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 デバッギング
- 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)

開発支援機能の強化

対応 IDE の追加

- Eclipse 4.6 (64 bit)

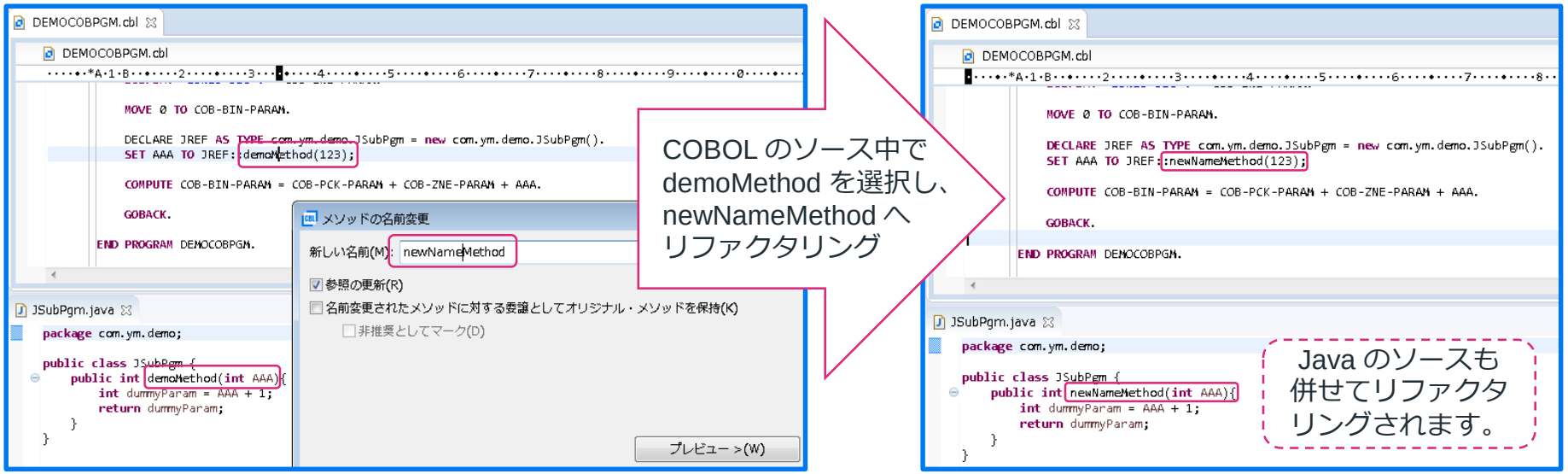
  - ✓ Eclipse 4.6 (64 bit) がインストーラーにバンドルされ、64 bit 版 JRE で Eclipse IDE の操作が可能となります。
  - ✓ プラグラインインストールで、Eclipse 4.4.1, 4.4.2, 4.5, 4.6 (32 bit) も利用可能です。

- 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

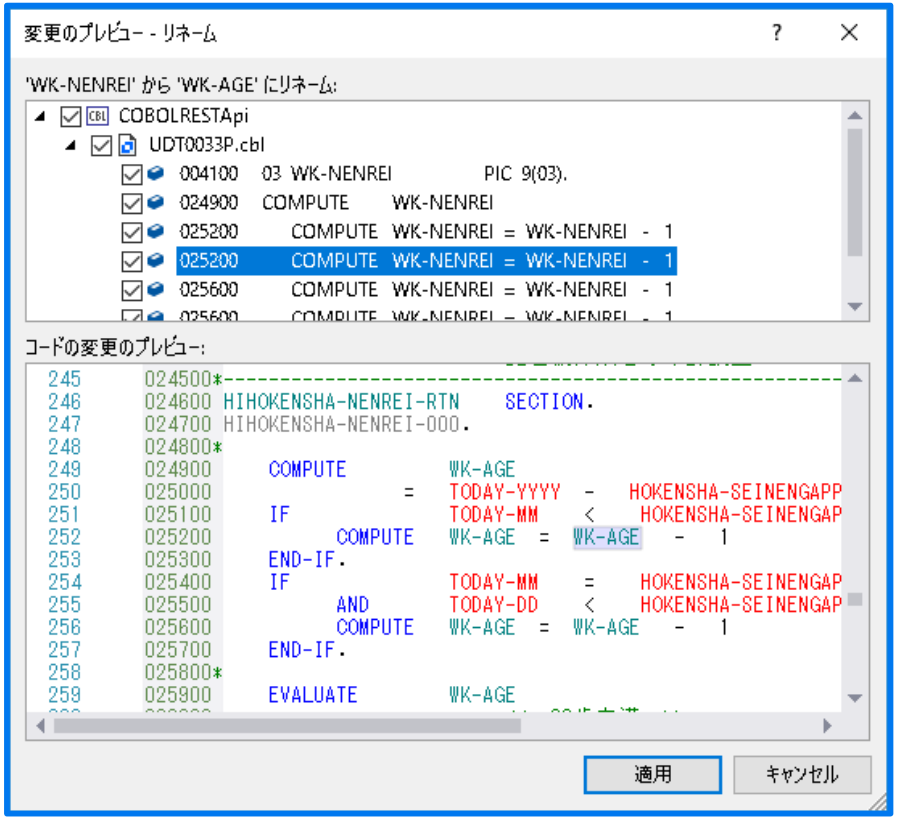


- Code Analysis ルールに 64 bit Readiness を追加
- ビルドパスにコピーファイルが格納されたプロジェクト外フォルダを追加
- 複数プロセスによる並列コンパイル

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

