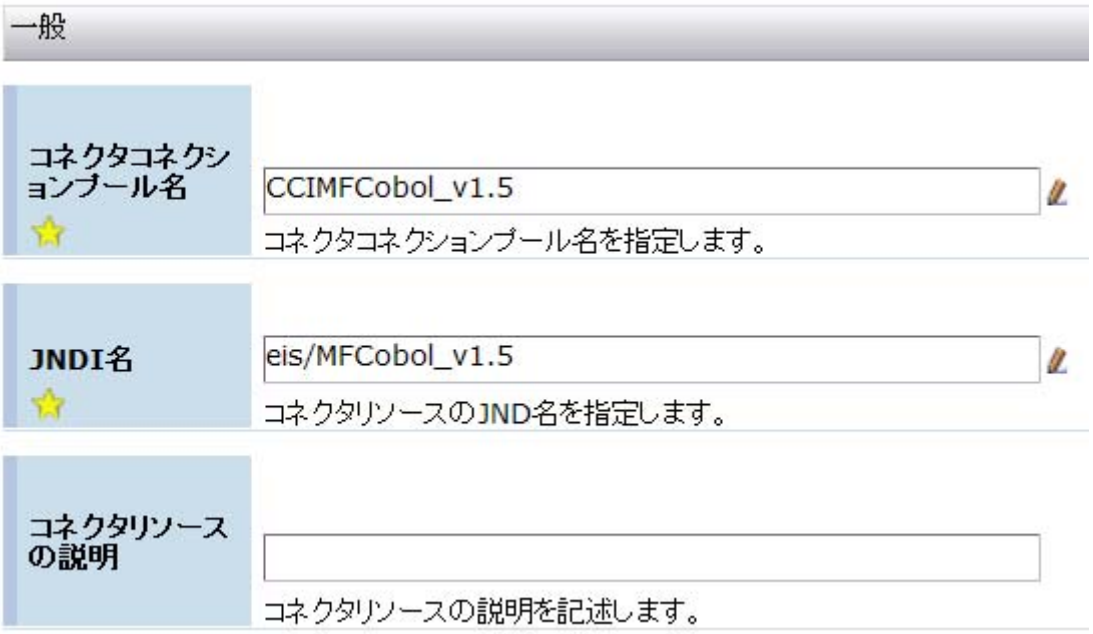


8. 統合運用管理コンソールのツリービューの [コネクションプール] の下に以下のように CCIMFCobol_v1.5 が作成されていることを確認します。これをクリックすると右ペインに以下のように登録されたコネクションプールの情報が表示されます。

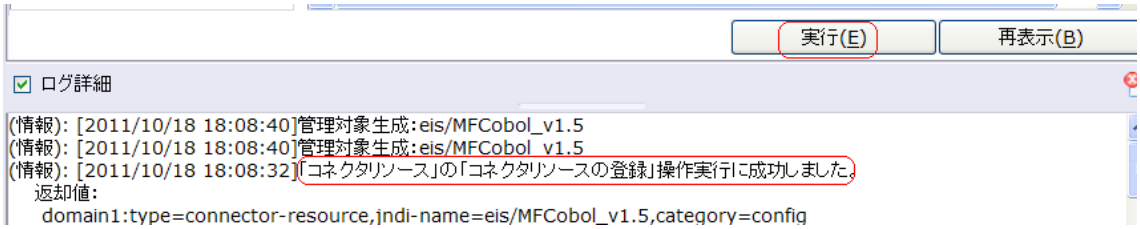
9. 最後にコネクタリソースを登録します。管理コンソールのツリービューで [リソース] > [コネクタリソース] を選択します。右ペインに現れる [操作リスト] の中から [コネクタリソースの登録] をクリックします。



10. 以下の通りに登録するコネクションプールの情報を入力します。



11. [実行] ボタンをクリックし、以下のように成功のログが表示されることを確認します。



12. 統合運用管理コンソールのツリービューの [コネクションプール] の下に以下のように CCIMFCobol_v1.5 が作成されていることを確認します。これをクリックすると右ペインに以下のように登録されたコネクションプールの情報が表示されます。

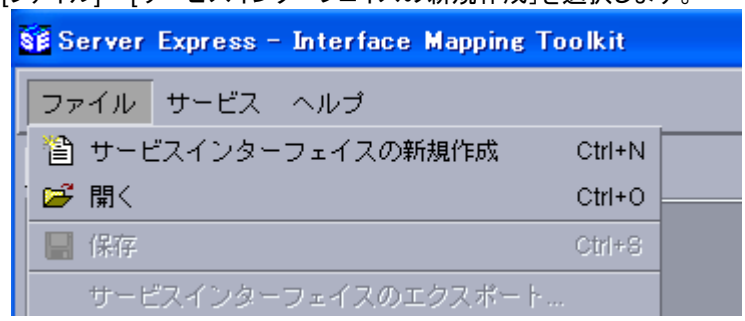
以上で、リソースアダプタの配備は完了しました。

3. COBOLプログラムの準備

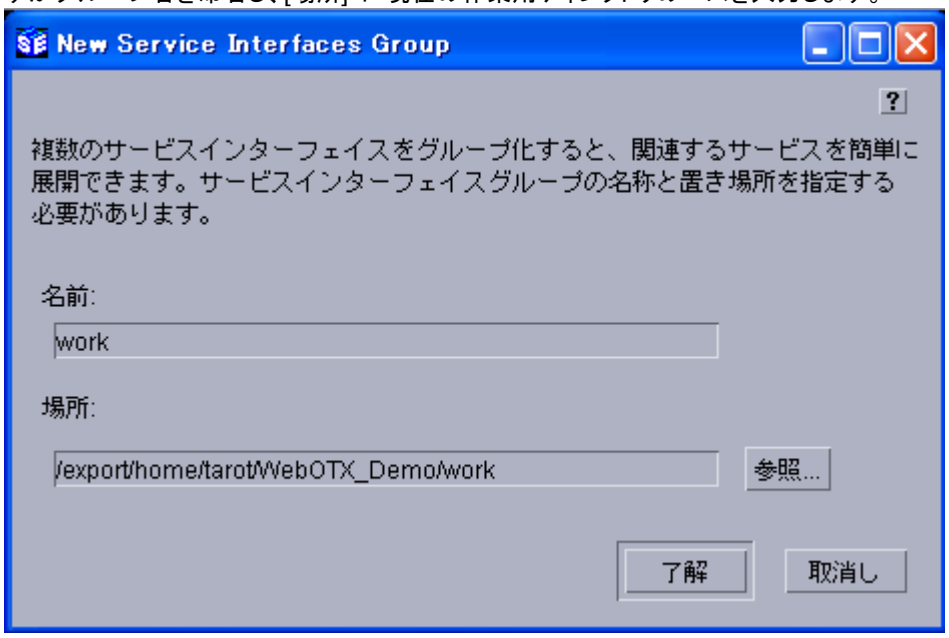
ここでは、検証で使用する COBOL プログラムを Enterprise Server に配備し、 同時に EJB ラッパーを自動生成します。

