Magic xpi 4.x

開発手法

プロジェクト開発とライフサイクル

OUTPERFORM THE FUTURE™

The information in this manual/document is subject to change without prior notice and does not represent a commitment on the part of Magic Software Enterprises Ltd.

Magic Software Enterprises Ltd. makes no representations or warranties with respect to the contents hereof and specifically disclaims any implied warranties of merchantability or fitness for any particular purpose.

The software described in this document is furnished under a license agreement. The software may be used or copied only in accordance with the terms and conditions of the license agreement. It is against the law to copy the software on any medium except as specifically allowed in the license agreement.

No part of this manual and/or databases may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying, recording or information recording and retrieval systems, for any purpose other than the purchaser's personal use, without the prior express written permission of Magic Software Enterprises Ltd.

All references made to third-party trademarks are for informational purposes only regarding compatibility with the products of Magic Software Enterprises Ltd.

Unless otherwise noted, all names of companies, products, street addresses, and persons contained herein are part of a completely fictitious scenario or scenarios and are designed solely to document the use of Magic xpi.

Magic® is a registered trademark of Magic Software Enterprises Ltd.
Btrieve® and Pervasive.SQL® are registered trademarks of Pervasive Software, Inc.
IBM®, Topview™, iSeries™, pSeries®, xSeries®, RISC System/6000®, DB2®, and WebSphere® are trademarks or registered trademarks of IBM Corporation.
Microsoft®, FrontPage®, Windows™, WindowsNT™, and ActiveX™ are trademarks or registered trademarks of Microsoft Corporation.
Oracle® and OC4J® are registered trademarks of the Oracle Corporation and/or its affiliates.
Linux® is a registered trademark of Linus Torvalds.
UNIX® is a registered trademark of UNIX System Laboratories.
GLOBEtrotter® and FLEXlm® are registered trademarks of Macrovision Corporation.
Solaris™ and Sun ONE™ are trademarks of Sun Microsystems, Inc.
HP-UX® is a registered trademark of the Hewlett-Packard Company.
Red Hat® is a registered trademark of Red Hat, Inc.
WebLogic® is a registered trademark of BEA Systems.
Interstage® is a registered trademark of the Fujitsu Software Corporation.
JBoss™ is a trademark of JBoss Inc.
Systinet™ is a trademark of Systinet Corporation.
Portions Copyright © 2002 James W. Newkirk, Michael C. Two, Alexei A. Vorontsov or Copyright © 2000-2002 Philip A. Craig
GigaSpaces, GigaSpaces eXtreme Application Platform (XAP), GigaSpaces eXtreme Application Platform Enterprise Data Grid (XAP EDG), GigaSpaces Enterprise Application Grid, GigaSpaces Platform, and GigaSpaces, are trademarks or registered trademarks of GigaSpacesTechnologies.

Clip art images copyright by Presentation Task Force®, a registered trademark of New Vision Technologies Inc.
This product uses the FreeImage open source image library. See http://freeimage.sourceforge.net for details.
This product includes software developed by the Apache Software Foundation (http://www.apache.org/).
This product includes software developed by Computing Services at Carnegie Mellon University (http://www.cmu.edu/computing/). Copyright © 1989, 1991, 1992, 2001 Carnegie Mellon University. All rights reserved.
This product includes software developed by the OpenSSL Project for use in the OpenSSL Toolkit (http://www.openssl.org/).
This product includes software that is Copyright © 1998, 1999, 2000 of the Thai Open Source Software Center Ltd. and Clark Cooper.
This product includes software that is Copyright © 2001-2002 of Networks Associates Technology, Inc All rights reserved.
This product includes software that is Copyright © 2001-2002 of Cambridge Broadband Ltd. All rights reserved.
This product includes software that is Copyright © 1999-2001 of The OpenLDAP Foundation, Redwood City, California, USA. All Rights Reserved.

All other product names are trademarks or registered trademarks of their respective holders.

Magic xpi

目次

第 1 章: Magic xpi 入門

Magic xpi スイートは、企業組織のために最大の統合機能を提供するように設計された統合ツールです。
Magic xpi インテグレーションプラットフォームは、ビジネスプロセスの効率を最大化することにフォーカスして、多様なアプリケーション、システム、およびデータベースの統合を可能にしている、SOA(Service Oriented Architecture)と BPM(Business Process Management)機能を提供する EAI(Enterprise Application Integration)ソリューションです。

Magic xpi のコンセプト

Magic xpi スイートは、接続性やデータ操作、ビジネスロジック、および監視などのすべての内容の処理を含む、包括的なビジネス主導の統合フレームワークを提供する、プロセス指向の複合アプリケーションのツールセットです。

Magic xpi は簡単に自動化されたビジネスプロセスを設計し、ビジネスプロセスで定義されたビジネスロジックに従い、フローとして実行させることができます。

フローを構成するために定義済みのサービスやアダプタ、トリガーが提供されます。また、提供されたコネクタビルダーを使用することで、提供されているツールセットに加えて自由に機能を追加することができます。

Magic xpi は以下のソリューションを提供します:

- プロジェクトの設計
- 完全な統合機能 (コネクタとアダプタ)
- ビジネス統合の自動化 (ロジック)
- データ変換 (コンバータ)
- 実装時のオプション (異なる場所にある Magic xpi サーバに対して)
- リアルタイムでのトレースと管理機能 (モニタリング)

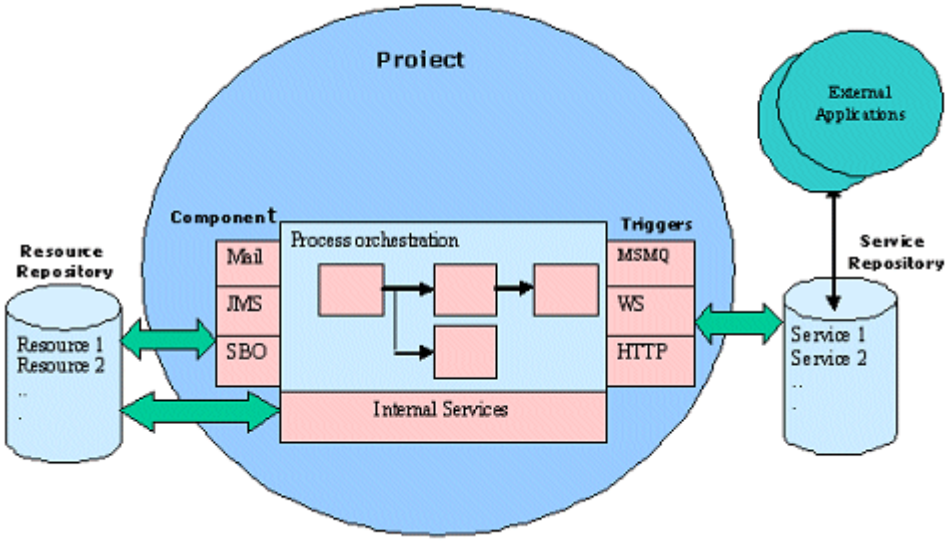
SOA を実現するツールとしての Magic xpi

成長する SOA を考慮して統合プロジェクトを実装することは、 新しいことではなく、その広範囲に及んだ利用内容は常に統合しています。Magic xpi プロジェクトを設計し、実装する際に考慮する必要のある要素の 1 つが、SOA の概念にどのように反映し適合していくのかということです。これについては以下で説明します。

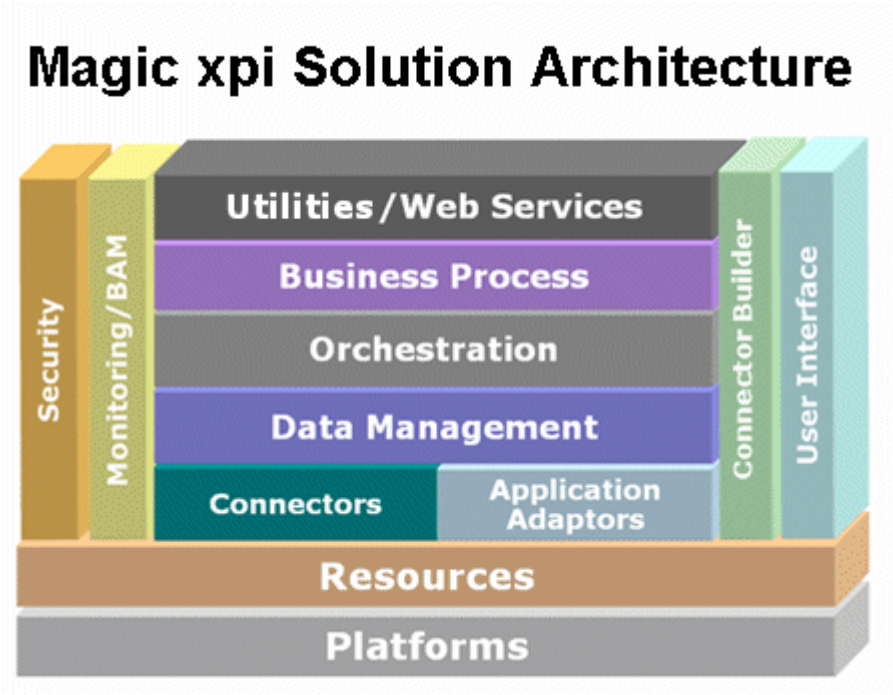
SOA は、インタフェースやインタフェースの実装、およびインタフェースの呼出に関するトポロジーを構築するソフトウェアアーキテクチャです。従って、SOA は再利用やカプセル化、インタフェース、最終的にはビジネスの俊敏性を目的としています。企業の俊敏性という必要性を満たすため、SOA の実行は以下のような中心的な原則を持っています:

