HITACHI
Inspire the Next

Cosminexus V11 アプリケーションサーバ & BPM/ESB 基盤 概説

概説書

3021-3-J01-50

前書き

■ 対象製品

●適用 OS：Windows Server 2022

P-2943-7KY4 uCosminexus Application Server 11-40

P-2943-7SY4 uCosminexus Service Platform 11-40

●適用 OS：Windows Server 2022，Windows 10 x64，Windows 11

P-2943-7FY4 uCosminexus Developer 11-40

P-2943-7TY4 uCosminexus Service Architect 11-40

P-2943-7HY4 uCosminexus Client 11-40

●適用 OS：Red Hat Enterprise Linux 8（AMD/Intel 64），Red Hat Enterprise Linux 9（AMD/Intel 64）

P-9W43-7KY1 uCosminexus Application Server 11-40※

P-9W43-7SY1 uCosminexus Service Platform 11-40※

注　AIX をご使用の場合はこのマニュアルをご利用になれません。ご使用の製品の「リリースノート」で該当するマニュアルを確認し，参照してください。

なお，このマニュアル中に AIX に関する記載がありますが，無視してください。

注※　この製品については，サポート時期をご確認ください。

これらのプログラムプロダクトのほかにもこのマニュアルをご利用になれる場合があります。詳細は「リリースノート」でご確認ください。

これらのプログラムプロダクトでは，日立トレース共通ライブラリをインストールします。

■ 輸出時の注意

本製品を輸出される場合には、外国為替及び外国貿易法の規制並びに米国輸出管理規則など外国の輸出関連法規をご確認の上、必要な手続きをお取りください。

なお、不明な場合は、弊社担当営業にお問い合わせください。

■ 商標類

HITACHI，Cosminexus，DABroker，DCCM，HA モニタ，HiRDB，JP1，Object Wrapper，OpenTP1，ServerConductor，TPBroker，uCosminexus，VOS3/XS，XDM は，株式会社 日立製作所の商標または登録商標です。

AIX は，世界の多くの国で登録された International Business Machines Corporation の商標です。

This product includes software developed by the University of California, Berkeley and its contributors.

This software contains code derived from the RSA Data Security Inc. MD5 Message-Digest Algorithm, including various modifications by Spyglass Inc., Carnegie Mellon University, and Bell Communications Research, Inc (Bellcore).

Regular expression support is provided by the PCRE library package, which is open source software, written by Philip Hazel, and copyright by the University of Cambridge, England. The original software is available from ftp://ftp.csx.cam.ac.uk/pub/software/programming/pcre/

This product includes software developed by Ralf S. Engelschall <rse@engelschall.com> for use in the mod_ssl project (http://www.modssl.org/).

Java is a registered trademark of Oracle and/or its affiliates.

1. This product includes software developed by the OpenSSL Project for use in the OpenSSL Toolkit. (http://www.openssl.org/)

2. This product includes cryptographic software written by Eric Young (eay@cryptsoft.com)

3. This product includes software written by Tim Hudson (tjh@cryptsoft.com)

4. 本製品には OpenSSL Toolkit ソフトウェアを OpenSSL License および Original SSLeay License に従い使用しています。OpenSSL License および Original SSLeay License は以下のとおりです。

LICENSE ISSUES

==========

The OpenSSL toolkit stays under a double license, i.e. both the conditions of

the OpenSSL License and the original SSLeay license apply to the toolkit.

See below for the actual license texts.

OpenSSL License

/*

==========

* Copyright (c) 1998-2019 The OpenSSL Project. All rights reserved.

*

* Redistribution and use in source and binary forms, with or without
* modification, are permitted provided that the following conditions
* are met:
*
* 1. Redistributions of source code must retain the above copyright
* notice, this list of conditions and the following disclaimer.
*
* 2. Redistributions in binary form must reproduce the above copyright
* notice, this list of conditions and the following disclaimer in
* the documentation and/or other materials provided with the
* distribution.
*
* 3. All advertising materials mentioning features or use of this
* software must display the following acknowledgment:
* "This product includes software developed by the OpenSSL Project
* for use in the OpenSSL Toolkit. (http://www.openssl.org/)"
*
* 4. The names "OpenSSL Toolkit" and "OpenSSL Project" must not be used to
* endorse or promote products derived from this software without
* prior written permission. For written permission, please contact
* openssl-core@openssl.org.
*
* 5. Products derived from this software may not be called "OpenSSL"
* nor may "OpenSSL" appear in their names without prior written
* permission of the OpenSSL Project.
*
* 6. Redistributions of any form whatsoever must retain the following
* acknowledgment:
* "This product includes software developed by the OpenSSL Project
* for use in the OpenSSL Toolkit (http://www.openssl.org/)"
*
* THIS SOFTWARE IS PROVIDED BY THE OpenSSL PROJECT ``AS IS'' AND ANY
* EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
* IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR
* PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE OpenSSL PROJECT OR

* ITS CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
* SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT
* NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES;
* LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
* HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT,
* STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE)
* ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED
* OF THE POSSIBILITY OF SUCH DAMAGE.
*
==========
*
* This product includes cryptographic software written by Eric Young
* (eay@cryptsoft.com). This product includes software written by Tim
* Hudson (tjh@cryptsoft.com).
*
*/
Original SSLeay License

/* Copyright (C) 1995-1998 Eric Young (eay@cryptsoft.com)
* All rights reserved.
*
* This package is an SSL implementation written
* by Eric Young (eay@cryptsoft.com).
* The implementation was written so as to conform with Netscapes SSL.
*
* This library is free for commercial and non-commercial use as long as
* the following conditions are aheared to. The following conditions
* apply to all code found in this distribution, be it the RC4, RSA,
* lhash, DES, etc., code; not just the SSL code. The SSL documentation
* included with this distribution is covered by the same copyright terms
* except that the holder is Tim Hudson (tjh@cryptsoft.com).
*
* Copyright remains Eric Young's, and as such any Copyright notices in
* the code are not to be removed.

* If this package is used in a product, Eric Young should be given attribution
* as the author of the parts of the library used.
* This can be in the form of a textual message at program startup or
* in documentation (online or textual) provided with the package.
*
* Redistribution and use in source and binary forms, with or without
* modification, are permitted provided that the following conditions
* are met:
* 1. Redistributions of source code must retain the copyright
* notice, this list of conditions and the following disclaimer.
* 2. Redistributions in binary form must reproduce the above copyright
* notice, this list of conditions and the following disclaimer in the
* documentation and/or other materials provided with the distribution.
* 3. All advertising materials mentioning features or use of this software
* must display the following acknowledgement:
* "This product includes cryptographic software written by
* Eric Young (eay@cryptsoft.com)"
* The word 'cryptographic' can be left out if the rouines from the library
* being used are not cryptographic related :-).
* 4. If you include any Windows specific code (or a derivative thereof) from
* the apps directory (application code) you must include an acknowledgement:
* "This product includes software written by Tim Hudson (tjh@cryptsoft.com)"
*
* THIS SOFTWARE IS PROVIDED BY ERIC YOUNG ``AS IS'' AND
* ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
* IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
* ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS BE LIABLE
* FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
* DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS
* OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
* HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT

* LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY
* OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
* SUCH DAMAGE.
*
* The licence and distribution terms for any publically available version or
* derivative of this code cannot be changed. i.e. this code cannot simply be
* copied and put under another distribution licence
* [including the GNU Public Licence.]
*/

本ソフトウェアに含まれる第三者ソフトウェアに関する情報を次に示します。

第三者ソフトウェア：Trang

著作権者：Thai Open Source Software Center Ltd

当該ソフトウェアのライセンス条文：以下の通り。

Copyright (c) 2002, 2003, 2008 Thai Open Source Software Center Ltd

All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

Neither the name of the Thai Open Source Software Center Ltd nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE REGENTS OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

本書には，X/Open の許諾に基づき X/Open CAE Specification System Interfaces and Headers, Issue4,（C202 ISBN 1-872630-47-2）Copyright（C）July 1992， X/Open Company Limited の内容が含まれています;

なお，その一部は IEEE Std 1003.1-1990,（C）1990 Institute of Electrical and Electronics Engineers，Inc.及び IEEE std 1003.2/D12,（C）1992 Institute of Electrical and Electronics Engineers，Inc.を基にしています。

■ 謝辞

Reliable Messaging は，経済産業省が 2003 年度から 3 年間実施した「ビジネスグリッドコンピューティングプロジェクト」の技術開発の成果を含みます。

■ 発行

2024 年 2 月 3021-3-J01-50

■ 著作権

変更内容

変更内容(3021-3-J01-50) uCosminexus Application Server 11-40，uCosminexus Client 11-40，uCosminexus Developer 11-40，uCosminexus Service Architect 11-40，uCosminexus Service Platform 11-40

追加・変更内容	変更個所
JDK17 サポートに関する記述を追加した。	2.2.2(6)，4.6.1(1)，4.6.1(2)
スケジュール駆動受付の記述を追加した。	9.5.9
Application Server，Developer，Client，Service Platform，および Service Architect から次の適用 OS を削除した。 • Windows Server 2019 Standard • Windows Server 2019 Datacenter	-

単なる誤字・脱字などはお断りなく訂正しました。

はじめに

このマニュアルは，Cosminexus（コズミネクサス）のアプリケーションサーバおよびBPM/ESB基盤の概要について説明したものです。アプリケーションサーバおよびBPM/ESB基盤の製品構成と，これらの製品で実現できることについて説明しています。また，アプリケーションサーバおよびBPM/ESB基盤のマニュアル体系についても説明しています。

アプリケーションサーバおよびBPM/ESB基盤では，次に示すプログラムプロダクトを使用してシステムの構築・運用，またはアプリケーションの開発をします。

- uCosminexus Application Server
- uCosminexus Client
- uCosminexus Developer
- uCosminexus Service Architect
- uCosminexus Service Platform

ここでは，これらのプログラムプロダクトに対応するマニュアルで共通に使用している用語，表記について説明します。

■ 関連マニュアルの表記

関連マニュアル，およびこのマニュアルで使用している関連マニュアル名の表記を次の表に示します。

アプリケーションサーバおよびBPM/ESB基盤関連

表記	正式名称	資料番号
アプリケーションサーバ ファーストステップガイド	Cosminexus V11 アプリケーションサーバ ファーストステップガイド	3021-3-J00
アプリケーションサーバ & BPM/ESB 基盤 概説	Cosminexus V11 アプリケーションサーバ & BPM/ESB 基盤 概説	3021-3-J01
アプリケーションサーバ システム構築・運用ガイド	Cosminexus V11 アプリケーションサーバ システム構築・運用ガイド	3021-3-J02
アプリケーションサーバ 仮想化システム構築・運用ガイド	Cosminexus V11 アプリケーションサーバ 仮想化システム構築・運用ガイド	3021-3-J03
アプリケーションサーバ システム設計ガイド	Cosminexus V11 アプリケーションサーバ システム設計ガイド	3021-3-J04
アプリケーションサーバ 機能解説 基本・開発編(Web コンテナ)	Cosminexus V11 アプリケーションサーバ 機能解説 基本・開発編(Web コンテナ)	3021-3-J05

表記	正式名称	資料番号
アプリケーションサーバ 機能解説 基本・開発編(EJB コンテナ)	Cosminexus V11 アプリケーションサーバ 機能解説 基本・開発編(EJB コンテナ)	3021-3-J06
アプリケーションサーバ 機能解説 基本・開発編(コンテナ共通機能)	Cosminexus V11 アプリケーションサーバ 機能解説 基本・開発編(コンテナ共通機能)	3021-3-J07
アプリケーションサーバ 機能解説 拡張編	Cosminexus V11 アプリケーションサーバ 機能解説 拡張編	3021-3-J08
アプリケーションサーバ 機能解説 セキュリティ管理機能編	Cosminexus V11 アプリケーションサーバ 機能解説 セキュリティ管理機能編	3021-3-J09
アプリケーションサーバ 機能解説 運用／監視／連携編	Cosminexus V11 アプリケーションサーバ 機能解説 運用／監視／連携編	3021-3-J10
アプリケーションサーバ 機能解説 保守／移行編	Cosminexus V11 アプリケーションサーバ 機能解説 保守／移行編	3021-3-J11
アプリケーションサーバ 機能解説 互換編	Cosminexus V11 アプリケーションサーバ 機能解説 互換編	3021-3-J12
アプリケーションサーバ アプリケーション設定操作ガイド	Cosminexus V11 アプリケーションサーバ アプリケーション設定操作ガイド	3021-3-J13
アプリケーションサーバ 運用管理ポータル操作ガイド	Cosminexus V11 アプリケーションサーバ 運用管理ポータル操作ガイド	3021-3-J14
アプリケーションサーバ リファレンス コマンド編	Cosminexus V11 アプリケーションサーバ リファレンス コマンド編	3021-3-J15
アプリケーションサーバ リファレンス 定義編(サーバ定義)	Cosminexus V11 アプリケーションサーバ リファレンス 定義編(サーバ定義)	3021-3-J16
アプリケーションサーバ リファレンス 定義編(アプリケーション／リソース定義)	Cosminexus V11 アプリケーションサーバ リファレンス 定義編(アプリケーション／リソース定義)	3021-3-J17
HTTP Server	Cosminexus V11 アプリケーションサーバ Cosminexus HTTP Server	3021-3-J18
Reliable Messaging	Cosminexus V11 アプリケーションサーバ Cosminexus Reliable Messaging	3021-3-J19
アプリケーションサーバ アプリケーション開発ガイド	Cosminexus V11 アプリケーションサーバ アプリケーション開発ガイド	3021-3-J20
アプリケーションサーバ リファレンス API 編	Cosminexus V11 アプリケーションサーバ リファレンス API 編	3021-3-J21
XML Processor ユーザーズガイド	Cosminexus V11 アプリケーションサーバ Cosminexus XML Processor ユーザーズガイド	3021-3-J22
アプリケーションサーバ Web サービス開発ガイド	Cosminexus V11 アプリケーションサーバ Web サービス開発ガイド	3021-3-J23

表記	正式名称	資料番号
アプリケーションサーバ Web サービスセキュリティ構築ガイド	Cosminexus V11 アプリケーションサーバ Web サービスセキュリティ構築ガイド	3021-3-J24
アプリケーションサーバ SOAP アプリケーション開発の手引	Cosminexus V11 アプリケーションサーバ SOAP アプリケーション開発の手引	3021-3-J25
XML Security - Core ユーザーズガイド	Cosminexus V11 アプリケーションサーバ Cosminexus XML Security - Core ユーザーズガイド	3021-3-J26
アプリケーションサーバ メッセージ(構築／運用／開発用)	Cosminexus V11 アプリケーションサーバ メッセージ(構築／運用／開発用)	3021-3-J27
TPBroker ユーザーズガイド	TPBroker Version 5 トランザクショナル分散オブジェクト基盤 TPBroker　ユーザーズガイド	3021-3-J28
TPBroker 運用ガイド	TPBroker Version 5 トランザクショナル分散オブジェクト基盤 TPBroker　運用ガイド	3021-3-J29
Borland(R) Enterprise Server VisiBroker(R) デベロッパーズガイド	VisiBroker Version 5 Borland(R) Enterprise Server VisiBroker(R)　デベロッパーズガイド	3021-3-J30
Borland(R) Enterprise Server VisiBroker(R) プログラマーズリファレンス	VisiBroker Version 5 Borland(R) Enterprise Server VisiBroker(R)　プログラマーズリファレンス	3021-3-J31
アプリケーションサーバ メッセージ(監査者用)	Cosminexus V11 アプリケーションサーバ メッセージ(監査者用)	3021-3-J32
サービスプラットフォーム ファーストステップガイド	Cosminexus V11 BPM/ESB 基盤 サービスプラットフォーム ファーストステップガイド	3021-3-J41
サービスプラットフォーム 解説	Cosminexus V11 BPM/ESB 基盤 サービスプラットフォーム 解説	3021-3-J42
サービスプラットフォーム 開発ガイド 基本開発編	Cosminexus V11 BPM/ESB 基盤 サービスプラットフォーム 開発ガイド 基本開発編	3021-3-J43
サービスプラットフォーム 開発ガイド 受付・アダプタ定義編	Cosminexus V11 BPM/ESB 基盤 サービスプラットフォーム 開発ガイド 受付・アダプタ定義編	3021-3-J44
サービスプラットフォーム システム構築・運用ガイド	Cosminexus V11 BPM/ESB 基盤 サービスプラットフォーム システム構築・運用ガイド	3021-3-J45
サービスプラットフォーム リファレンス	Cosminexus V11 BPM/ESB 基盤 サービスプラットフォーム リファレンス	3021-3-J46
サービスプラットフォーム メッセージ	Cosminexus V11 BPM/ESB 基盤 サービスプラットフォーム メッセージ	3021-3-J47
アプリケーションサーバ & BPM/ESB 基盤 用語解説	Cosminexus V11 アプリケーションサーバ & BPM/ESB 基盤 用語解説	3021-3-J99

製品とマニュアルの対応については，「3.4　製品とマニュアルの対応」を参照してください。

HiRDB 関連

表記	正式名称	資料番号
HiRDB 解説	HiRDB Version 10 解説	3020-6-551
	HiRDB Version 9 解説	3020-6-450
HiRDB システム導入・設計ガイド	HiRDB Version 10 システム導入・設計ガイド（UNIX(R)用）	3020-6-552
	HiRDB Version 10 システム導入・設計ガイド（Windows(R)用）	3020-6-553
	HiRDB Version 9 システム導入・設計ガイド（UNIX(R)用）	3000-6-452
	HiRDB Version 9 システム導入・設計ガイド（Windows(R)用）	3020-6-452
HiRDB システム運用ガイド	HiRDB Version 10 システム運用ガイド（UNIX(R)用）	3020-6-556
	HiRDB Version 10 システム運用ガイド（Windows(R)用）	3020-6-557
	HiRDB Version 9 システム運用ガイド（UNIX(R)用）	3000-6-454
	HiRDB Version 9 システム運用ガイド（Windows(R)用）	3020-6-454
HiRDB システム定義	HiRDB Version 10 システム定義（UNIX(R)用）	3020-6-554
	HiRDB Version 10 システム定義（Windows(R)用）	3020-6-555
	HiRDB Version 9 システム定義（UNIX(R)用）	3000-6-453
	HiRDB Version 9 システム定義（Windows(R)用）	3020-6-453
HiRDB SQL リファレンス	HiRDB Version 10 SQL リファレンス	3020-6-561
	HiRDB Version 9 SQL リファレンス	3020-6-457
HiRDB UAP 開発ガイド	HiRDB Version 10 UAP 開発ガイド	3020-6-560
	HiRDB Version 9 UAP 開発ガイド	3020-6-456
HiRDB XDM/RD E2 接続機能	HiRDB Version 10 XDM/RD E2 接続機能	3020-6-565
	HiRDB Version 9 XDM/RD E2 接続機能	3020-6-465
HiRDB コマンドリファレンス	HiRDB Version 10 コマンドリファレンス（UNIX(R)用）	3020-6-558
	HiRDB Version 10 コマンドリファレンス（Windows(R)用）	3020-6-559
	HiRDB Version 9 コマンドリファレンス（UNIX(R)用）	3000-6-455
	HiRDB Version 9 コマンドリファレンス（Windows(R)用）	3020-6-455
HiRDB メッセージ	HiRDB Version 10 メッセージ	3020-6-562
	HiRDB Version 9 メッセージ	3020-6-458
HiRDB SQL Executer オンラインヘルプ	HiRDB SQL Executer オンラインヘルプ（GUI 版）	3020-6-35B

表記	正式名称	資料番号
	HiRDB SQL Executer オンラインヘルプ（GUI 版）	3020-6-45C
	HiRDB SQL Executer オンラインヘルプ（ラインモード版）	3020-6-35C
	HiRDB SQL Executer オンラインヘルプ（ラインモード版）	3020-6-45B

JP1 関連

表記	正式名称	資料番号
JP1/Advanced Shell	JP1 Version 11 JP1/Advanced Shell	3021-3-B32
JP1/Audit Management - Manager 構築・運用ガイド	JP1 Version 11 JP1/Audit Management - Manager 構築・運用ガイド	3021-3-A17
	JP1 Version 10 JP1/Audit Management - Manager 構築・運用ガイド	3021-3-165
JP1/Automatic Job Management System 操作ガイド	JP1 Version 12 JP1/Automatic Job Management System 3 操作ガイド	3021-3-D27
JP1/Automatic Job Management System 構築ガイド 1	JP1 Version 12 JP1/Automatic Job Management System 3 構築ガイド	3021-3-D24
JP1/Automatic Job Management System 設計ガイド	JP1 Version 12 JP1/Automatic Job Management System 3 設計ガイド（システム構築編）	3021-3-D22
JP1/Automatic Job Management System 導入ガイド	JP1 Version 12 JP1/Automatic Job Management System 3 導入ガイド	3021-3-D21
JP1/Base 運用ガイド	JP1 Version 12 JP1/Base 運用ガイド	3021-3-D65
JP1/Cm2/Extensible SNMP Agent	JP1 Version 12 JP1/Extensible SNMP Agent	3021-3-E05
	JP1 Version 11 JP1/Extensible SNMP Agent	3021-3-A78
	JP1 Version 10 JP1/Cm2/Extensible SNMP Agent	3021-3-251
JP1/File Transmission Server	JP1 Version 12 JP1/File Transmission Server/FTP（Windows(R)用）	3021-3-D38
	JP1 Version 12 JP1/File Transmission Server/FTP（UNIX(R)用）	3021-3-D39
	JP1 Version 11 JP1/File Transmission Server/FTP（Windows(R)用）	3021-3-B36
	JP1 Version 11 JP1/File Transmission Server/FTP（UNIX(R)用）	3021-3-B37

表記	正式名称	資料番号
	JP1 Version 11 JP1/File Transmission Server -全銀 TCP	3021-3-B38
	JP1 Version 10 JP1/File Transmission Server/FTP（Windows(R)用）	3021-3-137
	JP1 Version 10 JP1/File Transmission Server/FTP（UNIX(R)用）	3021-3-138
	JP1 Version 10 JP1/File Transmission Server - 全銀 TCP	3021-3-139
JP1/Integrated Management - Manager 運用ガイド	JP1 Version 12 JP1/Integrated Management 2 - Manager 運用ガイド	3021-3-D53
JP1/Integrated Management - Manager 画面リファレンス	JP1 Version 12 JP1/Integrated Management 2 - Manager 画面リファレンス	3021-3-D54
JP1/Integrated Management - Manager 構築ガイド	JP1 Version 12 JP1/Integrated Management 2 - Manager 構築ガイド	3021-3-D52
JP1/Integrated Management - Manager コマンド・定義ファイルリファレンス	JP1 Version 12 JP1/Integrated Management 2 - Manager コマンド・定義ファイルリファレンス	3021-3-D55
JP1/IT Resource Management - Manager 設計・構築ガイド	JP1 Version 10 JP1/IT Resource Management - Manager 設計・構築ガイド	3021-3-231
JP1/Performance Management リファレンス	JP1 Version 12 JP1/Performance Management リファレンス	3021-3-D78
	JP1 Version 11 JP1/Performance Management リファレンス	3021-3-A39
	JP1 Version 10 JP1/Performance Management リファレンス	3021-3-043
JP1/Performance Management 運用ガイド	JP1 Version 12 JP1/Performance Management 運用ガイド	3021-3-D77
	JP1 Version 11 JP1/Performance Management 運用ガイド	3021-3-A38
	JP1 Version 10 JP1/Performance Management 運用ガイド	3021-3-042
JP1/Performance Management 設計・構築ガイド	JP1 Version 12 JP1/Performance Management 設計・構築ガイド	3021-3-D76
	JP1 Version 11 JP1/Performance Management 設計・構築ガイド	3021-3-A37
	JP1 Version 10 JP1/Performance Management 設計・構築ガイド	3021-3-041
JP1/Performance Management - Agent Option for uCosminexus Application Server	JP1 Version 11 JP1/Performance Management - Agent Option for uCosminexus Application Server	3021-3-A64
	JP1 Version 10 JP1/Performance Management - Agent Option for uCosminexus Application Server	3021-3-070

OpenTP1 関連

表記	正式名称	資料番号
OpenTP1 クライアント使用の手引 TP1/Client/J 編	OpenTP1 Version 6 分散トランザクション処理機能 OpenTP1 クライアント使用の手引 TP1/Client/J 編	3000-3-950
	OpenTP1 Version 7 分散トランザクション処理機能 OpenTP1 クライアント使用の手引 TP1/Client/J 編	3000-3-D59
OpenTP1 プログラム作成リファレンス C 言語編	OpenTP1 Version 7 分散トランザクション処理機能 OpenTP1 プログラム作成リファレンス C 言語編	3000-3-D54
OpenTP1 システム定義	OpenTP1 Version 7 分散トランザクション処理機能 OpenTP1 システム定義	3000-3-D52
OpenTP1 プログラム作成の手引	OpenTP1 Version 7 分散トランザクション処理機能 OpenTP1 プログラム作成の手引	3000-3-D51
OpenTP1 解説	OpenTP1 Version 7 分散トランザクション処理機能 OpenTP1 解説	3000-3-D50
OpenTP1 Version 7 メッセージキューイングアクセス機能 TP1/Message Queue - Access 使用の手引	OpenTP1 Version 7 メッセージキューイングアクセス機能 TP1/Message Queue - Access 使用の手引	3000-3-D94
	OpenTP1 Version 7 メッセージキューイングアクセス機能 TP1/Message Queue Access 使用の手引	
TP1/Server Base Enterprise Option 使用の手引	分散トランザクション処理機能 TP1/Server Base Enterprise Option 使用の手引	3000-3-982
TP1/Server Base Enterprise Option　プログラム作成の手引	分散トランザクション処理機能 TP1/Server Base Enterprise Option　プログラム作成の手引	3000-3-983

その他

表記	正式名称	資料番号
高信頼化システム監視機能　HA モニタ	高信頼化システム監視機能　HA モニタ　Linux(R)編	3000-9-201
コード変換ユーザーズガイド（Java 版）	Hitachi Code Converter for Java	3020-7-360

■ フォルダとパスの表記

このマニュアルでは，Windows，AIX および Linux で共通の内容の場合，Windows の「フォルダ」を「ディレクトリ」と表記しています。また，「¥」を「/」と表記しています。

Windows の場合，「ディレクトリ」を「フォルダ」に，「/」を「¥」に置き換えてお読みください。

■ 製品名と機能名の表記

このマニュアルでは，製品名と機能名を次のように表記しています。

表記		製品名と機能名
ACOS	AX2000	AX2000
	AX2500	AX2500
	BS320	BS320 ロードバランサブレード
Active Directory		Microsoft Active Directory
AMD-V		AMD Virtualization
Application Development Plug-in		Cosminexus Application Development Plug-in
Application Server		uCosminexus Application Server
		uCosminexus Application Server(64)
AWS		Amazon Web Services
BIG-IP	BIG-IP v9	BIG-IP ソフトウェアバージョン 9.1.0 以降
	BIG-IP v10.1	BIG-IP ソフトウェアバージョン 10.1.0 以降
	BIG-IP v10.2	BIG-IP ソフトウェアバージョン 10.2.0 以降
	BIG-IP v11	BIG-IP ソフトウェアバージョン 11.0.0 以降
BJEX または Batch Job Execution Server		Batch Job Execution System - Base uCosminexus Batch Job Execution Server
BPM/ESB 基盤		Cosminexus ビジネスプロセス管理/エンタープライズサービスバス
CJMSP ブローカー		Cosminexus JMS プロバイダのブローカー機能
CJMSP リソースアダプタ		Cosminexus JMS プロバイダのリソースアダプタ
CJMS プロバイダ		Cosminexus JMS プロバイダ
CJPA プロバイダ		Cosminexus JPA プロバイダ
Client		uCosminexus Client
Common Library		Cosminexus Common Library
Component Container		Cosminexus Component Container
Component Container - Client		Cosminexus Component Container - Client
Component Container - Redirector		Cosminexus Component Container - Redirector
Component Library		Cosminexus Component Library

表記	製品名と機能名
CTM または Component Transaction Monitor	Cosminexus Component Transaction Monitor
DABroker Library	Cosminexus DABroker Library
DB Connector	Cosminexus DB Connector
DB Connector for Reliable Messaging	DB Connector for Cosminexus Reliable Messaging
DCCM3	VOS1 DCCM3
	VOS3 XDM/DCCM3
Developer	uCosminexus Developer
Developer's Kit for Java	Cosminexus Developer's Kit for Java
Driver	Cosminexus Driver
Eclipse	Eclipse IDE for Java EE Developers
Excel	Microsoft Excel
	Microsoft Office Excel
HCSC	Hitachi Cosminexus Service Coordinator
HCSC-Business Process，ビジネスプロセス基盤，または BP 基盤	Hitachi Cosminexus Service Coordinator - Business Process
HCSC-Data Transform またはデータ変換基盤	Hitachi Cosminexus Service Coordinator - Data Transform
HCSC-Manager または HCSC-MNG	Hitachi Cosminexus Service Coordinator - Manager
HCSC-Messaging，HCSC-MSG，またはメッセージング基盤	Hitachi Cosminexus Service Coordinator - Messaging
HCSCTE	Hitachi Cosminexus Service Coordinator Tools for Eclipse
HiRDB または HiRDB サーバ	HiRDB Server
	HiRDB Server with Additional Function
	HiRDB/Single Server または組み込みデータベース
HiRDB クライアント	HiRDB/Run Time
	HiRDB/Developer's Kit
	HiRDB Developer's Suite
HiRDB Type4 JDBC Driver	HiRDB Type4 JDBC ドライバ
HTTP Server	Cosminexus HTTP Server
Hyper-V	Microsoft Hyper-V

表記			製品名と機能名
Intel VT			Intel Virtualization Technology
Internet Explorer			Windows Internet Explorer
Internet Explorer 11			Windows Internet Explorer 11
JAX-WS 機能			Cosminexus JAX-WS
JP1/AJS	JP1/AJS - Agent		JP1/Automatic Job Management System 3 - Agent
	JP1/AJS - Manager		JP1/Automatic Job Management System 3 - Manager
	JP1/AJS - View		JP1/Automatic Job Management System 3 - View
JP1/Audit Management - Manager			JP1/Audit Management - Manager
			JP1/NETM/Audit - Manager
JP1/Cm2	JP1/Cm2/ESA		JP1/Cm2/Extensible SNMP Agent
JP1/ESP			JP1/Extensible Service Probe
JP1/File Transmission Server/FTP			JP1 Version 9 JP1/File Transmission Server/FTP
JP1/IM	JP1/IM - Manager		JP1/Integrated Management - Manager
	JP1/IM - View		JP1/Integrated Management - View
JP1/ITRM	JP1/ITRM - Manager		JP1/IT Resource Management - Manager
JP1/PFM			JP1/Performance Management
JP1/PFM	JP1/PFM - Agent	JP1/PFM - Agent for Cosminexus	JP1/Performance Management - Agent Option for uCosminexus Application Server
		JP1/PFM - Agent for Virtual Machine	JP1/Performance Management - Agent Option for Virtual Machine
	JP1/PFM - Base		JP1/Performance Management - Base
	JP1/PFM - Manager		JP1/Performance Management - Manager
	JP1/PFM - Web Console		JP1/Performance Management - Web Console
JP1/SC/DPM			JP1/ServerConductor/Deployment Manager Standard Edition
Loadflowbal			HA8000-ie/Loadflowbal
Management Server			Cosminexus Management Server
Manager			Cosminexus Manager

表記		製品名と機能名
Microsoft Cluster Service		Microsoft Cluster Service
Microsoft IIS	Microsoft IIS 10.0	Microsoft Internet Information Services 10.0
Microsoft Sysprep		Microsoft Sysprep
MyEclipse		MyEclipse for Cosminexus
.NET Framework		Microsoft .NET Framework Version 2.0
.NET Framework SDK		Microsoft .NET Framework 2.0 Software Development Kit
Oracle，または ORACLE		ORACLE Oracle Database
PP インストーラ		日立 PP インストーラ
PRF または Performance Tracer		Cosminexus Performance Tracer
Process Modeler		Process Modeler 5 for Microsoft Visio Professional Edition
RM または Reliable Messaging		Cosminexus Reliable Messaging
SAP R/3		SAP R/3
Server Plug-in		Cosminexus Server Plug-in
Service Adapter Architect for Flat Files		uCosminexus Service Adapter Architect for Flat Files
Service Adapter Architect for FTP		uCosminexus Service Adapter Architect for FTP
Service Adapter Architect for Message Queue		uCosminexus Service Adapter Architect for Message Queue
Service Adapter Architect for Object Access		uCosminexus Service Adapter Architect for Object Access
Service Adapter Architect for TP1		uCosminexus Service Adapter Architect for TP1
Service Adapter for Flat Files		uCosminexus Service Adapter for Flat Files
Service Adapter for FTP		uCosminexus Service Adapter for FTP
Service Adapter for Message Queue		uCosminexus Service Adapter for Message Queue
Service Adapter for Object Access		uCosminexus Service Adapter for Object Access
Service Adapter for TP1		uCosminexus Service Adapter for TP1
Service Architect		uCosminexus Service Architect
Service Coordinator Interactive Workflow		uCosminexus Service Coordinator Interactive Workflow
Service Coordinator-Manager		Cosminexus Service Coordinator-Manager

表記		製品名と機能名
Service Coordinator または CSC		Cosminexus Service Coordinator
Service Development Plug-in		Cosminexus Service Development Plug-in
Service Platform		uCosminexus Service Platform
		uCosminexus Service Platform(64)
Service Platform - Base		uCosminexus Service Platform - Base
Smart Composer		Cosminexus Smart Composer
SOAP 通信基盤		Cosminexus SOAP 通信基盤
SQL Server	SQL Server 2017	Microsoft SQL Server 2017
	SQL Server 2019	Microsoft SQL Server 2019
SQL Server JDBC Driver	SQL Server の JDBC ドライバ	Microsoft JDBC Driver for SQL Server
Struts		Jakarta Struts 1.1
TMS-4V/SP		Transaction Management System-4V/System Product
TMS-4V/SP/Server		Transaction Management System-4V/System Product/Server
TP1 Connector		uCosminexus TP1 Connector
TP1/Base		uCosminexus TP1/Server Base
TP1/Client	TP1/Client/J	uCosminexus TP1/Client/J
	TP1/Client/P	uCosminexus TP1/Client/P
	TP1/Client/W	uCosminexus TP1/Client/W
TP1/COBOL adapter		TP1/COBOL adapter for Cosminexus
TP1/EE		TP1/Server Base Enterprise Option
TP1/Message Queue - Access		uCosminexus TP1/Message Queue - Access
TP1/Web		uCosminexus TP1/Web
TPBroker		Cosminexus TPBroker
TPBroker for C++		Cosminexus TPBroker for C++
TPBroker for Java		Cosminexus TPBroker for Java
UNIX	AIX	AIX V7.1
		AIX V7.2

表記			製品名と機能名
	Linux	Linux 8	Red Hat Enterprise Linux Server 8.1（AMD/Intel 64）以降
		Linux 9	Red Hat Enterprise Linux Server 9.1（AMD/Intel 64）以降
Virtual Server Manager			Cosminexus Virtual Server Manager
VMware	VMware ESX		VMware ESX(R)
	VMware Tools		VMware Tools(TM)
	VMware vCenter Server		VMware vCenter Server(R)
	VMware vSphere Client		VMware vSphere(R) Client
Web Redirector			uCosminexus Web Redirector
Web Server			Hitachi Web Server
Web Services			Cosminexus Web Services
Web Services - Base			Cosminexus Web Services - Base
Web Services - Security			Cosminexus Web Services - Security
Web サービスセキュリティ機能			Cosminexus Web サービスセキュリティ機能
Windows	Windows Server 2022	Windows Server 2022 Standard	Windows Server 2022 Standard 日本語版
		Windows Server 2022 Datacenter	Windows Server 2022 Datacenter 日本語版
	Windows 10	Windows 10 x64	Windows 10 Enterprise 日本語版(64 ビット版)
			Windows 10 Pro 日本語版(64 ビット版)
	Windows 11		Windows 11 Enterprise 日本語版
			Windows 11 Pro 日本語版
Windows Server Failover Cluster			Windows Server Failover Cluster
XDM/RD E2			Extensible Data Manager/Relational Database Extended Version 2
XML Processor			Cosminexus XML Processor
XML Security			Cosminexus XML Security
XML Security - Core			Cosminexus XML Security - Core
アダプタコマンド			Cosminexus アダプタコマンド

表記	製品名と機能名
アプリケーションサーバの JPA 機能	Cosminexus JPA
インストーラ	日立総合インストーラ
共通モジュール群	Cosminexus 共通モジュール群
クラス別統計	日立クラス別統計
コード変換 - Development Kit	uCosminexus 日立コード変換 - Development Kit
コード変換 - Server Runtime	uCosminexus 日立コード変換 - Server Runtime
製品情報ファイル	Cosminexus 製品情報ファイル
製品の JavaVM または JavaVM	日立 JavaVM
トレース共通ライブラリ	日立トレース共通ライブラリ
バージョン情報	日立バージョン情報
バッチライブラリ	Cosminexus バッチライブラリ
標準ログインモジュール	Cosminexus 標準ログインモジュール
モニタ起動コマンド	Cosminexus モニタ起動コマンド

アプリケーションサーバおよび BPM/ESB 基盤のマニュアルに記載している Windows のメニュー名の表記は，使用する Windows のバージョンによって異なります。ご使用の OS に合わせて読み替えて操作してください。

本文中では，JP1 関連製品を総称して JP1 と表記することもあります。

なお，Application Server および Developer を総称して，アプリケーションサーバと表記します。

また，Linux に関しては，バージョンごとに次のように表記することがあります。

表記	OS 名
Red Hat Enterprise Linux 8	Red Hat Enterprise Linux Server 8.1（AMD/Intel 64）以降
Red Hat Enterprise Linux 9	Red Hat Enterprise Linux Server 9.1（AMD/Intel 64）以降

■ 英略語

このマニュアルで使用している英略語を次に示します。

英略語	英字での表記
ACL	Access Control List
ACOS	Advanced Core Operating System
AES	Advanced Encryption Standard
API	Application Programming Interface

英略語	英字での表記
ASCII	American Standard Code for Information Interchange
AWT	Abstract Window Toolkit
BLOB	Binary Large Object
BMP	Bean-Managed Persistence
BMT	Bean-Managed Transaction
BOM	Byte Order Mark
BP	Business Process
BPEL	Business Process Execution Language
BPM	Business Process Management
BPMN	Business Process Modeling Notation
C14N	Canonicalization
CA	Certificate Authority
CDI	Contexts and Dependency Injection
CMP	Container-Managed Persistence
CMR	Container-Managed Relationship
CMT	Container-Managed Transaction
Concurrency Utilities for Java EE 1.0	JSR 236 Concurrency Utilities for Java EE
Connector 1.0	J2EE Connector Architecture 1.0
Connector 1.5	J2EE Connector Architecture 1.5
CORBA	Common Object Request Broker Architecture
CPU	Central Processing Unit
CR	Carriage Return
CRL	Certificate Revocation List
CSR	Certificate Signing Request
CSS	Cascading Style Sheets
CSV	Comma Separated Value
CUI	Character User Interface
CVS	Concurrent Versions System
DB	Database

英略語	英字での表記
DBMS	Database Management System
DD	Deployment Descriptor
DDL	Data Definition Language
DES	Data Encryption Standard
DI	Dependency Injection
DII	Dynamic Invocation Interface
DIT	Directory Information Tree
DLL	Dynamic Link Library
DMZ	Demilitarized Zone
DN	Distinguished Name
DNS	Domain Name System
DOM	Document Object Model
DoS	Denial of Service
DSA	Digital Signature Algorithm
DTD	Document Type Definition
EAR	Enterprise Archive
EIS	Enterprise Information System
EJB または Enterprise JavaBeans	Enterprise JavaBeans
EJB QL	EJB Query Language
EL	Expression Language
EL3.0	JSR 341 Expression Language
EOD	Ease of Development
ERP	Enterprise Resource Planning
ESB	Enterprise Service Bus
ETL	Extract Transform Loading
EUC	Extended UNIX Code
FAQ	Frequently Asked Questions
FF	Form Feed
FIFO	First-In First-Out

英略語	英字での表記
FK	Foreign Key
FLOPS	Floating point number Operations Per Second
FQDN	Fully Qualified Domain Name
FTP	File Transfer Protocol
GC	Garbage Collection
GIF	Graphic Interchange Format
GMT	Greenwich Mean Time
GUI	Graphical User Interface
HA	High Availability
HMAC	Hash based MAC
HNTRLib	Hitachi Network Objectplaza Trace Library
HTML	Hyper Text Markup Language
HTTP	Hyper Text Transfer Protocol
HTTPS	Hyper Text Transfer Protocol Security
I/O	Input/Output
IANA	Internet Assigned Numbers Authority
ID	Identifier
IDE	Integrated Development Environment
IIOP	Internet Inter-Orb Protocol
IIS	Internet Information Services
IP	Internet Protocol
IPv6	Internet Protocol Version 6
ISAPI	Internet Server Application Programming Interface
iSCSI	Internet Small Computer System Interface
ISO	International Organization for Standardization
IT	Information Technology
IV	Initialization Vector
J2EE または Java 2 Platform, Enterprise Edition	J2EE
	Java 2 Platform, Enterprise Edition

英略語	英字での表記
J2SE	Java 2 Platform, Standard Edition
JAR	Java Archive
Java 2 Runtime Environment, Standard Edition	Java 2 Runtime Environment, Standard Edition
Java 2 SDK または Java 2 SDK, Standard Edition	Java 2 Software Development Kit, Standard Edition
JavaAPI	Java Application Programming Interface
Java Batch 1.0	JSR 352 Batch Applications for the Java Platform
JavaBeans	JavaBeans
Java EE, Java Platform, または Enterprise Edition	Java Platform, Enterprise Edition
Java HotSpot Client VM	Java HotSpot Client Virtual Machine
Java HotSpot Server VM	Java HotSpot Server Virtual Machine
JavaMail	JavaMail
Java SE	Java Platform, Standard Edition
JavaVM または JVM	Java Virtual Machine
JAX-RS 2.1	JSR 370 Java™ API for RESTful Web Services（JAX-RS）
JAX-WS	Java API for XML-Based Web Services
JAXB	Java Architecture for XML Binding
JAXB または The Java Architecture for XML Binding	The Java Architecture for XML Binding
JAXP または Java API for XML Processing	Java API for XML Processing
JAXR	Java API for XML Registries
JCA	J2EE Connector Architecture
JCE	Java Cryptography Extension
JDBC	Java Database Connectivity
	JDBC
JDK	Java Development Kit
	JDK
JIS	Japanese Industrial Standards
JMS	Java Message Service

英略語	英字での表記
JMX	Java Management Extensions
JNDI	Java Naming and Directory Interface
JNI	Java Native Interface
JPA	Java Persistence API
JPQL	Java Persistence Query Language
JSF	JavaServer Faces Reference Implementation (RI) Version: 1.1_01 FCS
JSON-B 1.0	JSR 367 Java API for JSON Binding（JSON-B） 1.0
JSON-P 1.1	JSR 374 Java API for JSON Processing
JSP	JavaServer Pages
	JSP
JSR	Java Specification Request
JSSE	Java Secure Socket Extension
JST	Japan Standard Time
JSTL	JavaServer Pages Standard Tag Library
JTA	Java Transaction API
JTS	Java Transaction Service
JVMDI	Java Virtual Machine Debug Interface
JVMPI	Java Virtual Machine Profiler Interface
JVMTI	Java Virtual Machine Tool Interface
LAN	Local Area Network
LB	Load Balancer
LDAP	Lightweight Directory Access Protocol
LDIF	LDAP Data Interchange Format
LF	Line Feed
MAC	Message Authentication Code
MB/s	Megabyte per Second
MBean	Managed Bean
Mbit/s	Megabit per Second
MDA	Model Driven Architecture

英略語	英字での表記
MDB	Message-Driven Bean
MHP	Message Handling Program
MIB	Management Information Base
MIME	Multipurpose Internet Mail Extensions
MIPS	Million Instructions Per Second
MTU	Maximum Transmission Unit
MVC	Model View Controller
NIC	Network Interface Card
NTP	Network Time Protocol
OAEP	Optimal Asymmetric Encryption Padding
OASIS	Organization for the Advancement of Structured Information Standards
OID	Object Identifier
OLTP	On-Line Transaction Processing
OMG	Object Management Group
ORB	Object Request Broker
OS	Operating System
OTM	Object Transaction Monitor
OTS	Object Transaction Service
PK	Primary Key
PKI	Public Key Infrastructure
POA	Portable Object Adapter
POJO	Plain Old Java Object
POP3	Post Office Protocol - Version 3
PTP	Point-to-Point
QName	Qualified Name
QoS	Quality of Service
RAC	Real Application Clusters
RAR	Resource Adapter Archive

英略語	英字での表記
	Roshal Archive
RD または RDB	Relational Database
REST	Representational State Transfer
RFC	Request For Comments
RMD	Reliable Messaging Destination
RMI	Remote Method Invocation
RMS	Reliable Messaging Source
RPC	Remote Procedure Call
RSA	Rivest, Shamir and Adleman
SAAJ	SOAP with Attachments API for Java
SaaS	Software as a Service
SAS	Serial Attached SCSI
SAX	Simple API for XML
SAX1	Simple API for XML 1.0
SAX2	Simple API for XML 2.0
SDK	Software Development Kit
SEI	Service Endpoint Interface
Servlet またはサーブレット	Java Servlet
SFO	Session Fail Over
SHA	Secure Hash Algorithm
SMAP	Source Map
SMTP	Simple Mail Transfer Protocol
SMTPS	SMTP over SSL
SNMP	Simple Network Management Protocol
SOA	Service Oriented Architecture
SOAP	Simple Object Access Protocol
SpecInt	Standard Performance Evaluation Corporation Integer benchmark
SPI	Service Provider Interface
SPP	Service Providing Program

英略語	英字での表記
ssh	Secure Shell
SSL	Secure Sockets Layer
StAX	Streaming API for XML
SUP	Service Using Program
TCP	Transmission Control Protocol
TCS	Transaction Context Server
TLD	Tag Library Descriptor
TLS	Transport Layer Security
TrAX	Transformation API for XML
TSC	Time Stamp Counter
TSV	Tab Separated Values
UAC	User Account Control
UAP	User Application Program
UCS	Universal multi-octet coded Character Set
UDDI	Universal Description, Discovery and Integration
UML	Unified Modeling Language
UNC	Universal Naming Convention
UOC	User Own Coding
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
URN	Uniform Resource Name
UTC	Universal Time Coordinated
UTF	UCS Transformation Format
UTF-8	8-bit UCS Transformation Format
VM	Virtual Machine
WAR	Web Archive
WebSocket 1.1	JSR 356 Java API for WebSocket
W3C	World Wide Web Consortium
WFA	Work-Flow Architecture

英略語	英字での表記
WS	Web Service
WSDL	Web Services Description Language
WSDL4J	Web Services Description Language for Java Toolkit
WS-I	Web Services Interoperability
WS-R	Web Services Reliability
WS-RM	Web Service Reliable Messaging
WST	Web Standard Tools
WTP	Eclipse Web Tools Platform
XML	Extensible Markup Language
XPath	XML Path Language
XSL	Extensible Stylesheet Language
XSLT	XSL Transformations

■ KB（キロバイト）などの単位表記について

1KB（キロバイト），1MB（メガバイト），1GB（ギガバイト），1TB（テラバイト）はそれぞれ 1,024 バイト，$1{,}024^2$ バイト，$1{,}024^3$ バイト，$1{,}024^4$ バイトです。

目次

1 クラウドサービスプラットフォームとしてのアプリケーションサーバおよび BPM/ESB 基盤

この章では，アプリケーションサーバおよび BPM/ESB 基盤の概要について説明します。アプリケーションサーバおよび BPM/ESB 基盤は，クラウドを実現するためのプラットフォームとして位置づけられる製品です。

アプリケーションサーバは，業務システムの中核に位置し，アプリケーションを実行する基盤となる製品です。標準技術である Java EE 8 に対応した実行環境を構築・運用できます。

また，BPM/ESB 基盤は，BPEL 準拠のビジネスプロセス管理とエンタープライズサービスバスの機能によって，ビジネスニーズに合った業務システムの最適化と変化への即応を実現する基盤となる製品です。アダプタによって，さまざまな種類のシステムをサービスとして統合できます。

1.1 クラウドサービスプラットフォームでのアプリケーションサーバと BPM/ESB 基盤の位置づけ

アプリケーションサーバおよび BPM/ESB 基盤は，クラウドサービスプラットフォームとして，効率的なサービスの作成，実行を実現する製品です。

クラウドサービスを提供するシステムでは，常に変化し続けるビジネス環境で求められるサービスを，スピーディに作成，実行することが不可欠です。アプリケーションサーバと BPM/ESB 基盤は，サービスを作成・実行する基盤として位置づけられます。

アプリケーションサーバは，作成したサービスを用途に応じた形態で効率的に実行するためのオンライン処理・バッチ処理を実行する基盤となります。**BPM/ESB 基盤**は，複数のサービスを業務に応じて組み合わせ，1 つの新しいサービスとして実行するための制御を実現する基盤となります。

ここでは，クラウドサービスプラットフォームでのアプリケーションサーバおよび BPM/ESB 基盤の位置づけについて説明します。

1.1.1 アプリケーションサーバの位置づけ

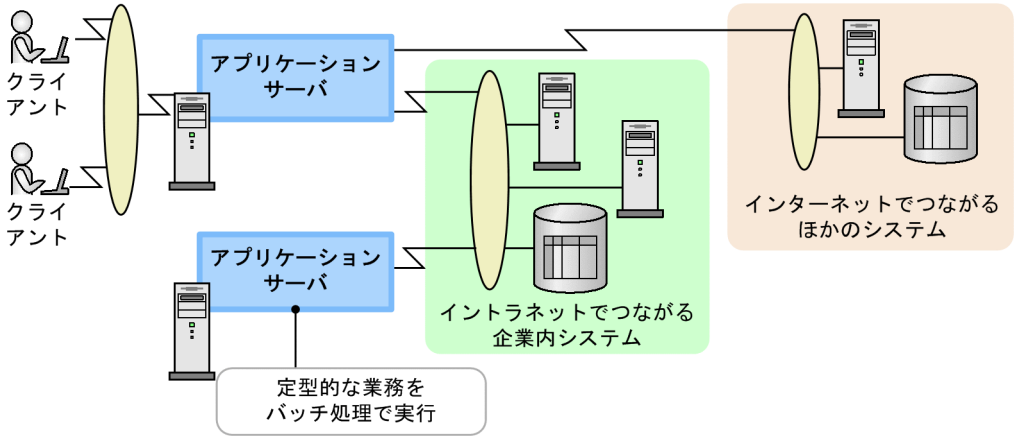
情報社会が急速に変化している現在，ビジネスを成長させるためには市場ニーズに柔軟に対応できるインフラが必要です。このようなインフラを実現するためには，拡張性に優れた情報システムの構築が必須であり，情報システムの優劣はビジネスの成功を左右する程の影響力を持ちます。さらに，社会変化のスピードに合わせて業務のライフサイクルが短くなっていくのに伴って，業務の変化に柔軟に対応でき，かつ既存資産をむだなく再利用できることも，情報システムの必須要件になってきています。

アプリケーションサーバは，このような要件に応えます。

アプリケーションサーバは，ビジネス環境やビジネス戦略の急激な変化に柔軟に対応できる，信頼性と拡張性の高い情報システムの基盤となる製品です。標準技術である Java EE に準拠したアプリケーションの実行基盤を構築できるので，Java EE の特長である柔軟性，信頼性の高いシステムを効率的に開発・運用できるようになります。

アプリケーションサーバの概要を次の図に示します。

図 1-1　アプリケーションサーバの概要



アプリケーションサーバは，クライアントからオンラインで送信されたリクエストを基に業務を実行する基盤，および定型的な業務をバッチ処理で実行するための基盤になります。

また，アプリケーションサーバは，Java のほか，CORBA などの業界標準に準拠しています。このため，ほかのシステムとの連携や，ほかのシステムからの移植も円滑に実現できます。

1.1.2　BPM/ESB 基盤の位置づけ

BPM/ESB 基盤では，サービスプラットフォームで SOA（Service Oriented Architecture（サービス指向アーキテクチャ））を適用したシステムを実現するための機能を提供しています。ここでは，BPM/ESB 基盤が実現するサービスプラットフォームの目的と位置づけについて説明します。

サービスプラットフォームとは，SOA を実現するシステムの開発・運用の基盤となる製品です。統一された開発・運用環境でビジネスプロセスからサービスの接続までを構築・実行できます。そのため，SOA の利点を引き出して，サービスを柔軟に組み合わせて新しいシステムを迅速に構築・実行できます。この中心となるのが，サービスをプロセスで統合する「プロセス統合」です。プロセス統合を実現するのが，サービスプラットフォームです。サービスプラットフォームを使用したシステムの実現例を次の図に示します。

図 1-2　サービスプラットフォームを使用したシステムの実現例

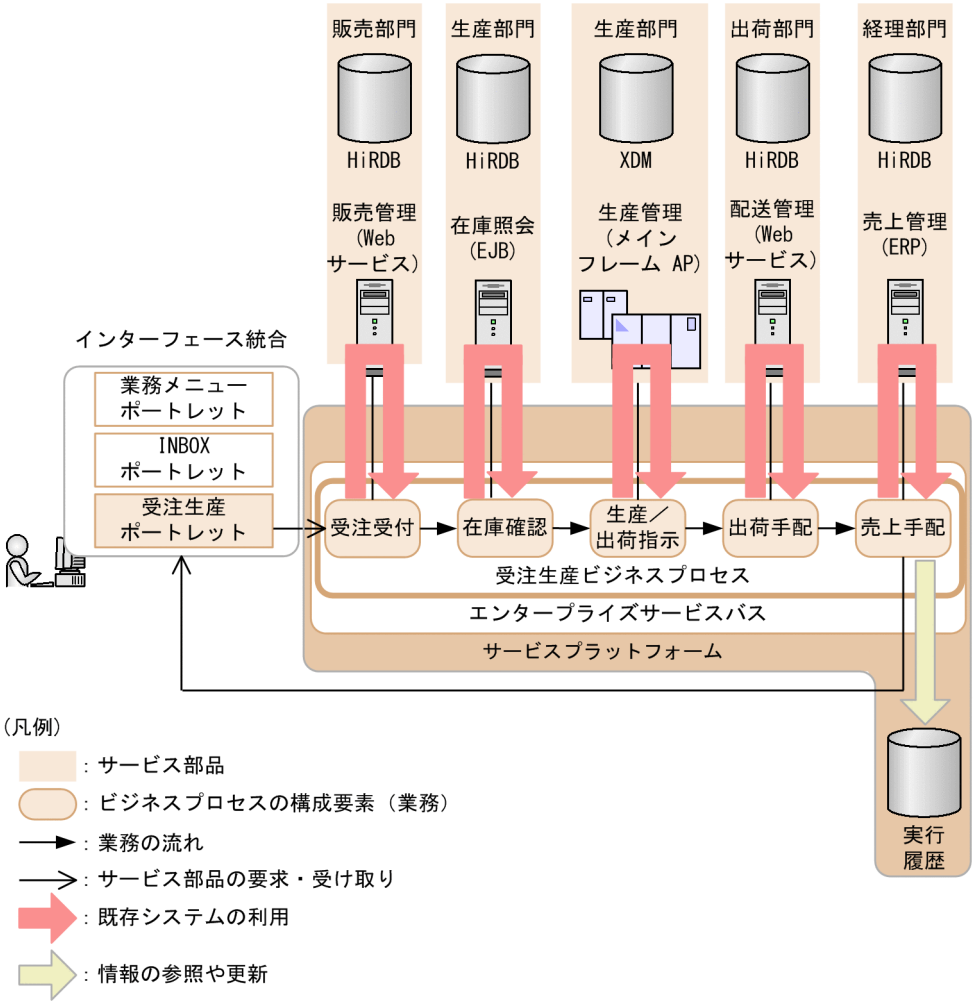


図 1-2 は，生産物流システムに SOA を適用した例です。業務の流れに沿って，サービスを自動的に呼び出せます。図 1-2 の場合，次のような利点があります。

- 業務を段階的にオープン化する場合の対応が容易になります。
- 実業務に応じたビジネスプロセスを実現できます。
- 在庫状況や生産進捗状況を的確に把握し，迅速な納期回答ができます。
- リードタイムを短縮できます。

サービスプラットフォームの実行環境および運用環境には，アプリケーションサーバの実行環境の機能に加えてサービス統合を実現するための機能があります。SOA の中心であるエンタープライズサービスバス機能を持ち，サービスを自由に組み合わせて，実行する戦略の変化に即応したシステムを構築できます。既存システムから切り出したサービスや外部から提供されるサービスも自由に組み合わせて，信頼性の高いシステムを構築できます。

