



# Micro Focus Visual COBOL 3.0J Micro Focus Enterprise Developer 3.0J

## 新機能・強化機能概要

マイクロフォーカス株式会社



# 主な新機能、強化機能一覧

## 対応プラットフォーム・対応データベースの拡大 開発支援機能の強化

- 対応 IDE の追加
- COBOL 開発機能の強化
- JSON API
- パフォーマンスの強化

## 日本国内ユーザー向け特別機能の追加

- COBUTF8 – UTF8 ロケールのまま SJIS 資産を扱う機能
- CBL\_SCR\_SET\_KEISEN\_COLOR ルーチン

## エンタープライズ DevOps 支援機能

- Undo デバッグ
- MFUnit – COBOL 専用の単体テストフレームワーク
- Test Coverage 機能の強化

## Enterprise Developer 固有の強化

- PL/I 言語 (デバッガーの刷新、エディターの強化)
- JES – JCLスプール機能の強化
- CICS – Web Service 化機能の強化
- iFileshare – 高可用性支援機能の強化

# 対応プラットフォーム・ 対応データベースの拡大

# 対応プラットフォーム・対応データベースの拡大

## ▶ OS



これまでサポートしていた OS、ミドルウェアに加え本リリースで新規に追加されたプラットフォーム

## ▶ データベース

- ✓ Oracle 12.2(COBSQL)
- ✓ DB2 LUW 11.1(DB2 ECM, OpenESQL)
- ✓ Microsoft SQL Server 2016
- ✓ PostgreSQL 9.6
- ✓ MySQL 5.7(OpenESQL – ODBC のみ)

ORACLE  
DATABASE

IBM DB2

Microsoft  
SQL Server  
2016



MySQL™

# 開発支援機能の強化

# 対応 IDE の追加

## ▶ Eclipse 4.6 (64 bit)



- ✓ Eclipse 4.6 (64 bit) がインストーラーにバンドルされ、64 bit 版 JRE で Eclipse IDE の操作が可能となります。
- ✓ プラグラインインストールで、Eclipse 4.4.1, 4.4.2, 4.5, 4.6 (32 bit) も利用可能です。

## ▶ Microsoft Visual Studio 2017



- ✓ Microsoft 社が提供する執筆時における最新の Visual Studio IDE で、NET Framework 4.6.2、4.7 ターゲットの COBOL アプリケーション開発が可能となります。

# COBOL 開発機能強化 – Eclipse

## ▶ リファクタリング

- ✓ Native、COBOL for JVM とともにサポート
- ✓ COBOL for JVM で同一ワークスペース中の Java と連携している場合は、Java のソースも併せてリファクタリング

COBOL のソース中で demoMethod を選択し、newNameMethod へリファクタリング

Java のソースも併せてリファクタリングされます。

```
DEMOCOBPGM.cbl
MOVE 0 TO COB-BIN-PARAM.
DECLARE JREF AS TYPE com.ym.demo.JSubPgm = new com.ym.demo.JSubPgm().
SET AAA TO JREF:demoMethod(123);
COMPUTE COB-BIN-PARAM = COB-PCK-PARAM + COB-ZNE-PARAM + AAA.
GOBACK.
END PROGRAM DEMOCOCPGM.
```

```
JSubPgm.java
package com.ym.demo;
public class JSubPgm {
    public int demoMethod(int AAA){
        int dummyParam = AAA + 1;
        return dummyParam;
    }
}
```

メソッドの名前変更

新しい名前(M): newNameMethod

参照の更新(R)

名前変更されたメソッドに対する委譲としてオリジナル・メソッドを保持(K)

非推奨としてマーク(D)

プレビュー >(W)