リファクタリング処理イメージ



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 AFILE ASSIGN TO "SimpleDemo.json"
           ORGANIZATION IS LINE SEQUENTIAL.
       DATA DIVISION.
       FILE SECTION.
       FD AFILE.
       01 AFILE-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 AFILE.
           JSON GENERATE AFILE-REC FROM DEMO-JSON-DATA.
           WRITE AFILE-REC.
           CLOSE AFILE.
           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 ....] <ICONV>¥n↓
# Key:    <...>    Mandatory↓
#         [...]    Optional↓
# Values can be delimited using any non-line breaking whitespace.↓
# <ICONV> 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↓
↓
```

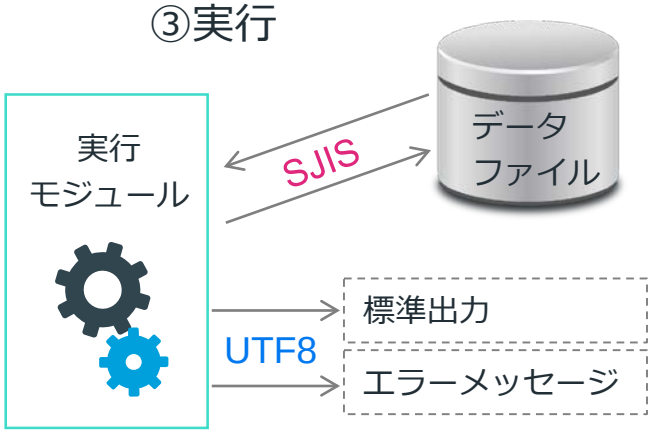
SJIS のみならず EUC 等にも対応

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



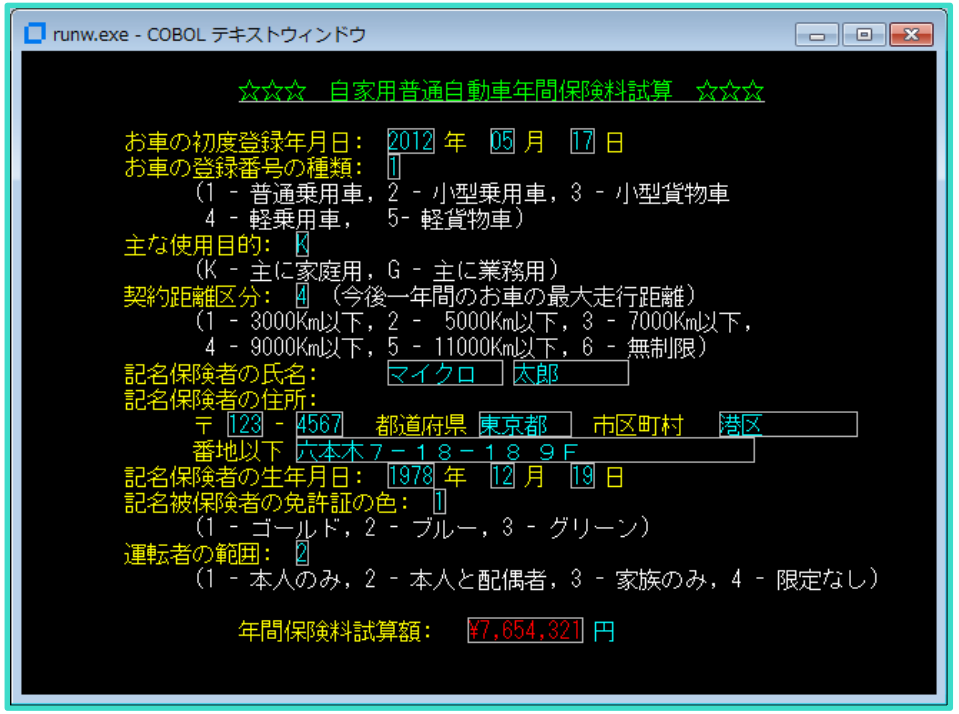
ログは他システム同様
UTF8 で記録可

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

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。

サンプルイメージ：

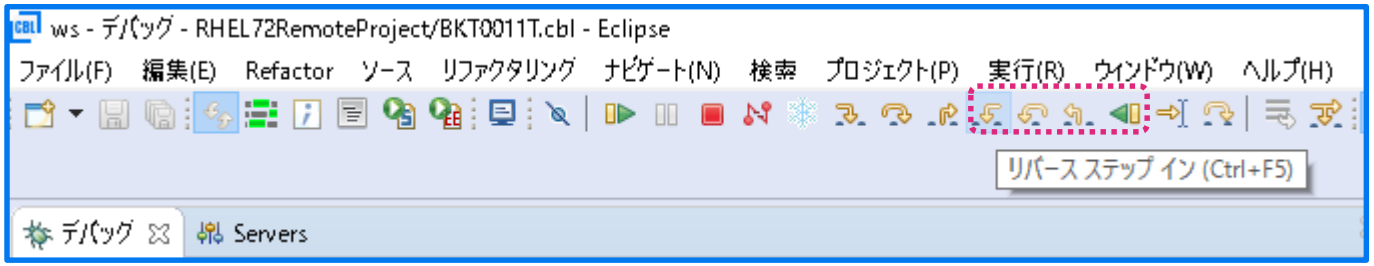


エンタープライズ DevOps 支援機能

Undoデバッギング

- 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：



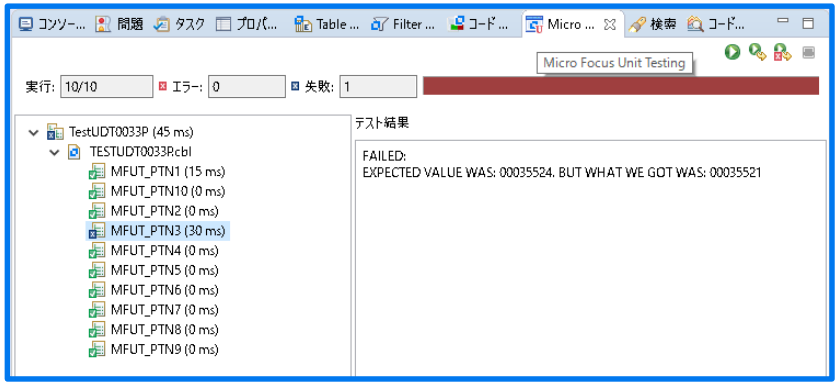
ユースケース例

- ✓ 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