サービスプラットフォームの開発環境には，アプリケーションサーバの開発環境の機能に加えてサービス統合を実現するための機能があります。ビジネスプロセス定義，データ変換定義，およびサービスアダプ

タ定義など，プロセス統合に必要な定義ツールを Eclipse の Plug-in として使用できます。ビジネスプロセスからサービスの接続まで，Eclipse 上の一連の操作でプロセス統合ができます。

SOA を適用したシステム開発手法の中で，サービスプラットフォームの開発環境では，インタフェースを含むビジネスプロセスの詳細設計から実装・テストまでをサポートしています。サービスプラットフォームの機能を利用すれば，コンポーネントの設計・実装ができます。

これによって，すでにアプリケーションの実行環境やサービスプラットフォーム以外の環境で稼働しているサービスを統合して，新しいサービスとしてユーザに提供できます。

1.2　アプリケーションサーバと BPM/ESB 基盤との関係

この節では，アプリケーションサーバと BPM/ESB 基盤との関係について説明します。

アプリケーションサーバは，業務内容に応じたサービスを実現するためのアプリケーションを開発するためのアプリケーション開発環境と，開発したアプリケーションを実行してユーザにサービスを提供するためのアプリケーション実行環境から構成されています。

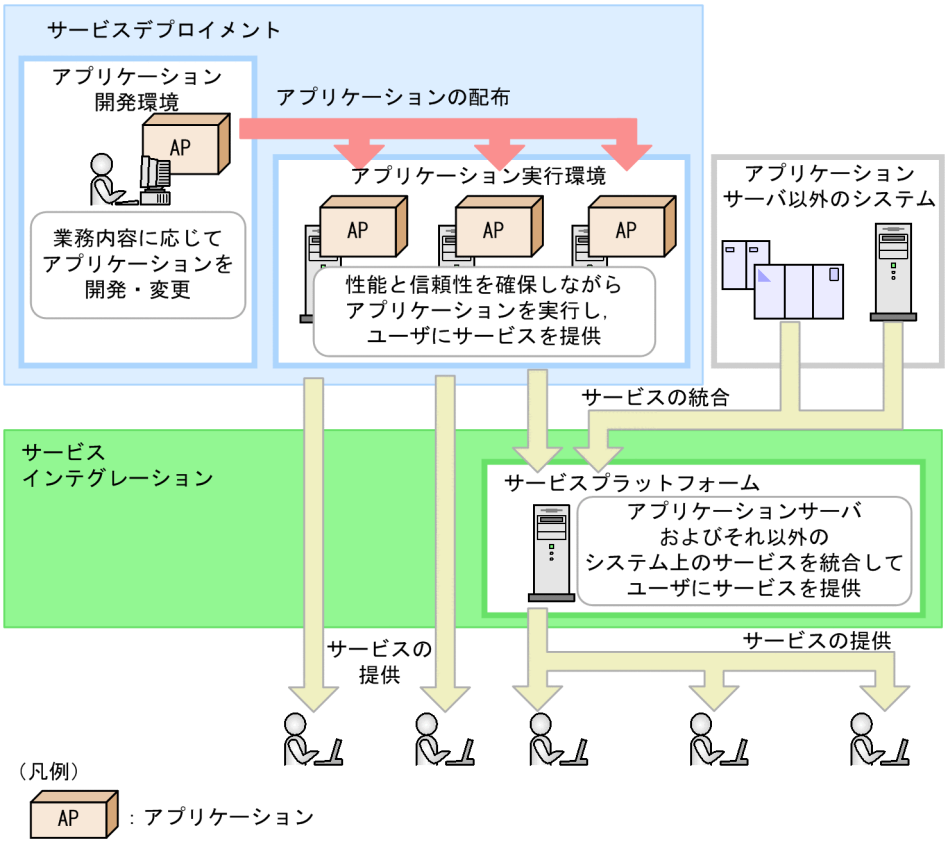
BPM/ESB 基盤であるサービスプラットフォームは，BPEL 準拠のビジネスプロセス管理を実現する機能と，サービスの統合を実現するエンタープライズサービスバスの機能を持つ製品です。

ESB では，業務に必要なアプリケーションをサービスとして利用します。既存のサービスや新規に作成したサービスを組み合わせて，新しいサービスの迅速な提供を実現します。複数のサービスを組み合わせる場合の実行順序は，ビジネスプロセスで定義します。

これらの機能によって，新規アプリケーションと既存アプリケーションを組み合わせた，ワンストップサービスを実現できます。

アプリケーションサーバとサービスプラットフォームを組み合わせた例を次の図に示します。

図 1-3　アプリケーションサーバとサービスプラットフォームを組み合わせた例



アプリケーション実行環境では，性能と信頼性を確保しながら，ユーザからのリクエストを迅速に処理して，ユーザにサービスを提供します。また，バッチジョブを効率良く実行します。可用性や耐障害性の高い実行基盤として，24 時間連続稼働が必要なサービス，停止することが許されない重要な業務，バッチ

ジョブとして実行される基幹業務など，さまざまな要件のアプリケーションを実行するための環境になります。また，セキュリティに配慮したシステムや，内部統制に対応したシステムを実現します。可用性を確保するために，1 つのアプリケーションを複数の実行環境で実行することもあります。

業務内容の変更や組織の組み替えなどが発生した場合に，新しいアプリケーションを開発したり，既存のアプリケーションを変更したりするためには，**アプリケーション開発環境**を使用できます。アプリケーション開発環境では，画面設計から，バックエンドシステムと連携する業務処理の実装まで，シームレスに効率良く実行できます。

業務内容の変更や組織の組み替えなどに対して，既存のアプリケーションや他システムが提供するサービスの再利用で対応する場合は，**サービスプラットフォーム**を使用します。サービスプラットフォームでは，アプリケーション実行環境が提供するサービスやアプリケーションサーバ以外が提供するサービスを統合して，新しいサービスとしてユーザに提供できます。また，サービスプラットフォームは，複数のアプリケーションやシステムから提供される既存のサービスを，統一されたインタフェースで提供するための基盤としても使用できます。

2 アプリケーションサーバおよび BPM/ESB 基盤の製品構成

この章では，アプリケーションサーバおよび BPM/ESB 基盤の製品構成について説明します。アプリケーションサーバおよび BPM/ESB 基盤には，実行環境を構築する製品，開発環境を構築する製品，およびそれらの環境で必要に応じて使用するオプション製品があります。また，これらの製品は，構成ソフトウェアと呼ばれる複数のソフトウェアで構成されています。

2.1　製品の分類と特長

製品は用途ごとに，次のように分類できます。製品の分類を次の表に示します。

表 2-1　製品の分類

分類	製品
アプリケーションサーバの実行環境を構築する製品	• Application Server
アプリケーションサーバの開発環境を構築する製品	• Developer
BPM/ESB 基盤の実行環境を構築する製品	• Service Platform
BPM/ESB 基盤の開発環境を構築する製品	• Service Architect
オプション製品	• Client

この節では，分類ごとの特長について説明します。

2.1.1　アプリケーションサーバの実行環境を構築する製品

Application Server は実行環境を構築する製品です。Java EE に準拠した J2EE アプリケーションを，サーバ側で実行するための実行環境を構築します。Web サービスに対応したアプリケーションや，メッセージによる非同期通信に対応したアプリケーションも実行できます。クライアントには，Web ブラウザ，または EJB クライアントアプリケーションを使用できます。

なお，実行環境では，システム構成に応じてオプション製品も利用できます。

2.1.2　アプリケーションサーバの開発環境を構築する製品

Developer は開発環境を構築する製品です。アプリケーションサーバの実行環境で実行する J2EE アプリケーションの開発環境を構築します。IDE などを使用して開発した J2EE アプリケーションに対して，アセンブル，デバッグ，デプロイ，テストなどができます。

また，J2EE アプリケーションの開発に Eclipse を使用する場合，WTP（Eclipse Web Tools Project）を使用できます。WTP は，eclipse.org によって提供されている，Eclipse 上でのアプリケーション開発で使用するプラグインをまとめた統合開発環境です。WTP を使用すると，コーディング，テスト，デバッグまで，開発の一連の操作をまとめて実行できます。

なお，Developer は Windows 環境だけで使用できます。

2.1.3　BPM/ESB 基盤の実行環境を構築する製品

Service Platform は，BPM/ESB 基盤の実行環境を構築する製品です。

SOA を適用したシステムを運用します。また，Service Architect で開発した内容をセットアップしたり，Service Architect 側に開発に必要なシステム情報を受け渡したりします。

Service Platform を利用した運用の詳細については，マニュアル「サービスプラットフォーム システム構築・運用ガイド」を参照してください。

2.1.4　BPM/ESB 基盤の開発環境を構築する製品

Service Architect は，BPM/ESB 基盤の開発環境を構築する製品です。

SOA を適用したシステムに必要なサービスを呼び出すための定義，ビジネスプロセス，データ変換の定義などを開発します。Service Architect で開発した内容は，Service Platform を利用してセットアップします。

Service Architect を利用したシステム開発の詳細については，マニュアル「サービスプラットフォーム 開発ガイド 基本開発編」，およびマニュアル「サービスプラットフォーム 開発ガイド 受付・アダプタ定義編」を参照してください。

2.1.5　オプション製品

アプリケーションサーバで実行環境を構築する場合に，システム構成に応じて使用できる製品です。

業務システム内に，用途を限定したマシンを配置する場合に使用できます。

オプション製品を次に示します。

- Client

  実行環境上の Enterprise Bean をクライアントマシンのアプリケーションから直接呼び出す構成の場合に使用できます。また，Web サービス実行環境上の Web サービスを，クライアントマシンのアプリケーションから呼び出す構成の場合にも使用できます。

2.2　構成ソフトウェア

この節では，アプリケーションサーバおよびBPM/ESB基盤の**構成ソフトウェア**について説明します。構成ソフトウェアは，単体で動作させるのではなく，ほかの構成ソフトウェアの機能と組み合わせて動作させることによって，アプリケーションサーバおよびBPM/ESB基盤としての機能を実現します。

構成ソフトウェアは，アプリケーションサーバおよびBPM/ESB基盤の機能に対応します。製品に含まれている構成ソフトウェアの種類によって，実現できる機能が異なります。

2.2.1　製品と構成ソフトウェアの対応

ここでは，製品と，その製品に含まれる構成ソフトウェアの対応を示します。

製品に対応する構成ソフトウェアを，次の表に示します。

表 2-2　製品と構成ソフトウェアの対応

構成ソフトウェア名	アプリケーションサーバ		BPM/ESB基盤		オプション製品
	Application Server	Developer	Service Platform	Service Architect	Client
Application Development Plug-in	−	○	−	○	−
Component Container	○	○	○	○	−
Component Container - Client	−	−	−	−	○
Component Container - Redirector	○	○	○	○	−
Component Transaction Monitor	○	○	○	○	−
Developer's Kit for Java	○	○	○	○	○
HTTP Server	○	○	○	○	−
Performance Tracer	○	○	○	○	○
Reliable Messaging	○	○	○	○	−
Service Coordinator	−	−	○	○	−
Service Development Plug-in	−	−	−	○	−
TPBroker	○	○	○	○	○
Web Services - Security	○	○	○	○	−

構成ソフトウェア名	アプリケーションサーバ		BPM/ESB 基盤		オプション製品
	Application Server	Developer	Service Platform	Service Architect	Client
XML Processor	○	○	○	○	○
HiRDB/Single Server Version 10	−	○	−	○	−

（凡例）

○：含まれます。 −：含まれません。

2.2.2 構成ソフトウェアの機能概要

それぞれの構成ソフトウェアの機能概要について説明します。

(1) Application Development Plug-in

開発環境で使用する次の機能を提供する構成ソフトウェアです。

- **開発環境のインスタントセットアップ機能**
  ウィザードプログラム（GUI）によってアプリケーションのデバッグ環境をセットアップできる機能です。
- **Eclipse セットアップ機能**
  Eclipse を使用した J2EE アプリケーション開発に必要な Eclipse 環境をセットアップできる機能です。

(2) Component Container

アプリケーションの実行基盤の中核として，次のような機能を提供する構成ソフトウェアです。

- J2EE アプリケーションの実行環境（J2EE サーバ）としての機能
- バッチアプリケーションの実行環境（バッチサーバ）としての機能
- Web サービスの実行・開発環境としての機能
- アプリケーションサーバを運用管理する機能

それぞれの機能の概要を示します。

(a) J2EE アプリケーションの実行環境（J2EE サーバ）としての機能

サーバサイドの業務処理プログラム（ビジネスロジック）をコンポーネントとして実行するためのフレームワークである，J2EE サーバを実現するための機能です。Web コンテナ，EJB コンテナなどの機能を含みます。Java Platform, Enterprise Edition（Java EE）に含まれる仕様に準拠しています。対応している仕様の詳細については，「4.6.2 アプリケーションサーバが対応する標準仕様」を参照してください。

さらに，CTM 機能による Enterprise Bean に対する動的負荷分散，流量制御，優先制御およびサービス閉塞に対応するための基盤機能も提供します。

(b)　バッチアプリケーションの実行環境（バッチサーバ）としての機能

バッチアプリケーションをサーバで実行するための機能です。バッチ処理の処理内容を Java で実装した Java アプリケーションを実行できます。サーバ上でアプリケーションを動作させることによって，JavaVM の起動コストを抑えられます。コネクションプールやステートメントプールを使用した効率の良いデータベースアクセスや，FullGC の制御なども実現できます。

(c)　Web サービスの実行・開発環境としての機能

Web サービスの実行環境および開発環境としての機能です。次のエンジンや API などを提供します。

- JAX-WS 仕様に準拠した SOAP Web サービスおよびクライアントの実行に必要な JAX-WS エンジンと，開発に必要な API・コマンド
- JAX-RS 仕様に準拠した RESTful Web サービス (Web リソース) のサーバおよびクライアントの実行に必要な JAX-RS エンジンと，開発に必要な API

> **参考**
>
> 既存機能である SOAP アプリケーション開発支援機能も使用できます。

(d)　アプリケーションサーバを運用管理する機能

アプリケーションサーバを運用管理するための機能です。次のような運用管理を実現できます。

- アプリケーションサーバの一括構築・一括運用
- J2EE サーバ内のアプリケーションやリソースの設定
- 複数の J2EE アプリケーションのユーザ管理機能をシームレスに連携した統合ユーザ管理
- アプリケーションサーバの各機能が出力するログの収集

また，JP1 などのほかの運用管理プログラム製品と連携して，アプリケーションサーバの運用管理をするためのコマンドも提供しています。

(3)　Component Container - Client

EJB クライアントアプリケーションの実行環境を構築するための構成ソフトウェアです。Component Container のサブセットです。

(4)　Component Container - Redirector

アプリケーションサーバの V9 互換モードを使用する際に，Web コンテナを Web サーバと連携させるための構成ソフトウェアです。この構成ソフトウェアで提供されるリダイレクタモジュールを Web サーバ

に登録することで，Web サーバあての HTTP リクエストのうち，特定のリクエストを指定した Web コンテナに処理させたり，複数の Web コンテナにリクエストを振り分けて処理させたりできます。

(5) Component Transaction Monitor

Enterprise Bean に対するクライアントからのリクエストをスケジューリングして，負荷分散や流量制御を実現する構成ソフトウェアです。アプリケーションごとにキューを管理して J2EE サーバの負荷状況に応じて処理を分散させたり，一度に処理するリクエストの数を制御したり，業務処理プログラムを入れ替える時に特定の J2EE アプリケーションだけを閉塞させたりできます。これによって，システムが安定した状態で運転し続けることができるので，業務システムの可用性と信頼性が向上します。

また，バッチアプリケーションの実行もスケジューリングできます。

(6) Developer's Kit for Java

Java Platform, Standard Edition 11 および Java Platform, Standard Edition 17 に準拠した構成ソフトウェアです。対応する Oracle 社製の JDK のバージョンは JDK 11，および JDK 17 です。使用できる機能，コマンドおよび API については，Oracle 社が提供している JDK 11 または JDK 17 のドキュメントを参照してください。

(7) HTTP Server

Apache HTTP Server をベースに Secure Sockets Layer（SSL）をサポートしたミッションクリティカル分野向けの Web サーバです。

(8) Performance Tracer

処理性能のボトルネックを解析するためのトレース情報を出力する構成ソフトウェアです。アプリケーションサーバで構築したシステムでは，リクエストが処理される時に，決められたポイントごとに性能解析用のトレース情報を出力します。この情報を収集して分析することで，システムのボトルネックが調査できます。また，障害が発生した場合には，障害の発生個所を特定することもできます。

(9) Reliable Messaging

業務コンポーネント間，社内システム間，および社内システムと社外システム間で，非同期に高信頼なメッセージ送受信を実現する高信頼メッセージング基盤としての機能を提供する構成ソフトウェアです。WS-Reliability の仕様に準拠しています。

(10) Service Coordinator

SOA を適用したシステムを構築・運用するための機能を提供する構成ソフトウェアです。次の機能が含まれます。

- ビジネスプロセス実行機能（HCSC-Business Process）

- データ変換機能（HCSC-Data Transform）
- 実行環境の運用管理機能（HCSC-Manager）
- メッセージング制御・サービス連携機能（HCSC-Messaging）
- 受付・アダプタ

(11) Service Development Plug-in

Service Coordinator の開発環境です。SOA を適用したシステムに必要な受付，アダプタ，ビジネスプロセス，データ変換などの定義作成を支援します。

(12) TPBroker

サーブレットまたは JSP と Enterprise Bean の間などの通信で使用される，Java EE での RMI-IIOP 通信基盤，および分散トランザクション基盤になる構成ソフトウェアです。また，Java EE 環境での CORBA クライアントアプリケーションの実行環境および開発環境を提供します。また，EJB クライアントアプリケーションの実行環境で使用する，RMI-IIOP 通信基盤，および CORBA クライアントアプリケーションの実行環境としての機能も提供します。

(13) Web Services - Security

SOAP Web サービスのセキュリティの標準規格 WS-Security に準拠した機能を提供する構成ソフトウェアです。SOAP メッセージに対して，XML 署名を付けたり，SOAP メッセージを暗号化したりします。Web Services - Security の利用によって，セキュアな状態で SOAP メッセージの送受信ができます。また，SOAP メッセージに付いている署名を検証したり，暗号化された SOAP メッセージを復号化したりできます。

XML 署名の生成・検証，または XML 暗号によるデータの暗号化・復号化を行うアプリケーションの開発を支援する機能も提供しています。

(14) XML Processor

業界標準の JAXP/JAXB をサポートした XML ドキュメントの読み取り，操作および生成を実行するための機能を提供します。

(15) HiRDB/Single Server Version 10

アプリケーション開発時，テストおよびデバッグに使用できるデータベースです。

2.3　アプリケーションサーバの動作環境

アプリケーションサーバの動作環境について説明します。

2.3.1　前提ソフトウェア

アプリケーションサーバの前提ソフトウェアについて説明します。

(1)　Web 環境

前提となる Web 環境について示します。なお，ロードバランサや SSL アクセラレータから直接 Web コンテナにリクエストを送信する場合，Web サーバは不要です。

(a)　Web サーバ

前提となる Web サーバについて次の表に示します。

表 2-3　アプリケーションサーバの前提となる Web サーバ

前提 OS	前提製品
Windows	Microsoft IIS HTTP Server※
UNIX	HTTP Server※

注※　アプリケーションサーバの構成ソフトウェアです。

(b)　Web ブラウザ

前提となる Web ブラウザについて，次の表に示します。

表 2-4　アプリケーションサーバの前提となる Web ブラウザ

Web 環境	前提製品
Web ブラウザ	HTTP1.0 または HTTP1.1 に対応したブラウザ。 ただし，運用管理ポータルの前提は次のブラウザです。 • Microsoft Edge（ネイティブ） • Microsoft Edge（Internet Explorer モード） ほかのブラウザを使用した場合は表示が崩れることがあります。

(2)　言語

前提となる言語は，Java です。

(3) データベース

前提となるデータベースについては，「6.1　データベースとの連携」を参照してください。

2.4　BPM/ESB 基盤の動作環境

BPM/ESB 基盤が実現するサービスプラットフォームの動作環境について説明します。

2.4.1　前提ソフトウェア

サービスプラットフォームの前提ソフトウェアについて説明します。

(1)　前提となるデータベース

サービスプラットフォームの前提となるデータベースについて，次の表に示します。

表 2-5　サービスプラットフォームの前提データベース

前提データベース	バージョン
HiRDB	HiRDB Server Version 9 HiRDB Server Version 10
Oracle	Oracle 11g Oracle 12c Oracle 18c Oracle 19c

2.4.2　関連ソフトウェア

サービスプラットフォームの関連ソフトウェアについて説明します。

(1)　Windows Server Failover Cluster

実行環境で，2 つの HCSC サーバを組み合わせて HA クラスタを構成する場合に利用します。Windows の場合にだけ使用できます。

HCSC サーバのクラスタ構成については，マニュアル「サービスプラットフォーム 解説」の「1.4.2　クラスタソフトウェアを利用した HCSC サーバの冗長構成」を参照してください。

(2)　HA モニタ

実行環境で，2 つの HCSC サーバを組み合わせて HA クラスタを構成する場合に利用します。UNIX の場合にだけ使用できます。

HCSC サーバのクラスタ構成については，マニュアル「サービスプラットフォーム 解説」の「1.4.2　クラスタソフトウェアを利用した HCSC サーバの冗長構成」を参照してください。

(3) TP1/Server Base Enterprise Option

サービスプラットフォームの実行環境と OpenTP1 の間で，データベースを介した，異なるコンポーネントでの通信を行う場合に利用します。

TP1/Server Base Enterprise Option は DB キューのプロトコルをサポートしています。サービスプラットフォームの実行環境とは，DB キューの受け付けとサービスアダプタで連携できます。

(4) Code Converter - Development Kit for Java

サービスプラットフォームの開発環境で，次に示す場合に利用します。

- 外字マッピングをユーザ独自で行う場合
- 文字コード変換 UOC を使用してコード変換をカスタマイズする場合

この製品は，Windows プラットフォーム専用です。

(5) Code Converter - Server Runtime for Java

サービスプラットフォームの実行環境で，次に示す場合に利用します。

- 外字マッピングをユーザ独自で行う場合
- 文字コード変換 UOC を使用してコード変換をカスタマイズする場合

この製品は，Windows プラットフォーム専用です。

(6) Code Converter - Runtime for Java

サービスプラットフォームの実行環境で，次に示す場合に利用します。

- 外字マッピングをユーザ独自で行う場合
- 文字コード変換 UOC を使用してコード変換をカスタマイズする場合

この製品は UNIX プラットフォーム専用です。

(7) JP1 関連製品

サービスプラットフォームで構築した業務システム全体の監視，問題の検知などの運用を効率良く実施する場合に利用します。

JP1 と連携したシステムの運用については，マニュアル「アプリケーションサーバ 機能解説 運用／監視／連携編」の「12. JP1 と連携したシステムの運用」を参照してください。

3 マニュアル体系と読書手順

この章では，アプリケーションサーバおよび BPM/ESB 基盤のマニュアルの体系と読書手順について説明します。

3.1　マニュアル体系

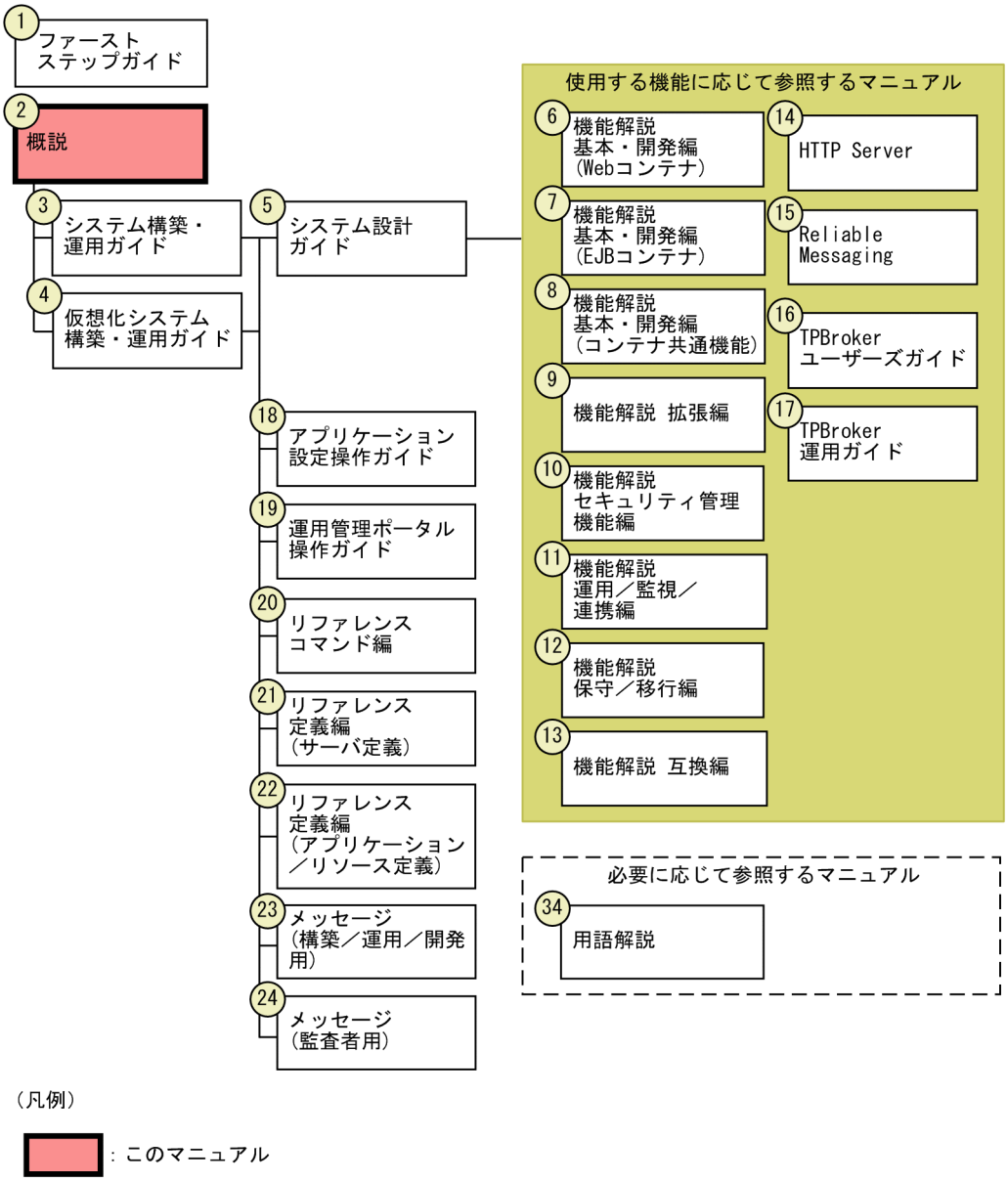
ここでは，アプリケーションサーバと BPM/ESB 基盤のマニュアル体系について説明します。なお，BPM/ESB 基盤を使用する場合，BPM/ESB 基盤独自のマニュアルに加えて，必要に応じてアプリケーションサーバのマニュアルを参照してください。

図中のマニュアル名の「アプリケーションサーバ & BPM/ESB 基盤」および「アプリケーションサーバ」は省略しています。

3.1.1　アプリケーションサーバのマニュアル体系

アプリケーションサーバのマニュアル体系を，次の図に示します。

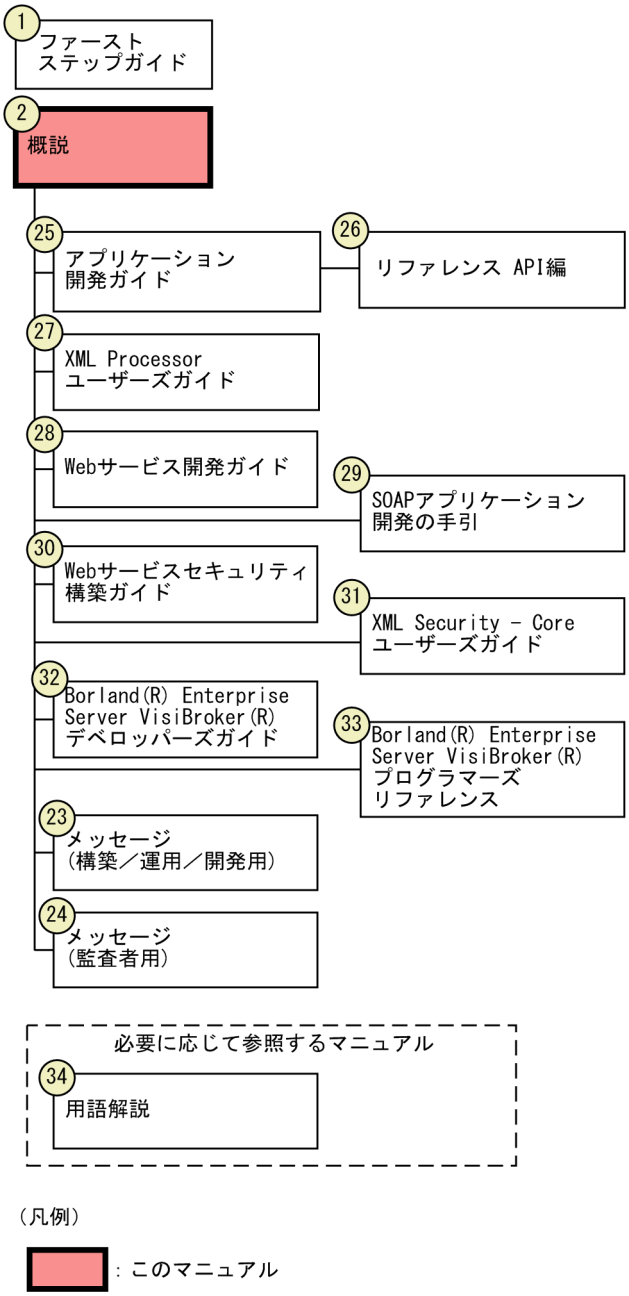
図 3-1　アプリケーションサーバのマニュアル体系（主に実行環境を構築・運用する場合に参照するマニュアル）



（凡例）

：このマニュアル

図 3-2　アプリケーションサーバのマニュアル体系（主にアプリケーションを開発する場合に参照するマニュアル）



それぞれのマニュアルで説明している内容の概要を次の表に示します。なお，表の項番は図中の項番と対応しています。

表 3-1　アプリケーションサーバのマニュアル概要

項番	マニュアル名	内容
1	アプリケーションサーバ ファーストステップガイド	サンプルプログラムを動作させるための開発環境または実行環境の構築手順について説明しています。

項番	マニュアル名	内容
2	アプリケーションサーバ & BPM/ESB 基盤 概説	アプリケーションサーバおよび BPM/ESB 基盤の製品概要について説明しています。このマニュアルです。
3	アプリケーションサーバ システム構築・運用ガイド	アプリケーションサーバを導入してシステムを構築・運用する方法について説明しています。アプリケーションサーバを本番稼働用の環境に導入する場合は，まずこのマニュアルを参照してください。
4	アプリケーションサーバ 仮想化システム構築・運用ガイド	アプリケーションサーバを仮想化したサーバ上に構築する場合の設計，構築，運用の手順について説明しています。
5	アプリケーションサーバ システム設計ガイド	「アプリケーションサーバ システム構築・運用ガイド」で説明しているシステム構成以外の構成でシステムを構築・運用したい場合に参照するマニュアルです。 システムの目的に応じたシステム構成や運用方法を検討するための指針について説明しています。また，チューニングの方法についても説明しています。
6	アプリケーションサーバ 機能解説 基本・開発編(Web コンテナ)	アプリケーションサーバの機能の詳細について，アプリケーションの実装方法や実行環境で必要な設定などを含めて解説しています。 このマニュアルでは，J2EE サーバの機能のうち，Web アプリケーションの実行基盤である Web コンテナの機能について説明しています。また，Web サーバと連携して実現する機能についても説明しています。
7	アプリケーションサーバ 機能解説 基本・開発編(EJB コンテナ)	アプリケーションサーバの機能の詳細について，アプリケーションの実装方法や実行環境で必要な設定などを含めて解説しています。 このマニュアルでは，J2EE サーバの機能のうち，Enterprise Bean の実行基盤である EJB コンテナの機能について説明しています。
8	アプリケーションサーバ 機能解説 基本・開発編(コンテナ共通機能)	アプリケーションサーバの機能の詳細について，アプリケーションの実装方法や実行環境で必要な設定などを含めて解説しています。 このマニュアルでは，J2EE サーバの機能のうち，Web コンテナ，EJB コンテナの両方で使用できる機能について説明しています。
9	アプリケーションサーバ 機能解説 拡張編	アプリケーションサーバの機能の詳細について，アプリケーションの実装方法や実行環境で必要な設定などを含めて解説しています。 このマニュアルでは，アプリケーションサーバ独自の拡張機能について説明しています。
10	アプリケーションサーバ 機能解説 セキュリティ管理機能編	アプリケーションサーバの機能の詳細について，アプリケーションの実装方法や実行環境で必要な設定などを含めて解説しています。 このマニュアルでは，アプリケーションサーバを中心としたシステムのセキュリティを確保するための機能について解説しています。また，セキュリティを確保するためのシステム構成や運用方法についても説明しています。
11	アプリケーションサーバ 機能解説 運用／監視／連携編	アプリケーションサーバの機能の詳細について，アプリケーションの実装方法や実行環境で必要な設定などを含めて解説しています。 このマニュアルでは，システム運用時に使用する機能，システムを監視するための機能，およびほかの製品と連携するための機能について説明しています。
12	アプリケーションサーバ 機能解説 保守／移行編	アプリケーションサーバの機能の詳細について，アプリケーションの実装方法や実行環境で必要な設定などを含めて解説しています。

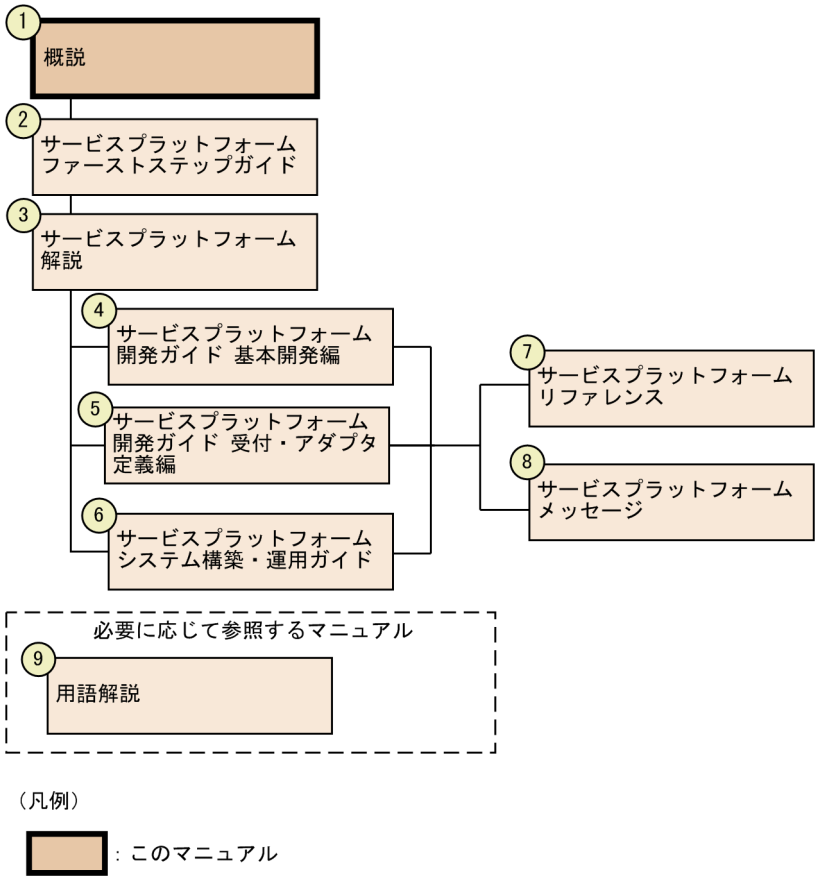
項番	マニュアル名	内容
		このマニュアルでは，トラブルが発生した場合のシステム保守に必要な機能，および製品のバージョンアップに伴うシステムの移行について説明しています。
13	アプリケーションサーバ 機能解説 互換編	アプリケーションサーバの機能の詳細について，アプリケーションの実装方法や実行環境で必要な設定などを含めて解説しています。 このマニュアルでは，旧バージョンのアプリケーションサーバで提供していた互換用の機能について説明しています。
14	HTTP Server	HTTP Server（Web サーバ）の構築，管理方法について説明しています。
15	Reliable Messaging	Reliable Messaging を使用した，メッセージの非同期通信によるアプリケーションの連携方法について説明しています。
16	TPBroker ユーザーズガイド	TPBroker の概要，機能，および運用方法について説明しています。
17	TPBroker 運用ガイド	TPBroker の機能のうち，ORB 機能のトラブルシュート，ORB 機能の運用に必要な拡張機能，TPBroker とほかの製品との連携方法，およびバージョンアップ時の移行について説明しています。
18	アプリケーションサーバ アプリケーション設定操作ガイド	サーバ管理コマンドを使用した J2EE アプリケーションおよびリソースの操作について説明しています。
19	アプリケーションサーバ 運用管理ポータル操作ガイド	運用管理ポータルの画面および操作について説明しています。
20	アプリケーションサーバ リファレンス コマンド編	システムを構築・運用するとき，またはアプリケーションを開発するときに使用するコマンドについて説明しています。
21	アプリケーションサーバ リファレンス 定義編(サーバ定義)	システムを構築・運用するとき，またはアプリケーションを開発するときに使用するファイルの定義方法について説明しています。 このマニュアルでは，J2EE サーバや Management Server などのサーバの定義に使用するファイルについて説明しています。
22	アプリケーションサーバ リファレンス 定義編(アプリケーション／リソース定義)	システムを構築・運用するとき，またはアプリケーションを開発するときに使用するファイルの定義方法について説明しています。 このマニュアルでは，アプリケーションやリソースの属性設定に使用するファイルについて説明しています。
23	アプリケーションサーバ メッセージ(構築／運用／開発用)	システムを構築・運用するとき，またはアプリケーションを開発するときに出力されるメッセージについて説明します。
24	アプリケーションサーバ メッセージ(監査者用)	システムを監査するときに使用するメッセージについて説明します。
25	アプリケーションサーバ アプリケーション開発ガイド	アプリケーションの開発方法について説明しています。また，開発環境のセットアップ方法についても説明しています。
26	アプリケーションサーバ リファレンス API 編	アプリケーションの開発で使用する API およびタグについて説明しています。
27	XML Processor ユーザーズガイド	XML Processor が提供する XML パーサ・XSLT トランスフォーマの機能，作成方法，および使用方法について説明しています。

項番	マニュアル名	内容
28	アプリケーションサーバ Web サービス開発ガイド	JAX-WS 仕様に従った SOAP Web サービスを開発・実行する方法と，V9 互換モードで JAX-RS 1.1 仕様に従った RESTful Web サービスを実行する方法について説明しています。
29	アプリケーションサーバ SOAP アプリケーション開発の手引	SOAP アプリケーション開発支援機能を使用して SOAP アプリケーションを開発，実行する方法について説明しています。
30	アプリケーションサーバ Web サービスセキュリティ 構築ガイド	Web サービスセキュリティ機能について説明しています。
31	XML Security - Core ユーザーズガイド	XML 署名データの生成・検証機能，および XML 暗号化・復号化機能について説明しています。
32	Borland(R) Enterprise Server VisiBroker(R) デベロッパーズガイド	Borland Enterprise Server VisiBroker の基本的な使用方法および高度な機能の取り扱い方法について説明しています。
33	Borland(R) Enterprise Server VisiBroker(R) プログラマーズリファレンス	Borland Enterprise Server VisiBroker が提供しているクラスとインタフェースの情報，プログラマツール，およびコマンドラインオプションについて説明しています。
34	アプリケーションサーバ & BPM/ESB 基盤 用語解説	アプリケーションサーバと BPM/ESB 基盤のマニュアル内で使用する用語について説明しています。

3.1.2 BPM/ESB 基盤のマニュアル体系

BPM/ESB 基盤のマニュアル体系を次の図に示します。

図 3-3　BPM/ESB 基盤のマニュアル体系



それぞれのマニュアルで説明している内容の概要を次の表に示します。なお，表の項番は図 3-3 の項番と対応しています。

表 3-2　BPM/ESB 基盤のマニュアル概要

項番	マニュアル名	内容
1	アプリケーションサーバ & BPM/ESB 基盤 概説	SOA を適用したシステムを実現するための機能の概要，製品の構成，および動作環境について説明しています。このマニュアルです。
2	サービスプラットフォーム ファーストステップガイド	SOA に対応したサービス統合環境（サービスプラットフォーム）の開発方法を理解するために参照します。 環境構築方法，動作・確認方法，開発方法について，サービスプラットフォームで提供しているサンプルプログラムを通して体験したい場合に参照してください。
3	サービスプラットフォーム 解説	SOA に対応したサービス統合環境（サービスプラットフォーム）で使用できる機能について解説しています。 システムの動作を円滑にする機能や性能向上のための機能について知りたい場合もこのマニュアルを参照してください。
4	サービスプラットフォーム 開発ガイド 基本開発編	SOA に対応したサービス統合環境（サービスプラットフォーム）を開発するために参照します。

項番	マニュアル名	内容
		開発環境の構築方法，XML スキーマ，ビジネスプロセス定義，サービスコンポーネントの定義，データ変換定義，およびサービスリクエスタなど一連の開発方法を知りたい場合に参照してください。
5	サービスプラットフォーム 開発ガイド 受付・アダプタ定義編	SOA に対応したサービス統合環境（サービスプラットフォーム）の実運用に必要なサービスコンポーネントのうち，受付とアダプタの開発方法を知りたい場合に参照してください。
6	サービスプラットフォーム システム構築・運用ガイド	SOA に対応したサービス統合環境（サービスプラットフォーム）を実際に構築・運用するために参照します。 開発から実運用までの流れ，運用環境・実行環境の構築方法，システムの運用方法，トラブル発生時の対処方法などを知りたい場合に参照してください。
7	サービスプラットフォーム リファレンス	SOA に対応したサービス統合環境（サービスプラットフォーム）で使用する画面，コマンド，および定義ファイルについて説明しています。 画面の内容，コマンドの文法，および定義ファイルの文法を知りたい場合に参照してください。
8	サービスプラットフォーム メッセージ	SOA に対応したサービス統合環境（サービスプラットフォーム）で出力されるメッセージについて説明しています。 メッセージの要因および対処方法を知りたい場合に参照してください。
9	アプリケーションサーバ & BPM/ESB 基盤 用語解説	アプリケーションサーバと BPM/ESB 基盤のマニュアル内で使用する用語について説明しています。

3.2　アプリケーションサーバのマニュアルの読書手順

ここでは，アプリケーションサーバのマニュアルの主な読書手順について説明します。

ここで説明するのは，次の目的に応じた読書手順です。

- アプリケーションサーバの動作を検証して，評価したい
- 本番環境にアプリケーションサーバを導入してシステムを構築・運用したい
- アプリケーションサーバで動作するアプリケーションを開発したい
- 発生したトラブルに対処したい
- アプリケーションサーバの機能を確認したい

> **参考**
>
> この節では，主に参照するマニュアルの読書手順を示しています。アプリケーションサーバのマニュアルをすべて網羅しているものではありません。ここで紹介しているマニュアル内での参照指示などを基に，必要に応じてここで示す以外のマニュアルも参照してください。

なお，次の内容を理解されていることを前提としています。

- Windows またはご使用の UNIX のシステム構築および運用に関する知識
- Java EE に関する知識
- SQL およびリレーショナルデータベースに関する基本的な知識
- CORBA に関する基本的な知識
- Web サーバに関する知識
- XML，XML Schema および Web サービス（セキュリティ含む）に関する基本的な知識

上記知識の詳細は，「4.6.2　アプリケーションサーバが対応する標準仕様」を参照してください。

3.2.1　アプリケーションサーバの動作を検証して，評価したい

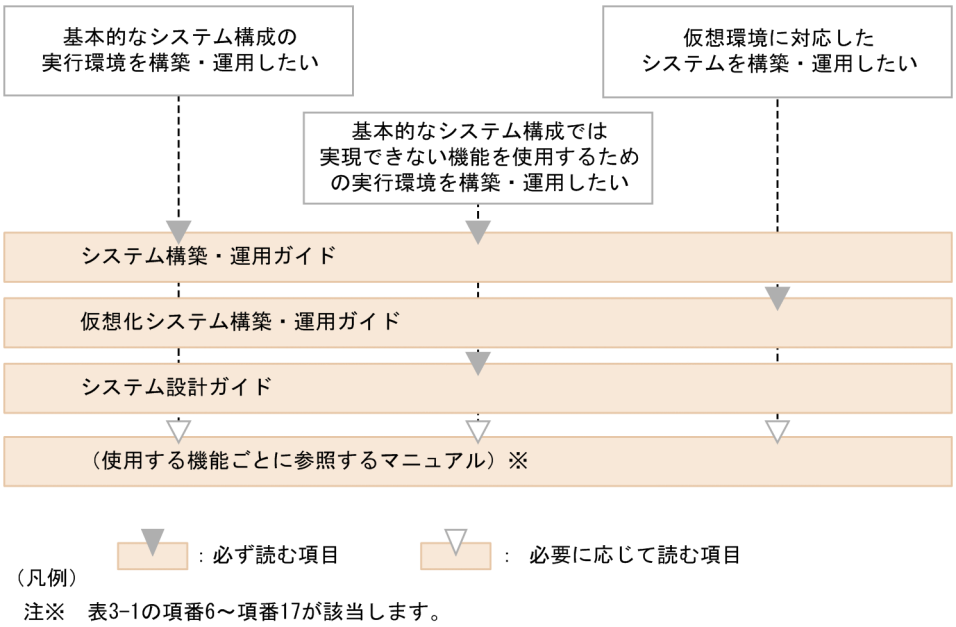
アプリケーションサーバの動作を検証，評価する場合，マニュアル「アプリケーションサーバ ファーストステップガイド」の手順に従って作業を進めることで，開発環境および実行環境を素早く構築して，サンプルプログラムを動かすことができます。

なお，このマニュアルで説明している内容は，テスト環境などでサンプルプログラムを動作させるための手順です。実際に本番環境として使用する環境を構築する手順ではありません。本番環境の構築について参照したい場合は，マニュアル「アプリケーションサーバ システム構築・運用ガイド」を参照してください。

3.2.2　本番環境にアプリケーションサーバを導入してシステムを構築・運用したい

本番環境にアプリケーションサーバを導入してシステムを構築・運用したい場合に参照するマニュアルを次の図に示します。

図 3-4　本番環境にアプリケーションサーバを導入してシステムを構築・運用したい場合の読書手順



注※　表3-1の項番6～項番17が該当します。

本番環境にアプリケーションサーバを導入する場合，仮想環境に対応するシステムを構築するかどうかで参照するマニュアルが異なります。仮想環境ではなく，物理環境に対応するシステムを構築する場合は，まず，マニュアル「アプリケーションサーバ システム構築・運用ガイド」を参照してください。このマニュアルでは，アプリケーションサーバを導入する場合の一般的なシステム構成に対応したシステムの構築・運用手順を示しています。

ただし，一部のシステム構成については，マニュアル「アプリケーションサーバ システム構築・運用ガイド」で示した内容だけでは構築・運用できない場合があります。マニュアル「アプリケーションサーバ システム構築・運用ガイド」の内容を参照した上で，必要に応じてマニュアル「アプリケーションサーバ システム設計ガイド」を参照してください。

また，仮想環境に対応したシステムを構築する場合は，マニュアル「アプリケーションサーバ 仮想化システム構築・運用ガイド」を参照してください。

なお，どの方法でシステムを構築・運用した場合も，使用する機能によっては個別の設定が必要な場合があります。個別の設定については，使用する機能について説明しているマニュアルを参照する必要があります。

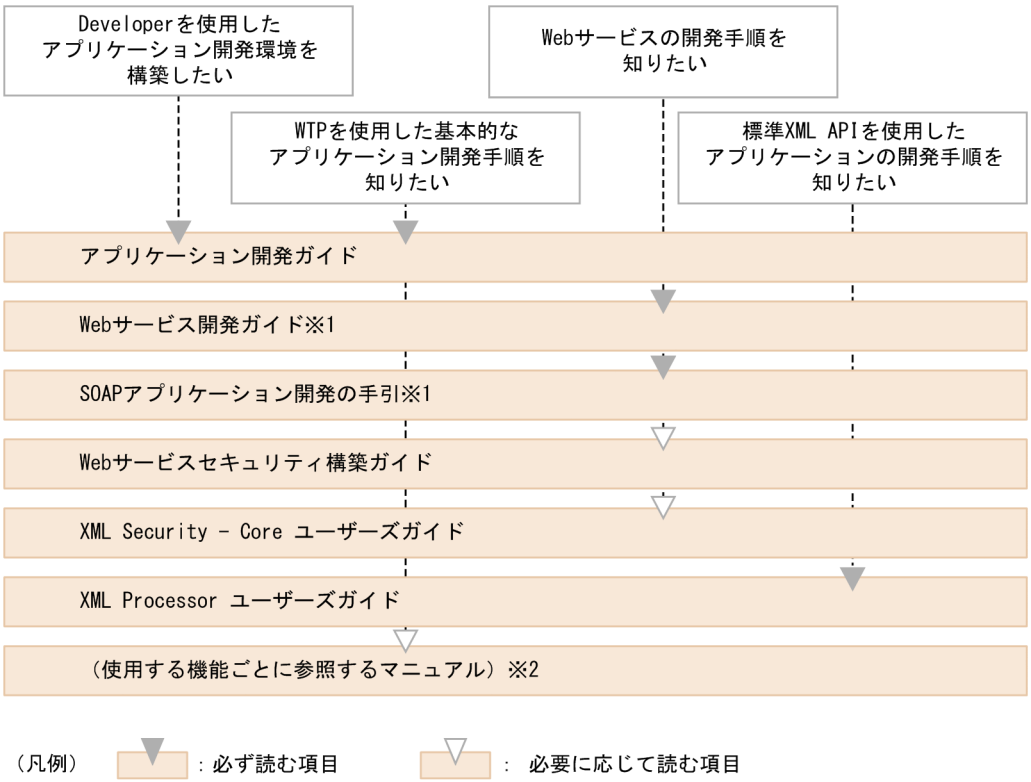
システムの目的と使用する機能，および対応するマニュアルについては，「5.　目的ごとに使用できるアプリケーションサーバの機能の紹介」を参照してください。次のような目的に対応する機能と，その機能を使用する場合のマニュアルの読み方について説明しています。

- システムの性能向上を図りたい
- システムの可用性を高めたい
- システムの信頼性を高めたい
- システムを効率良く運用したい

3.2.3　アプリケーションサーバで動作するアプリケーションを開発したい

アプリケーションサーバで動作するアプリケーションを開発したい場合に参照するマニュアルを次の図に示します。

図 3-5　アプリケーションサーバで動作するアプリケーションを開発したい場合の読書手順



注※1　準拠する仕様に応じてどちらかを選択してお読みください。

注※2　表3-1の項番6～項番17が該当します。

Developer では，WTP を使用してアプリケーションを開発するための環境を構築するための機能を提供しています。WTP を使用してアプリケーションを開発するための環境の構築手順については，マニュアル「アプリケーションサーバ アプリケーション開発ガイド」を参照してください。また，WTP を使用する場合の開発手順についても，このマニュアルを参照してください。

アプリケーションを Web サービスとして開発する場合の開発手順については，マニュアル「アプリケーションサーバ Web サービス開発ガイド」またはマニュアル「アプリケーションサーバ SOAP アプリケーション開発の手引」を参照してください。なお，Web サービス開発の中でセキュリティに関する機能を使

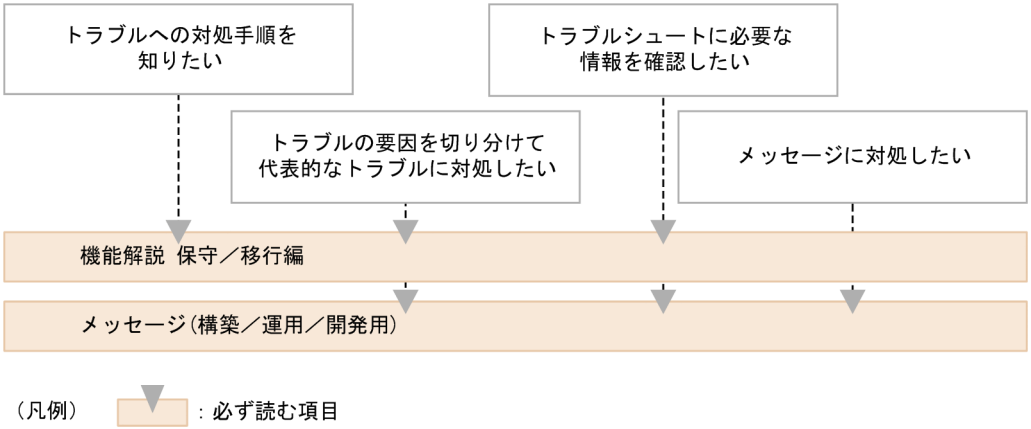
用する場合は，必要に応じてマニュアル「アプリケーションサーバ Web サービスセキュリティ構築ガイド」およびマニュアル「XML Security - Core ユーザーズガイド」も参照してください。

また，標準 XML API を使用したアプリケーションの開発手順について知りたい場合は，マニュアル「XML Processor ユーザーズガイド」を参照してください。

3.2.4　発生したトラブルに対処したい

アプリケーションサーバが動作するシステムにトラブルが発生した場合に参照するマニュアルを次の図に示します。

図 3-6　アプリケーションサーバでトラブルが発生した場合の読書手順



システムにトラブルが発生した場合は，まず，マニュアル「アプリケーションサーバ 機能解説 保守／移行編」を参照してください。トラブルへの対処手順や，代表的なトラブルへの対処方法，トラブルシュートに必要な情報などについて説明しています。

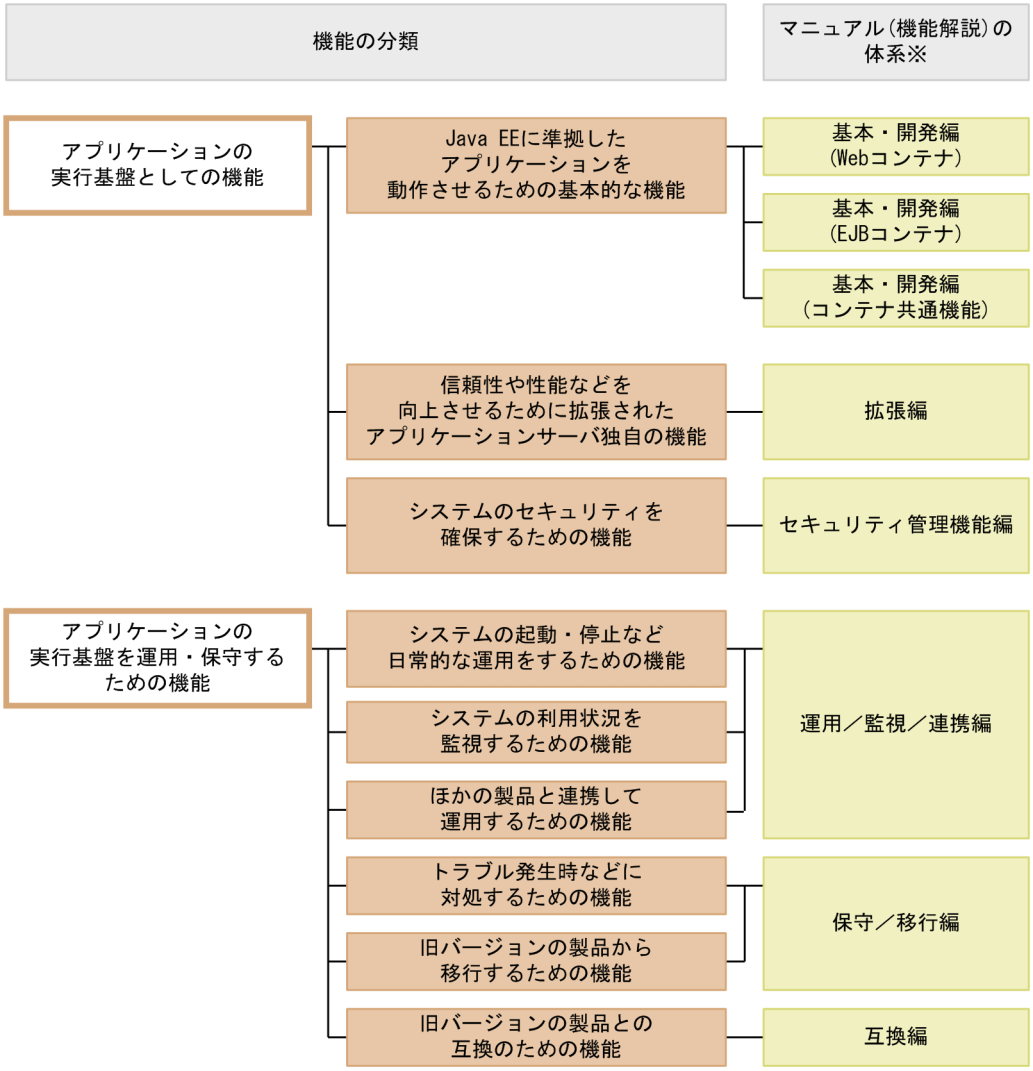
また，出力されたメッセージに直接対処する場合は，出力されたメッセージのプリフィックス，ID，メッセージテキストなどを基に，マニュアル「アプリケーションサーバ メッセージ(構築／運用／開発用)」を参照してください。