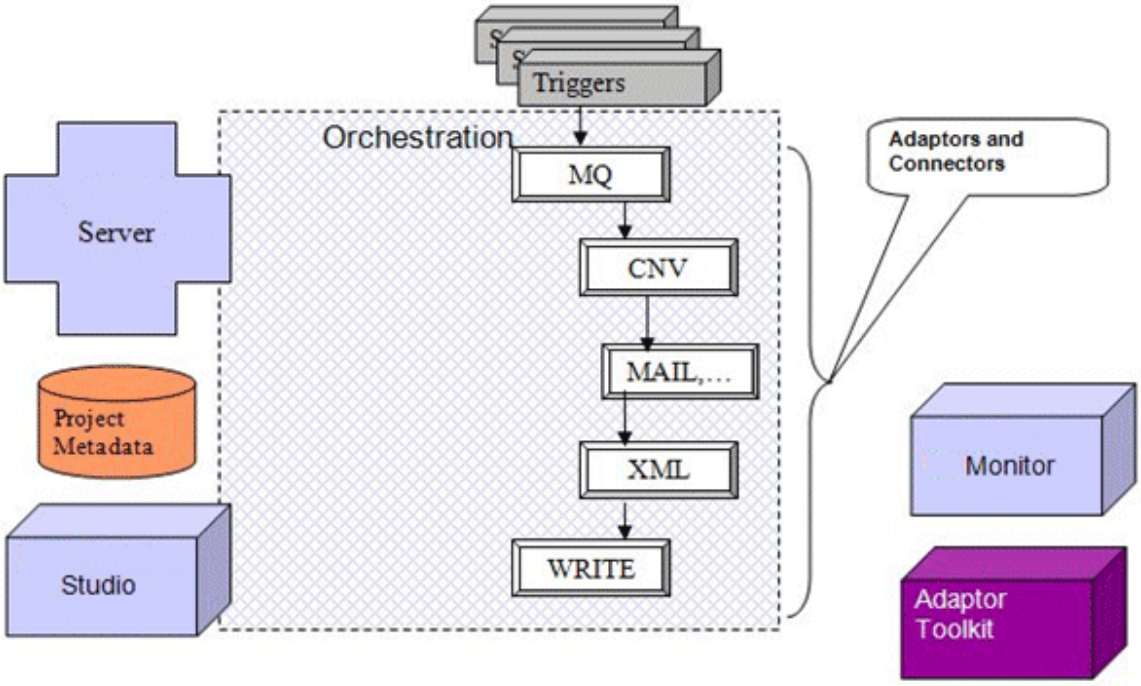
- 既存のテクノロジにもとづいたサービスの利用
- ビジネスが予測不可能なイベントを迅速に検出し、報告し、対応できるようなイベント駆動型アーキテクチャ
- プロセスの設計と統合を改善するビジネスプロセス管理(BPM)機能

以下のスキーマは、組織の中での SOA の考え方についての青写真を表しています:



SOA の実装は、既存の IT システム上でサービスと複合アプリケーションの階層を築くことができます。このシステムは、アプリケーションと DB レベルでエンドユーザから切り離され、統合された GUI を使用して必要な時にユーザに提供されます。





Magic xpi の根本的な設計思想は、ビジネス関数のカプセル化とサービスとしてのビジネスプロセスの公開に基づいています。Magic xpi の中の SOA サポートは、ビジネスプロセスのモデリングレベルから、統合フローとコンポーネントまで提供されます。

Magic xpi のオープン アーキテクチャー

Magic xpi サーバは、統合されたビジネス処理環境を背景としたエンジンです。サーバは、戦略的なビジネス統合ソリューションの効率的な運用を可能にする非常に信頼のおける、拡張性のあるプラットフォームを提供します。

Magic xpi のオープンアーキテクチャは、以下の機能を利用することができます:

プラットフォーム

Magic xpi サーバは、Windows や iSeries/IBM i(旧 Systemi)などのプラットフォーム上で実行させることができます(注：日本では Windows のみ提供)。これによりユーザは以下の事ができます:

- Windows プラットフォーム上でプロジェクトを開発し、異なるプラットフォーム上で変換処理を行うことなく実行させることができます。
- 拡張性とロードバランシングの機能を提供することで、異なるプラットフォーム上の異なるサーバ間でプロジェクトを分割させることができます。
- 各プロジェクトにトポロジーやビジネスプロセス、および統合フローを含めることで、マルチプロジェクト環境を構築することができます。

データベース

Magic xpi は以下のデータベースに接続することができます：Oracle, MSSQL, DB2/400, DB2, Pervasive.SQL, その他 ODBC をサポートしているデータベース。

ユーザ コンポーネントとアダプタ

包括的なツールセットに加え、ユーザが新しいコンポーネントとアダプタを定義することを可能にする組み込まれたコネクタビルダーを提供することによって、Magic xpi はその機能を拡張することができます。コネクタビルダーは、Magic xpa を使用したり、既存の Magic のコードや Java のクラスを Magic xpi のプロジェクトで利用できるコンポーネントとしてラッピングすることで､新しいコンポーネントコードを開発することができます。

動作環境

Magic xpi の動作環境は、中心となる環境リポジトリに保存されます。この内容は XML ファイルで管理されています。外部の XML ツールを使用して更新することもでき、複数のプロジェクトで共有させることができます。プロジェクト内で使用されたすべてのコンポーネントとサービスは、同じ環境リポジトリファイルをアクセスします。プロジェクトのライフサイクルの異なる段階で異なるファイルを使用する場合、プロジェクトコードにアクセスすることなく環境設定を変更することができます。

Magic xpi プロジェクト構築用ブロック

以下の表は、各 Magic xpi プロジェクトの主要なブロックの一覧です。各インテグレーション プロセスは、これらの項目全てに関係するわけではありません。

構築ブロック	説明
リソース	サーバや DBMS、アプリケーション、Web サーバ、およびプロジェクトの IT 環境におけるすべてのリソース定義。
サービス	Magic xpi は、フローを外部システムから呼び出すための方法についての定義を公開しています。サービスには、Web サービス、EJB、HTTP、およびその他が含まれます。
リポジトリ	コンポーネントやサーバ、アプリケーション、および変数を含むプロジェクト関連のリポジトリ。
ビジネスプロセス	Magic xpi は、ビジネスプロセスのモデリング機能を提供します。ここにはビジネス関数や活動、ビジネスルールやリーソス、および性能測定が含まれます。
インテグレーションフロー	インテグレーション フローは、ステップとフロー編成と呼ばれるステップの実行順序を制御するロジックの集まりです。インテグレーション フローは、Magic xpi サーバによって実行されます。
コンポーネント インスタンス	コンポーネントインスタンスは、インテグレーション フローのシーケンスの手順またはトリガーとして動作します。フローコンポーネントは、接続や適応、変換、処理といった機能を持たせることができます。
トリガー	トリガーは、HTTP または Web サービスの呼び出しなどの外部処理を実行したり、スケジューリングなどのフロー編成のための特定のフローを実行させるために使用されます。
Magic xpi ユーティリティ	Magic xpi サービスは、スケジューリングやパブリッシュ/サブスクライブ、フローの呼出などのフローロジック編成機能を提供しています。また、フロー データのようなデータを取り扱うサービスも提供します。
マッピング	データマッパは、データ操作や変更、変換、および拡張機能を提供している GUI ツールです。

第 2 章: Magic xpi プロジェクト ライフサイクル

Magic xpi プロジェクトのライフサイクルには、以下の行程が含まれています。:

