

OSI 開発者ガイド

OpenFrame OSI 7.2

TMAXSOFT

Copyright Notice

Copyright 2025. TmaxSoft Co., Ltd. All Rights Reserved.

Company Information

TmaxSoft Co., Ltd.

TmaxSoft Tower, 45 Jeongja-ilro, Bundang-gu, Seongnam-si, Gyeonggi-do, South Korea

Website: <https://www.tmaxsoft.com>

Restricted Rights Legend

このソフトウェア (Tmax OpenFrame®) マニュアルの内容とプログラムは、著作権法および国際条約によって保護されています。マニュアルの内容とプログラムは、TmaxSoft Co., Ltd.との使用許諾契約書の下でのみ使用することができ、マニュアルは使用許諾契約で許可されている範囲を除いては、配布または複製することができません。TmaxSoftの書面による事前の承諾を得ることなく、このマニュアルの全部または一部を電子的または機械的な方法を問わず、転送、複製、配布したり、二次的著作物を作成したりする等の行為を一切禁じます。

このソフトウェアのマニュアルとプログラムの使用許諾契約は、いかなる場合においても、マニュアル及びプログラムと関連する知的財産権（登録の有無を問わず）を譲渡するものと解釈されず、TmaxSoftのブランド、ロゴ、商標等の使用権限を与えるものではありません。このマニュアルは、情報を提供する目的でのみ提供しており、これに伴う契約上の直接的ないしは間接的な責任を負わず、マニュアルの内容は法律上もしくは商業的な特定の条件が満たされることを保証しません。マニュアルの内容は、製品のアップグレード及び修正により、その内容が予告なく変更されることがあり、内容上の誤りがないことを保証しません。

Trademarks

Tmax®とTmax OpenFrame®はTmaxSoft Co., Ltd.の登録商標です。本書に記載されている会社名、製品名などは、各社の商標または登録商標です。本書では、製品名、会社名などに必ずしも商標表示(™, ®)を付記していません。

Open Source Software Notice

この製品には、OpenSSL、RSA Data Security, Inc.、Apache FoundationおよびJean-loup Gaillyと Mark Adlerによって開発またはライセンス取得されたオープンソフトウェアが含まれています。詳細は、製品ディレクトリの\${INSTALL_PATH}/license/oss_licensesに記載されている事項を参照してください。

発行履歴

| 製品バージョン | ガイドバージョン | 発行日 | 備考 |
|-------------------|----------|------------|----|
| OpenFrame OSI 7.2 | 3.2.1 | 2025-08-14 | |
| OpenFrame OSI 7.2 | 3.1.2 | 2023-12-29 | |
| OpenFrame OSI 7.2 | 3.1.1 | 2023-08-30 | |

目次

| | |
|---------------------------|----|
| 1. 紹介 | 1 |
| 1.1. 概要 | 1 |
| 1.2. データベース | 1 |
| 1.3. データ通信 | 1 |
| 1.4. アプリケーション | 1 |
| 2. アプリケーション・インターフェース | 4 |
| 2.1. 概要 | 4 |
| 2.2. DL/Iインターフェース | 4 |
| 2.3. AIBTDLIインターフェース | 4 |
| 2.4. データベース呼び出し関数 | 4 |
| 2.5. データ通信呼び出し関数 | 6 |
| 3. アプリケーションの作成 | 9 |
| 3.1. アプリケーションの作成手順 | 9 |
| 3.1.1. アプリケーションの設計 | 9 |
| 3.1.2. プログラミング | 9 |
| 3.1.3. バッチ・アプリケーション | 10 |
| 3.2. MPPアプリケーション | 10 |
| 3.2.1. MPPユーザー・サーバーの準備 | 11 |
| 3.2.2. MPPプログラミング | 12 |
| 3.2.3. メッセージ・セグメント・フォーマット | 12 |
| 3.3. BMPアプリケーション | 14 |
| 付録 A: DL/Iステータス・コード | 16 |
| 付録 B: IO-PCBマスク | 20 |
| 付録 C: AIBマスク | 21 |
| 付録 D: DL/I呼び出し | 22 |
| D.1. システム・サービスDL/I呼び出し | 22 |
| D.1.1. CHKP(基本)呼び出し | 22 |
| D.1.2. CHKP(シンボリック)呼び出し | 22 |
| D.1.3. INIT呼び出し | 23 |
| D.1.4. INQY呼び出し | 23 |
| D.1.5. LOG | 25 |
| D.1.6. ROLB | 25 |
| D.1.7. SYNC | 25 |
| D.1.8. XRST | 26 |
| D.2. トランザクション管理 DL/I呼び出し | 26 |
| D.2.1. CHNG | 26 |
| D.2.2. CMD | 27 |
| D.2.3. GCMD | 27 |
| D.2.4. GN | 27 |