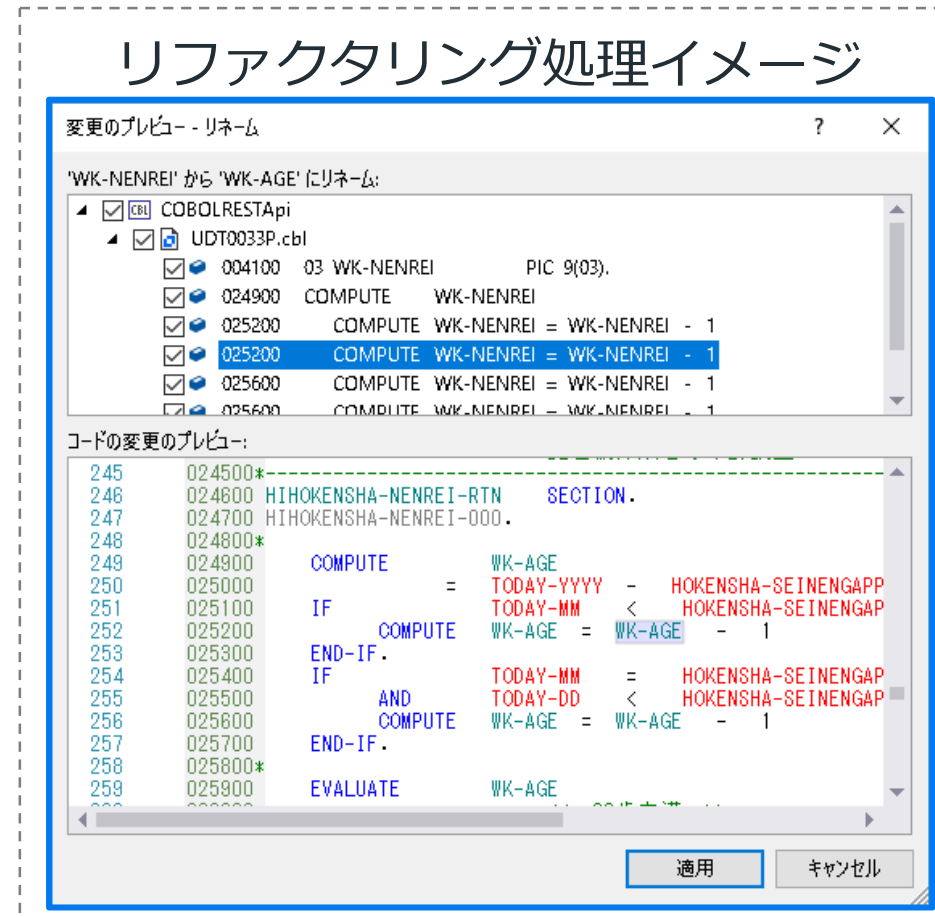
```
DEMOCOBPGM.cbl
MOVE 0 TO COB-BIN-PARAM.
DECLARE JREF AS TYPE com.ym.demo.JSubPgm = new com.ym.demo.JSubPgm().
SET AAA TO JREF:newNameMethod(123);
COMPUTE COB-BIN-PARAM = COB-PCK-PARAM + COB-ZNE-PARAM + AAA.
GOBACK.
END PROGRAM DEMOCOCPGM.
```

```
JSubPgm.java
package com.ym.demo;
public class JSubPgm {
    public int newNameMethod(int AAA){
        int dummyParam = AAA + 1;
        return dummyParam;
    }
}
```

- ▶ Code Analysis ルールに 64 bit Readiness を追加
- ▶ ビルドパスにコピーファイルが格納されたプロジェクト外フォルダを追加
- ▶ 複数プロセスによる並列コンパイル

# COBOL 開発機能の強化 – Visual Studio

- ▶ Dialogue System の統合 → Addpack 不要
- ▶ COBOL editor
  - ✓ リネームリファクタリング
  - ✓ Smart indent
  - ✓ END-IF, END-EVALUATE の自動挿入
  - ✓ 括弧の自動挿入(VS2015+)
  - ✓ アウトライン機能の強化
  - ✓ キーワードのサジェスト表示





# JSON API

COBOL のデータ構造を JSON 形式へ変換する COBOL 構文 **JSON GENERATE** 文が追加され、COBOL から作成された RESTful Web サービスを呼び出すための JSON データの作成が容易になります。

