

# IBM App Connect Enterprise V12 アップデートセミナー

日本アイ・ビー・エム システムズ・エンジニアリング  
2021/11/25

# はじめに

- 当資料はApp Connect Enterprise V12についてご紹介しています。
- 記載内容の一部はFixpack 12.0.2.0時点の実機検証による動作結果を基にしており、今後変更される可能性があります。

当資料に記載している内容は可能な限り稼動確認を取っておりますが、日本アイ・ビー・エム株式会社、及び日本アイ・ビー・エム システムズ・エンジニアリング株式会社の正式なレビューを受けておらず、当資料で提供される内容に関して日本アイ・ビー・エム株式会社、日本アイ・ビー・エム システムズ・エンジニアリング株式会社は何ら保証するものではありません。

従って、当資料の利用またはこれらの技法の実施はひとえに使用者の責任において為されるものであり、当資料によって受けたいかなる被害に関しても一切の補償をするものではありません。

# 目次

- ハイライト
- V12の主な新機能
  - ◆ 新しい開発インターフェースの提供
  - ◆ JSON妥当性検査のサポート
  - ◆ OpenAPI 3.0のサポート
  - ◆ 新しいテスト・ツールのサポート
  - ◆ ibmintコマンドの提供
  - ◆ ビジネス・トランザクション・モニタリング
- V11のFixpackで追加された主な機能
- マイグレーション
- Q&A

# ハイライト

# IBM App Connect

- 市場をリードする、あらゆる対象を統合可能な IBM ソリューション

提供機能:

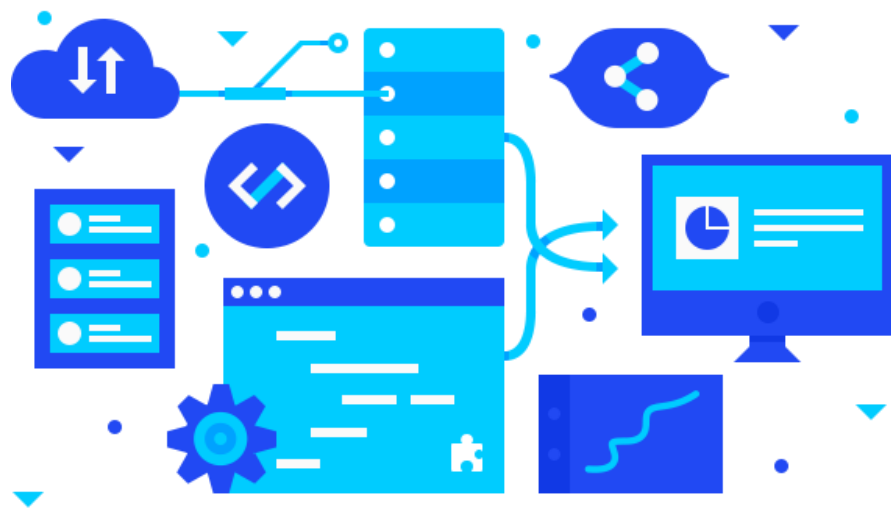
1. 接続
2. ルーティング
3. 変換

アプリケーションおよびサービスの間で、それらのシステムやサービスが使用するデータの形式やプロトコルに関係なく、セキュアにデータを共有可能に

稼働環境:

Linux, AIX, Windows, z/OS

Red Hat OpenShift, Cloud Pak for Integration



# IBM App Connect

高い信頼性と性能、多様な機能、豊富なデプロイの選択肢を提供



- 事前に構築されたスマートなアプリケーション・コネクタ
- 標準提供のプロトコル・コネクタ
- ユニバーサル・データ変換ツール

- Mapping Assist
- DFDL マッピング
- Java およびユーザー定義のノード

- 単体テスト・フレームワーク
- 記録および再生
- ノード/フロー・レベルのテスト

- ハイブリッド・ダッシュボード
- マルチプロセス、マルチスレッド
- パフォーマンスの微調整
- 高可用性ソリューション



オンプレミス



メイン  
フレーム



あらゆるパブリックまたは  
プライベートクラウド



ハイブリッド  
クラウド

# App Connect Enterprise のロードマップ

現在 V12.0.2.0

2011 2012 2013 2014 2015 2016 2017 2018 2019 2020 2021 2022 2023 2024 2025 2026

IBM Integration Bus (IIB) V10 のリリース開始からアジャイル・デリバリーに切り替え、四半期ごとに新機能を含む保守を提供

IBM App Connect Enterprise V12/ CP4I  
2021.2.1

2021年5月28日リリース

IBM App Connect Enterprise V11

IBM Integration Bus V10

アジャイル型

↑ IIBV10 の標準サポートは 2022 年 4 月に終了

ウォーターフォール型

IBM Integration Bus v9

WebSphere Message Broker v8

過去には (IIBV10 より前) ウォーターフォール型で開発していました。フィックスパックはどちらかというと保守目的で、提供頻度も高くありませんでした (6 カ月から 12 カ月に 1 回)。

# IBM App Connect Enterprise V12

## テストおよび DevOps 自動化

リスクを軽減しながら、  
アジャイル・プラク  
ティスの導入を加速

テスト主導の新しい  
開発機能により、  
シフト・レフト戦略  
を加速

## ハイブリッドクラウド 操作

多様な稼働環境に跨り  
運用の可視化と制御を  
強化

すべての環境の統合  
ランタイムの透明性  
をエンドツーエンド  
で提供する新たな  
ダッシュボード

## 開発生産性

インテリジェントで最適  
化されたツールで価値実  
現までの時間を短縮

*API Management* と  
統合された共通の  
ユーザー・エクスペ  
リエンス、新規の  
チュートリアルおよ  
びアクセラレーター  
を提供

## 接続の拡張

接続されたシステムと  
アプリケーションを  
簡素化、強化、自動化

100 を超えるクラウ  
ド・サービスとのコ  
ネクタ<sup>※</sup>を継続的  
に拡充

※IBM Cloud上で稼働するApp  
Connect on CloudのCloud  
Connector Planが同梱



# 刷新・強化されたツールキット

- 統合開発者向けに拡充されたユーザー体験と洗練されたデザイン

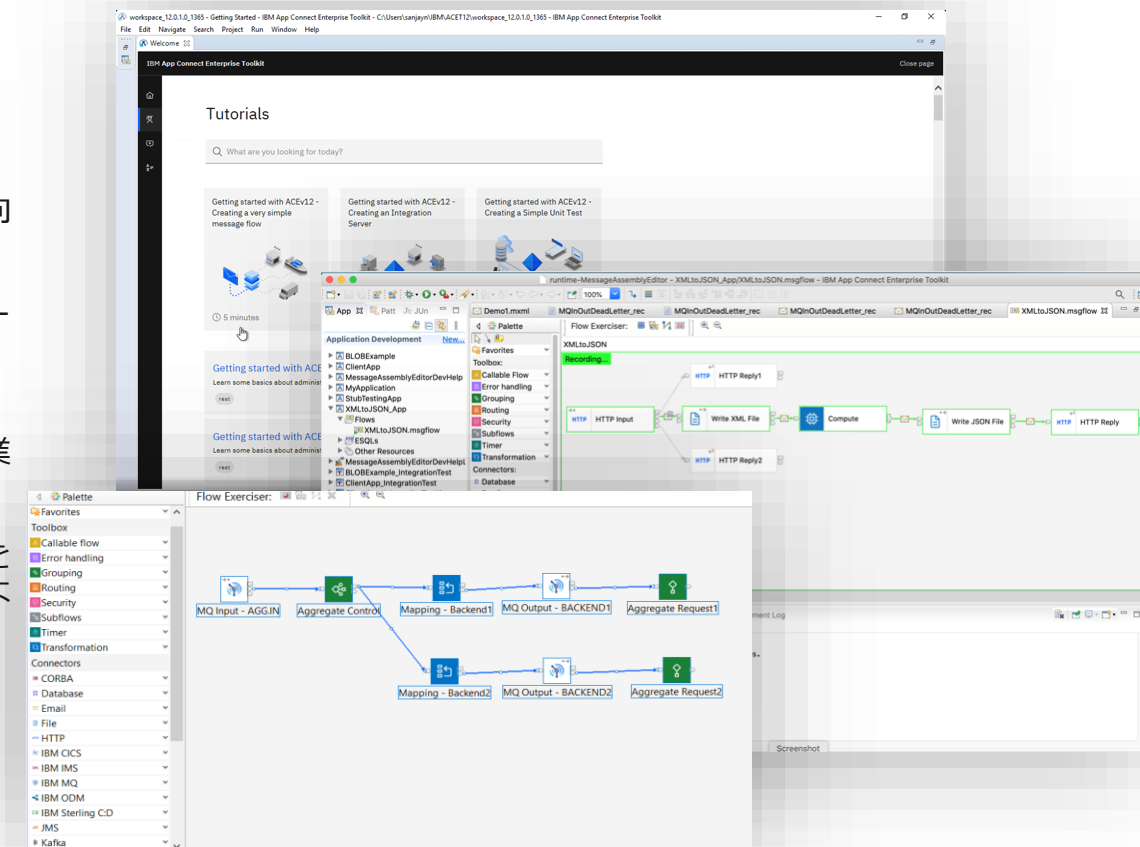
新しいユーザー・インターフェースにより、ツールキットを API Connect等の統合製品との連携を強化

ワークスペースの簡素化により、生産性を向上

色分けされたデザインによりプロセス・ノードの種別を素早く識別

新しいチュートリアル、ウィザード、クイックスタートガイドにより、新しい作業や開発をすぐに開始

プロセス・ノード、機能、チュートリアルを簡単に検索・フィルタリングできることにより、より素早く操作



# テスト自動化を支援する機能

テストを生成、実行、および管理する新しいテスト・フレームワークを提供

記録・保管されたメッセージ・フロー内のデータを用いて、期待される処理が行われているかを検証することが可能

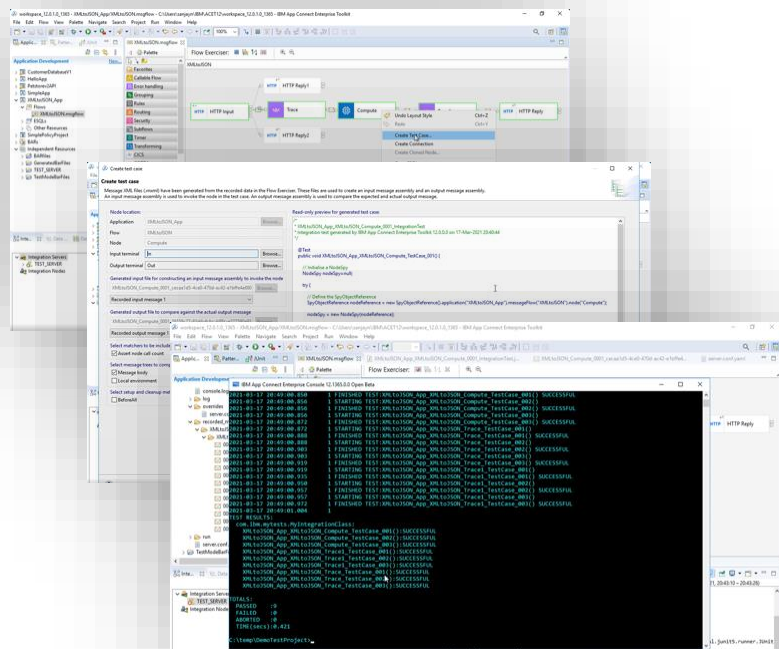
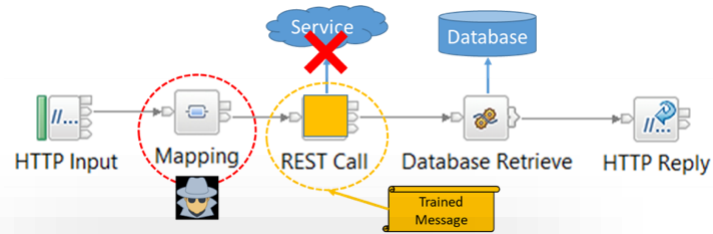
テストを**自動生成**、接続先の処理を**シミュレート**するモックアップの生成により、従来の統合テストにかかる調整の手間を大幅に低減可能

## 価値

**生産性**: テストの生産性を向上することで、全体の品質を向上し、ソリューションのコストを削減

**リスクの軽減**: ビジネスの変化に遅れることなく自信を持って対応でき、リスクを軽減

**市場投入までの時間の短縮**: リグレッション・テストを合理化することで、ソリューションの市場投入までの時間を短縮



# OpenAPI 3.0 をサポートする新しいAPI オーサリング機能

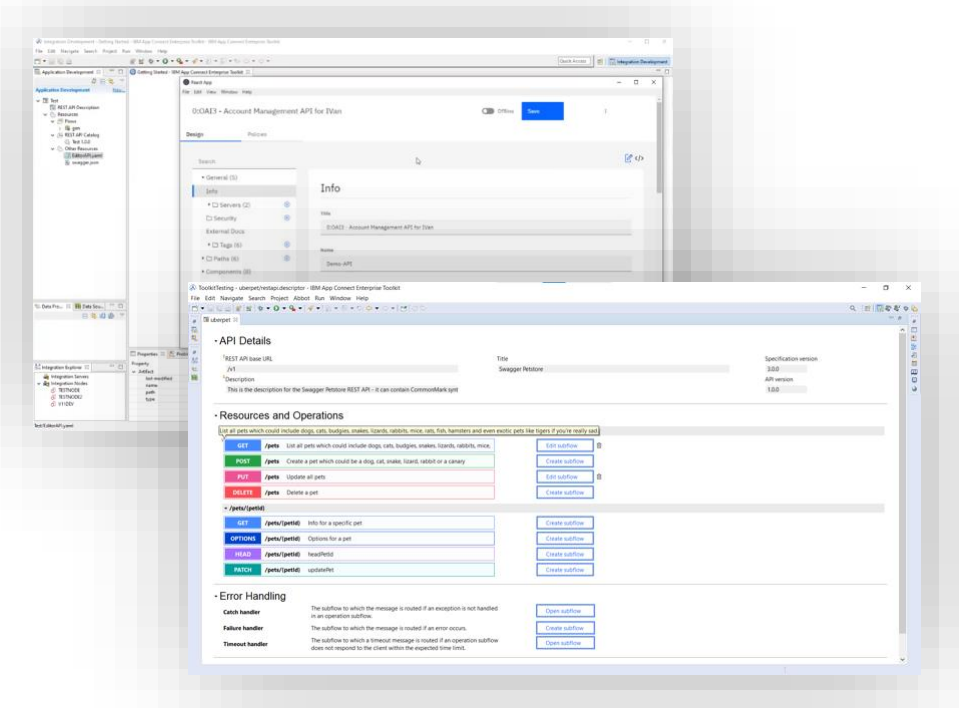
## REST API の実装をシームレスに作成、保護、および管理

App Connect (ツールキットおよび Designer) と API Connect を跨った単一の API 定義画面を提供

統合フローのコンポーネントおよび ゲートウェイ・ポリシーの定義など、API 作成の各ステップで**構築、テスト、デバッグ、の反復**を可能に

パイプラインの各ステージで管理可能な **単一の API 成果物**を作成および**デプロイ**

最新 API 標準の **OpenAPI 3.0** をサポート



# ハイブリッド・ダッシュボード

多様な環境にわたるACEのランタイムを  
統合・表示して管理

各種パブリック・クラウド、オンプレミス、コンテナ環境など、すべてのデプロイメント・ターゲットにわたりモニター、管理、および制御を行うための**単一の統合インターフェース**

**問題判別の迅速化**、可視性の向上、リアルタイムのトラブルシューティング、および問題解決時間の短縮

すべてのデプロイメントにわたり、すべての統合および資産の**正常性を迅速に把握**

デプロイ、管理、モニター、ログ、アラート、ヘルス・チェック、状態などのための**単一のコントロール・プレーン**提供



# App Connect Enterpriseのライセンス

## ■ App Connect Enterprise V11からのライセンスの変更点

### ◆ スタンバイライセンスの考え方の変更

- IBM Integration Bus V10まではHA構成時に待機サーバーに必須だったIdle StandbyライセンスがApp Connect Enterprise V11からIdle Standbyライセンスが不要に
- 既存のIIB Idle StandbyライセンスはApp Connect Enterpriseへのアップグレードも可能

### ◆ 非生産的使用（開発・検証環境用のライセンス）の提供

- 本番環境以外の環境（UAT環境、テスト環境、開発環境等）用に安価なNon-Productionライセンスを提供

## ■ Cloud Pak for IntegrationライセンスによるApp Connect Enterpriseの利用

- ◆ 既存のApp Connect Enterpriseライセンスからのアップグレードが可能
- ◆ App Connect Enterpriseをコンテナ環境で稼働させるためのOpenShiftのライセンスが同梱
  - 従来通りOSネイティブにインストールして利用することも可能
- ◆ 経費として計上可能なサブスクリプション・モデルも選択可能
- ◆ 余剰ライセンスをCloud Pak for Integrationで提供されている他のコンポーネント（MQ、API Connect, DataPower, Aspera等）の利用に充当することも可能

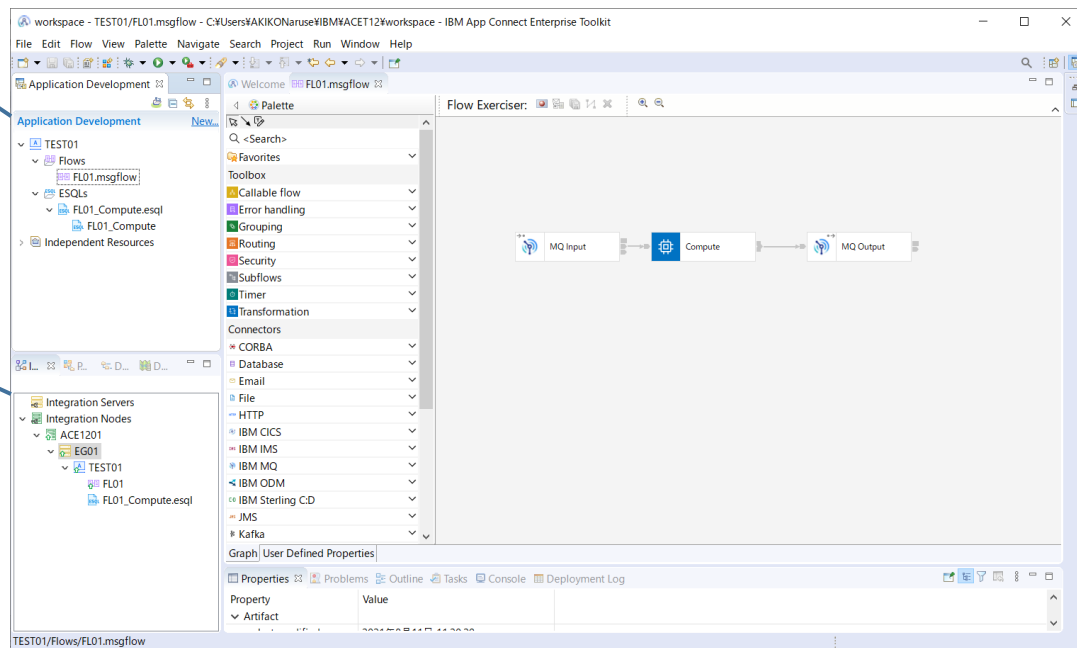
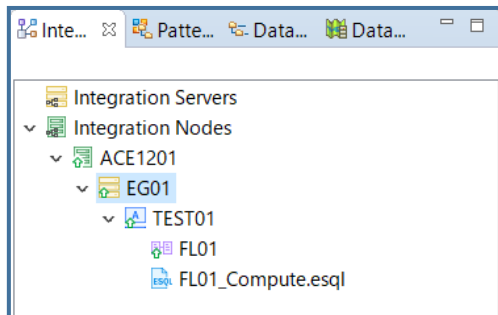
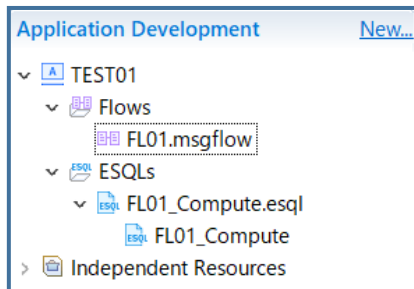
# V12の主な新機能

新しい開発インターフェースの提供

# 新しい開発インターフェースの提供

## ■ 新しいToolkitのインターフェースの提供

- ◆ Eclipse 4.15 (2020-03) を使用
- ◆ 新しいナビゲーター・アイコンや統合ノード/統合サーバのアイコンを提供

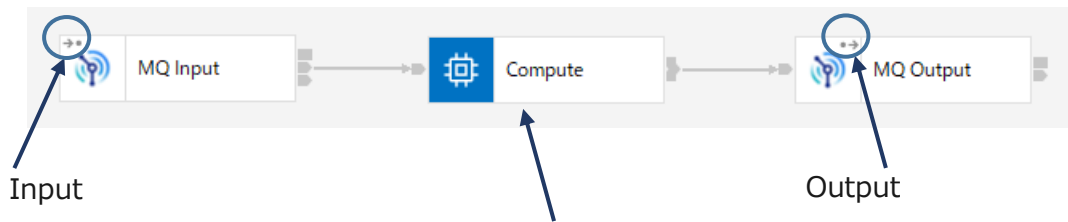




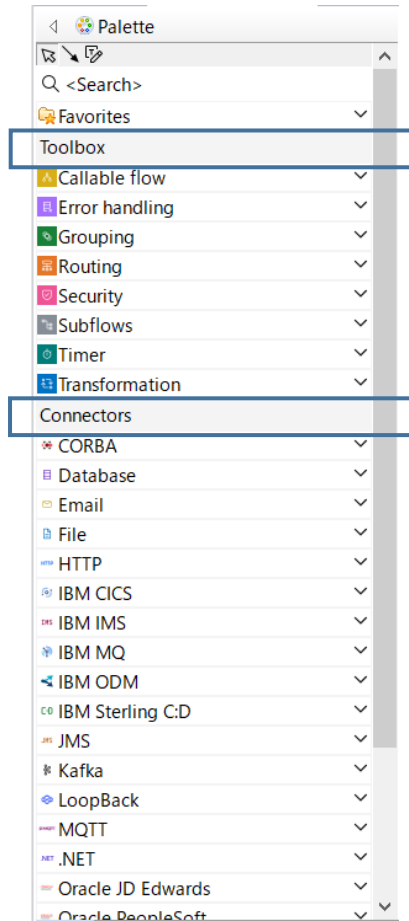
# 新しい開発インターフェースの提供

## ■ フロー・エディター・パレットの変更

- ◆ ノードをToolboxとConnectorsに大別
  - ◆ Toolboxノード（アイコンの背景が青）
    - ・ 処理系のノード群
  - ◆ Connectorノード（アイコンの背景が白）
    - ・ インターフェース系のノード群
    - ・ プロトコルごとにアイコンは共通
      - ✓ ノードの端に表示されている矢印で種別を表している