- 分析
- アーキテクチャ上の考慮事項
- 設計
- 開発
- テスト
- 実装

最初の 3 つの工程はこの章で、残りは次章で説明します。

プロジェクトの分析

インテグレーション プロジェクトは、多くの既存のアプリケーションやベンダ、サーバ、インフラ用リソース等とやりとりします。

主要な課題には、以下のものがあります。:

- 最も重要な組織間をまたがる処理の定義。
- 既存の IT 資産の認識。これは新しい処理の一部となります。
- 必要なサービスとインタフェースの定義。

プロジェクト分析は、以下の行程によって成り立っています:

- 使用されるリソースの識別
- サービスの定義
- プロジェクト フローの定義

リソースの識別

リソース設定は、Magic xpi がプロジェクト実行中にアクセスする必要のある外部システムを定義します。

設定 ダイアログボックスの **リソース** セクションの目的は次のとおりです。:

- **外部リソースを設定するために 1 つの場所を提供する。** 電子メールの設定などの外部リソースは、プロジェクト内で異なるエリアで使用することができます。それが使用される各インスタンス内で毎回設定を定義することなく、リソースリポジトリ内に一回だけ設定するだけでよくなります。
- **プロジェクトの外でリソースを設定する機能を提供する。** 開発者がプロジェクトを開発する場合、実運用の環境に実装させるまで、テストを行うための開発環境を構築します。外部にリソースを設定することができれば、開発環境から運用環境へのよりスムーズな移行が可能になります。これは、プロジェクトを変更することなく、環境設定のみを行えばいいという有利な点があります。

- **以前に定義された環境設定を再利用する機能を提供する。** 複数のプロジェクトに対応する場合に、電子メールの設定など継続的に使用されるような様々なリソースがあります。毎回各プロジェクトのために、これらの設定を再定義する代わりに、以前に定義された設定を再利用し、このプロジェクトに関係する内容のみを変更する方が作業は簡単です。

Magic xpi は、MSMQ や WebSphere、および JMS に対するメッセージング機能と Web サービスなどの追加設定がいらないリソースタイプを提供します。これらのあらかじめ決められたリソースタイプに加え、ユーザは、コネクタビルダーを使用して、追加のリソースタイプを定義することができます。

電子メールや MSMQ などの外部リソースにアクセスしているすべてのコンポーネントは、適切なリソースタイプにもとづいたリソース定義を含めなければなりません。コンポーネントは、それらの設定を内部に保存せず、リソースリポジトリ内のリソースに保存する必要があります。

サービスの定義

設定 ダイアログボックスの **サービス** セクションには、プロジェクトが外部に公開するサービスと、Magic xpi を呼び出すためのインターフェイスが含まれています。

サービスは、通常 Magic xpi のフロー内のトリガーに対応しています。外部リソースにもとづいた全てのトリガー定義は、適切なサービスタイプにもとづいたサービス定義を含まなければなりません。

リソースと同じように、Magic xpi は EJB や電子メール、および FTP などのあらかじめ決められたサービスタイプ定義を提供しています。これらのサービスタイプに加えて、ユーザは、コネクタビルダーを使用して、追加のサービスタイプを定義することができます。

フローの定義

ビジネスプロセスは、実行可能なフローに分解する必要があります。各フローは、活動のサブプロセスまたはグループとして表すことができます。フローは、ステップとフロー オーケストレーションと呼ばれるステップの実行の順番をコントロールするロジックの集まりから成り立っています。

フローとは、ビジネスプロセスの物理的で実行可能なレベルのことです。

1 つ以上のフローが 1 つのビジネスプロセスを表すことができます。

このフローには、エラー管理やセキュリティの問題を含む各プロセスのビジネスロジックが含まれます。

次のステップでは、フローを開発する際に考慮すべき主なトピックを紹介します。

フローの設定

管理しやすく、柔軟で、ダイナミックなフローの作成を試みることを推奨します。

例えば、いくつかの直線的でリニアなステップを呼び出す必要がある場合、1 つの長いプロセスを管理するのではなく、ループを作成することをお勧めします。

変数の使用

Magic xpi では、フローで使用できる変数を定義できます。 これらの変数は、フローロジックを管理し、あるコンポーネントから別のコンポーネントにデータを移動するために使用されます。 また、フロー内のステップを調整する場合にも便利です。

Magic xpi は以下のタイプの変数をサポートします。:

フロー変数 – フローごとに定義され、フローロジックおよびフローコンポーネント間のデータ転送の管理を可能にする変数。 各フローは、内部アクティビティに基づいて個別にフロー変数を定義することができます。

フロー変数を使用するシナリオは以下の通りです:

- フローの次ステップでの追加処理のためにデータを格納します。
- ビジネス上の決定を行う際、変数の値に基づいて、別々のルートを決定する。
- フロー変数が期待した値でない場合、エラーフローを呼び出します。

コンテキスト/モデル変数 – コンテキスト(スレッド)に関連する変数。 コンテキスト変数は、新しいコンテキスト(フロー)が開始されたときに作成され、コンテキストが存続している間は有効です(アクティビティを終了するために複数のフローにまたがっていても有効です)。

コンテキスト変数の特別なセットとして、事前に定義されている変数があります。 UserString、 UserCode、 UserBlob、 UserXML です。これらは、コンポーネント内で定義されるデフォルトの変数として使用され、同じタイプのどのようなユーザ定義変数とも変更することができます。ただし、以下の 2 つの例外があります。:

- コンポーネントの XML インターフェイスは常に UserXML を使用します。
- 全てのメッセージトリガーはそのメッセージ内容を UserBlob に格納されます。

コンテキスト変数を使用する場合のシナリオは次のとおりです。:

- 宛先の中でフロー呼び出しオプションを使用して、前のフローから呼び出されたデータマッパを使用してデータを処理します。
- メインプロセスで受信したデータに基づき、サブプロセスでデータを検証します。

グローバル変数 – これらの変数は、同じプロジェクト内のすべてのフローで利用可能で、同じメモリアドレスを共有します。 1 つのフロー内のグローバル変数を変更すると、他の全てのフローに影響します。

グローバル変数を使用するシナリオは以下の通りです:

- プロジェクトの最初に環境変数(論理名)に保存された値でグローバル変数を初期設定します。性能を向上させるために、プロジェクトで環境変数の代わりにグローバル変数を使用するようにしてください。

ビジネスプロセス変数 – これらの変数は、Magic xpi V3 で利用できるようになりました。これらは、同じビジネスプロセスにおけるすべてのフローで利用可能です。1 つのフロー内のビジネスプロセス変数を変更すると、同じビジネスプロセス内の全てのフローに影響します。

環境変数 – これらの変数は、Magic xpi プロジェクト内で外部の IT 環境特性を定義するために使用されます。環境変数によって、異なる実行環境へのプロジェクトの移行をスムーズに行うことができます。

環境変数リポジトリに入力された実行値に基づいて、実行時に環境変数の内容を変換することで、Magic xpi の移植性を高めることができます。環境変数の実行値は実行時に変換されますが、ほとんどが定数として使用されます。環境変数の明示的な使用は、動的に動作内容を変更したい場合で、式が使用できない場合のみ行ってください。

参考:

- Magic xpi V3 から、グローバル変数やモデル/コンテキスト変数、フロー変数を固定接頭辞を付加して作成します。これは、プロジェクト内で関連する変数を識別し、配置する処理を支援します。例えば、フロー変数名：MyVar は、フローまたはコンテキスト変数のどちらであるかを示すために F.MyVar または、C.MyVar と命名されます。
- プロジェクトの初期設定段階で、環境変数によってグローバル変数が初期設定されていることを確認してください。性能向上のために、プロジェクト間で利用できるに環境変数の代わりにグローバル変数を使用してください。

アーキテクチャー上の考慮事項

Magic xpi プロジェクトのアーキテクチャーは、Magic xpi サーバのパフォーマンスに大きな影響を与えます。アーキテクチャー上の問題として考慮するべき内容は以下の通りです。:

- 複数の Magic xpi サーバが同じマシン上で実行している場合、1 つの Magic xpi サーバで実行されている並行稼動数を最適な値にしてください。
- 複数の Magic xpi サーバを異なるマシン上で実行させることができます。
- 内部のデータベースは、Magic xpi サーバと同じマシンにまたは別のマシンに配備することができます。
- 外部のプロジェクトの入出力性能には、以下の項目が含まれます。:
    - ミドルウェアや HTTP、および Web サービスなどの外部のアプリケーションへの接続性
    - 外部のアプリケーション性能と制限によるボトルネック処理
    - 通信速度