## コード例 :

```
ENVIRONMENT DIVISION.  
INPUT-OUTPUT SECTION.  
    SELECT AFFILE ASSIGN TO "SimpleDemo.json"  
    ORGANIZATION IS LINE SEQUENTIAL.  
DATA DIVISION.  
FILE SECTION.  
FD AFFILE.  
01 AFFILE-REC          PIC X(800).  
WORKING-STORAGE SECTION.  
01 DEMO-JSON-DATA.  
    03 NUME-VARS.  
        05 PACKED-DEC-VAR      PIC S9(4) VALUE -123.  
        05 BIN-DEC-VAR         PIC S9(4) VALUE 987.  
        05 ZONED-DEC-VAR       PIC S9(4) VALUE -456.  
        03 ALPHA-NUME-VAR      PIC X(10) VALUE ALL "A".  
PROCEDURE DIVISION.  
OPEN OUTPUT AFFILE.  
JSON GENERATE AFFILE-REC FROM DEMO-JSON-DATA.  
WRITE AFFILE-REC.  
CLOSE AFFILE.  
GOBACK.
```

```
{  
  "DEMO-JSON-DATA":{  
    "NUME-VARS":{  
      "PACKED-DEC-VAR":-123,  
      "BIN-DEC-VAR":987,  
      "ZONED-DEC-VAR":-456  
    },  
    "ALPHA-NUME-VAR":"AAAAAAAAAA"  
  }  
}
```

## シンタックス :

(Ent) (MF)

### The JSON GENERATE Statement

The JSON GENERATE statement converts data to JSON format.

#### General Format

```
JSON GENERATE identifier-1 FROM identifier-2  
[ COUNT IN identifier-3 ]  
[ NAME OF { identifier-4 IS literal-1 }-... ]  
[ SUPPRESS { identifier-5 }-... ]  
[ ON EXCEPTION imperative-statement-1 ]  
[ NOT ON EXCEPTION imperative-statement-2 ]  
[ END-JSON ]
```

# パフォーマンスの強化

- ▶ Native、COBOL for JVM、.NET Managed  
いずれのランタイムに対しても最適化処理  
を改善
- ▶ 自社検証の結果、旧バージョン、旧製品  
(Net Express、Server Express)と比較して  
顕著な高速化を確認
  - ✓旧バージョン 2.3 U2 と比べて平均 30 % 以上高速化
  - ✓旧製品よりも全検証パターンにおいて高速化
- ▶ OpenESQL – ADO.NET における最適化  
処理の改善



# 日本国内ユーザー向け 特別機能の追加

# COBUTF8

ソースコードのエンコーディングやデータファイルのエンコーディングとロケールは一致させる必要があります。しかし、近年 Red Hat Enterprise Linux のように SJIS ロケールをサポートしない環境も確認されるようになりました。このようなマーケット事情に応じて弊社では Red Hat Enterprise Linux における標準ロケール UTF8 下でこれまで SJIS 環境でメンテナンスしてきた COBOL プログラム資産並びにデータファイルを扱う機能 **COBUTF8** を追加しました。

## 利用イメージ

### ①構成ファイルを準備

構成ファイルイメージ：

```
$ cat $COBDIR/etc/cobutf8.cfg↓
# Additional mappings can be added using the following format:↓
# <LOCALE> [LOCALE2 ...] <ICNV>#n↓
# Key: ...<...> Mandatory↓
# ...[...> Optional↓
# Values can be delimited using any non-line breaking whitespace.↓
# <ICNV> must be followed by a newline.↓
ja_JP.sjis.....ja_JP.SJIS.....SJIS-OPEN.....#Shift-JIS↓
ja_JP.ujis.....ja_JP.UJIS.....UJIS.....#UJIS↓
ja_JP.eucJP.....ja_JP.eucjp.....EUCJP-OPEN.....#EUC Japanese↓
```