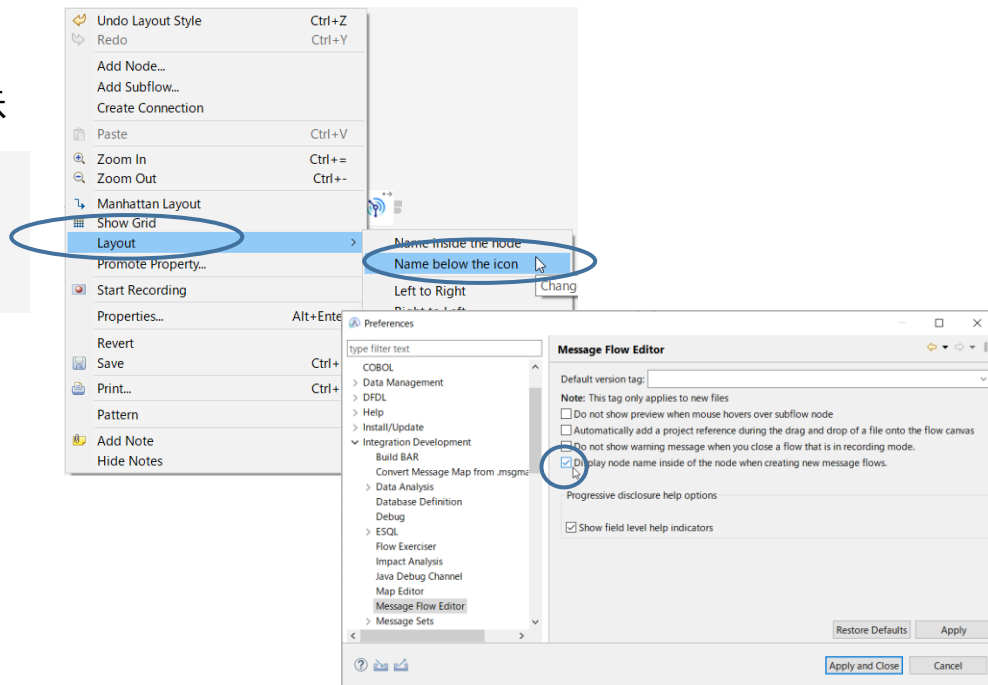
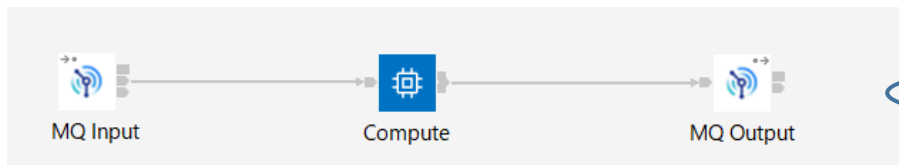
ノード名は変更可能だが、  
サイズが固定なので表示できる文字数が限られる



# 新しい開発インターフェースの提供

## ■ ノードの表示を変更することも可能

- ◆ フロー・エディターで右クリック> Layout> Name below the icon
- ◆ 全てのフローの設定を変更したいときは Window> Preferences> Integration Development> Message Flow Editorで“Display node name inside of the node when creating new message flows”のチェックを外す
- ◆ 従来通りノード名を下に表示
- ◆ マイグレーションしたフローはこちらの表示

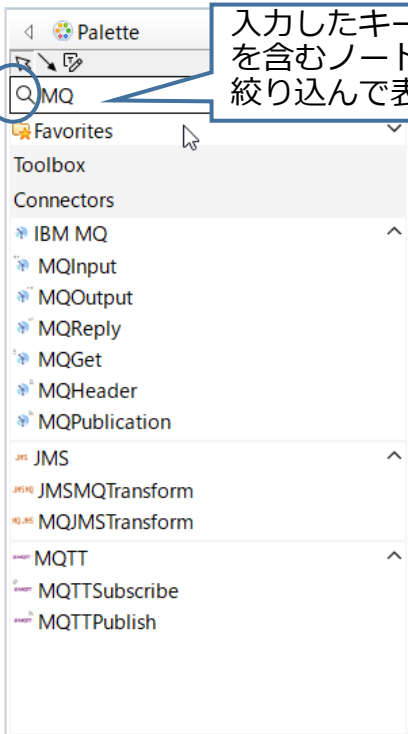


# 新しい開発インターフェースの提供

## ■ 便利な検索機能の提供

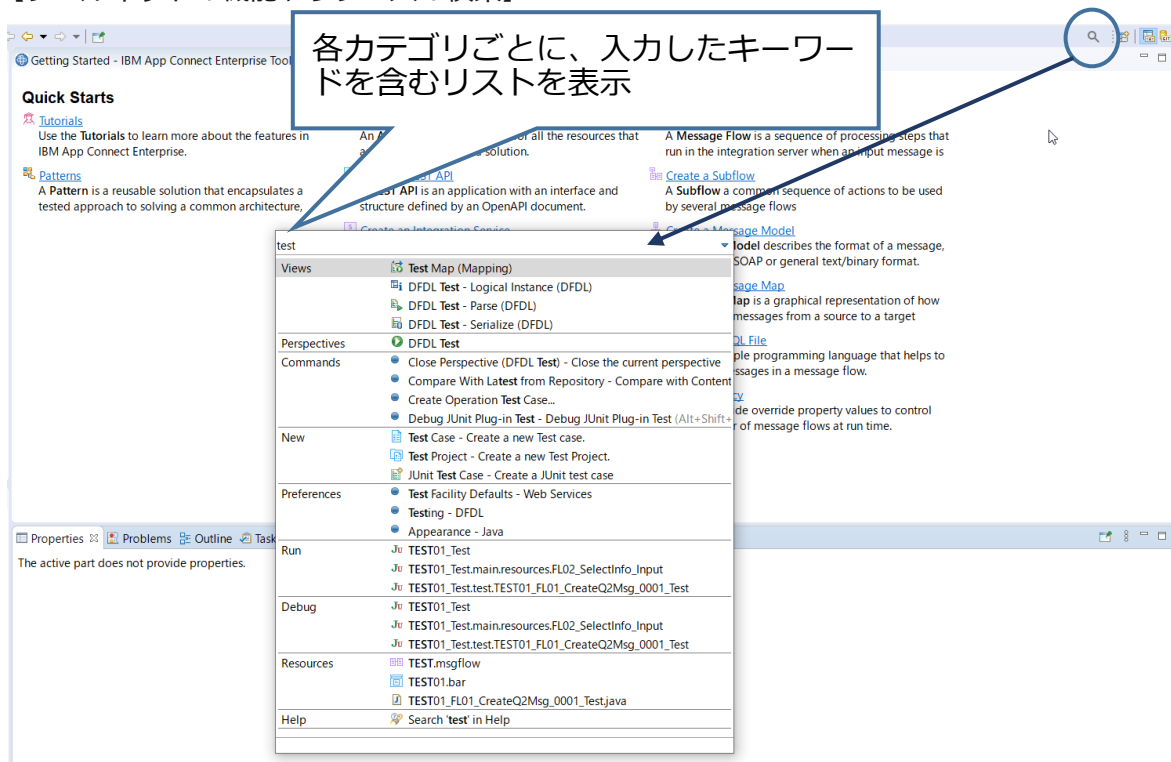
### 【ノードの検索】

入力したキーワードを含むノードのみ、絞り込んで表示可能



### 【ツールキットの機能やリソースの検索】

各カテゴリごとに、入力したキーワードを含むリストを表示



# フロー・エクササイザー

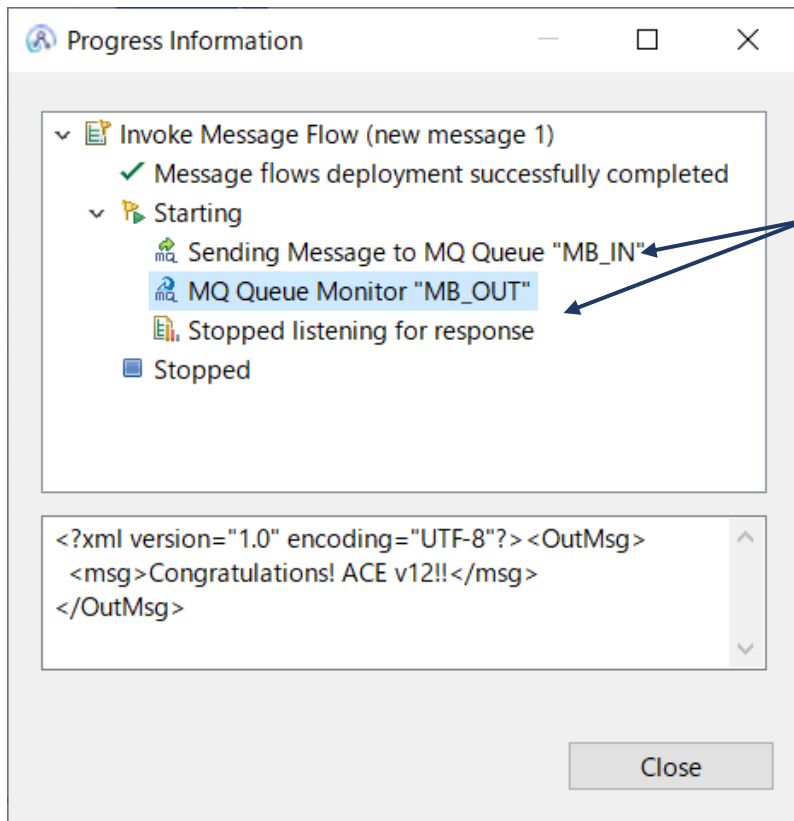
## ■ フロー・エクササイザーの機能拡張

- ◆ フロー・エディターの画面上にある「Start flow exerciser…」ボタンを押し、「Recording」状態で「Send a message to the flow」アイコンからテストメッセージを作成して送信

The screenshot displays the Flow Exerciser interface. At the top, a toolbar contains a red square button with a white play icon, which is circled in blue. A tooltip below it reads: "Start flow exerciser to record the path a message takes through the flow." Below this, the main workspace shows a flow diagram with an "MQ Input" component connected to a "Filter" component. A blue bar at the top of the workspace indicates the state "Recording...". A tooltip for the "Send a message to the flow" icon (a blue square with a white envelope icon) is also visible. A "Send Message" dialog box is open on the right side of the screen. The dialog has a "Name" field set to "new message 1" and an "Input Location" dropdown set to "MQ Input". The "Message Details" section contains a text area with "test message" entered. At the bottom right of the dialog, the "Send" button is circled in blue. A blue arrow points from the "Send" button in the dialog to the "Send a message to the flow" icon in the workspace.

# フロー・エクササイザー

- ◆ Progress Informationウィンドウで送受信メッセージの確認が可能



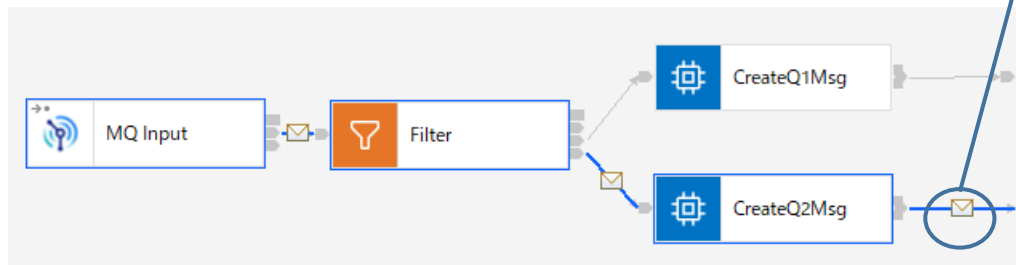
カーソルを合わせると入力メッセージ、出力メッセージの確認が可能

# フロー・エクササイザー

- ◆ 通った経路が青でハイライトされる
  - 「View path the message took through the flow and see the message content on the path」をクリック



View path the message took through the flow and see the message content on the path.



- ◆ メッセージ・アイコンをクリックすると、メッセージの中身を確認可能
  - Message Assemblyのツリー構造
- ◆ Saveすれば編集したり、再度テストに利用可能

**Recorded Message Assembly**

Save the recorded message assembly so that it can be used in the Flow Exerciser as input to a message flow input node or to use in a test case.

Name:

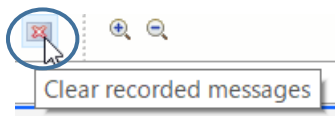
Message number: 1

Name	Type	Value	Namespace
MessageAssembly			
version	INTEGER	1	
checkpoint			
environment			
localEnvironment			
exceptionList			
message			
Properties			
MQMD			
XMLNSC			
OutMsg			
msg	CHARACTER	Congratulations! ACE v12!!	

Buttons: Save, Close

# フロー・エクササイザー

- ◆ 保管したレコードメッセージ はテストしたフローを含むApplication配下に保存
  - 「Other Resources」に保存
  - ファイルの拡張子は“.mxml”
  - Inputメッセージだけでなく、メッセージ・アイコン地点のメッセージをすべて保管可能
  - 保管したメッセージはMessage Assembly エディターで編集可能
- ◆ レコードメッセージを保管しない場合はClearアイコンで破棄



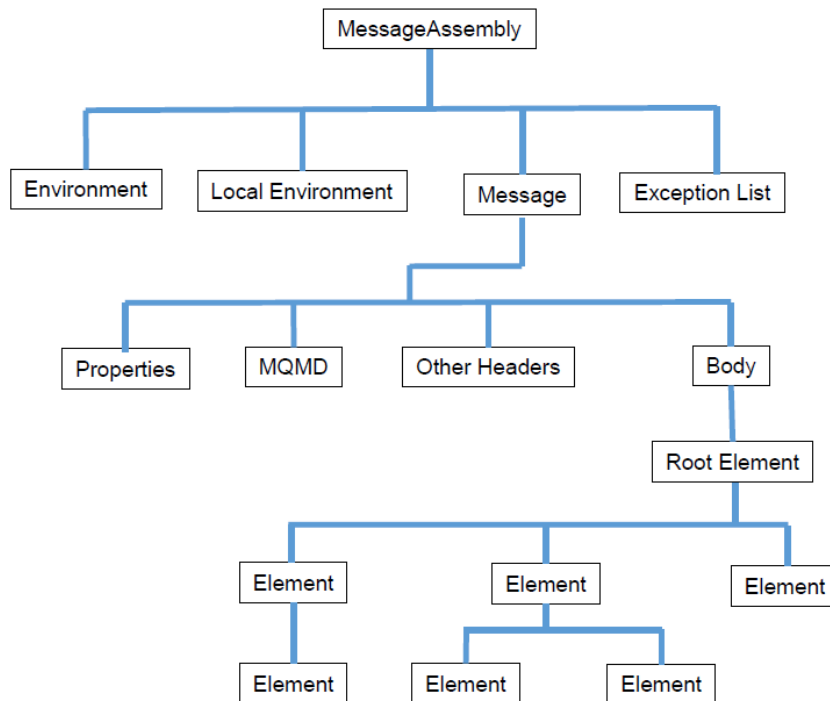
# Message Assembly

## ■ Message Assembly

- ◆ フロー内のどのポイントでもメッセージの論理ビューを表示可能

## ■ Message Assemblyのツリー構造

- ◆ 4つのツリーから成る
  - Environment Tree
  - Local Environment Tree
  - Message Tree
  - Exception List Tree





# Message Assembly

- ◆ Applicationに保管されたmxm1ファイルをMessage Assemblyエディターで編集可能
  - エレメントの値の編集だけでなく、エレメントの追加、トランスポート・ヘッダーの追加、LocalEnvironmentへの値設定なども可能

workspace - TEST01/FL01\_00000C08\_611351D9\_00000001\_1.mxml - C:\Users\AKIKONaruse\IBM\ACET12\workspace - IBM App Connect Enterprise Toolkit

File Edit Navigate Search Project Run Window Help

Application Development Welcome FL01.msgflow FL01\_00000C08\_611351D9\_00000001\_1.mxml

Application Development

- TEST01
  - Flows
    - FL01.msgflow
  - ESQLs
    - FL01\_Compute.esql
    - FL01\_Compute
  - Other Resources
    - FL01\_00000C08\_611351D9\_00000001\_0.r
    - FL01\_00000C08\_611351D9\_00000001\_1.r
    - FL01\_00000C08\_6113521D\_00000002\_0.r
    - FL01\_00000C08\_6113521D\_00000002\_1.r
    - FL01\_inputMessage.xml
  - BARs

Message Assembly Editor for JUnit Tests

Set the attributes and values for a test message

Name: FL01\_00000C08\_611351D9\_00000001\_1.mxml

Name	Type	Value	Namespace Declaration
MessageAssembly			
version	INTEGER	1	
checkpoint			
environment			
localEnvironment			
exceptionList			
message			
Properties			
MQMD			
XMLNSC			
OutMsg			
msg	CHARACTER	Congratulations! ACE v12!!	

Integration Servers

# JSONの妥当性検査のサポート

# JSONの妥当性検査

- JSONスキーマやOpenAPI定義ファイルを使用したJSONドキュメントの妥当性検査が可能に
- サポートされる規格
  - ◆ JSON Schema draft 04 (<https://datatracker.ietf.org/doc/html/draft-fge-json-schema-validation-00>)
  - ◆ JSON Schema draft 05 (<https://datatracker.ietf.org/doc/html/draft-wright-json-schema-validation-00>)
  - ◆ Swagger 2.0 (<https://swagger.io/specification/v2/>)
  - ◆ OpenAPI 3.0. x (<https://swagger.io/specification/>)
  - ◆ ただし、キーワード "default", "format", "discriminator" は未サポート
- server.conf.yamlでJSONスキーマの妥当性検査を無効化することも可能

server.conf.yamlの抜粋

```
JSON:  
disableSchemaLookupExceptionWhen: "
```

- JSONスキーマは初回のメッセージ処理時にロードされてコンパイルされる

# JSONの妥当性検査

## ■ パース時の妥当性検査

- ◆ 入力ノードなどの「Validation」タブの「Validate」で「Content and Value」を選択
- ◆ 妥当性検査でエラーがあった場合の「Failure action」を指定
- ◆ 「Input Message Parsing」の「Message domain」に「JSON」を指定
- ◆ 「Input Message Parsing」の「Message model」に妥当性検査に使用するJSONスキーマを指定
  - 参照スキーマが共有ライブラリの場合は {} でライブラリ名が参照される

Application Development

- JSON\_TEST
  - Flows
    - JSON01.msgflow
    - ESQLs
    - Referenced Libraries
      - JSON\_Schemas
        - JSON Schemas
          - JsonSchema\_DraftNOTSUPPORTED.json
          - JsonSchema\_EmployeeResponse.json
          - JsonSchema\_Insert.json
          - jsonschema\_insert\_ArrayAssumed.json
          - JsonSchema\_View.json
        - Other Resources

Properties Problems Outline Tasks Console Deployment Log

HTTP Input Node Properties - HTTP Input

Description Determine how the node parses the incoming messages.

Basic Message domain JSON : For JavaScript Object Notation messages

Advanced Message model {JSON\_Schemas}.jsonSchema\_Insert.json

Input Message Parsing Message <[if the JSON schema is not at the root of the schema document]

Parser Options

Error Handling Physical format

Validation

Validation settings for the HTTPInput node

Validate\* Content and Value

Failure action\* Exception

共有ライブラリのJSONスキーマを利用

妥当性検査に使用するJSONスキーマを指定

# JSONの妥当性検査

## ■ メッセージ作成時の妥当性検査

- ◆ メッセージを作成したComputeノードや出力ノードなどの「Validation」タブの「Validate」で「Content and Value」を選択
- ◆ 妥当性検査でエラーがあった場合の「Failure action」を指定
- ◆ 妥当性検査に使用するJSONスキーマをESQLなどでOutputRoot.Properties.MessageSetに指定

The image shows a screenshot of an IDE interface with several key elements:

- Left Panel (Application Development):** A tree view showing a project structure. Under "Referenced Libraries" > "JSON\_Schemas", the file "JsonSchema\_EmployeeResponse.json" is highlighted with a yellow box. A blue arrow points from this box to the code snippet above.
- Code Snippet (ESQLの抜粋):** A line of code: `set OutputRoot.Properties.MessageSet = '{JSON_Schemas}JsonSchema_EmployeeResponse.json';`. The path part is highlighted with a yellow box. A blue arrow points from this box to the "Failure action\*" field in the validation settings below.
- Right Panel (Properties):** The "HTTP Reply Node Properties - HTTP Reply" tab is selected. The "Validation settings for the HTTPReply node" section is expanded. The "Validate\*" field is set to "Content and Value", and the "Failure action\*" field is set to "Exception". A blue arrow points from the "Failure action\*" field to the "Compute Node Properties - Compute" tab below it.
- Bottom Panel (Compute Node Properties):** The "Compute Node Properties - Compute" tab is selected. The "Validation settings for the Compute node" section is expanded. The "Validate\*" field is set to "Content and Value", and the "Failure action\*" field is set to "Exception".