3.2.5　アプリケーションサーバの機能を確認したい

アプリケーションサーバでは，Java EE の標準仕様に対応した機能のほか，独自に拡張した機能を提供しています。

アプリケーションサーバのマニュアルでは，アプリケーションサーバの機能を次の表に示すように分類して，8 冊のマニュアルで説明しています。なお，これらのマニュアルは，使用する機能ごとに必要に応じて参照するマニュアルです。読書手順はありません。

図 3-7　アプリケーションサーバの機能の分類



注※　マニュアル名称の「アプリケーションサーバ 機能解説」を省略しています。

各分類に含まれる具体的な機能については，図 3-7 に示した各マニュアルの 1 章の説明を参照してください。

また，ここで示した機能のほか，次の機能については別のマニュアルで説明しています。

表 3-3　図 3-7 以外のマニュアルで説明している機能

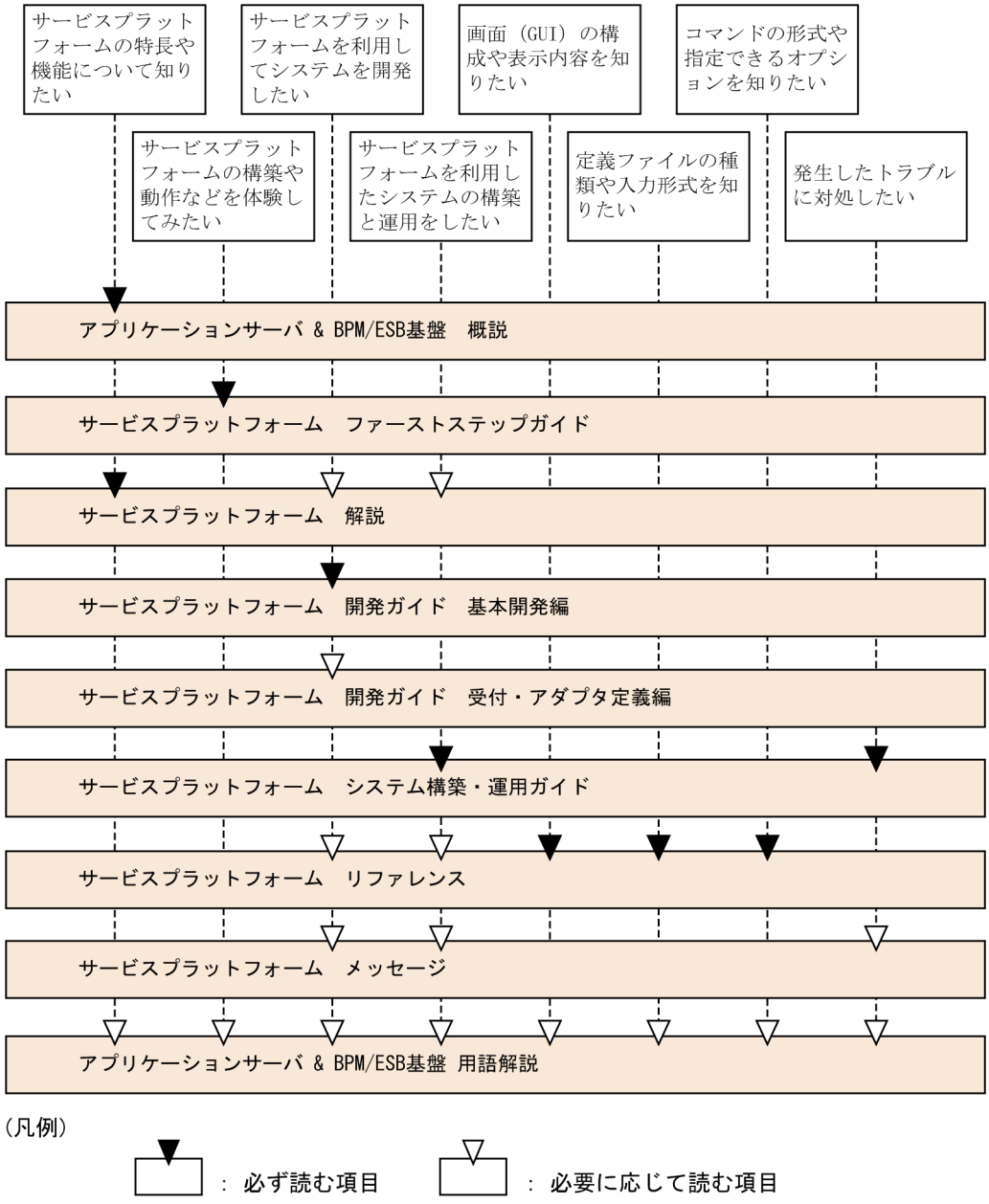
項番	機能	対応するマニュアル
1	アプリケーションサーバが提供する Web サーバ（HTTP Server）の機能	HTTP Server
2	標準 XML API を使用して XML 文書を操作するプログラムを開発するための機能	XML Processor ユーザーズガイド
3	JAX-WS 仕様に従った SOAP Web サービスを開発・実行するための機能と，V9 互換モードで JAX-RS 1.1 仕様に従った RESTful Web サービスを実行するための機能	アプリケーションサーバ Web サービス開発ガイド

項番	機能	対応するマニュアル
4	SOAP アプリケーションを開発，実行するための機能	アプリケーションサーバ SOAP アプリケーション開発の手引
5	SOAP Web サービスのセキュリティを確保するための機能	アプリケーションサーバ Web サービスセキュリティ構築ガイド XML Security - Core ユーザーズガイド
6	アプリケーションサーバが提供するリソースアダプタである CRM を使用してメッセージの非同期通信を実現するための機能	Reliable Messaging
7	TPBroker および VisiBroker の機能	TPBroker ユーザーズガイド TPBroker 運用ガイド Borland(R) Enterprise Server VisiBroker(R) デベロッパーズガイド Borland(R) Enterprise Server VisiBroker(R) プログラマーズリファレンス

アプリケーションサーバで使用できる機能については，「4.6.2(1)　Java EE の標準仕様」を参照してください。

3.3 BPM/ESB 基盤のマニュアルの読書手順

マニュアル体系図で示した BPM/ESB 基盤のマニュアルは，次の案内に従ってお読みいただくことをお勧めします。



ここでは，BPM/ESB 基盤のマニュアルの主な読書手順について説明します。

ここで説明するのは，次の目的に応じた読書手順です。

- サービスプラットフォームの動作を検証して，評価したい
- サービスプラットフォームの機能を確認したい
- SOA に対応したサービス統合環境（サービスプラットフォーム）を開発したい

- SOA に対応したサービス統合環境（サービスプラットフォーム）を構築して運用したい
- SOA に対応したサービス統合環境（サービスプラットフォーム）で発生したトラブルに対処したい

> **参考**
>
> この節では，主に参照するマニュアルの読書手順を示しています。ここで紹介しているマニュアル内での参照指示などを基に，必要に応じてここで示す以外のマニュアルも参照してください。

なお，次の内容を理解されていることを前提としています。

- Windows またはご使用の UNIX のシステム構築および運用に関する知識
- SOA に関する基本的な知識
- Java EE に関する知識
- SQL およびリレーショナルデータベースに関する基本的な知識
- XML に関する基本的な知識

3.3.1　サービスプラットフォームの動作を検証して，評価したい

実際にマシンを操作しながら，サービスプラットフォームの動作を検証，評価する場合，マニュアル「サービスプラットフォーム ファーストステップガイド」の手順に従って作業を進めることで，環境構築からサンプルプログラムを実行するまでの操作を体験できます。

なお，このマニュアルで説明している内容は，開発環境で構築したテスト環境でサンプルプログラムを動作させるための手順です。実際に運用環境として使用する実行環境を構築する手順ではありません。運用環境の構築について参照したい場合は，マニュアル「サービスプラットフォーム システム構築・運用ガイド」を参照してください。

3.3.2　サービスプラットフォームの機能を確認したい

BPM/ESB 基盤では，サービスプラットフォームで SOA を適用したシステムを実現するための機能を提供しています。

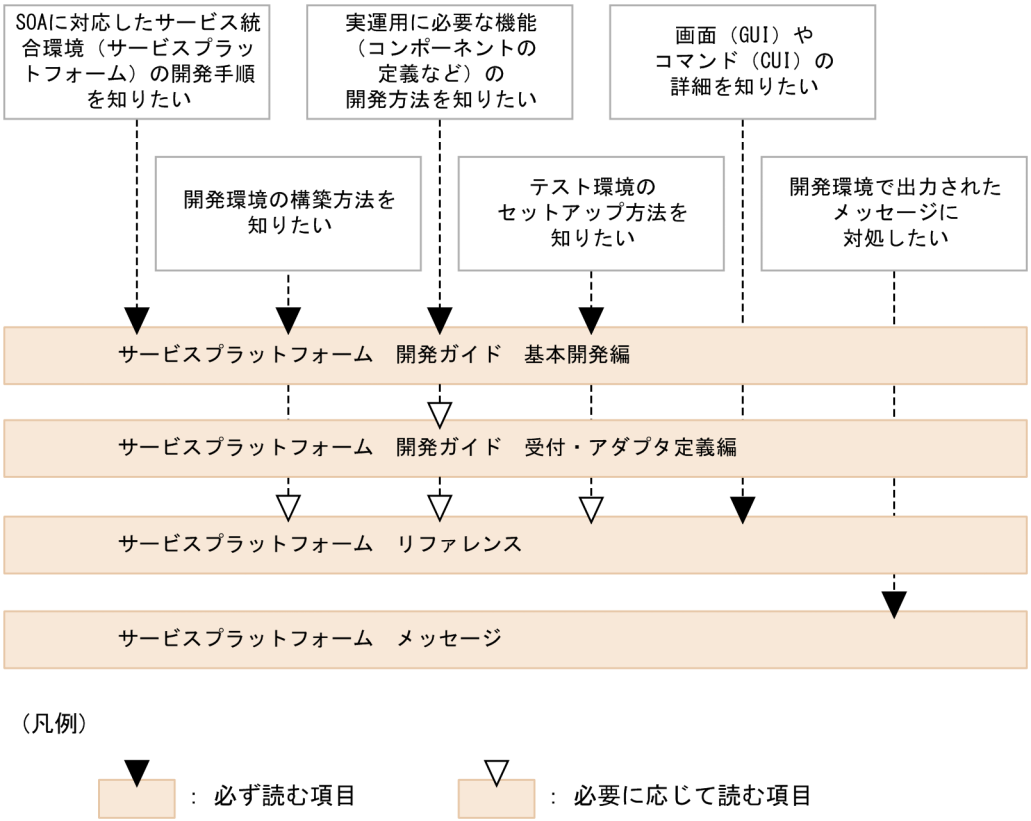
サービスプラットフォームでは，開発環境，実行環境および運用環境が相互に関連してシステム全体を構成します。

それぞれの環境で提供している機能を確認したい場合，マニュアル「サービスプラットフォーム 解説」を参照してください。なお，このマニュアルは，使用する機能ごとに，必要に応じて参照するマニュアルです。読書手順はありません。

3.3.3 SOA に対応したサービス統合環境（サービスプラットフォーム）を開発したい

SOA に対応したサービス統合環境（サービスプラットフォーム）を開発したい場合に参照するマニュアルを次の図に示します。

図 3-8 SOA に対応したサービス統合環境（サービスプラットフォーム）を開発したい場合の読書手順



XML スキーマ，ビジネスプロセス定義，サービスコンポーネントの定義，データ変換定義，およびサービスリクエスタなど一連の開発方法を知りたい場合，マニュアル「サービスプラットフォーム 開発ガイド 基本開発編」を参照してください。また，開発環境の構築，およびテスト環境のために実施するインストールと簡易セットアップについても，このマニュアルを参照してください。

なお，サービスコンポーネントのうち，受付とアダプタの定義については，マニュアル「サービスプラットフォーム 開発ガイド 受付・アダプタ定義編」を参照してください。

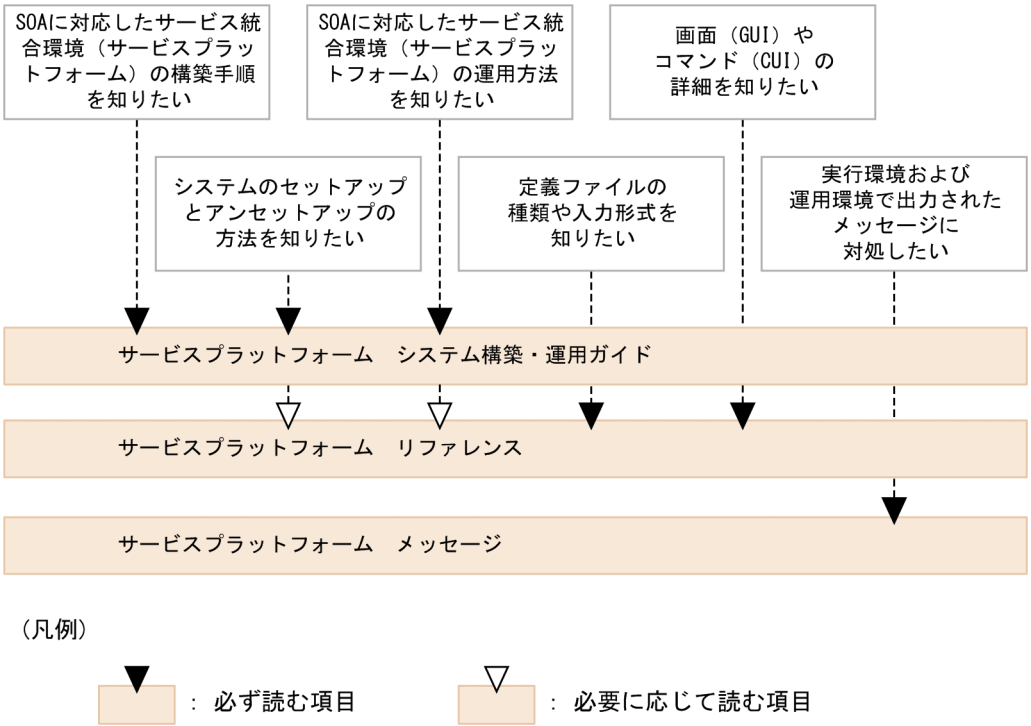
開発環境で使用する画面やコマンド，開発に必要な定義ファイルの詳細については，マニュアル「サービスプラットフォーム リファレンス」を参照してください。

また，開発環境に出力されたメッセージに対処する場合，マニュアル「サービスプラットフォーム メッセージ」を参照してください。

3.3.4 SOA に対応したサービス統合環境（サービスプラットフォーム）を構築して運用したい

SOA に対応したサービス統合環境（サービスプラットフォーム）を構築して運用したい場合に参照するマニュアルを次の図に示します。

図 3-9 SOA に対応したサービス統合環境（サービスプラットフォーム）を構築して運用したい場合の読書手順



BPM/ESB 基盤を導入して SOA に対応したサービス統合環境（サービスプラットフォーム）を構築する場合，マニュアル「サービスプラットフォーム システム構築・運用ガイド」を参照してください。

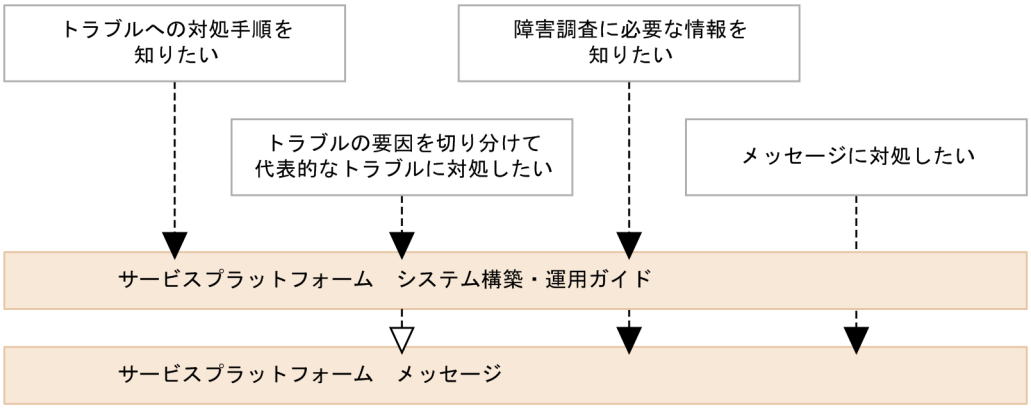
ただし，一部のシステムの構築については，「サービスプラットフォーム システム構築・運用ガイド」で示した内容だけでは構築できない場合があります。必要に応じてマニュアル「アプリケーションサーバ システム構築・運用ガイド」の内容を参照してください。

システムを構築したあとの運用についても，マニュアル「サービスプラットフォーム システム構築・運用ガイド」を参照してください。

3.3.5 SOA に対応したサービス統合環境（サービスプラットフォーム）で発生したトラブルに対処したい

SOA に対応したサービス統合環境（サービスプラットフォーム）でトラブルが発生した場合に参照するマニュアルを次の図に示します。

サービスプラットフォームでトラブルが発生した場合の読書手順



（凡例）

：必ず読む項目　　：必要に応じて読む項目

システムにトラブルが発生した場合は，まず，マニュアル「サービスプラットフォーム システム構築・運用ガイド」を参照してください。トラブルへの対処手順や，代表的なトラブルへの対処方法，トラブルシュートに必要な情報などについて説明しています。

また，出力されたメッセージに直接対処する場合は，出力されたメッセージのプリフィックス，ID，メッセージテキストなどを基に，マニュアル「サービスプラットフォーム メッセージ」を参照してください。

3.4　製品とマニュアルの対応

ここでは，アプリケーションサーバおよび BPM/ESB 基盤の製品と各マニュアルの対応について示します。

表 3-4　製品とマニュアルの対応

項番	マニュアル名称	製品名称				
		Application Server	Developer	Service Platform	Service Architect	Client
1	アプリケーションサーバ ファーストステップガイド	○	○	○	○	○
2	アプリケーションサーバ & BPM/ESB 基盤 概説	○	○	○	○	○
3	アプリケーションサーバ システム構築・運用ガイド	○	○	○	○	○
4	アプリケーションサーバ 仮想化システム構築・運用ガイド	○	－	○	－	－
5	アプリケーションサーバ システム設計ガイド	○	○	○	○	○
6	アプリケーションサーバ 機能解説 基本・開発編(Web コンテナ)	○	○	○	○	○
7	アプリケーションサーバ 機能解説 基本・開発編(EJB コンテナ)	○	○	○	○	○
8	アプリケーションサーバ 機能解説 基本・開発編(コンテナ共通機能)	○	○	○	○	○
9	アプリケーションサーバ 機能解説 拡張編	○	○	○	○	○
10	アプリケーションサーバ 機能解説 セキュリティ管理機能編	○	○	○	○	○
11	アプリケーションサーバ 機能解説 運用／監視／連携編	○	○	○	○	○
12	アプリケーションサーバ 機能解説 保守／移行編	○	○	○	○	○
13	アプリケーションサーバ 機能解説 互換編	○	○	○	○	○
14	アプリケーションサーバ アプリケーション設定操作ガイド	○	○	○	○	○
15	アプリケーションサーバ 運用管理ポータル操作ガイド	○	○	○	○	○
16	アプリケーションサーバ リファレンス コマンド編	○	○	○	○	○
17	アプリケーションサーバ リファレンス 定義編(サーバ定義)	○	○	○	○	○
18	アプリケーションサーバ リファレンス 定義編(アプリケーション／リソース定義)	○	○	○	○	○

項番	マニュアル名称	製品名称				
		Application Server	Developer	Service Platform	Service Architect	Client
19	HTTP Server	○	○	○	○	−
20	Reliable Messaging	○	○	○	○	−
21	アプリケーションサーバ アプリケーション開発ガイド	−	○	−	○	−
22	アプリケーションサーバ リファレンス API 編	○	○	○	○	○
23	XML Processor ユーザーズガイド	−	○	○	○	−
24	アプリケーションサーバ Web サービス開発ガイド	○	○	○	○	○
25	アプリケーションサーバ Web サービスセキュリティ構築ガイド	○	○	○	○	−
26	アプリケーションサーバ SOAP アプリケーション開発の手引	○	○	○	○	○
27	アプリケーションサーバ XML Security - Core ユーザーズガイド	○	○	○	○	−
28	アプリケーションサーバ メッセージ(構築／運用／開発用)	○	○	○	○	○
29	アプリケーションサーバ メッセージ(監査者用)	○	○	○	○	○
30	TPBroker ユーザーズガイド	○	○	○	○	−
31	TPBroker 運用ガイド	○	○	○	○	○
32	Borland(R) Enterprise Server VisiBroker(R) デベロッパーズガイド	○	○	○	○	−
33	Borland(R) Enterprise Server VisiBroker(R) プログラマーズリファレンス	○	○	○	○	○
34	サービスプラットフォーム ファーストステップガイド	−	−	○	○	−
35	サービスプラットフォーム 解説	−	−	○	○	−
36	サービスプラットフォーム 開発ガイド 基本開発編	−	−	○	○	−
37	サービスプラットフォーム 開発ガイド 受付・アダプタ定義編	−	−	○	○	−
38	サービスプラットフォーム システム構築・運用ガイド	−	−	○	○	−
39	サービスプラットフォーム リファレンス	−	−	○	○	−
40	サービスプラットフォーム メッセージ	−	−	○	○	−

項番	マニュアル名称	製品名称				
		Application Server	Developer	Service Platform	Service Architect	Client
41	アプリケーションサーバ & BPM/ESB 基盤 用語解説	○	○	○	○	○

（凡例）○：対応する。 －：対応しない。

注　ご利用の製品によっては，項番 1～33 のマニュアルで使用している用語をご利用の製品名に読み替える必要があります。次の表に従って，マニュアルで使用している用語をご利用の製品名に読み替えてください。なお，Developer または Service Architect の場合，テスト環境として使用している場合だけ読み替えが必要です。

ご利用の製品名	マニュアルで使用している用語
Developer	Application Server
Service Architect	
Service Platform	

4 アプリケーションサーバの概要

この章では，アプリケーションサーバの概要，および環境ごとの特長について説明します。

アプリケーションサーバは，業務システムの中核に位置し，アプリケーションを実行する基盤となる製品です。標準技術である Java EE に準拠した実行環境を構築・運用できます。

さらに，実行環境上で実行するアプリケーションを効率良く開発する環境も構築できます。

4.1　アプリケーション実行環境とアプリケーション開発環境

この節では，アプリケーションサーバの特長について説明します。

アプリケーションサーバは，サービスを実行するための基盤（**サービスデプロイメント**）となる製品です。性能と信頼性を確保しながらユーザにサービスを提供する，アプリケーション実行環境を構築します。また，サービスとなるアプリケーションを開発する環境も構築できます。

アプリケーションの実行環境とアプリケーションの開発環境の特長を次に示します。

アプリケーション実行環境

アプリケーションを実行することでユーザにサービスを提供する基盤となる環境です。

アプリケーション実行環境では，次の 2 種類の業務を実行できます。

- **オンライン業務（オンライン処理）**

  インターネットやイントラネット上のユーザから送信された要求を随時処理する形式の業務です。オンライン業務では，Java EE の技術を使用して開発されたアプリケーションを実行します。この環境を，**J2EE アプリケーション実行環境**といいます。

- **バッチ業務（バッチ処理）**

  定型的な業務を決まった時間にまとめて処理する形式の業務です。従来メインフレーム上などで実行されていたバッチジョブを，オープン環境の技術である Java を使用して実現できます。Java で開発したバッチジョブを実行するためのアプリケーションをバッチアプリケーションといいます。バッチアプリケーションを実行するための環境を**バッチアプリケーション実行環境**といいます。

アプリケーション開発環境

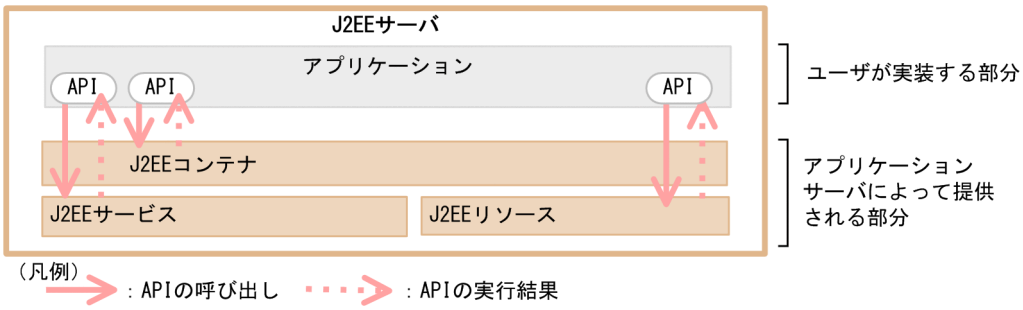
アプリケーション実行環境上で動作するアプリケーションを開発する環境です。アプリケーションサーバでは，実行環境上で動作するアプリケーションの開発からデバッグまでを統括的に支援する開発環境を構築・運用できます。

アプリケーションサーバは，標準仕様である Java EE に準拠したアプリケーションの実行環境を構築します。Java EE に準拠したアプリケーションを実行する機能を持つサーバプロセスを，**J2EE サーバ**といいます。

J2EE サーバは，J2EE コンテナ，J2EE サービス，J2EE リソースなどの Java EE で規定された仕様に従って，ユーザが開発した J2EE アプリケーションを実行するために必要な機能を提供します。例えば，J2EE サーバでは，トランザクション管理やセキュリティ管理など，複数の業務に共通する処理を実行する機能を，J2EE サービスや J2EE リソースによって提供しています。アプリケーション開発時には，アプリケーション内で J2EE コンテナ，J2EE サービス，J2EE リソースなどで提供されている API を呼び出すことで，煩雑なコーディングをしないで，複数の業務で共通の処理を実現できます。

アプリケーションと J2EE サーバの関係を次の図に示します。

図 4-1　アプリケーションと J2EE サーバの関係



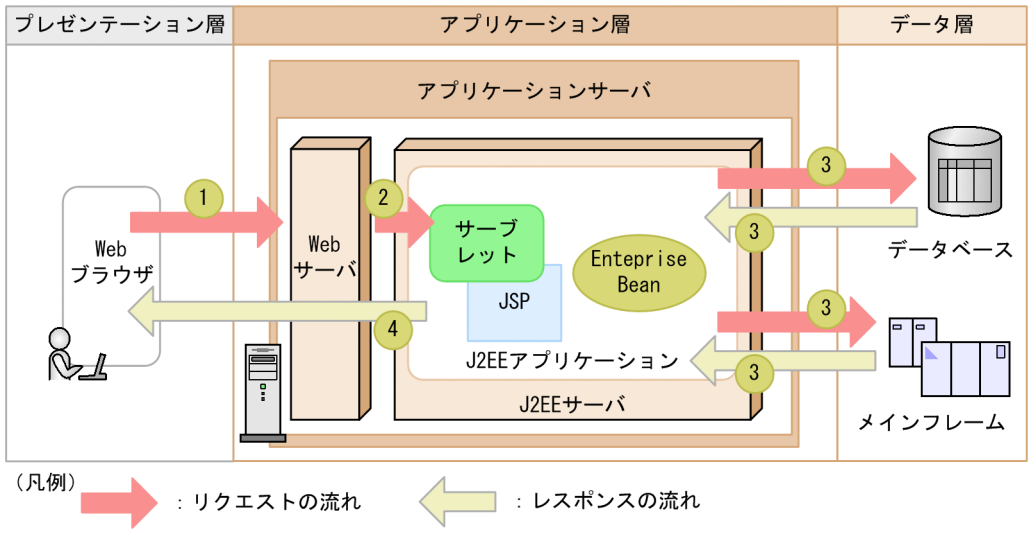
アプリケーションサーバは，J2EE サーバを中心とした，アプリケーションの実行環境になるサーバ基盤です。情報システムの中間に位置し，ユーザの要求とデータベースなど，業務システム間の処理の受け渡しをします。

アプリケーションサーバ上では，ユーザの要求に応じたサービスを提供するための J2EE アプリケーションが実行されます。J2EE アプリケーションとは，J2EE サーバ上で動作する，実現したい業務内容に応じて開発されたアプリケーションです。

J2EE アプリケーションは，ユーザからの要求を受け付け，処理を実行し，結果をユーザに返します。また，処理内容によっては，データベースやメインフレームなどのほかのシステムとデータをやり取りして，必要な情報を取得して処理を実行します。

アプリケーションサーバを中心としたシステムでのリクエスト処理の流れを次の図に示します。

図 4-2　アプリケーションサーバを中心としたシステムでのリクエスト処理の流れ



図で示した流れについて説明します。

1. ユーザが Web ブラウザ上で処理を実行すると，Web ブラウザからアプリケーションサーバに対してリクエストが送信されます。Web ブラウザからのリクエストは，アプリケーションサーバの一部である Web サーバが受け付けます。

2. アプリケーションサーバ内で，ユーザのリクエストに対応する J2EE アプリケーションが実行されます。J2EE アプリケーションは，Web サーバから転送されたリクエストを受け付けるためのプログラム（サーブレット，JSP）や，業務処理を実行するプログラム（Enterprise Bean など）で構成されています。
3. J2EE アプリケーションは，必要に応じて，ユーザのリクエストを処理するためにデータベースやほかのシステムにアクセスします。
4. 業務処理が完了したら，Web サーバ経由で Web ブラウザにレスポンスが送信されます。Web ブラウザ上で表示するための画面は，J2EE アプリケーション内のサーブレットや JSP で生成されます。生成された内容が，処理結果としてユーザが操作している Web ブラウザ上に表示されます。

アプリケーションサーバでは，安定稼働性や耐障害性が高く，優れたパフォーマンスを実現するシステムを構築できます。また，作業を円滑に実行するための機能によって，効率の良いシステム構築・システム運用を実現できます。

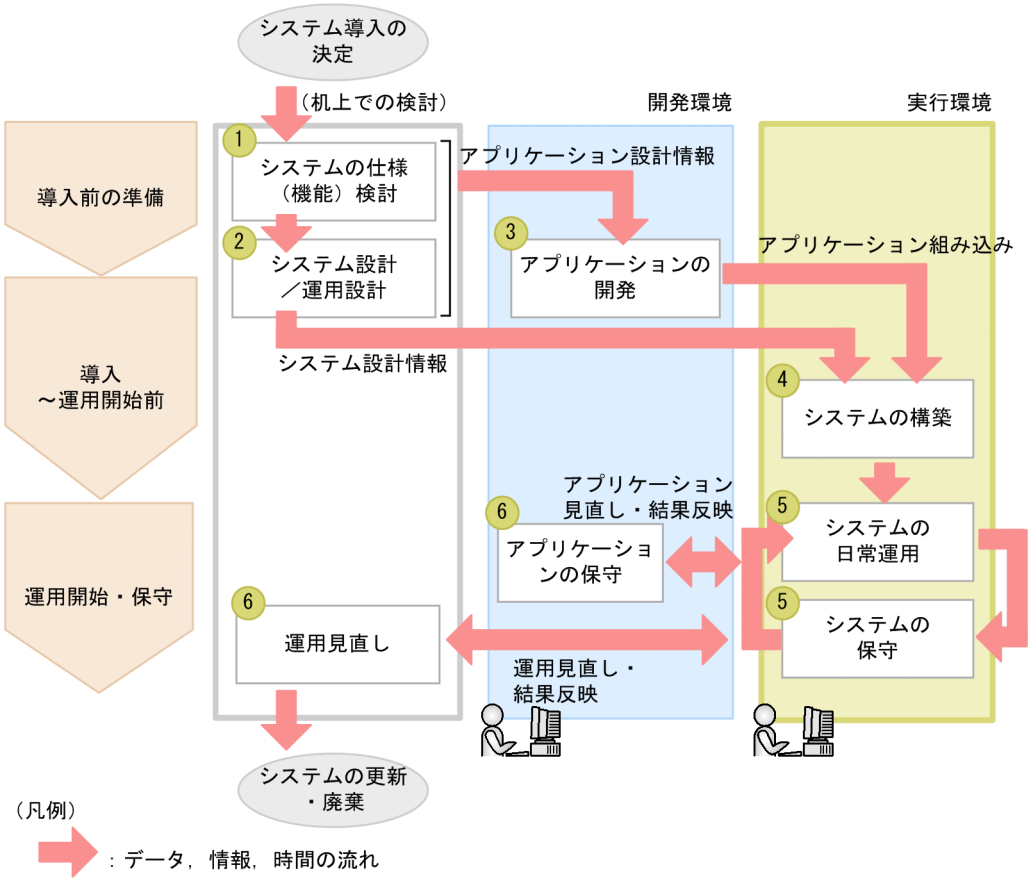
4.2　アプリケーションサーバで構築するシステムのライフサイクル

アプリケーションサーバで構築するシステムのライフサイクルには，次の段階があります。

1. システムの仕様（機能）検討
2. システム設計／運用設計
3. アプリケーションの開発
4. システムの構築
5. システムの日常運用と保守
6. 運用見直しとアプリケーションの保守

アプリケーションの開発環境および実行環境でのシステムのライフサイクルを図に表すと，次のようになります。

図 4-3　システムのライフサイクル



それぞれの段階について説明します。

1. システムの仕様（機能）検討

システムの目的や業務の内容に応じて，アプリケーションサーバのどの機能を使用するかを検討・決定します。

2. **システム設計／運用設計**
   システムの目的や実行する業務，およびシステムの使用方法に応じてシステムの構成と運用方法を設計します。

3. **アプリケーションの開発**
   1.および 2.で検討したアプリケーション設計情報に基づいて，業務に合ったアプリケーションを開発します。

4. **システムの構築**
   1.および 2.で検討したシステム設計情報に基づいて，システムを構築します。また，3.で開発したアプリケーションをシステムに組み込みます。

5. **システムの日常運用と保守**
   アプリケーションを実行して，システムの運用を開始します。システムの運用では，日常運用と定期的なシステム保守を繰り返します。

6. **運用見直し／アプリケーション保守**
   業務内容の変更やシステム規模の変更などに合わせて，運用方法を随時見直します。また，必要に応じてアプリケーションも保守します。

ライフサイクルは，システムの更新または廃棄を決定するまで継続します。

4.2.1　システムの仕様（機能）検討

システムの目的や業務の内容に応じて，実行環境でどの機能を使用するかを検討・決定します。

「3.2.5　アプリケーションサーバの機能を確認したい」で示したマニュアルを参照して，使用する機能を検討・決定してください。

4.2.2　システム設計と運用設計

アプリケーションサーバのシステム設計と運用設計では，次の点を検討します。

- **システム構成をどうするか／システムのスケーラビリティをどのように確保するか**
  まず，J2EE アプリケーションによってオンライン処理を実行するシステムか，バッチアプリケーションによってバッチ処理を実行するシステムかを明確にします。
  次に，システムの規模に応じて，Web サーバ，J2EE サーバ，バッチサーバおよび各プロセスを物理的にどのように配置するかを決めます。オンライン処理を実行するシステムの場合は，複数のサーバを使用して負荷を分散する必要があるかなどを検討します。スケーラビリティについてもよく検討しておく

必要があります。実際の日常運用が開始されたあとで，システム規模を変更したり，トラブル発生時の影響をできるだけ局所的に抑えたりするためには，システムのスケーラビリティが確保されていることが重要です。

- どのような運用のしかたをするか／可用性と信頼性をどのように確保するか

  業務システムを運用する場合には，システムの可用性と信頼性を確保しながら，できるだけ効率良く運用していく方法を検討します。アプリケーションサーバには，システム内の複数のサーバマシンを一括して管理，運用するための機能として，Management Server という運用管理機能があります。

  また，システム全体を効率良く運用する方法として，JP1 やクラスタソフトウェアと連携するかどうかも，あわせて検討してください。

- システムのセキュリティをどのように確保するか

  ミッション・クリティカルな業務システムの場合，信頼性の高いセキュリティを確保することは不可欠です。システム設計の段階で，セキュアなシステムを構築するための観点を明確にして，導入・運用時の手順を決めておく必要があります。また，外部ネットワークと接続するシステムを構築する場合は，ファイアウォール，侵入検知システムおよび SSL アクセラレータを適切に配置，設定してどのように信頼性の高いシステムを構築するかを，あわせて検討してください。

- 性能を向上させるためにどのようなチューニングが必要か

  システム要件によっては，厳密な性能設計が必要です。プールやキャッシュを効果的に利用したり，タイムアウトを適切に設定したりすることで，システム全体の性能向上が図れます。

- JavaVM をどのようにチューニングするか

  アプリケーションサーバで動作する J2EE サーバなどのプロセスは，JavaVM 上で実行されます。JavaVM で使用するメモリ空間を適切に管理することによって，FullGC の頻発を防ぎ，パフォーマンスの低下を防げます。

なお，標準的なパターンでアプリケーションサーバを構築・運用する場合は，まず，システムを構築・運用してから必要に応じて設定変更やチューニングを実施できます。システム設計の段階では，まず構築したいシステムが標準的なパターンに合致するかどうかについて，マニュアル「アプリケーションサーバ システム構築・運用ガイド」を参照して確認してください。

4.2.3 アプリケーションの開発

目的とする業務内容に合わせて，アプリケーションを開発します。

アプリケーションサーバでは，Java 言語を使用して開発した J2EE アプリケーションやバッチアプリケーションを実行できます。

一般的な J2EE アプリケーションは，MVC アーキテクチャに基づいて，コンポーネント化されたプログラム群によって構成されます。個々の業務処理プログラムは，EJB の仕様に従った Enterprise Bean として作成します。Enterprise Bean の組み合わせによって複雑なアプリケーションを開発したり，業務の変化に応じて Enterprise Bean を入れ替えたりすることで，再利用性の高い J2EE アプリケーションが開発できます。

アプリケーションサーバで構築したシステムで動作するアプリケーションは，次のような方法で開発できます。

- Developer の機能を使用して開発する

  Developer を使用する場合，Eclipse のプラグインである WTP を使用してアプリケーションを開発できます。

  WTP では，コーディングから，ビルド，デバッグまでの一連の開発作業を実行できます。構築したテスト環境上の J2EE サーバの起動・停止や，アプリケーションの開始・停止などの操作も実行できます。

- Developer および IDE を使用しないで開発する

  Developer や IDE を使用しなくても，アプリケーションは開発できます。この場合は，テキストエディタを使用してプログラムのソースを作成し，javac コマンドでコンパイルします。また，jar コマンドでアーカイブを作成して，アプリケーションを開発します。

4.2.4 システムの構築

システム設計の結果を基に，システムを構築します。

システム構築とは，アプリケーションサーバの実行環境を J2EE アプリケーションまたはバッチアプリケーションが実行できる状態にすることです。J2EE アプリケーションを実行するシステムを構築する場合は，製品のインストール，J2EE サーバの設定，Web サーバとの連携，J2EE リソースと J2EE アプリケーションの設定，運用環境の構築などが含まれます。バッチアプリケーションを実行するシステムを構築する場合は，製品のインストール，バッチサーバの設定，リソースの設定などが含まれます。

アプリケーションサーバでは，次の 3 種類の構築ツールを提供しています。

- セットアップウィザード

  対話型ウィザードで表示される画面に従って項目を選択または設定しながら，システムを構築できるツールです。J2EE アプリケーションの実行環境が構築できます。

- 運用管理ポータル

  Web ブラウザ上に表示した GUI 画面の画面項目を選択，またはテキストで値を入力することで，システムを構築できるツールです。J2EE アプリケーションまたはバッチアプリケーションの実行環境が構築できます。

- Smart Composer 機能

  XML 形式のファイルでパラメタを設定し，そのファイルを引数に設定してコマンドを実行することでシステムを構築できるツールです。J2EE アプリケーションまたはバッチアプリケーションの実行環境が構築できます。

セットアップウィザードを使用することで，細かな設定や定義ファイルの作成をしないで，本番環境で運用するシステムを構築できます。ただし，セットアップウィザードで構築できないシステムもあるため，システムを構築する際には，まず，目的のシステムがセットアップウィザードを使用できるかどうかを確

認してください。ツールの使い分けについては，マニュアル「アプリケーションサーバ システム構築・運用ガイド」を参照してください。

これらのツールを使用して構築したシステムは，Management Server というサーバプロセスの機能を使用して一括管理できます。

4.2.5 システムの運用と保守

アプリケーションの開発とシステムの構築ができたら，アプリケーションを実行して，運用を開始します。システムの運用では，日常の運用とシステム保守を繰り返します。

日常運用では，日常的なサーバの起動／停止のほか，システムを安定稼働させるために各種プロセスの監視，ログの収集，ユーザの管理などをします。また，さらに広範囲なシステムをまとめて統合運用したい場合は，JP1 を使用することもできます。

システムの保守では，スケールイン，スケールアウト，スケールアップ，スケールダウンなどのシステム規模の変更をしたり，発生したトラブルに対処（**トラブルシューティング**）したりします。

4.3　J2EE アプリケーションの実行環境の特長

この節では，J2EE アプリケーションの実行環境の特長について説明します。

4.3.1　標準仕様への対応

アプリケーションサーバは，Java EE 8 の標準仕様をサポートしています。

Java EE に準拠した実行環境では，複数の業務で共通に使用される機能を，Java EE の API を使用して実現できます。共通に使用される機能とは，例えば，データベースやメインフレームに接続する機能や，セッション管理機能，トランザクション管理機能などです。

これらの機能は，アプリケーションサーバに含まれるモジュールである，コンテナ，サービスなどの形式で，アプリケーションに提供されます。Java EE の API を使用することで，個々の J2EE アプリケーション内での煩雑なコーディングを減らすことができます。アプリケーションの動作に必要な属性をアノテーションで指定することもできます。

このほか，アプリケーションサーバで構築したシステムは，XML，Web サービス，SSL，分散オブジェクトなどに関連した標準仕様にも対応しています。アプリケーションサーバが対応する標準仕様については，「4.6.2　アプリケーションサーバが対応する標準仕様」を参照してください。

また，アプリケーションサーバでは，Jakarta EE 9 のアプリケーションを実行できます。詳細はマニュアル「アプリケーションサーバ 機能解説 基本・開発編(コンテナ共通機能)」の「20.　パッケージ名変換機能」を参照してください。

4.3.2　システムの安定稼働の実現

業務システムには，安定して稼働し続けることが求められます。業務の内容によっては，一定時間にアクセスが急増したり，特定の処理に要求が集中したりする場合があります。急激な状況の変化に即応して，システムを安定稼働させるためには，次の制御が有効です。

- **流量制御**
  クライアントからのアクセス要求や処理要求などのリクエストに対して，処理の同時実行数を制御することで，リクエスト増加時のパフォーマンス低下を防いでシステムを安定稼働させる方法です。
- **優先制御**
  クライアントからのリクエスト増加時に，特定の処理要求を優先して処理することで，処理の緊急度や優先度に応じたリクエスト処理を実現する方法です。
- **負荷分散**
  リクエストを処理するサーバを 1 か所に集中させないで，複数の実行環境で分散させることで，パフォーマンスの低下を防いでシステムを安定稼働させる方法です。

アプリケーションサーバでは，J2EE アプリケーションに対して，きめ細やかな流量制御，優先制御を実現できます。これによって，システムの安定稼働に加えて，システムリソースの有効活用も実現できます。

ここでは，Web アプリケーションの流量制御，および OLTP 技術を適用した Enterprise Bean の流量制御と負荷分散について説明します。なお，それぞれの機能の詳細については，マニュアル「アプリケーションサーバ 機能解説 基本・開発編(Web コンテナ)」の「2.13　同時実行スレッド数の制御の概要」およびマニュアル「アプリケーションサーバ 機能解説 拡張編」の「3.　CTM によるリクエストのスケジューリングと負荷分散」を参照してください。

(1)　Web アプリケーションの流量制御

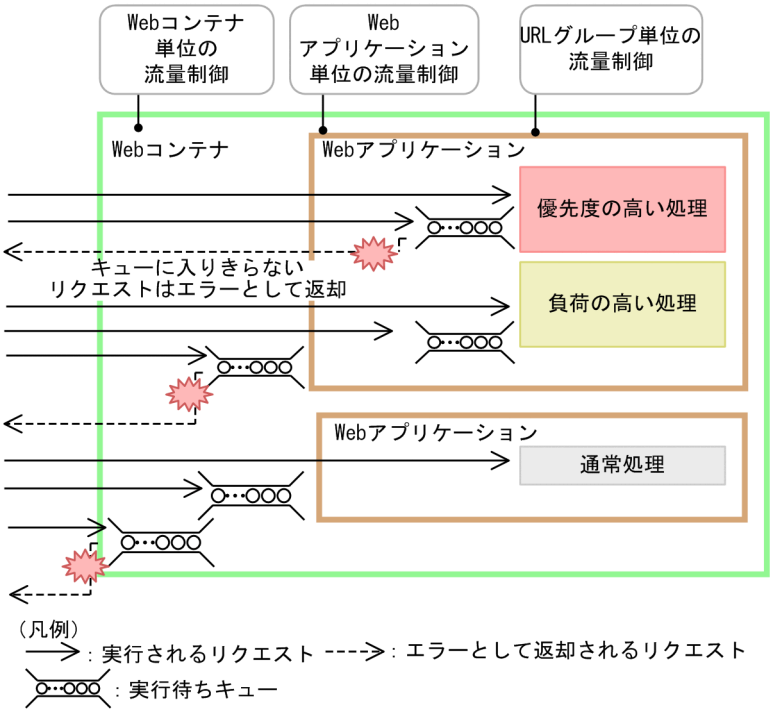
アプリケーションサーバでは，サーブレットや JSP によって構成される Web アプリケーションを，J2EE サーバの Web アプリケーション実行基盤である **Web コンテナ**で実行します。

アプリケーションサーバの Web コンテナでは，Web コンテナ単位，Web アプリケーション単位および Web アプリケーション内の **URL グループ**（業務ロジック）単位に同時に処理できるリクエスト数を設定したり，リクエストをキュー（待ち行列）の概念で管理したりできます。これによって，Web アプリケーションで同時に実行する処理数を，処理内容に応じて細かく制御できます。送信されたリクエスト数が突発的に急増した場合でも，処理するリクエスト数を一定の数に制御できるので，システムを安定した状態で稼働させることができます。

また，リクエストの重要度に応じたリクエストの実行が可能になります。例えば，緊急度の高い重要な処理は確実に処理できるようにすることで，負荷の高い業務処理がほかの業務に影響を与えることを防げます。

Web アプリケーションの流量制御の概要を次の図に示します。

図 4-4　Web アプリケーションの流量制御の概要



Web コンテナ単位，Web アプリケーション単位および URL グループ（業務ロジック）単位で流量制御を実施することによって，システムの安定稼働を実現しながら，負荷の高い処理があっても優先度の高い処理を確実に実行できるシステムを構築できます。

(2) OLTP 技術の Enterprise Bean への適用

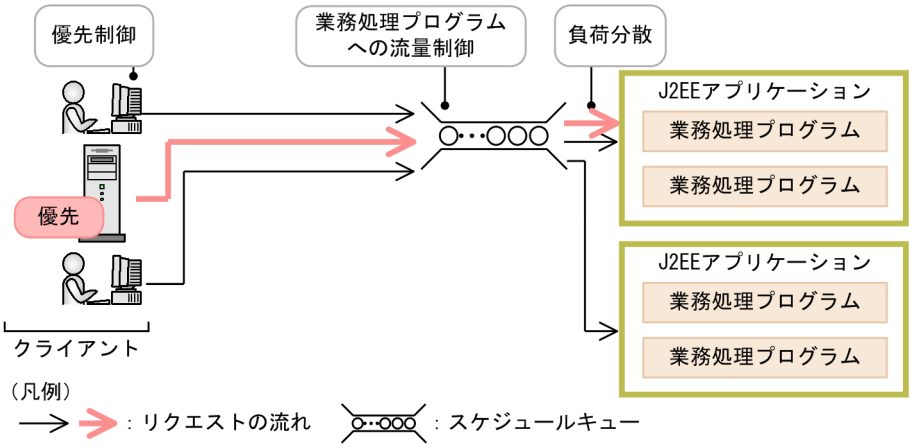
OLTP 技術は，大規模な業務システムには不可欠な技術です。アプリケーションサーバでは，J2EE アプリケーション内の業務処理プログラム（Enterprise Bean）に対して，高度な OLTP 技術を適用できます。これによって，次のような処理を実現できます。なお，対象となる Enterprise Bean は，Stateless Session Bean です。

- **リクエストのスケジューリングを利用した流量制御と負荷分散**

  業務処理プログラムに対するリクエストをスケジューリングすることで，特定の J2EE アプリケーションに大量のリクエストが集中した場合に，リクエストを複数のサーバに振り分けて処理させたり，一度に処理させるリクエストの数（流量）を制御したりできます。また，リクエストの送信元であるクライアントに優先順位を設定して，そのクライアントから送信されたリクエストを優先的に処理するようにできます。さらに，特定の業務処理プログラムに対する処理を適切に分散させて負荷の集中を防ぐことで，システム全体としての処理性能の向上と，システムの安定稼働を図れます。

  Enterprise Bean の優先制御，流量制御および負荷分散の概要を次の図に示します。

  図 4-5　Enterprise Bean の優先制御，流量制御および負荷分散の概要



  J2EE アプリケーション内の業務処理プログラムへのリクエストを，スケジュールキューを経由させて実行することで，一度に大量のリクエストが送信された場合も同時に実行する数を制御したり，負荷分散したりできます。また，優先制御によって，重要なリクエストを速やかに確実に実行できます。

- **サービス閉塞の実現**

  J2EE アプリケーションの稼働中に特定の業務処理プログラムを入れ替えたい場合に，対象の業務処理プログラムに対応するリクエストを制御して，関連するサービスだけを安全に閉塞できます。特定の業務処理プログラムに障害が発生した場合などに，該当個所を局所化し，縮退運転と回復によって，システム全体を止めないで業務処理プログラムの入れ替えができます。

4.3.3　可用性と耐障害性の向上

ミッションクリティカルな業務システムには，システムが提供するサービスをできるだけ停止することなく，安定して提供し続けられる仕組みが求められます。

例えば，提供するサービスの内容によっては，24 時間連続稼働が求められるサービスや，障害が発生して業務システムが止まることで大きな損失が発生するサービスもあります。

このため，業務システムの基盤であるアプリケーションサーバには，まず，障害を未然に防ぐこと，そして，もし障害が発生しても，障害の影響範囲を局所的に抑え，業務システムを止めることなく運用し続けられることが求められます。また，障害が発生した個所は迅速に回復できることが必要です。

ここでは，アプリケーションサーバで実現できる，可用性と耐障害性の高いシステムの特長について説明します。

(1)　FullGC の発生抑止

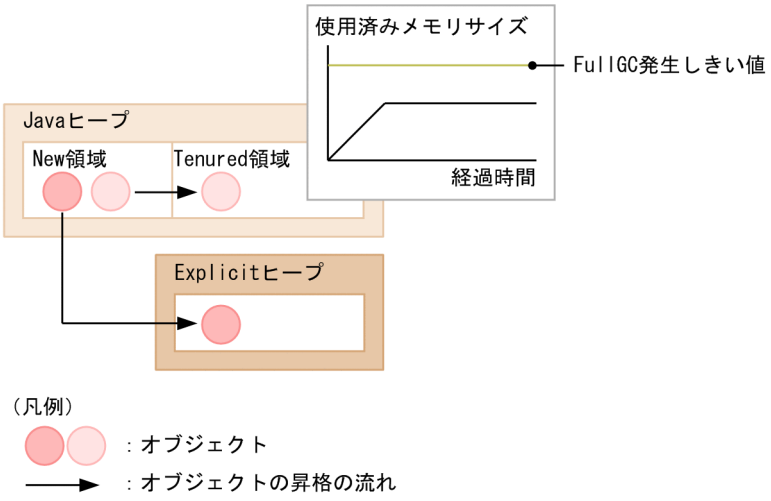
FullGC の発生を抑止することによって，システムが停止する回数を低減します。

FullGC は，Java ヒープの Tenured 領域のオブジェクトが増加することで発生します。Tenured 領域には，長寿命なオブジェクト（利用期間が長いオブジェクト）が配置されます。アプリケーションサーバでは，FullGC 発生の要因になる長寿命なオブジェクトのうち，利用期間が明確なオブジェクトを独自のメモリ空間に配置します。このメモリ空間を Explicit ヒープといいます。Explicit ヒープは FullGC の対象にはならないため，これらのオブジェクトによる FullGC 発生を抑止できます。

アプリケーションサーバでは，HTTP セッションに関するオブジェクトを Explicit ヒープに配置します。

HTTP セッションに関するオブジェクトの場合の Explicit ヒープを使用した FullGC の発生抑止の概要を次の図に示します。

図 4-6　Explicit ヒープを使用した FullGC の発生抑止の概要



New 領域から Tenured 領域に昇格するオブジェクトの一部を Explicit ヒープに移動します。これによって，Tenured 領域のメモリサイズ増加を抑え，FullGC 発生を抑止します。

このほか，ユーザアプリケーションの修正によって，アプリケーション内で FullGC の要因になっているオブジェクトを直接 Explicit ヒープに生成する実装もできます。

(2) 障害発生の未然防止

アプリケーションやリソースの稼働状態を監視して，障害の予兆を検知し，障害発生を予防できます。また，しきい値を設定して監視することで，しきい値を超えた場合に自動的に対処するためのアクションも定義できます。

例えば，リクエストの集中によって JavaVM で FullGC が多発する場合に，一定時間内に発生する FullGC の回数にしきい値を設定して監視できます。しきい値を超えた場合にリクエストの同時実行数を動的に減らすなどの処理を自動実行することで，システムのスローダウンを予防できます。

(3) 障害発生時の可用性向上

障害発生時の可用性を高めるためには，次の機能を使用できます。

- きめ細やかなタイムアウトの設定

  クライアントと Web サーバ間，Web サーバと Web コンテナ間，EJB クライアントと EJB コンテナ間，EJB クライアントとネーミングサービス間などできめ細かくタイムアウトが設定できます。これによって，通信先のマシンに障害が発生している場合でも，数秒で障害を検知でき，無応答などによってサービスを停滞させません。

  また，アプリケーション内のメソッドレベルの処理にもタイムアウトを設定できます。これによって，無限ループなどの問題がアプリケーション内で発生した場合に速やかにその処理をキャンセルして，業務を継続できます。

- クラスタソフトウェアとの連携

  クラスタソフトウェアと連携してアプリケーションサーバや運用管理用のサーバをクラスタ構成にすることで，障害発生時に速やかに系切り替えを実行し，サービスの停止を防げます（1:1 クラスタ構成，相互切り替え構成，またはホスト単位管理モデルを対象とした切り替え構成）。

  また，リソース使用中のアプリケーションサーバで障害が発生した場合に備え，1 台の待機サーバを用意しておくことで，障害発生時にすぐにリソースを解放しトランザクション処理を決着できます（N:1 クラスタ構成）。

- セッション情報の引き継ぎ

  Web システムで，すでにクライアントからのセッションが確立されているサーバに障害が発生したときに，セッション情報をほかのサーバに引き継いで，サービスを継続できます。セッション情報はデータベースで管理できます。Web サーバや J2EE サーバに障害が発生した場合も，データベースからセッション情報を回復し，サービスを継続します。

- バックエンドシステムとの再接続

  データベースなどのバックエンドシステムとの接続が異常終了した場合，自動再接続できます。

- 詳細なログの出力

アプリケーションサーバでは，この製品の JavaVM（以降，JavaVM と呼びます）を提供しています。JavaVM は，Java オブジェクトの状態に関する詳細なログを出力する機能を備えています。このログは，障害発生の要因分析や，性能向上のためのチューニングに活用できます。

4.3.4　システム導入および拡張の容易化

業務システムは，Web サーバ，J2EE サーバ，負荷分散機，運用管理用のサーバなど，複数の要素で構成されています。また，J2EE サーバ内にも，複数の機能があります。アプリケーションサーバを中心とした業務システムを，特長を生かして円滑に運用するためには，システムの用途に応じたシステム構成を検討し，それぞれの要素の多様なパラメタを，適切な関係で設定する必要があります。