注: 通信チャンネルと同様に、これらのアプリケーションを適切に選択し、調整することで性能を改善することができます。

- Magic xpi ロギング設定

リモートサービス(アクティビティ ログ、ODS、PSS)の使用やルーティングは、全体の性能を改善する可能性のあるアーキテクチャ上の追加のソリューションです。

注: 統計情報の収集、アクティビティ ログ出力等内部データベースに出力する情報を制限することで、Magic xpi サーバのスループットを向上させることができます。

パフォーマンスは負荷および応答時間の定義/制約に大きく依存します。これらの定義に基づき、以下にあげる対策をとる必要があります。:

- プロセスをサブプロセスに分割します。
- 可能な限り素早く、効率的に受け取った情報を処理するために疎結合の概念を採用してください。たとえば、リモートアプリケーションやデータベースを呼び出すスレッドを持ち続けず、処理を実行する新しいフローを呼び出し、終了時にメインのフローに戻るようにします。
- 同期処理から非同期処理へ処理モードを切り替えてください。
- プロジェクトを複数のサーバに分散してください。

参考: 同期処理を複数の非同期処理に分割する場合、メイン処理を継続するために、**Wait for completion(処理完了を待つ)**サービスを使用することができます。

疎結合アーキテクチャー

プロセスを設計する場合、ユニットを構築し、ユニット間で可能な限り緩やかな接続を作成するために特別な注意を払ってください。Magic xpi では、このような疎結合コンセプトを実装するツールとして、以下を提供しています。:

- マルチプロジェクト モード – この機能により同一のスペースで異なるプロジェクトを実行することが可能になります。各プロジェクトは個別の設定を持ち、互いに完全に独立して動作します。このマルチプロジェクト機能は Magic xpi プロジェクト内の各アクティビティを以下のように分離することができます。:
    - 各プロジェクトを個別に開始/終了することが可能。
    - (プロジェクトの)開発やテスト、およびアップグレードのプロセスは、スタンドアロンでの運用として各プロジェクト内で実行させることができます。
    - 各プロジェクトは、異なる管理者やビジネスユニットによって個別に監視することができます。

- 各プロジェクトに関連する項目やエラーなど、異なる環境設定を行うことができます。

- フロー呼出し – メインプロセスからサブプロセスを呼び出すことができます。新しいフローは、Magic xpi エンジン内で新しいスレッドを消費します。同じコードまたはプロセスを書き換える必要性を排除することで、フローはプロジェクト内のいろいろな場所から呼び出すことができます。
- パブリッシュ&サブスクライブ – トピックがパブリッシュされると、このトピックをサブスクライブする全てのプロセスは、直ちに起動されます。例えば、注文がパブリッシュされた場合、在庫システムはその製品の入手可能性を更新することができ、CRM システムは顧客データを更新することができて、財務システムは顧客カードを更新することができます。3 つのシステムの間には、接続に関する定義は必要ありません。新システムが導入されると、同様に関連情報にサブスクライブさせることができます。

ライセンスに関する考慮事項

フローの並行実行と Magic xpi のライセンス

以下のような場合にフローは並行実行されます:

- 非同期でフローが呼び出された場合
- パラレルまたはスタンドアロン分岐を使用した場合
- トリガーを使用して、外部からフローを呼出す場合

並行実行する各フローは、Magic xpi サーバのスレッドが必要とします。従って、Magic xpi の同時並行スレッドのライセンスを必要とします。

並行実行するフローを呼び出すリクエストの数は、入手可能な Magic xpi サーバの並行スレッドライセンスの数によって制限されます。

ライセンスの上限に達した場合、フロー起動リクエストはキューの中で待ちます。Magic xpi サーバが有効なライセンスを持っている場合、リクエストは処理されます。

注: システムを運用する上で十分なスレッドを使用しない場合、フローの呼び出しで遅延が発生する場合があります。これは Magic xpi サーバの全体的なスループットに影響します。

予約済みライセンススレッド

マルチプロジェクト モードで実行する場合、適切にプロジェクトを実行するために、特定のスレッド数を確保する必要がある場合があります。定義は、**設定** ダイアログボックスの**プロジェクト環境** セクションの **一般設定** のプロジェクトの**予約済みライセンススレッド(Reserved License Threads)**エントリで行う必要があります。あるいはプロジェクトの ifs.ini ファイルに直接以下のように指定します。:

[MAGICXPI_GS]ReservedLicenseThreads = <必要な予約済みスレッド数>

注: このフラグが 0（デフォルト値）に設定されている場合、プロジェクトはライセンスを予約せず、ライセンスプールで使用可能なライセンスを使用します。

ライセンスメカニズム

Magic xpi はフローティング・ライセンス・メカニズムを採用しており、オプションとしてプロジェクトのスレッド数を確保する機能を提供しています。

マルチ プロジェクト環境でサーバ間のライセンスを共有することによりライセンスの最適利用

をすることができます。したがって、特定のプロジェクトがアイドル状態のとき、他のプロジェクトはライセンスを利用できます。

注: ライセンスはホスト単位で固定されるので、ネットワーク上に(ライセンスの **HOSTID** フラグと一致する)プロジェクトを実行するサーバがなければなりません。ホスト ID 生成に関する更なる情報は license's zip 内の **Magic_License_Activation_Procedure.pdf** を参照してください。

処理モード

インテグレーションプロジェクト内の各ステップとサブプロセスは、いくつかの処理モードの 1 つとして処理されます。処理モードの定義は、(利用可能なスレッド数を含めた)リソースの入手可能性と同様にデータの潜在的な考慮事項と密接に関連しています。

リニア(同期モード)

リニアのステップは、フロー内で定義された順番とレベルに依存し、順番に実行されます。1 つのリニアなステップだけが各レベルで実行することができます。リニアなステップは、同じレベルで定義された他のパラレルステップ(あるいはステップ)が完了するまで待つことになります。

パラレル(非同期モード)

パラレルのステップは、同じレベルに定義された他のステップと同時に実行させることができます。フロー変数やエラー管理などの内部の定義は、親のステップから割り当てられます。**Wait for completion(処理完了を待つ)** オプションは、必要に応じて、プロセスを分割し、再グループ化する能力があります。

スタンドアロン

スタンドアローン処理モードを持つコンポーネントは、(並行処理のステップと同じ方法で)フロー内の別のステップと同時に実行することができます。しかし、これはフローと無関係に実行されます。この処理が実行されると、**Wait for completion(処理完了を待つ)** オプションは無効になります。

次ステップの評価ロジック

同じレベルのサブステップが複数ある場合、Magic xpi は以下のようにして次に実行させるステップを決めます。:

1. 最初に、パラレルまたはスタンドアロンのステップが存在している場合、これらすべてのステップはステップ条件に基づいて並行に実行されます。
2. 次に、条件付きのリニアステップの条件が評価され、左から右の順で実行されます。
   - 「True」と評価された条件を持つ最初のリニアなステップが実行されます。
   - 条件を持つリニアなステップが存在していないか、全てのリニアなステップの条件値が「False」の場合、条件が定義されていない一番左のリニアなステップが実行されます。
   - 上記のステップが見つからない場合、リニアなパスは終了します。

コードの再利用性

SOA プロジェクトの重要な課題の 1 つは、既存のコードを再利用することです。既存のコードを再利用するほど、インテグレーション プロジェクトを作成するために必要な投資が減り、プロジェクトのリスクも減少します。

複数のモジュールで共有される特定のタスクを実装するフローが独立しており、プロジェクトの複数の場所から呼び出されるように、プロジェクトを設計することをお勧めします。 これには、エラー処理、データキャプチャ、クリーンアップフローなどが含まれます。

さらに、既存のコードを Web サービス、または COM や HTTP などの他のインターフェイスとして公開することもできます。

長期間トランザクションの管理

インテグレーション プロジェクトでは、しばしばプロセスの変更が数日、数週間または数ヶ月に及ぶ可能性のある長いトランザクション管理が必要であり、複数のセッションが必要になることがあります。 ほとんどの場合、長いトランザクション管理は人間とのやりとりに密接に関連しています。

Magic xpi からタスクを送信し、回答があったときに Magic xpi フローをトリガーする共通フローメカニズムを使用して、単純な長いトランザクションを管理できます。

より複雑な状況では、少なくとも以下の内容を定義する必要があるテーブル(XML またはデータベーステーブル)を設計することができます。:

- プロセスのユニークな ID
- 重要なステップ
- 各重要なステップに対するステータス定義

プロセスのオーケストレーションはこのステータス定義に基づき Magic xpi により実行されます。

マン・マジンに基づく相互作用

インテグレーション プロジェクトの中には、人間とマシンの間の相互作用を混在させる必要があるものがあります。マシンにもとづいた相互作用は、各システムに対応する時間間隔やエラー管理、冗長性などの定義の集まりを含めて、Magic xpi サーバによって完全に管理することができます。人間にもとづいた相互作用は、より柔軟で、管理しにくいタスクになります。