共有ライブラリのJSONスキーマを利用

ESQLの抜粋

妥当性検査に使用するJSONスキーマを指定

# JSONの妥当性検査

## ■ OpenAPI定義を使用した妥当性検査

- ◆ REST APIプロジェクトではREST APIエディタのチェックボックス設定でノードプロパティが自動的にセット

REST API エディタ

---

### ▾ Message Validation

Enable for request messages

Allow unexpected request body

---

- ◆ アプリケーションではJSONスキーマと同様にノードのプロパティで指定
  - 「Message model」に妥当性検査に使用するOpenAPI定義ファイルを指定
  - 「Message」にはOpenAPI定義ファイル内のスキーマ・モデルへのJSONポインターを指定
    - ✓ 例: `"/paths/~1customers/get/responses/200/content/application~1json/schema"`
- ◆ ファイルを分割してスキーマを管理する場合は、「Physical format」にOpenAPIのバージョンを指定
  - 設定可能な値は「Swagger 2.0」または「OpenAPI 3.0」

# OpenAPI 3.0のサポート

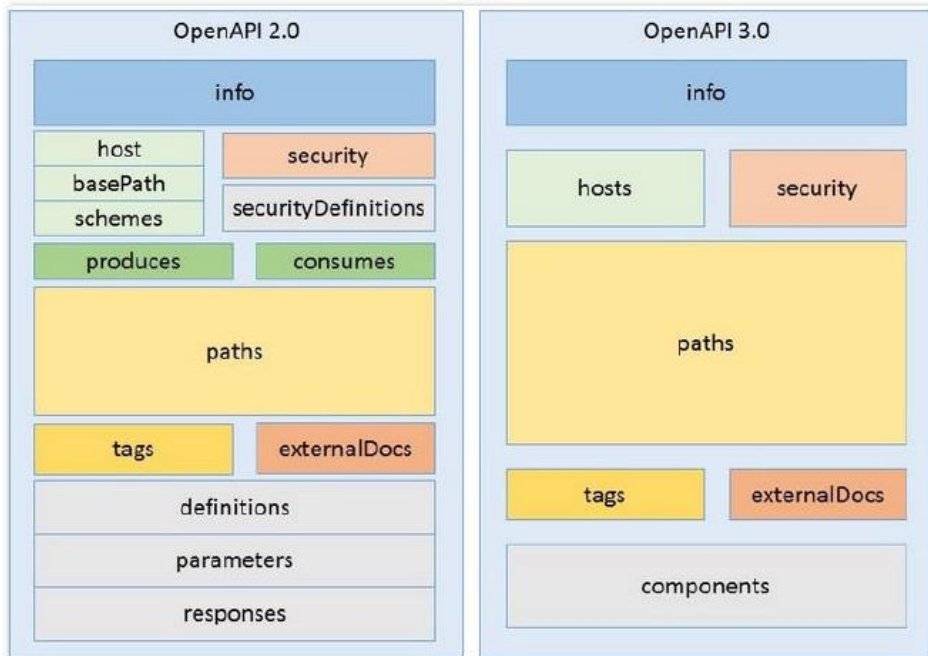
# OpenAPI 3.0 のサポート

- OpenAPI 定義とはREST APIのインターフェース仕様を記述するもの
  - ◆ V10からOpenAPI 2.0 (Swagger 2.0) をサポート
  - ◆ V12ではOpenAPI 3.0のサポートを追加
    - ・ 3.0ではAPI仕様をよりシンプルに記述できるようになり、再利用性も向上
- OpenAPI 3.0 はインバウンドのリクエストでのみサポートされる
  - ◆ RESTRequestやRestAsyncRequest/Responseノードではサポートされない
- 制限事項
  - ◆ 外部参照を含むOpenAPI 3.0定義のインポートや実装は可能だが、ACEのOpenAPIエディタを使用した編集は不可
  - ◆ OpenAPI 3.0の仕様のいくつかはOpenAPIエディタでサポートされないため、ソース・ビューで直接ファイルの編集が必要
  - ◆ その他詳細については以下のリンクを参照
    - ・ <https://www.ibm.com/docs/en/app-connect/12.0?topic=apis-restrictions-openapi-30-documents>



# (参考) OpenAPI 2.0と3.0の主な違い

- API仕様の記述がよりシンプルになり、使いやすさが向上
  - ◆ securityDefinitions/definitions/parameters/responses/といったオブジェクトをcomponentsに統合
- JSONスキーマのoneOf,allOf,anyOfのサポート
- bodyとformDataがrequestBodyに置き換えられ、パラメータ記述が改善



# OpenAPI 3.0を使用したREST API定義

- REST API定義エディタでOpenAPI 定義ファイルを読み込み、定義を生成

The screenshot displays the OpenAPI Editor interface for a REST API. The main window is titled '\*HR\_Services\_openAPI' and contains several sections:

- API Details:** A table with the following information:

Property	Value
REST API base URL	/HR_Services_openAPI
Title	HR Employee and Department Services
Description	HR REST OpenAPI document for the Employee and Department Services used by the ACE eXp (
API version	3.0.0
- Resources and Operations:** A section for defining API endpoints. It shows a resource path '/employees' with a 'GET' operation to 'retrieve EMPLOYEE data'. An 'Edit subflow' button is present next to the operation.
- Error Handling:** A section for defining error handling logic. It includes three handler types: 'Catch handler', 'Failure handler', and 'Timeout handler'. Each handler has a description and a 'Create subflow' button.
- Security:** A section for defining security settings. It includes a checkbox for 'Enable HTTPS' and a note: 'You can enable other security settings in the BAR file editor.'
- Message Validation:** A section for defining message validation rules.

OpenAPI定義ファイルに定義されたリソースとオペレーションに対し、「Edit subflow」でフローを実装

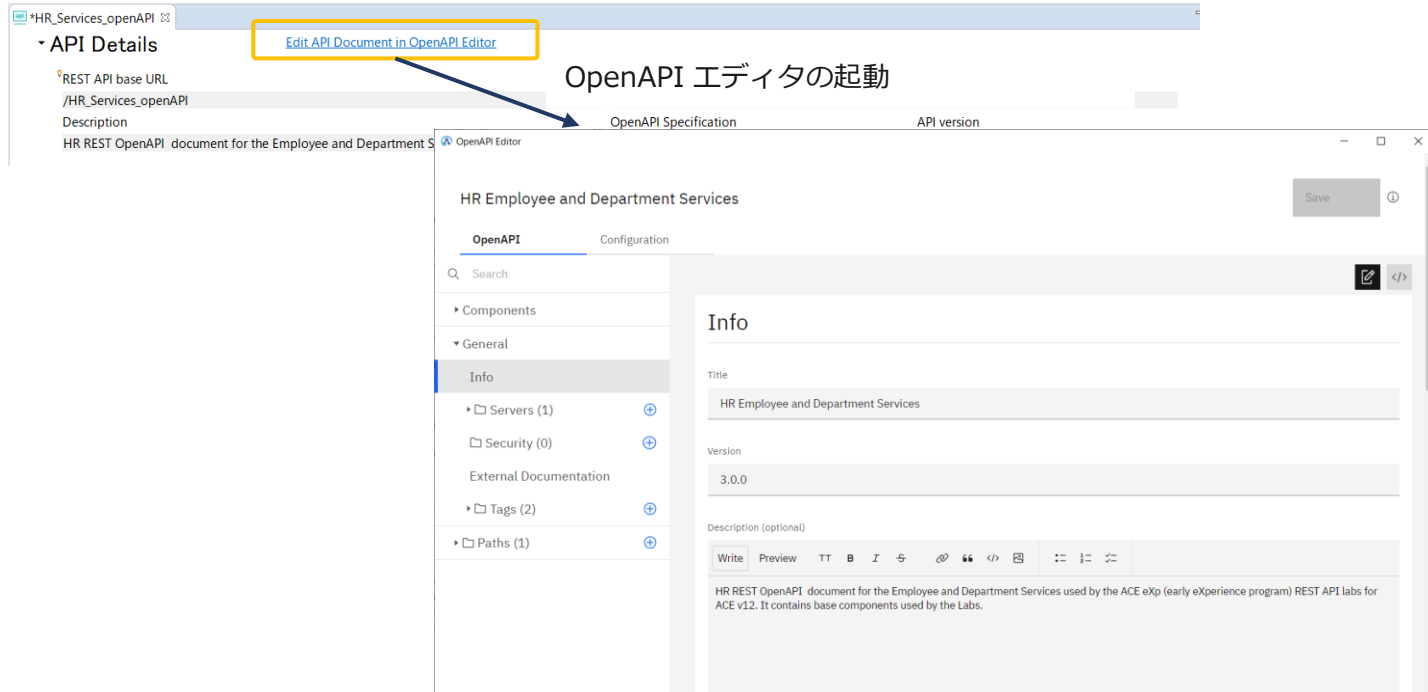
エラー処理の実装

HTTP/HTTPSの設定

メッセージの妥当性検査の有無を指定

# OpenAPI エディタ

- REST API定義エディタからOpenAPI エディタを起動可能
  - ◆ OpenAPI 3.0定義ファイルの参照、編集が可能
  - ◆ DesignerやAPI Connectと共通の独立したエディタが起動
  - ◆ OpenAPI エディタが起動すると、REST APIエディタは編集不可の状態になる（参照のみ）



# OpenAPI エディタ

- ◆ フォーム・ビューとソース・ビュー
  - OpenAPIエディタでサポートされないOpenAPI 3.0の仕様については、ソース・ビューで直接ファイルを編集

The image displays two instances of the OpenAPI Editor interface. The top instance is in 'Form View', showing a configuration page for 'HR Employee and Department Services' with fields for Title, Version (3.0.0), and Description. A yellow box highlights the 'Form View' icon in the top right corner, with an arrow pointing to it from the label 'フォーム・ビュー'. The bottom instance is in 'Source View', showing the same configuration page but with the OpenAPI specification code visible in a dark-themed editor. A yellow box highlights the 'Source View' icon in the top right corner, with an arrow pointing to it from the label 'ソース・ビュー'. Both screenshots show the 'Info' tab selected in the left sidebar, displaying details for 'HR Employee and Department Services' version 3.0.0.

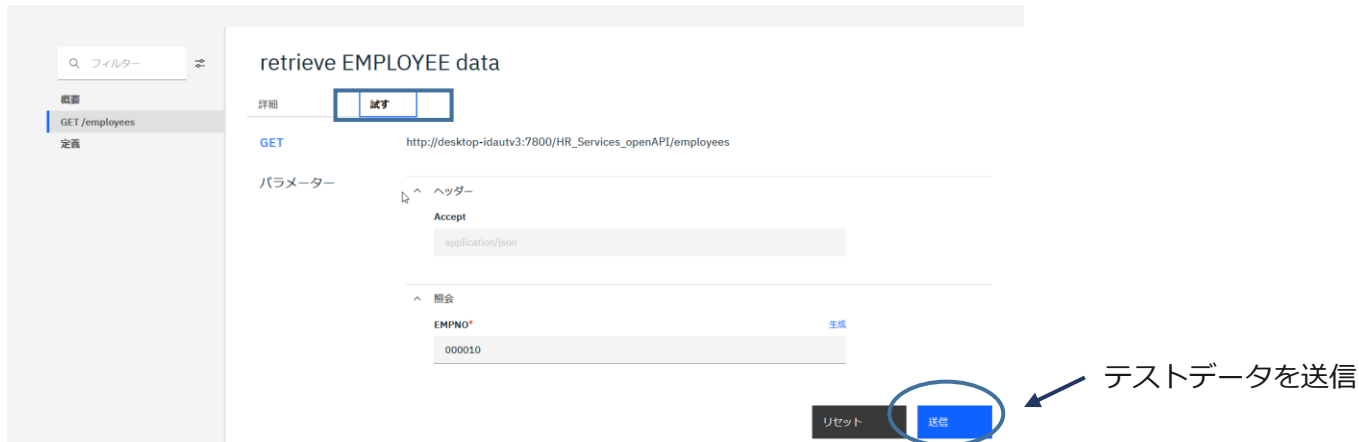
# WebUIを使用したREST APIのテスト

- WebUI からデプロイしたRESTアプリケーションのテストが可能
  - ◆ 統合サーバ上にデプロイされたREST APIアプリケーションをクリック
  - ◆ 「詳細」では呼び出しに必要なパラメーターや要求/応答のサンプルメッセージが確認可能

The screenshot displays the IBM App Connect WebUI interface for testing a REST API. The main content area shows the endpoint `http://desktop-ida37800/HR_Services_openAPI/employees` and the `GET` method. A yellow box highlights the `パラメーター` (Parameters) section, which lists the `EMPNO` parameter as a required string with a length of 6. Another yellow box highlights the `要求の例` (Request Example) section, which shows a `curl` command: `curl --request GET \ --url 'http://desktop-ida37800/HR_Services_openAPI/employees?EMPNO=REPLACE_THIS_VALUE' \ --header 'accept: application/json'`. A third yellow box highlights the `応答` (Response) section, which shows a `200 OK` status and a JSON response body: `{ "DSRResp": { "UserReturnCode": 15075264, "RowsRetrieved": 16905874, "RowsAdded": 42466149, "RowsUpdated": 44994028, "RowsDeleted": 15990886, "SQLCODE_Errorcode": 30137095, "..." } }`. The interface also includes a search filter, tabs for '概要' (Overview), '詳細' (Details), and '試す' (Test), and a '開始済み' (Completed) indicator.

# WebUIを使用したREST APIのテスト

- ◆ 「試す」では実際にRESTアプリケーションのテストが可能



The screenshot shows a REST client interface for testing a GET request. The main area displays the endpoint `http://desktop-ida0tv3:7800/HR_Services_openAPI/employees` and the request parameters, including the header `Accept: application/json` and the parameter `EMPNO: 000010`. A blue circle highlights the `送信` (Send) button, with an arrow pointing to it and the text `テストデータを送信` (Send test data).

- ◆ Web UIはCORSを使ってREST APIにアクセス
  - この機能を使用するには`server.conf.yaml`の`CORSEnabled`を`true`に設定する必要がある

## HTTPConnector:

```
#ListenerPort: 0
#ListenerAddress: '0.0.0.0'
#AutoRespondToHTTPHEADRequests: false
#ServerName: ''
```

**CORSEnabled: true**

# Set the value to true to make the listener respond to valid HTTP CORS requests

:

新しいテストツールの提供

# 新しいテストツール

## ■ 新しいテストツールを提供

- ◆ 従来のFlow Exerciserはメッセージ・フロー全体のテスト
  - Inputノードにメッセージを投入し、Outputノードの結果を確認する
  - テストするための環境の準備が必要（MQ構成の定義、データベースの準備・・・）
- ◆ 新しいテストツール（テストケース）では任意の単位でテストが可能
  - ノード単体、フローの一部、フロー全体など

### <テストケースの作成ウィザード>

Create test case

Use this wizard to create a test case for a message flow. Choose an input and output file to create input and output message assemblies. An input message assembly is used to invoke the node in the test case. An output message assembly is used to compare the expected and actual output message.

Node location:

Test Project: TEST01\_Test  
Application: TEST01  
Flow: FLO2  
Node: SelectInfo

Input terminal: In  
Output terminal: Out

Select an input file to construct a message assembly to invoke the node.  
Browse...

Select an output file to compare against the actual output message.  
Browse...

Select matchers to be included in the generated test:  
 Assert node call count  Assert terminal propagate count

Select message trees to compare in the generated test:  
 Message body  Environment  
 Local environment  Exception list

Select setup and cleanup methods  
 BeforeAll  AfterAll

Read-only preview for generated test case:

```
/**  
 * TEST01_FL02_SelectInfo_0001_Test  
 * Test generated by IBM App Connect Enterprise Toolkit 12.0.1.0 on 2021/08/18 11:41:50  
 */  
  
@AfterEach  
public void cleanupTest() throws TestException {  
    // Ensure any mocks created by a test are cleared after the test runs  
    TestSetup.restoreAllMocks();  
}  
  
@Test  
public void TEST01_FL02_SelectInfo_TestCase_001() throws TestException {  
  
    // Define the SpyObjectReference  
    SpyObjectReference nodeReference = new SpyObjectReference() application("TEST01").messageFlow("FLO2").node("Se  
  
    // Initialise a NodeSpy  
    NodeSpy nodeSpy = new NodeSpy(nodeReference);  
  
    // Check that the actual Node name matches the expected name  
    assertEquals("SelectInfo", nodeReference.getName());  
  
    // Assert the number of times that the node is called  
    assertEquals(0, nodeSpy.getCallCount());  
  
    // Assert the terminal propagate count for the message  
    assertEquals("failure", 0, nodeSpy.getTerminalPropagateCount());  
    assertEquals("out", 0, nodeSpy.getTerminalPropagateCount());  
    assertEquals("out1", 0, nodeSpy.getTerminalPropagateCount());  
}
```

### <テストケースの実行結果確認 (JUnitビュー)>

Application Development JUnit

Finished after 1.954 seconds

Runs: 1/1 Errors: 0 Failures: 0

TEST01\_FL02\_SelectInfo\_0001\_Test [Runner: JUnit 5] (1.450 s)  
TEST01\_FL02\_SelectInfo\_TestCase\_001() (1.450 s)

Failure Trace



# 新しいテストツール

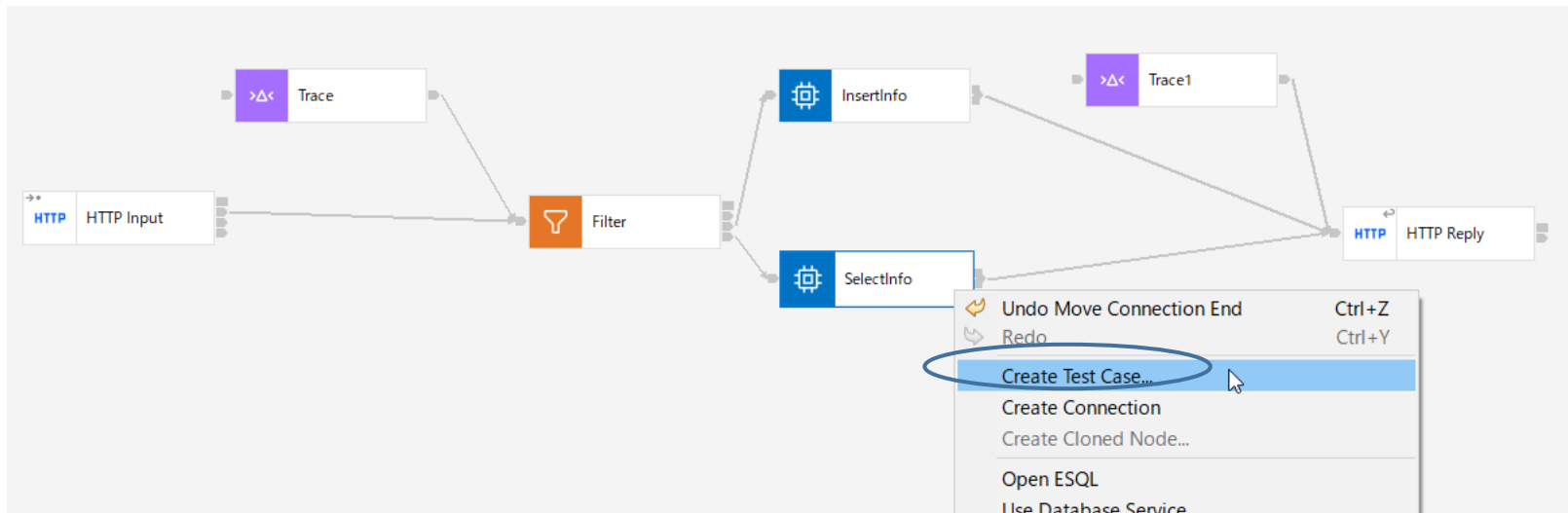
- テストケースはJavaで生成
  - ◆ 製品同梱のIntegrationTest.jarでテストに必要なJavaクラスを提供
  - ◆ 任意のテキスト・エディター、ソース・コード・エディタ、ツールキットなどのJava IDEを使用して作成/変更
  - ◆ テスト・プロジェクトに保管される
- ツールキットから「テストケースの作成」ウィザードを起動し、簡単に生成することも可能
- `ibmint generate tests`コマンドからの生成も可能
- テストはJUnitテスト独自のスレッドで実行
  - ◆ メッセージ・フローのスレッドとは異なるため、テスト対象のメッセージ・フローが統合サーバ上で稼働している必要はない
    - 実際のフローが実行されないように、メッセージ・フローは停止
  - ◆ デプロイされていることが前提

# 新しいテストツール

- 2つのタイプのテストダブル (Test Double) が利用可能
  - ◆ Node Spy (Mock)
    - ノードの処理結果を確認し、検証
    - 通常のテストケースではこちらを使用
  - ◆ Node Stub (Fake)
    - 特定の処理をスキップさせたり、代替処理を呼び出したりする
- テストの起動や結果の比較のためのメッセージを用意
  - ◆ Message AssemblyやXMLまたはJSONデータ・ファイルなど
- テストを実行するとMatcherがテストの成功/失敗を判断
  - ◆ Matcherの検証内容
    - Nodeの呼び出し回数
    - ターミナルから出力された回数
    - 実出力メッセージと期待値との比較
    - 例外のチェック
  - ◆ matcherは以下をサポート
    - Junitアサーション
    - 3rd Partyのmatcherライブラリ (Hamcrest、JSONAssert、XMLUnitなど)

# Toolkitを使用したテストケースの生成

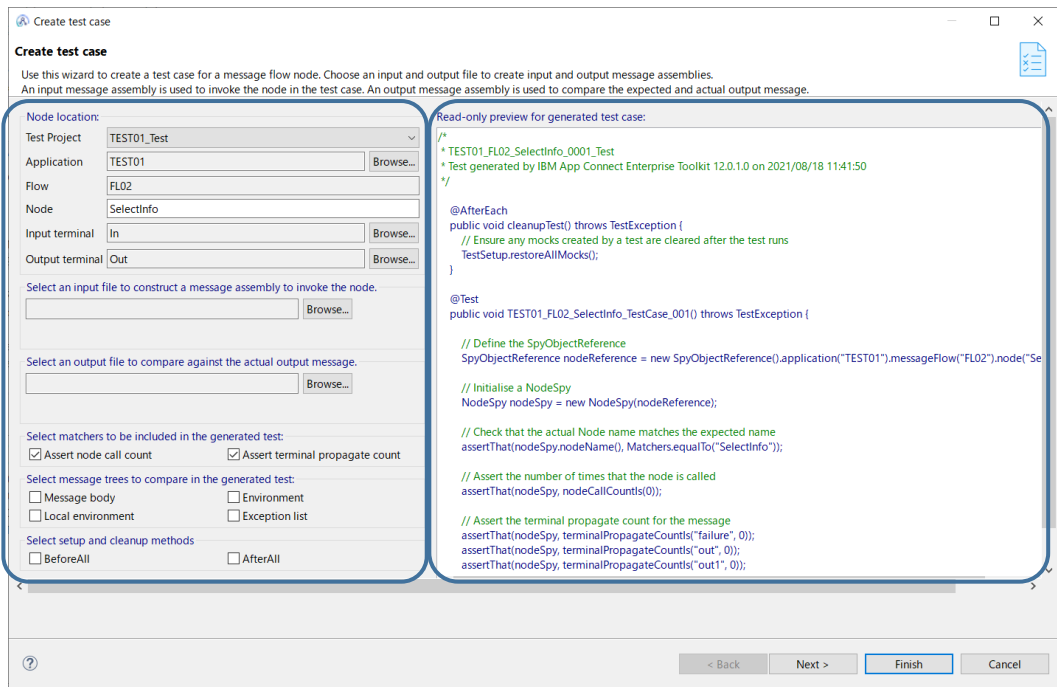
- ツールキットからテストケース・ウィザードを起動
  - ◆ テスト対象のノードを右クリック→「Create Test Case…」



# Toolkitを使用したテストケースの生成

## ■ テストケース・ウィザード

- ◆ 設定可能なオプションが左側のペインに表示される
- ◆ オプションの設定条件に応じてテストケース（Javaコード）が生成され、右側のペインに表示される
  - Javaコードはここでは編集不可で、参照のみ



# Toolkitを使用したテストケースの生成

初回テストケース作成時に  
テストプロジェクトを生成  
その他、テスト対象のノード  
に関連した情報が含まれる

テストの実行に使用するInput  
メッセージや、実行結果を確  
認するための期待値となる  
Outputメッセージを指定可能  
(Recordingしたメッセージも  
指定可能)

使用するmatchersの指定

結果比較するMessage  
Assemblyツリーの指定

テストの実施前後処理をテスト  
ケースに追加するかどうか  
を指定

**Create test case**

Use this wizard to create a test case for a message flow node. Choose an input and output file to create input and output message assemblies. An input message assembly is used to invoke the node in the test case. An output message assembly is used to compare the expected and actual output message.