ここでは，システムを導入および拡張するときの，アプリケーションサーバの特長について説明します。

(1)　システム構築の容易化

J2EE アプリケーションを実行するシステムは，**セットアップウィザード**を使用して構築できます。セットアップウィザードを使用することで，対話形式の CUI プログラムの画面に設定値を入力しながらシステムを構築できます。

セットアップウィザードが対応していない構成・機能を使用するシステムを構築したい場合には，**運用管理ポータル**や **Smart Composer 機能**を使用した構築ができます。それぞれのツールの特長や使い分けの指針については，マニュアル「アプリケーションサーバ システム構築・運用ガイド」を参照してください。

(2)　開発環境でのアプリケーションの属性設定

アプリケーションサーバ独自の定義情報を開発環境で XML ファイル（cosminexus.xml）に定義できます。このファイルを含むアプリケーションを実行環境の J2EE サーバにインポートすることで，実行環境での属性設定が不要になります。

例えば，リソースとのリンク解決などを開発環境で定義できるため，実行環境ではインポートしてすぐにアプリケーションを開始できます。また，DD を変更してアプリケーションを入れ替えるときなどに，アプリケーションサーバ独自の定義をし直す必要がありません。

なお，実行環境でアプリケーションの属性を設定したい場合は，CUI（サーバ管理コマンド）を使用します。

4.3.5　システム監査によるシステムのセキュリティ確保

会計不祥事などの問題が多く発生する中で，組織には，内部統制の強化が強く求められています。

内部統制の目的は，いつ，だれが，どんな業務を実行したかを把握して，業務が各種法規制に準拠して遂行されていることを検証することです。このため，内部統制に対応するには，次のことが求められます。

- 正しい権利を持つ担当者が，正しい操作によって適切に業務を実施したことを検証できること。

- 検証結果に問題がないことを，監査者や評価者に対して証明できること。

これらに対応するためには，業務システムで，「だれが」「いつ」「何を実施したか」を監査し，監査結果を記録として管理しておく必要があります。

アプリケーションサーバでは，次の機能を提供しています。

- 監査ログの出力
- データベースと連携した監査証跡情報の出力

ここでは，これらの機能の概要を説明します。監査ログの出力の詳細については，マニュアル「アプリケーションサーバ 機能解説 運用／監視／連携編」の「6.4　監査ログの出力」を参照してください。データベースと連携した監査証跡情報の出力の詳細については，マニュアル「アプリケーションサーバ 機能解説 運用／監視／連携編」の「7.6.2　データベースの監査証跡情報の取得」を参照してください。

(1)　監査ログの出力

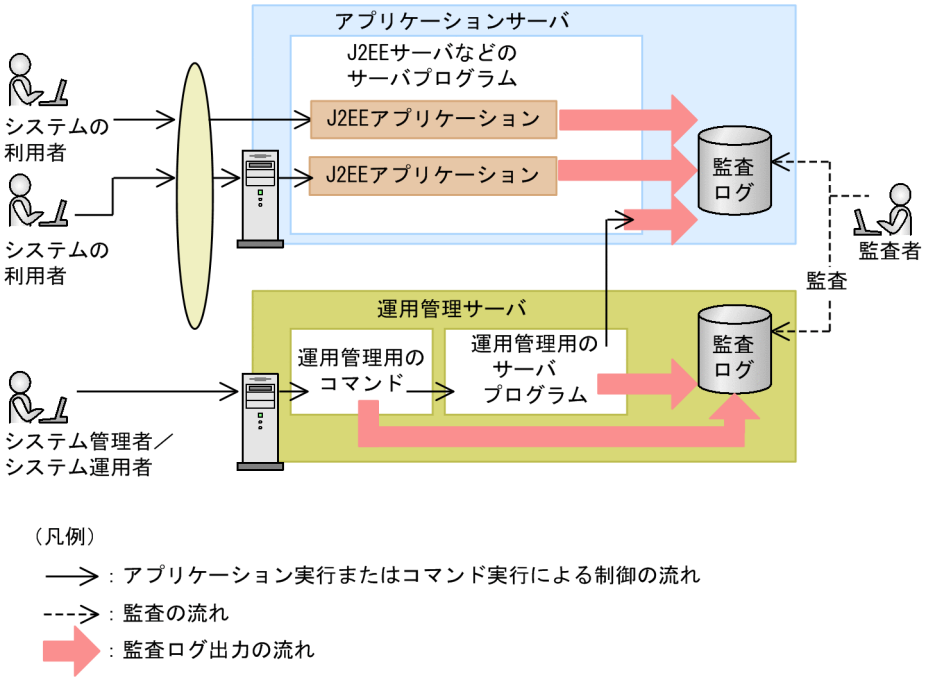
アプリケーションサーバで構築した業務システムでは，「だれが」「いつ」「何を実施したか」についての情報を，**監査ログ**として出力できます。

監査ログに出力されるのは，次の情報です。

- システム管理者やシステム運用者が実施した操作の履歴と，それに伴うプログラムの動作の履歴
- J2EE アプリケーションを通してシステムの利用者が実行した操作の履歴と，それに伴うプログラムの動作の履歴

アプリケーションサーバでの監査ログ出力の概要を次の図に示します。

図 4-7　アプリケーションサーバでの監査ログ出力の概要



なお，アプリケーションサーバが出力した監査ログは，JP1 と連携して，アプリケーションサーバ以外のミドルウェアが出力する監査ログとまとめて管理できます。

JP1 との連携については，「6.　ほかの製品との連携」を参照してください。

(2)　データベースと連携した監査証跡情報の出力

アプリケーションサーバで構築した業務システムのバックエンドでは，多くの場合，データベースが動作しています。データベースには，漏洩や改ざんが許されない重要なデータが多く格納されています。これらの情報は，適正なセキュリティ管理が行われ，厳重に管理される必要があります。

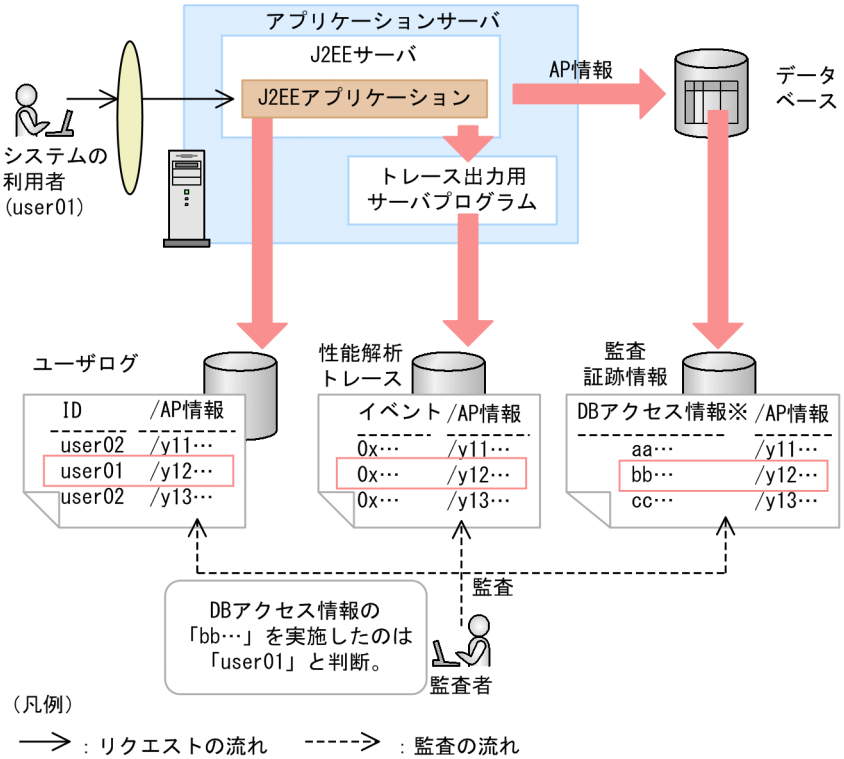
データベースには，「だれが」「いつ」「どのようなデータベースアクセスを実行したか」を示す情報を出力できるものがあります。この情報を，**監査証跡情報**といいます。

アプリケーションサーバでは，データベースが出力する監査証跡情報に，アプリケーションサーバのどのリクエストでデータベースアクセスが実行されたのかを示す情報を出力できます。この情報と J2EE アプリケーションで出力するログ情報などを組み合わせると，データベースアクセスが，アプリケーションサーバのどのユーザの操作の延長として実行されたのかを追跡できます。

なお，アプリケーションサーバが監査証跡情報を出力するために連携できるデータベースは，HiRDB です。

データベースと連携した監査証跡情報の出力の概要を次の図に示します。

図 4-8　データベースと連携した監査証跡情報の出力の概要



注　　AP情報は，ルートアプリケーション情報（リクエストを特定するための情報）を示します。

注※　データベースが管理しているデータベースの操作履歴が出力されます。

この例では，システムの利用者（user01）が J2EE アプリケーションを経由してデータベースにアクセスするときに，次の 3 種類の情報が出力されています。

- J2EE アプリケーションで出力するユーザログ
- トレース出力用サーバプログラム（パフォーマンストレーサ）が出力する性能解析トレース
- データベースで出力する監査証跡情報

これらの情報には，すべてリクエストを特定するための情報（ルートアプリケーション情報）が出力されています。監査者は，この情報を利用して，データベースアクセスがどのリクエストの延長として実行されたのか，そのリクエストを実行したのはどのユーザなのか，などを検証します。また，性能解析トレースを使用すると，そのリクエストがどのような流れで処理されたのかを検証することもできます。

4.3.6 業務効率を向上させる運用管理の実現

業務システムの運用管理は，規模が大きくなるほど複雑になります。できるだけ効率良く，運用コストを抑え，かつシステムの性能を最大限に生かせる運用方法が求められます。そのためには，操作性に優れたインタフェースや，自動運用の仕組みが必要です。

アプリケーションサーバでは，これらの要件を満たすために，次の運用管理機能を提供しています。

- **業務システムの一括運用**

  セットアップウィザード，運用管理ポータルまたは Smart Composer 機能で構築した環境は，運用管理ドメインという概念で管理します。運用管理ドメインは，Management Server というプロセスで管理される範囲です。複数の J2EE サーバや Web サーバなどの各種サーバに対する，起動停止処理などを一括管理できます。

  また，JP1 と連携すると，アプリケーションサーバ以外で構築されたシステムも含めた業務システム全体に対して，監視，運用の自動化，稼働状況の分析などの一括運用ができるようになります。

- **システムの稼働情報やリソースの使用状況の出力**

  システムの稼働状態やリソースの使用状況を出力できます。出力内容を監視することで，トラブル発生を未然に防止したり，発生したトラブルに対処したりできます。また，出力された情報は，システムのチューニングにも利用できます。これらの情報は，ファイルに出力することもできます。

- **運用作業の自動化**

  障害発生時のサーバ再起動や，運用監視機能と連携したイベント発行・アクション実行などの機能を利用することで，障害発生時の回復作業の自動化や，障害予防作業の自動化を図れます。

  また，JP1 と連携することで，アプリケーションサーバ以外のシステムも含めた業務システム全体の運用作業を自動化できます。

- **ログの運用**

  アプリケーションサーバのシステムが出力したログを一括して収集できます。また，ユーザが開発した J2EE アプリケーションのログをアプリケーションサーバのシステムが出力したログと同じように扱うことができます。システム全体のログを一括管理することで，信頼性の高いログ運用を実現できます。

また，アプリケーションサーバで使用する JavaVM では，障害発生時の要因分析やシステムの状態確認に利用できるよう，ログの出力内容が拡張されています。このログを利用して適切なチューニングを実施することで，システムの可用性向上が図れます。

- **トレース情報による性能解析**

  リクエストの実行時に，Web サーバからデータベース接続までの各ポイントでトレース情報を出力します。この情報を収集して分析することで，処理のボトルネックを明確にしたり，障害発生の原因になった個所を特定したりできます。

  なお，トレース情報は，システムだけでなく，ユーザが開発したアプリケーションでも出力できます。

- **統合ユーザ管理**

  アプリケーションサーバで構築したシステムにログインするユーザを統合管理できます。それぞれの J2EE アプリケーションが管理しているユーザの情報が関連づけられて管理されるため，一度のログイン処理でさまざまな J2EE アプリケーションにログインできるようになります。なお，ユーザ情報を管理するリポジトリとしては，LDAP ディレクトリサービスやデータベースが利用できます。

- **メンテナンス時のアプリケーション入れ替えの効率化**

  アプリケーションをアーカイブ形式と展開ディレクトリ形式の 2 種類で管理できます。展開ディレクトリ形式のアプリケーションの場合，アプリケーションを入れ替えるときにアーカイブし直す必要がなく，変更したクラスファイルをリロードするだけで入れ替えができます。また，アーカイブ形式のアプリケーションの場合も，局所的な変更の場合にはアプリケーションの停止や再設定が不要な入れ替え（リデプロイ）ができます。これらによって，メンテナンス時の入れ替え効率を高め，システム運用コストを抑えられます。

4.3.7 Web サービスへの対応

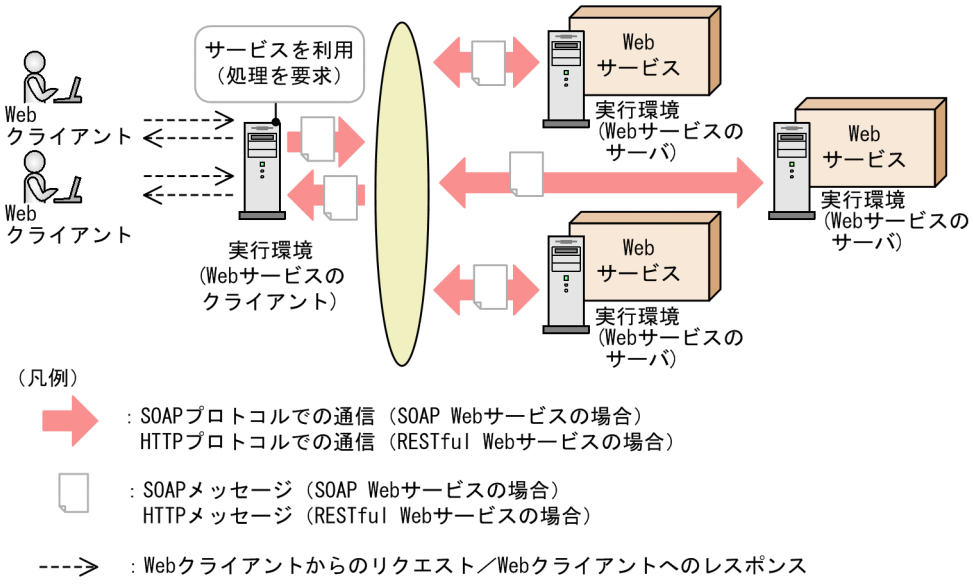
Web サービスとは，Web 関連の技術を利用して，システム間で情報をやり取りし，ほかのシステムが持つアプリケーションを Web 上で使用できるようにした仕組みです。また，Web サービスの仕組みによってネットワーク経由で公開，実行できるアプリケーションのことも，Web サービスといいます。アプリケーションサーバでは，SOAP Web サービスと RESTful Web サービスを実行できます。

SOAP Web サービスでは，SOAP と呼ばれるプロトコルを使用してメッセージをやり取りします。SOAP プロトコルの下位のトランスポート層には HTTP プロトコルを利用します。アプリケーションサーバでは，JAX-WS 仕様に従って，SOAP メッセージと SOAP Web サービス・クライアントとの間のバインディングを実現します。

RESTful Web サービスでは，HTTP プロトコルの GET や POST などのメソッドを直接使用してメッセージをやり取りします。アプリケーションサーバでは，JAX-RS 仕様に従って，HTTP メッセージと RESTful Web サービスの間のバインディングを実現します。RESTful Web サービスの利用環境には，RESTful Web サービスのサーバになる実行環境が必要です。アプリケーションサーバではこの実行環境を構築できます。RESTful Web サービスのクライアントは，RESTful Web サービス用クライアント API か，または標準的な Java API を使用して開発します。

Web サービス利用環境の概要を次の図に示します。

図 4-9　Web サービス利用環境の概要



Web サービスの利用環境には，Web サービスのクライアントになる実行環境と，Web サービスのサーバになる実行環境が必要です。アプリケーションサーバでは，これら両方の実行環境を構築できます。

Web クライアントのユーザからのリクエストを受け付けた実行環境（Web サービスのクライアント）は，Web サービスを提供する実行環境（Web サービスのサーバ）に SOAP メッセージ（SOAP Web サービスの場合）または HTTP メッセージ（RESTful Web サービスの場合）を送ります。Web サービスのサーバでは，処理を実行して，結果をレスポンスとして Web サービスのクライアントに送信します。クライアントと Web サービスのサーバ間の通信には，SOAP プロトコル（SOAP Web サービスの場合）または HTTP プロトコル（RESTful Web サービスの場合）を使用します。

RESTful Web サービス機能の詳細については，マニュアル「アプリケーションサーバ 機能解説 基本・開発編(Web コンテナ)」の「4.　JAX-RS 2.1 の利用」を参照してください。

SOAP Web サービス機能の詳細については，マニュアル「アプリケーションサーバ Web サービス開発ガイド」を参照してください。

参考

既存機能である SOAP アプリケーション開発支援機能および SOAP 通信基盤を使用して，Web サービスを開発・実行することもできます（このときに開発するアプリケーションを「SOAP アプリケーション」と呼びます）。SOAP アプリケーションを開発・実行する場合は，マニュアル「アプリケーションサーバ SOAP アプリケーション開発の手引」を参照してください。

4.3.8　信頼性の高い非同期通信の実現

アプリケーション間で，メッセージを使用した非同期通信を実現できます。

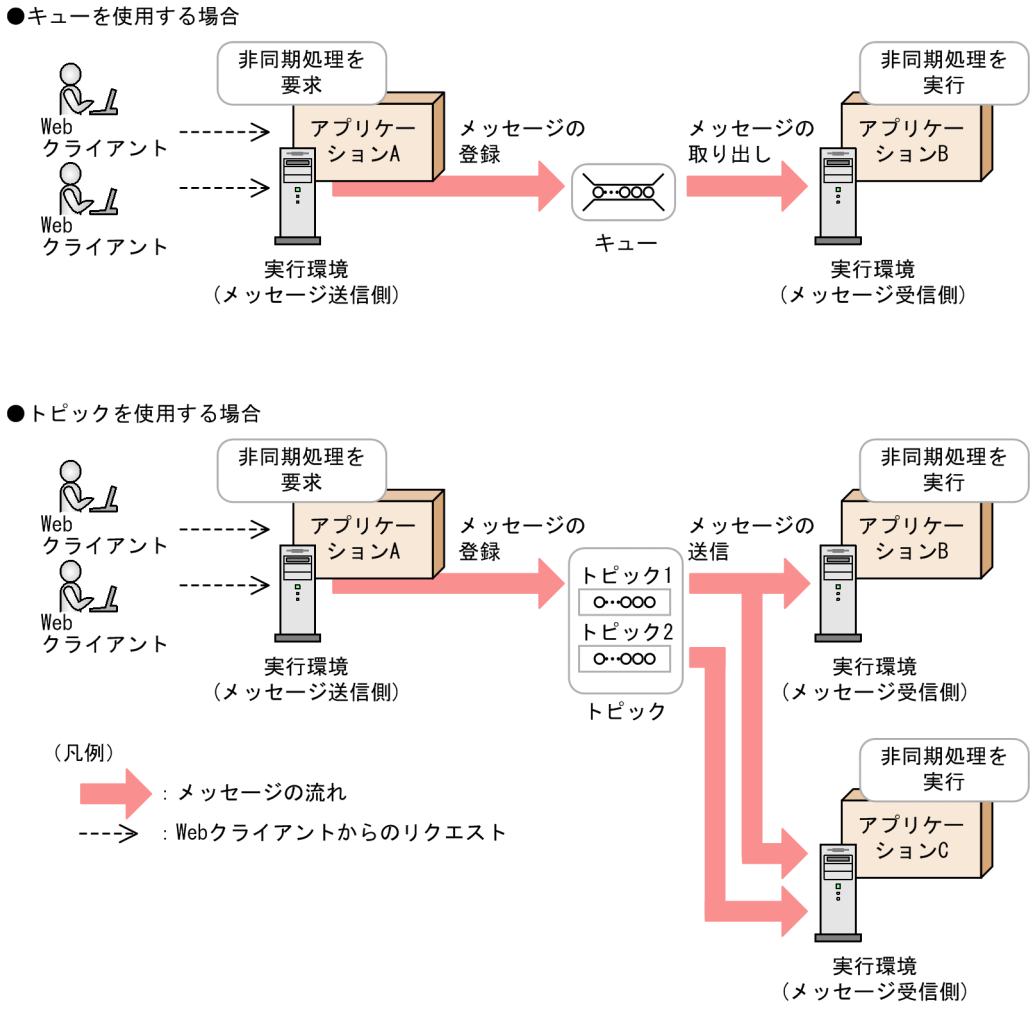
非同期通信では，送信側のアプリケーションがメッセージを送信する際，キューまたはトピックにメッセージを登録します。

キューを使用する場合，受信側のアプリケーションは，メッセージを受信して，キューからメッセージを取り出します。これによって，送信側，受信側のアプリケーションが同時に動作していない場合も，メッセージのやり取りが可能になります。

トピックを使用する場合，あらかじめ購読を申し込んでいた受信側のアプリケーションに対して，メッセージが送信されます。購読は，複数のアプリケーションから申し込めます。

アプリケーションサーバで実現できる非同期通信の概要を次の図に示します。

図 4-10　アプリケーションサーバで実現できる非同期通信の概要



なお，トピックは，CJMS プロバイダの機能を使用した場合だけ使用できます。

また，Reliable Messaging を使用する場合，メッセージを管理するキューをデータベースで管理するため，高信頼なメッセージ管理ができます。また，確実に 1 回の配送保証，または順序保証の QoS（通信品質）などによってメッセージ通信の信頼性も確保できます。さらに，JMS インタフェースを採用し，高信頼な標準プロトコルである WS-Reliability へも対応しているため，ほかのシステムとの連携やほかのシス

テムからのアプリケーションの移植などにも柔軟に対応できます。Reliable Messaging の機能の詳細については，マニュアル「Reliable Messaging」を参照してください。

参考

この項の説明は，アプリケーションサーバが提供する機能を使用した場合の非同期通信の説明です。

このほか，Connector 1.5 仕様に準拠した任意のリソースアダプタを使用して，外部のリソースからのメッセージ送信を受け付けてアプリケーションサーバ上の処理を非同期で実行することもできます。詳細は，マニュアル「アプリケーションサーバ 機能解説 基本・開発編(コンテナ共通機能)」の「3.16.3　メッセージインフロー」を参照してください。

4.3.9　OpenTP1 との連携

アプリケーションサーバと OpenTP1 を連携することで，OpenTP1 を使用したシステムからアプリケーションサーバ上で動作する業務ロジックを呼び出すことができます。これによって，OpenTP1 を使用した実績あるシステムに新たなサービスを追加する場合などに，Java を使用して効率良く開発できるようになります。なお，OpenTP1 とアプリケーションサーバ間の処理の流れは，トレース情報を突き合わせて確認できます。

4.4　バッチアプリケーションの実行環境の特長

この節では，バッチアプリケーションの実行環境の特長について説明します。

4.4.1　オープン環境でのバッチジョブの実行

バッチジョブをオープン環境である Java の実行環境で実行します。

バッチアプリケーションの実行環境は，Java プログラムとして開発されたバッチアプリケーションを実行する環境です。この環境によって，バッチジョブのオープン環境への移行を図れます。

現在，基幹業務の処理を実行するバッチジョブの多くは，メインフレーム上で実行されています。アプリケーションサーバでは，バッチジョブの実行環境を，オープン環境である Java の実行環境として構築します。これによって，Java の特長である柔軟性や運用容易性を兼ね備えたバッチジョブの実行環境を構築・運用できます。

バッチアプリケーションは，常駐型の JavaVM プロセスである**バッチサーバ**で実行します。これによって，バッチジョブを実行するたびに JavaVM を起動するコストを抑えられます。これはレスポンスタイムの比較的短いバッチアプリケーションを繰り返し実行する場合に効果があります。

また，次に示す関連プログラムと連携することで，メインフレームで実現していた内容に類似した処理の制御や自動実行が実現できます。

- JP1/AJS

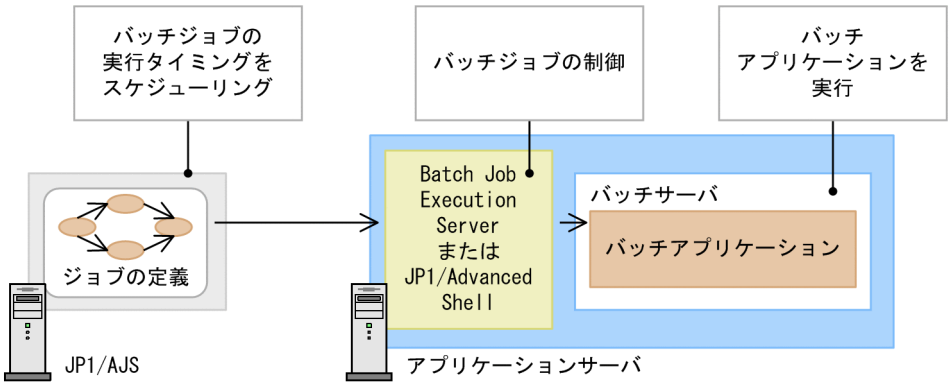
  バッチアプリケーションを実行するタイミングをスケジューリングできます。これによって，バッチアプリケーションを使用する業務を自動化できます。
- Batch Job Execution Server

  メインフレームで実現していたジョブ制御言語によるジョブ管理のイメージで，バッチアプリケーションを実行できます。
- JP1/Advanced Shell

  UNIX の Korn シェルをベースとした，クロスプラットフォームで使用できるスクリプトファイルから，バッチアプリケーションを実行できます。

バッチアプリケーションの実行環境でのバッチジョブ実行の概要を次の図に示します。

図 4-11　バッチアプリケーションの実行環境でのバッチジョブ実行の概要



この構成の場合は，バッチジョブを実行するタイミングを JP1/AJS で制御できます。また，Batch Job Execution Server または JP1/Advanced Shell を使用して定義した内容を基に，バッチジョブを実行できます。

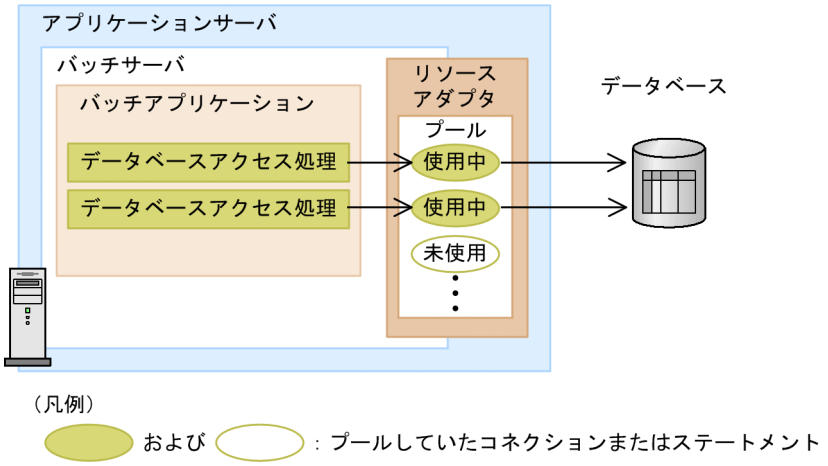
4.4.2　コネクションプール／ステートメントプールを使用したデータベースアクセスの高速化

データベースへの接続で使用するコネクションやステートメントの生成は，時間の掛かる処理です。コネクションプールおよびステートメントプールは，生成したコネクションやステートメントをプールしておくことで，処理性能の向上を図る機能です。

バッチアプリケーションの実行環境では，コネクションプールやステートメントプールなどの機能を使用できます。これによって，バッチアプリケーションからデータベースに高速にアクセスできます。なお，コネクションプールおよびステートメントプールの機能は，リソースアダプタを使用して実現します。

コネクションプールまたはステートメントプールを使用したデータベースアクセスの概要を次の図に示します。

図 4-12　コネクションプールまたはステートメントプールを使用したデータベースアクセスの概要



バッチアプリケーションからデータベースにアクセスするときには，プールしていたコネクションまたはステートメントを使用します。

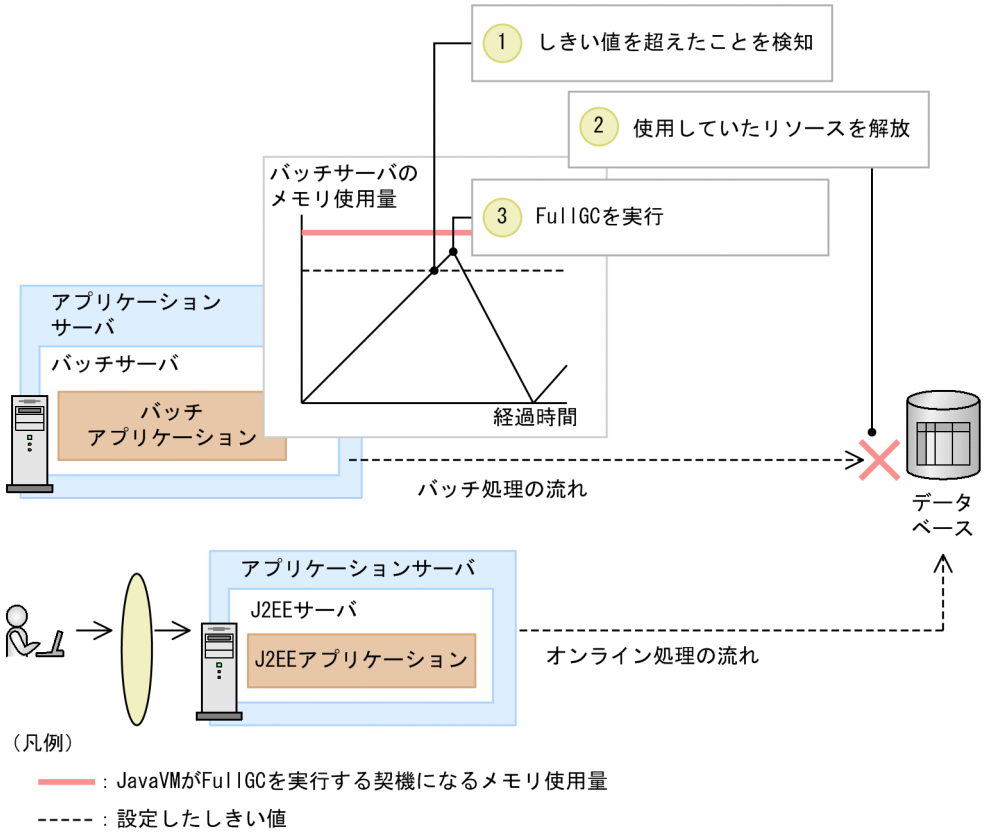
4.4.3　リソース排他状態での FullGC 実行を抑止

JavaVM では，空きメモリが一定のサイズ以下になると，FullGC が実行されます。FullGC が実行されると，その JavaVM 上での処理はすべて中断されます。バッチサーバ上のアプリケーションがリソースを排他状態で使用している時に FullGC が実行されると，排他したままで処理が中断されます。この場合，同じリソースを使用したいオンライン処理の実行も中断されてしまいます。

バッチアプリケーションの実行環境では，バッチサーバ上で FullGC を実行するタイミングを制御できます。この制御では，リソースが排他されていないタイミングを見計らって FullGC を実行して，不要なオブジェクトを解放します。これによって，意図しないタイミングで FullGC が実行されることを抑止できます。

バッチサーバでの FullGC 制御の概要を次の図に示します。

図 4-13　バッチサーバでの FullGC 制御の概要



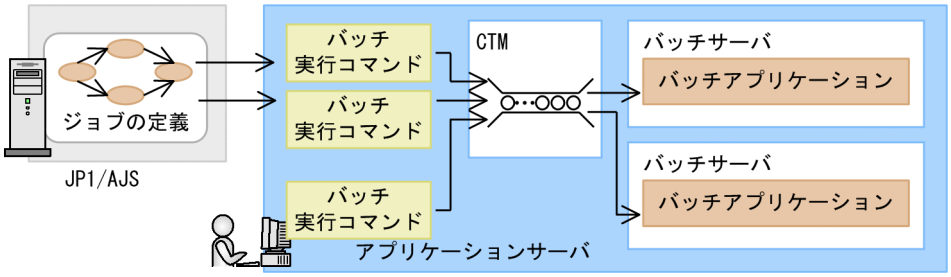
1.で FullGC 制御によって，メモリ使用量がしきい値を超えたことが検知されます。バッチサーバでは，使用していたリソースを解放したことを 2.で確認してから，3.の FullGC を実行します。

4.4.4　バッチアプリケーションのジョブスケジューリング

バッチアプリケーションは，バッチサーバごとに 1 つ実行できます。バッチサーバの数を超えてバッチアプリケーションを実行しようとすると，エラーが返却されてしまいます。バッチアプリケーションのジョブスケジューリング機能を使用すると，バッチサーバの数を超えたジョブの実行要求があった場合に，実行要求を待たせることができます。これによって，エラーを返却することを防ぎます。

バッチアプリケーションのジョブスケジューリングには，CTM の機能を使用します。CTM を使用したジョブスケジューリングを次の図に示します。

図 4-14　CTM を使用したジョブスケジューリング



JP1/AJS 経由で実行したバッチ実行コマンドや，直接実行したバッチ実行コマンドによって，バッチアプリケーションの実行要求が送信されます。CTM では，キューの概念によって，実行要求を管理します。実行可能なバッチサーバがある場合は処理を振り分け，ない場合はスケジュールキューに実行要求を滞留します。

4.4.5　そのほかの特長

ここまでで示した以外に，バッチアプリケーションの実行環境には次の特長があります。これらは，J2EE アプリケーションの実行環境と同じです。

- **可用性と耐障害性の向上**

  「4.3.3　可用性と耐障害性の向上」を参照してください。ただし，セッション情報の引き継ぎは該当しません。

- **システム導入および拡張の容易化**

  「4.3.4　システム導入および拡張の容易化」を参照してください。

- **システム監査によるシステムのセキュリティ確保**

  「4.3.5　システム監査によるシステムのセキュリティ確保」を参照してください。

- **業務効率を向上させる運用管理の実現**

  「4.3.6　業務効率を向上させる運用管理の実現」を参照してください。ただし，統合ユーザ管理，メンテナンス時のアプリケーション入れ替えの効率化は該当しません。

4.5　アプリケーション開発の特長

アプリケーションサーバの開発環境は，Developer をインストールして構築します。Developer では，実行環境上で動作するアプリケーションを，わかりやすい操作で効率良く開発できるようにするための開発環境を構築できます。

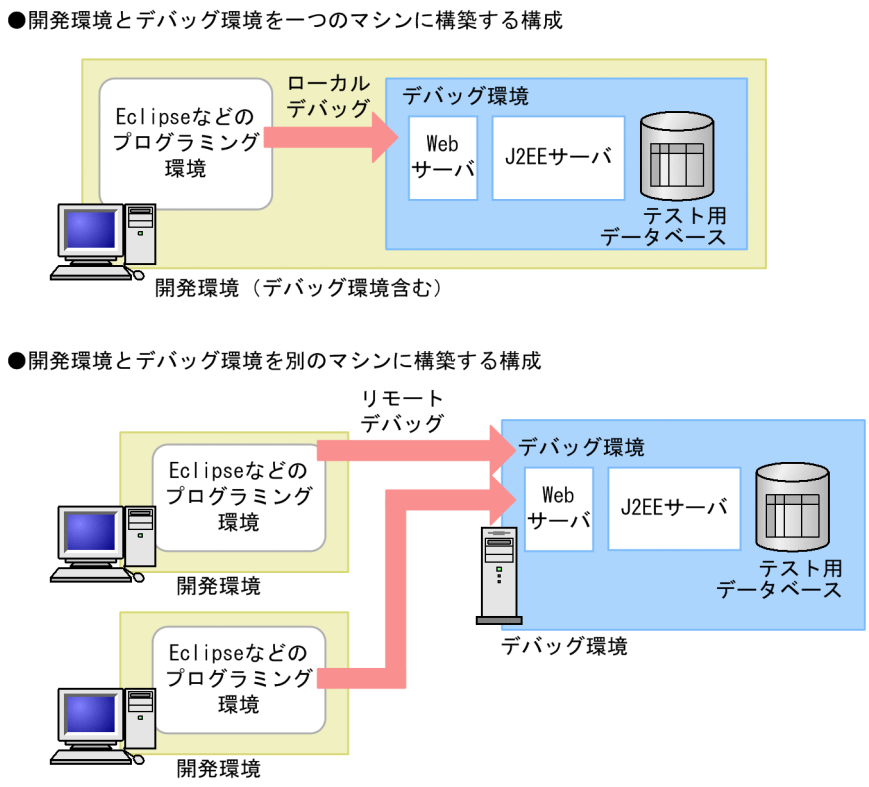
IDE として Eclipse を使用してアプリケーションを開発する場合，Developer で提供するプラグインを組み込んで使用することで，アプリケーション開発時の一連の作業をすべて Eclipse，または Eclipse のプラグインで実現できるようになります。

また，Developer で構築した開発環境では，開発したアプリケーションのデバッグ環境として，実行環境と同じ機能を持つ環境を構築できます。このため，アプリケーションのプログラミングからテスト，デバッグまでの一連の流れを，すべて Developer で構築した開発環境上で実施できます。

4.5.1　アプリケーション開発環境の構築

開発環境は，アプリケーションの開発方法やデバッグの実行方法に応じて，次の図に示す構成で構築できます。

図 4-15　開発環境の構成



- **開発環境とデバッグ環境を 1 つのマシンに構築する構成**

  アプリケーションを一人の開発者が 1 台のマシンで開発する場合などに適しています。ローカルデバッグを実行できます。

- **開発環境とデバッグ環境を別のマシンに構築する構成**
  アプリケーションを複数の開発者のチームで開発する場合などに適しています。リモートデバッグを実行できます。なお，実行環境が Windows 以外の OS でテストを実行するときには，この構成にします。

それぞれの場合にインストールする製品を次の表に示します。

表 4-1　開発環境の構成ごとにインストールする製品

構成	開発環境	デバッグ環境
開発環境とデバッグ環境を 1 つのマシンに構築する構成	• Developer • Service Architect	
開発環境とデバッグ環境を別のマシンに構築する構成	• Developer • Service Architect	• Application Server • Service Platform

4.5.2　開発するアプリケーションの種類

アプリケーションサーバでは，J2EE アプリケーション，Web サービスに対応したアプリケーション，およびメッセージを使用した非同期通信用のアプリケーションを開発できます。それぞれのアプリケーション開発の概要を説明します。

- **J2EE アプリケーションおよびバッチアプリケーションの開発**
  JSP，サーブレット，Enterprise Bean を作成し，J2EE アプリケーションを開発できます。アプリケーションサーバでは，J2EE アプリケーションの開発に WTP を利用できます。WTP を使用すると，JSP，サーブレット，Enterprise Bean の作成からテスト・デバッグまでシームレスなアプリケーション開発が実現します。
  また，バッチサーバ上で動作するバッチアプリケーションも開発できます。
- **Web サービスに対応したアプリケーションの開発**
  Web サービスに対応したアプリケーション（Web サービス）を開発できます。アプリケーションサーバでは，SOAP Web サービスの開発および実行に必要なファイルやソースコード（スタブやスケルトンなど）を生成できます。
  また，Web サービスセキュリティ技術を使用したアプリケーションを開発することもできます。
- **メッセージを使用した非同期通信用のアプリケーションの開発**
  メッセージを使用した非同期通信用のアプリケーションを開発できます。アプリケーションサーバが提供する JMS インタフェースを利用してアプリケーションを開発できます。
- **標準 XML API を利用したアプリケーションの開発**
  JAXP，JAXB，StAX などの標準 XML API を使用したアプリケーションを開発できます。
- **Java EE 環境で動作する CORBA クライアントアプリケーションの開発**
  CORBA クライアントアプリケーションの開発ができます※。

注※　アプリケーションサーバで使用できる機能については，リリースノートを参照してください。

4.6　アプリケーションサーバの機能の一覧および対応する標準仕様

この節では，J2EE アプリケーションを実行するシステムとバッチアプリケーションを実行するアプリケーションサーバの機能の一覧を示します。また，アプリケーションサーバが対応する標準仕様について説明します。

4.6.1　アプリケーションサーバの機能の一覧

アプリケーションサーバの機能は，J2EE アプリケーションを実行するシステムとバッチアプリケーションを実行するシステムとで使用できる機能が異なります。

(1)　J2EE アプリケーションを実行するシステムで使用できる機能

J2EE アプリケーションを実行するシステムの場合に使用できる主な機能について次の表に示します。

表 4–2　アプリケーションサーバの主な機能（J2EE アプリケーションを実行するシステムの場合）

機能分類	概要
Java 言語	Java SE 11 および Java SE 17 に対応しています。※
Web サーバ	Apache HTTP Server をベースに Secure Sockets Layer（SSL）をサポートしたミッションクリティカル分野向けの Web サーバ（HTTP Server）を提供しています。
Web コンテナ	Servlet および JSP に対応した Web コンテナを使用できます。WebSocket や JSF を利用した Web アプリケーションも開発・実行できます。
EJB コンテナ	次の Enterprise Bean を実行できる EJB コンテナを提供しています。 • Session Bean • Entity Bean • Message-driven Bean また，Timer Service の機能も使用できます。
リソース接続とトランザクション管理	次の機能に対応したリソース接続とトランザクション管理ができます。 • コネクションプーリング • コネクションシェアリング • JNDI によるルックアップ • ローカルトランザクション • 2 フェーズコミットメント • 分散トランザクションの実現 また，OpenTP1 との接続や JMS 仕様に準拠した送信先（キューまたはトピック）との接続も実現できます。
スレッドの非同期並行処理	次の機能を使用できます。 • TimerManager を使用した非同期タイマ処理 • WorkManager を使用した非同期スレッド処理

機能分類	概要
XML プロセッサ	JAXP 仕様（DOM/SAX/XSLT/XPath/XMLSchema の各仕様を含む），StAX 仕様，JAXB 仕様に対応しています。
Web サービス	次の Web サービスの実行および開発ができます。 • JAX-WS 仕様に対応した SOAP Web サービス • JAX-RS 仕様に対応した RESTful Web サービス また，既存機能である SOAP 通信基盤および SOAP アプリケーション開発支援機能も使用できます。
OLTP 技術の適用	次の機能を使用できます。 • Web アプリケーションおよび URL グループ（業務ロジック）単位の同時実行スレッド数制御 • Enterprise Bean の同時実行スレッド数制御（CTM の利用）
可用性向上	次の機能を使用できます。 • J2EE サーバ間のセッション情報の引き継ぎ • リソース枯渇監視 • 稼働情報監視 • 性能解析トレース／障害解析トレースの出力
セキュリティ管理	次の機能を使用できます。 • 統合ユーザ管理 • 監査ログ出力 • データベース監査証跡連携
セキュリティ管理（SOAP Web サービス）	次の機能を使用できます。 • SOAP メッセージの完全性および秘匿性の保証 • SOAP メッセージの認証 • XML 署名データの生成および検証 • XML 署名データの暗号化および復号化
フレームワーク・ライブラリ・DI 仕様への対応	次の機能を使用できます。 • JSF および JSTL • CDI • アプリケーションサーバ提供の汎用部品
アプリケーション開発	次の機能を使用できます。 • WTP を使用したアプリケーションの開発 • ローカルマシンでのデバッグおよびリモートマシンでのデバッグ
システム構築	次の機能を使用できます。 • システム構成の一括定義と簡易構築（Smart Composer 機能またはセットアップウィザード） • GUI 画面を使用したシステム構築（運用管理ポータル） • リモート環境からの各種サーバの設定 • アーカイブ形式または展開ディレクトリ形式でのアプリケーションのデプロイ • リソースアダプタのデプロイ • アプリケーションの設定

機能分類	概要
	• リソースの設定
システム運用	次の機能を使用できます。 • サーバプロセスの一括起動，個別起動，自動起動，監視，再起動 • JSP 事前コンパイル • J2EE アプリケーションのリロード • 稼働情報監視によるイベント発行と自動アクション制御 • ログ／トレース収集 • ドメイン一括管理
クラスタソフトウェアとの連携	次のクラスタシステムを運用できます。 • アプリケーションサーバまたは運用管理サーバを対象にした，コールドスタンバイでの 1:1 の系切り替え • 相互スタンバイ構成 • 1 台のリカバリ専用サーバを用意した N:1 リカバリシステム構成 • ホスト単位管理モデルを対象にしたコールドスタンバイでの系切り替え
JP1 との連携	JP1 の各製品と連携して次の機能を使用できます。 • 障害の集中監視 • 稼働性能の監視 • ジョブによる運用の自動化 • SNMP での稼働情報の取得 • 監査ログの収集と一元管理

注※　対応する Oracle 社の JDK バージョンは JDK 11 または JDK 17 です。使用できるコマンドや API の使用方法については，該当ページ（https://docs.oracle.com/javase/11/または https://docs.oracle.com/javase/17/）を参照してください。

(2)　バッチアプリケーションを実行するシステムで使用できる機能

バッチアプリケーションを実行するシステムの場合に使用できる主な機能について次の表に示します。

表 4-3　アプリケーションサーバの主な機能（バッチアプリケーションを実行するシステムの場合）

機能分類	概要
Java 言語	Java SE 11 および Java SE 17 に対応しています。※
バッチサーバ	次の機能を実現できるバッチサーバを提供しています。 • バッチ処理を実装した Java アプリケーションの実行 • FullGC 実行制御
リソース接続とトランザクション管理	次の機能に対応したリソース接続とトランザクション管理ができます。 • コネクションプーリング • コネクションシェアリング • JNDI によるルックアップ • ローカルトランザクション

機能分類	概要
XML プロセッサ	JAXP 仕様（DOM/SAX/XSLT/XPath/XMLSchema の各仕様を含む），StAX 仕様，JAXB 仕様に対応しています。
可用性向上	次の機能を使用できます。 • リソース枯渇監視 • 稼働情報監視 • 性能解析トレース／障害解析トレースの出力
セキュリティ管理	次の機能を使用できます。 • 監査ログ出力 • データベース監査証跡連携
アプリケーション開発	次の機能を使用できます。 • WTP を使用したアプリケーションの開発 • ローカルマシンでのデバッグおよびリモートマシンでのデバッグ
システム構築	次の機能を使用できます。 • システム構成の一括定義と簡易構築（Smart Composer 機能） • リモート環境からの各種サーバの設定 • リソースアダプタのデプロイ • リソースの設定
システム運用	次の機能を使用できます。 • サーバプロセスの一括起動，個別起動，自動起動，監視，再起動 • 稼働情報監視によるイベント発行と自動アクション制御 • ログ／トレース収集 • ドメイン一括管理
クラスタソフトウェアとの連携	次のクラスタシステムを運用できます。 • アプリケーションサーバを対象にした，コールドスタンバイでの 1:1 の系切り替え • 相互スタンバイ構成 • ホスト単位管理モデルを対象にしたコールドスタンバイでの系切り替え
JP1 との連携	JP1 の各製品と連携して次の機能を使用できます。 • 障害の集中監視 • 稼働性能の監視 • ジョブによる運用の自動化 • SNMP での稼働情報の取得 • 監査ログの収集と一元管理

注※　対応する Oracle 社の JDK バージョンは JDK 11 または JDK 17 です。使用できるコマンドや API の使用方法については，該当ページ（https://docs.oracle.com/javase/11/または https://docs.oracle.com/javase/17/）を参照してください。

4.6.2　アプリケーションサーバが対応する標準仕様

アプリケーションサーバが対応している標準仕様について示します。また，アプリケーションサーバでの実装の詳細や留意事項について主に説明しているマニュアルも合わせて示します。

(1) Java EE の標準仕様

表 4–4 Java EE の標準仕様

仕様	備考	主な参照先マニュアル※
Servlet 2.3/Servlet 2.4/Servlet 2.5/Servlet 3.0/Servlet 3.1/ Servlet 4.0	Servlet 3.0 以降の仕様で規定された機能のうち，アプリケーションサーバで使用できる機能については，マニュアル「アプリケーションサーバ 機能解説 基本・開発編(Web コンテナ)」の「8.1 Servlet 仕様および JSP 仕様で追加，変更された機能のサポート範囲」を参照してください。	機能解説 基本・開発編(Web コンテナ)
JSF 2.3	－	
EL 3.0	－	
JSP 1.2/JSP 2.0/JSP 2.1/JSP 2.2/JSP 2.3	JSP 1.1 は使用できません。	
JSP Debugging 1.0	－	
JAX-RS 2.1	－	
WebSocket 1.1	－	
EJB 2.0 • Message-driven Bean • ローカルインタフェース • CMP 1.1 • CMP 2.0	－	機能解説 基本・開発編(EJB コンテナ)
EJB 2.1	CMP 機能のうち EJB2.1 での拡張部分，<service-ref>タグを使用した Web サービス連携機能は使用できません。	
EJB 3.0（Session Bean）	－	
EJB 3.1	－	
Bean Validation 2.0	－	機能解説 基本・開発編(コンテナ共通機能)
CDI 2.0	－	
Common Annotations 1.2	アプリケーションサーバで使用できるアノテーションについては，マニュアル「アプリケーションサーバ リファレンス API 編」の「2.1 アノテーションのサポート範囲」を参照してください。	
JDBC 2.0 コア/JDBC 2.0 オプションパッケージ	－	
JDBC 3.0	接続に使用する JDBC ドライバが，JDBC 3.0，JDBC 4.0，または JDBC 4.3 仕様で規定された機能をサポートしている必要があります。	
JDBC 4.0		
JDBC 4.3		

仕様	備考	主な参照先マニュアル※
JMS 1.0.2	Reliable Messaging または TP1/Message Queue - Access を使用する場合は，JMS1.0.2 であることが前提です。また，Topic を含む一部の機能に制限があります。	
JMS 1.1	JMS1.1 を使用する場合，次の条件があります。 • JMS プロバイダが JMS1.1 に対応していること • Message-driven Bean が EJB2.1 に対応していること（Message-driven Bean を使用する場合） • 使用するリソースアダプタが Connector 1.5 に対応していること	
Connector 1.0（JCA 1.0）	－	
Connector 1.5（JCA 1.5）	－	
JTA 1.0.1 • local • global	リソースアダプタの DD（ra.xml）の transaction-support で LocalTransaction を指定し，ビジネスロジック中にリモートで JavaVM の呼び出しをしない場合に，ローカルトランザクション（local）が利用できます。 ライトトランザクションが有効になっているときは，グローバルトランザクション（global）を使用できません。なお，ライトトランザクションについては，マニュアル「アプリケーションサーバ 機能解説 基本・開発編(コンテナ共通機能)」の「3.14.5　ライトトランザクション」を参照してください。	
JTA 1.1	－	
JTA 1.2	javax.transaction パッケージの @Transactional アノテーション，および Transactional.TxType 列挙型だけをサポートします。その他の JTA 1.2 から追加・変更された仕様は使用できません。	
JPA 2.2	－	
JavaMail 1.2	送信のプロトコルとしては SMTP および SMTPS を，受信のプロトコルとしては POP3 を使用できます。	
JavaMail 1.3		
JavaMail 1.4		
Java Batch 1.0	－	
Concurrency Utilities for Java EE 1.0	－	
JSON-B 1.0	－	

仕様	備考	主な参照先マニュアル※
JSON-P 1.1	－	
Interceptors 1.2	－	

（凡例）－：該当なし

注※　マニュアル名の「アプリケーションサーバ」は省略しています。

(2) XML，Web サービス関連の標準仕様

- JAX-WS 2.2
- JAXB 2.2
- JAXP 1.4（StAX 含む）
- JAXR 1.0
- SAAJ 1.2 と SAAJ 1.3
- SOAP 1.1 と SOAP 1.2
- WSDL 1.1
- WS-RM 1.1 と WS-RM 1.2
- UDDI 2.0 と UDDI 3.0
- WS-I Basic Profile 1.1
- WS-Security 1.1
- WS-Reliability 1.1
- XML-Signature Syntax and Processing
- XML Encryption Syntax and Processing
- Web サービスメタデータ（JSR-181）

XML 関連の標準仕様に関する留意事項については，マニュアル「XML Processor ユーザーズガイド」を参照してください。

Web サービス関連の標準仕様のアプリケーションサーバでの留意事項については，マニュアル「アプリケーションサーバ Web サービス開発ガイド」，マニュアル「アプリケーションサーバ SOAP アプリケーション開発の手引」，マニュアル「アプリケーションサーバ Web サービスセキュリティ構築ガイド」およびマニュアル「XML Security - Core ユーザーズガイド」を参照してください。

(3) SSL 関連の標準仕様

- SSL バージョン 2，SSL バージョン 3
- TLS バージョン 1，TLS バージョン 1.1，TLS バージョン 1.2，TLS バージョン 1.3

SSL 関連の標準仕様に関する留意事項については，マニュアル「HTTP Server」およびマニュアル「アプリケーションサーバ 機能解説 セキュリティ管理機能編」を参照してください。

(4) OMG 分散オブジェクト関連の標準仕様

- CORBA 2.5
- CORBA Object Transaction Service 1.3

OMG 分散オブジェクト関連の標準仕様に関する留意事項については，マニュアル「Borland(R) Enterprise Server VisiBroker(R) デベロッパーズガイド」，マニュアル「Borland(R) Enterprise Server VisiBroker(R) プログラマーズリファレンス」およびリリースノートを参照してください。

5 目的ごとに使用できるアプリケーションサーバの機能の紹介

この章では，システムの目的ごとに使用できるアプリケーションサーバの機能について説明します。

この章の内容を基に，システムの目的を達成するためにアプリケーションサーバのどの機能を使用するかを検討してください。なお，詳細な説明は参照先として記載した各マニュアルでご確認ください。

5.1　システムを構築したい

アプリケーションサーバが動作するシステムは，複数の方法で構築できます。目的に合った方法で構築してください。

ここでは，次の 3 種類の場合について説明します。

- アプリケーションサーバをすぐに動かしてみたい場合
- システム設計からじっくり取り組みたい場合
- 仮想環境にも対応したい場合

5.1.1　アプリケーションサーバをすぐに動かしてみたい場合

まずは標準的な構成のシステムを稼働させて，アプリケーションを実行しながら必要なチューニングを実行したい場合や，すでにほかのシステムで動作していた Java EE に準拠したアプリケーションがあり，すぐに動かしてみたい場合などが該当します。

この場合は，まず，マニュアル「アプリケーションサーバ システム構築・運用ガイド」を参照して，想定しているシステムが**セットアップウィザード**で構築できる構成かどうかを確認してください。

アプリケーションサーバでは，セットアップウィザードで構築できる標準的な構成のシステムとして，次のパターンを想定しています。

- 同一ホストに Web サーバと J2EE サーバを 1 台ずつ配置するシステム
- 同一構成のホストを複数台配置するシステム（リクエストの振り分けには負荷分散機を利用）
- 同一ホストに複数の Web サーバと J2EE サーバを配置するシステム
- 同一の J2EE サーバから複数のリソースにアクセスするシステム

これらの構成のシステムは，構築後，必要に応じて**運用管理ポータル**を使用して設定を変更できます。

5.1.2　システム設計からじっくり取り組みたい場合

マニュアル「アプリケーションサーバ システム構築・運用ガイド」を参照した結果，目的のシステムがセットアップウィザードでの構築対象外だった場合や，パラメタを検証しながら定義ファイルを作成し，システム設計からじっくり取り組みたい場合，**運用管理ポータル**または Smart Composer 機能を使用してシステムを構築します。

これらの機能を使用した構築手順については，マニュアル「アプリケーションサーバ システム構築・運用ガイド」およびマニュアル「アプリケーションサーバ 運用管理ポータル操作ガイド」で説明しています。また，チューニングで使用するパラメタ設定の考え方については，マニュアル「アプリケーションサーバ システム設計ガイド」で説明しています。パラメタの詳細については，マニュアル「アプリケーションサー

バ リファレンス 定義編(サーバ定義)」およびマニュアル「アプリケーションサーバ リファレンス 定義編(アプリケーション／リソース定義)」で説明しています。

5.1.3 仮想環境にも対応したい場合

アプリケーションサーバでは，仮想環境に対応するための機能として，次のような機能を提供しています。

- 複数の仮想サーバへの業務システム（アプリケーションサーバ）の一括構築
- 複数の仮想サーバ上のアプリケーションサーバにある業務（アプリケーション）の一括起動と一括停止
- 業務の規模に合わせた仮想サーバのスケールアウトとスケールイン
- 仮想サーバマネージャを実行できるユーザのアカウント管理