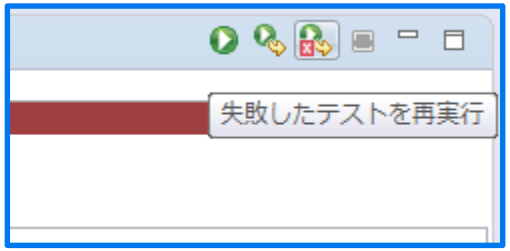
MFUnit – COBOL 専用の単体テストフレームワーク

- 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出力

利用例：

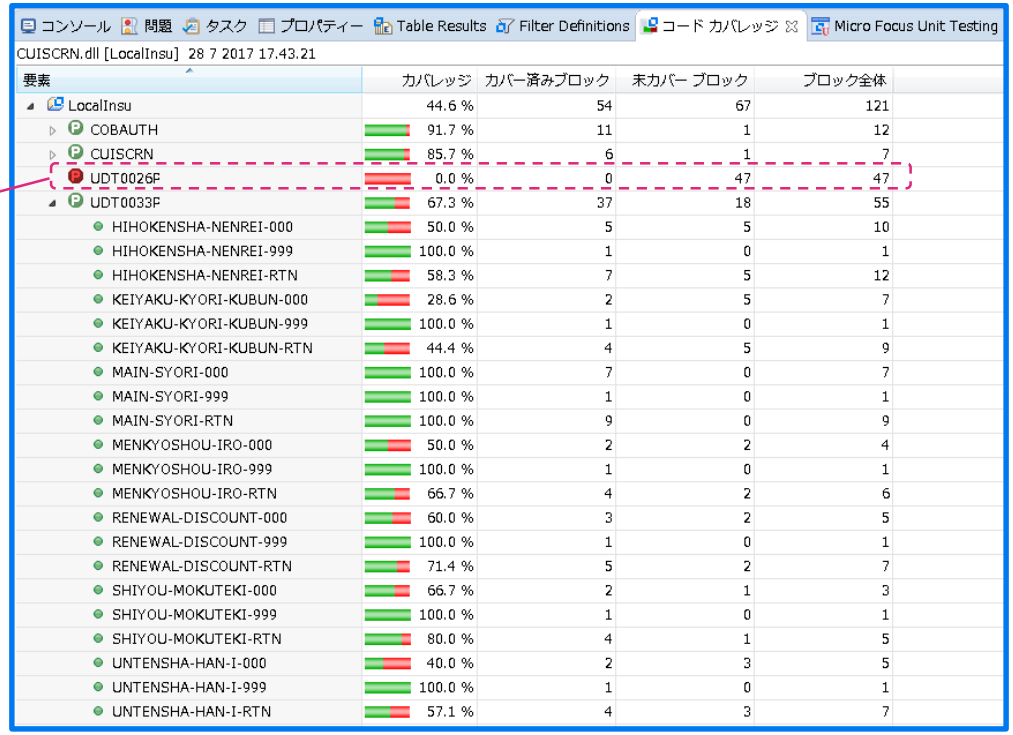
```
    if result not equal 42
        exhibit named result
        call MFU-ASSERT-FAIL-Z using by reference z"Result does not equal 42"
    end-if
```

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

出力イメージ：



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
KEIYAKU-KYORI-KUBUN-000	28.6 %	2	5	7
KEIYAKU-KYORI-KUBUN-999	100.0 %	1	0	1
KEIYAKU-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