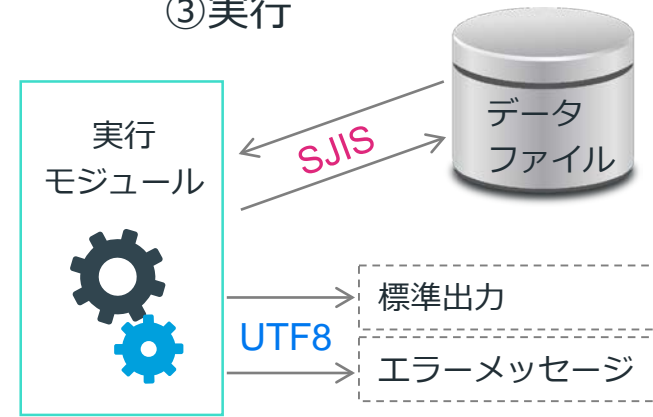
SJIS のみならず EUC 等にも対応

### ②コンパイル

<SJIS でエンコードされたソース>

- ・文字定数
- ・変数名
- ・節・段落名
- ：

### ③実行



ログは他システム同様

UTF8 で記録可



# ライブラリルーチン 「CBL\_SCR\_SET\_KEISEN\_COLOR」

罫線色と文字色でそれぞれ異なる色の指定が可能

サンプルイメージ：

Micro Focus が提供する SCREEN SECTION による CUI 機能では、  
FOREGROUND-COLOR 句で指定した色が罫線及び文字へ反映されます。しかし、国内の ACUCOBOL ユーザーやオフコンユーザーは同じ行・列位置においても罫線と文字とで異なる画面をデザインされることが多いようです。このようなマーケット事情を勘案し本リリースよりこのようなデザインを可能とする機能を追加しました。

runw.exe - COBOL テキストウィンドウ

☆☆☆ 自家用普通自動車年間保険料試算 ☆☆☆

お車の初度登録年月日: 2012 年 05 月 17 日  
お車の登録番号の種類: 1  
(1 - 普通乗用車, 2 - 小型乗用車, 3 - 小型貨物車  
4 - 軽乗用車, 5 - 軽貨物車)  
主な使用目的: K  
(K - 主に家庭用, G - 主に業務用)  
契約距離区分: 4 (今後一年間のお車の最大走行距離)  
(1 - 3000Km以下, 2 - 5000Km以下, 3 - 7000Km以下,  
4 - 9000Km以下, 5 - 11000Km以下, 6 - 無制限)  
記名保険者の氏名: マイクロ 太郎  
記名保険者の住所:  
〒123 - 4567 都道府県 東京都 市区町村 港区  
番地以下 六本木7-18-9F  
記名保険者の生年月日: 1978 年 12 月 19 日  
記名被保険者の免許証の色: 1  
(1 - ゴールド, 2 - ブルー, 3 - グリーン)  
運転者の範囲: 2  
(1 - 本人のみ, 2 - 本人と配偶者, 3 - 家族のみ, 4 - 限定なし)

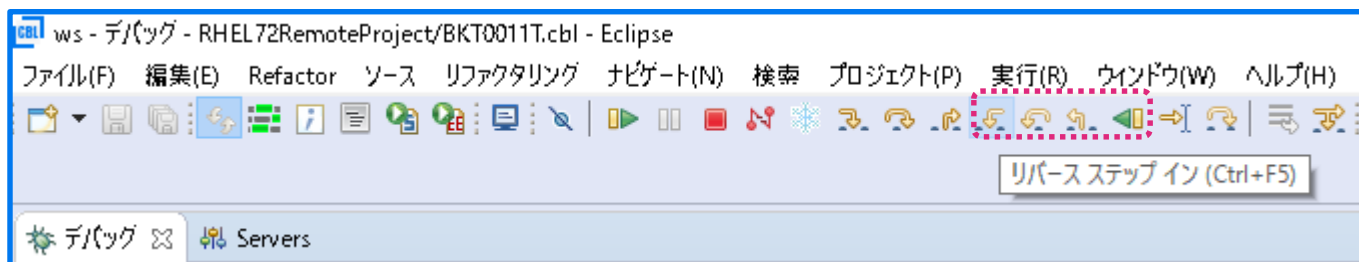
年間保険料試算額: ¥7,654,321 円

# エンタープライズ DevOps 支援機能

# Undoデバッキング

- ▶ UNDO 社の Undo デバッグ技術を製品へ取り込み
- ▶ リモートプロジェクトで利用可能
  - ✓ デバッグ構成で有効・無効の切り替えが可能
  - ✓ リバースステップイン、リバースステップオーバー、リバースを再開（ブレークポイントまで自動リバース）が可能