Magic xpi は、長時間トランザクションの処理を実装する方法を使用することで、人にもとづいた相互作用を取り入れ、統合することができます。

人にもとづいた相互作用の管理の別の部分は、Magic xpi の中から情報を取得し、ユーザ用の画面に表示させることで対応しています。Magic xpi は、いくつかの作業モードをサポートしています。:

- 電子メールにもとづいた人的相互作用の管理：電子メール用のモードには、タスクを受け付けたり、拒否したり、Magic xpi フローに自動的に応答を渡したりするためのプッシュボタンを含めることができます。
- 同じタスクを扱うことができる何人かのユーザを持った処理は、人的相互作用が、ユーザグループをまたがって作業量を均一化することにフォーカスを置くことになる場所で、特定の人的ワークフローに対するソリューションを必要とします。Magic xpi は、洗練されたフォームを作成したり、パラメータ設定にもとづいたどのような時点での仕事量でも管理する能力を含んだワークフローのソリューション(W4：※日本では W4 はサポートされていません)を提供します。

キー プロセス インディケータ (KPI)

プロジェクトを設計する場合、常にビジネス上の必要性と成功要因を留意しておく必要があります。これらは KPI として、プロセスとフローの設計で実装する必要があります。Magic xpi は、プロジェクトのどのようなポイントに対してもビジネス活動メッセージ(BAM)を定義することができます。BAM メッセージの定義は簡単ですが、高い柔軟性があります。実行中のプロジェクトからデータを監視スクリーンに含めることが可能になります。

KPI を定義することで、実行中のプロジェクトを完全にコントロールすることができます。

セキュリティの問題

プロセスが明確になり可視化されると、以下のようなセキュリティ上の考慮が必要になります。:

- リソースへのアクセス – FTP サイトやサーバ、アプリケーション サーバ、DBMS など
- Web サービス セキュリティに対するサポート
- データ変換の暗号化の必要性 – ファイル(XML)やデータベースに対して
- ユーザ認証の必要性 – 内部機構を使用して LDAP サーバから行う

ビジネスプロセスの描画に戻り、関連する場所の全てのセキュリティ問題をマークし、強調されていることを確認してください。

データの暗号化/復号化が必要であれば、これらのステップに関連するプロセスまたはサブプロセスに追加する必要があります。

参考:	データの暗号化/復号化などのタスクを処理するサブルーチンを作成しておくことを推奨します。

エラーの管理

エラー動作の定義は、統合プロジェクト内で重要な定義です。より多くのエラー状況を認識し、処理するための自動ルーチンを定義することで、プロジェクトはより頑強で、生産性の高いものになります。

次にビジネスプロセスに戻り、潜在的なエラー状況を考慮する必要があります。例えば、最初のステップがポータルから XML ファイルを受け取っている場合は、潜在的なエラーには、以下の内容が含まれることになります。:

- ファイル内の未確認のデータ
- XML ファイルが、定義されたスキーマと合っていない
- XML ファイルが、デフォルトのデータ設定を含んでいない
- 必須データが入力されていない

各エラー状況は、2 つのフェーズの定義を通して行う必要があります。:

- エラー検出 – エラー状況が宣言されるビジネスルールの定義
- エラー処理 – エラーが発生した場合の対処方法

Magic xpi のエラー管理メカニズムには、以下の内容がを含まれます。:

- エラー変数
- ステップ or フローのエラー動作
- エラーおよび例外処理フロー

- アボート フロー サービス – このサービスによって、フローの再起動が発生したり、定義されたエラーコードに基づいてエラーポリシーが実施されます。

*Magic xpi ヘルプ*内の Magic xpi 技術情報にある *Magic xpi のエラー処理* には、エラー管理メカニズムについての詳細情報があります。

パフォーマンスのガイドライン

メモリ使用量

Magic xpi サーバは、内部利用のために、プロジェクトとは別に初期メモリを必要とします。プロジェクトの読み込みや実行には、さらにメモリが必要です。必要なメモリ量は、以下の内容に依存します。:

- フロー および ステップ数
- 処理されたデータ(メッセージ データ)のサイズ
- 同時実行フロー数
- データ マッパー ステップの性質(タイプとマッピングの量)

Magic xpi サーバがプロジェクトを実行する場合、物理メモリを効率的に管理するために、OS がスワップ機能を使用します。

注: システムの物理メモリが不十分な場合は、OS は非効率的にメモリページの交換を行います。サーバ用のホストマシン上で実行されるプロセスの処理が遅くなるため、スループットが低下します。

マルチ CPU

Magic xpi サーバは、マルチ CPU のマシン上で実行させることができます。マルチ CPU 構成ではオペレーティングシステムはマシン上で実行されるプロセスに CPU リソースの割り当てを行います。

Magic xpi サーバのプロセスは、OS の CPU 割り当てに干渉しません。他の実行プロセスもまた CPU リソースをを使用します。これによって全体の CPU 割り当てに影響されます。

Magic xpi サーバを表すプロセスがマルチスレッドのため、異なる CPU が Magic xpi サーバプロセスの異なるスレッドを実行することになるかもしれません。これは Magic xpi ではなく、OS によってコントロールされ、OS とは独立しています。Magic xpi は指定されたマシン上で実行する CPU の数に制限をかけません。

ライセンスファイル内の同時リクエスト数は、異なるスレッドを実行する CPU に対して、同時リクエスト数を制限しています。

Magic xpi サーバは、他のアプリケーションやデータベースを実行しているサーバ上で実行させる場合もあります。このため、Magic xpi プロジェクトの全体の性能が、より多くの CPU を使用(追加)することで改善される場合があります。

注:
- Magic xpi サーバは、I/O に制限があります。従って、CPU を追加してもプロジェクトの性能が改善されない場合があります。
- プロセスにおけるボトルネックを捜すことで、プロジェクトの I/O 性能を強化することを推奨します。

オペレーティングシステムのセットアップ

サーバの全体的なパフォーマンスを向上させるために、次のオペレーティングシステムのリソースをチューニングすることができます。:

- CPU の割り当てとプロセスの優先度
- メモリの割り当て
- ファイル ハンドル
- TCP/IP 通信
- I/O リソースの割り当て

注: マシンのチューンアップを行わない場合、マシンのハードウェアとオペレーティングシステムの使用が非効率的になり、マシン全体のパフォーマンスに悪影響を与えることがあります。

プロジェクトの設計

前に述べたように、実装フェーズの前に、プロジェクトの低レベル設計をすることが不可欠です。

詳細なプロセス定義には、以下の主となる部品を含みます。:

- トリガー定義
- 実行フロー

トリガー

このセクションには、ビジネスロジックやフローを実行させるアクティビティやイベントが含まれています。フローは、1 つまたは複数のアクティビティまたはイベントによって起動され、関連するエントリ情報を提供する必要があります

トリガーは定義には次のものがあります。:

- トリガータイプ (1 つまたは複数のトリガでプロセスを呼び出すことができます):
    - HTTP
    - Web サービス
    - 新しいファイル(ディレクトリ, FTP)
    - 新しいイベント(Java, EJB, COM, etc.)
    - スケジューラ
    - Email
    - 新しいメッセージ(メッセージ キュー)
- 主データ フロー – 入力データを処理するために実行させる必要があるロジックとビジネスルールの定義
- データのフォーマット
- 頻度
- セキュリティと認証

処理フロー

ビジネスプロセスは、それぞれサブプロセスまたはアクティビティのグループを表すフローに分解されるはずです。 フローには、Magic xpi コンポーネントを使用して設定された 1 つまたは複数のステップが含まれます。

フローの呼び出しは、以下を使用して実行できます。:

- Magic xpi トリガー ：HTTP、Web サービス、ディレクトリスキャナ等
- フロー起動ユーティリティ – 親フローから子フローを呼び出す
- 自動起動 – サーバが起動された時に自動的に開始されるフロー
- 指定された時間、または一定の時間が経過後に開始されるようにスケジュールされたフロー
- PSS メカニズム

フローの基本的な定義には、以下を含める必要があります。:

- トリガー定義
- ステップ定義
- データ フロー定義:
    - ステップの一部として、データ フォーマットの関連する変換を含むデータ フォーマット
    - データ タイプ
    - データ ロジック

- ロジックのアクティブ化 – 既存 or 新規(情報の処理 or 既存手続の再利用)
- 変数定義 – データ処理を可能にするための関連した変数のリスト
- 処理モード(スタンドアロン, 同期, 非同期)
- エラー動作の定義
- リカバリー定義(セーブポイント)

フロー ステータス

Magic xpi プロジェクト内のフローを個々に無効／有効／非アクティブに切り替えることができます。:

- アクティブ / 非アクティブ – フローを非アクティブとしてマークすることによって、チェッカーのパフォーマンスを高めることができます。フローはチェッカープロセスや実行プロセスから除外されます。
- 有効 / 無効 – フローを無効にすると、プロジェクト実装時にフローは実行されません。

第 3 章: Magic xpi プロジェクトの開発

This chapter discusses main considerations when developing a new project, giving you best practices to fulfill your task efficiently and successfully.