D.2.5. GU 28
D.2.6. ISRT 28
D.2.7. PURG 29

1. 紹介

本章では、OpenFrame OSIアプリケーション・プログラムの概念と特徴的な構造について説明します。

1.1. 概要

開発者は、OpenFrame OSI(以下、OSI)システムで運用される新しいアプリケーションを開発するか、既存のIMS/DCで運用していたアプリケーションをマイグレーションしてトランザクション・サービスを構成することができます。OSIシステムで提供されるほとんどのリソースは、IMS/DCが提供する機能やサービスを同様に提供しています。

OSIプログラミングは、**DL/I**関数に基づいて各関数の機能を組み合わせて業務プログラムを構成します。DL/I関数の種類と機能の詳細については、[アプリケーション・インターフェース](#)を参照してください。

1.2. データベース

IMS/DCシステムが使用するデータベースは、階層型データベース管理システムであり、クライアントの業務データを構造的に保存および管理します。このデータベースは、IMS/DBと呼ばれており、IMS/DCで実行されるアプリケーション、または純粋なバッチ・アプリケーションがこのIMS/DBにアクセスするには、標準DL/I関数を呼び出す必要があります。

アプリケーションは、各プログラミング言語が提供する標準アプリケーション・インターフェースである**DL/I**を介して標準DL/I関数を使用することができます。

OSIでは、UNIXでデータベース構造を再実装し、OpenFrame HiDBと連携してIMS/DCと同じ機能を提供します。

1.3. データ通信

IMS/DCは、ユーザーが端末を介してアプリケーションが処理したデータの結果を確認できるように、システム・サーバー(制御領域)とユーザー・サーバー(従属領域)を提供します。

OSIは、Tmaxとの連携を通じてIMS/DCの機能を提供し、トランザクションを効率的に処理します。また、IMS/DCで運用中のクライアントのプログラムをOSIでも同様に運用できるよう、IMS/DCで提供するDL/Iインターフェースと同じインターフェースを提供します。

DL/Iインターフェースを使用して、ユーザー・サーバー(従属領域)のMPP、BMPで実行されるユーザー・プログラムを簡単にマイグレーションし、OSIで実行することができます。

1.4. アプリケーション

OSIは、IMS/DCで実行されていたアプリケーションをIMS/DCでも実行できるように、IMS/DCと同じ機能と基盤構造を提供しています。

IMS/DCアプリケーションは、非IMS/DCアプリケーションと違って、PSB(プログラム仕様ブロック)が必要です。PSBは、IMSがアプリケーションに提供するサービスを実行するためのインターフェースとして機能します。これらのサービスには、メッセージを端末に送信する方法、データベースにアクセスする方法、IMSコマンド、IMSサービス呼び出しが含まれます。OSIアプリケーションは、PSBを介してIMSサービスを実行する方法を決定できます。

PSBとアプリケーションは、OSIシステムによって同時にロードされます。OSIの実行モジュールは、アプリケーション内で実行される呼び出し関数を使用して、アプリケーションが要求する機能を実行します。各MPPアプリケーションは、MPPタイプのユーザー・サーバーを通じてOSI機能(DL/I呼び出し関数、コマンドなど)を実行することができます。そのため、必要なシステム定義が事前に登録されている必要があります。システムは、IMS/DCが使用したシステム・マクロを使用して定義できます。

バッチ・アプリケーションは、上記の場合とは異なる方法で実行されます。OSIシステムでバッチ・アプリケーションを実行するには、BMP(Batch Message Processing Region)サーバーが必要です。このサーバーは、バッチJOB(JCL開始)を介して要求されるアプリケーションの関数を処理できます。

アプリケーションを作成する際、ENTRY文にプログラムで使用するPCBを定義します。その定義されたPCBを通じてOSIシステムの機能を使用できます。PCBは、アプリケーションで使用されるOpenFrame HiDBに関するビューやメッセージ・ソース、またはメッセージ宛先と通信するためにシステムが提供する制御ブロックです。OSIではPCBを400個まで使用できます。