システムを仮想環境に構築する場合に，これらの機能を使用して仮想環境の構築・運用をしたい場合は，マニュアル「アプリケーションサーバ 仮想化システム構築・運用ガイド」を参照してください。

5.2　システムの性能向上を図りたい

システムの性能向上を図りたい場合は，まず，マニュアル「アプリケーションサーバ システム設計ガイド」で説明している**パフォーマンスチューニング**の手順を確認してください。同時実行数の最適化やタイムアウト設定の考え方など，システムの性能を向上させるための考え方と対応する機能について説明しています。

このほか，次のような機能の使用を検討してください。

- NIO HTTP サーバ

  Web アプリケーションを実行するシステムの場合に有効です。Web サーバの機能を J2EE サーバと同一プロセス内で実行することで，性能向上を図ります。この機能については，マニュアル「アプリケーションサーバ 機能解説 基本・開発編(Web コンテナ)」で説明しています。

- リソース接続とトランザクション管理でのパフォーマンスチューニングのための機能

  コネクションやステートメントをプールして再利用したり，ローカルトランザクションの処理を最適化したりすることによって，性能向上を図ります。この機能については，マニュアル「アプリケーションサーバ 機能解説 基本・開発編(コンテナ共通機能)」で説明しています。

また，ここで示した以外の機能を使用してさらにシステムの性能向上を図りたい場合は，「アプリケーションサーバ 機能解説」で始まる各マニュアルの「システムの目的と機能の対応」の説明を確認して，「性能」に該当する機能の使用を検討してください。

5.3 システムの信頼性（アベイラビリティ／フォールトトレランス）を高めたい

システムの信頼性のうち，アベイラビリティ（安定稼働性）およびフォールトトレランス（耐障害性）を高めたい場合は，次のような機能の使用を検討してください。

- セッションフェイルオーバ機能

  Web アプリケーションを実行するシステムで有効です。

  セッション情報をデータベースに格納することで冗長化し，障害発生時のセッション情報の引き継ぎを実現することで，アベイラビリティ／フォールトトレランスの向上を図ります。この機能については，マニュアル「アプリケーションサーバ 機能解説 拡張編」で説明しています。

- FullGC を抑止するための機能（明示管理ヒープ機能）

  FullGC の要因となるオブジェクトを Java ヒープ以外の独自の領域に格納することで，FullGC を抑止し，アベイラビリティの向上を図ります。この機能については，マニュアル「アプリケーションサーバ 機能解説 拡張編」で説明しています。

- リソース接続とトランザクション管理でのフォールトトレランスのための機能

  コネクションの障害検知，コネクションの取得リトライ，コネクションの自動クローズなどによって，フォールトトレランスの向上を図ります。この機能については，マニュアル「アプリケーションサーバ 機能解説 基本・開発編(コンテナ共通機能)」で説明しています。

また，ここで示した以外の機能を使用して，さらにシステムの信頼性を高めたい場合は，「アプリケーションサーバ 機能解説」で始まる各マニュアルの「システムの目的と機能の対応」の説明を確認して，「信頼性」に該当する機能の使用を検討してください。

5.4　システムの信頼性（セキュリティ）を高めたい

システムの信頼性のうち，セキュリティを高めたい場合は，次のような機能の使用を検討してください。

- 不正なユーザからのアクセスを防止する機能（認証機能）
- 通信路でのセキュリティを確保する機能（暗号化機能）
- 不正な処理の実行を防止する機能

このほか，システム構成の検討や，運用方法の検討によっても，システムのセキュリティを確保できます。

これらの機能については，マニュアル「アプリケーションサーバ 機能解説 セキュリティ管理機能編」で説明しています。

5.5　システムを効率良く運用したい

システムを効率良く運用したい場合は，次のような機能の使用を検討してください。

- **日常運用の効率化**

  セットアップウィザード，運用管理ポータルまたは Smart Composer 機能によって構築したシステムは，運用管理ポータルまたは Smart Composer 機能を使用して，サーバの起動や停止などの日常運用が一括して実行できます。

  さらに，JP1 と連携することで，運用の自動化も図れます。

- **稼働情報の監視によるチューニングおよび処理の自動化**

  アプリケーションサーバが出力するサーバの稼働情報を確認することで，システムを最適な状態で運用できるようにメモリなどのリソースをチューニングできます。また，稼働情報に対してしきい値を設定しておくことで，常にユーザが監視していなくても，リソースなどの使用率が一定の値を超えた場合にアラートを受け取るように設定できます。さらに，JP1 と連携することで，稼働情報の集中監視や，アラートを出力した場合の処理の自動化も図れます。

これらの機能については，マニュアル「アプリケーションサーバ 機能解説 運用／監視／連携編」で説明しています。また，ここで示した以外の機能を使用して，さらにシステムを効率良く運用したい場合は，「アプリケーションサーバ 機能解説」で始まる各マニュアルの「システムの目的と機能の対応」の説明を確認して，「運用・保守」に該当する機能の使用を検討してください。

5.6　トラブルに対処したい

アプリケーションサーバが動作するシステムにトラブルが発生した場合は，まずトラブルの発生要因が何かを判断した上で，メッセージなどのログを参照して対処するか，または必要な情報を収集して問い合わせを実施する必要があります。

アプリケーションサーバは，トラブルの発生要因を判断するための，詳細なログを出力する機能を備えています。

システムにトラブルが発生した場合は，まず，マニュアル「アプリケーションサーバ 機能解説 保守／移行編」で説明している**トラブルシューティングの手順**を確認してください。このマニュアルでは，ログを基にしたトラブルの切り分け方法と，代表的なトラブルに対する対処方法について説明しています。また，トラブルシューティングで使用する各種ログの出力方法，出力先，出力内容などについても説明しています。

5.7　アプリケーションを開発したい

アプリケーションを開発したい場合は，次のような機能の使用を検討してください。

- **開発環境インスタントセットアップ機能（Developer を使用してアプリケーションを開発する場合）**
  Developer の機能を使用してアプリケーションを開発する場合に使用できます。テストやデバッグで使用する環境をウィザードに従って構築できます。この機能については，マニュアル「アプリケーションサーバ アプリケーション開発ガイド」で説明しています。
- **Eclipse セットアップ機能（Developer を使用してアプリケーションを開発する場合）**
  コーディングからデバッグ・テストまでを実行できる Eclipse 環境を構築できます。この機能については，マニュアル「アプリケーションサーバ アプリケーション開発ガイド」で説明しています。

このほか，Java EE の標準仕様に準拠したアプリケーションを開発する場合に，アプリケーションサーバの実装を確認したいときには，「4.6.2　アプリケーションサーバが対応する標準仕様」で示したマニュアルを必要に応じて参照してください。

また，アプリケーションサーバが提供する独自の API を使用したアプリケーションを開発したい場合は，マニュアル「アプリケーションサーバ リファレンス API 編」を必要に応じて参照してください。

5.8　アプリケーションサーバが対応している標準仕様の詳細を確認したい

アプリケーションサーバが対応している標準仕様について，アプリケーションサーバでの実装の詳細や注意事項について確認したい場合は，「4.6.2　アプリケーションサーバが対応する標準仕様」で示したマニュアルを必要に応じて参照してください。

6 ほかの製品との連携

この章では，アプリケーションサーバとほかの製品との連携について説明します。

アプリケーションサーバは，データベース，JP1 およびクラスタソフトウェアと連携して，システムを構築および運用できます。

6.1　データベースとの連携

アプリケーションサーバは，次のデータベースと接続できます。

- HiRDB
- XDM/RD E2
- Oracle
- SQL Server
- MySQL
- PostgreSQL

データベースごとに，使用できる機能が異なります。なお，SQL Server は，Windows の場合にだけ使用できます。

ここでは，J2EE サーバから接続できるデータベースおよび接続に使用する JDBC ドライバについて説明します。データベースごとに接続に使用する JDBC ドライバが異なります。サポートしているデータベースおよび JDBC ドライバはリリースノートをご確認ください。なお，データベースとの接続には，JDBC インタフェースを使用して接続する方法と，JDBC インタフェースに加えて JMS インタフェースを使用して接続する方法があります。

J2EE サーバから接続できるデータベース，JDBC ドライバおよび接続に使用するインタフェースの対応を，次の表に示します。

表 6-1　J2EE サーバから接続できるデータベース

データベース	JDBC ドライバ	JDBC インタフェース		JMS インタフェースおよび JDBC インタフェース	
		ローカルトランザクション	グローバルトランザクション	ローカルトランザクション	グローバルトランザクション
HiRDB	HiRDB Type 4 JDBC Driver	○	○	○	○
XDM/RD E2	HiRDB Type 4 JDBC Driver	○	−	−	−
Oracle	Oracle JDBC Thin Driver	○	○	○	○
SQL Server	SQL Server JDBC Driver	○	−	−	−
MySQL	MySQL Connector/J	○	−	−	−
PostgreSQL	PostgreSQL JDBC Driver	○	−	−	−

（凡例）○：使用できます。 －：使用できません。

アプリケーションサーバでは，これらのデータベースと接続するためのリソースアダプタを提供しています。接続できるデータベースおよび使用できる機能の詳細については，マニュアル「アプリケーションサーバ 機能解説 基本・開発編(コンテナ共通機能)」の「3.3　リソース接続」を参照してください。

6.2 JP1 との連携

JP1 は，複合的な業務システムの統合運用管理を実現する，統合運用ソフトウェアです。アプリケーションサーバで構築したシステムは，JP1 と連携することによって，高い運用性を持つシステムとして構築・運用できます。

JP1 では，ジョブ管理，アベイラビリティ管理，ネットワーク管理など，多様なシステムの統合管理を実現できます。アプリケーションサーバで構築したシステムとほかの業務システムとを統合して，障害監視や稼働性能監視をしたり，サーバやアプリケーションの起動／停止を自動化したりできるようになります。

JP1 との連携で実現できる機能と使用する製品について，次の表に示します。

表 6–2 JP1 との連携で実現できる機能と使用する製品

機能	概要	使用する製品
障害の集中監視	システム全体を対象に，障害の集中監視ができます。	JP1/IM
稼働性能の監視	システム全体を対象に，稼働性能の監視ができます。	JP1/PFM
ジョブによる運用の自動化	アプリケーションサーバで管理しているサーバやプロセスの起動／停止を JP1/AJS のジョブを使用して自動化できます。	JP1/AJS
SNMP での稼働情報の取得	SNMP で稼働情報を取得します。	JP1/Cm2/ESA
監査ログの収集と一元管理	アプリケーションサーバなど，システム内の製品が出力した監査ログを収集して一元管理します。	JP1/Audit Management - Manager

JP1 との連携については，マニュアル「アプリケーションサーバ 機能解説 運用／監視／連携編」の「12. JP1 と連携したシステムの運用」を参照してください。また，SNMP での稼働情報の取得については，マニュアル「アプリケーションサーバ 機能解説 運用／監視／連携編」の「8. 運用管理コマンドによる稼働情報の出力」を参照してください。

使用する製品の詳細については，それぞれの製品のマニュアルを参照してください。

6.3　クラスタソフトウェアとの連携

アプリケーションサーバで構築したシステムは，クラスタソフトウェアと連携させることによって，可用性を高めた運用を実現できます。

クラスタソフトウェアと連携してアプリケーションサーバで構築したシステムを運用することで，アプリケーションサーバに障害が発生したときに，待機させておいたアプリケーションサーバに切り替えたり，障害が発生したアプリケーションサーバのリカバリ処理を待機しているリカバリサーバで実施したりできます。また，運用管理用のサーバに障害が発生したときにも，待機させておいたサーバに切り替えることができます。これによって，障害によるサーバの不稼働時間を短縮でき，業務処理の中断を最小限に抑えることができます。

アプリケーションサーバが連携できるクラスタソフトウェアを次の表に示します。

表 6-3　連携できるクラスタソフトウェア

アプリケーションサーバが動作する OS	Windows Server Failover Cluster	HA モニタ
Windows※	○	－
AIX	－	○
Linux	－	○

（凡例）○：使用できます。　－：使用できません。

注※　次の OS が該当します。

- Windows Server 2022 Standard
- Windows Server 2022 Datacenter

クラスタソフトウェアと連携して実現できる機能の詳細については，マニュアル「アプリケーションサーバ 機能解説 運用／監視／連携編」の「15.　クラスタソフトウェアとの連携」を参照してください。

7 SOA の概要

この章では，サービスプラットフォームの基になる考え方である「SOA」の概要について説明します。

7.1 SOA とは

ビジネスを取り巻く環境は，グローバル化，規制緩和，M&A，TOB など，常にダイナミックに変化しています。そして，企業はこのような環境の変化に対して，迅速かつ柔軟に対応する必要があり，それに合わせて，情報システムの変更を必要とする機会が多くなっています。しかし，このような状況のもと，IT インフラがビジネス環境の変化に追従できないといった悩みを抱える企業が増えています。

このように，ビジネスに対して IT は，より深く関係してきており，そのためには，情報システムに対して，ビジネス環境の変化に即応でき，柔軟に対応できるアーキテクチャが必要となります。そこで登場したのが，**SOA**（Service Oriented Architecture（**サービス指向アーキテクチャ**））です。

SOA とは，業務に必要な機能を再利用できる「サービス」として作成し，サービスの組み合わせでシステムを構築しよう，という考え方および技術です。また，業務を実現するために，呼び出すサービスの種類や順序を規定した「ビジネスプロセス」を用いることで，サービスの追加・変更・並び替えが容易になります。

SOA のねらいとして，次のものが挙げられます。

●サービスの再利用

サービスを再利用することによって，重複開発を無くし，生産性を向上させたり，開発単位を局所化したりできます。

●業務プロセスの自動化

業務プロセスを自動化することによって，人の介在を減らし，ミスや不正が混入する可能性を排除したり，ターンアラウンドタイムを短縮したりできます。

●業務プロセスの可視化

業務プロセスを可視化することによって，タイムリーな状況を把握したり，継続的なプロセスを改善したりできます。

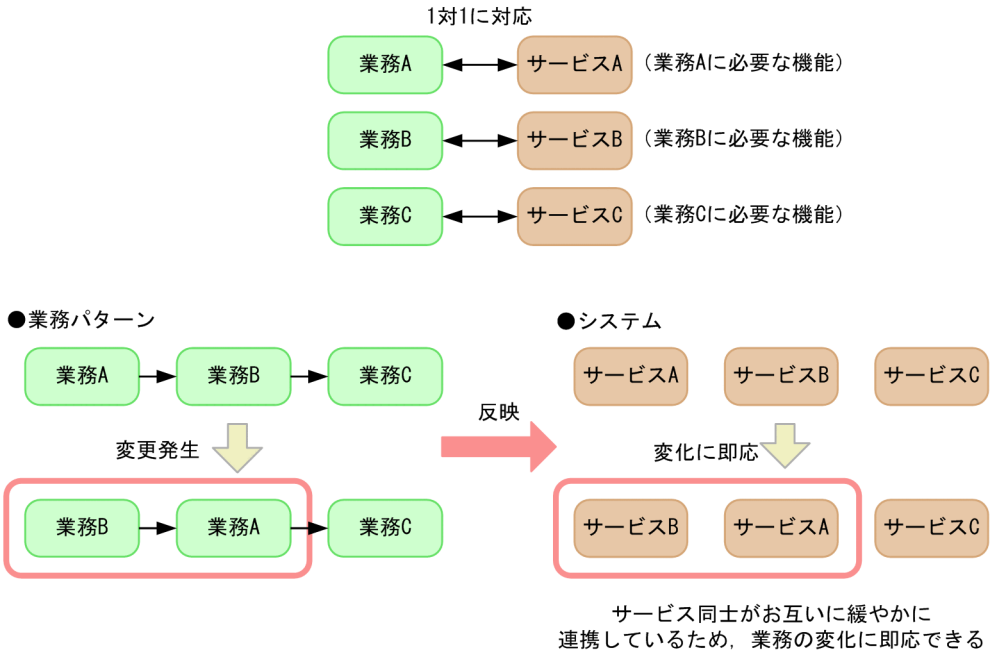
7.2　SOA の目的と利点

SOA には，次に示す 3 つの目的と利点があります。

7.2.1　業務の変化に対してシステムを即応

業務の変化へ即応できるシステムにするために，業務と 1 対 1 に対応する「サービス」という再利用できるソフトウェア部品を組み合わせて業務システムを構築します。従来はアプリケーション単位でシステムを構築していましたが，「サービス」という業務単位でシステム構築します。このため，業務の変化に応じて，システムの改修範囲が特定され，ビジネスニーズに応じた変更や拡張が迅速にできます。業務とサービスの対応を次の図に示します。

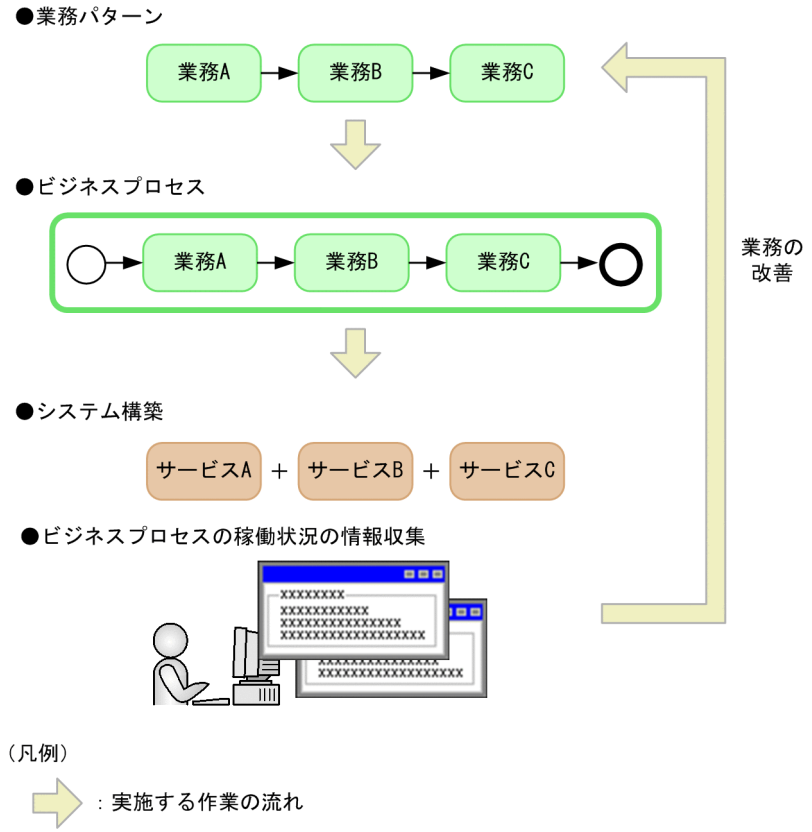
図 7-1　業務とサービスの対応



7.2.2　業務の効率化や最適化

業務の効率化や最適化ができるシステムにするために，複数のサービスを組み合わせた処理の流れをビジネスプロセスとして定義することで，サービスの組み換えや処理の流れを変更しやすくする BPEL 準拠の**ビジネスプロセス管理**を適用します。ビジネスプロセス管理の適用によって，対象となる複数のサービスを業務の流れに従って組み合わせ，ビジネスプロセスとして自動化できます。そのため，従来，人が行っていたシステム間の連携を自動化したり，迅速化したりできます。また，ビジネスプロセスの稼働状況を統一された形式で収集し，1 か所で集中管理できます。業務を可視化できるため，業務の改善を継続的に支援できます。ビジネスプロセス管理と業務の効率化を次の図に示します。

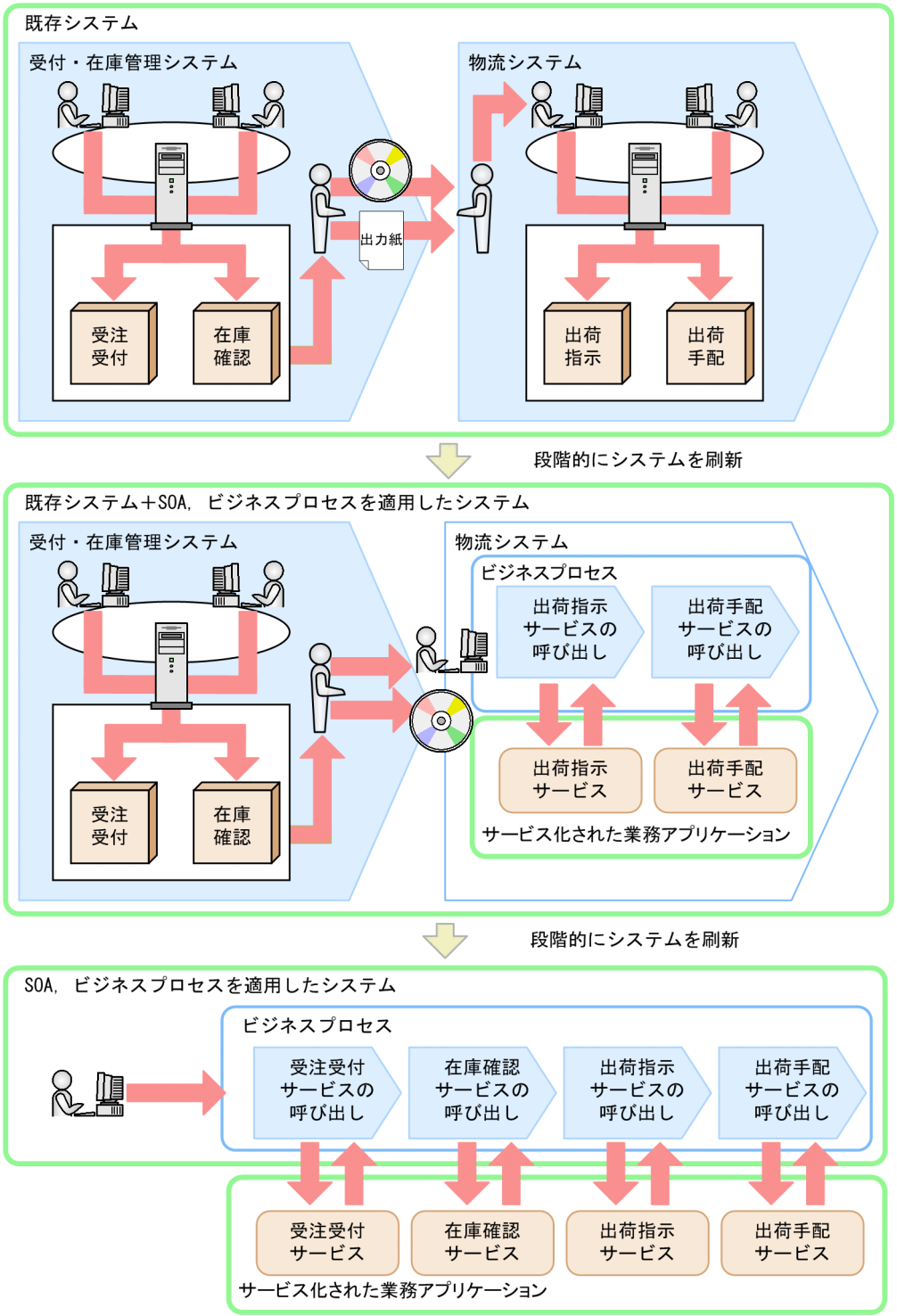
図 7-2　ビジネスプロセス管理と業務の効率化



7.2.3　システムの段階的な刷新

サービスとビジネスプロセス管理の導入による業務システムの最適化を，より優先度の高いところから部分的に着手していけるようにすることで，段階的にシステムを刷新していきます。全体が最適なシステムとなるよう横断的に業務を洗い出し，最終的な目標となる，全体が最適なシステム構成を設定してから計画的に刷新を図ります。既存システムを活用しながら，不足機能やシステムを追加し，老朽化した部分を作り変えていくことで，新しいシステムへの段階的な移行ができます。サービスやビジネスプロセス管理の導入を次の図に示します。

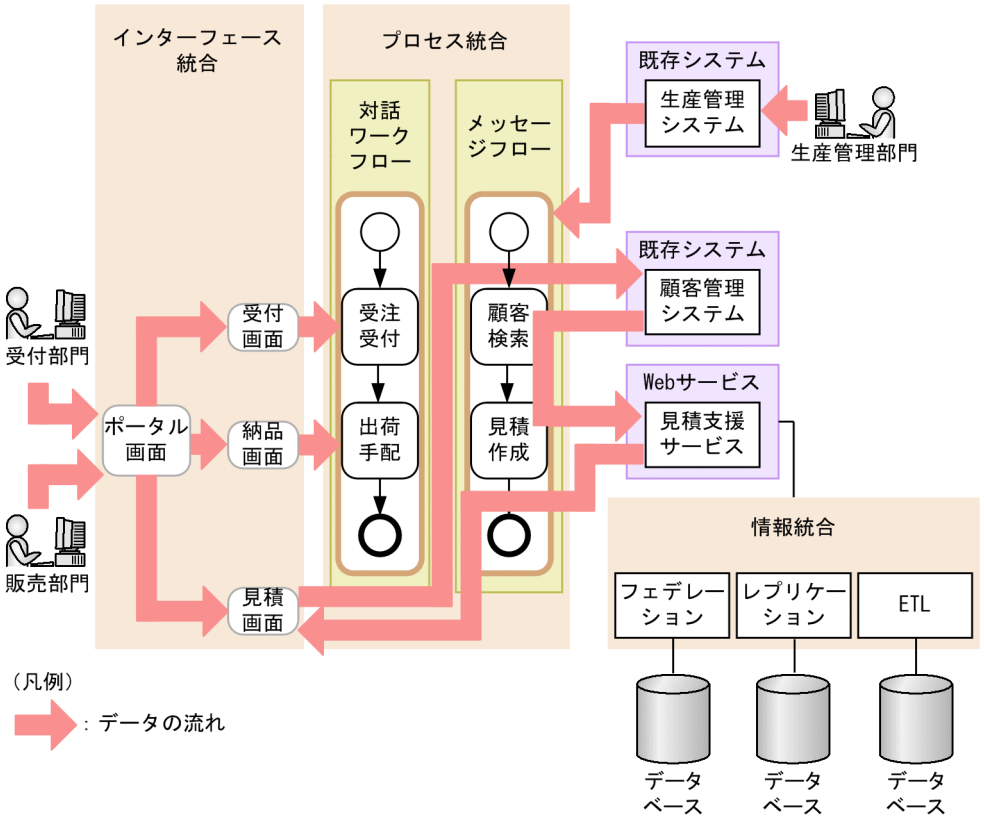
図 7-3　サービスやビジネスプロセス管理の導入



7.3　SOA を適用したシステムの実現

SOA を適用したシステムの実現例を次の図に示します。

図 7-4　SOA を適用したシステムの実現例



SOA を適用したシステムは，サービスプラットフォームを使ってインターフェース統合，プロセス統合，および情報統合によって実現できます。

- **インターフェース統合**

  利用者の複数の担当業務を中心に，画面インターフェースを統合することで，利用しやすい操作環境を実現できます。また，画面上でサービスを連携させる直感的な操作性によって，利用者の生産性を向上できます。

- **プロセス統合**

  SOA では，サービスを柔軟に組み合わせることで，新しいシステムを迅速に構築できますが，この中心となるのが，サービスをプロセスで統合する「プロセス統合」です。サービスを自動的に呼び出すメッセージフローのほか，人がかかわる業務を統合する対話ワークフローを使用できます。

- **情報統合**

  次に示す連携機能によって，システム内に分散したデータを統合・一元化できます。これによって，整合性の取れたデータを各種サービスで共有できます。システム内に分散したデータを必要なときに使用できます。

  - **フェデレーション**

異なるデータベースが管理するマスタから必要なデータを抽出して結合し，仮想表として参照できます。アクセス頻度がそれほど高くない，取得データの少ない業務に対応できます。

- レプリケーション

  データベースを複製してレプリカを作成します。アクセス頻度が高く，取得データの多い業務に対応できます。

- ETL（Extract Transform Loading）

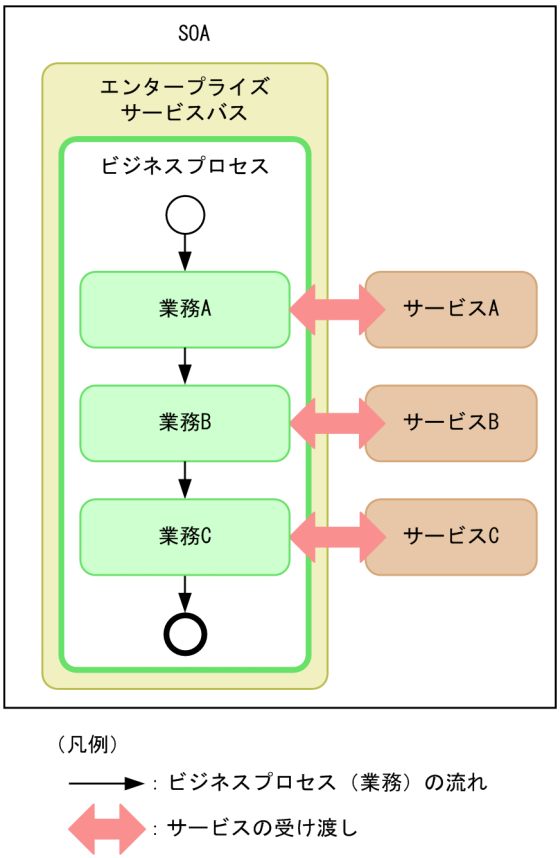
  複数のデータベースからデータを抽出・加工して統合し，新たなデータベースに格納します。複数システムで使用するデータを統合して一元管理するマスタデータの管理などに対応できます。

これらによって，さまざまなサービスを容易に利用できます。

7.4　SOA を構成する要素

SOA では，「サービス」という考え方，およびビジネスプロセス管理を適用していますが，SOA でのサービスとビジネスプロセスとの関係を次の図に示します。

図 7-5　SOA でのサービスとビジネスプロセスとの関係



業務の流れに従ったビジネスプロセスを基に，必要なサービスを呼び出して利用します。サービスの利用者，ビジネスプロセス，およびサービスの間の連携をエンタープライズサービスバス（ESB：Enterprise Service Bus）を介して実現します。

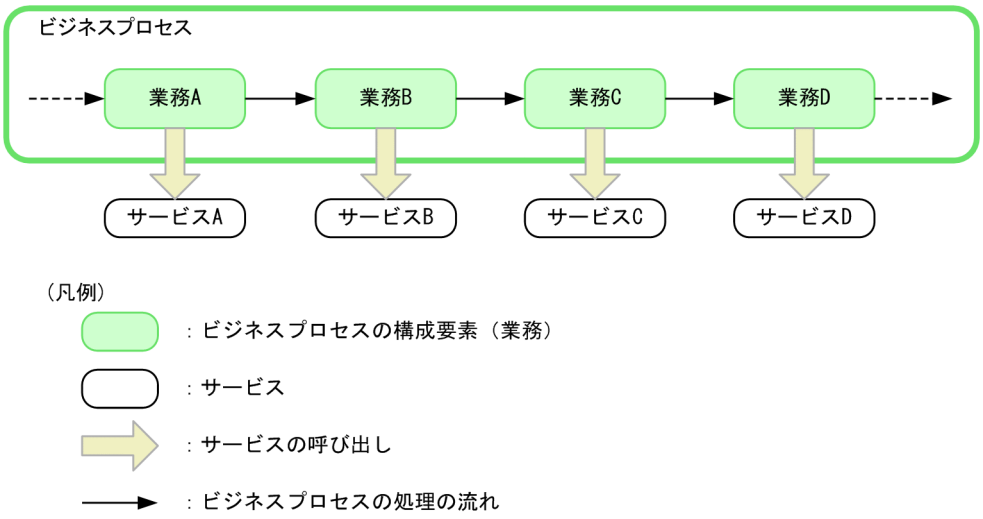
エンタープライズサービスバスは，サービスを組み合わせて実行する部分で，SOA の中心となる部分です。

次に SOA のビジネスプロセスとサービスについて説明します。

7.4.1　ビジネスプロセス

一連の業務処理を実現する作業の流れを**ビジネスプロセス**といいます。ビジネスプロセスを構成する要素を，業務（または業務と 1 対 1 であることからサービス）といいます。また，ビジネスプロセスと複数のサービスから構成される場合，複合サービスといいます。ビジネスプロセスは，複数の業務を緩やかに接続し，いつでも取り替えられる状態で連携させています。ビジネスプロセスの概要について，次の図に示します。

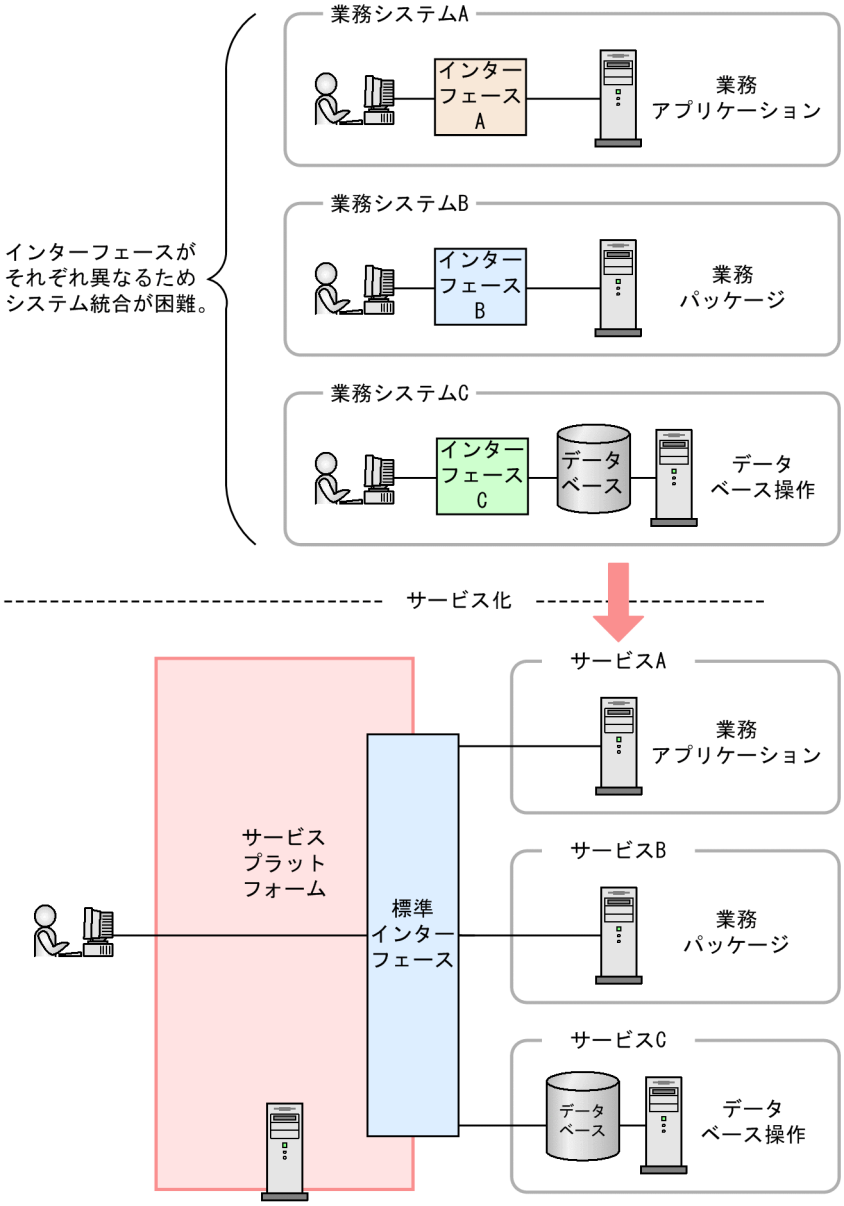
図 7-6　ビジネスプロセスの概要



7.4.2　サービス

SOA で利用する**サービス**とは，業務と 1 対 1 に対応し，再利用できるソフトウェア部品のことです。業務アプリケーションをサービスとして扱う（サービス化する）ことで，これまで分散していたシステムを，SOA を適用したシステムとして統合できます。SOA の考え方を適用した業務システムのサービス化とシステムの統合について，次の図に示します。

図 7-7　業務システムのサービス化とシステムの統合



ネットワーク上に分散する業務アプリケーション，業務パッケージ，データベース操作などは，それぞれインターフェースが異なる場合があります。そのため，システムの統合が困難でした。サービスプラットフォームでは，業務アプリケーション，業務パッケージ，データベース操作などをサービス化して，共通の標準的なインターフェースを使用することで，分散している業務システムを統合できます。

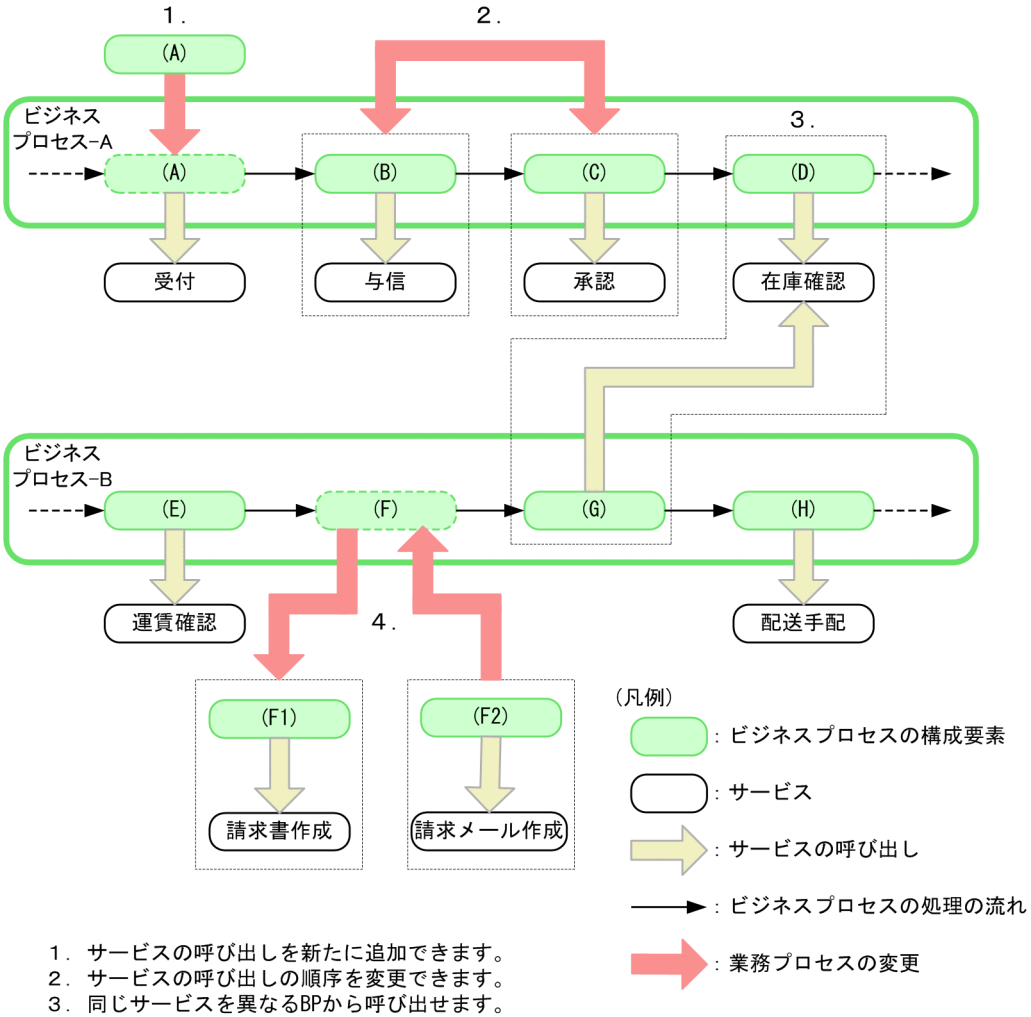
サービスは独立性が高いため，あるサービスで障害が発生したり，業務のプロセスの見直しに伴ってあるサービスを追加・変更したりしても，ほかのサービスに与える影響が少ないという特長があります。また，既存のサービスを再利用することもできます。

(1)　サービスとビジネスプロセスとの関係

ビジネスプロセスを構成するサービスは，互いに緩やかに連携しているため，障害が発生したり，システムを変更する必要があってサービスを入れ替えたりする場合にも，影響範囲がより小さいという利点があります。そのため，業務の手順の変更に即応してビジネスプロセスを変更できます。

また，あるビジネスプロセスで利用しているサービスを，ほかのビジネスプロセスで再利用することもできます。ビジネスプロセスを構成するサービスを追加・変更する場合の例を次の図に示します。

図 7-8　ビジネスプロセスを構成するサービスの追加・変更例



1．サービスの呼び出しを新たに追加できます。
2．サービスの呼び出しの順序を変更できます。
3．同じサービスを異なるBPから呼び出せます。
4．ビジネスプロセスの構成要素を入れ替え，呼び出す処理を変更できます。

(2)　サービスのインターフェース

サービスには，業務に提供する機能を利用するためのインターフェースがあります。このインターフェースを通じて，ビジネスプロセスと連携します。**インターフェース**は，サービスの接点であり，サービスが提供できる機能や必要な入出力データを定義します。サービスが提供する機能の実装は，このインターフェースが入口となり，コンポーネントが担います。**コンポーネント**とは，サービスとして，要求のあった内容を処理する部分をいいます。コンポーネントには，同期処理や非同期処理をするオンライン型アプ

リケーションや，対話型アプリケーションなどがあり，新規開発や，既存システムの再利用，およびパッケージ導入などの方法が利用できます。サービスの構造を次の図に示します。

図 7-9　サービスの構造

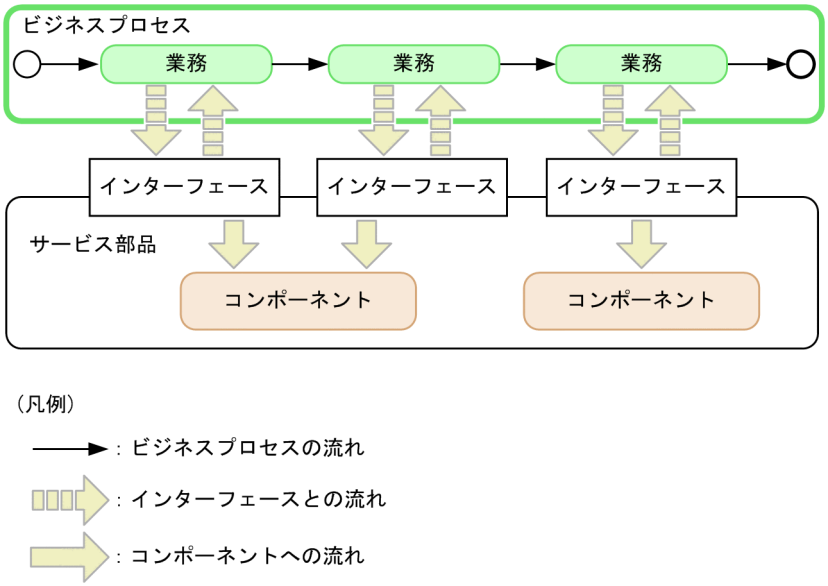


図 7-9 では，各業務は，要求に応じたコンポーネントを，インターフェースを経由して利用することを示しています。

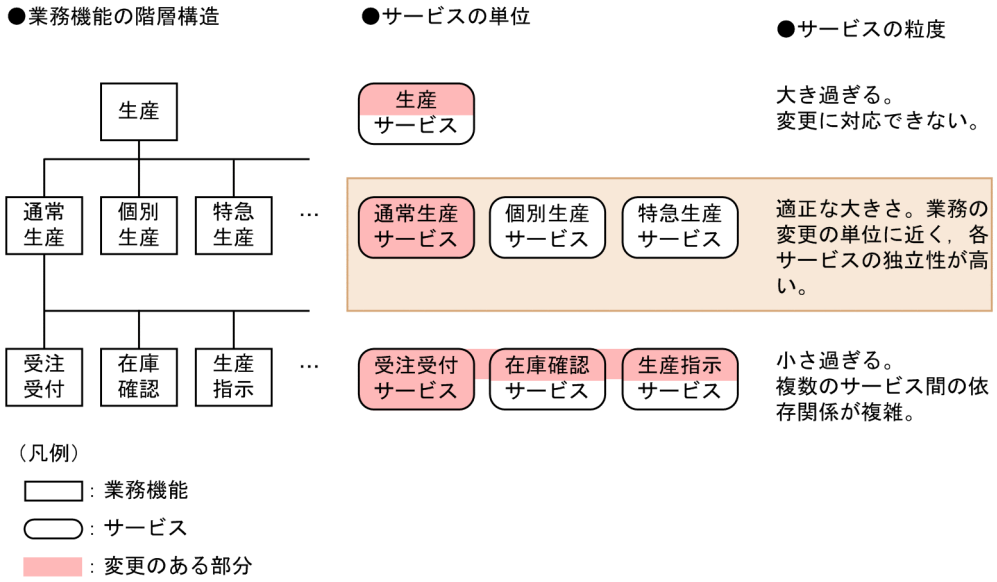
(3)　サービスの抽出とサービスの粒度

SOA を適用したシステムでは，開発から運用まですべてサービス単位で考えます。そのため，サービスの切り出し方やサービスの大きさによって，システムの性能や柔軟性が異なってきます。このサービスの切り出し方のことを**サービスの抽出**といいます。サービスの抽出では，業務内容を調査し，サービスが提供する機能を決めます。また，サービスが提供する機能の範囲（大きさ）を**粒度**といいます。SOA を適用したシステム開発では，粒度の決め方によって，システムがビジネス環境の変化に対応しやすくなるかどうかが決まります。

サービスの粒度は，業務体系や業務フロー，データの依存関係などから，総合的に決定します。サービスとして適切な粒度は，業務上の変更が発生する単位を基に決定します。業務の変化が発生する単位に近く，各サービスの独立性が高いと，サービスの組み替えによる変更で変化を吸収できます。このような単位がサービスの粒度として適切となります。

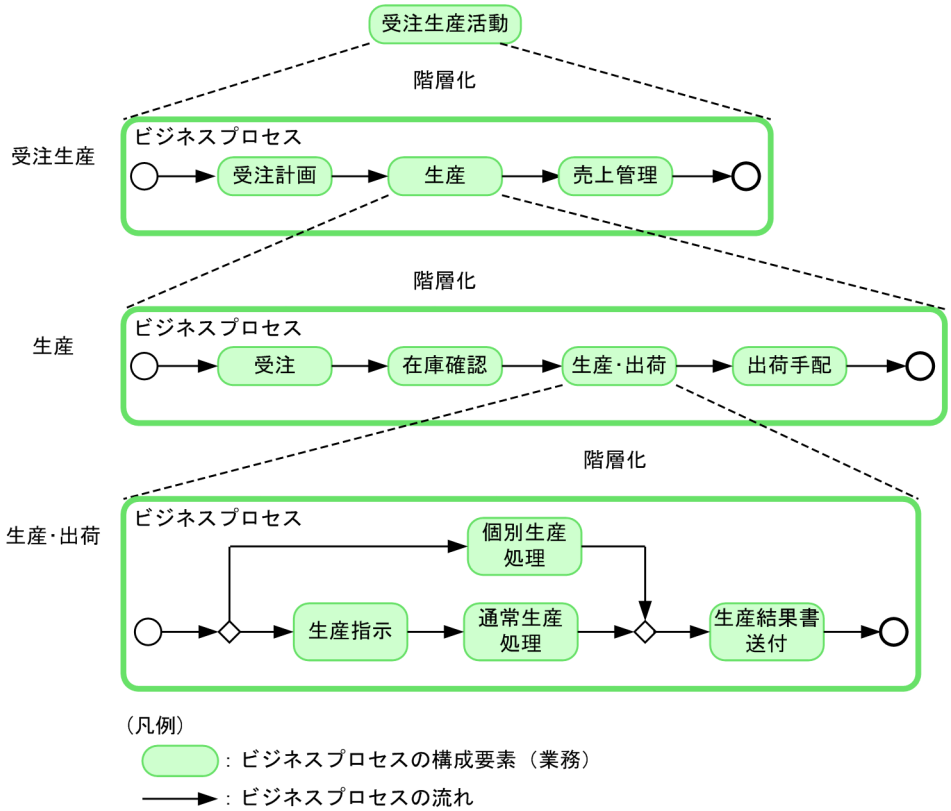
業務の変化が発生する単位に対してサービスが大き過ぎると，サービスの組み替えで変化に対応できなくなります。サービスの単位が小さ過ぎると，サービス間の依存関係が複雑になって，変化対応時の変更が 1 つのサービスで完結できなくなります。サービスの単位とサービスの粒度との関係を次の図に示します。

図 7-10　サービスの単位とサービスの粒度との関係



ただし，業務の変化の内容によっては，上位階層のサービスが変化に対応しやすい場合があります。このような場合は，ビジネスプロセスを階層化することで対応できます。ビジネスプロセスと業務の階層化の例を次の図に示します。

図 7-11　ビジネスプロセスと業務の階層化の例



8 サービスプラットフォームの概要

この章では，サービスプラットフォームの概要について説明します。

8.1　サービスプラットフォームとは

ここでは，サービスプラットフォームの目的や位置づけなどを説明します。

サービスプラットフォームとは，SOA を実現するシステムの開発・運用の基盤となる製品です。統一された開発・運用環境でビジネスプロセスからサービスの接続までを構築・実行できます。そのため，SOA の利点を引き出して，サービスを柔軟に組み合わせて新しいシステムを迅速に構築・実行できます。この中心となるのが，サービスをプロセスで統合する「プロセス統合」です。プロセス統合を実現するのが，サービスプラットフォームです。サービスプラットフォームを使用したシステムの実現例を次の図に示します。

図 8-1　サービスプラットフォームを使用したシステムの実現例

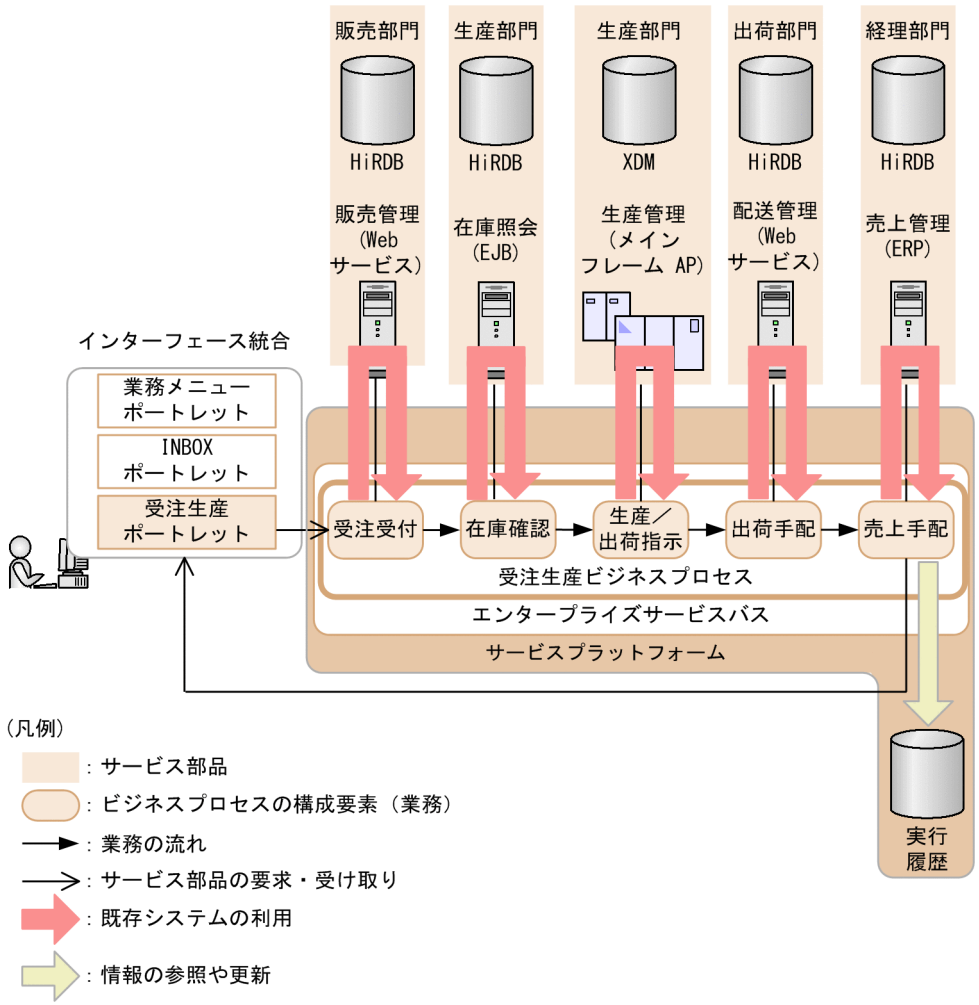


図 8-1 は，生産物流システムに SOA を適用した例です。業務の流れに沿って，サービスを自動的に呼び出せます。図 8-1 の場合，次のような利点があります。

- 業務を段階的にオープン化する場合の対応が容易になります。
- 実業務に応じたビジネスプロセスを実現できます。
- 在庫状況や生産進捗状況を的確に把握し，迅速な納期回答ができます。
- リードタイムを短縮できます。

サービスプラットフォームは，Service Platform および Service Architect から構成されています。Service Platform は実行環境および運用環境に当たり，Application Server の実行環境の機能に加えてサービス統合を実現するための機能があります。Service Platform は，SOA の中心であるエンタープライズサービスバス機能を持ち，サービスを自由に組み合わせて，実行する戦略の変化に即応したシステムを構築できます。既存システムから切り出したサービスや外部から提供されるサービスも自由に組み合わせて，信頼性の高いシステムを構築できます。

Service Architect は開発環境に当たり，Developer の開発環境の機能に加えてサービス統合を実現するための機能があります。Service Architect は，ビジネスプロセス定義，データ変換定義，およびサービスアダプタ定義など，プロセス統合に必要な定義ツールを Eclipse の Plug-in として使用できます。ビジネスプロセスからサービスの接続まで，Eclipse 上の一連の操作でプロセス統合ができます。

SOA を適用したシステム開発手法の中で，Service Architect では，インターフェースを含むビジネスプロセスの詳細設計から実装・テストまでをサポートしています。サービスプラットフォームの機能を利用すれば，コンポーネントの設計・実装ができます。

これによって，すでにアプリケーションの実行環境やサービスプラットフォーム以外の環境で稼働しているサービスを統合して，新しいサービスとしてユーザに提供できます。

8.2　サービスプラットフォームの特長

ここでは，サービスプラットフォームの特長について説明します。

8.2.1　ビジュアル環境でのシステム開発支援

サービスプラットフォームでは，サービスを呼び出すためのサービスアダプタ，ビジネスプロセス，データの変換方法の定義およびシステムに必要なそのほかの各種定義を，画面を利用して視覚的に開発できます。

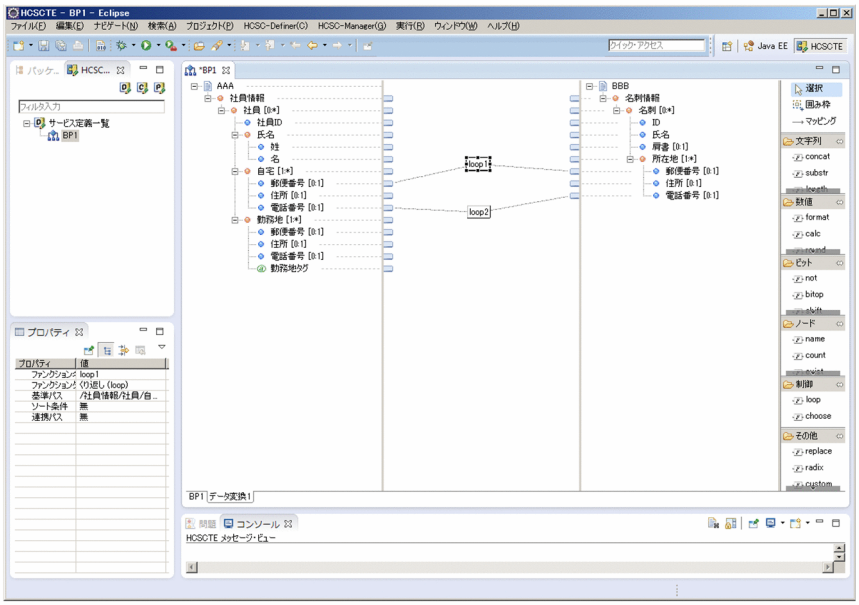
例えば，サービス部品の実行を要求するためのデータ（要求電文）の構造と，実際のサービス部品で利用するデータの構造が異なる場合，構造内の各要素のデータを変換する方法を定義する必要があります。サービスプラットフォームには，異なる構成のデータ間の変換を定義する場合に，データの要素を線で連結（マッピング）する画面が用意されています。