データ管理の問題

この章では、新しいプロジェクトを開発する際の主要な考慮事項を説明し、タスクを効率的に成功裏に完成させるための最善の方法を提案します。:

- 別のアプリケーションやメッセージング キューなどの次の宛先に転送されるデータ
- 別のフォーマットに変換されるデータ
- 1 つ以上のテーブルと DBMS に保存されるデータ
- 他のアクティビティのための条件として使用されるデータ

Magic xpi は、データ管理を行う以下のようなメソッドを提供しています。:

- フローデータ ユーティリティ
- データマッパー
- フロー オーケストレーション メカニズム
- フロー起動ユーティリティ – 起動されるフローの変数に値を受け渡します

データマッピングのルール

以下のセクションでは、Magic xpi のデータマッパによって実行されるマッピングルールについて説明しています。

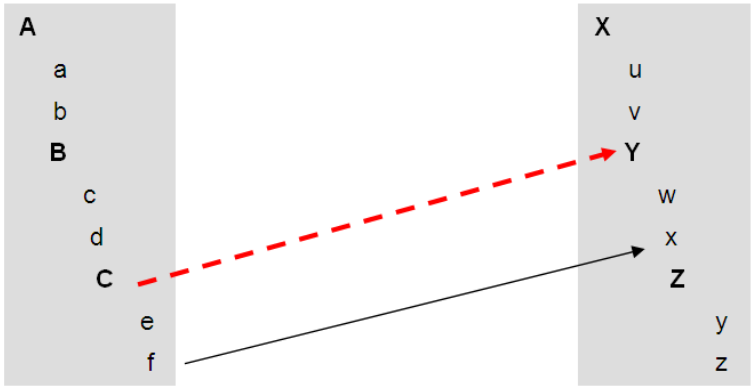
自動的な複合マッピング

単一要素が親の複合マッピングを行わずにマッピングされた場合は、常に Magic xpi はソースの最初の親と **Max occurrences>1** が定義された宛先要素の組合せに接続します。

f と **x** がマッピングされると、Magic xpi は、複合 **C** と複合 **Y** を自動的にマッピングします。

Magic xpi は、**C** のすべてのインスタンスのために **Y** のインスタンスを作成します。

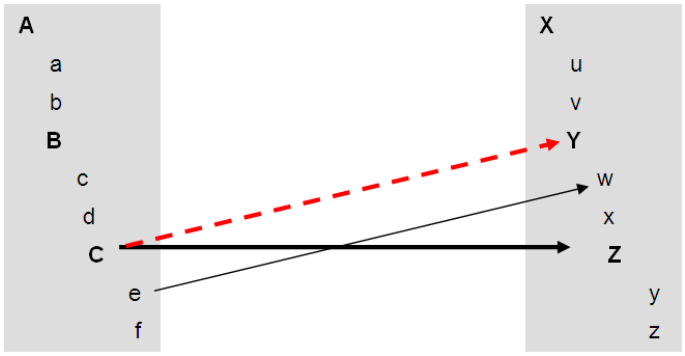
C の Max occurrences の値が 1 の場合、Magic xpi のデータマッパは、自動的に、1 より大きい Max occurrence を持つ親の 1 つにマップされます。すなわち **B** または **A** は、**Y** にマッピングされます。



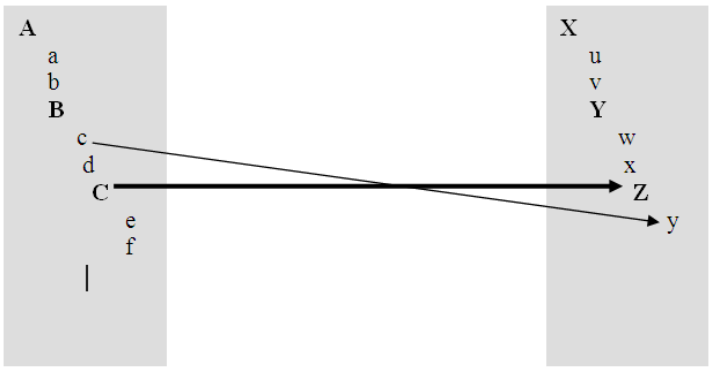
クロス マッピング

クロスマッピング定義は、ソースの要素のマッピングが複合マッピングと交差している時に行われます。

i. **アップクロス(Up crossing) :** このシナリオでは、**w** にマッピングされている **e** は、(**C** と **Z** がマッピングされている状態に追加して) 動的に **C** と **Y** をマップします。

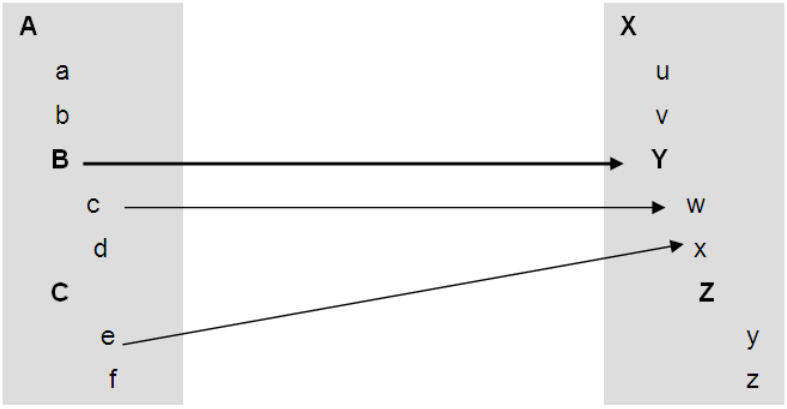


ii. **ダウンクロス(Down crossing) :** このマッピングでは、**y** は **C** のすべてのインスタンスのために **c** の値を取得します。**B** が複合的に発生した複合要素の場合、すべての **B** のために、**B** の配下のすべての **y** インスタンスにマッピングされた 1 つの c の値を持ちます。次の **B** のために、新しい **B** などの配下ですべての **y** インスタンスにマッピングされた **c** の新しい値を持ちます。



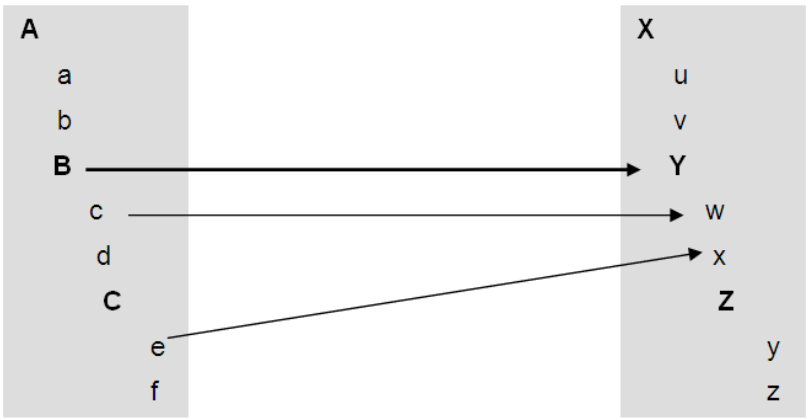
フラットな送り元から階層的な送り先へ

次のマッピングは、**C** の単一のインスタンスである場合にのみ有効です。これは、**e** のどのインスタンスが **x** のどのインスタンスにマップされるべきかを知ることができないためです。 **C** が複数発生している場合は、ソースフィルタを適用するか、または **C** のソース複製を作成して、単一オカレンスにする必要があります。



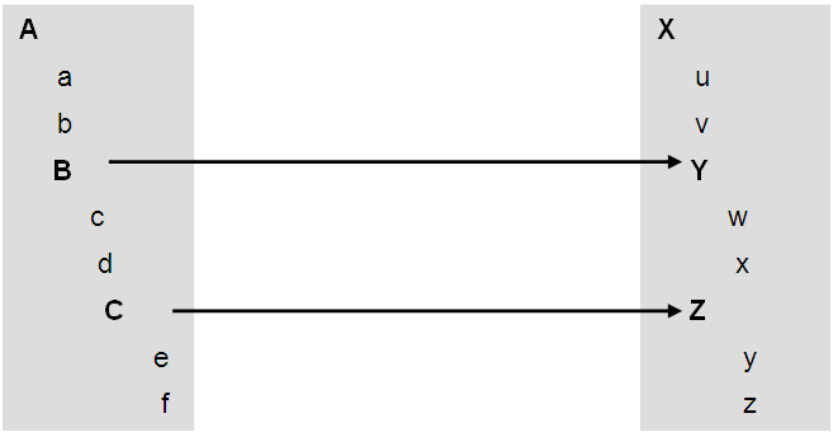
階層的な送り元から階層的な送り先へ

以下の状況は、**C** のインスタンスが **x** のどのインスタンスにマップされるべきかを知ることができないため、**C** が単一のインスタンスである場合にのみ可能です。 **C** が複数発生している場合は、ソースフィルタを適用するか、または **C** のソース複製を作成して、単一オカレンスにする必要があります。