Read-only preview for generated test case:

```
/*  
 * TEST01_FL02_SelectInfo_0001_Test  
 * Test generated by IBM App Connect Enterprise Toolkit 12.0.1.0 on 2021/08/18 11:41:50  
 */  
  
@AfterEach  
public void cleanupTest() throws TestException {  
    // Ensure any mocks created by a test are cleared after the test runs  
    TestSetup.restoreAllMocks();  
}  
  
@Test  
public void TEST01_FL02_SelectInfo_TestCase_001() throws TestException {  
  
    // Define the SpyObjectReference  
    SpyObjectReference nodeReference = new SpyObjectReference().application("TEST01").messageFlow("FL02").node("Se  
  
    // Initialise a NodeSpy  
    NodeSpy nodeSpy = new NodeSpy(nodeReference);  
  
    // Check that the actual Node name matches the expected name  
    assertThat(nodeSpy.nodeName(), Matchers.equalTo("SelectInfo"));  
  
    // Assert the number of times that the node is called  
    assertThat(nodeSpy, nodeCallCounts(0));  
  
    // Assert the terminal propagate count for the message  
    assertThat(nodeSpy, terminalPropagateCounts("failure", 0));  
    assertThat(nodeSpy, terminalPropagateCounts("out", 0));  
    assertThat(nodeSpy, terminalPropagateCounts("out1", 0));  
}
```

# Toolkitを使用したテストケースの生成

- ウィザードを終了すると生成されたテストケースがJava エディタで表示される

テスト・プロジェクトが生成され、その下にJavaコードが格納

テストで使用するメッセージがresourcesフォルダに

The screenshot shows an IDE window with the following components:

- Project Explorer (Left):** Displays a project structure with folders like QA01, TEST01, and TEST01\_Test. A blue box highlights the 'test' folder containing the file 'TEST01\_FL02\_SelectInfo\_0001\_Test.java'. Below it, the 'resources' folder contains 'FL02\_SelectInfo\_Input.xml' and 'FL02\_SelectInfo\_Output.xml'.
- Code Editor (Right):** Shows the content of 'TEST01\_FL02\_SelectInfo\_0001\_Test.java'. The code includes package declarations, imports, and test methods like 'cleanupTest()' and 'TEST01\_FL02\_SelectInfo\_TestCase\_001()'. A blue arrow points to the editor with the text 'Javaエディターで編集可能'.
- Properties View (Bottom):** Shows a table of properties for the selected file.

Property	Value
derived	false
editable	true
last modified	2021年8月18日 12:07:44
linked	false
location	C:\Users\AKIKONaruse\IBM\ACET12\workspace\TEST01_Test\src\main\java

← Javaエディターで編集可能

# Toolkitを使用したテストケースの生成

## ■ 生成されたテストケース例

```
package test;

import java.io.InputStream;

import org.junit.jupiter.api.AfterEach;
import org.junit.jupiter.api.Test;

import com.ibm.integration.test.v1.NodeSpy;
import com.ibm.integration.test.v1.SpyObjectReference;
import com.ibm.integration.test.v1.TestMessageAssembly;
import com.ibm.integration.test.v1.TestSetup;
import com.ibm.integration.test.v1.exception.TestException;

import static com.ibm.integration.test.v1.Matchers.*;
import static org.hamcrest.MatcherAssert.assertThat;

public class TEST01_FL02_SelectInfo_0001_Test {

    /*
     * TEST01_FL02_SelectInfo_0001_Test
     * Test generated by IBM App Connect Enterprise Toolkit 12.0.1.0 on 2021/08/18 11:51:47
     */

    @AfterEach
    public void cleanupTest() throws TestException {
        // Ensure any mocks created by a test are cleared after the test runs
        TestSetup.restoreAllMocks();
    }
}
```

# Toolkitを使用したテストケースの生成

```
@Test
public void TEST01_FL02_SelectInfo_TestCase_001() throws TestException {

    // Define the SpyObjectReference
    SpyObjectReference nodeReference = new SpyObjectReference().application("TEST01").messageFlow("FL02").node("SelectInfo");

    // Initialise a NodeSpy
    NodeSpy nodeSpy = new NodeSpy(nodeReference);

    // Declare a new TestMessageAssembly object for the message being sent into the node
    TestMessageAssembly inputMessageAssembly = new TestMessageAssembly();

    // Create a Message Assembly from the input data file
    try {
        String messageAssemblyPath = "/FL02_SelectInfo_Input.mxml";
        InputStream messageStream = Thread.currentThread().getContextClassLoader()
            .getResourceAsStream(messageAssemblyPath);
        if (messageStream == null) {
            throw new TestException("Unable to locate message assembly file: " + messageAssemblyPath);
        }
        inputMessageAssembly.buildFromRecordedMessageAssembly(messageStream);
    } catch (Exception ex) {
        throw new TestException("Failed to load input message", ex);
    }

    // Call the message flow node with the Message Assembly
    nodeSpy.evaluate(inputMessageAssembly, true, "in");

    // Assert the number of times that the node is called
    assertThat(nodeSpy, nodeCallCountIs(1));

    // Assert the terminal propagate count for the message
    assertThat(nodeSpy, terminalPropagateCountIs("out", 1));
}
```

SpyObjectReferenceはMockを生成するときのオブジェクト。  
テストするノードへのリファレンス

テストするノードで  
NodeSpy生成

ウィザードで指定した  
入力メッセージがセット

入力メッセージを使用して  
テストを実行

NodeSpyがノードのふるまいを観察。  
入力メッセージのPUTや呼び出し回数、  
出カターミナルなど



# Toolkitを使用したテストケースの生成

```
/* Compare Output Message 1 at output terminal out */
try {
    TestMessageAssembly actualMessageAssembly = null;
    TestMessageAssembly expectedMessageAssembly = null;

    // Get the TestMessageAssembly object for the actual propagated message
    actualMessageAssembly = nodeSpy.propagatedMessageAssembly("out", 1);

    // Assert output message body data
    // Get the TestMessageAssembly object for the expected propagated message
    expectedMessageAssembly = new TestMessageAssembly();

    // Create a Message Assembly from the expected output mxml resource
    String messageAssemblyPath = "/FL02_SelectInfo_Output.mxml";
    InputStream messageStream = Thread.currentThread().getContextClassLoader()
        .getResourceAsStream(messageAssemblyPath);
    if (messageStream == null) {
        throw new TestException("Unable to locate message assembly file: " + messageAssemblyPath);
    }
    expectedMessageAssembly.buildFromRecordedMessageAssembly(messageStream);

    // Assert that the actual message tree matches the expected message tree
    assertThat(actualMessageAssembly, equalsMessage(expectedMessageAssembly));
} catch (Exception ex) {
    throw new TestException("Failed to compare with expected message", ex);
}
}
```

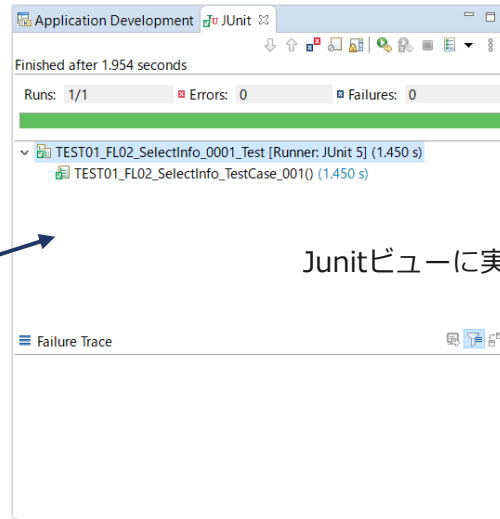
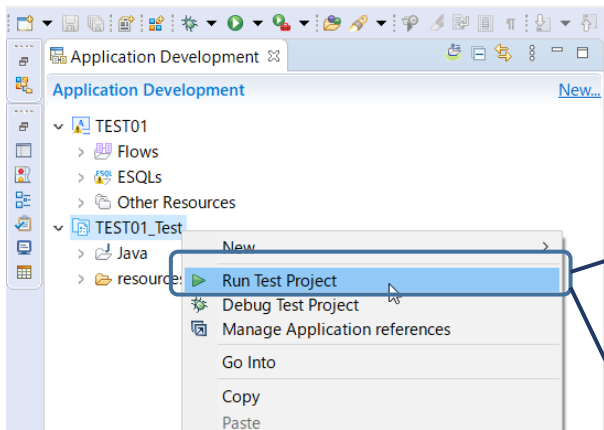
ノードからの出力  
メッセージを取得

ウィザードで指定した出力  
メッセージがセット

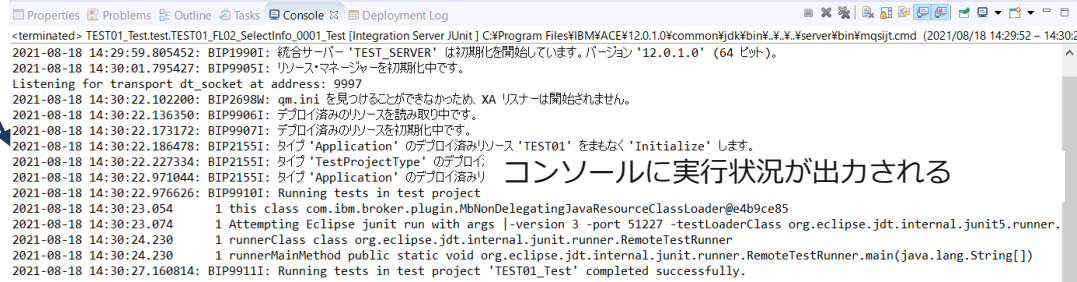
ウィザードで出力メッセ  
ージを指定すると、自動で期  
待値との比較が有効になる

# テストケースの実行

- テストプロジェクトを右クリックし「Run Test Project」で実行



Junitビューに実行結果が表示



コンソールに実行状況が出力される

# テストケースの実行

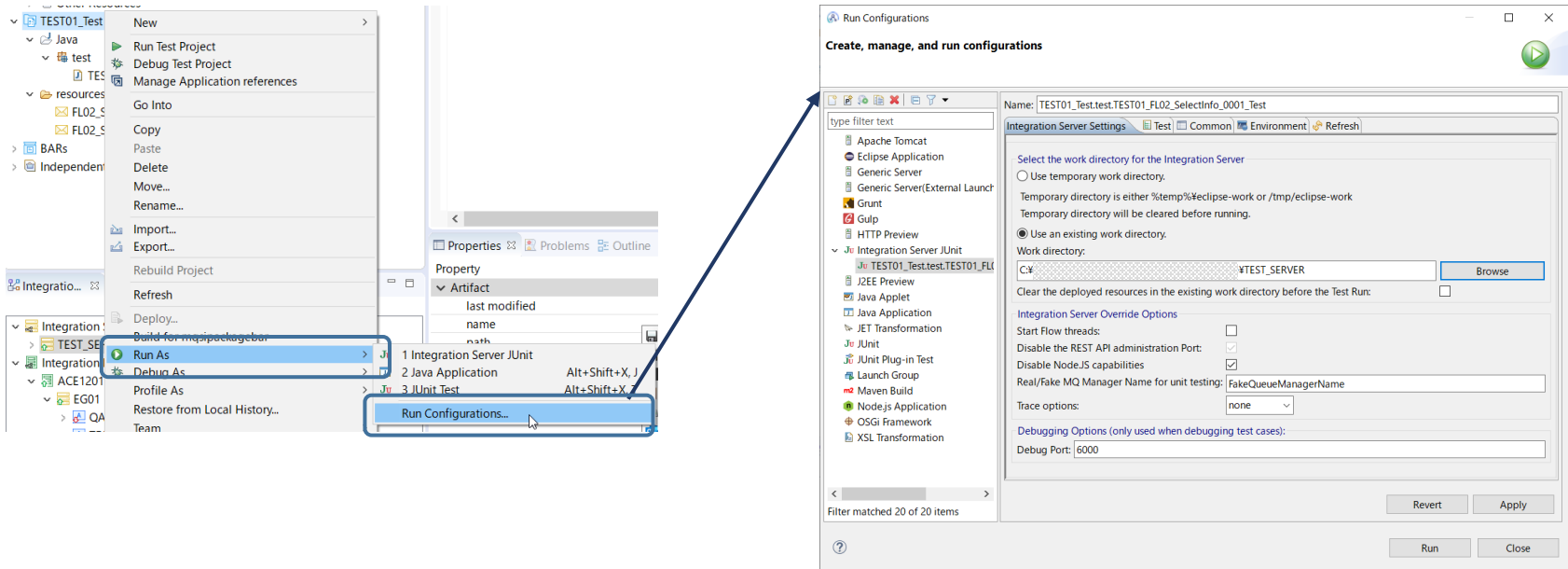
## ■ エラー発生時はFailure TraceやStack Traceから調査

The screenshot shows an IDE window with the following components:

- JUnit Results:** Shows a test run that finished after 1.784 seconds with 1 failure. The failed test is `TEST01_FL02_SelectInfo_0001_Test`.
- Failure Trace:** Displays the error message: `java.lang.AssertionError: Expected: TestMessageAssemblies should have matching message trees. but: Expected value '002' but was '003' (path: /Root/JSON/Data/refEMPLOYEE/PKEY) Expected value 'Taro' but was 'Taro' (path: /Root/JSON/Data/refEMPLOYEE/CCODE) Expected value 'AA000002' but was 'AA000003' (path: /Root/JSON/Data/refEMPLOYEE/CCODE)`. A blue box highlights the error message, and a callout box explains: **エラー原因の出力 (今回は実行結果と期待値との比較でエラーになった例)**.
- Stack Trace:** Shows the call stack starting from `org.hamcrest.MatcherAssert.assertThat`. A blue box highlights the stack trace, and a callout box explains: **Stack Traceの起動** (with a sub-callout **Show Stack Trace in Console View** pointing to the icon).

# テストケースの実行

- デフォルトでテスト用にeclipse-workという名前の独立統合サーバを一時的に生成して使用（変更可能）
- テスト実行時の構成を変更するにはテストプロジェクトを右クリック→「Run As」→「Run Configurations…」を選択し、構成をカスタマイズ



# テストケースの実行

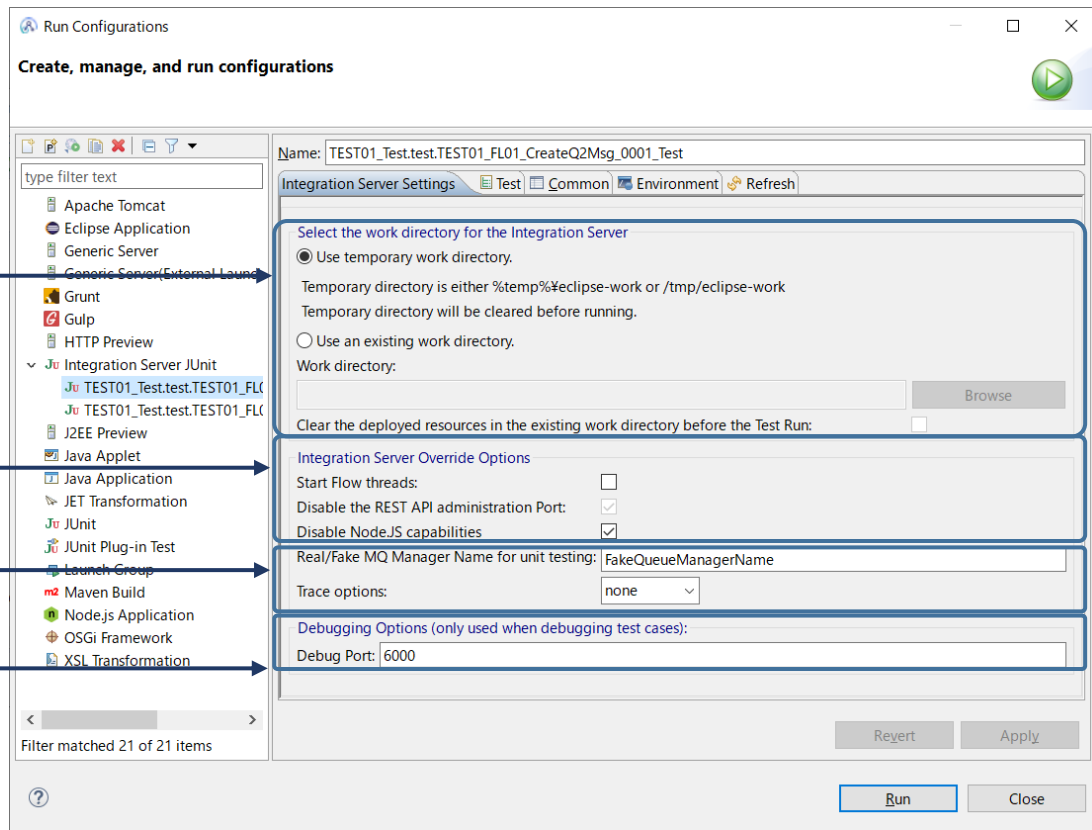
## ■ テスト実行時の構成をカスタマイズ

テストの実行にテンポラリーの独立統合サーバを使用するか、既存の統合サーバを使用するかを指定可能

パフォーマンスの観点から、テスト実行前のフロー起動、REST API Administration PortやNode.JS機能は無効化されている

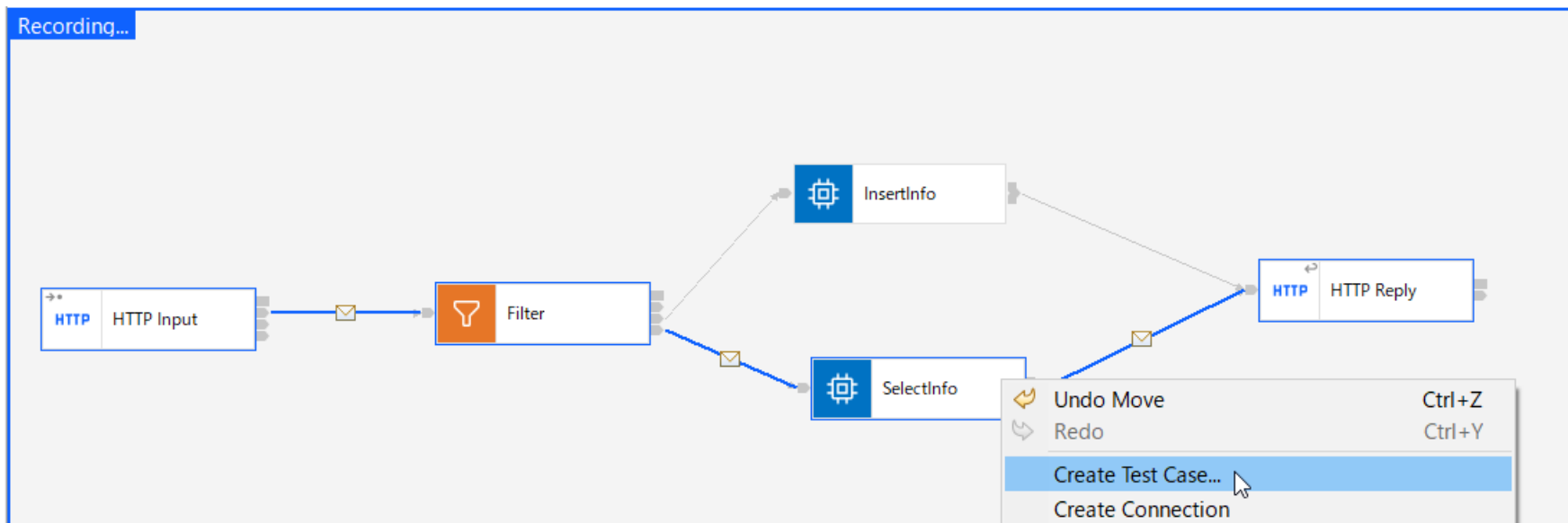
MQフローのテスト時にテスト用の疑似キュー・マネージャーを使用するか、トレースを取得するかの設定

デバッガー使用時のポート指定



# (参考)

- フロー・エクササイザーでRecordingした状態で「Create Test Case」を実行すると、レコードされたノードの入出カメッセージを使用してテストケースを生成



# ibmintコマンドを使用したテストケースの生成

- `ibmint generate tests`コマンドからテストプロジェクトとテストケースを作成
  - ◆ 事前に取得したレコードメッセージを使用してテストケースを生成
    - 引数に指定したレコードメッセージ保管ディレクトリから、ノードに対する入出力メッセージを検出し、ノードのテストケースを自動生成

## コマンド構文

```
ibmint generate tests
--recorded-messages      ... テストで使用するレコードメッセージの保管ディレクトリ名を指定
--output-test-project    ... 生成するテストプロジェクトの名前と保管ディレクトリを指定
--java-class             ... 生成されたテストを含む Java クラスの完全修飾クラスパス
```

## コマンド実行例

```
C:¥z_Test¥ACE12¥testcase>ibmint generate tests
--recorded-message C:¥z_Test¥ACE12¥testcase¥recordedMessage
--output-test-project C:¥z_Test¥ACE12¥testcase¥cmdGenTestProject
--java-class com.ibm.cmdtest01tes.CmdFL02TestClass
```

```
BIP15215I: Found '2' recorded messages in directory 'C:¥z_Test¥ACE12¥testcase¥recordedMessage'.
```

```
BIP15216I: Generating test project from recorded messages.
```

```
BIP15183I: '1' テスト・ケースを含むテスト・スイートは、正常に生成され、
'C:¥z_Test¥ACE12¥testcase¥cmdGenTestProject' に書き込まれました。
```

```
BIP8071I: コマンドは正常終了しました。
```

# ibmintコマンドを使用したテストケースの生成

- ◆ 指定したテストプロジェクト用のディレクトリにテストケースとテストで使用するレコードメッセージファイルが生成される
- ◆ 生成されたテストプロジェクトはツールキットにインポートすることも可能

前頁のコマンド実行例の場合、C:¥z\_Test¥ACE12¥testcase¥cmdGenTestProject配下に以下のファイルが生成される

The screenshot displays a file explorer window showing the directory structure of a generated test project. The main view shows a 'src' folder containing 'build.gradle', 'settings.gradle', and 'testproject.descriptor'. A zoomed-in view of the 'src/main/resources' directory shows two XML files: 'FL02\_SelectInfo\_Input.mxml' and 'FL02\_SelectInfo\_Output.mxml', both dated 2021/08/26 11:16.