以下は、アプリケーションのPCBについての説明です。

- Database Program Control Block (DB-PCB)

データベースに格納されている情報をアプリケーションが読み取る方法について説明します。同じデータベースでも、DB-PCBでの記述方法に応じて異なる方法で読み取ることができます。

- IO Program Control Block (IO-PCB)

OSIは基本的に端末を使用して動作します。オンラインで実行されるアプリケーションにはIO-PCBが必要です。

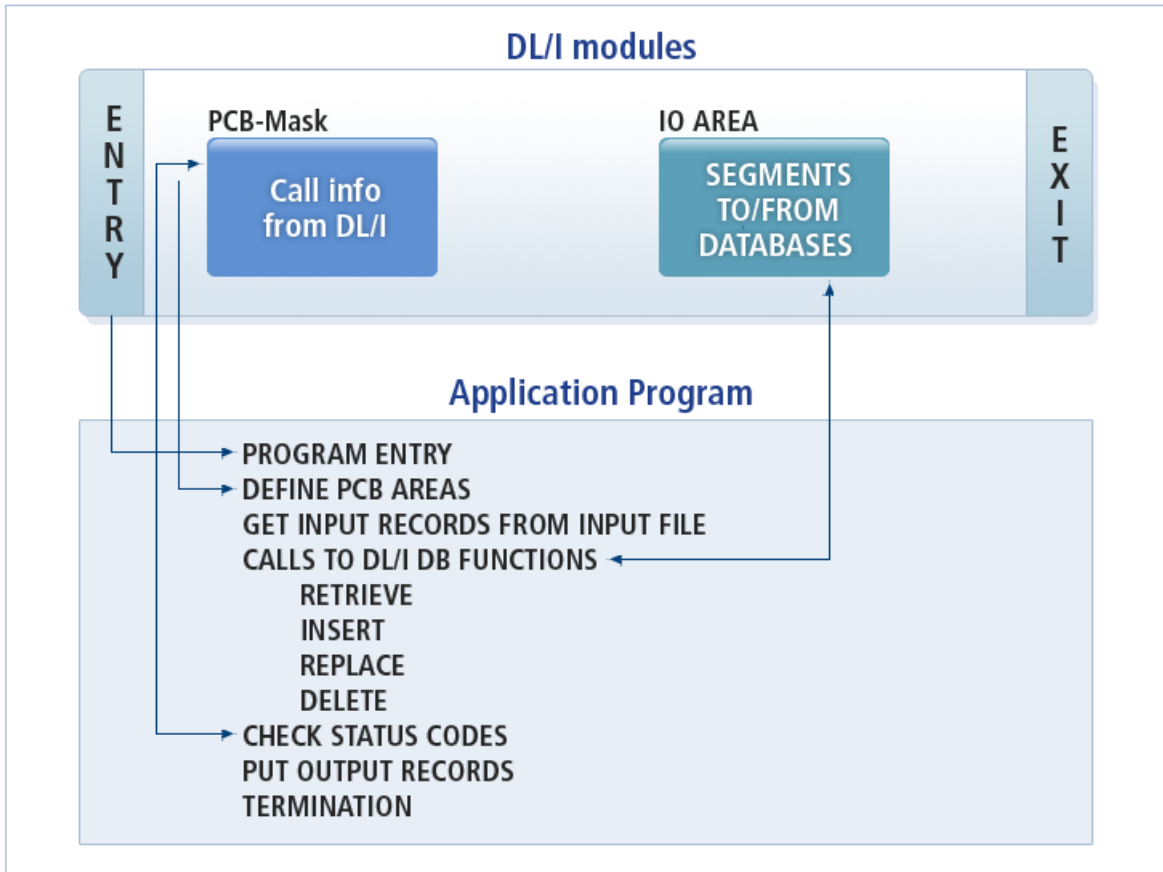
- Alternative Program Control Block (ALT-PCB)