また，ビジネスプロセスは，XML をベースにしたワークフロー記述言語である BPEL で定義する必要があります。サービスプラットフォームでは，ビジネスプロセスを構成する要素（アクティビティ）を画面上に配置，連結してビジネスプロセスを定義できます。このように定義されたビジネスプロセスから BPEL を生成できます。

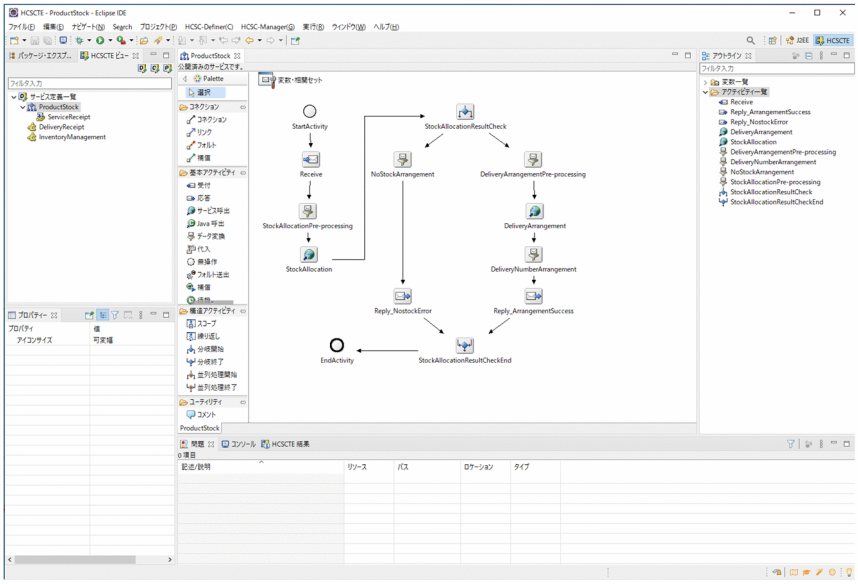
次の図は，サービスプラットフォームで利用するシステム開発用の画面の例です。

図 8-2　システム開発用の画面の例

●データの変換を定義する画面の例



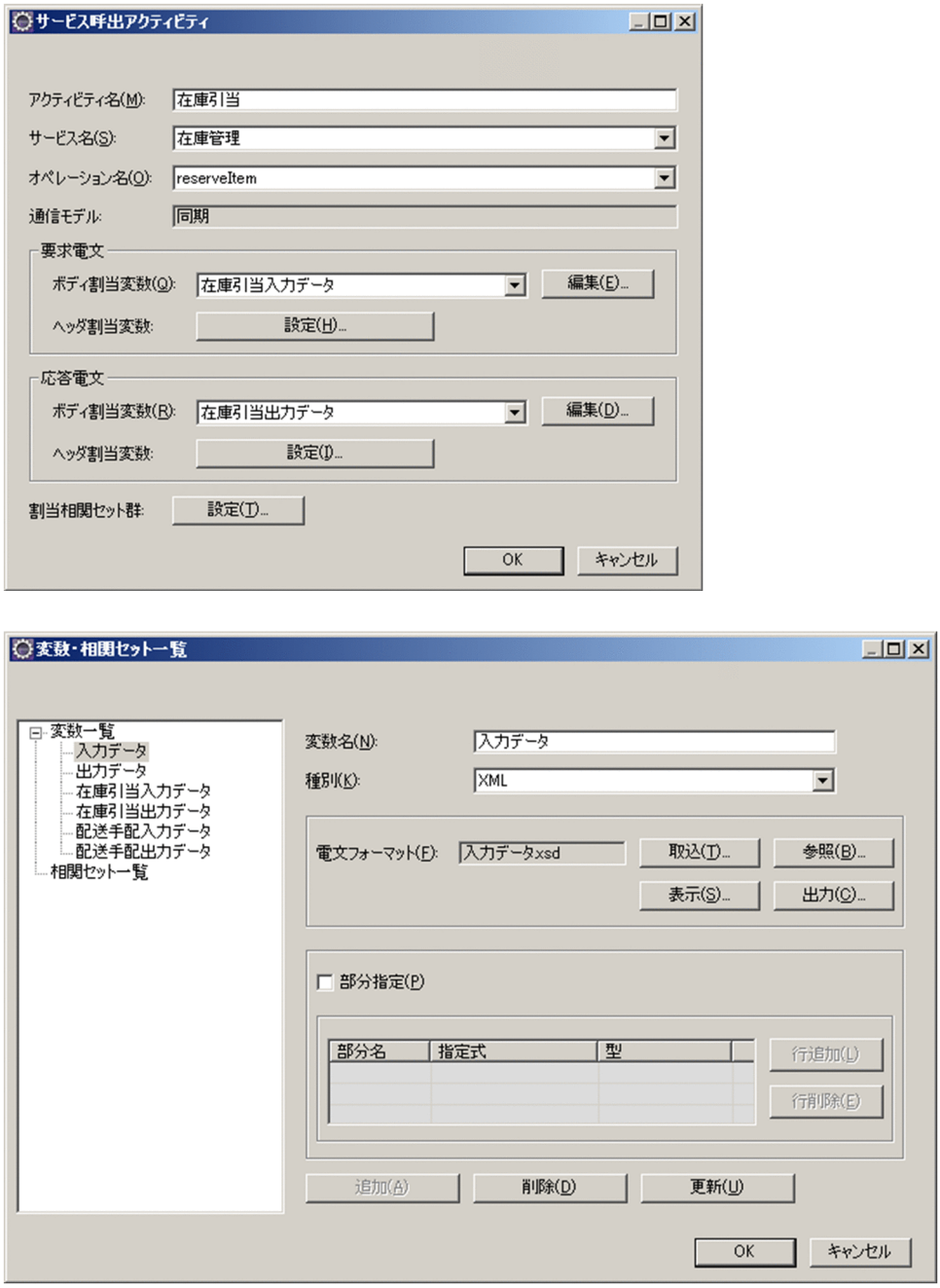
●ビジネスプロセスを定義する画面の例



また，データ変換，ビジネスプロセスおよびサービスアダプタ（Web サービスのサービス部品を呼び出すための SOAP アダプタやデータベースを操作するための DB アダプタなど）の詳細なパラメタを定義する場合，画面から適宜ダイアログを表示して定義できます。

次の図は，サービスプラットフォームで利用するシステム開発用のダイアログの例です。

図 8-3　定義内容の詳細を設定するダイアログの例



このように，ビジュアルな環境でのシステム開発ができるため，システム開発者のプログラミング作業の負担が少なく，開発のスピードが向上します。開発のスピードが向上することで，ビジネス環境の変化に合わせて迅速に対応できます。また，システムの構築に必要なコストの低減を図れます。

8.2.2　業界標準技術を利用した可用性，拡張性の確保

サービスプラットフォームでは，SOA を適用したシステムを実現するために，次のような技術を利用・サポートしています。

基礎となる技術

サービスプラットフォームで扱うデータの形式は，Web サービスとの親和性の高い XML 形式を採用しています。また，構築するシステムは，Java をベースにしており，プラットフォーム間のポータビリティが確保できます。

利用できるサービス

サービスプラットフォームで構築するシステムでは，利用できるサービスとして Web サービス，SessionBean，および MDB をサポートしています。

開発環境の画面

開発環境の画面は，Eclipse を利用します。サービスプラットフォームで利用する機能は，Eclipse へのプラグインを導入して利用できます。

ビジネスプロセスの記述言語

ビジネスプロセスの記述には，BPEL を利用しています。サービスプラットフォームでは，画面を利用して定義したビジネスプロセスが BPEL として保存されます。

サービスプラットフォームでは，これらの標準的な技術を利用することで，汎用的で，可用性の高いシステムの構築を実現しています。

また，システムを開発する際には，既存のリソースをより有効活用でき，新たな技術の導入に掛かるコストの低減も図れます。

8.2.3 データベース操作のサービス化

サービスプラットフォームでは，データベースの操作をサービス部品の 1 つとして定義し，利用できます。データベースと連携する場合，DB アダプタを利用してデータベースを Web サービス化して連携します。DB アダプタは，サービスプラットフォームの画面を利用して作成できます。

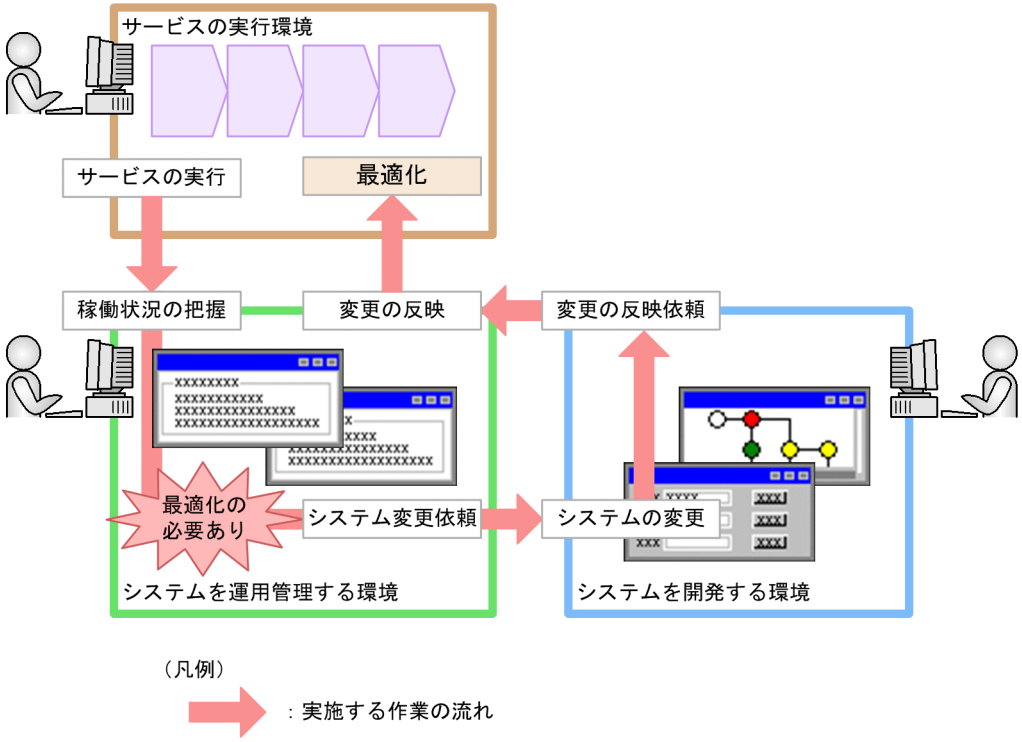
データベース操作をサービス化すると，サービスの利用者は，データベーステーブルへのアクセス制御を意識しないで，データベースを操作できます。

8.2.4 稼働状況の把握とシステムの最適化

サービスプラットフォームでは，システムの稼働状況を確認できます。システムの稼働状況を把握，分析することで，システムの最適化に必要な情報を収集できます。収集された情報と，ビジネス環境を考慮した上で，継続的にシステムを最適化するサイクルを確立できます。

システムの最適化のサイクルを次の図に示します。

図 8-4　システムの最適化サイクル



また，各種ログファイル，トレースファイルも取得できます。障害が発生した場合，各種ログファイル，トレースファイルから，障害の発生個所，要因を調査できます。

8.2.5　インテリジェントな配送制御

サービス部品の実行要求として送信される電文は，サービス部品の種類（単体のサービス部品か，ビジネスプロセスか）に応じて自動的に適切なサービス部品に送信されます。業務担当者は，実行するサービス部品の種類や所在を意識しないで，サービス部品を利用できます。

8.2.6　データ変換による利用データの相違の解消

サービスプラットフォームには，サービス部品の実行要求を受け付けるための標準的なデータの構造（電文フォーマット）があります。これを**標準電文**といいます。しかし，サービス部品の種類はさまざまで，サービス部品が要求するデータの構造（電文フォーマット）もさまざまです。そのため，サービス部品側で要求するデータの構造が，標準電文と異なる場合もあります。

このような場合に，標準電文のデータの構造と，サービス部品側で要求するデータの構造の差異を吸収する機能がデータ変換機能です。

データ変換機能では，異なる構造のデータの相互変換のしかたを定義しておきます。この定義を利用して，サービス部品の実行要求時に自動的にデータ構造の相違を解消し，さまざまなデータの構造を持つサービス部品の実行を要求できます。

8.2.7　既存システムの有効活用

サービスプラットフォームには，サービス部品を利用するために，次に示すようなさまざまなサービスアダプタが用意されています。これらのサービスアダプタを利用することで，柔軟にシステムを構築でき，既存システムのデータをサービス部品として有効に活用できます。

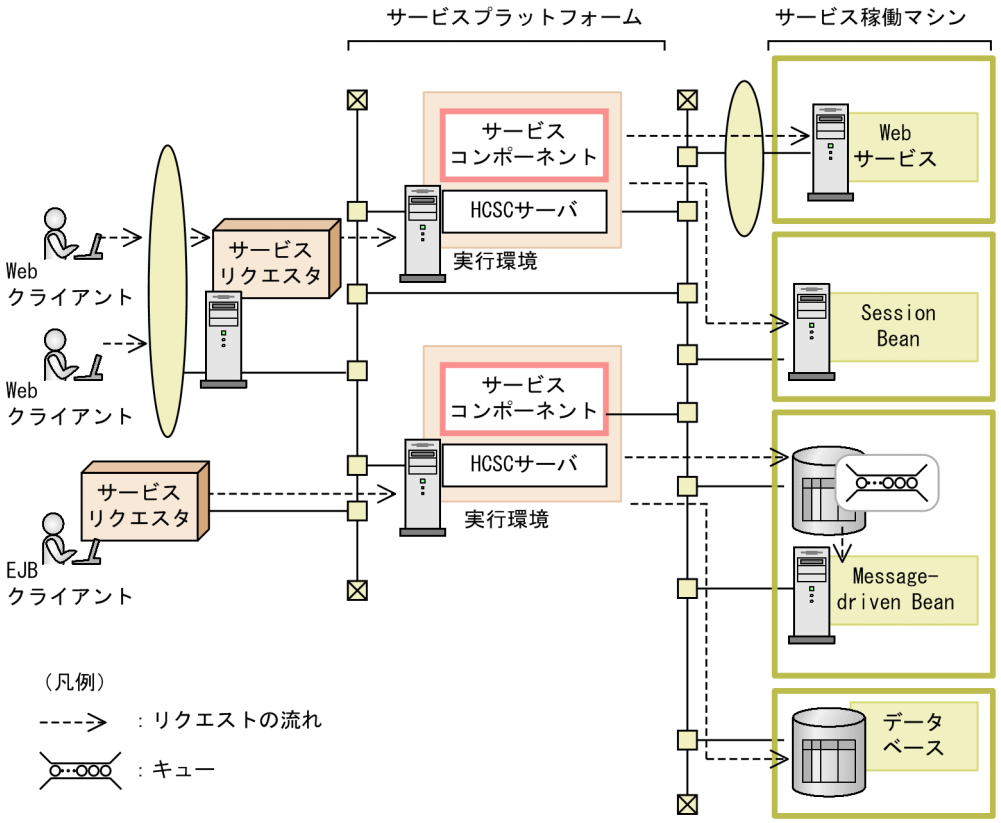
- Web サービス（SOAP 通信）のサービス部品を利用するための SOAP アダプタ
- EJB（Stateless Session Bean または Stateful Session Bean）で作成されたサービス部品を利用するための SessionBean アダプタ
- WS-R（WS-Reliability）を使用して非同期の MDB（Message Driven Bean）のサービス部品を利用するための MDB（WS-R）アダプタ
- DB キューを使用して TP1/EE の非同期のサービス部品を利用するための MDB（DB キュー）アダプタ
- データベースをサービス部品として利用するための DB アダプタ
- OpenTP1 や XDM/DCCM3 上のシステムを利用するための TP1 アダプタ
- ファイルを介した業務処理システムを利用するためのファイルアダプタ
- メインフレームなどの基幹システムを利用するための Object Access アダプタ
- 既存のメッセージキュー（IBM MQ システム）に対してメッセージの送受信をするための Message Queue アダプタ
- FTP サーバと接続してファイル転送をするための FTP アダプタ
- ファイルのフォーマット変換，複製，削除などをするためのファイル操作アダプタ
- メールサーバと接続して，メール送信をするためのメールアダプタ
- RESTful Web サービス（JAX-RS エンジンを利用した Web サービス）を利用するための HTTP アダプタ
- オペレーティングシステム上で動作する外部コマンドを起動し，実行結果を取得するためのコマンドアダプタ
- サービスプラットフォームと SFTP サーバとの間で SFTP プロトコルによるファイル転送をするための SFTP アダプタ
- Apache Kafka と連携してメッセージを送信するための Kafka アダプタ
- gRPC で作成されたサービスへメッセージを送信するための gRPC アダプタ

8.3　サービスプラットフォームを利用したリクエストの流れ

サービスプラットフォームは，サービスリクエスタから受け付けたリクエストを，サービス内容やプロトコル種別に応じて各サービス稼働マシンに送信します。サービス稼働マシンで処理が実行された結果は，サービスプラットフォーム経由でサービスリクエスタに返却されます。なお，サービスリクエスタがビジネスプロセスを呼び出した場合は，サービス稼働マシンで動作する複数のサービスがビジネスプロセス定義に従って呼び出されます。

サービスプラットフォームで構築した実行環境でのリクエストの流れを次の図に示します。

図 8-5　サービスプラットフォームで構築した実行環境でのリクエストの流れ



1 つの実行環境にリクエストが集中するのを防ぐために，実行環境を冗長構成にして，リクエストを負荷分散することもできます。負荷分散は，利用するサービスが Web サービスまたは Session Bean の場合に実現できます。この場合，負荷分散には，サービスの形態に応じて，負荷分散機（Web サービスの場合），またはアプリケーションサーバの実行環境の機能である CTM（Session Bean の場合）を使用します。サービスプラットフォームでのリクエストの負荷分散については，マニュアル「サービスプラットフォーム 解説」の「1.4.1　ロードバランス機能を利用した HCSC サーバの冗長構成」を参照してください。

9 サービスプラットフォームの機能

この章では，サービスプラットフォームで SOA を適用したシステムを実現するための機能の概要について説明します。

なお，この章で説明している機能の詳細については，マニュアル「サービスプラットフォーム 解説」の「1.　サービスプラットフォームの機能概要」を参照してください。

9.1　サービスプラットフォームの機能概要

サービスプラットフォームは SOA を実現するための基盤製品であり，主に次の機能を提供しています。

●サービス部品呼び出し機能

業務・機能を「サービス」として管理し，要求に対して適切なサービス部品を呼び出して実行します。

●ビジネスプロセス実行機能

サービス呼び出しの流れを制御するためのビジネスプロセスの開発および実行を行います。

●データ変換機能

電文フォーマットの違いを吸収するためのデータ変換機能を提供します。

●各種受付

さまざまな種類のサービスリクエスタから実行要求を受け付けるための受付を提供します。

●各種サービスアダプタ

さまざまな種類のサービスを呼び出すためのサービスアダプタを提供します。

●実行履歴管理機能

ビジネスプロセスの実行履歴をデータベースに永続化して管理し，エラー発生時のリトライやビジネスプロセスの可視化を行うことができます。

それぞれの機能の内容を次に説明します。

9.2　サービス部品呼び出し機能

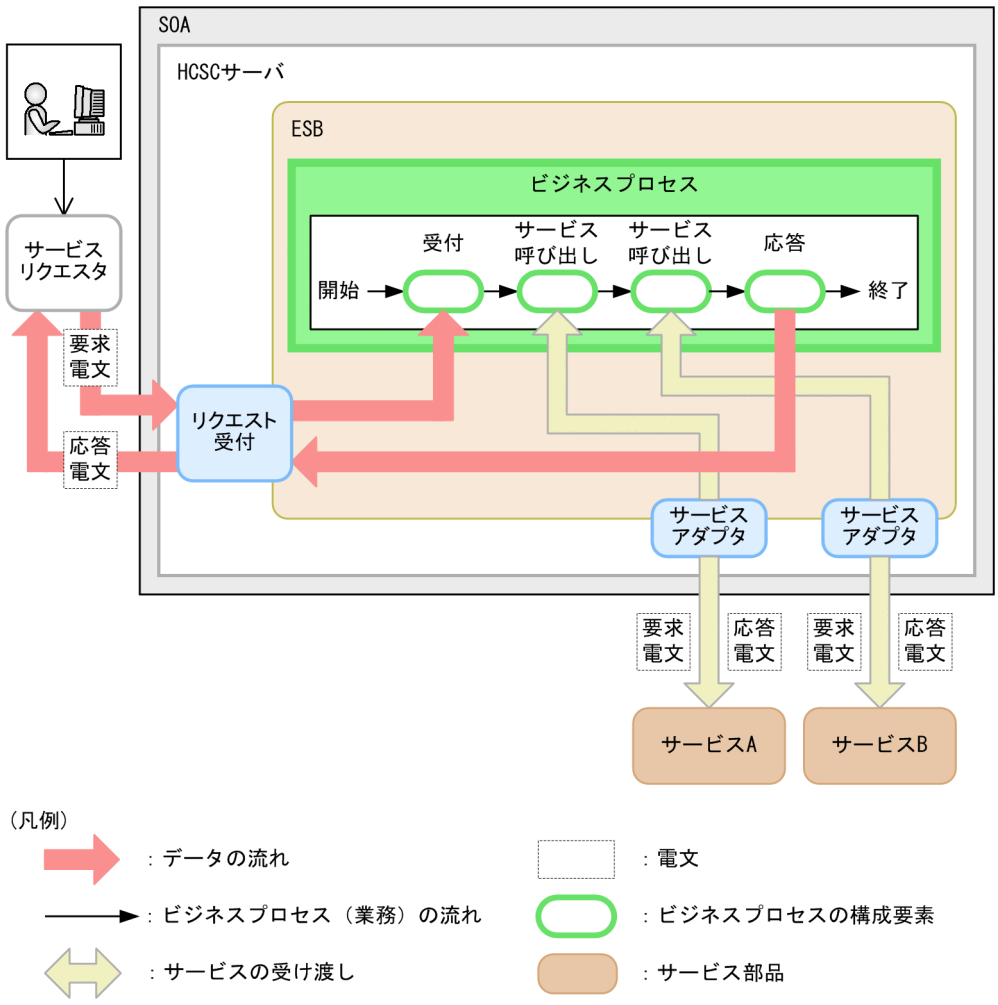
サービスプラットフォームでは，エンタープライズサービスバス（ESB：Enterprise Service Bus）を介して，サービスの利用者，ビジネスプロセス，およびサービスの間の連携を実現します。

SOA を適用したシステムでは，要求を行う利用者側のシステムを「サービスリクエスタ」と呼び，機能を提供する提供者側を「サービス部品」と呼びます。また，ESB やビジネスプロセス，およびサービスアダプタを実現する SOA 環境を「HCSC サーバ」と呼びます。

利用者がサービス部品の実行要求をすると，SOA は HCSC サーバに配備したビジネスプロセスやサービスアダプタの中から，適切なサービス部品を呼び出して実行します。

サービス部品を呼び出す流れを次の図に示します。

図 9-1　サービス部品を呼び出す流れ



ビジネスプロセスについては「9.3　ビジネスプロセス実行機能」を，サービスアダプタの種類については「9.6　サービスアダプタの種類」を参照してください。

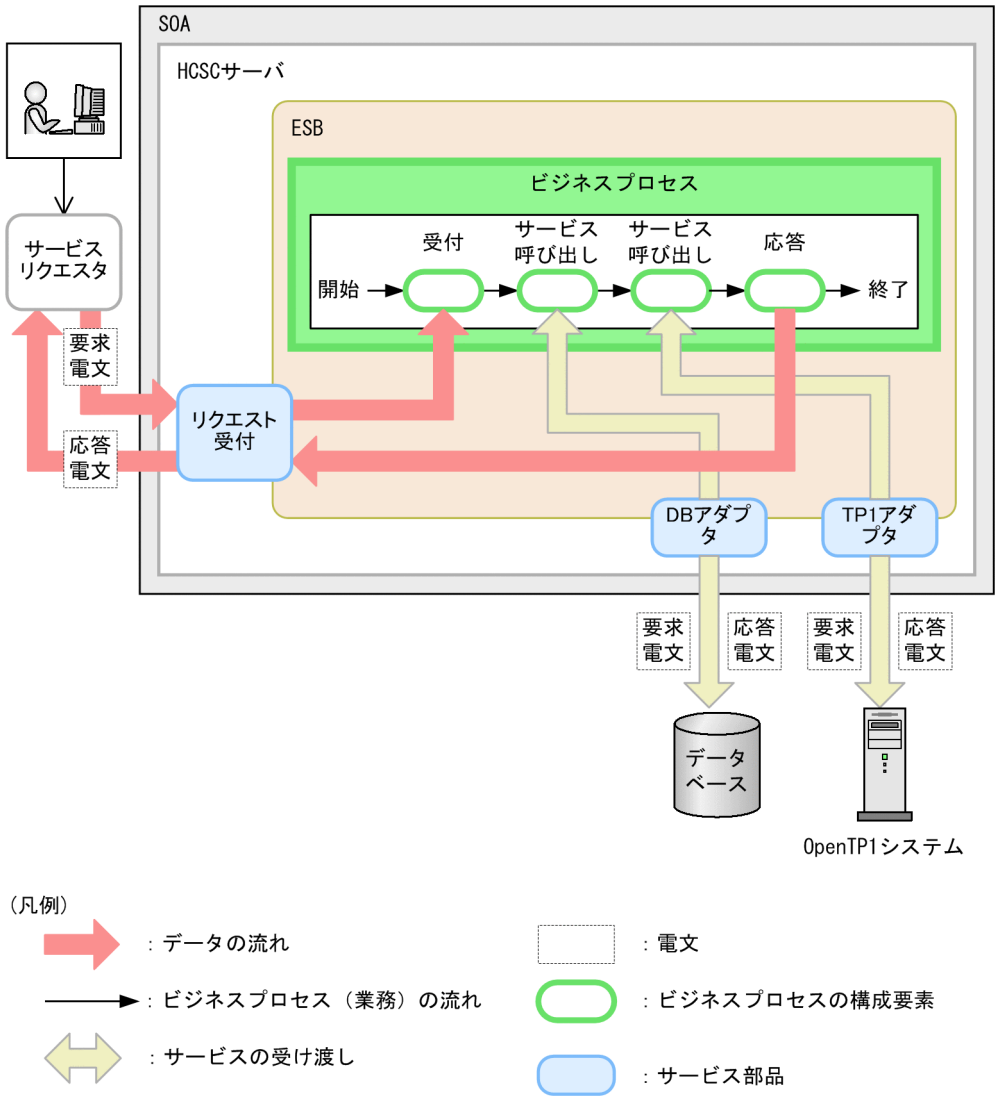
9.3　ビジネスプロセス実行機能

ビジネスプロセスとは，複数のサービス部品の処理の順番や条件などを定義して，一連の業務の流れとして定義したものです。

サービスプラットフォームでは，ビジネスプロセスに定義されている順に，連続してサービス部品を呼び出して実行できます。

ビジネスプロセスからサービス部品を実行する場合の概略を次の図に示します。

図 9-2　ビジネスプロセスからのサービス部品実行



HCSC サーバへサービスの実行要求（要求電文）を送信するアプリケーションであるサービスリクエスタは，サービス部品の実行要求を受け付けると，ビジネスプロセスへ要求電文を送信します。

ビジネスプロセスは，各サービスアダプタへ順に情報を送信し，サービスアダプタを介してサービス部品を実行します。

1 つのサービスの実行が完了したら，次のサービスの実行に移り，ビジネスプロセスが終了するまで実行されます。

各サービス部品での処理が終了すると，処理結果として応答電文がサービスアダプタに送信されます。応答電文は，サービスリクエスタを経由して業務担当者へ通知されます。

ビジネスプロセスには，[受付]，[応答]，[サービス呼び出し]などのビジネスプロセスの基本要素や，[分岐処理]，[並列処理]，[繰り返し処理]などのビジネスプロセスの処理構造を定義します。

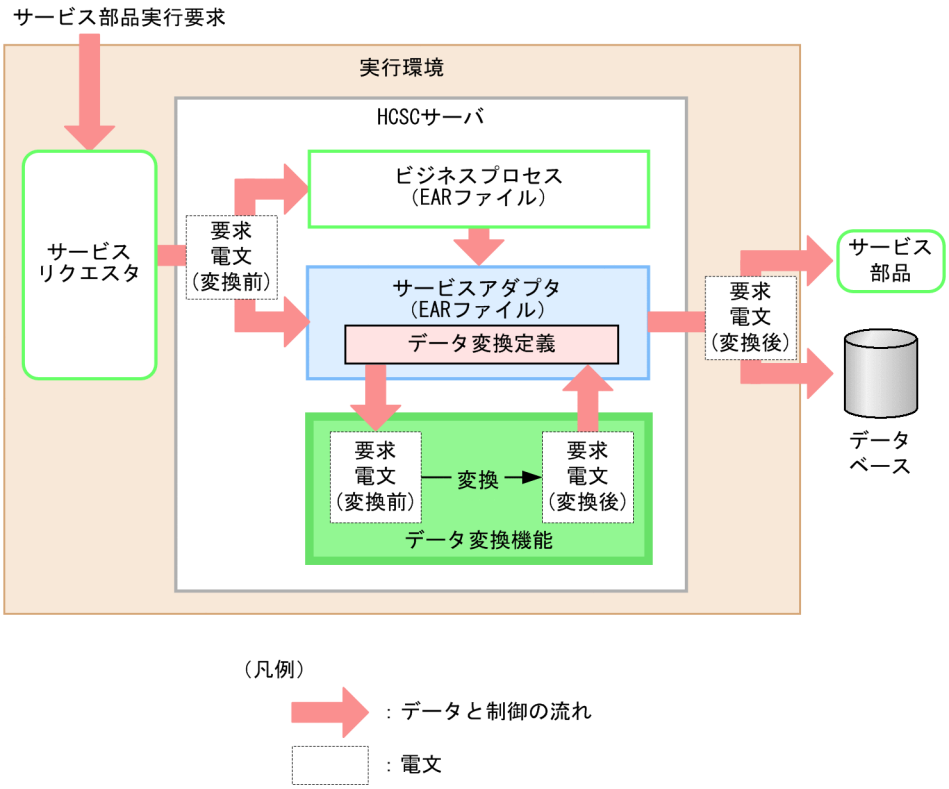
なお，ビジネスプロセスは，コーディングによって定義するのではなく，Eclipse で統合された GUI ツールを使用して定義します。

9.4　データ変換機能

サービスプラットフォームでは，SOA を適用したシステムでの業務を円滑にするために，XML とバイナリとのデータ変換ができます。データ形式の違いを気にすることなく，既存システムとのシームレスなデータ連携ができます。

データ変換とは，実行環境でサービスリクエスタからサービスアダプタに送信される要求電文と，サービスアダプタからサービス部品に送信される要求電文のフォーマットが異なる場合に，適切なフォーマットに変換する機能です。実行環境でのデータ変換機能の概要を次の図に示します。

図 9-3　データ変換機能の概要



HCSC サーバへサービスの実行要求（要求電文）を送信するアプリケーションであるサービスリクエスタは，サービス部品の実行要求を受け付けると，要求電文をサービスアダプタまたはビジネスプロセスに送信します。

サービスアダプタ（ビジネスプロセスから呼び出されるサービスアダプタも含む）にデータ変換定義が含まれる場合，要求電文は，データ変換機能で変換されます。

データ変換は，データ変換定義に設定されている内容に従って実行されます。サービスアダプタは，変換された要求電文を実行するサービス部品に送信します。

なお，サービス部品の実行後，業務担当者に実行結果などの応答を返す場合は，実行要求時とは逆のデータ変換が実行されます。

9.5　受付の種類

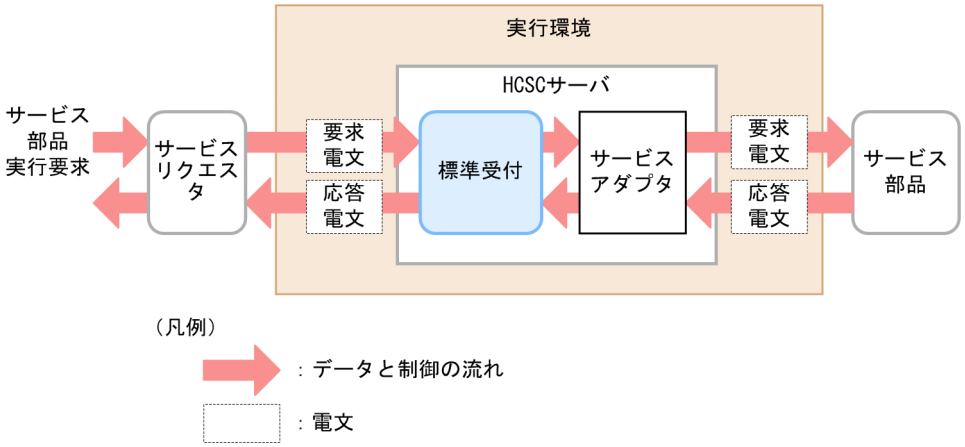
サービスプラットフォームでは，受付でサービスリクエスタからの要求電文を受け付けます。受付には，HCSC サーバに含まれる標準受付と，ユーザが任意のインターフェースを定義できるユーザ定義受付があります。

9.5.1　標準受付

標準受付は，HCSC サーバに含まれる機能です。標準受付には，同期受付（Web サービス／SessionBean）と，非同期受付（MDB（WS-R）／MDB（DB キュー））があります。標準受付を開始すると，サービスリクエスタからの要求電文を受け付けられる状態になります。

サービスアダプタからの要求電文を標準受付で受け付けてサービス部品を実行する場合の概略を次の図に示します。

図 9-4　要求電文を受け付けてサービス部品を実行するときの流れ（標準受付の場合）



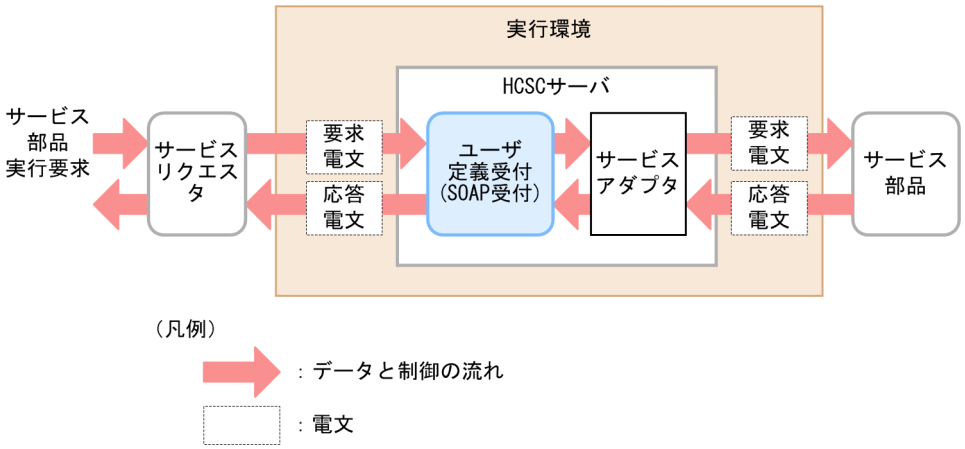
標準受付は，ほかの受付と異なりユーザが独自に定義する必要がありません。それぞれの標準受付の形式に合わせたサービスリクエスタを作成すれば，そのまま使用できます。ただし，既存のサービスリクエスタを再利用する場合，標準受付のインターフェースの形式と合わないときは，サービスリクエスタの修正や再作成が必要です。

9.5.2　ユーザ定義受付（SOAP 受付）

サービスプラットフォームでは，Web サービス（SOAP 通信）を使用したシステムでサービス部品の実行要求をする場合，任意のインターフェースを定義したユーザ定義受付（SOAP 受付）を使用できます。

サービスアダプタからの要求電文を SOAP 受付で受け付けてサービス部品を実行する場合の概略を次の図に示します。

図 9-5　SOAP 受付を使用したサービス部品の実行

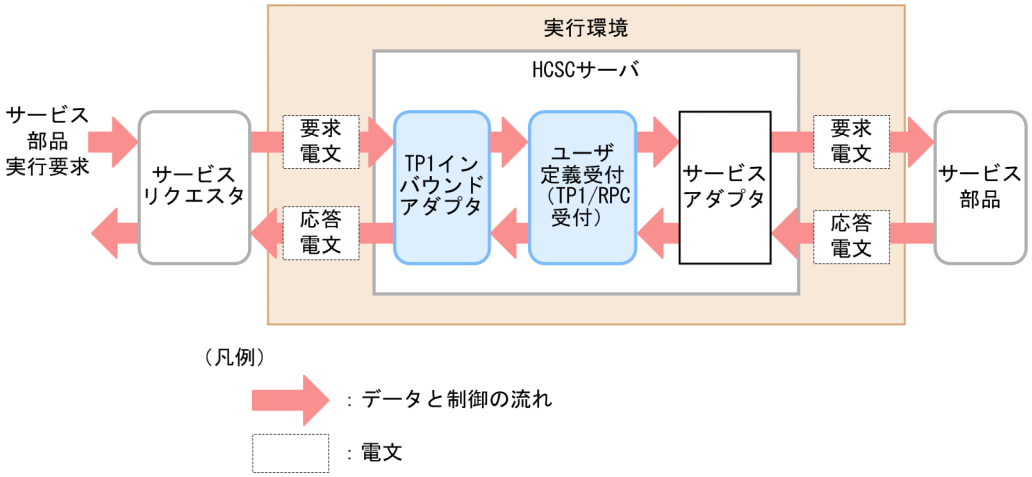


9.5.3　ユーザ定義受付（TP1/RPC 受付）

サービスプラットフォームでは，既存の OpenTP1 システムからサービス部品の実行要求をする場合，OpenTP1 のインターフェースに合わせて定義したユーザ定義受付（TP1/RPC 受付）を使用します。

TP1/RPC 受付を使用して既存の OpenTP1 システムからサービス部品を呼び出して実行する場合の概略を次の図に示します。

図 9-6　TP1/RPC 受付を使用したサービス部品の実行

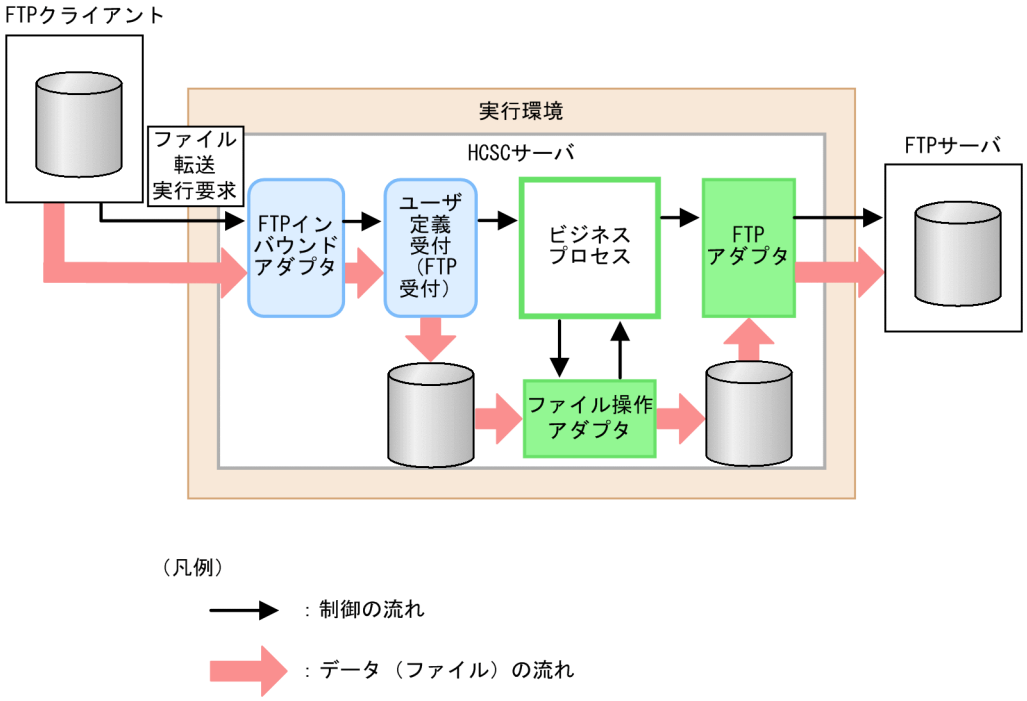


TP1/RPC 受付は，サービスリクエスタから TP1 インバウンドアダプタを経由して要求電文を受け付けます。サービス部品の実行要求を受け付けた TP1/RPC 受付は，受付内で定義されている処理（データ変換）を実行したあと，サービス部品へ要求電文を送信し，サービス部品を実行します。

9.5.4　ユーザ定義受付（FTP 受付）

サービスプラットフォームでは，FTP 受付を利用して，FTP クライアントから転送されたファイルを受信したり，FTP クライアントと FTP サーバの間のファイル転送を中継したりすることができます。ファイル転送を中継する場合の概略を，次の図に示します。

図 9-7　FTP 受付を使用した FTP クライアントと FTP サーバの間のファイル転送



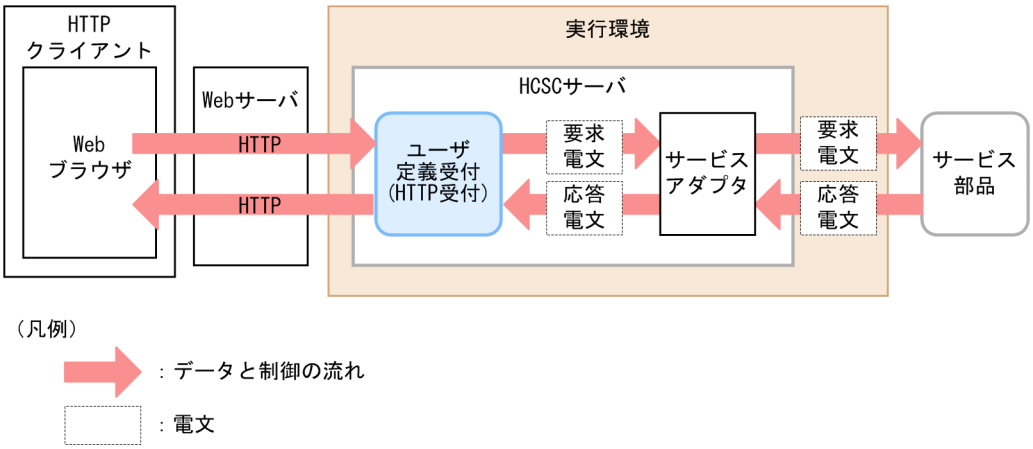
FTP 受付は，FTP クライアントから FTP インバウンドアダプタを経由して接続要求を受け付け，ビジネスプロセスを呼び出します。ビジネスプロセスから呼び出される FTP アダプタやファイル操作アダプタと連携して動作することによって，FTP クライアントと FTP サーバ間でファイル転送をします。

9.5.5　ユーザ定義受付（HTTP 受付）

サービスプラットフォームでは，HTTP 受付を利用することで，携帯端末や Web ブラウザから接続要求を出す場合に，Web フロントシステムや SOAP 受付を経由しないで直接ビジネスプロセスを呼び出せるようになります。

HTTP 受付を使用して HTTP クライアントから HTTP 通信でサービス部品を呼び出して実行する場合の概略を次の図に示します。

図 9-8　HTTP 受付を使用したサービス部品の実行



HTTP クライアント上の Web ブラウザから発信された HTTP リクエストは，Web サーバを経由して HTTP 受付で受け付けられます。そのあと，HTTP リクエストは要求電文（サービス部品呼び出し要求）に変換され，サービスアダプタを介してサービス部品を呼び出します。

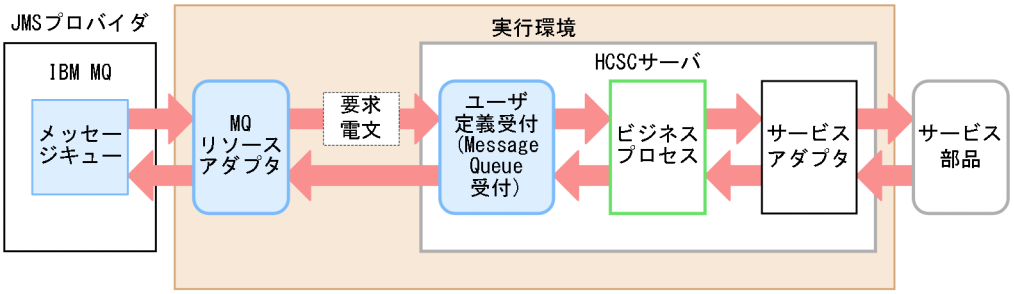
応答時は，サービスアダプタから渡された応答電文が HTTP 受付で HTTP レスポンスの形式に変換され，Web ブラウザに応答を返します。

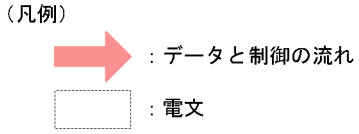
9.5.6　ユーザ定義受付（Message Queue 受付）

サービスプラットフォームでは，Message Queue 受付を利用することで，メッセージキュー（IBM MQ システム）から MQ リソースアダプタ経由でビジネスプロセスを呼び出し，各種サービス部品を実行できます。

Message Queue 受付を使用してメッセージキューからサービス部品を呼び出して実行する場合の概略を次の図に示します。

図 9-9　Message Queue 受付を使用したサービス部品の実行





Message Queue 受付は，MQ リソースアダプタを経由して要求電文を受け付けます。サービス部品の実行要求を受け付けた Message Queue 受付は，ビジネスプロセスを呼び出し，サービスアダプタを介して各種サービス部品を実行します。

要求電文の受け取りは，MDB（Message-Driven Bean）を使用します。

Message Queue 受付からは，応答電文を返しません。

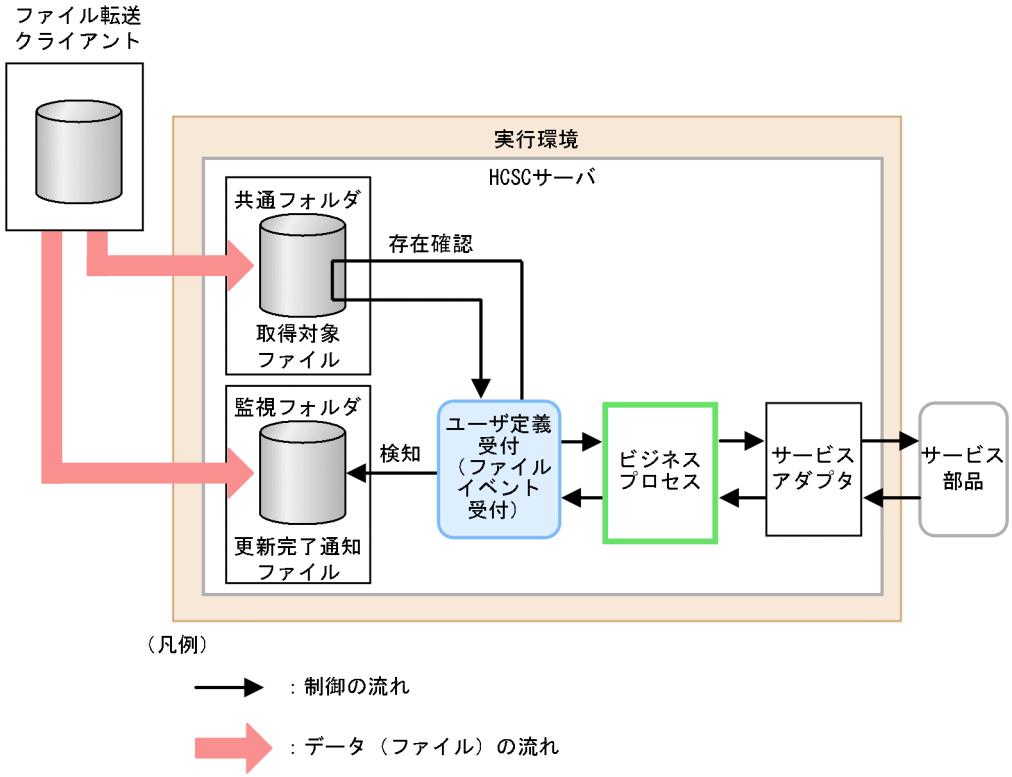
9.5.7　ユーザ定義受付（ファイルイベント受付）

サービスプラットフォームでは，ファイルイベント受付を利用することで，ファイルが更新されたことを検知して，ビジネスプロセスを呼び出すことができます。

ビジネスプロセス内で，ファイル操作アダプタを利用してファイル変換したり，FTP アダプタまたは HTTP アダプタを利用して一般的なプロトコルとファイル連携したりできます。

ファイルイベント受付を使用して，ビジネスプロセスを呼び出す場合の概略を次の図に示します。

図 9-10　ファイルイベント受付を使用したサービス部品の実行

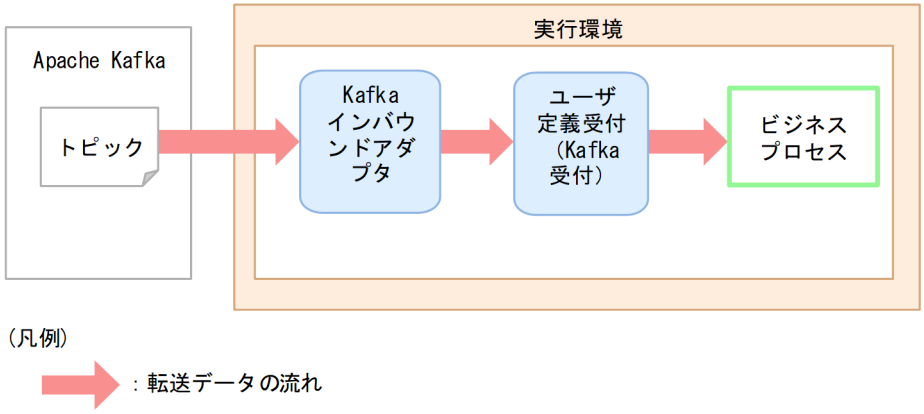


9.5.8　ユーザ定義受付（Kafka 受付）

サービスプラットフォームでは，Kafka 受付を利用することで，Apache Kafka から Kafka インバウンドアダプタを経由してメッセージを取得し，ビジネスプロセスに渡すことができます。

Kafka 受付を使用して Apache Kafka からメッセージを取得し，ビジネスプロセスに渡す場合の概略を次の図に示します。

図 9-11　Kafka 受付を使用したメッセージの処理



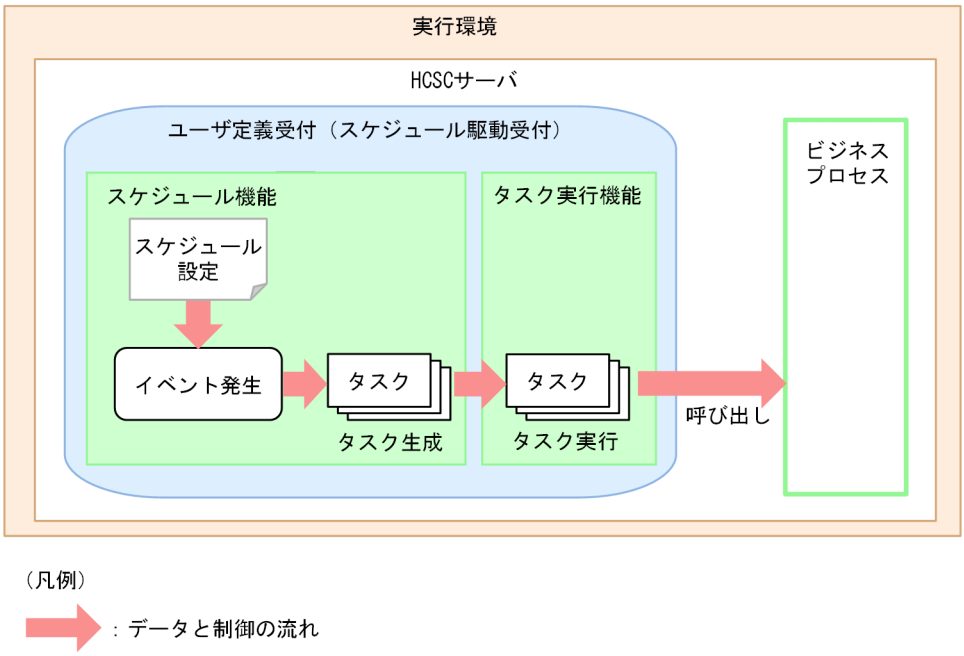
Kafka 受付は，Kafka インバウンドアダプタを経由して，Apache Kafka からメッセージを取得します。取得したメッセージは，ビジネスプロセス内で，データ変換アクティビティなどを使用して編集できます。

9.5.9　ユーザ定義受付（スケジュール駆動受付）

サービスプラットフォームでは，スケジュール駆動受付を利用することで，スケジュール設定に定義した間隔でタスクを実行し，ビジネスプロセスを呼び出すことができます。

スケジュール駆動受付を使用してビジネスプロセスを呼び出す場合の概略を次の図に示します。

図 9-12　スケジュール駆動受付を使用したビジネスプロセスの呼び出し



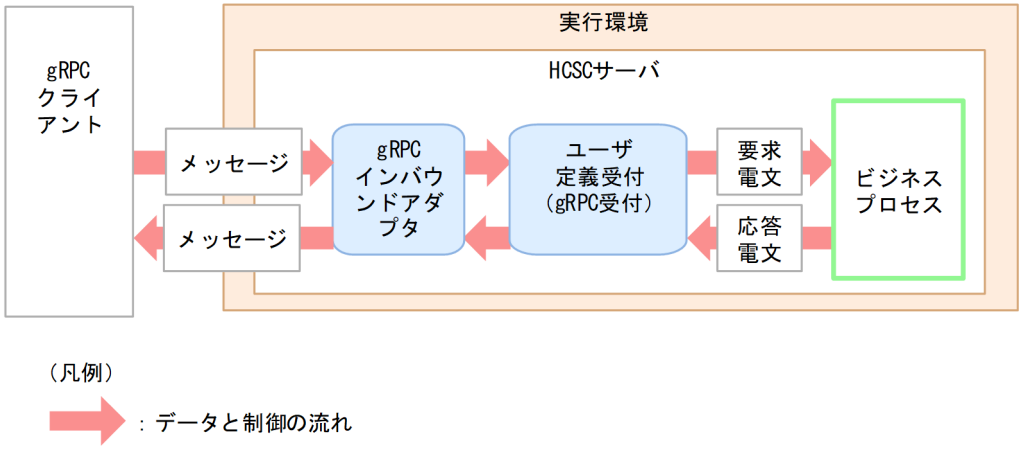
スケジュール駆動受付のスケジュール設定に，あらかじめイベントを発生させる間隔を定義しておきます。定義したスケジュールでイベントが発生すると，イベント発生時にタスクが生成されます。生成されたタスクはタスク実行機能によって実行され，タスクの実行に合わせてビジネスプロセスが呼び出されます。

9.5.10　ユーザ定義受付（gRPC 受付）

サービスプラットフォームでは，gRPC 受付を利用することで，gRPC クライアントから gRPC インバウンドアダプタを経由してメッセージを取得し，ビジネスプロセスに渡すことができます。

gRPC 受付を使用して gRPC クライアントからメッセージを取得し，ビジネスプロセスに渡す場合の概略を次の図に示します。

図 9-13　gRPC 受付を使用したメッセージの処理



gRPC 受付は，gRPC インバウンドアダプタを経由して，gRPC クライアントからメッセージを取得します。メッセージは，gRPC 受付で要求電文に変換され，ビジネスプロセスに渡されます。要求電文は，ビジネスプロセス内でデータ変換アクティビティなどを使用して編集できます。