名前	更新日時	種類	サイズ
<input checked="" type="checkbox"/> src	2021/08/26 11:16	ファイル フォルダ	
<input type="checkbox"/> build.gradle		GRADLE ファイル	2 KB
<input type="checkbox"/> settings.gradle		GRADLE ファイル	2 KB
<input type="checkbox"/> testproject.descriptor		GRADLE ファイル	2 KB

名前	更新日時
<input type="checkbox"/> java	2021/08/26 11:16
<input type="checkbox"/> resources	2021/08/26 11:16

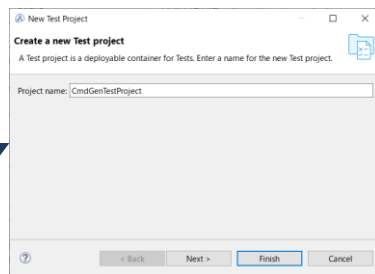
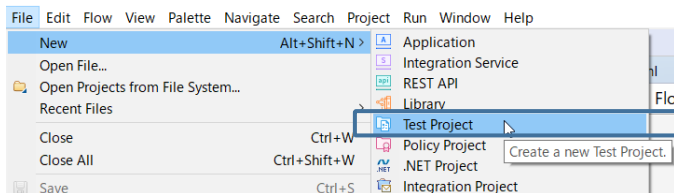
名前	更新日時
<input type="checkbox"/> FL02_SelectInfo_Input.mxml	2021/08/26 11:16
<input type="checkbox"/> FL02_SelectInfo_Output.mxml	2021/08/26 11:16



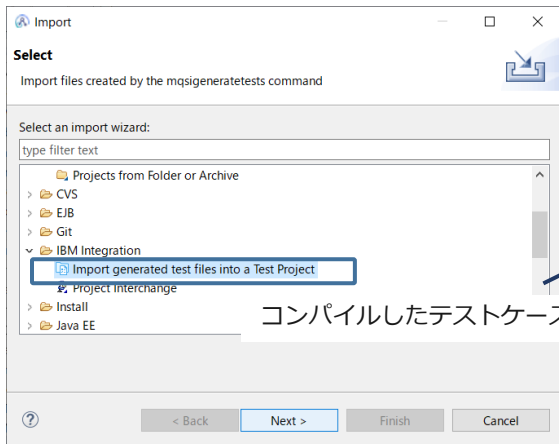
# ibmintコマンドを使用したテストケースの生成

## ■ 生成したテストケースをツールキットにインポート可能

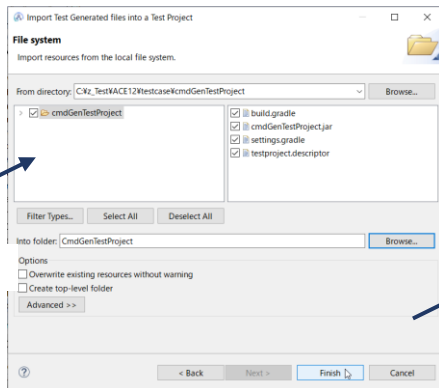
### ◆ 事前にテスト・プロジェクトを生成



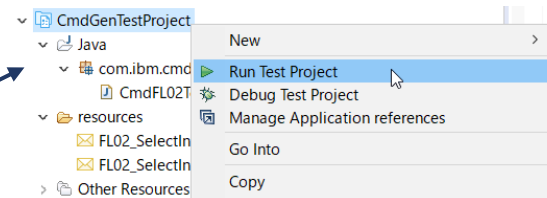
### ◆ 作成したテスト・プロジェクトにテストファイルをインポートする



コンパイルしたテストケースを指定



(インポートしたテストケースの実行)



# ibmintコマンドを使用したテストケースの生成

- ibmintコマンドで生成したテストケースをgradleコマンドでコンパイル
  - ◆ gradleはACEでは提供されないため、別途インストールが必要
    - <https://gradle.org/install/>

## コマンド実行例

```
C:¥z_Test¥ACE12¥testcase¥cmdGenTestProject>gradle
Starting a Gradle Daemon (subsequent builds will be faster)
```

```
BUILD SUCCESSFUL in 52s
3 actionable tasks: 3 executed
```

## コマンド実行前のディレクトリ

```
C:¥z_Test¥ACE12¥testcase¥cmdGenTestProject のディレクトリ
```

```
2021/08/26 11:16 <DIR>      .
2021/08/26 11:16 <DIR>      ..
2021/08/26 11:16          1,089 build.gradle
2021/08/26 11:16          38 settings.gradle
2021/08/26 11:16 <DIR>      src
2021/08/26 11:16          332 testproject.descriptor
```



## コマンド実行後のディレクトリ

```
C:¥z_Test¥ACE12¥testcase¥cmdGenTestProject のディレクトリ
```

```
2021/08/26 11:53 <DIR>      .
2021/08/26 11:53 <DIR>      ..
2021/08/26 11:53 <DIR>      .gradle
2021/08/26 11:53 <DIR>      build
2021/08/26 11:16          1,089 build.gradle
2021/08/26 11:53          3,610 cmdGenTestProject.jar
2021/08/26 11:53 <DIR>      resources
2021/08/26 11:16          38 settings.gradle
2021/08/26 11:16 <DIR>      src
2021/08/26 11:16          332 testproject.descriptor
```

# ibmintコマンドを使用したテストケースの生成

## ■ gradleコマンドでコンパイルしたらそのまま実行可能

- ◆ 統合サーバのワークディレクトリ下のrunディレクトリにテストプロジェクトをコピーして、実行！
  - ・ 事前にテスト対象のアプリケーションはデプロイしておく

コマンド実行例

```
C:¥z_Test¥ACE12¥testcase>xcopy /e cmdGenTestProject C:¥z_Test¥TEST_SERVER¥run¥cmdGenTestProject
C:¥z_Test¥ACE12¥testcase> C:¥z_Test¥ACE12¥testcase>IntegrationServer --work-dir C:¥z_Test¥TEST_SERVER --test-project cmdGenTestProject
--start-msgflows false
```

```
2021-08-30 16:52:02.480531: BIP1990I: 統合サーバー 'TEST_SERVER' は初期化を開始しています。バージョン '12.0.1.0' (64 ビット)。
2021-08-30 16:52:02.523185: BIP9905I: リソース・マネージャーを初期化中です。
Listening for transport dt_socket at address: 9997
2021-08-30 16:52:41.081636: BIP9906I: デプロイ済みのリソースを読み取り中です。
2021-08-30 16:52:41.245676: BIP9907I: デプロイ済みのリソースを初期化中です。
2021-08-30 16:52:41.264856: BIP2155I: タイプ 'TestProjectType' のデプロイ済みリソース 'cmdGenTestProject' をまもなく 'Initialize' します。
2021-08-30 16:52:41.292660: BIP2155I: タイプ 'Application' のデプロイ済みリソース 'TEST01' をまもなく 'Initialize' します。
2021-08-30 16:52:41.382464: BIP2155I: タイプ 'TestProjectType' のデプロイ済みリソース 'TEST01_Test' をまもなく 'Initialize' します。
2021-08-30 16:52:42.319260: BIP2155I: タイプ 'Application' のデプロイ済みリソース 'TEST01' をまもなく 'Start' します。
2021-08-30 16:52:53.588172: BIP2866I: IBM App Connect Enterprise 管理セキュリティーは inactive です。
2021-08-30 16:52:53.645204: BIP3132I: HTTP リスナーがポート '7600' で 'RestAdmin http' 接続の listen を開始しました。
2021-08-30 16:52:53.660496: BIP9910I: Running tests in test project 'cmdGenTestProject'.
2021-08-30 16:52:55.461    1 STARTING TEST:TEST01_FL02_SelectInfo_TestCase_001()
2021-08-30 16:52:56.965    1 FINISHED TEST:TEST01_FL02_SelectInfo_TestCase_001() SUCCESSFUL
2021-08-30 16:52:57.108    1
```

### TEST RESULTS:

```
com.ibm.cmdtest01tes.CmdFL02TestClass:
TEST01_FL02_SelectInfo_TestCase_001():SUCCESSFUL
```

### TOTALS:

```
PASSED :1
FAILED :0
ABORTED :0
TIME(secs):1.991
```

```
2021-08-30 16:52:57.118160: BIP9911I: Running tests in test project 'cmdGenTestProject' completed successfully.
```

コピー

実行!

テストの実行  
結果

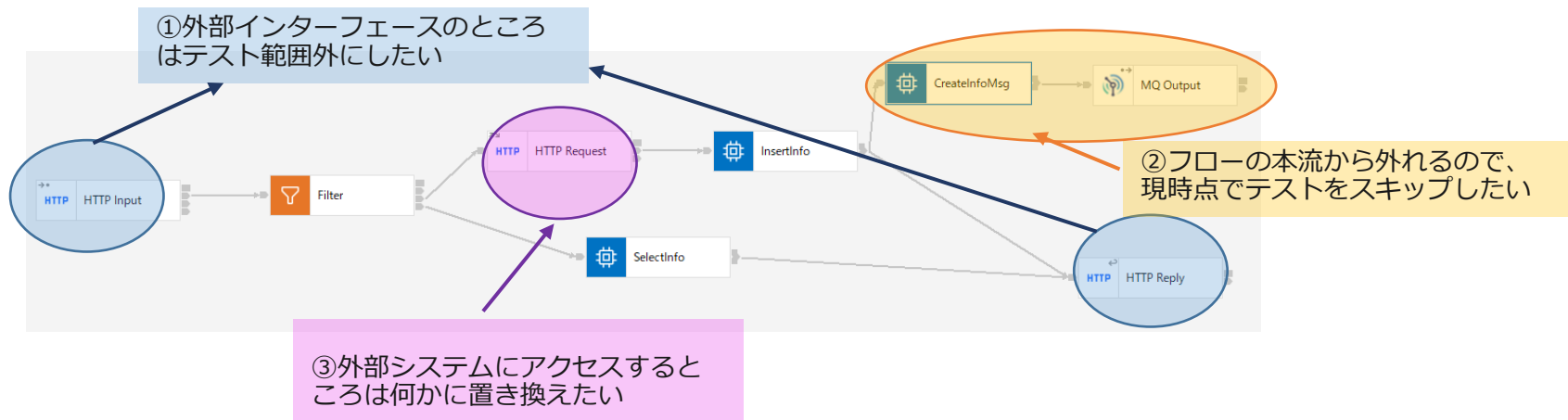
# 一連のメッセージ・フロー・ノードのテスト

## ■ 任意の範囲でテストの実施が可能

- ◆ フローの一部をテストする場合
  - ・ 最初のノードにメッセージを渡して、Stop Pointかフローの最後までテストを実行可能
- ◆ 全フローをテストする場合
  - ・ 普通に本来の外部IFから呼び出す

## ■ テストケースでは柔軟なテストの実施が可能

- ◆ メッセージ・フローを開発&テストする中で少しずつテスト範囲をひろげたいという要件にも対応
- ◆ 特に外部と連携する部分はテストをする前に環境のセットアップが必要となり、作業負荷が高い



# 一連のメッセージ・フロー・ノードのテスト

- evaluate()メソッド、propagate()メソッドを使用して後続の処理へメッセージを伝搬することで、フローの任意の部分のテストが可能

- ◆ evaluate()メソッド

- 一連のメッセージ・フロー・ノードのテストを行う場合は、singleNodeTestをfalseに設定

```
public void evaluate(final TestMessageAssembly testAssembly, boolean singleNodeTest, String inputTerminalName) throws TestException
```

実装例

```
compute1NodeSpy.evaluate(inputMessageAssembly, false, "in");
```

- ◆ propagate()メソッド

- ノードの出カターミナルにメッセージを投入し（ノード自体は呼び出されない）、後続のフローを実行

実装例

```
httpInputNodeSpy.propagate(inputMessageAssembly, "out");
```

# 一連のメッセージ・フロー・ノードのテスト

## ■ テストのエンドポイントの指定が可能

- ◆ `setStopAtInputTerminal()`メソッド、または`setStopAtOutputTerminal()`メソッドで特定のポイントでテストを終了させることが可能

実装例

```
mqOutputNodeSpy.setStopAtInputTerminal("in");
```

`setStopAtInputTerminal()` `setStopAtOutputTerminal()`



## ■ 代替応答の指定が可能

- ◆ 外部リソースにアクセスするような処理をNode Stubに置き換えることが可能
  - 外部リソースにアクセスすることなく、事前に準備した代替応答メッセージを後続ノードに引き渡す

実装例

```
DBCStub.onCall().propagatesMessage("in", "out", DBResultMessageAssembly);
```

(実際のノード処理を実行せずに事前に準備した代替応答 (DBResultMessageAssembly) をoutターミナルに流す)

- ノード処理を実行せずに、入力メッセージをそのまま出力メッセージとして出力ターミナルに流すことも可能

実装例

```
DBCStub.onCall().propagatesInputMessage("in", "out");
```

# 新しいテストツールの注意点

- テスト実行時にフローはデプロイされている必要があるため、単一ノードのテストでも、デプロイできる状態までフローとして実装されている必要がある
- テストを実施するためにテストデータが必要
  - ◆ テスト処理を実行するためのテストデータ
  - ◆ テスト結果を比較するための期待値データ
    - ・ 実装処理が複雑化するほど、より実データに近いテストデータが必要となる
  - ◆ 代替応答のためのテストデータ
- リグレッション・テストなどが容易に
  - ◆ 現行バージョンでテストしたデータを使用してテストケースを作成
  - ◆ アプリの改修や製品のバージョンアップの際に、テストケースを利用することで、テスト工数の削減、既存の動作に影響がないことなどを容易に確認可能

ibmintコマンドの提供



# ibmintコマンドの提供

## ■ 新しいibmintコマンドを提供

- ◆ 既存のコマンドも継続して利用可能
- ◆ コマンドから“mqsi”や、コンポーネント名 (broker / executiongroup)を外し、共通的なコマンド体系に変更

```
C:\Program Files\IBM\ACE\12.0.2.0>ibmint --help  
BIP15186I: Run commands for App Connect Enterprise
```

### User Interaction Commands

- help

### Administration Commands

- create (server | node)
- delete (server | node)
- start (server | node)
- stop (server | node)

### Build Commands

- apply overrides
- compile msgset
- deploy
- generate (tests | msgindex)
- package

For more detailed usage help, use --help with any of the above commands. For example:

```
ibmint create node --help  
ibmint deploy --help
```

# ibmintコマンドの提供

コマンド	説明
ibmint apply overrides	BARファイルの上書き
ibmint compile msgset	MRM メッセージセットのコンパイル
ibmint create node	統合ノードの作成
ibmint delete node	統合ノードの削除 <span>V12.0.2.0以降</span>
ibmint create server	統合サーバの作成
ibmint delete server	統合サーバの削除（独立統合サーバは削除不可）
ibmint deploy	リソースのデプロイ
ibmint generate tests	テストケースの作成
ibmint generate msgindex	レコードメッセージのインデックスの作成 <span>V12.0.2.0以降</span>
ibmint help	ibmintコマンドのヘルプ表示
ibmint package	BARの作成
ibmint start node ibmint stop node	統合ノードの起動/停止
ibmint start server ibmint stop server	統合サーバの起動/停止

# ibmintコマンドの提供

- パラメータは識別できるところまで書けば、推測してくれる

```
ibmint deploy --input-bar-file C:¥Test¥temp¥Demo.bar --output-work-directory C:¥Test¥temp¥MyDemoWorkDirectory  
--overrides-file C:¥Test¥temp¥demopropertyoverrides.txt
```

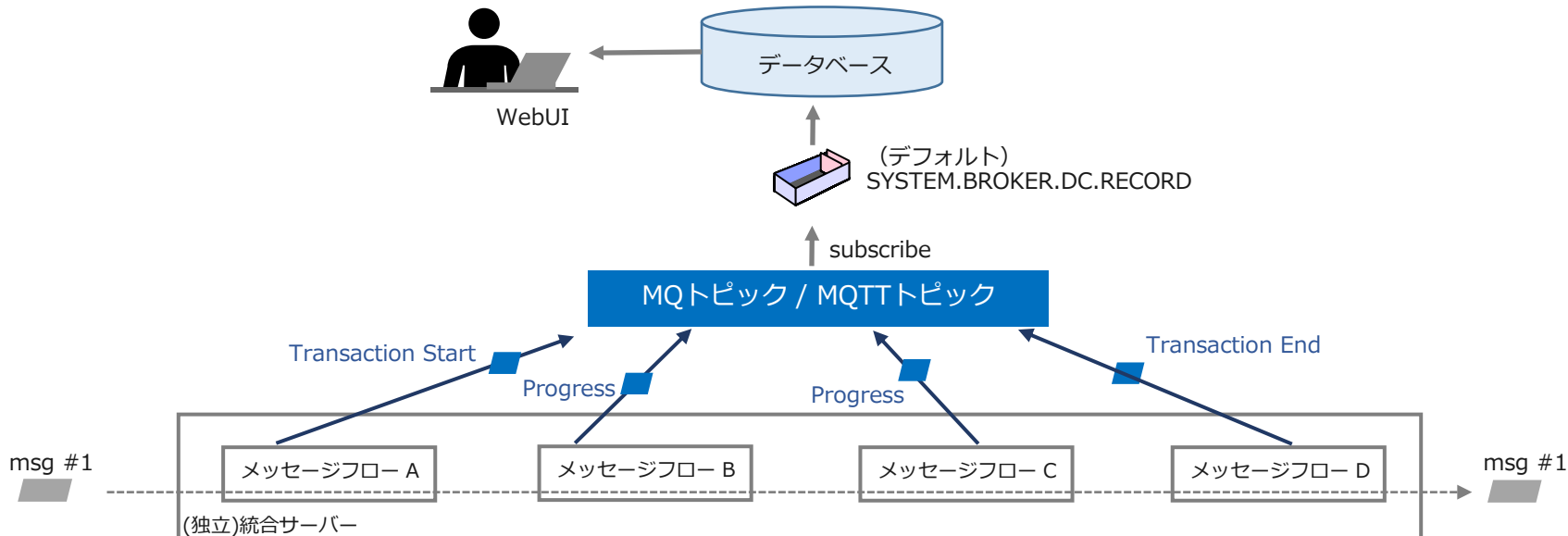


```
ibmint deploy --input-b C:¥Test¥temp¥Demo.bar --output-w C:¥Test¥temp¥MyDemoWorkDirectory --ov  
C:¥Test¥temp¥demopropertyoverrides.txt
```

ビジネス・トランザクション・モニ  
タリング

# ビジネス・トランザクション・モニタリング

- 複数のフローをまたがる一連のビジネス・トランザクションをモニタリングする機能を提供 (12.0.2.0)
  - ◆ 複数のフローをまたがる一連のビジネス・トランザクションがどの処理まで完了しているか
  - ◆ 途中でエラーが発生していないか など



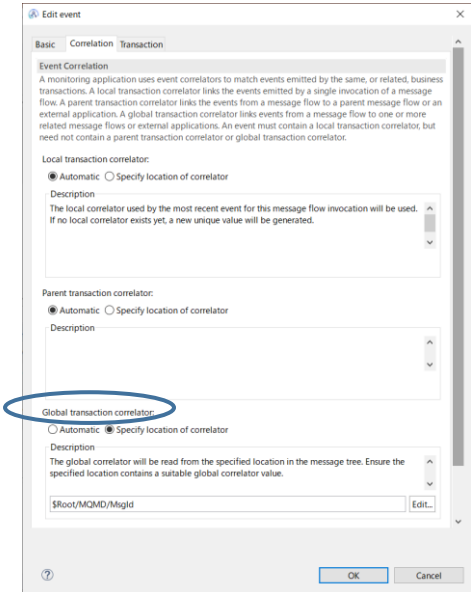
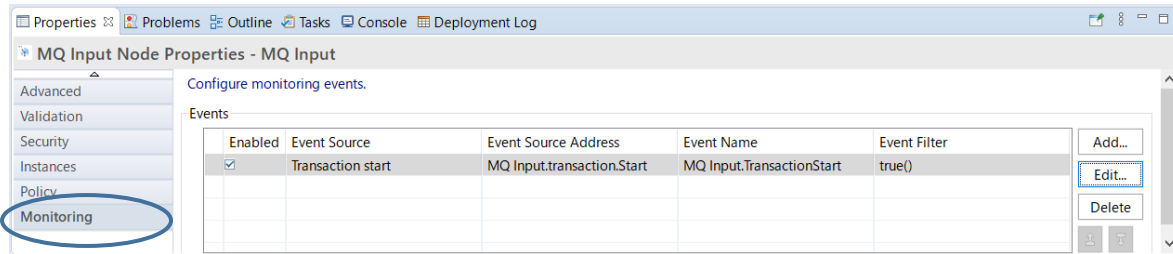
# ビジネス・トランザクション・モニタリング

- ポリシーとserver.conf.yamlで必要な情報を定義
  - ◆ ビジネス・トランザクションに含まれる各フローでモニタリングの設定を有効化
  - ◆ ビジネス・トランザクション・ポリシーでビジネス・トランザクションとしてモニタリングするイベントを定義
    - ・ 主なイベントとしては開始、終了、進行中、失敗など
    - ・ WebUIでポリシー定義を作成することも可能
  - ◆ server.conf.yamlでポリシー名や使用するデータソース名などの構成情報を定義
- グローバル・トランザクションIDを使用して、1つのビジネス・トランザクション内の各イベントを関連付け、処理の遷移を管理
  - ◆ メッセージ・ツリー内の任意のフィールドをグローバル・トランザクションIDとして指定可能
- WebUIからビジネス・トランザクションの結果を確認可能
  - ◆ Configureタブ . . . モニタリングの定義を作成
  - ◆ Monitorタブ . . . ビジネス・トランザクションのステータスをモニター

# 1.各フローでモニタリング設定を有効化

## ■ 各フローでモニタリングの設定を有効化

- ◆ いずれかの方法で有効化が可能
  - メッセージ・フロー・ノードの「Monitoring」プロパティ
  - モニタリング・プロファイル
- ◆ グローバル・トランザクション・IDを利用し、相関情報がセットされている必要がある



## 2. モニタリング機能を有効化

### ■ 統合サーバでモニタリング機能を有効化

- ◆ server.conf.yamlで以下の設定を有効にする

server.conf.yaml例

```
Monitoring:
  MessageFlow:
    publicationOn: 'active'           # choose 1 of : active|inactive, default is inactive
                                       # Ensure Events.BusinessEvents.MQ|MQTT is set
    eventFormat: 'MonitoringEventV2' # choose 1 of : MonitoringEventV2|WMB
```

### ■ 統合ノードまたは独立統合サーバでビジネス・モニタリング・イベントを有効化

- ◆ node.conf.yamlまたはserver.conf.yamlで以下の設定を有効にする

- イベントをパブリッシュするキュー・マネージャーをdefaultQueueManagerプロパティまたはポリシーで指定
- イベント情報を保管するキュー・マネージャーと同一のキュー・マネージャーを指定

node.conf.yaml例

```
Events:
  BusinessEvents: # Monitoring events
  MQ:
    policy: "" # Specify a {policy project}:policy if not using 'defaultQueueManager'
    enabled: true # Set true or false, default false
    format: "" # Set string or none
    outputFormat: 'xml' # Set comma separated list of one or more of : json,xml. Defaults to 'xml'
    publishRetryInterval: 0 # Set the retry interval (in milliseconds), to pause all publications and retry,
                           # when publication failures are causing serious delay to the transaction.
```



# 3. データベースやMQの準備

- モニタリング・イベントやビジネス・トランザクション・データを保管するためのデータベースを用意
  - ◆ レコード&リプレイ機能と同一のテーブルを使用可能
  - ◆ DataCaptureSchema.sqlを実行して、必要なテーブルを作成
  - ◆ BusinessCaptureSchema.sqlを実行して、必要なテーブルを作成
    - ・ いずれも<製品導入ディレクトリ>/server/ddl/配下にddlが格納

- 統合ノードまたは統合サーバでデフォルト・キュー・マネージャーを設定
  - ◆ node.conf.yaml または server.conf.yaml でデフォルト・キュー・マネージャーを設定
  - ◆ 独立統合サーバの場合は、リモートのデフォルト・キュー・マネージャーの指定も可能

node.conf.yaml例

```
defaultQueueManager: 'BTMQM'
```

- 必要なシステム・キューが作成されていることを確認
  - ◆ ビジネス・トランザクション・モニタリングでは、SYSTEM.BROKER.DC.RECORDキューとSYSTEM.BROKER.DC.BACKOUTキューを使用
  - ◆ <製品導入ディレクトリ>/server/sample/wmq/iib\_createqueues スクリプトを実行して作成可能

# 3. データベースやMQの準備

## ■ ビジネス・トランザクション・モニタリングに使用するリソース情報を設定

- ◆ server.conf.yaml のレコード&リプレイのStoresセクションでデータソースやキューの情報を設定  
server.conf.yaml例

```
RecordReplay:
Stores:
  BTMDataStore:
    dataSource: 'MBRECORD'
    schema: ''
    storeMode: 'all'
    queue: 'SYSTEM.BROKER.DC.RECORD'
    backoutQueue: 'SYSTEM.BROKER.DC.BACKOUT'
    useCoordinatedTransaction: false
    commitCount: 10
    threadPoolSize: 10
    commitIntervalSecs: 5
```

## ■ レコード&リプレイ機能を有効化

- ◆ server.conf.yaml でレコード&リプレイを有効化する  
server.conf.yaml例

```
RecordReplay:
  recordReplayEnabled: true
```

## ■ 管理セキュリティを有効化している場合は、必要な権限を必要なユーザーに付与

# 4. ビジネス・トランザクション・ポリシーの定義

- Webユーザー・インターフェースを使用してビジネス・トランザクション・ポリシー定義を作成

The image shows a sequence of three screenshots from the IBM App Connect web interface, illustrating the process of creating a business transaction definition. The first screenshot shows the main dashboard with the 'Business transactions' icon circled in blue. The second screenshot shows the 'Business transactions' page with the 'Configure' tab selected and the 'Create a transaction definition' button circled in blue. The third screenshot shows the 'Create a business transaction definition' form with the 'Create new policy project...' option selected in the 'Policy project' dropdown.