IDE イメージ：



## ユースケース例

- ✓ MFUnit で自動で流した Test Suite のうち失敗したテストケースのみを Undo デバッグを有効にして自動で再テストする仕組みを CI のシナリオに埋め込む。これにより失敗したテストケースのみを担当者が再生させつつ問題を解析可能。
- ✓ 本番環境でのみ発生する問題を開発環境や調査環境にて再生

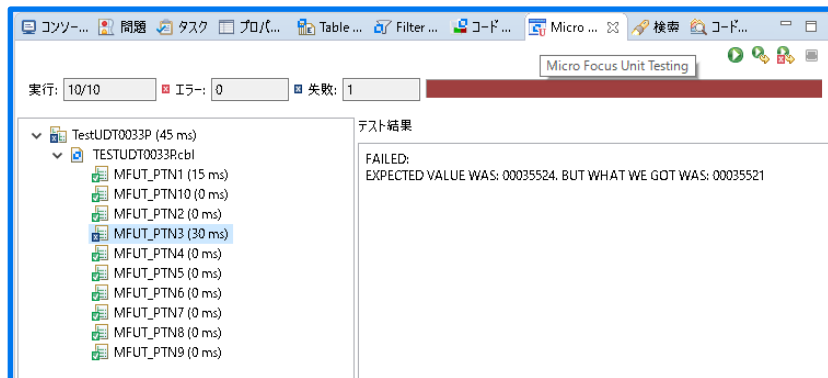


# MFUnit – COBOL 専用の単体テストフレームワーク

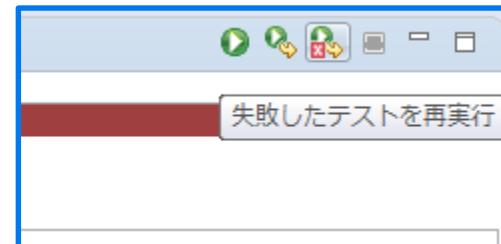
## ▶ Eclipse、Visual Studio IDE へ統合

- ✓ MFUnit 用のプロジェクトを各 IDE へ追加
- ✓ 任意のプログラムからスケルトンを生成可能
- ✓ テスト結果を表示するビューを追加

結果ビューのイメージ：



失敗したテストのみ再実行させる等、  
様々なオプションも用意



## ▶ 様々な CI 関連ツール向けの互換出力オプションを追加

## ▶ EXHIBIT NAMED 文を追加 → 結果ファイルに戻り値等を入力

利用例：

```
if result not equal 42
  exhibit named result
  call MFU-ASSERT-FAIL-Z using by reference z"Result does not equal 42"
end-if
```



# TestCoverage 機能の強化

未実行プログラムも含めた総合的な解析を可能としました。

- ▶ COLLECTION コンパイラ指令による切り替えが可能
- ▶ IDE 上で実行する場合は自動で有効化

出カイメージ：

要素	カバレッジ	カバー済みブロック	未カバーブロック	ブロック全体
LocalInsu	44.6 %	54	67	121
COBAUTH	91.7 %	11	1	12
CUISCRN	85.7 %	6	1	7
UDT0026P	0.0 %	0	47	47
UDT0033P	67.3 %	37	18	55
HIHOKENSHA-NENREI-000	50.0 %	5	5	10
HIHOKENSHA-NENREI-999	100.0 %	1	0	1
HIHOKENSHA-NENREI-RTN	58.3 %	7	5	12
KEYYAKU-KYORI-KUBUN-000	28.6 %	2	5	7
KEYYAKU-KYORI-KUBUN-999	100.0 %	1	0	1
KEYYAKU-KYORI-KUBUN-RTN	44.4 %	4	5	9
MAIN-SYORI-000	100.0 %	7	0	7
MAIN-SYORI-999	100.0 %	1	0	1
MAIN-SYORI-RTN	100.0 %	9	0	9
MENKYOSHOU-IRO-000	50.0 %	2	2	4
MENKYOSHOU-IRO-999	100.0 %	1	0	1
MENKYOSHOU-IRO-RTN	66.7 %	4	2	6
RENEWAL-DISCOUNT-000	60.0 %	3	2	5
RENEWAL-DISCOUNT-999	100.0 %	1	0	1
RENEWAL-DISCOUNT-RTN	71.4 %	5	2	7
SHIYOU-MOKUTEKI-000	66.7 %	2	1	3
SHIYOU-MOKUTEKI-999	100.0 %	1	0	1
SHIYOU-MOKUTEKI-RTN	80.0 %	4	1	5
UNTENSHA-HAN-I-000	40.0 %	2	3	5
UNTENSHA-HAN-I-999	100.0 %	1	0	1
UNTENSHA-HAN-I-RTN	57.1 %	4	3	7