アプリケーション内で他のプログラムまたは端末にメッセージを送信する場合は、ALT-PCBを使用します。ALT-PCBは、宛先を変更して使用できます。



PSBとDBDの構文については、OpenFrame HiDB『HiDBガイド』を参照してください。

以下の図は、アプリケーションの一般的な動作フローを示します。



アプリケーションの動作フロー

2. アプリケーション・インターフェース

本章では、OpenFrame OSI環境で作成されたCOBOLプログラムが使用するインターフェースの種類と使用方法について説明します。

2.1. 概要

OSI環境では、アプリケーションがデータベースまたはMQにアクセスするには、決められたインターフェース(DL/I呼び出し関数)に従う必要があります。

各関数は、データベースだけでなく、OSI領域(オンライン・プログラム)でも使用されます。使用方法はデータベースの場合と同じです。ただし、SSA(Segment Search Argument)などの特定のオプションは、オンラインでは使用されません。オンラインでは、データベースをMQ(Message Queue)と見なし、関数の機能を代入してください。

インターフェースに従わない場合は、ユーザーが認識できるステータス・コードを戻り値として返します。ステータス・コードについては、[DL/Iステータス・コード](#)を参照してください。



DL/I呼び出し関数については、OpenFrame HiDB『HiDBガイド』を参照してください。

2.2. DL/Iインターフェース

アプリケーションがDL/I呼び出し関数を使用するには、DL/Iインターフェースを使用する必要があります。アプリケーションの言語によって、CBLTDLI、PLITDLI、ASMTDLIなど、さまざまなDL/Iインターフェースを使用することができます。

以下はDL/Iインターフェースの使用方法です。argument 1にはDL/I呼び出し関数名を、argument 2にはPCBを指定します。

```
CALL 'CBLTDLI' USING argument 1, argument 2, ..., argument n
```

2.3. AIBTDLIインターフェース

AIB(Application Interface Block)は、PCBのアドレスが不明な場合の代替として使用するか、またはDL/I呼び出し関数がPCBを必要としない場合に使用できます。DL/I呼び出し関数でPCBが必要な場合は、AIBのリソース名とpsbgen実行時に登録したPCB名が同一でなければなりません。

以下は、アプリケーションがAIBTDLIインターフェースを介してDL/I呼び出し関数を使用する方法です。argument 1にはDL/I呼び出し関数を、argument 2にはPCBの代わりにAIBを指定します。

```
CALL 'AIBTDLI' argument 1, argument 2, ..., argument n
```

2.4. データベース呼び出し関数

以下は、アプリケーションがデータベースにアクセスするために使用する関数の使用方法です。

- 使用方法

Function Code(argument 1) PCB(argument 2) データ領域(argument 3) SSA(argument 4)

- 引数

- argument 1

DL/I関数を指します。関数コードはデータベース・セグメントの処理種類を決定するコードであり、このフィールドは4バイト長の領域としてアプリケーション内で宣言されます。

以下は、argument 1に指定できる関数です。

| 関数 | 説明 |
|---|--|
| GU(GET UNIQUE) / GHU(GET HOLD UNIQUE) | データベースの特定セグメントを検索する関数コードです。SSA(引数の4番目からn番目まで指定)を指定してセグメントを取得できます。 オンライン・プログラムでは、MQ内のメッセージを読み取るように指示します。 |
| GN(GET NEXT) / GHN(GET HOLD NEXT) | 現在配置されているセグメント(呼び出し直前に検索されたセグメント)に続くセグメントを検索する関数コードです。データベース・セグメントを階層順に検索する場合に使用します。 オンライン・プログラムでは、MQはデータベースとして機能するため、関数の機能もデータベースではなく、MQを対象としています。 |
| GNP(GET NEXT WITHIN PARENT) / GHNP(GET HOLD NEXT WITHIN PARENT) | データ構造の特定のセグメント・タイプを親とし、そのセグメントの下位レベルにあるセグメントを含むことができます。これをファミリー・セグメントといいます。GNPとGHNP関数コードは、このファミリー・セグメントの階層順に検索されます。 オンライン・プログラム(IO-PCB)を使用する場合、この関数は使用されません。 |
| ISRT(INSERT) | argument 3に入力されたデータを、指定されたデータベースまたはMQに挿入します。 |
| DLET(DELETE) | セグメントを削除する場合に使用する関数コードです。 セグメントを削除する場合は、ホールド系の検索呼び出しを使用してセグメントを配置した後、DLET呼び出しを使用します。削除するセグメントは、削除規則に従ってデータベースから削除されます。 |
| REPL(REPLACE) | セグメントを置き換える場合に使用する関数コードです。DLET呼び出しと同様に、ホールド系の検索呼び出しを使用してセグメントを配置した後、REPL呼び出しを使用します。REPL呼び出しを使用してセグメントのキーを変更しないでください。 オンライン・プログラム(IO-PCB)を使用する場合、この関数は使用されません。 |