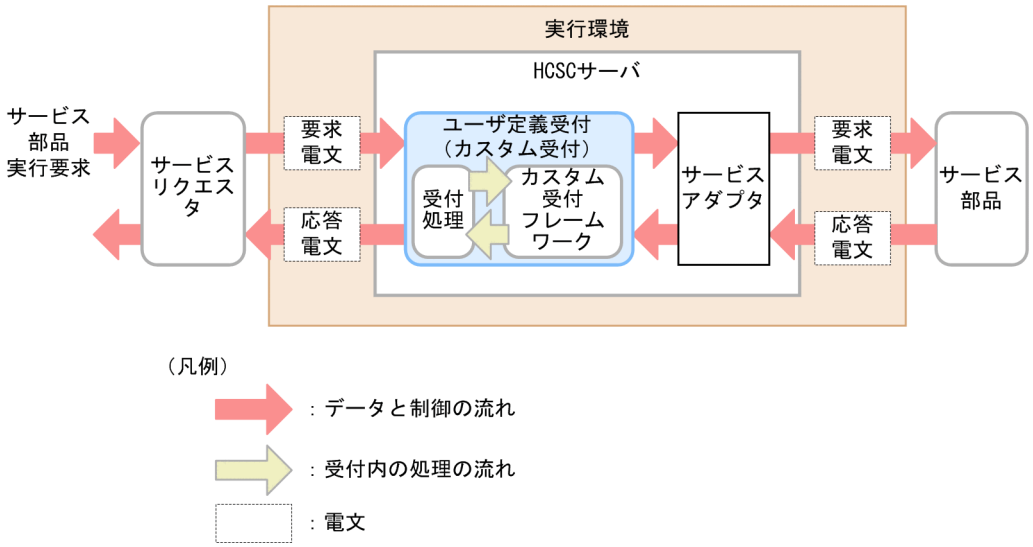
9.5.11　ユーザ定義受付（カスタム受付）

サービスプラットフォームでは，任意のシステムのサービス部品を実行するために，カスタム受付フレームワークを提供しています。

提供しているカスタム受付フレームワークと開発者が作成する受付処理を組み合わせたカスタム受付から，任意のサービス部品を呼び出して実行できます。

カスタム受付からサービス部品を実行する場合の概略を次の図に示します。

図 9-14　カスタム受付を使用したサービス部品の実行



受付処理は，サービスリクエスタから要求電文を受け付けます。受け付けた要求電文は，カスタム受付フレームワークに渡され，処理（データ変換）を実行したあと，サービス部品へ要求電文を送信し，サービス部品を実行します。

9.6　サービスアダプタの種類

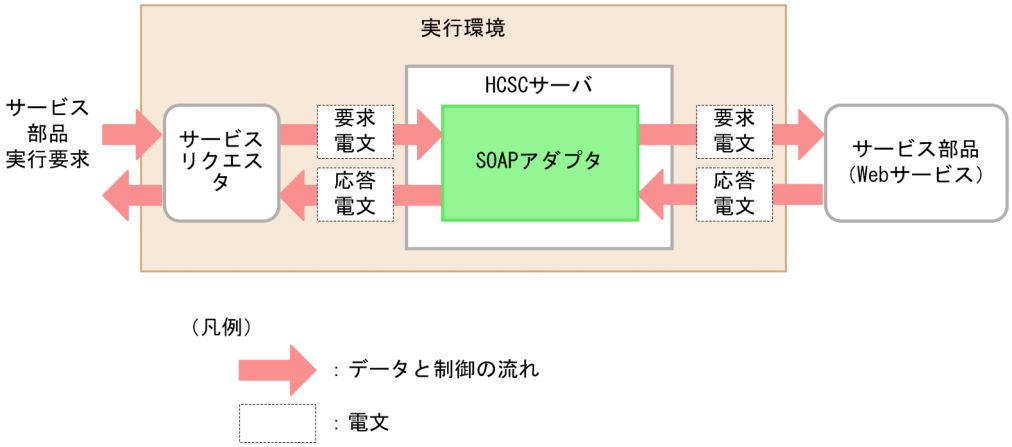
サービスプラットフォームでは，利用できるサービスとして SOAP 通信を使用した Web サービス，SessionBean，および MDB などの業界標準技術を利用したサービスアダプタを提供しています。また，メインフレームシステムやオンラインシステムなどの既存システムと接続するための各種サービスアダプタも用意しています。

9.6.1　SOAP アダプタ

サービスプラットフォームでは，Web サービス（SOAP 通信）のサービス部品を呼び出すための SOAP アダプタを提供しています。

SOAP アダプタからサービス部品を呼び出して実行する場合の概略を次の図に示します。

図 9-15　SOAP アダプタからのサービス部品実行



HCSC サーバへサービスの実行要求（要求電文）を送信するアプリケーションであるサービスリクエスタは，サービス部品（Web サービス）の実行要求を受け付けると，SOAP アダプタへ要求電文を送信します。

SOAP アダプタは，アダプタ内で定義されているサービス部品へ要求電文を送信し，サービス部品を実行します。

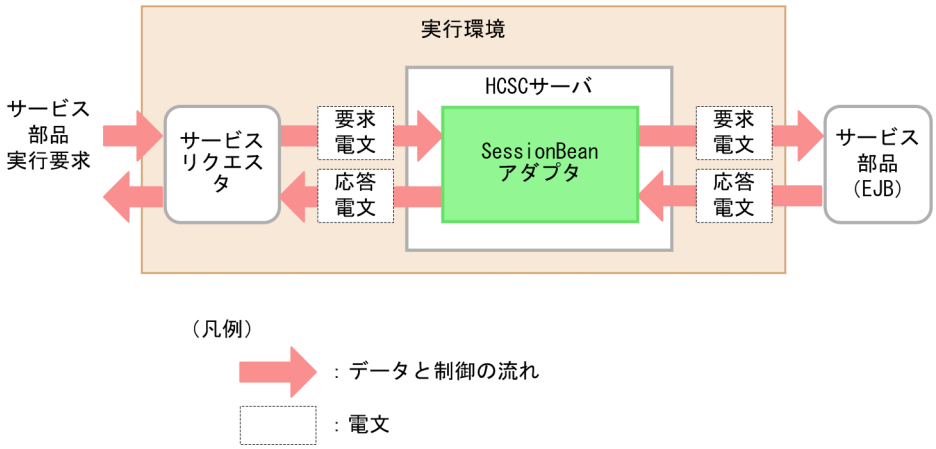
サービス部品での処理が終了すると，処理結果として応答電文が SOAP アダプタに送信されます。応答電文は，サービスリクエスタを経由して業務担当者へ通知されます。

9.6.2　SessionBean アダプタ

サービスプラットフォームでは，EJB（Stateless Session Bean または Stateful Session Bean）で作成されたサービス部品を呼び出すための SessionBean アダプタを提供しています。

SessionBean アダプタからサービス部品を呼び出して実行する場合の概略を次の図に示します。

図 9-16　SessionBean アダプタからのサービス部品実行



HCSC サーバへサービスの実行要求（要求電文）を送信するアプリケーションであるサービスリクエスタは，サービス部品（EJB）の実行要求を受け付けると，SessionBean アダプタへ要求電文を送信します。

SessionBean アダプタは，アダプタ内で定義されているサービス部品へ要求電文を送信し，サービス部品を実行します。

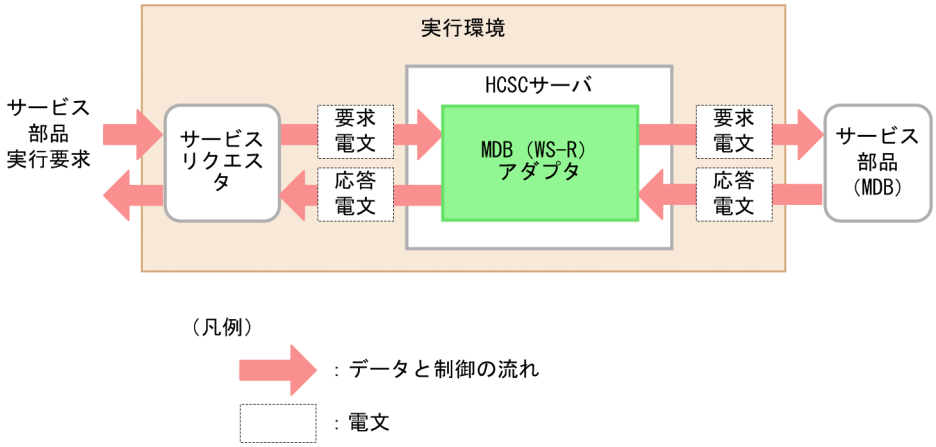
サービス部品での処理が終了すると，処理結果として応答電文が SessionBean アダプタに送信されます。応答電文は，サービスリクエスタを経由して業務担当者へ通知されます。

9.6.3　MDB（WS-R）アダプタ

サービスプラットフォームでは，WS-R（WS-Reliability）を使用して非同期の MDB（Message Driven Bean）のサービス部品を呼び出すための MDB（WS-R）アダプタを提供しています。

MDB（WS-R）アダプタからサービス部品を呼び出して実行する場合の概略を次の図に示します。

図 9-17　MDB（WS-R）アダプタからのサービス部品実行



HCSC サーバへサービスの実行要求（要求電文）を送信するアプリケーションであるサービスリクエスタは，サービス部品（MDB）の実行要求を受け付けると，MDB（WS-R）アダプタへ要求電文を送信します。

MDB（WS-R）アダプタは，アダプタ内で定義されているサービス部品へ要求電文を送信し，サービス部品を実行します。

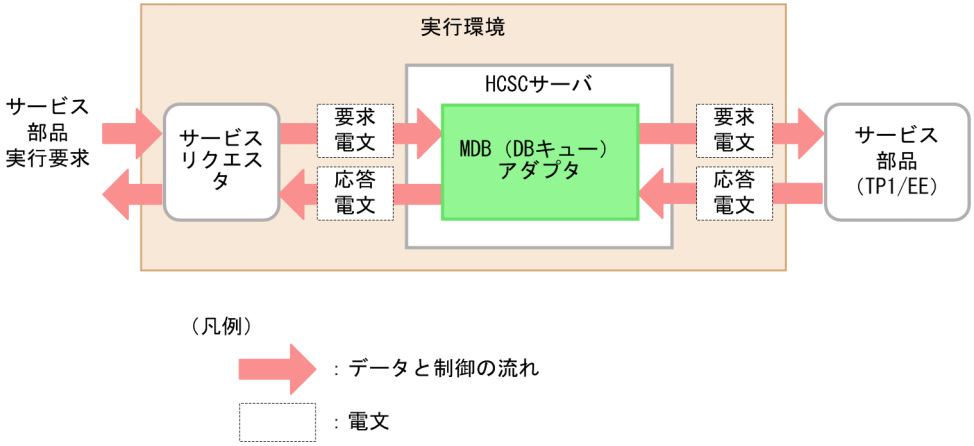
サービス部品での処理が終了すると，処理結果として応答電文が MDB（WS-R）アダプタに送信されます。応答電文は，サービスリクエスタを経由して業務担当者へ通知されます。

9.6.4　MDB（DB キュー）アダプタ

サービスプラットフォームでは，DB キューを使用して TP1/EE の非同期のサービス部品を呼び出すための MDB（DB キュー）アダプタを提供しています。

MDB（DB キュー）アダプタからサービス部品を呼び出して実行する場合の概略を次の図に示します。

図 9-18　MDB（DB キュー）アダプタからのサービス部品実行



HCSC サーバへサービスの実行要求（要求電文）を送信するアプリケーションであるサービスリクエスタは，サービス部品（TP1/EE）の実行要求を受け付けると，MDB（DB キュー）アダプタへ要求電文を送信します。

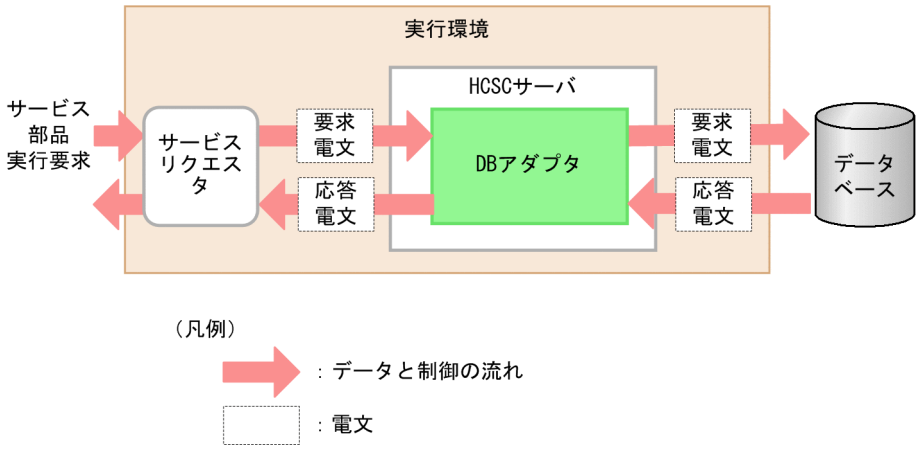
MDB（DB キュー）アダプタは，アダプタ内で定義されているサービス部品へ要求電文を送信し，サービス部品を実行します。

サービス部品での処理が終了すると，処理結果として応答電文が MDB（DB キュー）アダプタに送信されます。応答電文は，サービスリクエスタを経由して業務担当者へ通知されます。

9.6.5　DB アダプタ

サービスプラットフォームでは，DB アダプタからデータベースに対して SQL を実行できます。DB アダプタからデータベースへの SQL の実行要求の概略を次の図に示します。

図 9-19　DB アダプタからの SQL 実行



HCSC サーバへサービスの実行要求（要求電文）を送信するアプリケーションであるサービスリクエスタは，サービス部品の実行要求を受け付けると，DB アダプタへ要求電文を送信します。

DB アダプタは，アダプタ内で定義されているデータベースへ SQL の実行を要求する電文を送信し，SQL を実行します。

SQL 実行が終了すると，実行結果として応答電文が DB アダプタに送信されます。応答電文は，サービスリクエスタを経由して業務担当者へ通知されます。

注意事項

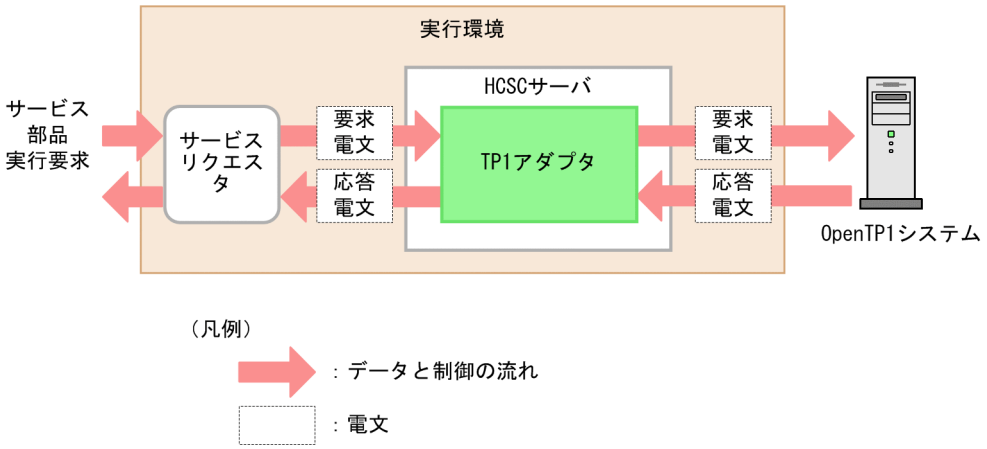
DB アダプタを利用してサービス部品（データベースの操作）を実行する場合，実行対象として利用できるデータベースは次に示すものだけです。

- HiRDB
- Oracle

9.6.6　TP1 アダプタ

サービスプラットフォームでは，TP1 アダプタから OpenTP1 を使用したシステムのサービス部品を呼び出して実行できます。TP1 アダプタからサービス部品を実行する場合の概略を次の図に示します。

図 9-20　TP1 アダプタからのサービス部品実行



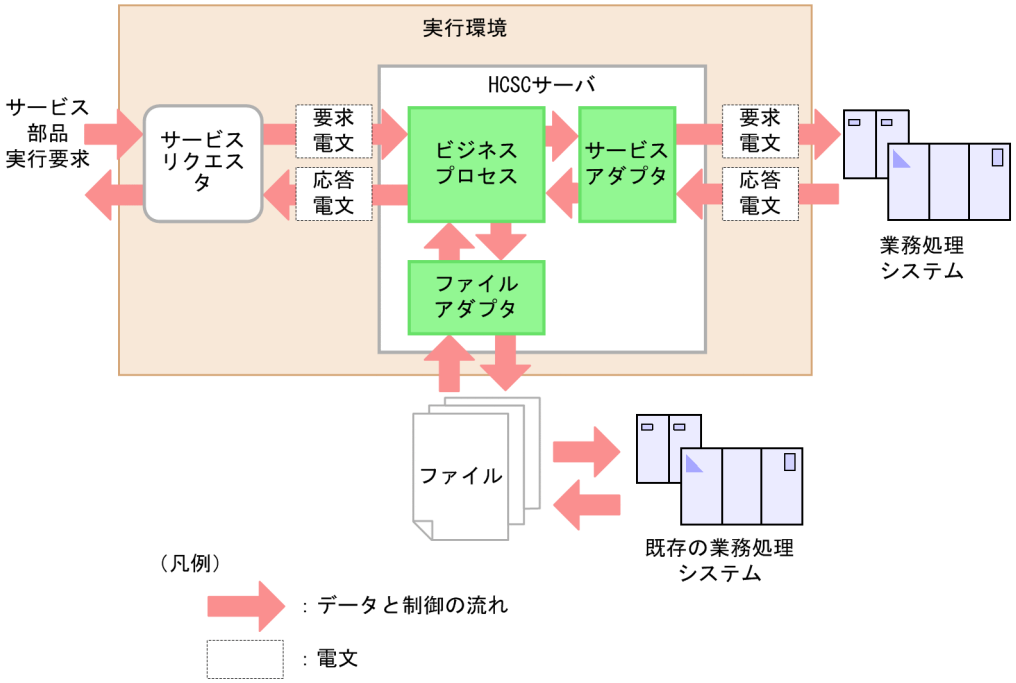
HCSC サーバへサービスの実行要求（要求電文）を送信するアプリケーションであるサービスリクエスタは，サービス部品の実行要求を受け付けると，TP1 アダプタへ要求電文を送信します。

TP1 アダプタは，アダプタ内で定義されている OpenTP1 システムのサービス部品へ要求電文を送信し，サービス部品を実行します。

9.6.7　ファイルアダプタ

サービスプラットフォームでは，ファイルアダプタから既存の業務処理システムのファイルを入出力できます。ファイルアダプタからファイルを入出力する場合の概略を次の図に示します。

図 9-21　ファイルアダプタからのファイルの入出力



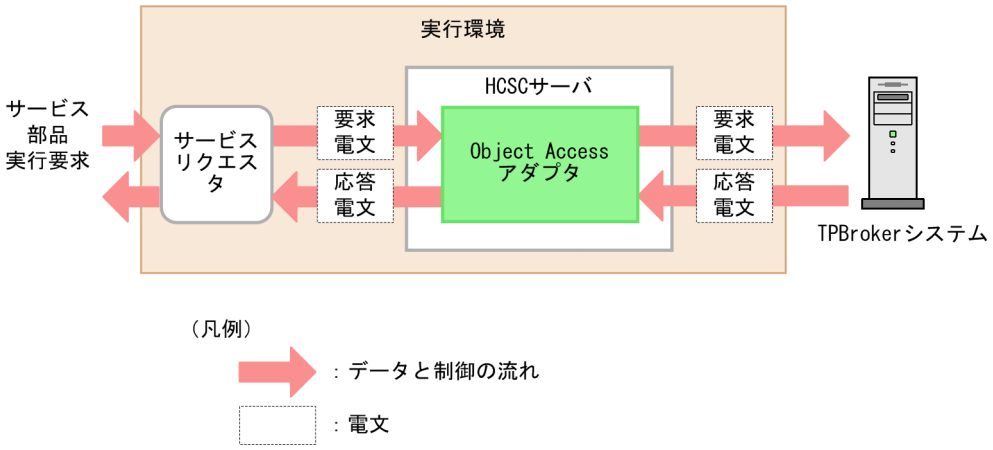
既存の業務処理システムのファイルを利用する場合は，ビジネスプロセスからファイルアダプタを呼び出して，ファイルのデータを読み込むことができます。

また，サービスの実行結果を既存の業務処理システムに渡したい場合も，ビジネスプロセスからファイルアダプタを呼び出して，ファイルのデータを受け渡します。

9.6.8　Object Access アダプタ

サービスプラットフォームでは，Object Access アダプタから既存の TPBroker システム（Object Wrapper システム）のサービス部品を呼び出して実行できます。Object Access アダプタからサービス部品を実行する場合の概略を次の図に示します。

図 9-22　Object Access アダプタからのサービス部品実行



HCSC サーバへサービスの実行要求（要求電文）を送信するアプリケーションであるサービスリクエスタは，サービス部品の実行要求を受け付けると，Object Access アダプタへ要求電文を送信します。

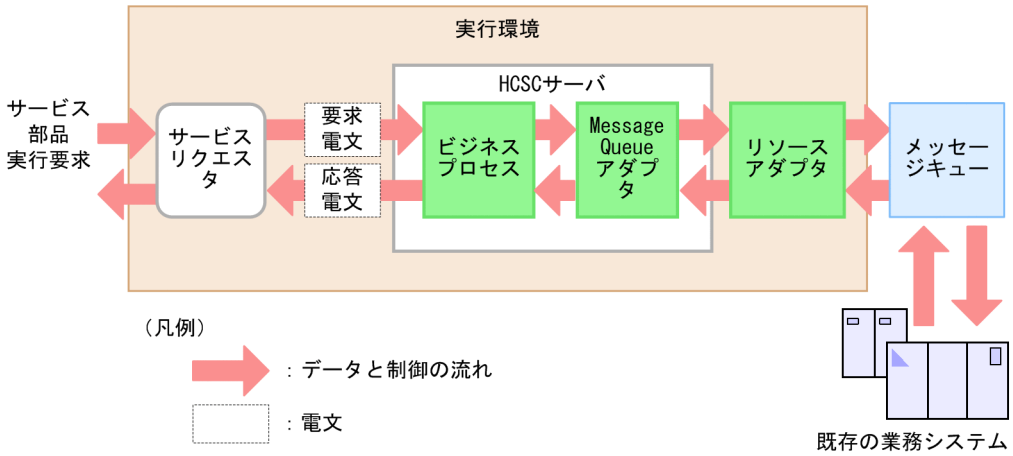
Object Access アダプタは，アダプタ内で定義されている TPBroker システムのサービス部品へ要求電文を送信し，サービス部品を実行します。

これによって，既存の TPBroker システム（Object Wrapper システム）で動作する CORBA サーバの業務メソッドをサービス部品として連携できます。

9.6.9　Message Queue アダプタ

サービスプラットフォームでは，Message Queue アダプタから既存のメッセージキュー（IBM MQ システム）に対してメッセージを送受信できます。Message Queue アダプタからメッセージを送受信する場合の概略を次の図に示します。

図 9-23　Message Queue アダプタからのメッセージキュー制御



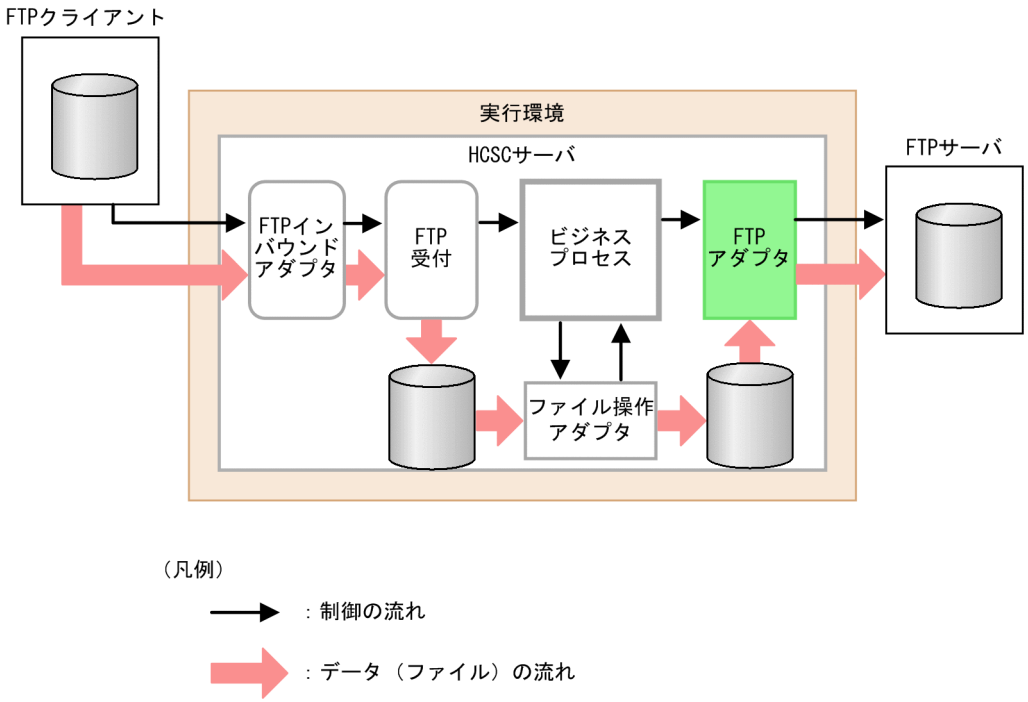
既存のメッセージキューを利用する場合は，ビジネスプロセスから Message Queue アダプタを呼び出して，メッセージの送受信，およびブラウズをすることができます。

9.6.10　FTP アダプタ

サービスプラットフォームでは，FTP クライアントと FTP サーバ間のファイル転送に対応する FTP アダプタを提供しています。FTP アダプタは，FTP サーバとのファイル転送処理を行います。

FTP アダプタと FTP サーバとの間でファイルを送受信する場合の概略を次の図に示します。

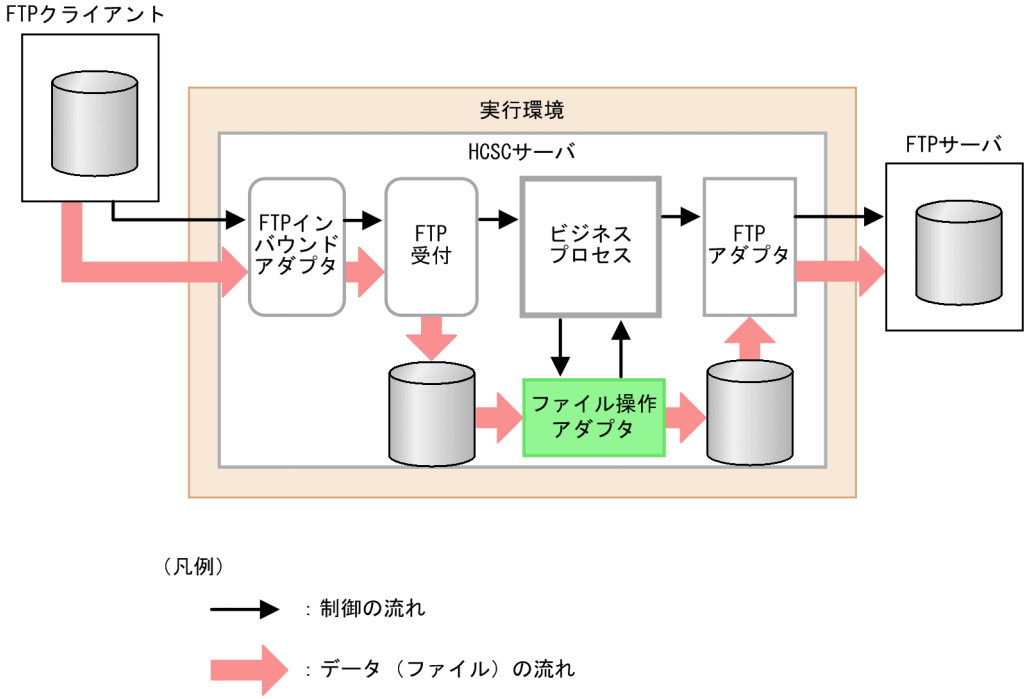
図 9-24　FTP アダプタと FTP サーバとの間のファイルの送受信



9.6.11　ファイル操作アダプタ

サービスプラットフォームでは，ファイル操作アダプタから送受信ファイルのフォーマット変換，複製，削除などができます。ファイル操作アダプタからファイルを操作する場合の概略を次の図に示します。

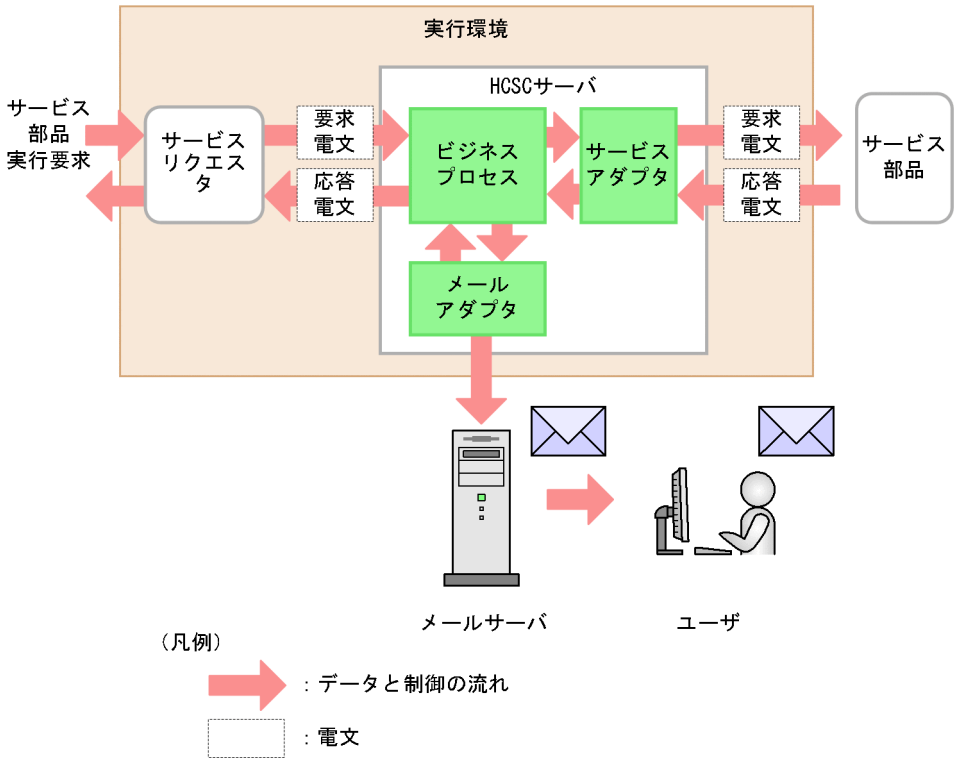
図 9-25　ファイル操作アダプタからのファイルの操作



9.6.12　メールアダプタ

サービスプラットフォームでは，メールアダプタから，SMTP プロトコルをサポートしたメールサーバをサービスとして呼び出して，メールを送信することができます。メールアダプタを利用して，メールを送信する場合の概略を次の図に示します。

図 9-26　メールアダプタからのメールの送信

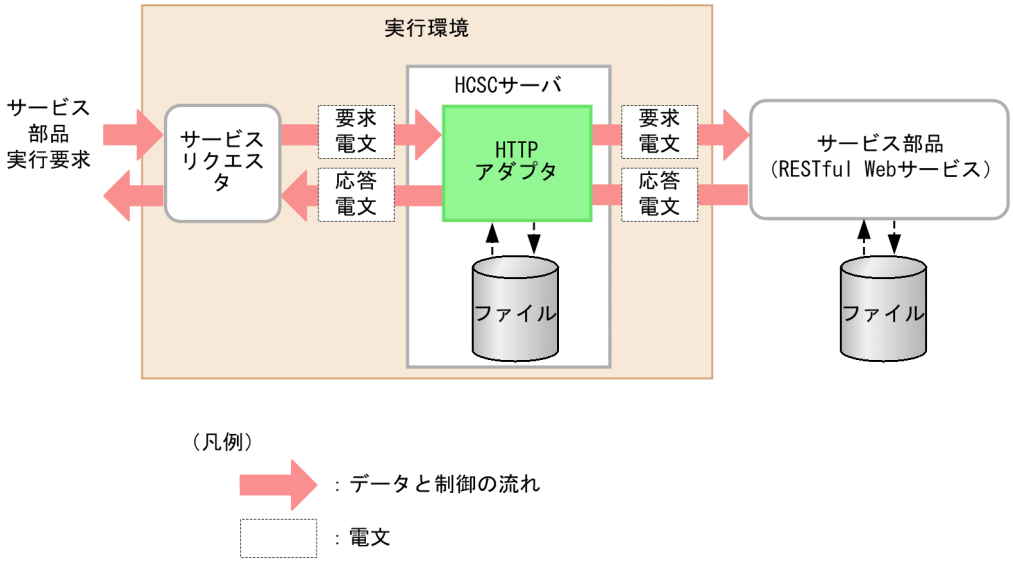


9.6.13　HTTP アダプタ

サービスプラットフォームでは，Web サーバ上に公開されているリソースや，REST スタイルで公開されている Web サービス（RESTful Web サービス）を呼び出すための HTTP アダプタを提供しています。

HTTP アダプタからサービス部品を呼び出して実行する場合の概略を次の図に示します。

図 9-27　HTTP アダプタからのサービス部品実行



HCSC サーバへサービスの実行要求（要求電文）を送信するアプリケーションであるサービスリクエスタは，サービス部品（RESTful Web サービス）の実行要求を受け付けると，HTTP アダプタへ要求電文を送信します。

HTTP アダプタは，アダプタ内で定義されているサービス部品へ要求電文を送信し，サービス部品を実行します。

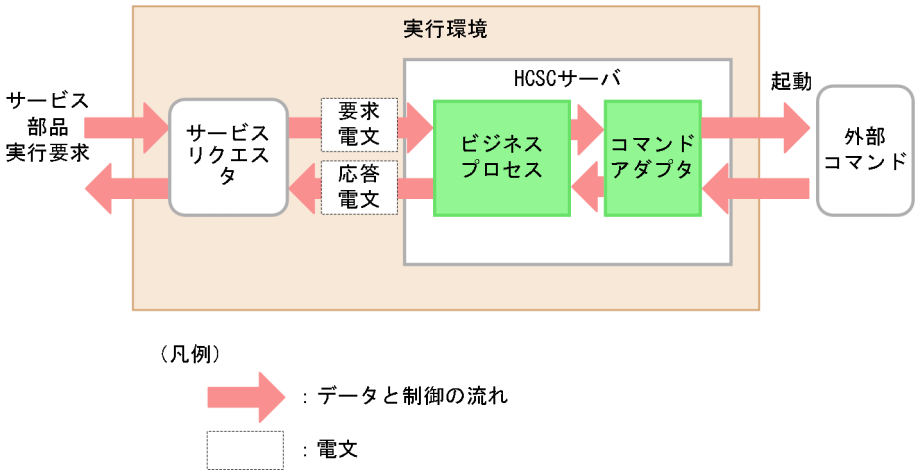
サービス部品での処理が終了すると，処理結果として応答電文が HTTP アダプタに送信されます。応答電文は，サービスリクエスタを経由して業務担当者へ通知されます。

なお，HTTP アダプタでは，HCSC サーバとサービス部品との間でファイルを送受信することもできます。これにより，クライアントから送信したファイルを Web サーバに転送したり，Web サーバ同士でファイルを送受信したりといった使い方ができます。

9.6.14　コマンドアダプタ

サービスプラットフォームでは，コマンドアダプタから EXE ファイルや sh ファイルなどの外部コマンドを実行できます。コマンドアダプタを使用して外部コマンドを実行する場合の概略を次の図に示します。

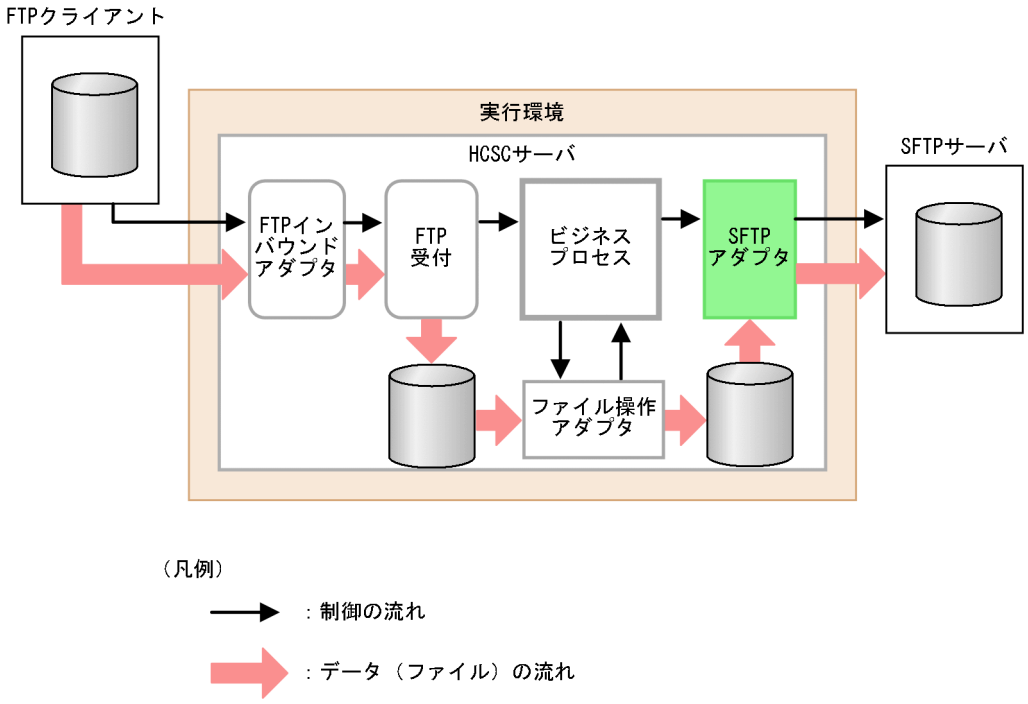
図 9-28　コマンドアダプタからのサービス部品実行



9.6.15　SFTP アダプタ

サービスプラットフォームでは，SFTP サーバとのファイル転送に対応する SFTP アダプタを提供しています。SFTP アダプタから SFTP サーバへファイルを送信する場合の概略を次の図に示します。

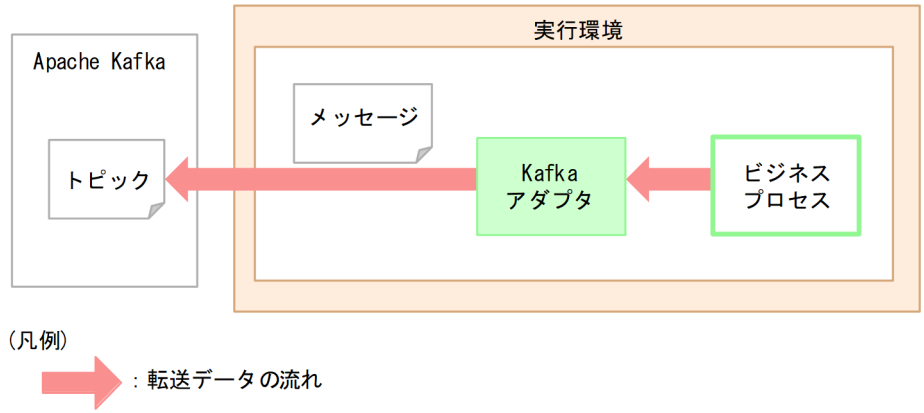
図 9-29　SFTP アダプタから SFTP サーバへのファイル送信



9.6.16　Kafka アダプタ

サービスプラットフォームでは，Apache Kafka にメッセージを送信するためのサービスアダプタとして Kafka アダプタを提供しています。Kafka アダプタは，ビジネスプロセスからのメッセージ送信要求に従って，Apache Kafka にメッセージを送信します。Kafka アダプタから Apache Kafka へメッセージを送信する場合の概略を次の図に示します。

図 9-30　Kafka アダプタから Apache Kafka へのメッセージ送信

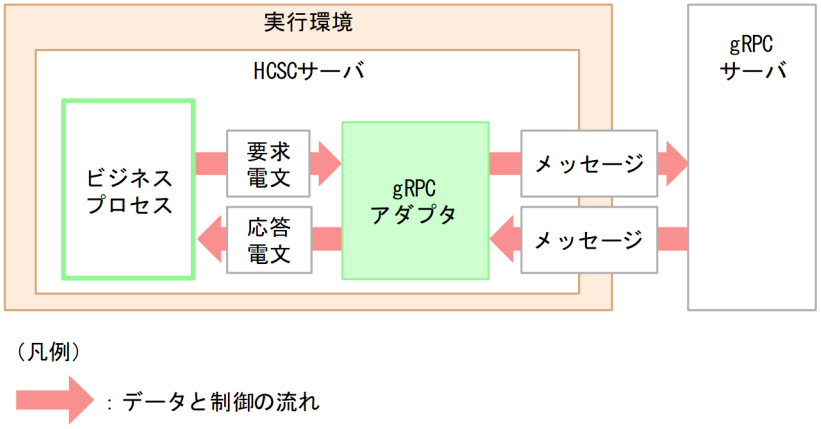


9.6.17　gRPC アダプタ

サービスプラットフォームでは，gRPC サーバとメッセージを送受信するためのサービスアダプタとして gRPC アダプタを提供しています。gRPC アダプタは，ビジネスプロセスからのメッセージ送信要求に従って，gRPC サーバにメッセージを送信します。gRPC サーバから受信したメッセージは，gRPC アダプタで応答電文に変換され，ビジネスプロセスに返されます。

gRPC アダプタと gRPC サーバとでメッセージを送受信する場合の概略を次の図に示します。

図 9-31　gRPC アダプタと gRPC サーバとのメッセージの送受信



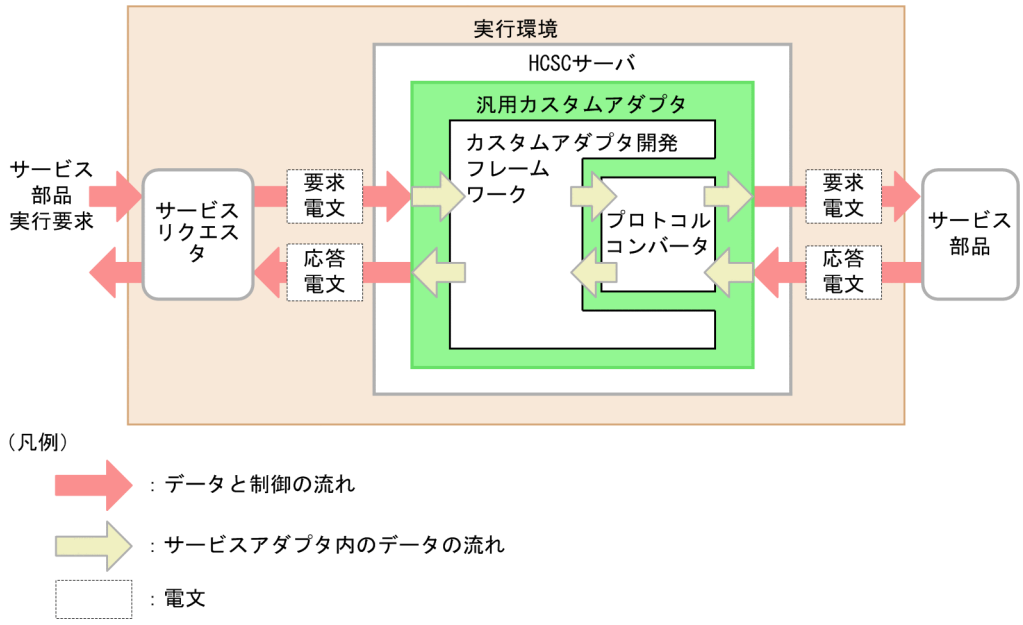
9.6.18　汎用カスタムアダプタ

サービスプラットフォームでは，製品が提供するサービスアダプタでは対応できないプロトコルを持つシステムのサービス部品を実行するために，カスタムアダプタ開発フレームワークを提供しています。

提供しているカスタムアダプタ開発フレームワークとプロトコルコンバータを組み合わせた汎用カスタムアダプタから，任意のサービス部品を呼び出して実行できます。

汎用カスタムアダプタからサービス部品を実行する場合の概略を次の図に示します。

図 9-32　汎用カスタムアダプタからのサービス部品の実行



HCSC サーバへサービスの実行要求（要求電文）を送信するアプリケーションであるサービスリクエスタは，サービス部品の実行要求を受け付けると，汎用カスタムアダプタへ要求電文を送信します。

要求を受けた汎用カスタムアダプタは，任意のシステムへ接続できるようにアダプタ内で定義されている処理（データ変換，およびプロトコル変換）を実行したあと，サービス部品へ要求電文を送信し，サービス部品を実行します。

9.7　実行履歴の管理機能

ビジネスプロセスの実行状態を管理する機能について説明します。

9.7.1　プロセスインスタンスの実行履歴の管理

ビジネスプロセスのプロセスインスタンスの実行状態を実行履歴として管理できます。

ビジネスプロセスの実行状態を管理するには，ビジネスプロセスのプロセスインスタンスの実行状態をデータベースに保存（永続化）するよう設定します。

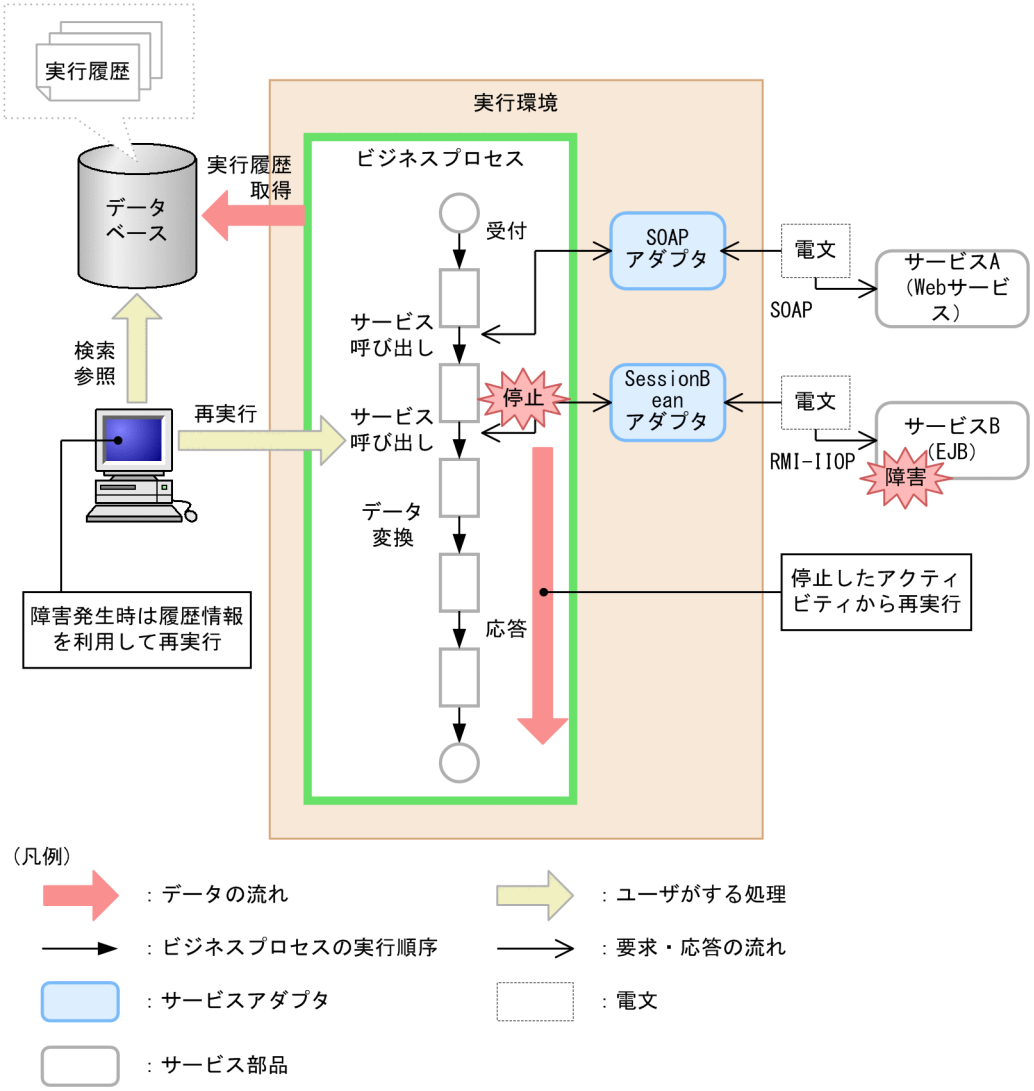
注意事項

データベースを使用する場合だけ，ビジネスプロセスの実行状態を管理できます。

実行履歴を管理することで，ビジネスプロセスから実行するサービスの呼び出しでエラーが発生した場合に，実行履歴を利用し，エラーが発生しているサービスからリトライすることができます。リトライは，一件ずつ実行することも一括して実行することもできます。

プロセスインスタンスの実行履歴の管理機能について次の図に示します。

図 9-33　プロセスインスタンスの実行履歴の管理機能



実行履歴の検索には，相関セット，開始日時または状態などを検索キーとして利用できます。また，ビジネスプロセス上を流れる電文の形式を参照することもできます。

実行履歴の内容から，各サービスの稼働状況を把握したり，取得稼働状況を分析したりできるため，システム全体の最適化を図るなど，ビジネスプロセスの可視化が可能です。

10 システムの開発と運用

この章では，サービスプラットフォームを利用したシステムの開発と運用について説明します。

なお，この章で説明している開発に関する作業の詳細については，マニュアル「サービスプラットフォーム 開発ガイド 基本開発編」，およびマニュアル「サービスプラットフォーム 開発ガイド 受付・アダプタ定義編」を参照してください。運用に関する作業の詳細については，マニュアル「サービスプラットフォーム システム構築・運用ガイド」を参照してください。

10.1　SOA を適用したシステム開発

この節では，SOA を適用したシステム開発の利点や，特徴などについて説明します。

10.1.1　SOA を適用したシステム開発の利点

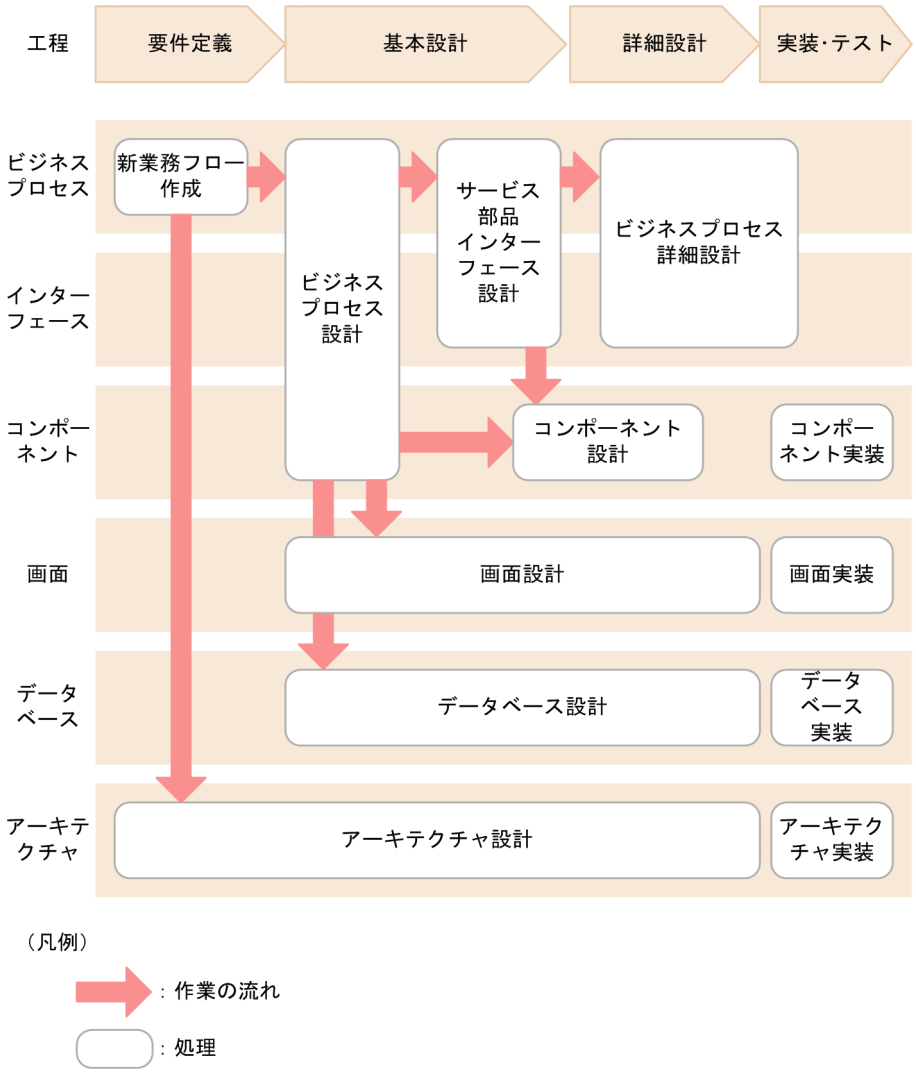
システム開発の点でも SOA を適用すると，次に示す利点があります。

- **サービスの再利用の促進**
  新規に IT 化するビジネスプロセスに対して，すでに IT 化されているビジネスプロセスの中から，類似パターンのサービスを見いだすことで，既存のビジネスプロセスで使用されているサービスの再利用を促進できます。
- **保守性の向上**
  業務とソフトウェアの単位を一致させることで，業務の変更に応じたソフトウェアの変更個所を容易に特定できます。また，サービスを適切な粒度で設計することで，サービスの独立性を高めて，ソフトウェアの変更をサービス内に限定できます。これによって，情報システムの保守性を向上できます。
- **開発費用の削減**
  既存システムをサービスとして活用することで，新規開発部分を削減して，開発費用を削減できます。

10.1.2　SOA を適用したシステム開発手法

SOA を適用したシステム開発では，その利点を生かすために，ビジネスプロセスおよびサービスのインターフェースの設計後に，サービスのコンポーネントを設計します。SOA を適用したシステム開発手法の全体像を次の図に示します。

図 10-1　SOA を適用したシステム開発手法の全体像



SOA を適用したシステム開発手法には，大きく分けて，次に示す工程があります。

- **要件定義**

  ビジネスプロセスの基となる新業務フローを作成します。また，アーキテクチャ設計として機能以外の要件を調査し，システム方式を検討します。

- **基本設計**

  定義された要件を実現するためのビジネスプロセス，およびサービスのインターフェースを設計します。また，アプリケーション（コンポーネント，画面，データベース，およびアーキテクチャ）の機能も設計します。

- **詳細設計**

  ビジネスプロセスの各種仕様，およびアプリケーション設計の各種仕様に基づき，実装のための詳細な仕様を設計します。

- **実装・テスト**

  アプリケーションを実装し，運用に向けてテストします。

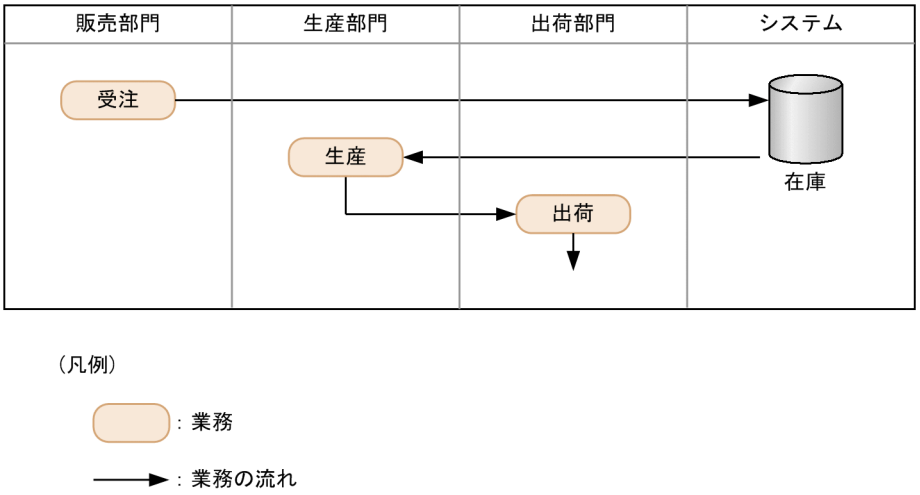
各工程で実施する作業について次に説明します。

(1)　新業務フローの作成

業務改革・改善方針，および業務問題分析結果に基づいて，機能要件を調査，新業務フローを作成します。WFA（Work-Flow Architecture）などの記法を使用して作成します。業務フローには，業務にかかわる組織や担当者，業務の流れ，および流れる情報を明示します。

業務フロー作成時には，システム化対象となる業務の機能に対して，サービス候補を決めておきます。また，各サービスの開発方針を決定し，新規開発，既存システムの再利用，パッケージソフトの導入，およびサービス型ソフトウェアの利用などを方針として決定します。これらの利用が決定している場合は，対象とする業務をサービスとして決めておきます。新業務フローの作成例を次の図に示します。