本機能強化により未実行のプログラムについてもカバレッジの分析結果に反映されるようになりました。

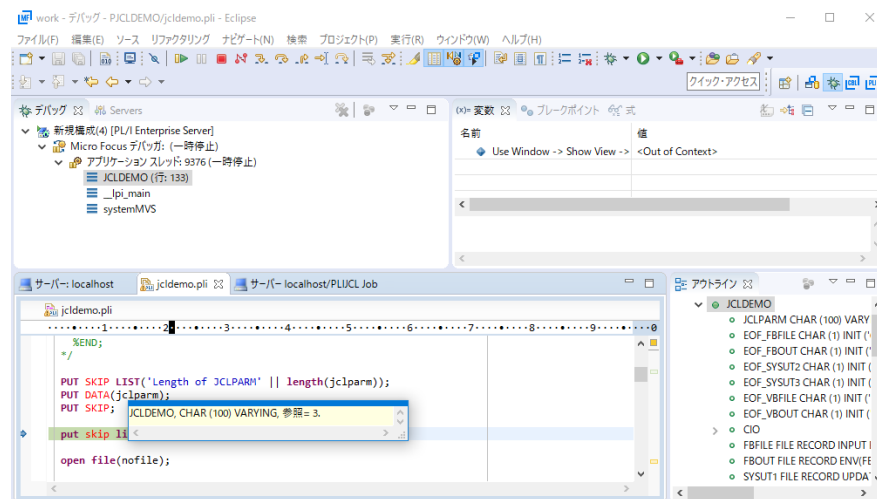
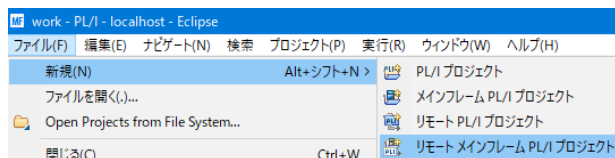
# Enterprise Developer 固有の強化

# PL/I 言語

## ▶ デバッガの刷新

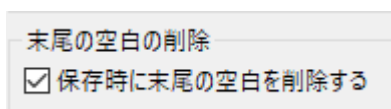
- ✓ Eclipse IDE 上で PL/I ソースのデバッグが可能になりました。引き続き CodeWatch も使用可能です。

- ✓ Eclipse IDE から Linux に存在する PL/I ソースのリモートデバッグが可能になりました。



## ▶ エディターの強化

- ✓ 行末尾の空白を保存時に削除する指定が可能になりました。



# JES – JCL スプール機能の強化

JCL スプールレポジトリが再構築され、パフォーマンスが向上しました。

- ▶ 内部処理のディスク使用効率向上により、スプールファイルが必要とする容量を最大 75 % 削減しました。
- ▶ 内部索引作成処理が改善され、メッセージおよび SYSOUT レコード生成時間が短縮されました。
- ▶ 完了したジョブの出力結果を、任意の組み合わせで一覧に表示するようにスプールキューナビゲーションが改善されました。Complete ステータスに Output、Out Hold、Printed ステータスが含まれます。

JES Output Queue

Refresh Interval (Secs)

Job

Filters: Name: \* User: \* Job No: \* Apply

From Date: YYYY/MM/DD From Time: HH:MM:SS

To Date: YYYY/MM/DD To Time: HH:MM:SS

Queue:  Input  In Hold  Dispatch  Active  Complete

Output  Out Hold  Printed

Display Delete

Name	JobID	C	User	Cond	Submitted
VSAMWRT2	J0001023	A	JESUSER	0000	2017/06/01 14:56:35.59
COPY1	J0001024	A	JESUSER	0000	2017/06/01 14:57:08.71