- argument 2

アプリケーションがデータ・アクセスとインターフェイスを使用するには、PCBマスクをDB-PCB、IO-PCB、ALT-PCBのいずれかで定義します。PCBは、アプリケーションがCOBOLで作成された場合は、LINKAGEセクションに定義します。アプリケーションに定義されたPCBが呼び出し機能を実行する場合は、OSI内のCONTROLセクションとリンクされます。

| PCB Mask | 説明 |
|----------|---|
| DB-PCB | データベース呼び出し機能が定義されたPCBを DB-PCB といいます。 |
| IO-PCB | オンライン機能を使用する場合に使用します。 |
| ALT-PCB | 他のトランザクションまたは端末に送信する場合に使用します。 |

◦ argument 3

アプリケーション内のデータ領域を指定します。アプリケーションとデータ・アクセスのデータ処理は、データ領域に指定して使用します。ユーザーはセグメントの長さでSSAのコマンド・コードを使用して、必要な長さをアプリケーション内のXX領域に宣言する必要があります。

◦ argument 4

SSAの処理対象となるセグメントとSensitiveセグメントの条件を定義します。SSAの先頭の名前を、データ構造で定義した階層レベルに従って指定します。nの値は最大値は18です。IO-PCBマスクを使用する場合は、MOD名を設定することで、端末に表示される画面を変更することができます。

SSAは、データベース呼び出し機能の特殊な引数であり、セグメントに転送される条件の存在に応じて、unqualified SSA(非修飾SSA)とqualified SSA(修飾SSA)に分けられます。

| SSA | 説明 |
|-----------------|--|
| unqualified SSA | セグメント名を使用してSSAを処理し、セグメント内のフィールド情報に条件を付与しません。 |
| qualified SSA | セグメントのフィールド情報に条件を付与します。 |



データベース呼び出し関数の詳細については、『IBM IMS Application Programming:Database Manual』を参照してください。

2.5. データ通信呼び出し関数

以下は、データ通信呼び出し関数の機能です。

- メッセージ・セグメントの送受信
- メッセージの宛先の変更
- SPA(Scratch Pad Area)の送信

データ通信呼び出し関数は、データベース呼び出し関数と機能は同じです。ただし、オンラインでMQに蓄積されているメッセージとSPA領域のメッセージは、階層構造(OpenFrame HiDBが持つ特徴的な構造)を持たないため、階層的な検索条件である、データベース呼び出しのSSAは不要です。



SPA(Scratch Pad Area)についての詳細は、『IBM IMS Conversational Transactions』を参照してください。

- 使用方法

- 引数

- argument 1

関数コードを指定する引数です。関数コードは、メッセージ・セグメントとSPA処理のタイプを指定するコードです。このフィールドは、データベース呼び出し機能と同様に、4バイト領域でアプリケーション内で宣言します。

以下は、argument 1に指定できる関数です。

| 関数 | 説明 |
|----------------|--|
| GU(GET UNIQUE) | メッセージの最初のセグメント(ヘッダー・セグメント)を受信します。アプリケーションはメッセージを受信する際、GUを最初に呼び出す必要があります。 会話モードで処理される場合は、SPAが受信されます。 |
| GN(GET NEXT) | ヘッダー・セグメントに続くメッセージ・セグメントを受信します。 |
| ISRT(INSERT) | メッセージ・セグメントまたはSPAを送信します。 |
| PURG(PURGE) | ISRTによって送信されたメッセージ・セグメントを1つのメッセージとして端末に出力します。 |
| CHNG(CHANGE) | ALT-PCBを使用して、他の端末またはトランザクションにメッセージを送信します。 |

- argument 2

データ通信呼び出し関数のためのPCBは、データベースPCBと同じ方法で定義します。これを**論理端末PCB**といます。

以下は、argument 2に指定できる論理端末PCBのタイプです。

| PCB | 説明 |
|-------------------------|--|
| IO-PCB | トランザクションを要求した論理端末のメッセージを入出力するために使用します。 |
| Alternative PCB | デフォルト端末以外の他の端末にメッセージを送信するために使用します。 |
| Alternative Program PCB | メッセージを他のプログラムに送信するために使用します。 |