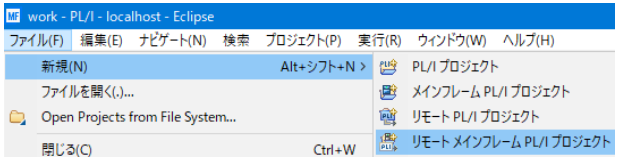
本機能強化により未実行のプログラムについてもカバレッジの分析結果に反映されるようになりました。

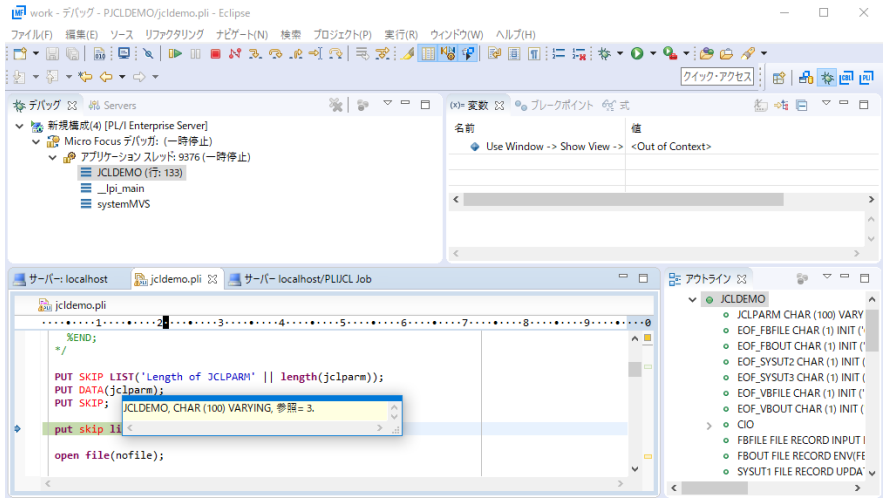
Enterprise Developer 固有の強化

PL/I 言語

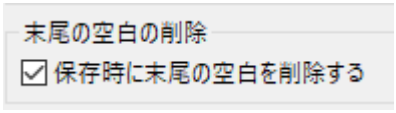
- デバッガーの刷新
  - ✓ Eclipse IDE 上で PL/I ソースのデバッグが可能になりました。
    引き続き CodeWatch も使用可能です。

  - ✓ 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。





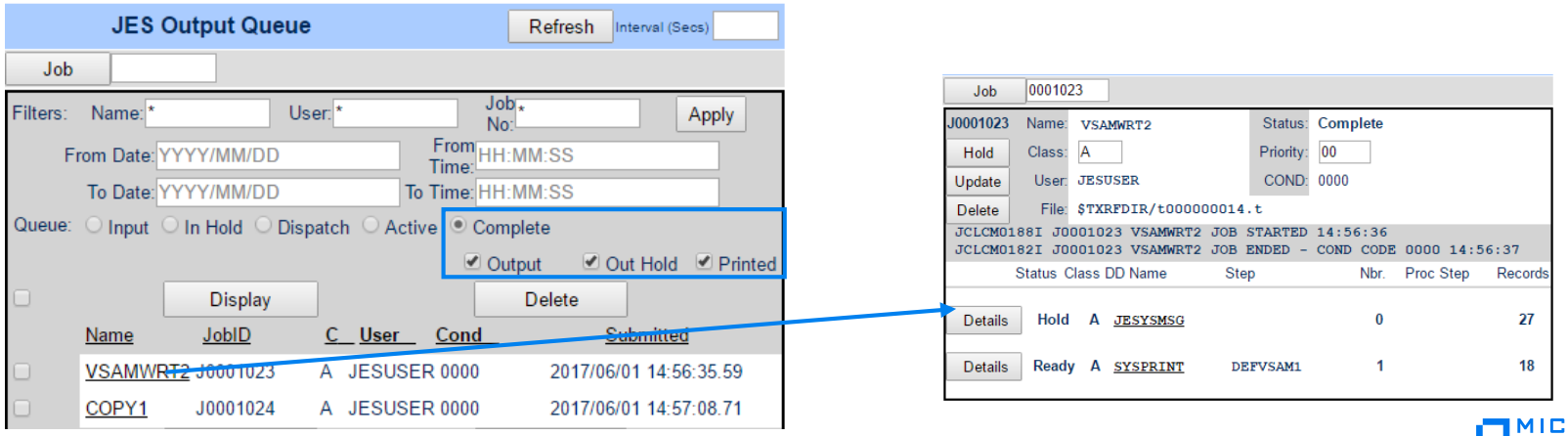
- エディターの強化
  - ✓ 行末尾の空白を保存時に削除する指定が可能になりました。



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。

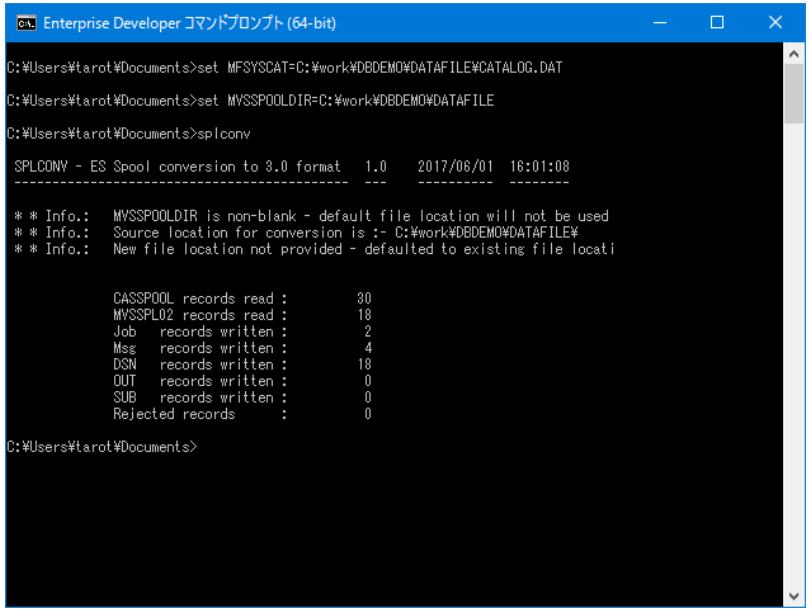


JES – JCL スプール機能の強化

ジョブ番号を 5 桁から 7 桁へ拡大

!! 重要 !!