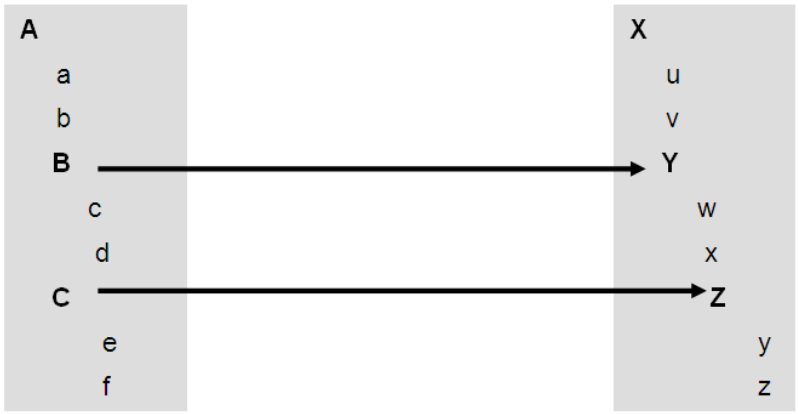
階層的な送り元からフラットは送り先へ

このシナリオは、階層構造から関連するものまで、1 対多の関係をブレイクダウンした内容で表しています。**Y** と **Z** は、それに応じて **B** と **C** の要素を得るフラットな構造です。各 **B** インスタンスは、**Y** にマッピングされるため、この **B** の子となるすべての **C** インスタンスは、合成要素 **Z** を追加します。このシナリオは、**c** または **d** と、**y** または **z** のマッピングを可能にします。条件が、**C** をフィルタにかけるための条件が使用されない場合、合成要素 **Z** は、**B x C** 回分 **C** で満たされます。



フラットな送り先から階層的な送り先へ

このシナリオでは、**B** と **C** は兄弟関係で、階層構造でマッピングされています。**B** の各インスタンスに対して、**Y** の 1 つのインスタンスを持ち、**C** のインスタンスの全てが **Z** にマッピングされています。ユーザは、ソースが必要な親に応じて分類されているかを確認する必要があります。



データ管理の最善の方法

データマッパは非常に強力なツールですが、時間とメモリのオーバヘッドを持っています。マッピング機能が必要な時だけ、データマッパを使用するようにしてください。例えば以下のように対応してください。:

- 変数を更新する場合は、データマッパの代わりにフローデータを使用します。
- 多くのレコードを処理する必要がある時だけ、データマッパの Call Flow オプションを使用してください。1 つのレコードだけを処理する場合は、フロー起動ユーティリティ(Invoke Flow)を使用してください。
- 一時的にデータを格納する必要がある場合、変数に保存されたフラットファイルを使用することは、ODS を使用することより効率的です。ODS はデータベースに保存されたテーブルであり、メモリ内のフラットファイルよりメモリを消費します。しかし、ODS は永続的で、障害が発生した場合に復旧させることができます。

独自コンポーネントの開発

Magic xpi を使用した場合、Magic xpi によって提供されない必要な機能を実行するために、独自のコードを追加する必要があるかもしれません。

Magic xpi スタジオのコネクタビルダーは、この機能を提供します。

ユーザ定義コンポーネントは、Magic xpa または Java を使用して作成することができます。ここでは、Magic xpa を使用して、ユーザ定義コンポーネントの作成方法について説明しています。

コネクタビルダー

Magic xpi は、ユーザがユーザ定義コンポーネントを作成したり修正することができるコネクタビルダーを提供しています。
Magic xpi によって提供された様々なコンポーネントと同様に、ユーザ定義コンポーネントは、独自のパラメータを持ち、フロー内にステップやトリガーとして使用できるユーザ定義メソッドとして公開することができます。

これらのメソッドは、メソッドと XML インタフェース内で公開されます。

詳細は *Magic xpi Help* テクニカルノートの*コンポーネント SDK* を参照してください。

Magic xpi のリカバリー ポリシーの利用

Magic xpi のリカバリー システムは、システムクラッシュやフローのタイムアウトのために消失したデータを回復させます。各フローに対して、プロジェクト開発中にリカバリー ポリシーを定義することができます。リカバリーポリシーは、**フロー特性**ダイアログボックスで定義します。

チェッカーとクロスリファレンス

チェッカーは、実行用パッケージを作成する際に自動的に実行されます。この段階で、プロジェクト上の総チェックを行うことができるため、メモリ内にすべてのフローが読み込まれます。

クロスリファレンス機能は、変数やフローなどの Magic xpi オブジェクトに対する参照関係を検索します。

これらのユーティリティは非アクティブ フローには実行されません。

第 4 章: プロジェクトのテスト

デバッグ

Magic xpi デバッガは、開発中の統合プロジェクトをテストしたり、リモート環境で稼働しているサーバをデバッグすることができる Magic xpi スタジオの一部です。デバッガは、プロジェクトサーバを起動したり、すでに実行しているプロジェクトサーバに接続します。デバッガは、プロジェクトの実行方法をコントロールしたり、実行フローの手順や変数、およびブレイクポイントを参照することができます。

デバッガを起動すると、プロジェクトがデバッグモードで自動的に作成されます。このプロセスには、すべてのアクティブなフロー上でチェッカーを実行させる処理も含まれています。開発中にプロジェクトのデバッグを支援するために、致命的エラーを含むフローがあっても、デバッガは実行可能なファイルを作成します。エラーを含んだフローは、実行しないものとしてマークされ、デバッグセッション中は無効になります。

デバッガーのステータス

デバッガには 2 つのステータスがあります。:

- 停止 – サーバが、コマンドを実行し終わると、停止モードになります。これは、コマンドが実行しておらず待ち状態であることを意味しています。これは初期状態です。
- 実行中 – デバッガは、現在 1 つ以上のフローを動かしています。

プロジェクト実行の制御

デバッグセッション中は、以下の方法でプロジェクトの実行をコントロールする事ができます。:

- **ブレイクポイント設定**
  ブレイクポイントは、有効なデバッグツールです。これは、コンテキストビューとアクティビティログを確認するために、指定されたステップ上でフローの実行を停止させます。

  ブレイクポイントに達すると、次のステップを実行する前に、スレッドや分岐などの実行されたコンテキストが全て停止します。ユーザがステップまたは継続を選択するまで、デバッガはその状態を維持します。ブレイクポイントは、変数や環境変数などに基づいたブレイクポイントを設定する条件設定も可能です。条件式は、プロジェクト内で使用される式と同じように定義します。ただし、この式は、プロジェクトソース外に保存されるため、クロスリファレンスやテキスト検索、およびチェッカーの対象にはなりません。

- **ステップ、フロー、ビジネスプロセスの一時停止**
  デバッグする際、問題をより正確に特定するために、特定の要素で中断したい場合があります。
  一時停止されたステップは無視され、ステップが実行されたかのように、デバッガは次のステップに進みます。

一時停止されたフローは実行されず、そのモードから解除されるまで「停止」とみなされます。
同様の方法は、フロー又は分岐を一時停止する際にも使用することができます。

デバッガによって作成されたプロジェクト(.ibp)ファイルは、デバッガや**全ビルド**オプションによって作成された既存の.ibp ファイルを上書きします。もし特定のフローをデバッグしたい場合は、**フローデバッグ**オプションの使用を推奨します。このオプションは、ナビゲーションペインの中のフローのコンテキストメニューからアクセスすることができます。また、フローを非アクティブとし、デバッグしないように設定することもできます。

注:

- (フローで右クリックして、デバッグを選択することで)一つのフローをデバッグするとき、まるでフローに自動起動が設定されているように、フローの最初の実行内容が処理されます。このフローにトリガーが含まれている場合、フローは二回実行されます。最初の実行はトリガーが含まれない自動起動によって起動されます。そして、2 番目の実行はトリガー自体によって行われます。
- デバッグセッション後にサーバモードでプロジェクトを実行するには、**全ビルド**オプションを使用して、プロジェクトを再構築する必要があります。デバッガプロセスの一部として作成されたプロジェクトは、サーバモード内では実行されません。サーバは、プロジェクトを終了させ、メッセージを **ifs.log** ファイルに書き込みます。