- Alternative Program PCBはALT-PCBと呼ばれます。アプリケーション内における論理端末PCBは、IO-PCB、ALT-PCBの順で定義します。
- IO-PCBはOSIシステムで自動的に管理され、ALT-PCBはプログラム定義ユーティリティとして定義します。

- argument 3

メッセージ・セグメント、コマンド、SPAの送受信のために使用されるデータ領域です。

- argument 4

MFSを実行する端末には、送信するメッセージの形式を、事前にメッセージ・フォーマット定義ユーティリティで定義しておいたMOD名(8バイト)が設定されている領域のアドレスを設定します。MOD名はISRTを行う場合にのみ指定します。



データ通信呼び出しの詳細については、『IBM IMS Application Programming: Transaction Manager Manual』を参照してください。

3. アプリケーションの作成

本章では、アプリケーション・プログラムを作成する方法について説明します。

3.1. アプリケーションの作成手順

アプリケーションを作成する手順は、アプリケーションの設計、プログラミング、デバッグ、実行に分けられます。

各ステップには、OSIで定義する必要があるリソースと、ユーザーが任意で作成するプログラミングが含まれます。

3.1.1. アプリケーションの設計

アプリケーションの設計は、以下の手順で行われます。

1. 開発者は、アプリケーションが実行されるユーザー・サーバーのタイプを決定する必要があります。

アプリケーションを処理するユーザー・サーバーには、MPPユーザー・サーバーとBMPユーザー・サーバーがあります。この2つのサーバー・タイプからアプリケーションが実行される方法を選択します。

2. サーバーのタイプが決定したら、アプリケーションとユーザーサーバー間の関係を定義する必要があります。

アプリケーションとユーザー・サーバーの関係は、システム・リソース定義(System Resource Definition)で定義します。

以下は、システム定義の例です。

```
APPLCTN PSB=OIVPI001,PGMTYPE=(, ),SCHDTYP=PARALLEL
TRANSACT CODE=OIVPMPP1,MSGTYPE=(SNGLSEG,RESPONSE,1),PRTY=(1,5),      X
      MODE=SNGL
APPLCTN PSB=OIVPI002,PGMTYPE=(TP, ,1),SCHDTYP=PARALLEL
TRANSACT CODE=OIVPMPP2,MSGTYPE=(SNGLSEG,RESPONSE,1),PRTY=(1,5),      X
      MODE=SNGL,MAXRGN=1

APPLCTN PSB=OIVPIL05,PGMTYPE=BATCH,SCHDTYP=PARALLEL
TRANSACT CODE=OIVPBMP5,MSGTYPE=(SNGLSEG,RESPONSE,1),PRTY=(1,5),      X
      MODE=SNGL
```

3. アプリケーションを設計する際は、メンテナンスとスケーラビリティを考慮したプログラムの構造やモジュールの設計が必要です。アプリケーションを設計するときは、次の点に注意してください。

- DL/I関数の呼び出しを効果的に使用します。
- データの入出力回数を最小限に抑えます。
- データベースのデータ構造と処理オプション(検索、追加、削除)を考慮します。
- 処理モード(会話モード処理と一般トランザクション・モード)の設定を考慮します。
- メッセージのパスを考慮します。

3.1.2. プログラミング

COBOLでOSIアプリケーションを作成し、以下に注意してください。

- プログラムの開始条件の設定
- DB-PCB、IO-PCB、ALT-PCBの設定
- 関数呼び出しインターフェースの設定
- ステータス・コードのチェックとエラー処理の設定
- データ処理時の整合性 (例：データ長)
- プログラムの終了条件の設定 (例：リターン・コード)
- 入出力データ・サイズまたはSSAサイズに関係なく、プログラミング時のデータ転送領域は十分に大きくなければなりません。