1. 作業用ディレクトリを新規作成し、COBOL 例題プログラム Calcu.cbl をコピーします。
2. Server Express を使用する環境変数を設定します。 検証では ShiftJIS ロケールを使用しましたので LANG 環境変数は ja_JP.PCK に設定されています。
3. Calcu.cbl をコンパイルします。
4.
```
$ cob -ug Calcu.cbl
```

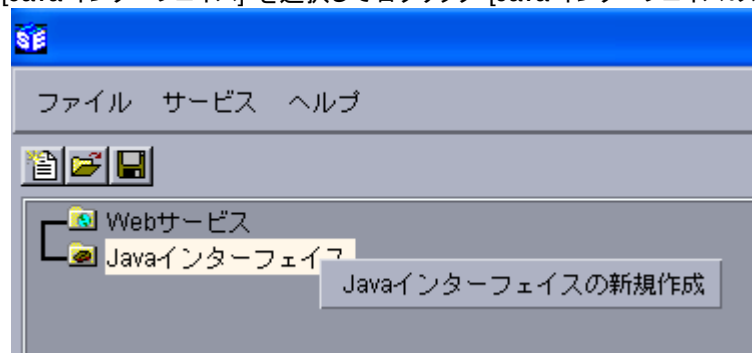
5. X Window をホストする環境を整え、cobimtk コマンド打鍵し、インターフェイスマッピングツールキットを起動します。
6. [ファイル] > [サービスインターフェイスの新規作成]を選択します。



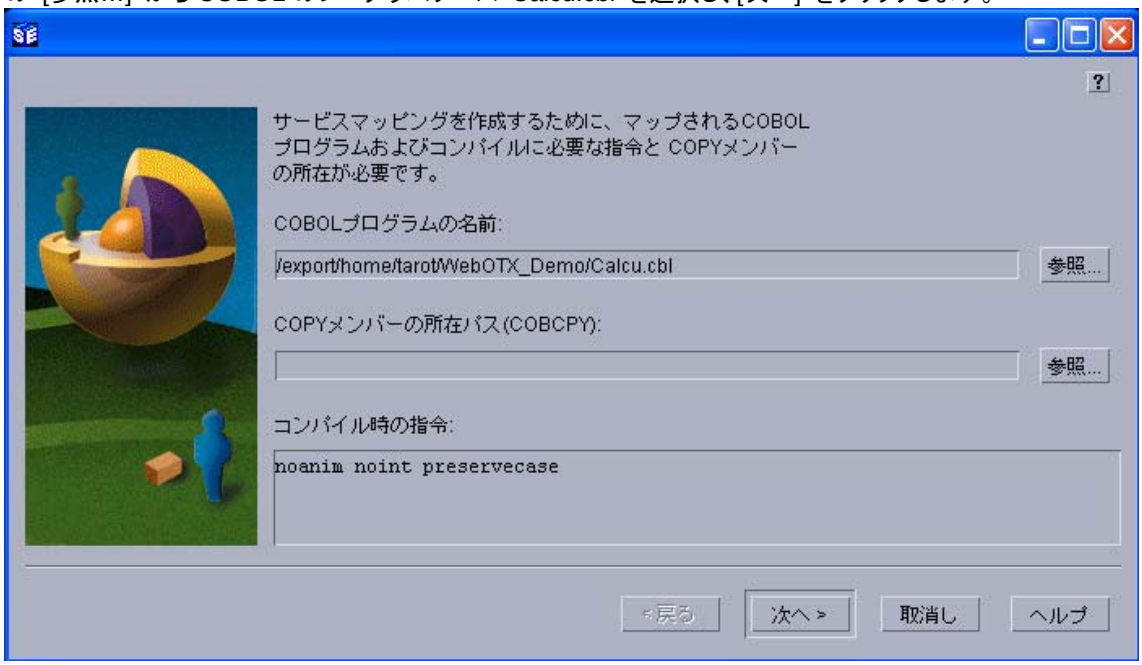
7. サービスインターフェイスグループの新規作成ダイアログが現れます。ここで [名前] には何でも良いですがグループ名を命名し、[場所] に現在の作業用ディレクトリのパスを入力します。



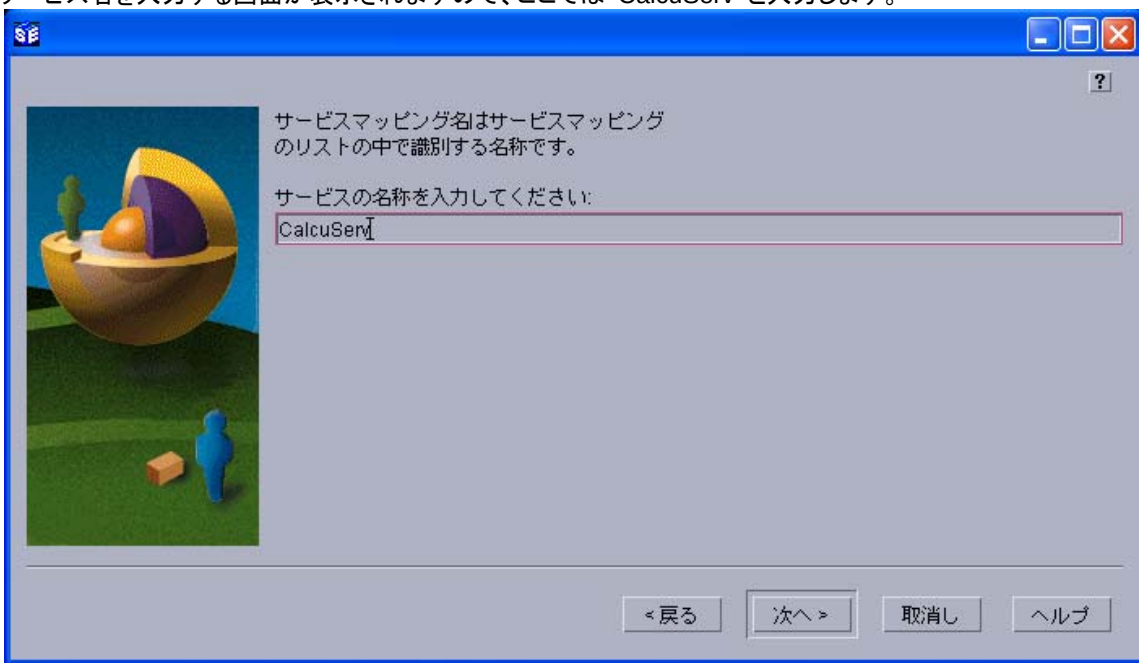
8. [Java インターフェイス] を選択して右クリック [Java インターフェイスの新規作成] を選択します。