IBM App Connect

ACE1202

ビジネス・トランザクションアイコンをクリック

Business transactions

Monitor Configure

Create a transaction definition +

「Configure」タブで「Create a transaction definition」をクリック

Create a business transaction definition ×

A business transaction definition requires an associated business transaction policy that defines the business events that comprise a business transaction. This may be a new policy or an existing policy.

Integration server

EG01

Name

OrderProcess\_Mon1

Policy project

Select a policy project...

Create new policy project...

ビジネス・トランザクション・ポリシーを作成

# 4. ビジネス・トランザクション・ポリシーの定義

Create a business event

OrderProc1

Type

Start

Select a monitoring event

Integration server

EG01

Integration

BMT\_TEST

Library

None

Message flow

RECEIVE\_ORDER

Monitoring event source

MQ Input.transaction.Start

Cancel Create

ビジネス・トランザクションのStartイベント、Endイベントなどの定義を作成

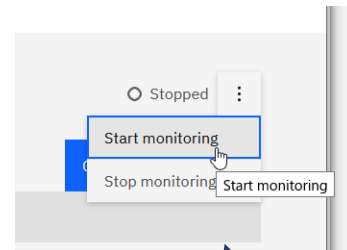
Business transactions

Monitor Configure

Business events for OrderProcess\_Mon1

Event name	Event type	Message flow	Event source	
OrderProc1	Start	RECEIVE_ORDER	MQ Input.transaction.Start	
OrderProc2	Progress	PROC_ORDER	MQ Input.transaction.Start	
OrderProc3	End	COMPLETE_ORDER	MQ Input.transaction.End	

定義を作成後、「Start monitoring」でモニタリングを開始



# 5. ビジネス・トランザクションのモニター

🔍 Business transactions

「Monitor」タブで実行状況を確認

Monitor Configure

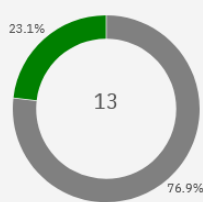
EG01

OrderProcess\_Mon1

## Transactions for OrderProcess\_Mon1

Started

最終更新: 数秒前



In progress	76.9% (10)
Complete	23.1% (3)
Inconsistent	0.0% (0)
Failed	0.0% (0)



Last 30 minutes



Transaction ID	Transaction status	Start time local	Last updated local	Duration
<a href="#">414d51205445535430312020202020aa86836124772d03</a>	In progress	2021-11-05 13:29:22.112	2021-11-05 13:29:22.131	27 min 1 sec
<a href="#">414d51205445535430312020202020aa86836124773202</a>	In progress	2021-11-05 13:29:45.151	2021-11-05 13:29:45.159	26 min 38 sec
<a href="#">414d51205445535430312020202020aa86836124773203</a>	In progress	2021-11-05 13:29:46.029	2021-11-05 13:29:46.032	26 min 37 sec
<a href="#">414d51205445535430312020202020aa86836124773204</a>	In progress	2021-11-05 13:29:46.799	2021-11-05 13:29:46.805	26 min 36 sec
<a href="#">414d51205445535430312020202020aa86836124773205</a>	In progress	2021-11-05 13:29:47.523	2021-11-05 13:29:47.525	26 min 36 sec
<a href="#">414d51205445535430312020202020aa86836124773206</a>	In progress	2021-11-05 13:29:48.240	2021-11-05 13:29:48.244	26 min 35 sec

# V11 Fix Packでの主な追加機能

# ACE V11 FixPackで追加された主な新機能

- ACEはFixPackで新機能を随時提供
- ACE V11のFixPackで提供された新機能については以下を参照
  - ◆ <https://www.ibm.com/docs/en/app-connect/11.0.0?topic=wniv1-new-function-added-in-version-110-fix-packs>
- ACE V11のFixPackで追加された主な新機能
  - ◆ ExceptionLog リソース・マネージャーの追加 (11.0.0.4)
  - ◆ ACE vaultを利用した新たな資格情報管理の提供
    - 独立統合サーバでのvaultの利用サポート (11.0.0.6)
    - 統合ノードでのvaultの利用サポート (11.0.0.7)
    - HTTPリクエストのベーシック認証 (11.0.0.9)
  - ◆ LDAPによるACEの管理権限の認可 (11.0.0.6)
  - ◆ リモート・デフォルト・キュー・マネージャー (11.0.0.7)
  - ◆ Kafkaサポートの拡張 (11.0.0.7、11.0.0.9、11.0.0.10)
  - ◆ Web UIへの統計情報のパブリッシュ (11.0.0.8)
  - ◆ Admin logの提供 (11.0.0.10)
  - ◆ RDQMのサポート (11.0.0.10)
  - ◆ IBM MQ uniform clusterのサポート (11.0.0.10)
  - ◆ IBM App Connect Dashboardの提供 (11.0.0.11)

# ExceptionLogリソース・マネージャーの追加 (11.0.0.4)

- ExceptionLogリソース・マネージャーにより、メッセージ・フロー内で発生した例外をファイルに出力することが可能
  - ◆ メッセージ・フロー内でキャッチした例外も出力される
    - これまでは、フロー内で発生した例外はフロー内でキャッチしなければSystemLogに出力、フロー内でキャッチした場合はどこにも出力されない
- server.conf.yamlの設定により、有効化
  - ◆ ResourceManagersセクションのExceptionLogで必要なパラメーターを指定 (次ページ)
  - ◆ デフォルトでは無効
- 出力先
  - ◆ 独立統合サーバの場合：  
<WorkDirectory>/config/common/log/integration\_server.<IntegrationServerName>.exceptionLog.txt
  - ◆ 統合ノード構成の統合サーバの場合：  
<構成Dir>/common/log/<IntegrationNodeName>.<IntegrationServerName>.exceptionLog.txt
  - ◆ 4つのファイルを循環して出力
    - ファイル名に1~3のsuffixがつく
    - 4つ目まで行くと最初のファイルを上書き
    - 統合サーバがリスタートされると、ファイルが切り替えられる
  - ◆ server.conf.yamlのパラメータで変更可能



# ExceptionLogリソース・マネージャーの追加 (11.0.0.4)

## ■ ResourceManagersセクション-ExceptionLogのパラメーター

パラメーター名	設定内容
enabled	ExceptionLogを有効化するかどうか (true/false)
exceptionLog	出力先ファイルのパスと名前を指定 ( <code>'[iib.system-common-log-dir]/[iib.system-node-label].[iib-system-server-label].exceptionLog.txt'</code> )
exceptionLogFileSize	出力先ファイルのサイズ (MB) ( <u>25</u> )
exceptionLogFileCount	出力先ファイルの数 (4)
includeFlowThreadReporter	エラー発生時に詳細なエラー情報を出力するかどうか (true/false)
showNestedExceptionDetails	ネスト化された例外を含めるかどうか (true/false)

server.conf.yamlの例

```
ResourceManagers:
  ExceptionLog:
    enabled: true          # Enables logging of exceptions
    #exceptionLog: '[iib.system-common-log-dir]/[iib.system-node-label].[iib-system-server-label].exceptionLog.txt'
                        # The location in which the rotating exception log file should be written
                        # This path must already exist and be writeable by the integration server user.
    #exceptionLogFileSize: 25    # The maximum size in MB of a single file that the exception log can use.
    #exceptionLogFileCount: 4    # The maximum number of files that the exception log can rotate through.
    includeFlowThreadReporter: true # Toggles whether exception in the exception log include a flow stack and history from the flow thread reporter
    showNestedExceptionDetails: true # Toggles whether nested exceptions are shown by default in the exception log
```

# ExceptionLogリソース・マネージャの追加 (11.0.0.4)

例外 (SystemLogに出力された場合の内容)

```
Nov 21 11:26:51 vagrant ACE[4182]: IBM App Connect Enterprise v12020 (IN01.is01) [Thread 4534] (Msg 1/5) BIP3120E: Exception condition detected on input node 'http01.HTTP Input'.
Nov 21 11:26:51 vagrant ACE[4182]: IBM App Connect Enterprise v12020 (IN01.is01) [Thread 4534] (Msg 2/5) BIP2230E: Error detected whilst processing a message in node 'http01.HTTP Request'.
Nov 21 11:26:51 vagrant ACE[4182]: IBM App Connect Enterprise v12020 (IN01.is01) [Thread 4534] (Msg 3/5) BIP3162S: An HTTP error occurred. The HTTP Request-Line was: 'POST /http01 HTTP/1.1#015 '.
Nov 21 11:26:51 vagrant ACE[4182]: IBM App Connect Enterprise v12020 (IN01.is01) [Thread 4534] (Msg 4/5) BIP3152S: Socket error detected whilst invoking Web service located at host localhost, port 9999, path /http01.
Nov 21 11:26:51 vagrant ACE[4182]: IBM App Connect Enterprise v12020 (IN01.is01) [Thread 4534] (Msg 5/5) BIP3150S: A socket error occurred.
Operation: ::connect::poll(). Error Code: 111. Error Text: Connection refused.
```

上記例外をExceptionLogで出力した場合

※includeFlowThreadReporterとshowNestedExceptionDetailsはfalseに設定

```
2021-11-21 11:57:10.547726 4359 THROWN BIP3150S SocketException ImbBasicSocket::connectTimeout 'An error occurred whilst performing a socket operation: getsockopt' ['::connect::poll()', 111, 'Connection refused'] CommonServices/ImbBasicSocket.cpp:744
2021-11-21 11:57:10.549714 4359 THROWN BIP3152S RecoverableException Imb::WSRequest::makeWSRequest 'A Web Service request has detected a SOCKET error whilst invoking a web service located at host &1, on port &2, on path &3.' ['localhost', 9999, '/http01'] WebServices/WSBase/ImbWSRequest.cpp:723 <Nested: BIP3150S>
2021-11-21 11:57:10.550509 4359 THROWN BIP3162S RecoverableException ImbWSRequestNode::evaluate 'WebService Request Exception'
[X'X'436f6e746556e742d4c656e6774683a2031370d0a436f6e74656e742d547970653a206170706c69636174696f6e2f6a736f6e3b636861727365743d7574662d380d0a486f73743a206c6f63616c686f73743a393939390d0a534f4150416374696f6e3a20222d0d0a436f6e6e56374696f6e3a204b6565702d416c6976650d0a0d0a", ", " 'POST /http01 HTTP/1.1
'] WebServices/WSLibrary/ImbWSRequestNode.cpp:699 <Nested: BIP3152S, BIP3150S>
2021-11-21 11:57:10.551947 4359 THROWN BIP2230E RecoverableException ImbWSRequestNode::evaluate 'Caught exception and rethrowing'
WebServices/WSLibrary/ImbWSRequestNode.cpp:787 <Nested: BIP3162S, BIP3152S, BIP3150S>
```

# ExceptionLogリソース・マネージャの追加 (11.0.0.4)

前述例外をExceptionLogで出力した場合

※includeFlowThreadReporterはtrue、showNestedExceptionDetailsはfalseに設定

※便宜上、BIP2230Eのみ抜粋

```
2021-11-21 12:17:36.352291 4906 THROWN BIP2230E RecoverableException ImbWSRequestNode::evaluate 'Caught exception and rethrowing'
WebServices/WSLibrary/ImbWSRequestNode.cpp:787 <Nested: BIP3162S, BIP3152S, BIP3150S>
Flow Thread Stack :
: Node: 'HTTP Request[ComIbmWSRequestNode]', From: 'Compute/out', Time: 2021-11-21T12:17:36Z, Sequence number: 1, Flow message ID: (0000132A-619A38E0-00000001)
: Node: 'Compute[ComIbmComputeNode]', From: 'HTTP Input/out', Time: 2021-11-21T12:17:36Z, Sequence number: 0, Flow message ID: (0000132A-619A38E0-00000001)
: Node: 'HTTP Input[ComIbmWSInputNode]', From: 'Flow thread pool', Time: 2021-11-21T12:16:19Z, Sequence number: 0, Flow message ID: (00000000-00000000-00000000)
End of Flow Thread Stack :
Flow Thread History :
: Node: 'HTTP Request', From: 'Compute/out', Time: 2021-11-21T12:17:36Z, Time difference: 0us, Sequence number: 1, Flow message ID: (0000132A-619A38E0-00000001)
: Node: 'Compute', From: 'HTTP Input/out', Time: 2021-11-21T12:17:36Z, Time difference: 0us, Sequence number: 0, Flow message ID: (0000132A-619A38E0-00000001)
: Node: 'HTTP Input', From: 'Flow Thread Pool', Time: 2021-11-21T12:16:19Z, Time difference: 76us, Sequence number: 0, Flow message ID: (00000000-00000000-00000000)
End of Flow Thread History :
```

前述例外をExceptionLogで出力した場合

※includeFlowThreadReporterとshowNestedExceptionDetailsはtrueに設定

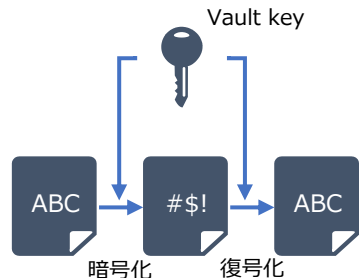
※便宜上、BIP2230Eのみ抜粋

```
2021-11-21 12:23:42.143444 5360 THROWN BIP2230E RecoverableException ImbWSRequestNode::evaluate 'Caught exception and rethrowing'
WebServices/WSLibrary/ImbWSRequestNode.cpp:787 <Nested: BIP3162S, BIP3152S, BIP3150S>
[--> BIP3162S RecoverableException ImbWSRequestNode::evaluate 'WebService Request Exception'
[X'X'436f6e74656e742d4c656e6774683a2031370d0a436f6e74656e742d547970653a206170706c69636174696f6e2f6a736f6e3b636861727365743d7574662d380d0a486f73743a206c6f63616c6
86f73743a393939390d0a534f4150416374696f6e3a2022220d0a436f6e6e656374696f6e3a204b6565702d416c6976650d0a0d0a"', X'X'7b22726573706f6e7365223a226f6b227d"', ", ", 'POST
/http01 HTTP/1.1
'] WebServices/WSLibrary/ImbWSRequestNode.cpp:699 <Nested: BIP3152S, BIP3150S>
[--> BIP3152S RecoverableException Imb::WSRequest::makeWSRequest 'A Web Service request has detected a SOCKET error whilst invoking a web service located at host &1, on port &2, on
path &3.' [localhost', 9999, /http01'] WebServices/WSBase/ImbWSRequest.cpp:723 <Nested: BIP3150S>
[--> BIP3150S SocketException ImbBasicSocket::connectTimeout 'An error occurred whilst performing a socket operation: getsockopt' [':connect::poll()', 111, 'Connection refused']
CommonServices/ImbBasicSocket.cpp:744
Flow Thread Stack :
: Node: 'HTTP Request[ComIbmWSRequestNode]', From: 'Compute/out', Time: 2021-11-21T12:23:42Z, Sequence number: 1, Flow message ID: (000014F0-619A3A4E-00000001)
: Node: 'Compute[ComIbmComputeNode]', From: 'HTTP Input/out', Time: 2021-11-21T12:23:42Z, Sequence number: 0, Flow message ID: (000014F0-619A3A4E-00000001)
: Node: 'HTTP Input[ComIbmWSInputNode]', From: 'Flow thread pool', Time: 2021-11-21T12:22:46Z, Sequence number: 0, Flow message ID: (00000000-00000000-00000000)
End of Flow Thread Stack :
Flow Thread History :
: Node: 'HTTP Request', From: 'Compute/out', Time: 2021-11-21T12:23:42Z, Time difference: 0us, Sequence number: 1, Flow message ID: (000014F0-619A3A4E-00000001)
: Node: 'Compute', From: 'HTTP Input/out', Time: 2021-11-21T12:23:42Z, Time difference: 0us, Sequence number: 0, Flow message ID: (000014F0-619A3A4E-00000001)
: Node: 'HTTP Input', From: 'Flow Thread Pool', Time: 2021-11-21T12:22:46Z, Time difference: 55us, Sequence number: 0, Flow message ID: (00000000-00000000-00000000)
End of Flow Thread History :
```

# ACE vaultを利用した新たな資格情報管理の提供

## ■ ACE vaultを利用して、よりセキュアに資格情報を管理

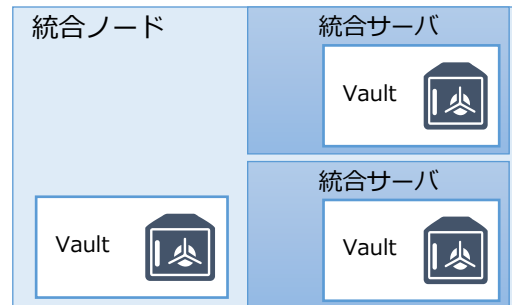
- ◆ ランタイム上で資格情報を管理するセキュアな暗号化ストアを提供
  - vault内のデータは対称鍵暗号方式で暗号化/復号化を行う
- ◆ 独立統合サーバ構成はV11.0.0.6、統合ノード構成はV11.0.0.7でサポート
  - 統合ノードでvaultを作成すると、紐づく統合サーバにもそれぞれvaultが構成される
  - 同一統合ノード下では同一のvaultキーを使用
  - 資格情報は統合サーバごとに定義が必要
- ◆ 従来通りmqsisetdbparmsコマンドも使用可能



## ■ mqsvaultコマンドでvaultを作成

- ◆ 統合ノード構成の場合は、統合ノード作成時にパラメータを指定してvaultを作成することも可能
  - mqsicreatebrokerコマンドの --vault-key、--vaultrc-location

## ■ mqsicredentialsコマンドで資格情報を登録・更新・削除



統合ノード構成

# ACE vaultを利用した新たな資格情報管理の提供

- 管理REST APIやWeb UIを使用して資格情報の参照が可能
  - ◆ 管理REST APIは資格情報の作成も可能
- 独立統合サーバのvaultに保管した認証情報を利用したHTTPリクエストのベーシック認証をサポート（11.0.0.9）
  - ◆ HTTPInputノード等が受け付けるHTTPリクエストに対し、vault保管の資格情報を基にベーシック認証を行うことができる
  - ◆ これまではLDAPやTFIMが必要

(参考情報)

<https://www.ibm.com/docs/en/app-connect/11.0.0?topic=enterprise-configuring-encrypted-security-credentials>  
SIL「ACE資格情報管理」（2020年発行）

# LDAPによるACEの管理権限の認可 (11.0.0.6)

- 統合ノード、統合サーバ、独立統合サーバに対する運用操作を実行するためのユーザー権限を制御
  - ◆ V9まではMQベースで権限設定
  - ◆ V10からファイルベースでの権限設定が追加
  - ◆ V11.0.0.6でLDAPによる権限設定が可能に
- 管理セキュリティを有効化したうえで、認可モードとしてLDAPを選択する
  - ◆ 事前にLDAP認証を有効化しておく必要がある
    - (参考) Enabling LDAP authentication  
<https://www.ibm.com/docs/en/app-connect/12.0?topic=administration-enabling-ldap-authentication>
  - ◆ LDAPによる認証を行う場合は、ノード構成の場合はnode.conf.yaml、独立統合サーバ構成の場合はserver.conf.yamlにて設定

# LDAPによるACEの管理権限の認可 (11.0.0.6)

- ◆ node.conf.yamlの設定例 (※下記以外にもLDAP認証有効化の設定が必要)

- 管理セキュリティの有効化

```
authorizationEnabled: true # Clients web user role will be authorized when set true
```

- 認可モードとしてLDAPを指定

```
authorizationMode: 'ldap' # Set authorization mode. Choose 1 of : ldap, file or mq
```