Job 0001023

J0001023 Name: VSAMWRT2 Status: Complete

Hold Class: A Priority: 00

Update User: JESUSER COND: 0000

Delete File: \$TXRFDIR/t000000014.t

JCLCM0188I J0001023 VSAMWRT2 JOB STARTED 14:56:36

JCLCM0182I J0001023 VSAMWRT2 JOB ENDED - COND CODE 0000 14:56:37

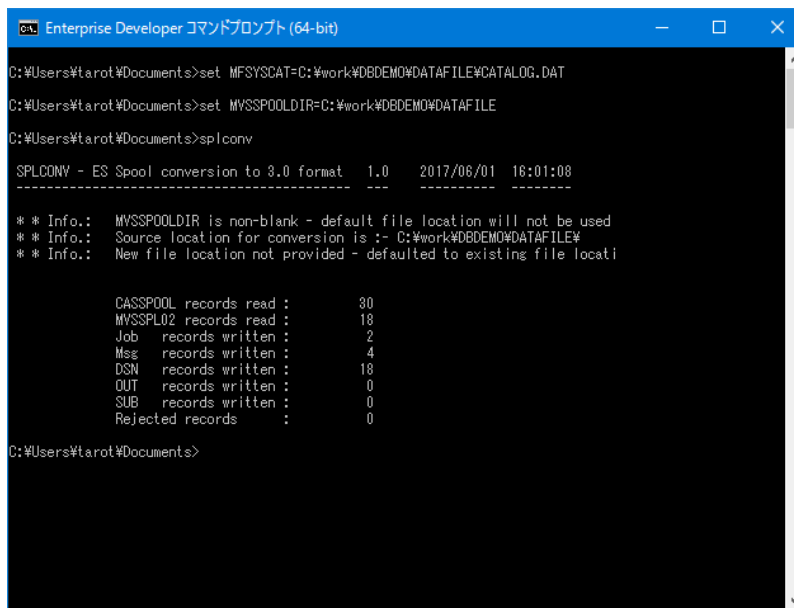
Status	Class	DD Name	Step	Nbr.	Proc Step	Records
Details	Hold	A	JESYSMSG	0		27
Details	Ready	A	SYSRPRINT	DEFVSAM1	1	18

# JES – JCL スプール機能の強化

ジョブ番号を 5 桁から 7 桁へ拡大

**!! 重要 !!**

JES機能を使用する既存のインスタンスの場合、JCL 実行前に各インスタンスに対してスプール変換ユーティリティを実行する必要があります。JES 機能を使用しない既存のインスタンス、新規のインスタンスでは変換の必要はありません。



```
Enterprise Developer コマンドプロンプト (64-bit)
C:\Users\tarot\Documents>set MFSYSCAT=C:\work\DBDEMO\DATAFILE\CATALOG.DAT
C:\Users\tarot\Documents>set MVSSPOOLDIR=C:\work\DBDEMO\DATAFILE
C:\Users\tarot\Documents>splconv

SPLCONV - ES Spool conversion to 3.0 format 1.0 2017/06/01 16:01:08
-----
** Info.: MVSSPOOLDIR is non-blank - default file location will not be used
** Info.: Source location for conversion is :- C:\work\DBDEMO\DATAFILE#
** Info.: New file location not provided - defaulted to existing file locati

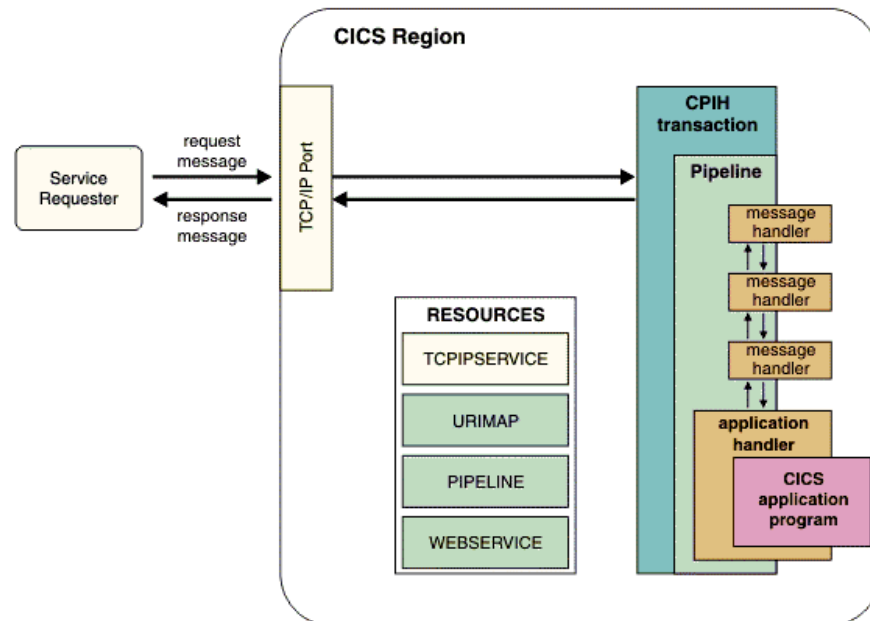
CASSPOOL records read :      30
MVSSPOOL2 records read :     18
Job records written :        2
Msg records written :        4
DSN records written :     18
OUT records written :        0
SUB records written :        0
Rejected records :          0

C:\Users\tarot\Documents>
```