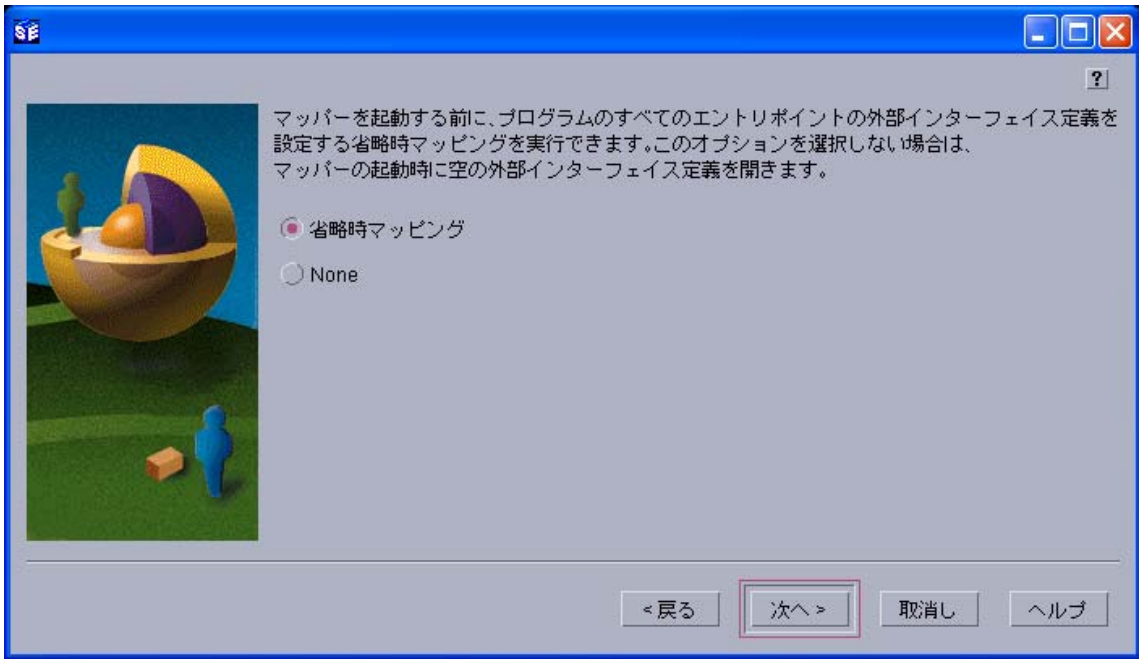
9. マッピングするプログラムを選択する画面が表示されます。 以下のように [COBOL プログラムの名前] の [参照…] から COBOL のプログラムソース Calcu.cbl を選択し、[次へ] をクリックします。