- Roleの作成

```
Permissions:  
# Set Admin Security Authorization file permissions by web user role using 'read+:write+:execute+' , or 'all+'  
# '+' grants permission, '-' denies permission  
# e.g. define the following web user roles 'viewRole' and 'adminRole'  
viewRole: 'read+:write-:execute-'  
adminRole: 'all+'  
supportRole: 'read+:write-:execute+'  
administrator: 'all+'  
developer: 'read+:write-:execute+'
```

- LDAPのユーザーや属性とRoleの紐づけ

```
Security:  
LdapAuthorizeAttributeToRoleMap:  
  'graham': 'adminRole' # (user mapped to role)  
  'DB_ADMIN': 'adminRole' # (field mapped to role)  
  'o=AceViewersGroup',o=groups,ou=ace': 'viewRole' # (group mapped to role)  
  'o=AceAdminsGroup',o=groups,ou=ace': 'adminRole' # (group mapped to role)
```

# リモート・デフォルト・キュー・マネージャー (11.0.0.7)

- 従来は内部的にMQを利用する機能を使用する場合、統合サーバをローカルのキュー・マネージャーと関連付ける必要があった
  - ◆ タイムアウト制御機能、アグリゲーション機能、メッセージ・シーケンス機能など・・・
- 独立統合サーバでリモートで稼働するキュー・マネージャーをデフォルト・キュー・マネージャーとして指定可能に
  - ◆ 統合ノード構成の場合はV12でもデフォルト・キュー・マネージャーはローカルに構成する必要あり
  - ◆ 複数の独立統合サーバでキュー・マネージャーを共有することも可能

## ■ 構成方法

- ◆ 接続するキュー・マネージャー情報をセットしたMQ Endpoint Policyを作成
- ◆ server.conf.yamlでremoteDefaultQueueManagerに作成したポリシー情報をセット

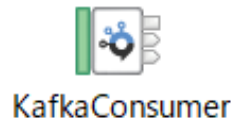
server.conf.yamlの例

```
remoteDefaultQueueManager: '{myPolicyProject};policy1'
```

- ◆ キュー・マネージャー作成後に必要なシステム・キューを定義する
  - 複数独立統合サーバでキュー・マネージャーを共有する場合は、キューを共有しないようにPrefixをつける
    - ✓ server.conf.yamlのreplacementQueuePrefixで使用するPrefix指定
    - ✓ 指定したPrefixがキュー名の「SYSTEM.BROKER.」部分にセットされる



# Kafkaサポートの拡張



## ■ Kafka Readノードの提供 (11.0.0.7)

- ◆ Kafkaトピックの中で特定のメッセージを読み込むことが可能

(参考情報) <https://www.ibm.com/docs/en/app-connect/11.0.0?topic=nodes-kafkaread-node>

## ■ Kafka PolicyによるKafkaノード設定の動的変更のサポート (11.0.0.7)

(参考情報) <https://www.ibm.com/docs/en/app-connect/11.0.0?topic=properties-kafka-policy>

## ■ Kafkaカスタム・ヘッダー・プロパティのサポート (11.0.0.7)

(参考情報) <https://www.ibm.com/docs/en/app-connect/11.0.0?topic=enterprise-setting-retrieving-kafka-custom-header-properties>

## ■ SASL/SCRAMを使用したKafkaクラスターでの認証をサポート (11.0.0.9)

(参考情報) <https://www.ibm.com/docs/en/app-connect/11.0.0?topic=messages-authenticating-connections-kafka-cluster-by-using-saslscram>

## ■ Kafka Consumerノードでの機能拡張 (11.0.0.10)

- ◆ オフセットのコミットが完了するまで、フローによるメッセージの処理を待機することが可能に
- ◆ ノードプロパティに「Wait for message offset commit to complete」のチェックボックスが追加
  - ・ デフォルト「Yes」 (チェックあり)

(参考情報) <https://www.ibm.com/docs/en/app-connect/11.0.0?topic=nodes-kafkaconsumer-node>

(参考情報) <https://www.ibm.com/docs/en/app-connect/11.0.0?topic=messages-using-kafka-app-connect-enterprise>

# Web UIへの統計情報のパブリッシュ (11.0.0.8)

- デフォルトでメッセージ・フロー統計、リソース統計のスナップショットがWebUIにパブリッシュされるように
  - ◆ server.conf.yaml/node.conf.yamlの以下のプロパティで動作を制御
    - Web UIへのメッセージ・フロー統計のスナップショットのパブリッシュ
      - ✓ Snapshot: publicationOn:'active'
      - ✓ Snapshot: outputFormat:'json'
    - Web UIへのリソース統計のパブリッシュ
      - ✓ Snapshot: reportingOn: true
  - ◆ アrchive情報はWebUIでは取得不可

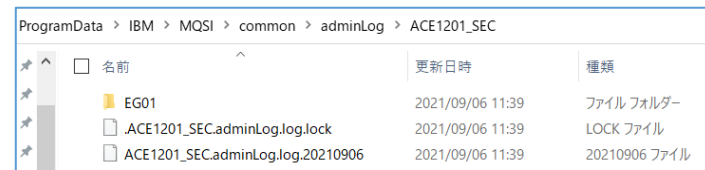


(参考情報)

<https://www.ibm.com/docs/en/app-connect/11.0.0?topic=ccmfasd-configuring-collection-message-flow-statistics-by-using-yaml-configuration-file>  
<https://www.ibm.com/docs/en/app-connect/11.0.0?topic=performance-managing-resource-statistics-collection>

# Admin logの提供 (11.0.0.10)

- 統合ノード、統合サーバに対する運用管理の操作ログが取得可能に
  - ◆ デフォルトで有効化
  - ◆ 無効化する場合は、node.conf.yaml または server.conf.yamlで設定 (①)
- Web UIのAdmin Logタブ、または管理REST APIで表示
  - ◆ データは統合ノードまたは統合サーバ内のメモリ上に保持
- Adminログをファイルに出力することも可能
  - ◆ 以下のディレクトリに出力
    - ・ 統合ノードの場合： workDir/common/adminLog/
    - ・ 独立統合サーバの場合： workDir/adminLog
  - ◆ 独立統合サーバの場合はコンソールに出力することも可能
  - ◆ 有効にする場合は、node.conf.yaml または server.conf.yamlで設定 (②)



名前	更新日時	種類
EG01	2021/09/06 11:39	ファイルフォルダ
.ACE1201_SEC.adminLog.log.lock	2021/09/06 11:39	LOCK ファイル
.ACE1201_SEC.adminLog.log.20210906	2021/09/06 11:39	20210906 ファイル

node.conf.yamlの例

```
AdminLog:  
#enabled: true
```

```
#fileLog: false  
#fileLogRetentionPeriod: 30  
#fileLogCountDaily: 10  
#fileLogSize: 100
```

# ①Admin logを無効化する場合は、この値をfalseに変更。

② # ファイル出力する場合は、この値をtrueに変更。  
# ログファイルの保管日数を指定。デフォルト30日。  
# 1日に出力する管理ログファイルの最大数を指定。デフォルト10ファイル/日。  
# 各管理ログファイルのサイズを指定。デフォルト100MB。

# Admin logの提供 (11.0.0.10)

## ■ ログに含まれる内容

- ◆ タイムスタンプ、BIPメッセージ番号、内容、実行ユーザー名、そのユーザーが持つ権限ロール

IBM App Connect

ノード: ACE1201\_SEC

ACE1201\_SEC

サーバー プロパティ フロー統計 データ **管理ログ**

最終更新: 数秒前

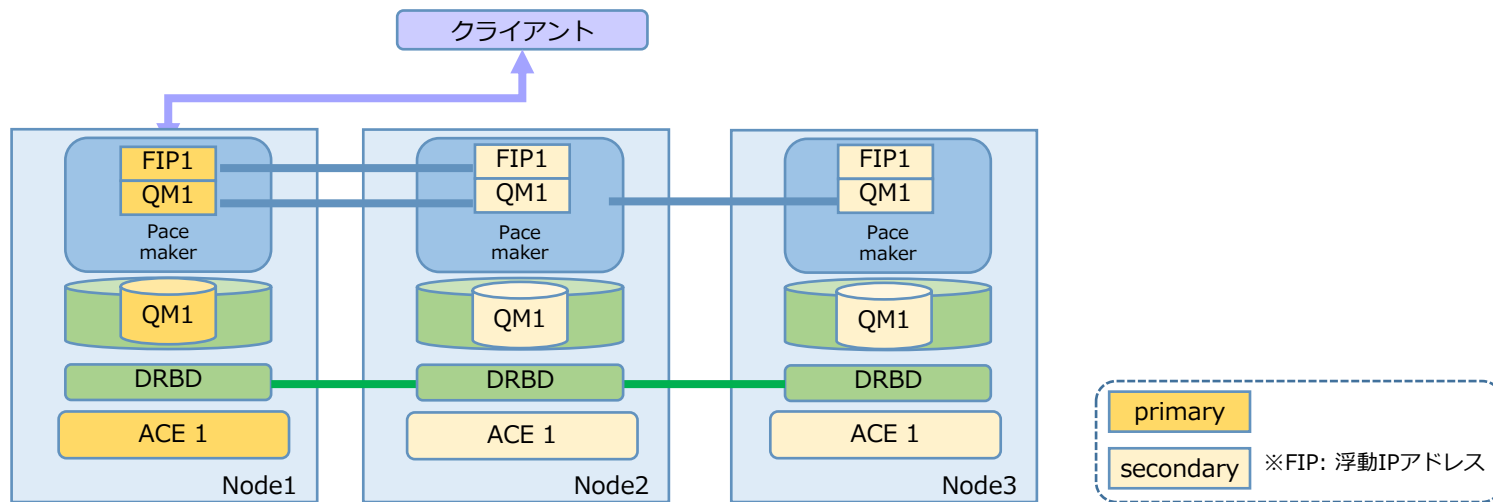
タグによるフィルター。例: USER:admin AUTHORIZED\_ROLE:ops HTTP\_METHOD:GET

タイム・スタンプ ローカル	メッセージ	メッセージ・テキスト	ユーザー	許可されている役割	タグ
2021-09-06 10:52:31.009	BIP20010I	ユーザー'aceadm'はログインしました。	aceadm		REQUEST_ID:20210906015231008000-adminui-0 <u>ROLES:iibadmin</u>
2021-09-06 10:44:34.632	BIP20002I	統合ノード'ACE1201_SEC'がバージョン'12.0.1.0'で開始されました。			VERSION:12.0.1.0

1 ページあたりの項目数 25 1 から 2 の項目 (全 2 件中) 1 /1 ページ

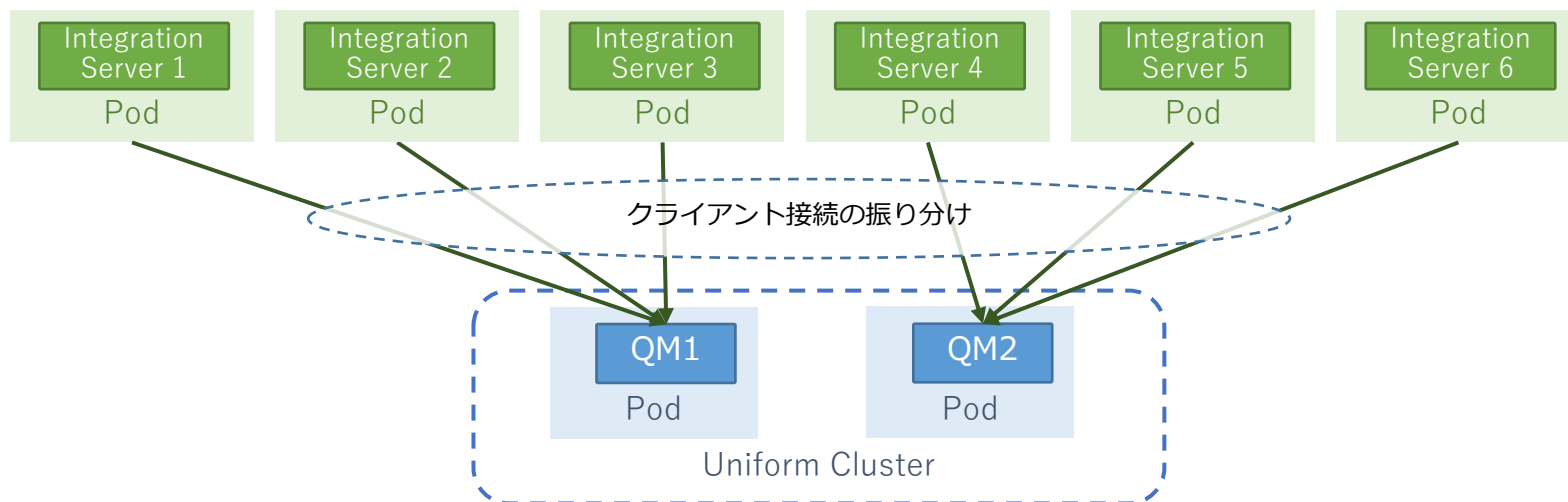
# RDQMのサポート (11.0.0.10)

- Replicated Data Queue Manager (RDQM) とは、MQV9.0.4から登場した新たな高可用構成
  - ◆ Linux RHELプラットフォームで使用可能な高可用性ソリューション
  - ◆ 外部ディスクなしで、オンプレミス環境だけでなくクラウド環境でも高可用性を実現可能
- 統合ノードをRDQMと連動するように構成することが可能



# IBM MQ uniform clusterのサポート (11.0.0.10)

- IBM MQ Uniform Clusterとは、MQV9.2から登場した新たな高可用構成
  - ◆ Uniform Clusterに参加する複数のキュー・マネージャーの間で、クライアント・アプリケーションからの接続を自動的に平準化しロード・バランシングする機能
- ACEからUniform Cluster構成のキューへのアクセスをサポート
  - ◆ MQ Endpointポリシーを使用して、接続先の情報や再接続オプション・プロパティを設定



# IBM App Connect Dashboardの提供 (11.0.0.11)

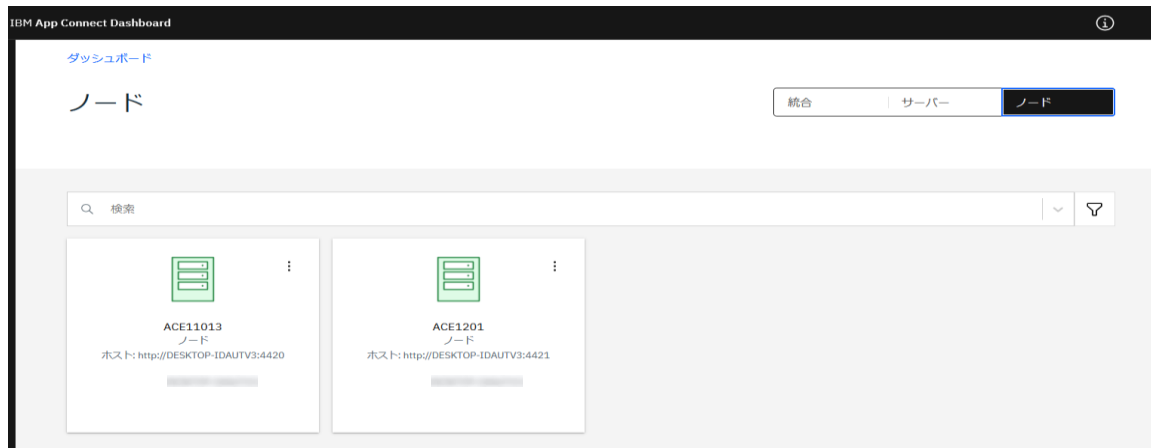
## ■ IBM App Connect Dashboardから統合サーバの管理が可能

- ◆ 複数の統合ノードや統合サーバを1つの管理画面から管理可能
- ◆ Dashboardコマンドで起動

```
C:¥Program Files¥IBM¥ACE¥12.0.1.0>Dashboard --work-dir C:¥z_Test¥Dashboard  
BIP3132I: The HTTP Listener has started listening on port '7700' for 'http' connections.
```

## ■ Dashboardをブラウザで開く

- ◆ *protocol://hostname:port*
- ◆ 複数のユーザーで同じダッシュボードを共有可能



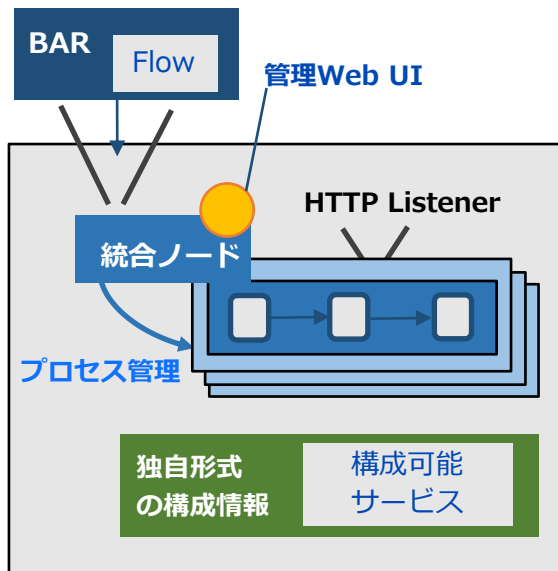
# マイグレーション



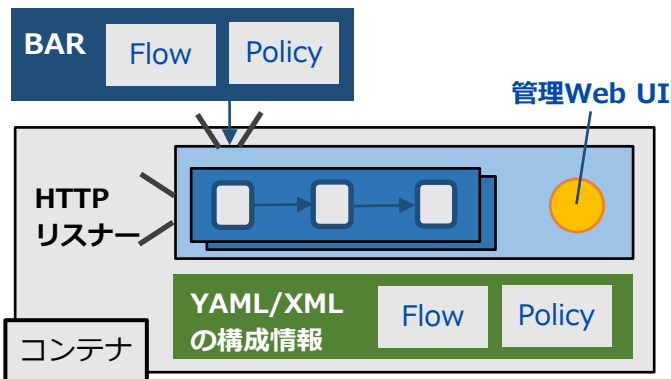
# 主な変更点

# 製品アーキテクチャーの変更

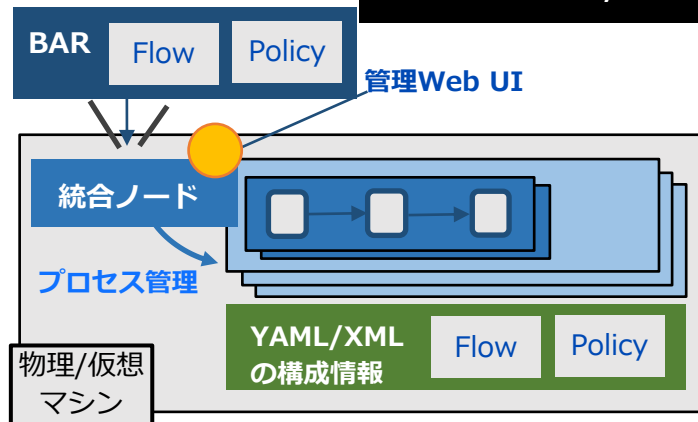
## IIB V10までのアーキテクチャー



UNZIP and GO !



## ACE V11, V12



### 独立統合サーバー

- ・単一プロセスで稼働する統合サーバー
- ・管理UIやHTTPリスナー

### ポリシー

- ・BARファイルに構成情報を開発物と一緒にデプロイ可能に
- ・構成情報はYAML/XMLの標準形式
- ・ファイルを直接編集可能

### 統合ノード構成

- ・統合ノードによる管理も可能
- ・構成情報はYAML/XMLの標準形式

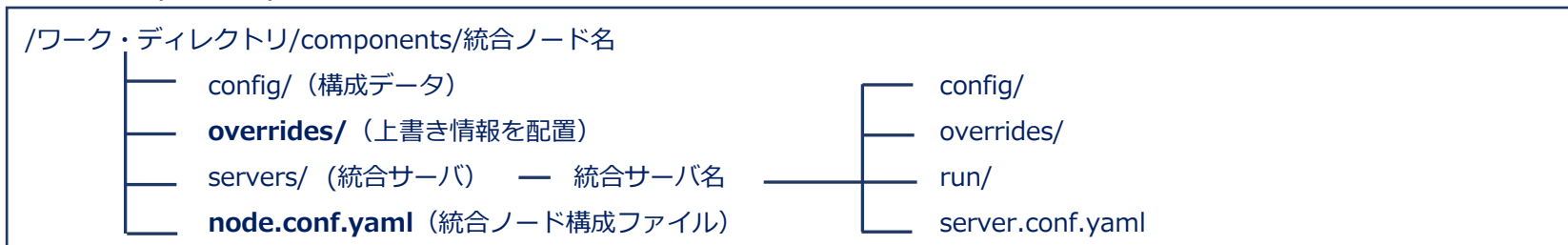
# 独立統合サーバの登場

- ACE V11から統合サーバ単体で稼働するランタイムを提供
  - ◆ 従来の統合ノード配下で稼働する統合サーバと区別して、「独立統合サーバ」と呼称
    - ・ プロセス名は、IntegrationServer（従来の統合サーバは、DataFlowEngine）
  - ◆ デプロイ/稼働できる開発リソースは従来の統合サーバと同じ
  - ◆ 統合ノード・レベルの機能は利用できない
    - ・ 統合ノード・リスナー(biphttplistener)、組み込みMQTTサーバ(bipMQTT)など
- コンテナ型仮想化技術との親和性が高いアーキテクチャー
  - ◆ 統合サーバ単位のワーク・ディレクトリ下に構成設定、アプリケーションを配置
  - ◆ YAML形式の構成ファイル「server.conf.yaml」による構成プロパティの設定
  - ◆ 統合サーバが停止している状態でアプリケーション（BARファイル）のデプロイが可能
    - ・ デプロイ・コマンドとして新たにmqsibarコマンドを提供
  - ◆ すべての構成設定を統合サーバ起動前に実施可能
    - ・ 従来通りmqsicommandで設定する項目もあるが、統合サーバ停止中に実行可能
  - ◆ コンテナ起動と同時に、必要な設定とアプリケーションの配置が完了した統合サーバを起動できる
- 独立統合サーバと統合ノードの製品コードを共通化するため、従来の統合ノードの構成、運用方法が変更