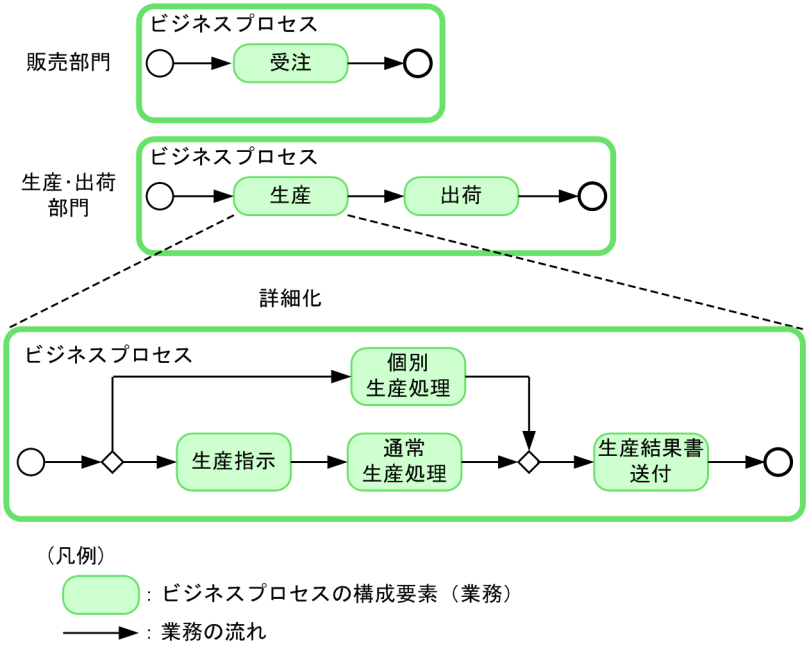
図 10–2　新業務フローの作成例



(2)　ビジネスプロセスの設計

作成した新業務フローを基に，ビジネスプロセス図（基本フロー）を作成します。また，業務の観点で，ビジネスプロセスを構成する業務を，さらに詳細なビジネスプロセスとして段階的に階層化して，サービスの粒度を見直します。ビジネスプロセス図の作成例を次の図に示します。

図 10–3　ビジネスプロセス図の作成例



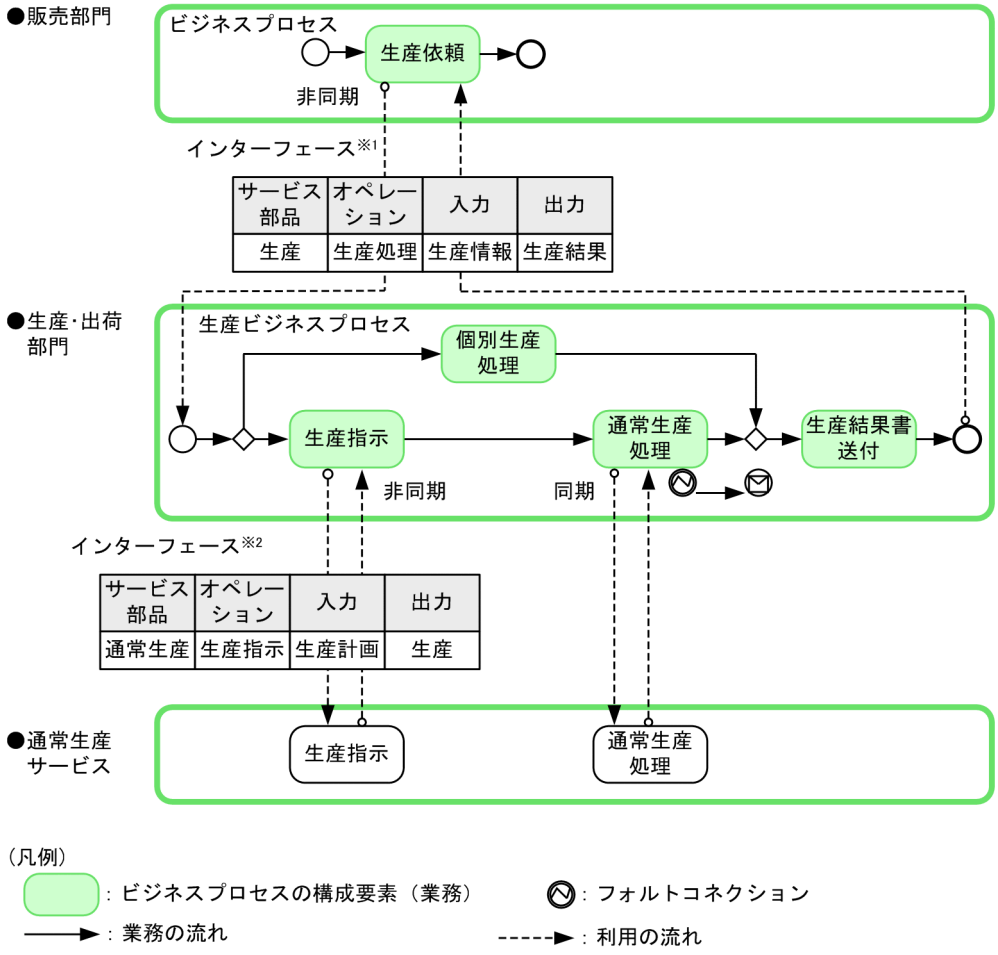
(3)　サービスのインターフェースの設計

システムの観点でビジネスプロセスを見直し詳細化します。これとともに，サービスの実装に必要なインターフェース（メッセージの構造や例外処理の追記など）を決定します。

対話ワークフローパターン，対話型アプリケーションパターン，およびオンライン型アプリケーションパターンに対応したインターフェースを設計します。また，この段階で，サービスの粒度についても最終的に決定します。

インターフェースには，サービスに対するインターフェースのほかに，ビジネスプロセス自体のインターフェースがあります。ビジネスプロセス自体も，連携先のサービスを組み合わせた複合的なサービスとなるため，インターフェースが必要となります。サービスおよびビジネスプロセスのインターフェースの設計例を次の図に示します。

図 10-4　サービスおよびビジネスプロセスのインターフェースの設計例



（凡例）

：ビジネスプロセスの構成要素（業務）　：フォルトコネクション

：業務の流れ　：利用の流れ

注※1　ビジネスプロセスを呼び出すときの定義内容を示します。

注※2　サービス部品の持つ機能を呼び出すときの定義内容を示します。

(4)　ビジネスプロセスの詳細設計

ビジネスプロセス，およびインターフェースの各種基本設計仕様を基に，実装に必要なビジネスプロセス定義や，サービスが授受する電文の詳細定義などを設計します。ビジネスプロセスについては BPEL を，インターフェースや電文については WSDL または XML の言語を使用して設計や実装をします。

(5)　コンポーネント設計

フロントシステムやサービスを実現するアプリケーションを設計します。サービスのコンポーネントは，プレゼンテーション層（画面関係），ファンクション層（業務機能），データ層の 3 層で構成され，3 層ごとに設計します。Struts，JSF など，業界標準の各種フレームワークを使用することを前提とした設計ができます。

(6)　画面設計

フロントシステムやサービスで，対話で必要な画面や画面遷移を設計します。画面設計は，開発全体にわたって設計を継続します。各工程では，次のように設計します。

- 要件定義：画面の簡易レイアウトによって，エンドユーザの要件を洗い出します。
- 基本設計：作成されたシナリオを基に，対話アプリケーションでの画面や画面遷移を設計します。
- 詳細設計：作成されたシナリオを基に，対話アプリケーションでの HTML，JSP などを対象とした実装のための設計をします。

(7) データベース設計

フロントシステムやサービスで扱われるデータの分析から，データベースのテーブル設計までをします。データベース設計は，開発全体にわたって設計を継続します。各工程では，次のように設計します。

- 要件定義：データの分析，およびキー項目を抽出して，対象システム全体で扱うデータの関連を把握します。
- 基本設計：作成されたシナリオを基に，必要なデータを分析し，構成要素とその関連を抽出し，データの論理設計をします。
- 詳細設計：物理テーブルを対象としたデータ物理設計をします。

(8) アーキテクチャ設計

システム化計画に基づき，機能以外の要件を調査し，対象システム全体のアーキテクチャ概要を設計します。その際，新業務フロー作成でサービス候補として挙げたサービスの開発方針を考慮して，既存システムとの連携方式，およびパッケージソフトウェアとの連携方式を検討します。基本設計工程から詳細設計工程にわたって設計を継続し，機能以外の要件に対応するシステムの方式を設計します。

10.2　各環境の関係とシステム構成

サービスプラットフォームでは，開発環境，運用環境および実行環境の 3 つの環境を構築します。開発環境は「Service Architect」を，運用環境および実行環境は「Service Platform」をインストールして構築します。

サービスプラットフォームの各環境の関係について説明します。

10.2.1　ソフトウェア製品と各環境の関係

(1)　サービスプラットフォームを構成する環境

サービスプラットフォームは，次に示す 3 つの環境から構成されています。

- **開発環境**
  サービスを統合するために必要な HCSC コンポーネント（開発環境で作成するサービスアダプタ，ビジネスプロセス，およびユーザ定義受付の総称）を作成して，EAR ファイルにパッケージングするための環境です。
- **実行環境**
  要求に応じたサービス部品やビジネスプロセスを呼び出して，業務を実行するための環境です。
- **運用環境**
  開発環境で作成した HCSC コンポーネントを，実行環境に配備したり，以降の運用操作を実行したりするための環境です。また，実行環境から情報を収集して，サービス部品の稼働状態を確認できます。

(2)　開発環境・実行環境・運用環境の関係

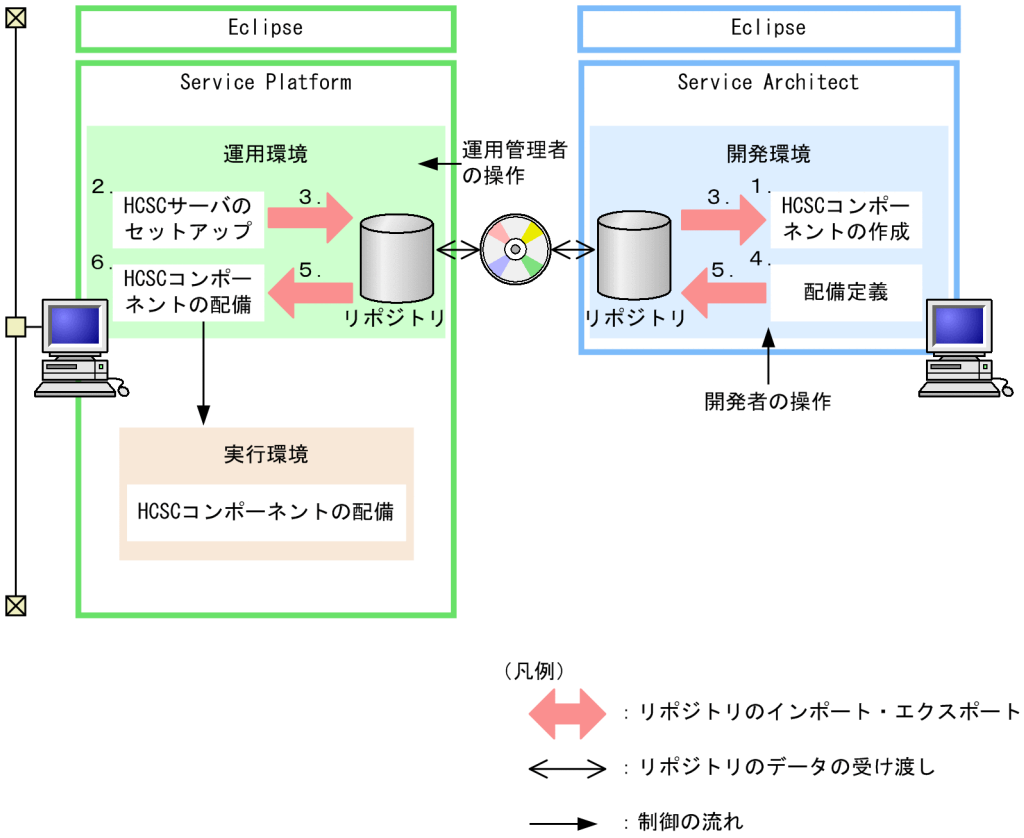
開発環境，実行環境・運用環境をそれぞれ異なるマシンに構築します。

開発環境には Service Architect と Eclipse を，実行環境・運用環境には Service Platform と，Windows の場合は Eclipse をインストールします。

また，運用環境から実行環境を操作するには，リポジトリというデータモデルが必要です。

サービスプラットフォームを構成する環境を次の図に示します。

図 10-5　サービスプラットフォームを構成する環境



これらの環境は，相互に連携してサービスの統合環境を実現しています。各環境は，環境構築後，次に示す流れを経て実際に運用できるようになります。

1. 開発環境で HCSC コンポーネントを作成します。
2. 実行環境・運用環境で，HCSC サーバをセットアップし，システムの構成を定義します。
3. 実行環境・運用環境からリポジトリをエクスポートし，運用環境で定義したシステム構成定義を開発環境にインポートします。
4. 実行環境・運用環境で定義したシステム構成定義を基に，システム構成のどこに配備するかを定義して更新します（配備定義）。
5. 開発環境で定義した配備定義を含むリポジトリをエクスポートし，実行環境・運用環境にインポートします。
6. 開発環境で定義した配備定義を基に，コンポーネントを配備します。

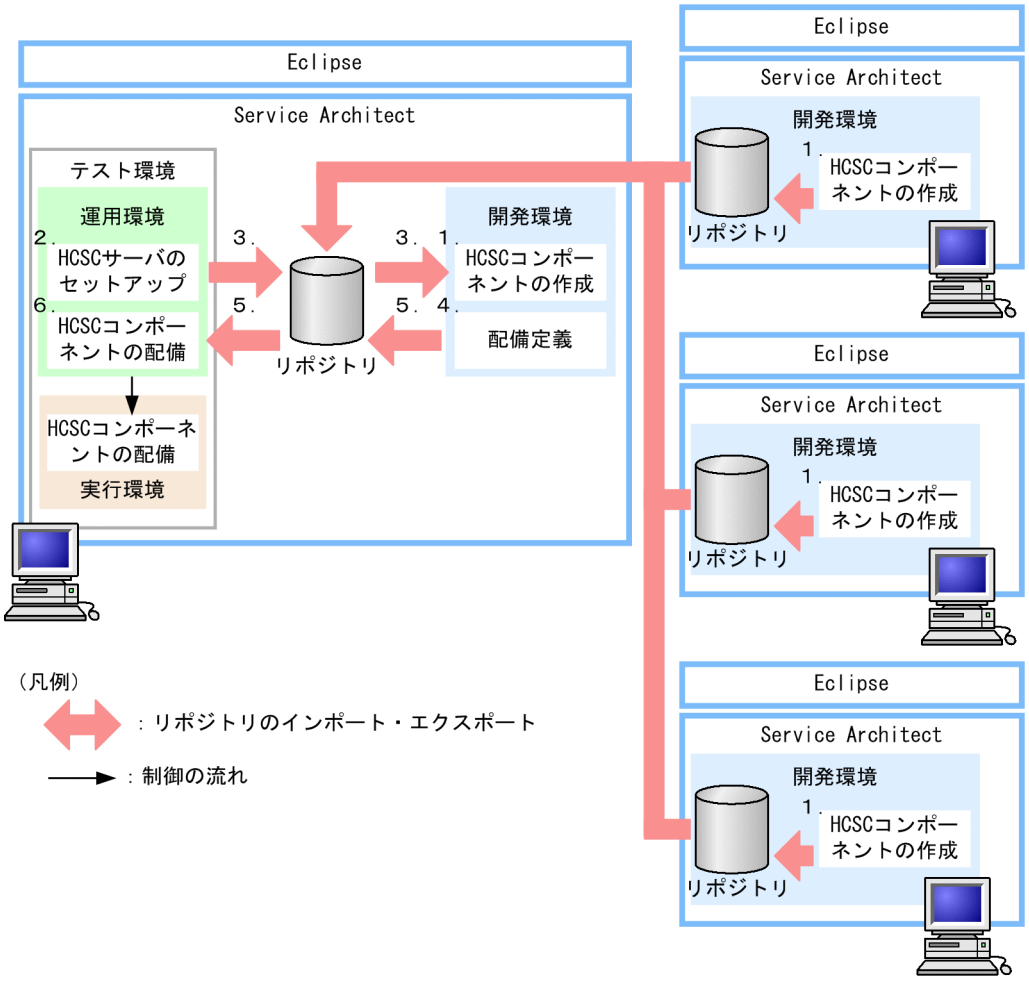
環境間の情報の受け渡しには，環境間で共有する情報を格納したリポジトリを使用します。リポジトリは媒体を経由して，ZIP ファイル形式で保存したり，読み込んだりします。

(3)　開発環境とテスト環境との関係

開発環境では，複数台のマシンで作成した HCSC コンポーネントの情報を，リポジトリを通じて 1 つにまとめることができます。そして，開発環境と同じマシンに，作成した HCSC コンポーネントのテスト・デ

バッグを実施するためのテスト環境を構築できます。テスト環境は，テストに必要な簡易的な環境で，一括構築できます。開発環境とテスト環境との関係を次の図に示します。

図 10-6　開発環境とテスト環境との関係



開発環境とテスト環境を利用する場合，次に示す流れを経て運用します。

1. 開発環境で HCSC コンポーネントを作成します。
2. テスト環境を構築します（HCSC 簡易セットアップ機能を使うことで，HCSC サーバのセットアップや，システムの構成が定義できます）。テスト環境の構築時にも，本番用の実行環境を想定します。
3. テスト環境からリポジトリをエクスポートし，テスト環境で定義したシステム構成定義を開発環境にインポートします。
4. テスト環境で定義したシステム構成定義を基に，システム構成のどこに配備するかを定義し更新します（配備定義）。
5. 開発環境で定義した配備定義を含むリポジトリをエクスポートし，テスト環境にインポートします。
6. 開発環境で定義した配備定義を基に，テスト環境にコンポーネントを配備します。

この場合のような環境間の情報の受け渡しにも，リポジトリを使用します。開発環境とテスト環境は同じマシンに構築されるため，媒体を使用しないで情報を受け渡します。

(4) テスト環境と本番環境との関係

サービスプラットフォームでは，まず，テスト環境を構築してテストやデバッグを実施します。テスト環境の構築には，HCSC 簡易セットアップ機能を使用できます。

実際のシステム開発では，テスト環境のほかに本番で使用する環境を構築する必要があります。そのため，テスト環境で使用したリポジトリを，本番環境に移行する必要があります。

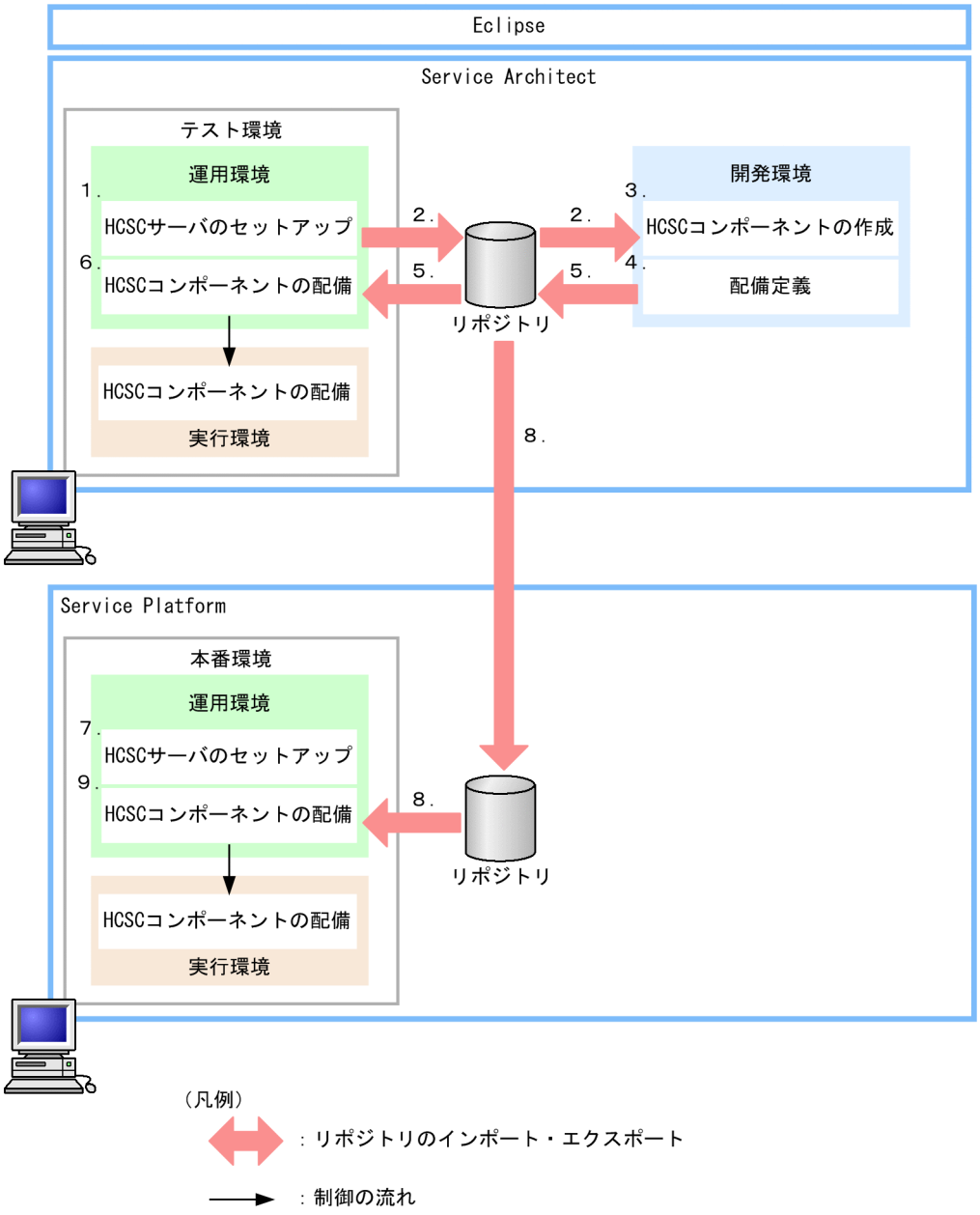
次のようにテスト環境と本番環境で，データベースと Reliable Messaging の使用有無を同じ設定で構築した場合，テスト環境で使用したリポジトリをそのまま本番環境へ移行できます。

- テスト環境と本番環境で，データベースと Reliable Messaging の両方を使用する場合
- テスト環境と本番環境で，データベースと Reliable Messaging の両方を使用しない場合
- テスト環境と本番環境で，データベースを使用し Reliable Messaging を使用しない場合

なお，テスト環境と本番環境で，データベースと Reliable Messaging の使用有無の設定が同じでない場合でも，リポジトリは移行できます。詳細については，マニュアル「サービスプラットフォーム システム構築・運用ガイド」の「1.3　テスト環境と本番環境との関係」を参照してください。

テスト環境と本番環境で，データベースと Reliable Messaging の使用有無を同じ設定で構築した場合の移行の流れを次の図に示します。

図 10-7　データベースと Reliable Messaging の使用有無を同じ設定で構築した場合の移行の流れ



1. テスト環境の HCSC サーバをセットアップし，システムの構成を定義します。

   HCSC 簡易セットアップ機能を使うことで，HCSC サーバのセットアップや，システムの構成が定義できます。ただし，データベースを使用して Reliable Messaging を使用しない構成を構築する場合，HCSC 簡易セットアップ機能は使用できません。

2. 運用環境からリポジトリをエクスポートし，テスト環境で定義したシステム構成定義を開発環境にインポートします。

3. 開発環境で，HCSC コンポーネントを作成します。

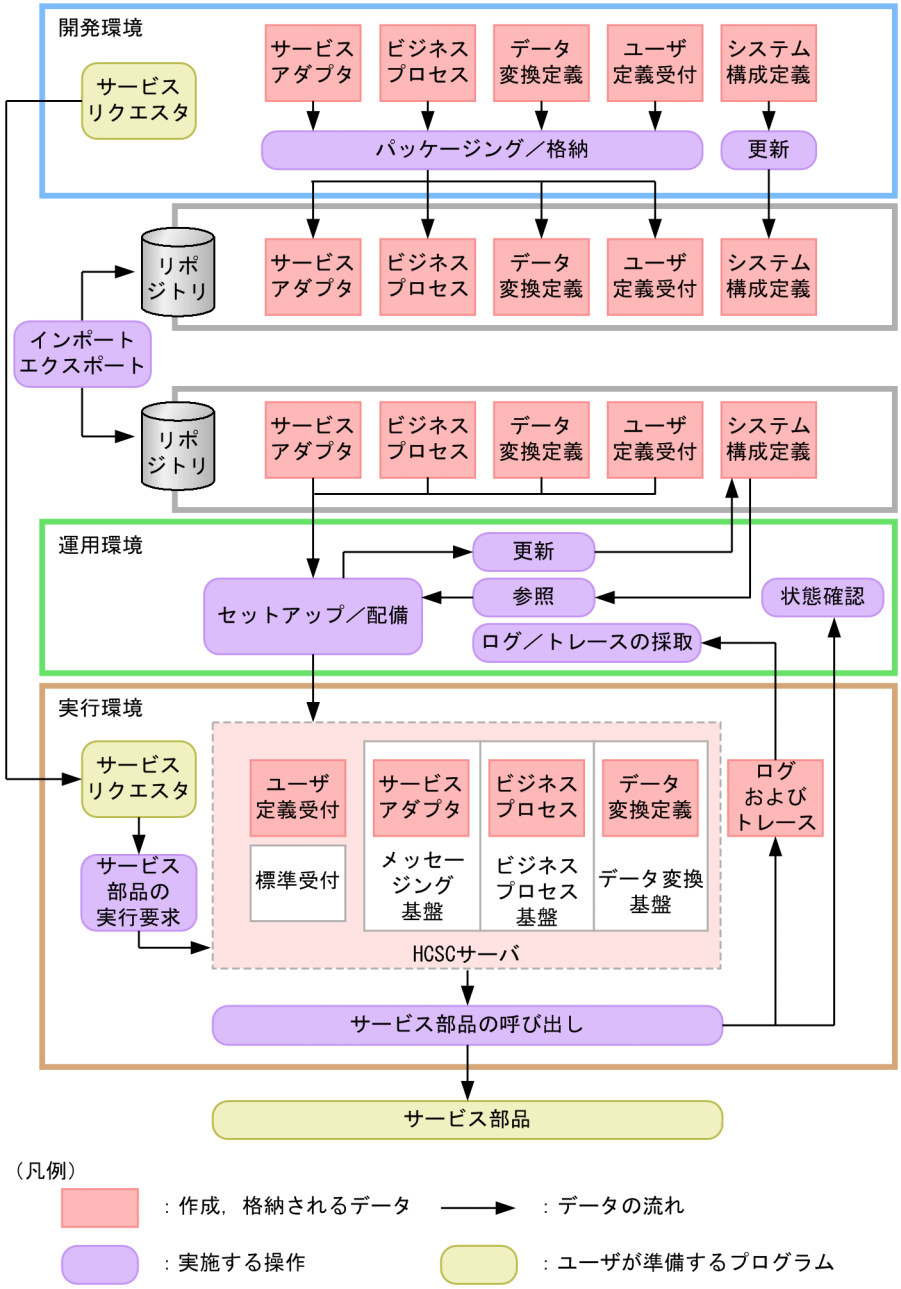
4. テスト環境で定義したシステム構成定義を基に，システム構成のどこに配備するかを定義し更新します（配備定義）。

5. 開発環境で定義した配備定義を含むリポジトリをエクスポートし，運用環境にインポートします。
6. 開発環境で定義した配備定義を基に，テスト環境に HCSC コンポーネントを配備します。
7. 本番環境の HCSC サーバをセットアップし，システムの構成を定義します。
   テスト環境と本番環境で，データベースと Reliable Messaging の使用有無の設定を同じにします。
8. 開発環境で定義した配備定義を含むリポジトリをエクスポートし，運用環境にインポートします。
   標準インポートを使用すると，HCSC サーバ名や IP アドレスなどが異なる場合でも，テスト環境のリポジトリをそのまま本番環境へ移行できます。
9. 開発環境で定義した配備定義を基に，本番環境に HCSC コンポーネントを配備します。

10.2.2　システムの運用と各環境の関係

サービスプラットフォームでは，開発環境，運用環境および実行環境が相互に関連してシステム全体を構成します。サービスプラットフォーム全体の運用と開発環境，運用環境および実行環境の関係を次の図に示します。

図 10-8　システムの運用と開発環境，運用環境および実行環境の関係



開発環境で定義した内容は，リポジトリを使用して運用環境に取り込みます。取り込んだ内容は，実行環境にセットアップしたり配備したりします。あらかじめ作成したサービスリクエスタからサービス部品の実行要求がくると，HCSC サーバからサービス部品が呼び出されます。運用環境からは，サービス部品の呼び出しなどの状態の確認，およびログやトレースの採取による管理ができます。

図 10-8 に示したサービスプラットフォームの開発環境，運用環境および実行環境についてそれぞれ説明します。

(1)　開発環境

サービス部品やビジネスプロセスを実行するために必要な HCSC コンポーネントとシステム構成定義を作成する環境です。

HCSC コンポーネントとは，開発環境で作成するサービスアダプタおよびビジネスプロセスを総称したものです。

システム構成定義とは，HCSC コンポーネントを実行環境にどう配備するか定義したものです。システム構成定義には，運用環境での HCSC サーバおよびクラスタのセットアップ情報と，HCSC コンポーネントを実行環境にどう配備するかの情報が含まれます。開発環境では，運用環境で作成，更新したセットアップ情報を，リポジトリを利用して取得し，HCSC コンポーネントをどのように配備するかを定義します。

作成した HCSC コンポーネントは，EAR ファイルに組み立てます。**EAR ファイル**とは，HCSC コンポーネントに関するファイルを，実行環境に配備できるように組み立てたものです。EAR ファイルを作成することを**パッケージング**と呼びます。パッケージングした EAR ファイルはリポジトリに格納します。

リポジトリとは，定義した情報を格納するディレクトリです。格納した情報は，リポジトリの管理機能（リポジトリのインポート／エクスポート機能）を利用して，開発環境と運用環境との間で受け渡しをします。

また，開発環境では，実行環境でサービス部品を実行するための要求電文を受け付けて，サービスアダプタおよびビジネスプロセスに要求電文を送信する**サービスリクエスタ**も作成します。

(2) 運用環境

開発環境で作成した EAR ファイルをリポジトリから読み込み，実行環境に配備する環境です。また，実行環境で利用する HCSC サーバをセットアップします。

運用を開始したあとは，システムの起動・停止，および状態を監視したり，ログやトレースを採取したりします。

(3) 実行環境

サービスリクエスタで受け付けた要求電文に応じて，HCSC サーバを介してサービス部品およびビジネスプロセスを呼び出し，業務を実行する環境です。HCSC サーバには，メッセージング基盤，ビジネスプロセス基盤，データ変換基盤が含まれます。また，開発環境で作成した HCSC コンポーネントは，運用環境から HCSC サーバに配備されます。

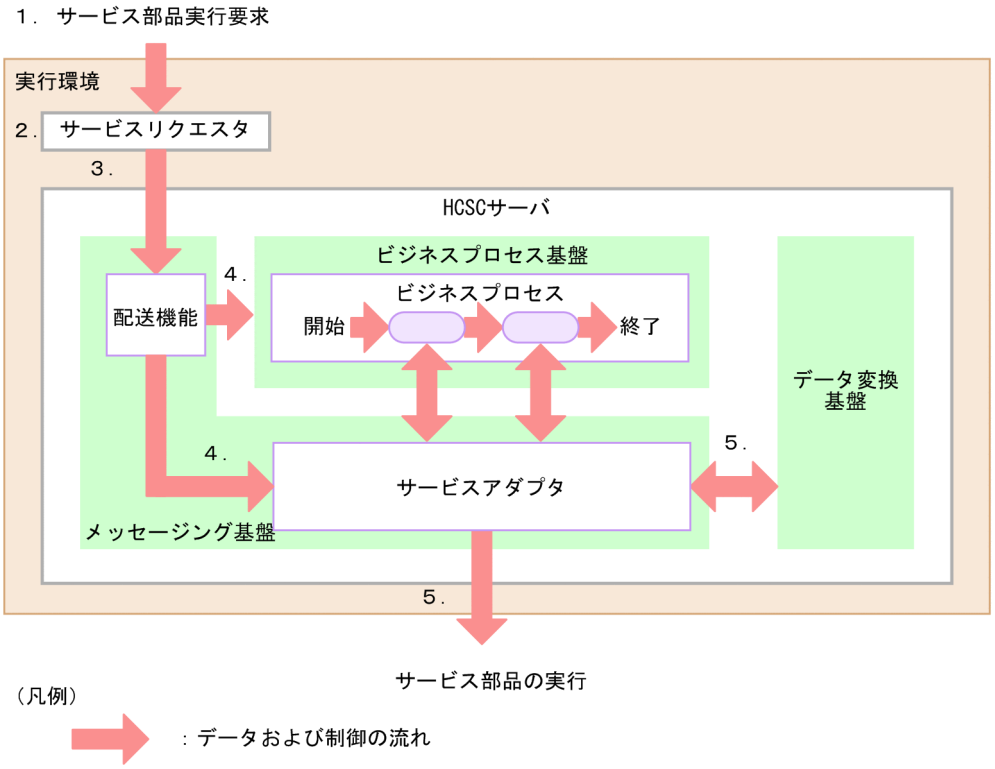
サービスリクエスタが要求電文を受け付けると，メッセージング基盤に要求電文が送信されます。そのあと，メッセージング基盤の配送機能によって，要求電文に応じて適切なサービスアダプタまたはビジネスプロセスへ要求が送信されます。

要求電文がビジネスプロセスへの要求の場合，要求電文がビジネスプロセス基盤に送信されます。ビジネスプロセス基盤ではビジネスプロセスの定義に従って，メッセージング基盤を介して順次サービス部品を呼び出します。

サービス部品の実行に際してデータ変換するよう設定されている場合，データ変換基盤を利用してデータ変換をして，サービス部品が実行されます。

実行環境での制御の流れを次の図に示します。

図 10-9　実行環境での制御の流れ



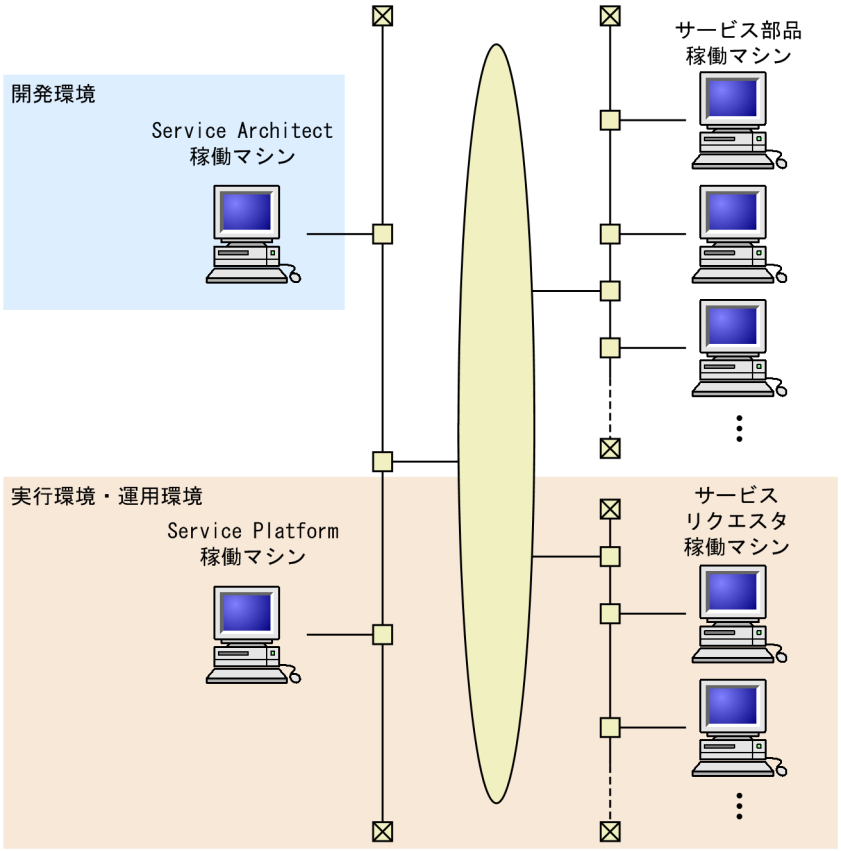
1. 業務担当者がサービス部品またはビジネスプロセスの実行を要求します。
2. サービスリクエスタが要求電文を受け付けます。
3. サービスリクエスタから要求電文が送信されます。
4. 要求電文は，配送機能によって，適切なサービスアダプタまたはビジネスプロセスへ送信されます。
5. データ変換定義に従って，必要に応じて要求電文のデータが変換されてサービス部品が呼び出されます。

10.2.3　ネットワークの構成と各環境の関係

サービスプラットフォームには，開発環境，および実行環境・運用環境があり，ネットワークで接続してシステムを構成します。開発環境，および実行環境・運用環境を，それぞれ別のマシンに構築することを推奨します。

サービスプラットフォームで構築する各環境とネットワークの構成を次の図に示します。

図 10-10　各環境とハードウェアの構成

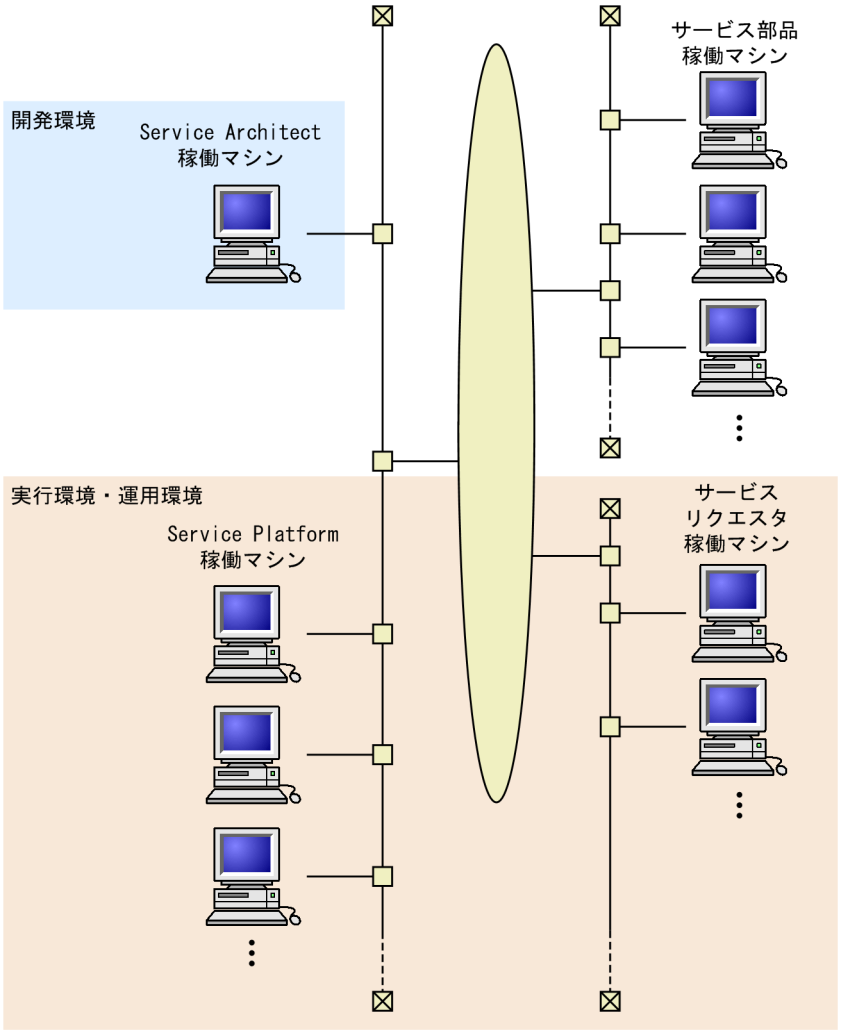


開発環境は別のネットワーク上に構築することもできます。

なお，実行環境・運用環境では，2 つの HCSC サーバを組み合わせてクラスタを構成し，冗長で信頼性の高い構成のシステムを構築できます。

HCSC サーバを組み合わせてクラスタを構成した場合の各環境と，ハードウェアの構成について，次の図に示します。

図 10–11　各環境とハードウェアの構成（HCSC サーバを組み合わせてクラスタを構成する場合）



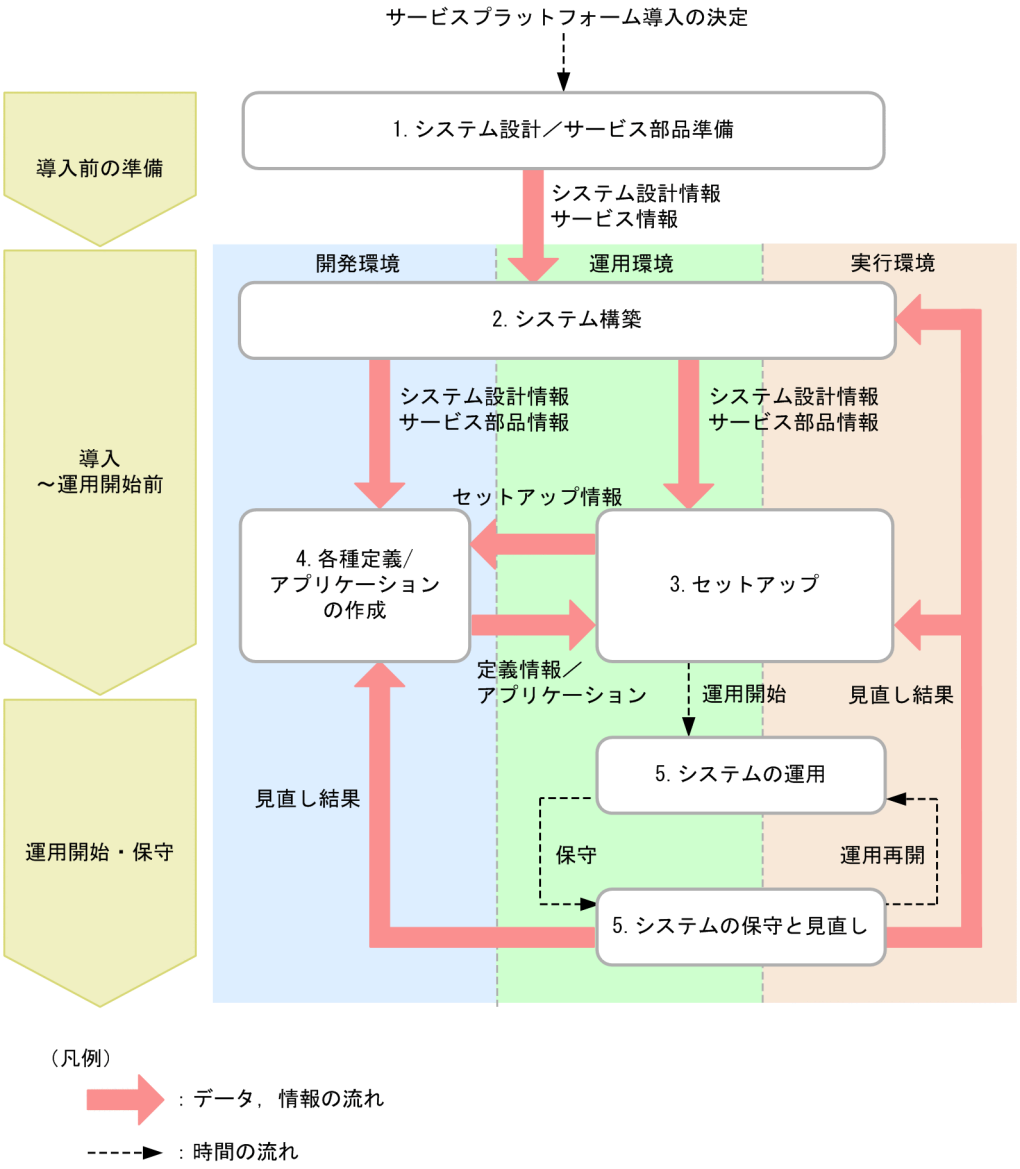
開発環境は別のネットワーク上に構築することもできます。

クラスタ構成を利用したシステムについては，マニュアル「サービスプラットフォーム 解説」の「1.4.1 ロードバランス機能を利用した HCSC サーバの冗長構成」を参照してください。

10.3　サービスプラットフォームを導入したシステムのライフサイクル

サービスプラットフォームを導入したシステムのライフサイクルを図に表すと，次のようになります。

図 10–12　サービスプラットフォームのシステムのライフサイクル



サービスプラットフォームを導入したシステムのライフサイクルには，次の段階があります。

- システム設計／サービス部品準備
- システム構築
- セットアップ
- 各種定義／アプリケーションの作成
- システムの運用
- システムの保守と見直し

ライフサイクルの中での各段階の位置づけは次のとおりです。

1. システム設計／サービス部品準備

   サービスプラットフォームの導入を決定したら，まず，システムで実行する業務を分析し，業務をサービス化します。また，サービスの利用方法に応じたシステムを設計します。

2. システム構築

   1.で検討したシステムの設計情報およびサービス部品情報に基づいて，開発環境，運用環境および実行環境を構築します。

3. セットアップ

   システムの運用開始に向けて運用環境，実行環境をセットアップします。1.で検討したシステムの設計情報およびサービス部品情報に加えて，開発環境で作成された定義情報やアプリケーションも利用します。

4. 各種定義／アプリケーションの作成

   システムの運用に必要な各種定義とアプリケーションを開発します。

   開発には，1.で検討したアプリケーション設計情報およびサービス部品情報に加えて，運用環境および実行環境のセットアップ情報も利用します。

5. システムの運用／システムの保守と見直し

   システムの運用を開始します。システムの運用では，日常運用と定期的なシステム保守を繰り返します。

   また，業務内容の変更やシステム規模の変更などに合わせて，システムを随時見直します。見直しの結果，システムを再構築・再セットアップする場合や，各種定義およびアプリケーションを追加・変更する場合は，2.～4.の段階を実施してから運用を再開します。

サービスプラットフォームの更新または破棄を決定するまで継続します。

以降の項では，それぞれの段階の概要について説明します。

10.3.1　システム設計／サービス部品準備

サービスプラットフォームの導入を決定したら，システムで実行する業務を分析し，業務をサービス化します。既存の業務アプリケーションは，再利用性などを考慮してサービス化して利用することもできます。

新たにサービスを作成する場合は，再利用性や寿命を考慮して作成すると，より可用性のあるシステムが構築できます。

利用するサービスの選定・作成のあと，サービスの利用方法に応じたシステムを設計します。

10.3.2　システム構築

システム設計の結果を基に，開発環境，運用環境および実行環境を構築します。

(1)　開発環境の構築

開発環境を構築して，各種定義やアプリケーション開発をできる状態にします。

前提ソフトウェアと Service Architect のインストール，各種定義を行うときに利用するプラグインの組み込み，リポジトリの設定などが含まれます。

(2)　運用環境の構築

運用環境を構築して，開発環境と情報を受け渡しできる状態にします。また，運用環境から実行環境のセットアップを実行できる状態にします。

前提ソフトウェアと Service Platform のインストール，実行環境のセットアップ作業に必要な定義ファイルの作成，リポジトリの設定などが含まれます。

(3)　実行環境の構築

実行環境を構築して，開発環境および運用環境から受け渡される定義やアプリケーションをセットアップできる状態にします。

前提ソフトウェアと Service Platform のインストール，J2EE サーバの設定，データベースの設定などが含まれます。

10.3.3　セットアップ

システムの運用開始に向けて運用環境から実行環境をセットアップします。

HCSC サーバのセットアップ，開発環境で作成された定義情報の実行環境へのセットアップなどが含まれます。

ポイント

運用環境では，開発環境で作成した定義情報を実行環境にセットアップします。逆に，開発環境で各種定義を作成する場合には，運用環境で実施したセットアップの情報を利用します。

運用環境による実行環境のセットアップと，開発環境での各種定義の作成は，リポジトリを利用して運用環境と開発環境で情報の受け渡しをして実施します。

10.3.4　各種定義／アプリケーションの作成

運用環境・実行環境のセットアップや実運用の開始に必要な各種定義やアプリケーションを作成します。

サービス部品を利用するための定義（サービスアダプタ），ビジネスプロセスの定義，各種定義を実行環境に配備するための定義（配備定義），実行環境で業務の実行要求を受け付けてサービス部品に実行要求を送信するアプリケーション（サービスリクエスタ）などの作成が含まれます。

ポイント

開発環境では，運用環境で実施された実行環境のセットアップの情報を基に定義する情報があります。逆に，運用環境で実行環境をセットアップするには，開発環境で定義した情報を利用します。

運用環境による実行環境のセットアップと，開発環境での各種定義の作成は，リポジトリを利用して運用環境と開発環境で情報の受け渡しをしながら実施します。

10.3.5　システムの運用／システムの保守と見直し

開発環境での各種定義やアプリケーションの作成，および運用環境での実行環境のセットアップが完了したら，運用を開始します。

システムの運用では，日常の運用およびシステムの保守・見直しを繰り返します。

日常運用では，日常的な実行環境の起動／停止のほか，システムを安定稼働させるために各種プロセスの監視やログの収集などを行います。

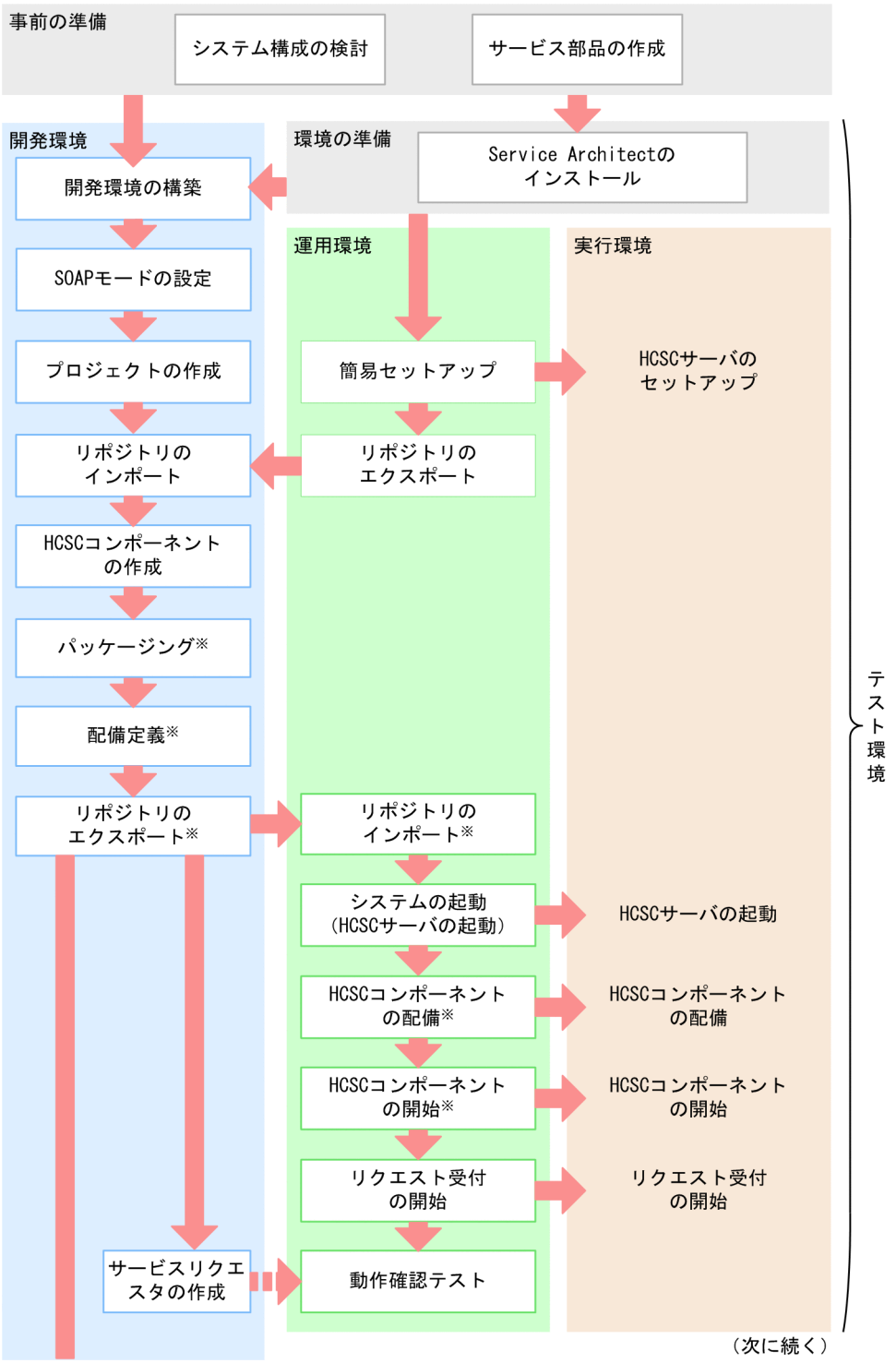
システムの保守と見直しでは，各環境の構成の変更，各種定義およびアプリケーションを追加・変更，発生したトラブルの対処などを実施します。

なお，JP1 と連携してサービスプラットフォームをより効率良く運用することもできます。

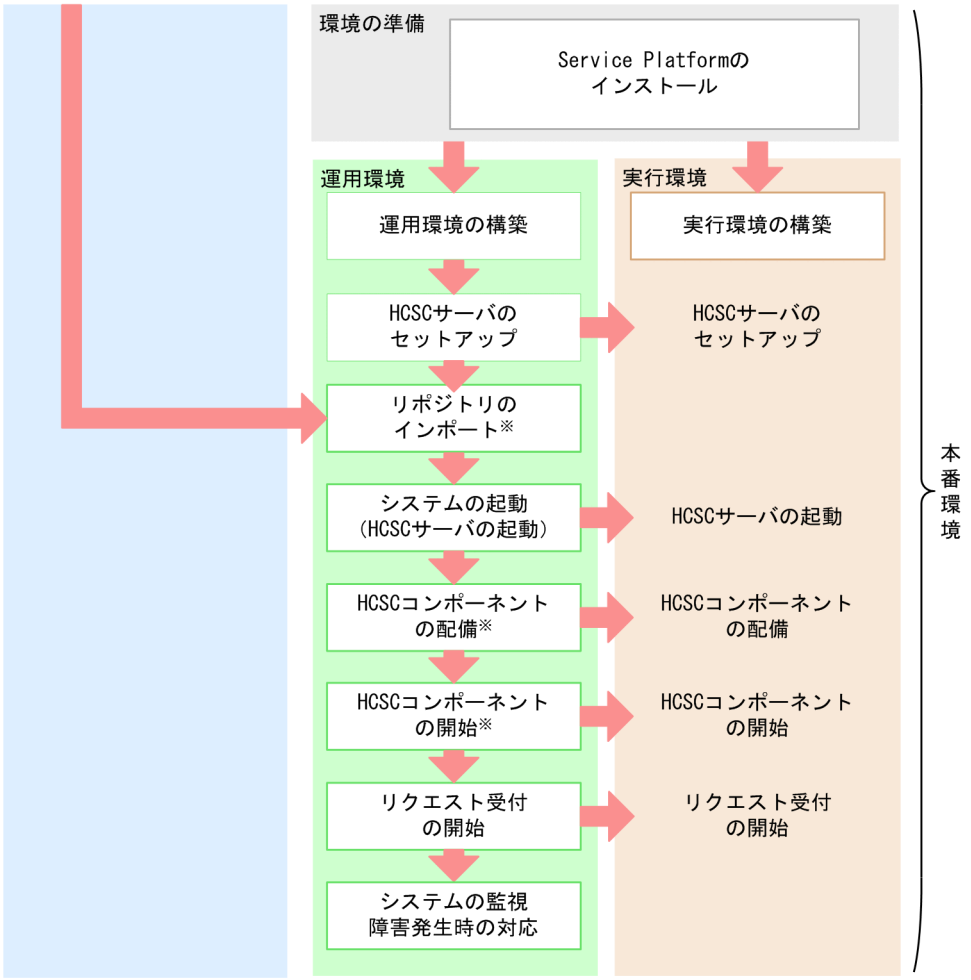
10.4　開発から実運用までの流れ

サービスプラットフォームを利用したシステムの開発から実運用までの流れを次の図に示します。

図 10–13　開発から実運用までの流れ



（続き）



注※

これらの作業を開発環境で一括して実行することもできます。ただし，一括実行は，システム開発時，または単体テストから結合テスト時を対象としています。詳細については，マニュアル「サービスプラットフォーム 開発ガイド 基本開発編」の「8.5　HCSC コンポーネントを HCSC サーバに配備して開始する処理の一括実行」を参照してください。

事前の準備作業を実施したあと，サービスプラットフォームの開発環境，運用環境および実行環境でそれぞれ必要な作業を実施します。

開発環境で必要な作業については，マニュアル「サービスプラットフォーム 開発ガイド 基本開発編」，およびマニュアル「サービスプラットフォーム 開発ガイド 受付・アダプタ定義編」を参照してください。運用環境および実行環境で必要な作業については，マニュアル「サービスプラットフォーム システム構築・運用ガイド」を参照してください。

索引

い

う

え

お

か

き

く

け

よ

ら

り

れ

ろ