10. サービス名を入力する画面が表示されますので、ここでは CalcuServ と入力します。

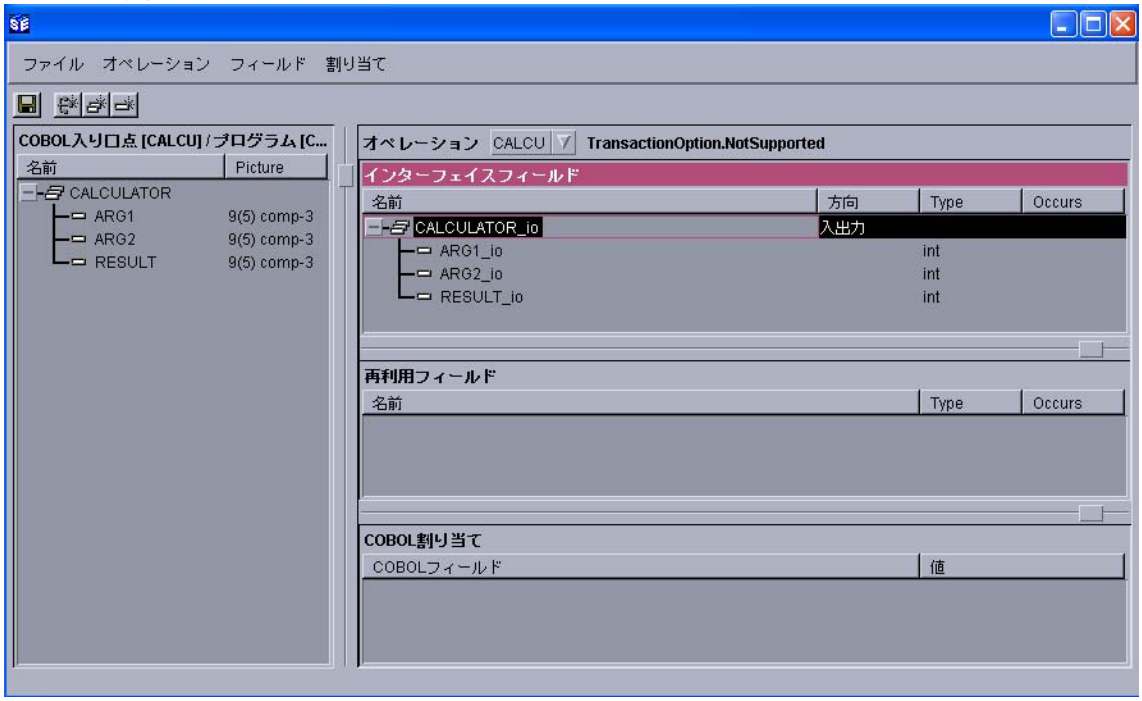


11. 以下のダイアログでは [省略時マッピング] がチェックされたままの状態で [次へ] をクリックします。

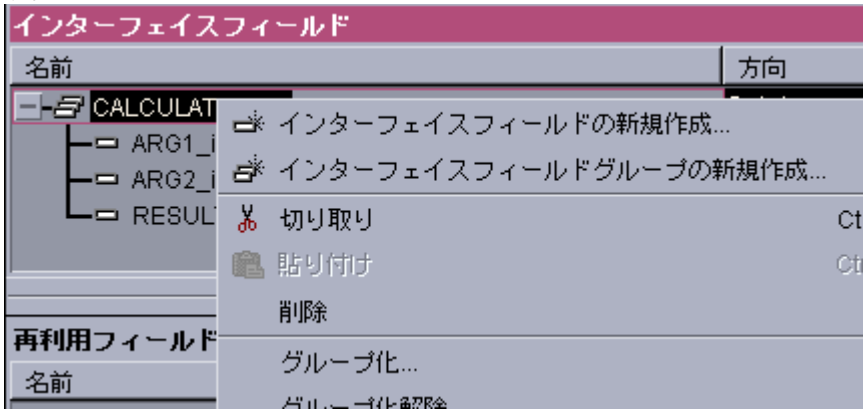


12. 以下のダイアログで [完了] をクリックします。

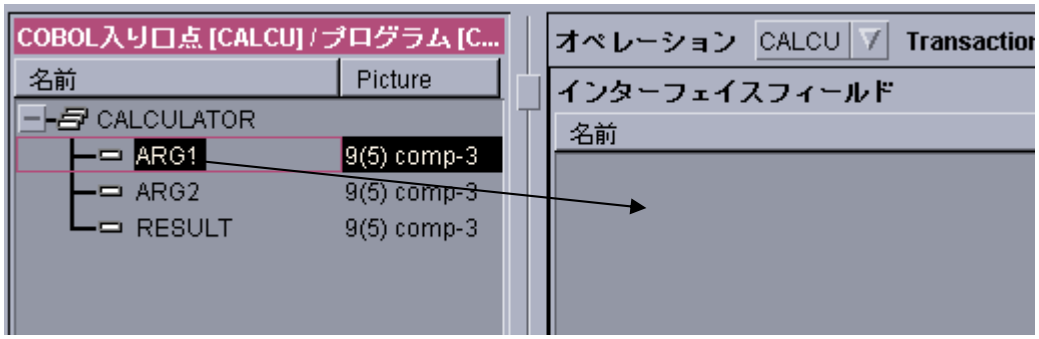
13. マッパーウィンドウが現れます。左ペインには CALCU.cbl の LINKAGE パラメタに書かれた宣言がそのまま表示されています。右ペインにはこれを EJB メソッドとしてマッピングする方法を示しています。省略時マッピングでは以下のように LINKAGE SECTION の集団項目がグループフィールドとしてマップされています。



14. 一旦グループフィールドを削除します。右ペインで CALULATOR_io を右クリックし、[削除] を選択します。

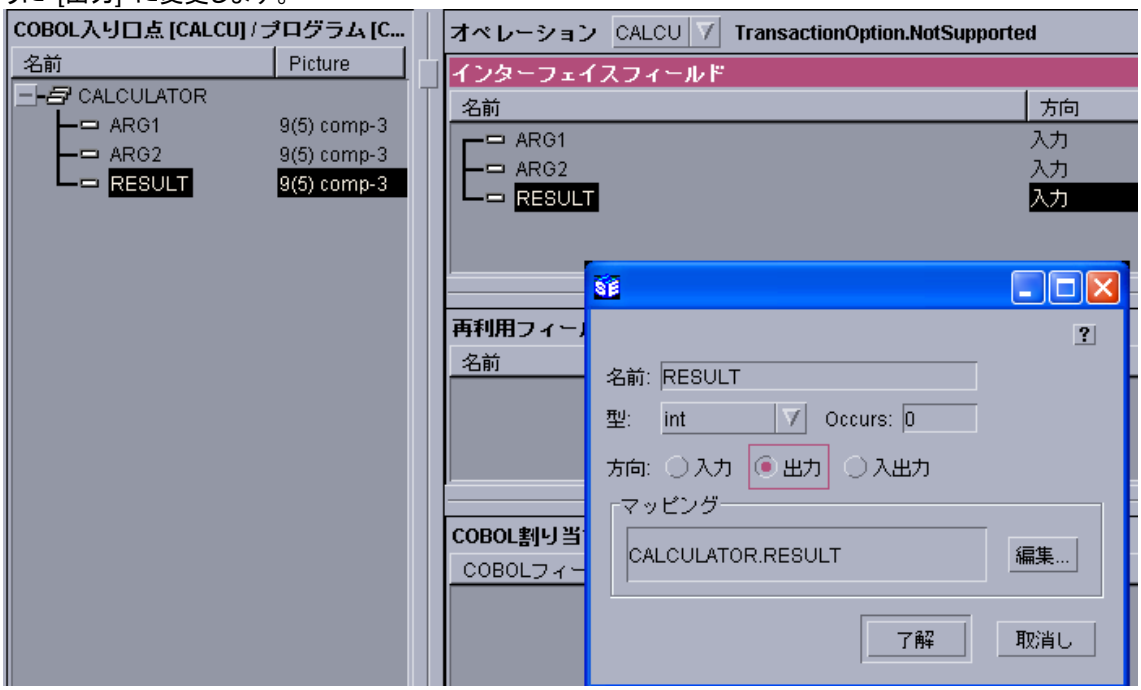


15. 右ペインがクリアされます。ここで左ペインの ARG1 を右ペインにドラッグします。

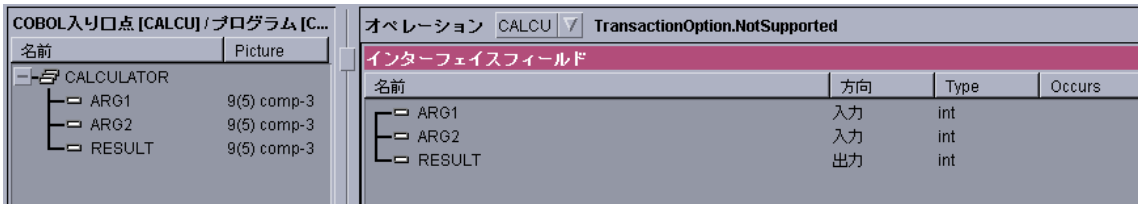


16. 引き続き ARG2 と RESULT も右ペインにドラッグします。デフォルトではすべてのフィールドが入力パラメタとしてマップされています。この例題では最後の RESULT は出力パラメタなので、これを以下のよ