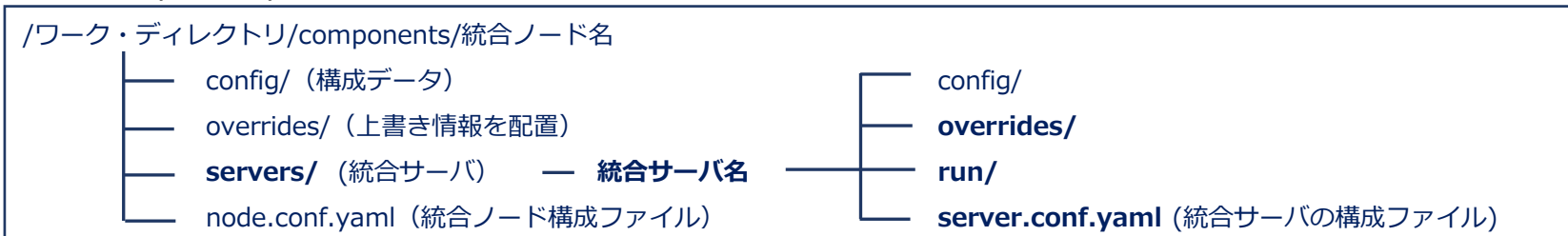
# 構成定義

## ■ 統合ノード、統合サーバの構成はYAMLファイルで設定

- ◆ 統合ノードを作成すると、node.conf.yamlファイルが生成される
  - /var/mqsi/components/<統合ノード名> 配下



- ◆ 統合サーバを作成すると、server.conf.yamlファイルが生成される
  - /var/mqsi/components/<統合ノード名>/servers/<統合サーバ名> 配下



## ■ 構成ファイルの設定は起動時にのみ読み込まれる

- ◆ 起動後の変更は統合ノード / 統合サーバの再起動が必要

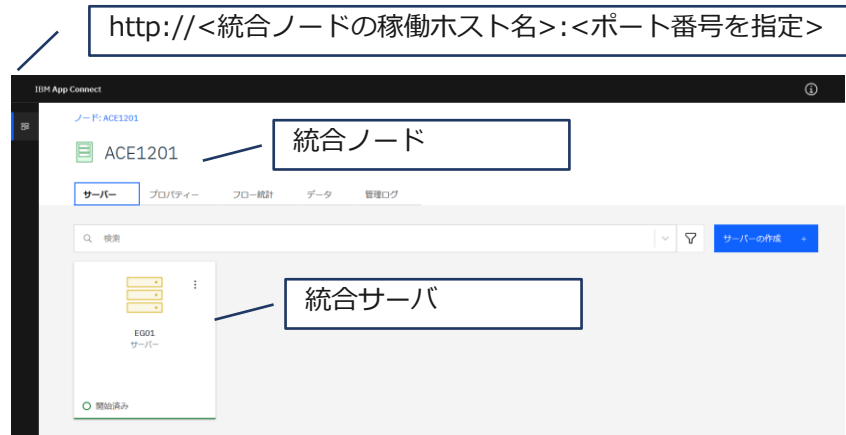
# ポリシー

- 外部リソース／サービスへの接続設定をポリシーに設定
  - ◆ V10以前は構成可能サービスやポリシーで定義
  - ◆ V11以降も非推奨だがmqsicreateconfigurableseviceコマンドは使用可能
    - コマンドを実行すると、定義はポリシーとして格納される
      - ✓ 統合ノードレベルのデフォルトのポリシー・プロジェクト (DefaultPolicies) にポリシーを作成
      - ✓ <ACEデータディレクトリ>/components/統合ノード名/policies/DefaultPolicies配下に格納
      - ✓ 明示的なデプロイは不要 (再起動は必要)
- ポリシーを利用することで開発や運用の負荷を軽減
  - ◆ 開発者は構成定義にポリシーを使用することで複数ノードで定義内容を再利用可能
  - ◆ 運用、管理者はフローの実行環境に応じて、フローを改修することなく、接続情報やモニタリング設定などの定義を変更可能
- HTTPConnector / HTTPSConnector ポリシーは廃止
  - ◆ 統合サーバのserver.conf.yamlを使用

# 運用まわりの変更

## ■ Web UIの変更

- ◆ ユーザー・インターフェースが変更
- ◆ IBM Integration ExplorerはV10から使用不可



## ■ RESTやIBM Integration API (Java) を使用した管理も可能

- ◆ REST APIで統合ノード、統合サーバ、その他リソースの管理を実施

## ■ トレース取得の変更

- ◆ mqsichangetraceコマンドなどで取得設定すると平文でトレースファイルが出力されるように変更
  - mqsireadlog / mqsiformatlog コマンドによるフォーマットが不要に
- ◆ /var/mqsi/common/log ディレクトリに参照可能なファイルが生成される

## ■ 管理セキュリティの変更

- ◆ ACEでは認証のみ有効化することが可能
  - V10までは管理セキュリティを有効にすると認証と認可が有効に
  - V11以降では認証のみ有効化、または認証と認可を有効化することが可能

# 運用まわりの変更

## ■ 管理コマンドの追加、廃止

### ◆ V11以降の追加コマンド

追加コマンド		用途
ace		iibコマンドに対応
IntegrationServer		独立サーバ構成関連コマンド
mqsibar		デプロイ関連コマンド
mqsicreateworkdir		独立サーバ構成関連コマンド
mqsixtractcomponents		マイグレーション関連コマンド
ibmint	V12	ACE管理用コマンド
aceDataCollector	V12	ACE構成情報の取得コマンド
Dashboard	V12	ACE Dashboard関連コマンド
mqsicredentials	V12	ACE vault関連コマンド
mqsivault	V12	ACE vault関連コマンド

### ◆ 廃止コマンド

廃止コマンド	用途
mqsireadlog	トレース取得方法の変更により廃止
mqsiformatlog	

# 運用まわりの変更

## ■ 管理コマンドの追加、廃止

### ◆ 廃止コマンド（つづき）

廃止コマンド	用途
mqsigratecomponents	parallelマイグレーションのみのサポートのため廃止
mqsicreatepolicy	ポリシーへの定義方法の変更により廃止
mqsichangepolicy	
mqsireportpolicy	
mqsideletepolicy	
mqsattachpolicy	
mqsidetachpolicy	
mqsdeleteconfigurablesevice	
mqschangebluemixreporting	
mqsireportbluemixreporting	

### ◆ 変更コマンド

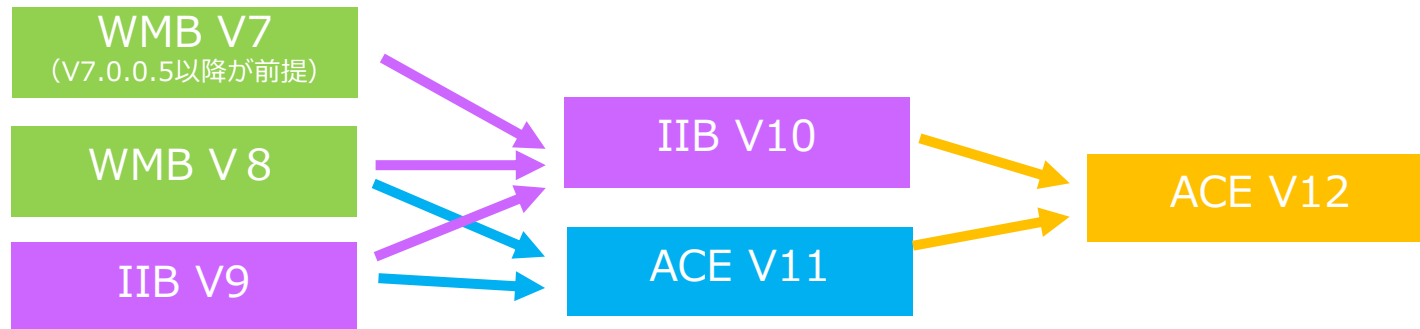
追加コマンド	用途
mqschangebroker	mqschangepropertiesコマンドで対応
mqsireportbroker	mqsireportpropertiesコマンドで対応
iib	aceコマンドに変更



# マイグレーションの考慮点

# ACE V12への移行パス

- IIB V10、ACE V11からACE V12への移行が可能
  - ◆ 旧バージョンからFull Editionまたは Standard Editionへのマイグレーションが可能
    - Developer Editionへのマイグレーションは不可

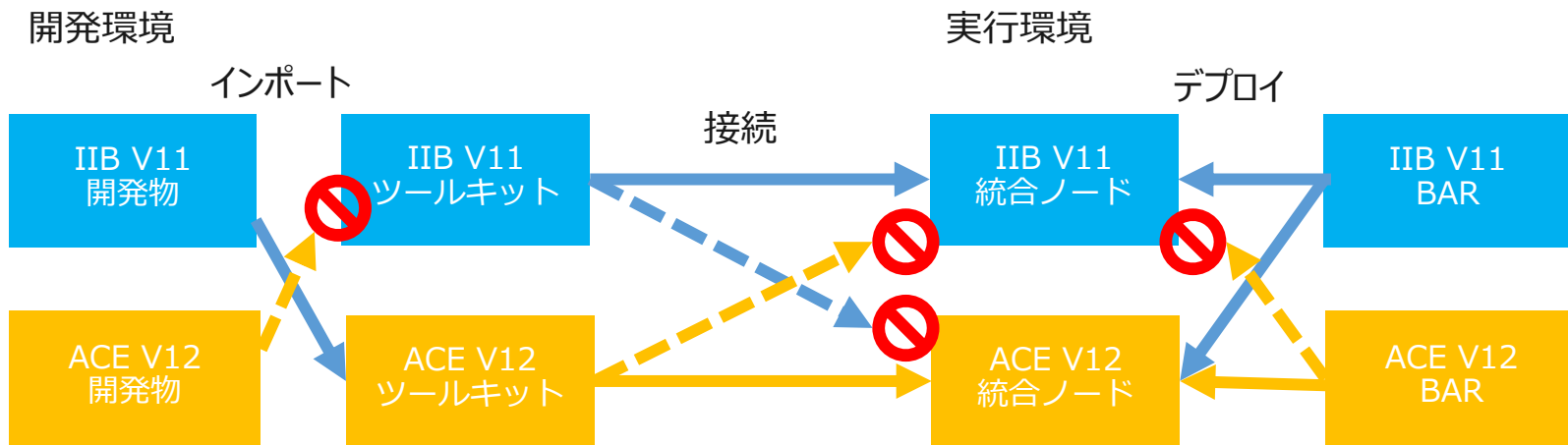


※直接移行できないバージョンは、一旦V10/V11に移行後、ACE V12に移行

※移行の前提条件の詳細については各移行先バージョンのKnowledge Centerを参照

# 互換性

## ■ 開発環境と実行環境の互換性



- ◆ 下位バージョンのツールキットからACE V12の統合ノードには接続不可

## ■ BARファイルの互換性

- ◆ 下位バージョンで作成したBARファイルはACE V12にデプロイ可能

# 互換性

## ■ 開発物の互換性

- ◆ 下位バージョンで作成した開発物はACE V12に移行可能
  - 以下のケースに該当する場合は、考慮が必要
- ◆ メッセージ・フロー
  - ACE V11以降Withdrawされている以下のノードを使用している場合は、再作成が必要
  - Database関連ノード
    - ✓ DataDelete、DataInsert、DataUpdate
    - ✓ Warehouse
  - SCA関連ノード
    - ✓ SCAInput、SCAReply、SCARequest、SCAAsyncRequest、SCAAsyncResponse
  - ODM関連ノード
    - ✓ Decision Service
  - その他ノード
    - ✓ Extract
    - ✓ V7以前のMapping
    - ✓ MQOptimizedFlow、Real-timeOptimizedFlow

# 互換性

- ◆ メッセージ・マップ
  - V7以前のマッピング定義 (.msgmap) は、グラフィカル・データ・マップ (.map)への変換が必要
    - ✓ .msgmapを使用したフローのインポートは可能
    - ✓ read-onlyで編集や実行環境へのデプロイは不可
    - ✓ 変換方法は以下を参照
      - <https://www.ibm.com/docs/en/app-connect/12.0?topic=uclrimm-converting-message-map-from-msgmap-file-map-file>
- ◆ サブフロー
  - 旧バージョンで.msgflow形式で作成したサブフローをACE V11以降で継続開発する場合、.subflow形式への変換が必要
    - ✓ 変換方法は以下を参照
      - <https://www.ibm.com/docs/en/app-connect/12.0?topic=resources-converting-between-message-flows-subflows>
- ◆ DFDLスキーマの妥当性検査の向上 (V9からの移行時)
  - DFDL1.0の仕様により準拠するために妥当性検査が強化されているため、既存の定義がエラーになる可能性がある
  - 詳細は以下を参照
    - ✓ [https://www.ibm.com/docs/en/app-connect/12.0?topic=SSTTDS\\_12.0/com.ibm.etools.mft.doc/bh44620\\_.html](https://www.ibm.com/docs/en/app-connect/12.0?topic=SSTTDS_12.0/com.ibm.etools.mft.doc/bh44620_.html)

# 互換性

- ◆ 追加の操作（もしくは考慮）が必要なケース
  - SSLv3を使用するフローの移行
  - JMS ノードを含むフローの移行
  - File ノードを含むフローの移行
  - ?wsdl 照会をサポートするフローの移行
  - SAPアダプター接続プロジェクトを含むアプリケーションの移行

※詳細は以下のKnowledge Centerを参照

<https://www.ibm.com/docs/en/app-connect/11.0.0?topic=tasks-migrating-deployable-resources>

<https://www.ibm.com/docs/en/app-connect/12.0?topic=tasks-migrating-deployable-resources>

# 互換性

## ■ 運用・監視の互換性

- ◆ ACE V11以降でWithdrawされる機能
  - WAS管理コンソールからの管理機能
- ◆ 管理コマンドの変更
  - 管理コマンドに追加、廃止がある
  - 統合ノードに接続するコマンドを使用したスクリプトはパラメータ変更による見直しが必要
- ◆ IB Explorerの廃止（V9からの移行時）
  - 旧バージョンでIB Explorer またはMB Explorerを使用していた場合、Web ユーザー・インターフェースに切替が必要

# 旧バージョンとの共存

## ■ 実行環境

- ◆ 異なるversion、release、modification、fixpack レベルの共存も可能
  - ACE V12はACE V11、IIB V10との共存が可能
  - 例えば、V12.0.0.1 とV12.0.0.2との共存も可能
- ◆ 各バージョンの構成コンポーネントを1筐体に構成可能
  - ただし、コンポーネント名は筐体内でユニークに定義する必要あり

## ■ 開発環境

- ◆ ツールキットは、異なるバージョンのツールキットと共存可能
  - 下位バージョンのツールキットからV12の統合ノードには接続できない
  - 開発物の共有は不可

## ■ コマンド実行環境はコマンド・コンソール／プロファイルでバージョンを切替

- ◆ Windows環境ではバージョンごとに専用のコマンド・コンソールを提供
- ◆ UNIX環境は各バージョンでプロファイルを提供
  - <導入ディレクトリ>/binにmqsipfileを提供



# マイグレーションの流れ

# ACEの移行対象

## ■ 実行環境

- ◆ 構成コンポーネントの移行
  - 統合ノード（旧ブローカー）の移行
  - 移行方法は、extract マイグレーションまたは parallel マイグレーションを選択

## ■ 開発環境

- ◆ ツールキットの移行
- ◆ 開発物の移行
  - ツールキットで開発したリソース（メッセージフロー、メッセージ定義など）の移行

## ■ 運用・監視機能

- 運用シェル、監視機能などユーザ実装のツールや監視製品の移行

# 実行環境の移行の流れ

## ■ 移行前タスク

- ◆ バックアップの取得
  - ・ システムバックアップ
  - ・ 統合ノードの構成 (mqsisbackupbroker )
  - ・ ツールキットで開発したリソース (メッセージフロー、メッセージ定義など)
  - ・ ODBC定義ファイル、構成可能サービス設定 (mqsiexportproperties の出力結果)
- ◆ DB接続設定の変更
  - ※「参考情報」参照
- ◆ オプション：MQの導入、移行

## ■ 構成コンポーネントの移行

- ◆ extractマイグレーション
- ◆ parallelマイグレーション

## ■ 移行後タスク

- ◆ 運用・監視スクリプト、運用監視ツール影響調査および修正
- ◆ 旧バージョンでIB Exploreを使用していた場合には、Web ユーザー・インターフェースへの切替
- ◆ その他の技術的な変更点の確認

# 構成コンポーネントの移行方法

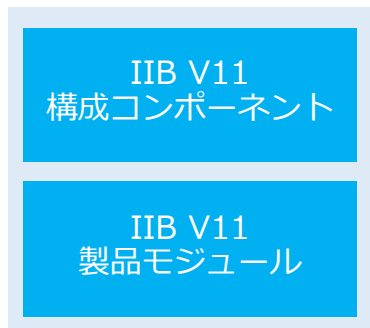
## ■ extract マイグレーション

- ◆ 既存の統合ノードの環境のバックアップから統合ノード構成/独立統合サーバ構成にマイグレーション
- ◆ 詳細はDocumentを参照
  - <https://www.ibm.com/docs/en/app-connect/12.0?topic=120-performing-extract-migration-integration-node-integration-server>

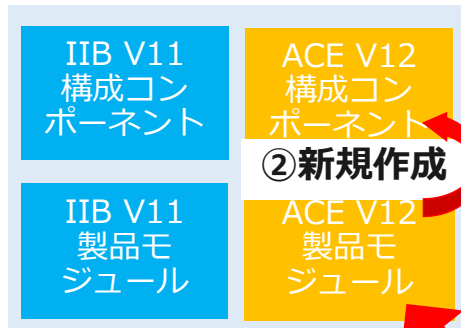
## ■ parallel マイグレーション

- ◆ 既存の統合ノードとは別に、新規に統合ノードを構成

移行前



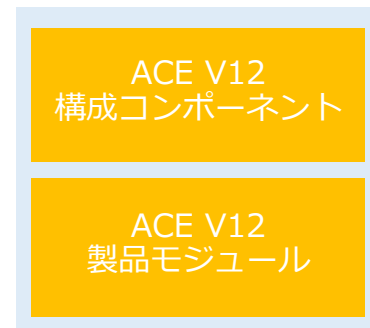
移行中（共存期間）



移行中（共存期間）



移行後



# parallelマイグレーション

## 1. ACE V12のインストール

- ◆ 現行サーバ、もしくは新規サーバにインストール

## 2. V12のツールキットに開発物を移行

- ◆ 現行ツールキットからプロジェクト交換ファイルの形式でエクスポートし、V12ツールキットにインポート

## 3. V12のコマンド実行環境をセットアップ

- ◆ Linux/Unix : V12の環境プロファイルmqsiprofileを実行
- ◆ Windows : V12のコマンド・コンソールを開く

## 4. 統合ノードを新規に作成

- ◆ V12 コマンド実行環境でmqsicreatebrokerコマンドを使用
- ◆ 現行サーバ上に作成する場合は既存とは異なる名前を指定

## 5. 統合ノードを起動し、必要な構成を定義

- ◆ 既存環境に合わせて、統合ノードのプロパティの設定、統合サーバーの作成

## 6. 統合サーバーへ開発物のデプロイ・稼働確認

## 7. 既存コンポーネントの削除、アンインストール

- ◆ 旧バージョンのコマンド環境にて、既存コンポーネントを削除し、旧バージョンをアンインストール

# 移行後のタスク

- IIB の運用・監視で使用しているコマンドの変更点を確認し、変更がある場合には、運用・監視スクリプト、運用監視ツール設定を更新
  - ◆ コマンド実行結果のフォーマット
  - ◆ コマンド実行時の動作
  - ◆ コマンド/APIのパラメーター
- IIBの運用監視で旧バージョンのIB ExplorerまたはMB Explorerを使用していた場合には、Web ユーザー・インターフェースへの切替が必要
- Global Cacheを利用している場合には移行後に作業が必要
  - ◆ <https://www.ibm.com/docs/en/app-connect/12.0?topic=tasks-setting-up-global-cache>
- その他の技術的な変更点の確認

※詳細は下記を参照

<https://www.ibm.com/docs/en/app-connect/12.0?topic=tasks-migrating-deployable-resources>

# フォールバック

## ■ 実行環境のフォールバック

- ◆ parallelマイグレーションの場合
  - V12環境の使用を停止し、既存環境の使用を再開

## ■ ツールキット/開発物のフォールバック

- ◆ 旧バージョンのワークスペースをV12に移行した場合は、バックアップから移行前のワークスペースを復元





# (参考) 移行前タスク : DB接続設定変更

## ■ Windows環境

- ◆ ODBCデータソースの再作成 (Oracle / Sybase 利用時のみ)
  - 既存のデータソースを削除し、以下のドライバーを指定して新規にデータソースを作成
  - Windowsの管理ツール (ODBCデータソース アドミニストレータ) で設定

DBMS	新規ODBCドライバー
Oracle	IBM App Connect Enterprise 12.0.1.n- DataDirect Technologies 64-BIT Oracle Wire Protocol
Sybase	IBM App Connect Enterprise 12.0.1.n- DataDirect Technologies 64-BIT Sybase Wire Protocol

※nはインストールされたフィックスパックのレベル

- ◆ XAリソース・マネージャーのSwitchFile 設定の変更
  - MQ エクスプローラーにてキューマネージャー・プロパティのSwitchFile 設定を変更

DBMS	現行設定 (以下のいずれか)	新規設定
Oracle	install_dir¥server¥bin¥ukor8dttc22.dll install_dir¥server¥bin¥ukor8dttc23.dll install_dir¥server¥bin¥ukora24.dll install_dir¥server¥bin¥ukora26.dll	WBIMB¥bin¥ukora95.dll
Sybase	install_dir¥server¥bin¥ukase22.dll install_dir¥server¥bin¥ukase23.dll install_dir¥server¥bin¥ukase24.dll install_dir¥server¥bin¥ukora26.dll	WBIMB¥bin¥ukase95.dll

# (参考) 移行前タスク : DB接続設定変更

## ■ UNIX/Linux環境

- ◆ IIB V8以降、データ・ベースへのODBC接続にunixODBC ドライバー・マネージャーを使用
  - ACE V12 提供の2つのサンプルODBCファイルを基にODBC設定ファイルを作成
    - ✓ odbc.ini
    - ✓ odbcinst.ini
    - ✓ install\_dir/server/ODBC/unixodbc 下に提供
  - 必要な定義を追加したODBC設定ファイルをODBCINI/ODBCSYSINI環境変数で指定
    - ✓ odbc.ini へのパスをODBCINIで指定
    - ✓ odbcinst.ini へのパスをODBCSYSINIで指定

# (参考) 移行前タスク : DB接続設定変更

- ◆ XAリソース・マネージャーのSwitchFile 設定の変更
  - キューマネージャーのqm.ini ファイルでSwitchFileの設定を変更

ODMS	現行設定 (以下のいずれか)	新規設定
Oracle	SwitchFile=UKor8dtc22.so SwitchFile=UKoradtc22.so SwitchFile=UKor8dtc23.so SwitchFile=UKoradtc23.so SwitchFile=UKoradtc24.so SwitchFile=UKoradtc26.so	SwitchFile=UKoradtc95.so
Sybase	SwitchFile=UKasedtc22.so SwitchFile=UKasedtc23.so SwitchFile=UKasedtc24.so SwitchFile=UKasedtc26.so	SwitchFile=UKasedtc95.so