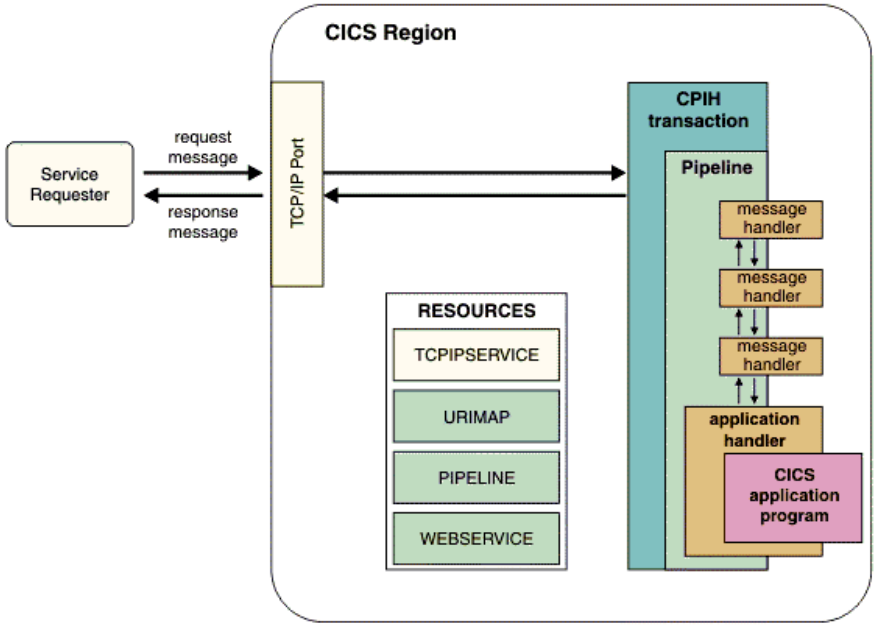
JES機能を使用する既存のインスタンスの場合、JCL 実行前に各インスタンスに対してスプール変換ユーティリティを実行する必要があります。
JES 機能を使用しない既存のインスタンス、新規のインスタンスでは変換の必要はありません。



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：

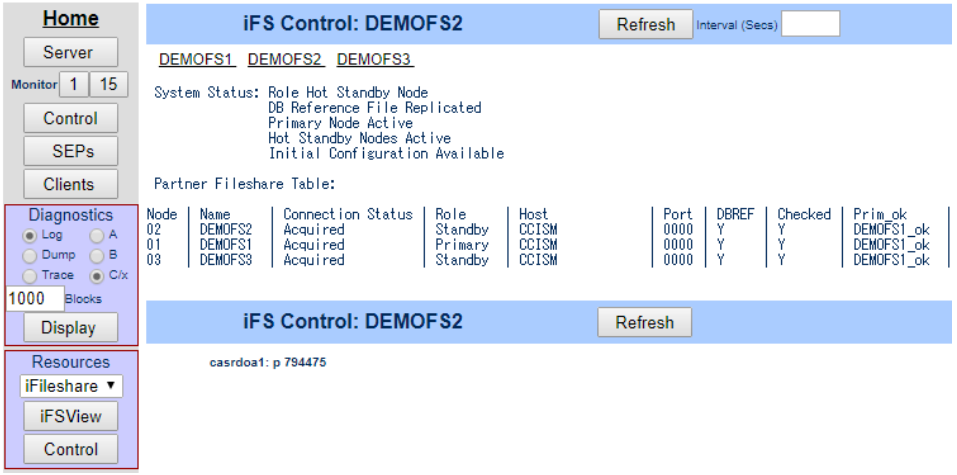


iFileshare – 高可用性支援機能の強化

- 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。



www.microfocus.co.jp