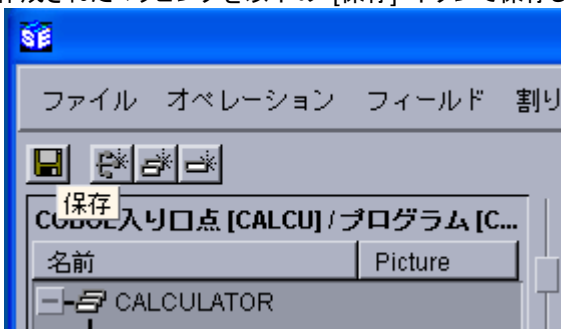
うに [出力] に変更します。



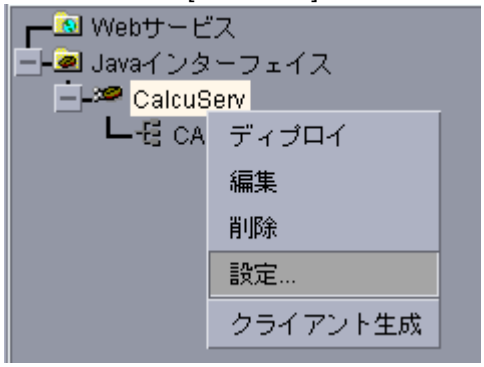
17. マッピングが以下のようになっていることを確認してください。



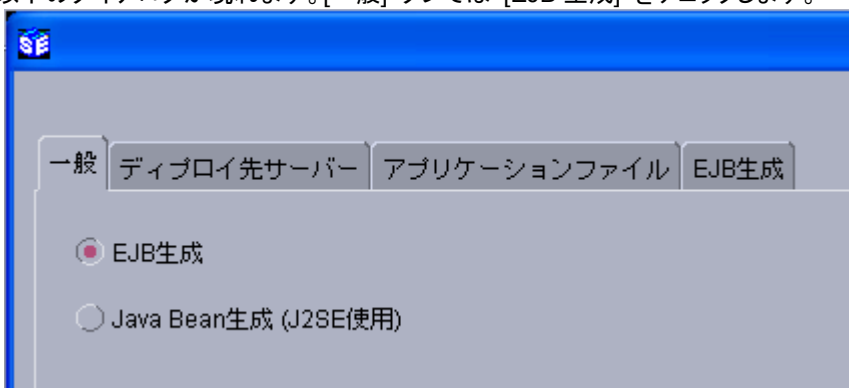
18. 作成されたマッピングを以下の [保存] ボタンで保存しマッパーウィンドウを閉じます。



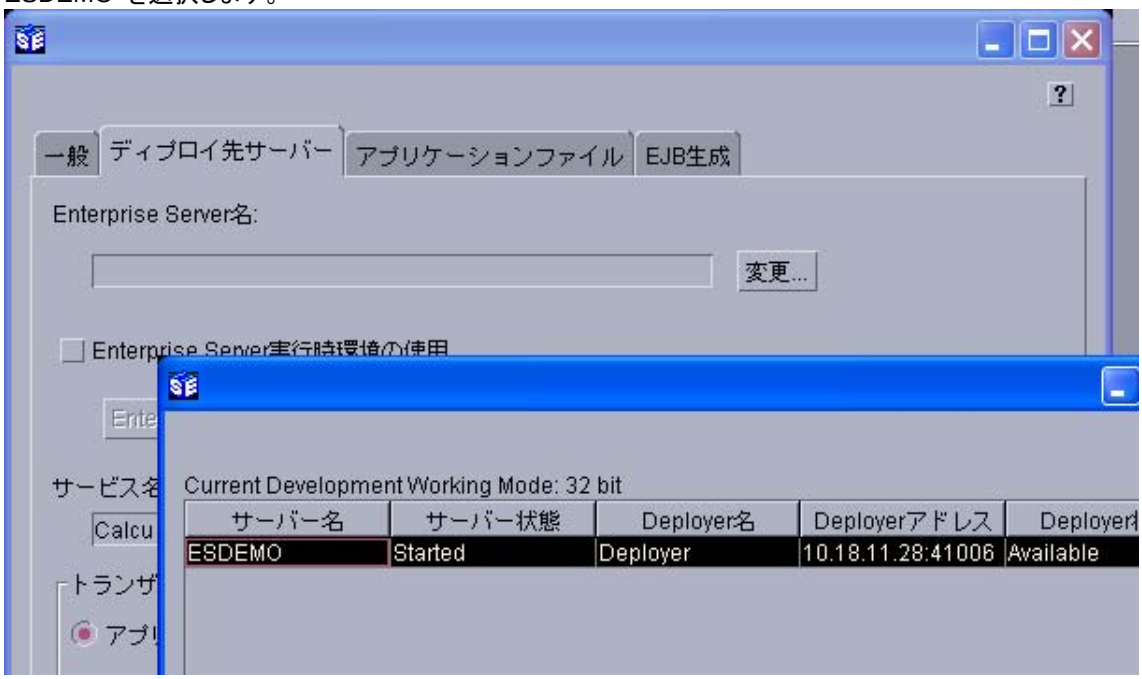
19. ツリービュー上で [CalcuServ] を右クリックし、[設定...] を選択します。



20. 以下のダイアログが現れます。[一般] タブでは [EJB 生成] をチェックします。



21. [ディプロイ先サーバー] タブで、[Enterprise Server 名] の [変更…] ボタンをクリックし、開始している ESDEMO を選択します。