# CICS – Web Service 化機能の強化

Enterprise Developer に実装された Web Services assistant は既存の CICS アプリケーションを Web Service 化し、外部システムが CICS アプリケーションへ Web Service としてアクセスさせることを可能とするバッチユーティリティセットです。昨今の企業アプリケーションは様々なシステムが絡み合いビジネスをサポートします。しかし、各システムへのアクセスメソッドは統一されていないことが多く、これを解決する手段として Web Service は注目を集める技術です。このようなトレンドに合わせて本リリースでは企業のシステム間連携の統一を支援し得る本機能を強化し、一層の使い勝手の向上を図りました。

Web Services assistant の  
動作イメージ：



# iFileshare – 高可用性支援機能の強化

- ▶ プライマリ iFileshare サーバーとスタンバイ iFileshare サーバーを構成し、高可用性グループを形成します。
- ▶ フェールオーバーと回復プロセスが改善されました。
- ▶ Enterprise Server インスタンスに iFileshare コントロールが追加されました。
- ▶ リカバリログ目的で使用されるデータベース参照ファイル（dbase.ref）へワイルドカード指定が可能になりました。

例) fs /d dbase.ref /f data¥\*

“data” ディレクトリ全体を参照対象とします。

The screenshot displays the iFileshare management interface. On the left, a table lists server configurations:

編集...	MFES (MSS)	DEMOCICS	開始
編集...	MFES (MSS)	DEMDFS1	開始
編集...	MFES (MSS)	DEMDFS2	開始
編集...	MFES (MSS)	DEMDFS3	開始

The main interface shows the 'iFS Control: DEMDFS2' panel. The 'Diagnostics' section is highlighted with a red box, showing the following settings:

- Log:  A
- Dump:  B
- Trace:  C/x
- 1000 Blocks
- Display

The 'Resources' section is also highlighted with a red box, showing:

- iFileshare ▼
- iFSView
- Control

The 'iFS Control: DEMDFS2' panel includes a 'Refresh' button and an 'Interval (Secs)' input field. Below this, the 'System Status' is displayed:

System Status: Role Hot Standby Node  
DB Reference File Replicated  
Primary Node Active  
Hot Standby Nodes Active  
Initial Configuration Available

The 'Partner Fileshare Table' is shown below:

Node	Name	Connection Status	Role	Host	Port	DBREF	Checked	Prim_ok
02	DEMDFS2	Acquired	Standby	CCISM	0000	Y	Y	DEMDFS1_ok
01	DEMDFS1	Acquired	Primary	CCISM	0000	Y	Y	DEMDFS1_ok
03	DEMDFS3	Acquired	Standby	CCISM	0000	Y	Y	DEMDFS1_ok

At the bottom of the interface, the 'iFS Control: DEMDFS2' panel is repeated, showing a 'Refresh' button and the text 'casrdoa1: p 794475'.



[www.microfocus.co.jp](http://www.microfocus.co.jp)

© 2017 Micro Focus

本スライドに記載の会社名・製品名は、各社の商標または登録商標です。  
本スライドに記載の内容をマイクロフォーカスの許可無く転載することを禁じます。