ロギング

エラー ロギング

実行中に発生したエラーに関する情報は、いくつかのログファイルに記録されます。Magic xpi フォルダ内には、以下のファイルを含む **logs** フォルダがあります。:

- studio_error.log
- monitor_error.log
- <プロジェクト名>_error.log
- ifs.log

logs フォルダ内には、プロジェクト内で Java に接続する際に発生したエラーを含む Java フォルダもあります。

Mapper.log と呼ばれるログファイルも作成されます。マッピングの送り元または送り先のどちらかがデータベースの場合に、**Mapper.log** ファイルは有益です。この場合、**Mapper.log** ファイルには、データベースに送られた SQL ステートメントとステートメントの結果として受け取られたエラーの全てが含まれます。

Magic xpi スタジオは、ユーザ定義のメッセージをモニタウィンドウに送ることができる機能や、コンポーネントの構成プロパティ内のロギングオプション等のようなロギング機能を持っています。

コンポーネント ロギング

Magic xpi V3 以降では、すべてのコンポーネントの構成でロギングオプションの定義を行うことができます。ロギングのスコープとして、ロギングが何時実行されたか、ステップの実行に関連しているのかを以下のように指定します: ロギングしない , 実行前 , 実行後 , 両方。記録されたメッセージの一部として、テキストメッセージやファイルを添付することができます

メソッドを呼び出すことでロギングを設定することもできます。この場合、ロギングメッセージは各メソッドの実行後にモニタに送られます。
Magic xpi モニタから、ロギングを無効にしたり、有効にしたり、コンポーネントのロギングを有効にしたり、無効にしたりすることが可能です。

セーブメッセージ

このコンポーネントを使用すると、モニタ画面にプロジェクトベースのメッセージを表示することができ、プロジェクトの管理が簡単になります。

BAM メッセージ

このコンポーネントを使用すると、ビジネスプロセスを追跡することができます。例えば、プロジェクトが並行実行しているプロセスが多少あり、各プロセスのステータスを追跡する必要がある場合、BAM メッセージコンポーネントを重要な場所に追加し、BAM キーに（受注番号や分岐番号のような）ユニーク ID を与えることができます。

モニタ画面は、ユニーク ID に基づいて、メッセージをグループ化し、各プロセスの詳細を表示させることができます。

第 5 章: Magic xpi プロジェクトの実装

実行時の管理

実行管理は、2 つの主要な層で構築することができます。:

- **テクニカルモニタリング** – 管理者は、エラーメッセージを含めた実行環境を表示参照することができます。
  モニタ画面からプロジェクトのフローを追跡することを保証する各関連ポイントで**ユーザ定義メッセージ**を追加することを考慮する必要があります。**ファイルの添付機能**を使用することで、どのようなステップ(障害ポイント)に関係するデータファイルを含めることができます。開発者が障害ポイントと破損データファイルの両方に関するすべての関連情報を持つことができます。
- **ビジネスモニタリング** – どのようなビジネスステップでも **BAM メッセージ**を定義するで、プロセスフェーズを完全に追跡することができます。複数の位置やアプリケーションに関連するプロセスを管理している場合、**カテゴリー**オプションを使用することで、コンテキストメニューに表示されるメッセージによって確認することができます。

参考:	モニタデータベーステーブルで使用できるグラフィカルな報告ツールを使用することで、より多彩な監視機能を利用することができます。

第 6 章: Magic xpi プロジェクトの保守

アクティビティ ログ テーブル

Magic モニタには、実行プロジェクトのオンラインのビューがあります。Magic モニタは、全てのレコードを格納する **IFS_ACTLOG** と呼ばれるテーブルに基づいています。

統合プロジェクトは、1 日中多くのプロセスを管理するため、モニタテーブルが非常に大きくなります。一方で、エラーポイントや性能問題などを分析しようとする時に、モニタ情報が極めて役立ちます。従って、モニタテーブルの管理手続を検討する必要があります。

管理手続は、以下の項目にもとづきます。:

- モニタテーブルの手動消去
- 事前に定義された時間帯にモニタテーブルを消去する自動タスクを実行させます。例：先週のメッセージを消去する。スケジューラによって動作し、関連する WHERE 句を使用してこのテーブルから削除する専門のフローを使用することで、簡単にこのタスクを実装することができます。

ifs.ini ファイル内の以下のフラグを使用することで、データベースに集められるデータ量を減らすことができます。:

[MAGIC_IBOLT]DisableActivityLog = N

このフラグの値が Y に設定されると、アクティビティログが無効になります。どのアクティビティログのデータもデータベースに書き込まれず、どのメッセージもモニタに表示されません。

[MAGIC_IBOLT]MonitorLogLevel =* All

このフラグの値は、以下のどれかになります。カンマ区切りで複数設定することができます。:

- User – ユーザ定義のメッセージだけを記録し表示します。
- Error – エラーを表示し、エラーコンポーネントを呼び出します。
- Exec – サーバの起動やフローの実行開始などの実行関連のメッセージを表示します。このスイッチには、エラーとユーザメッセージが含まれています。
- Service – ODS やロック、および PSS などのサービス関連のメッセージを表示します。
- All – 全てのメッセージを表示します。(フラグが存在していない場合、デフォルトになります。)
- None –メッセージを表示しません。

注: このフラグに複数の値が定義する場合、(上記のように)= の後にアスタリスク(*)を入力してください。

[MAGIC_IBOLT]MonitorMode = Activity

このフラグは、以下の 3 つのオプションから 1 つを設定してください。:

- Activity – モニタはアクティビティログを表示しますが、統計表示を行いません。
- Statistic – モニタは統計を表示しますが、アクティビティログは表示されません。
- Both – モニタは、統計とアクティビティログの両方を表示します。

[MAGIC_IBOLT]ActivityLogSynchronizationMode = Async

このフラグは、アクティビティログの処理方法を指定します。::

- Asynch (デフォルト): 1 つのスレッドで全てのアクティビティを記録します。このオプションを選択した場合、Magic xpi サーバはシングルスレッドでアクティビティの書込みバッチ処理を行います。この設定では、システムのスピードと性能を向上させます。
- Synch: 各スレッドが個別にアクティビティを記録します。このオプションを選択した場合、Magic xpi サーバは、処理を実行した同じスレッドでリアルタイムにアクティビティを記録します。また、多くのアクティビティとプロジェクトの負荷を分散させることができます。

注: さらに、 **ifs_mytp** と呼ばれる DB テーブルはすべてのアクティビティ ログ メッセージを格納します。このテーブルには、**Suspend** と呼ばれるカラムがあります。特定のメッセージタイプに対してこのカラムの値が **1** の場合、このタイプのメッセージはアクティビティログに保存されません。

付録 A – 開発手法に関する用語

次の主要な用語は、Magic xpi の開発手法の説明の中で使用されています。:

用語	説明
プロセス	Magic xpi インテグレーション プラットフォーム内で実装されるビジネス シナリオ。
フロー	実行可能なステップ内へのプロセスの実装。1 つのフローが複数のプロセスと関連していたり、1 つのビジネスプロセスが複数のフローを使用していると解釈したりするように、1 つのフローが 1 つのプロセスと関係する場所では、フローとプロセスの関係はフレキシブルです。
コンポーネント	Magic xpi は独自のまたはユーザ定義の機能の集まりを含んでいます。これは、ウィザードベースの GUI で業界標準にもとづいた統合プロセスを作成するための必要な機能が含まれています。
ユーティリティ	プロジェクト設計者にプロジェクトのコントロールを強化し、BAM メッセージ、Data Mapper ユーティリティ、Flow Manager、Locking ユーティリティ、Post Event ユーティリティなど、プロジェクトのモデリング中に特定の動作を指定するためのビルディングブロック。
リソース	サーバや DBMS、アプリケーション、Web サーバ、等 IT 環境におけるすべてリソースの定義。
サービス	外部に公開する機能を定義します。フローが外部システムからどのように呼び出されるかを設定することができます。サービスには Web サービスや EJB、HTTP 等が含まれます。
ステップ	ステップは、フロー内での Magic xpi フロー サービスやコンポーネントのインスタンスです。すべてのフローは、フローロジックを実装する 1 つ以上のステップで構成されています。
コンテキスト	フローとそれによって呼び出される全てのリニアなフローの共用メモリ環境。コンテキストは、フローが開始される際に、初期設定が行われ、初期フローが終了するまでフローとすべてのリニアの子孫で利用可能な共有コンテキスト変数が含まれます。