22. [サービス名] に CalcuServ が入っていることを確認してください。 入っていない場合は修正してください。

23. [アプリケーションファイル] タブでは、コンパイル済みの Calcu.gnt、Calcu.idy、Calcu.cbl を追加します。

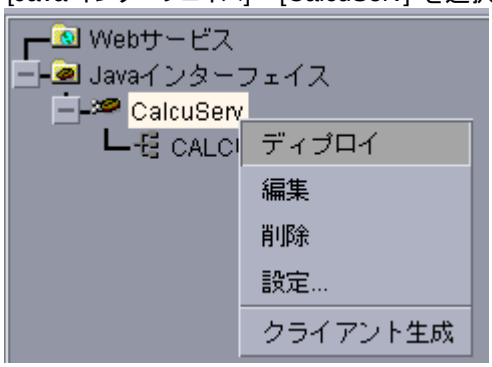
24. [EJB 生成] タブでは以下のように指定します。

[Application Server] は [Java EE 5] と [Weblogic 10.3.4] を選択します。 [Java Compiler] にお使いの Java DK の bin ディレクトリのパス名を入力します。[J2EE Class Path] には WebOTX の

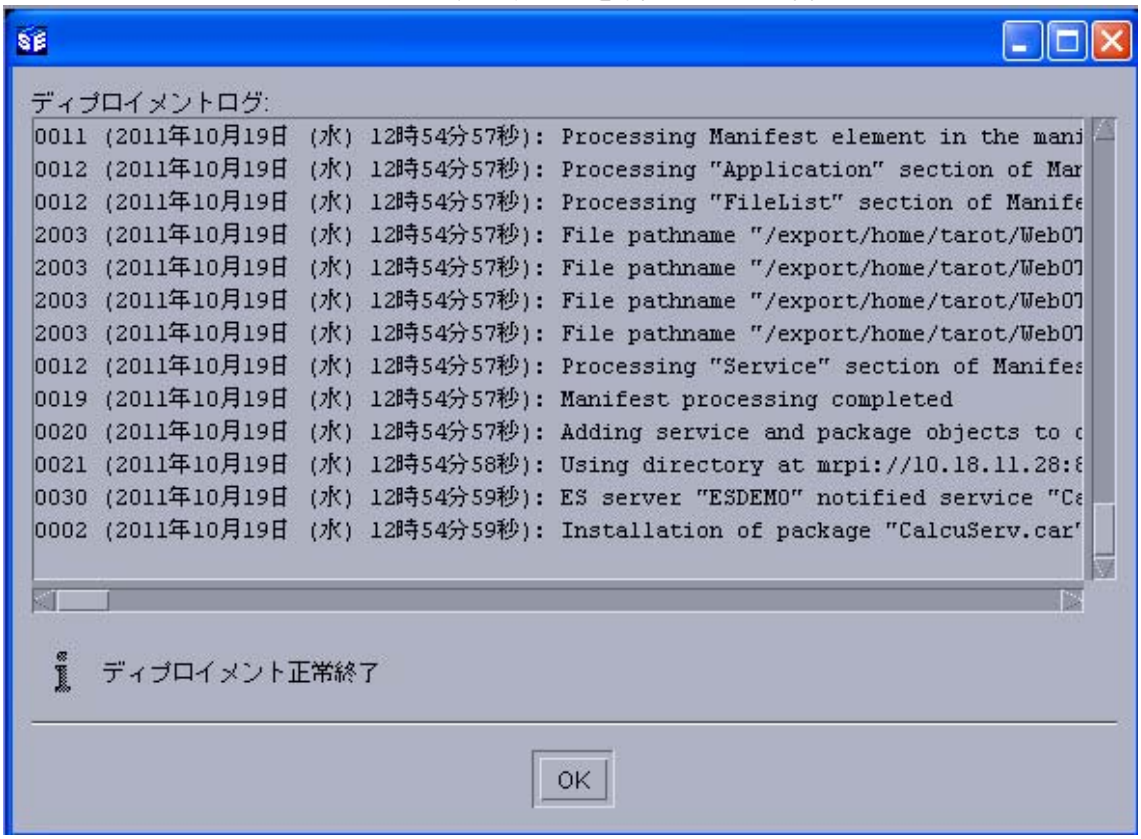
/opt/WebOTX/javaee.jar を入力します。それ以外はディフォルト値のままにします。

25. [了解] をクリックしダイアログを閉じます。

26. [Java インターフェイス] > [CalcuServ] を選択して右クリックして [ディプロイ] を選択します。



27. 以下のダイアログでサービスのディプロイが完了することを確認してください。。



28. これで Enterprise Server 上に COBOL サービスマッピングが配備されました。 Enterprise Server Admin 画面で以下のようにディプロイが完了していることを各 y 人してください。

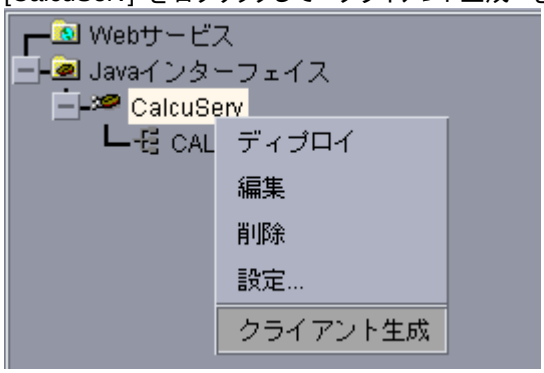
	サービスネームスペース	オペレーション	サービスクラス	探索順序	リスナー	要求ハンドラ	実装パッケージ	現ステータス	ステータスログ	カスタム構成
	Test	Test 編集...		1	1 CP 1 HTTP Echo tcp:10.18.11.28*:9002 (tok-wedge +)			Available	OK	
	Deployer	Deployer 編集...	MF deployment	1	1 CP 1 Web tcp:10.18.11.28*:41006* (tok-wedge +)			Available	OK	[MF client] scheme=h accept=application/x- listener=Web Service
	JES	JES 編集...	MF JES	1	1 CP 1 Web Services and J2EE tcp:10.18.11.28*:9003 (tok-wedge +)			Available	OK	
	CICS	CICS 編集...	MF CICS	1	1 CP 1 Web Services and J2EE tcp:10.18.11.28*:9003 (tok-wedge +)			Available	OK	
	ES	ES 編集...	MF ES	1	1 CP 1 Web Services and J2EE tcp:10.18.11.28*:9003 (tok-wedge +)			Available	OK	
削除...	CalcuServ	1 of 1 operations shown								
		.CALCU 編集...		1	1 CP 1 Web Services and J2EE tcp:10.18.11.28*:9003 (tok-wedge +)	MFRHBINP	CalcuServ	Available	OK	

29. この時作業用ディレクトリの repos/CalcuServ.deploy に EJB ラッパー CalcuServ.jar が自動生成されています。ソースファイルとともに生成されていますので確認してください。

4. テスト用 Webクライアントの生成

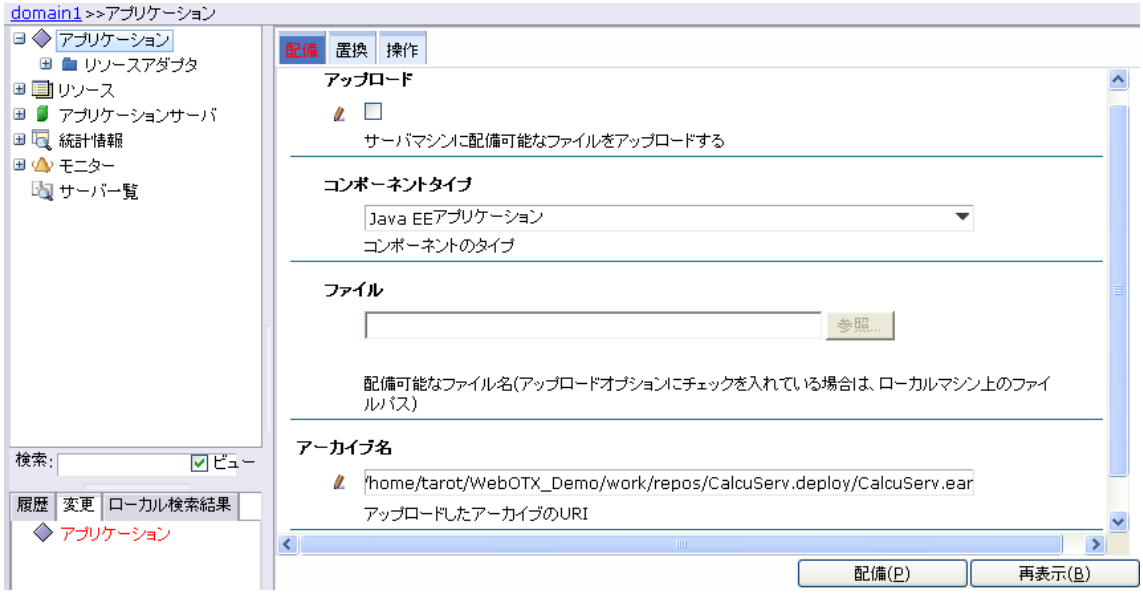
インターフェイスマッピングツールキットのクライアント生成機能を使用すると、 対話型でパラメタの値を受け取り、EJB のメソッドを呼び出して結果を表示するような、 簡単な Servlet モジュールを含む、.ear パッケージを作成できます。 ここでは、これを使用して WebOTX 上の Web クライアントからの呼び出しを行います。

1. COBOL サービスを配備したインターフェイスマッピングツールキットに戻り、 [Java インターフェイス] > [CalcuServ] を右クリックして "クライアント生成" を選択します。

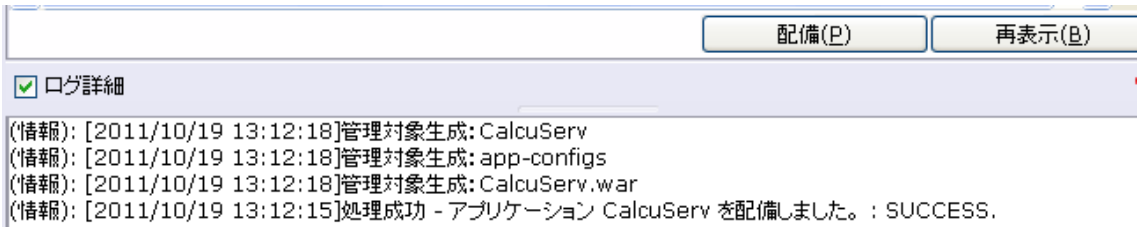


2. この時作業用ディレクトリの repos/CalcuServ.deploy に パッケージ CalcuServ.ear が自動生成されています。ソースファイルとともに生成されていますので確認してください。

3. WebOTX 統合運用管理コンソールに戻り、左ペインのツリービューで [アプリケーション] を選択します。右ペインで以下のように、[アップロード] のチェックをオフにし、[コンポーネントタイプ] のプルダウンで “Java EE アプリケーション” を選択し、[アーカイブ名] に上記で生成された CulcuServ.ear のパス名を入力します。

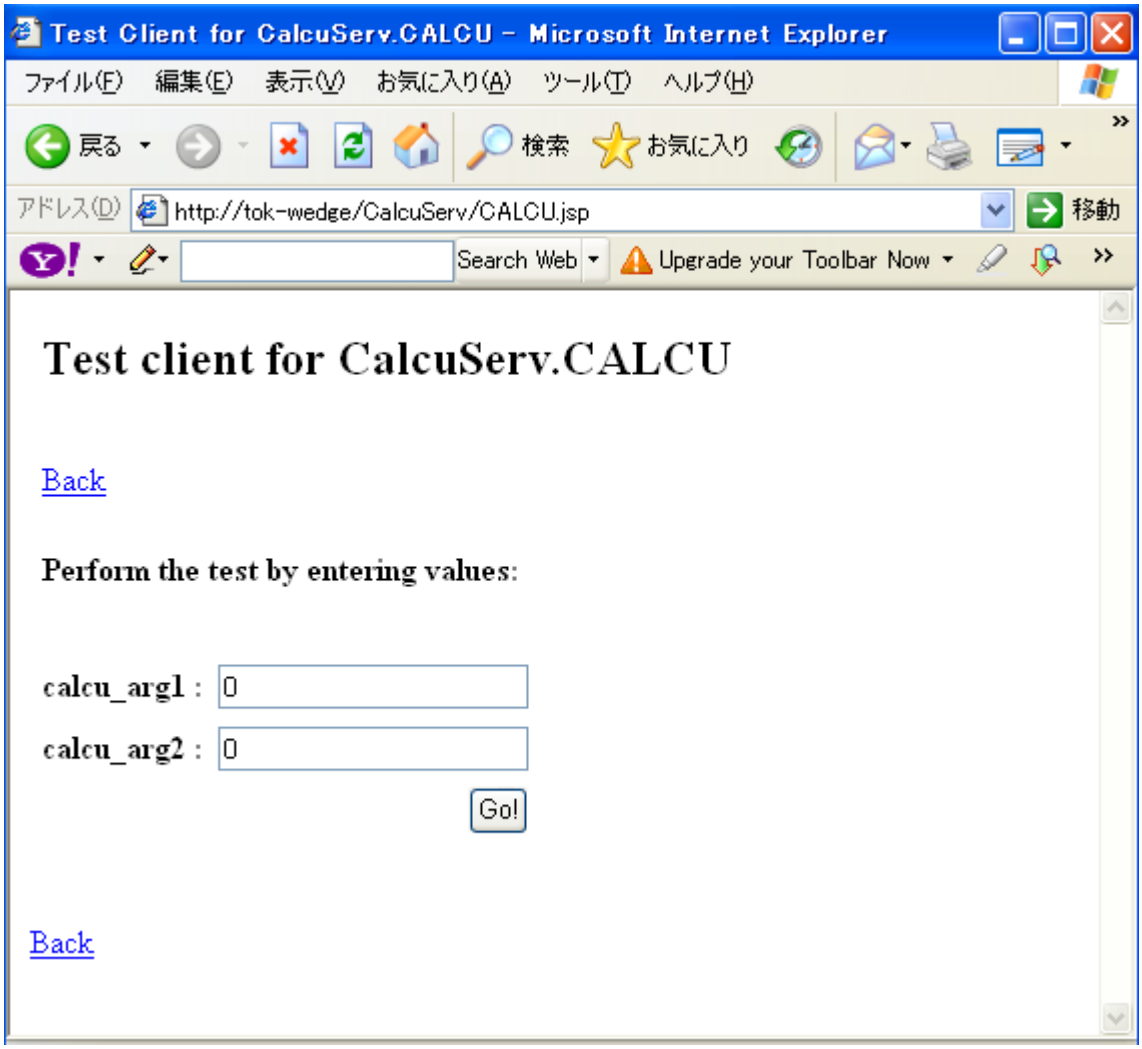


4. [配備] ボタンをクリックし、以下のように配備が成功したログ表示を確認します。



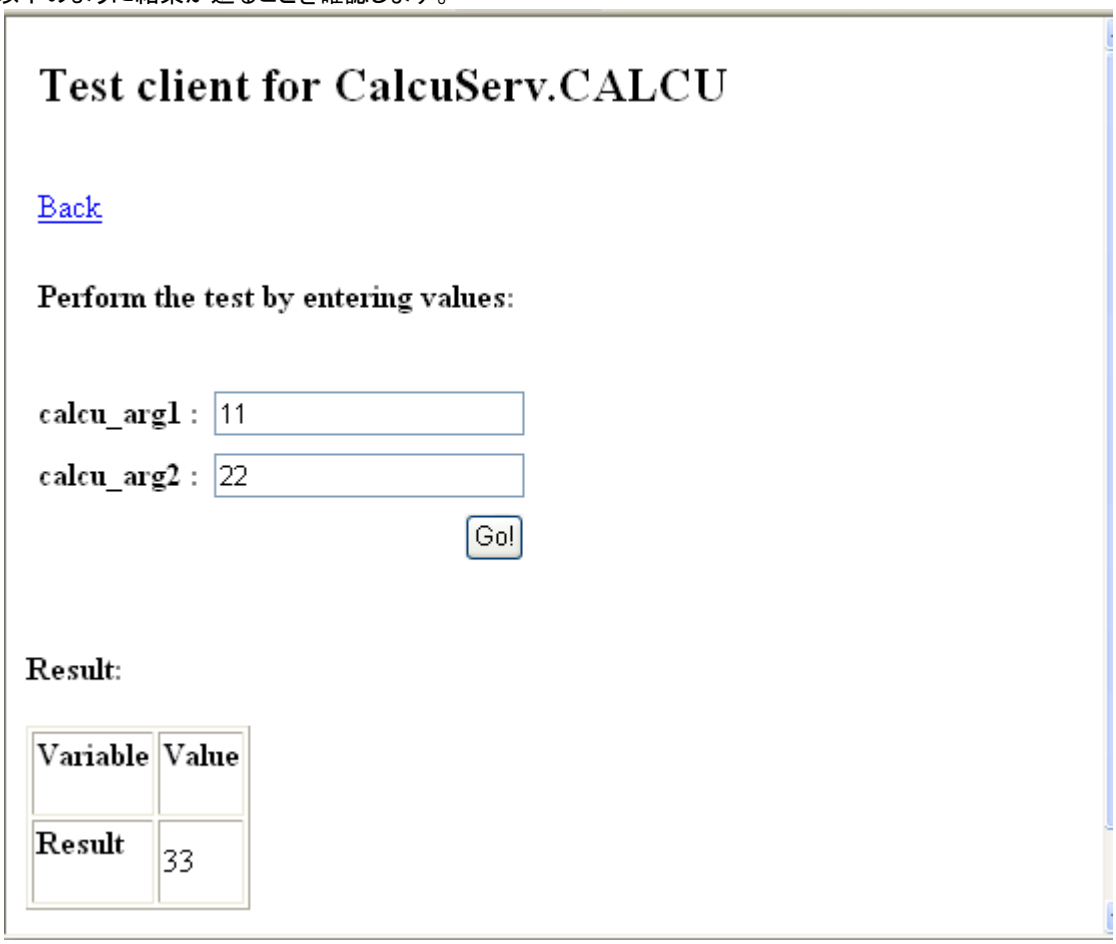
5. 以上で、EJB と Web アプリケーションが同時に配備されました。

6. Web ブラウザを開き、"http://<Solaris サーバーアドレス>:80/CalsuServ/CALCU.jsp" を開きます。



7. 入力フィールドに適宜数値を入力し、[Go!] をクリックします。

8. 以下のように結果が返ることを確認します。



以上