3.1.3. バッチ・アプリケーション

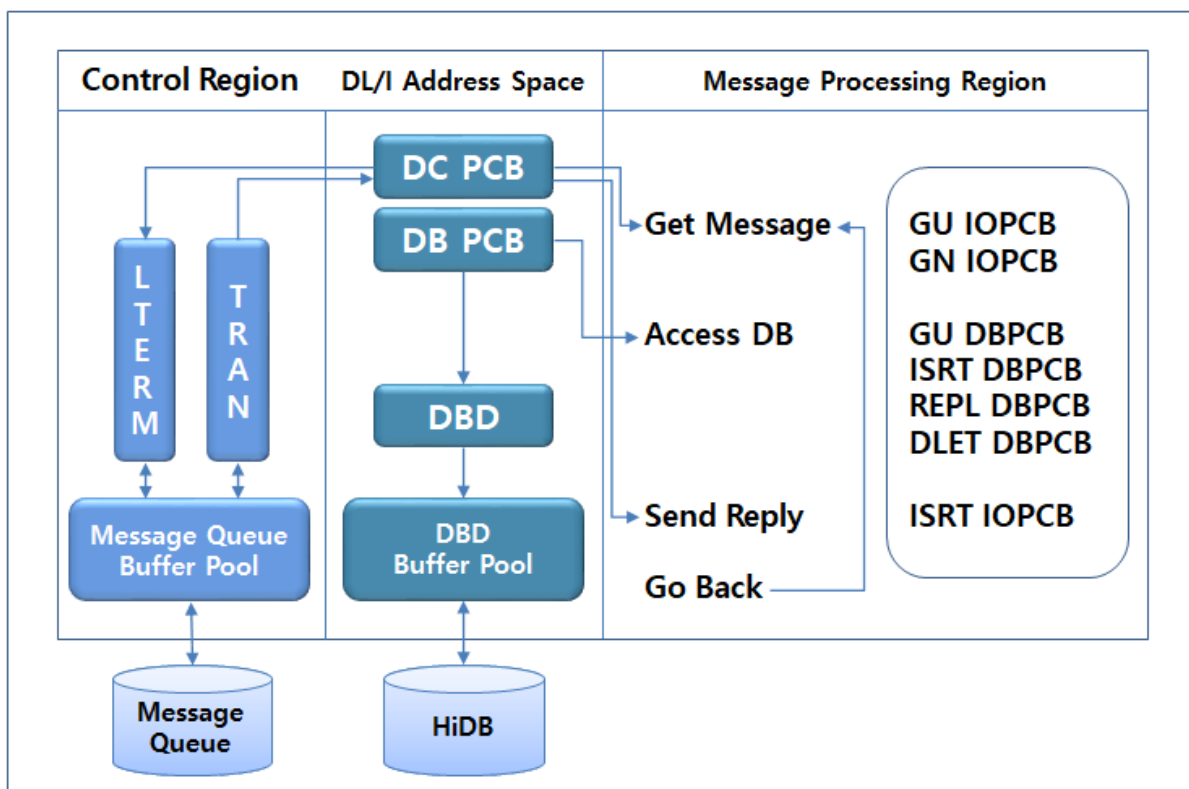
バッチ・アプリケーションは、OpenFrameのユーティリティである**tjesmgr**によって実行されます。バッチ・アプリケーションの作成方法は、基本的にMPPアプリケーションと同じですが、JCLで実行されるという点が異なります。tjesmgrは、JCLを効果的に実行できるユーティリティです。

バッチ・アプリケーションはオンライン業務のためのプログラムではないため、端末との通信のためのデータ領域を定義する必要はありません。プログラムがデータベースに接続できるロジックとMQ(Message Queue)でメッセージをロードできるロジック、またはMQにデータが蓄積できるロジックのみ必要です。

3.2. MPPアプリケーション

IMS/DCオンライン環境でメッセージを処理するプログラムを、MPP(Message Processing Program)またはMPR(Message Processing Region)と呼びます。OSIでは、それらをシステム・サーバーとユーザー・サーバーに区別します。

以下の図は、オンライン環境でのOSIシステムとアプリケーションの構成を示しています。



OSIシステムとアプリケーションの構成

通常、端末(LTERM)から入力されたトランザクションは、システム・サーバー(Control Region)によって分析され、そのトランザクションを処理するMPPアプリケーションは、OSIのユーザー・サーバーであるMPPユーザー・サーバーにロードされ実行されます。このトランザクションがOSIシステムに渡すデータ領域は、トランザクション・コード(最大8文字)とデータ領域で構成されます。システム・サーバーは、受け取ったトランザクション・コードを使用してRTSD(RunTime System Definition)表からクラス情報を取得し、クラスを処理するMPPユーザー・サーバーにスケジューリングします。

MPPアプリケーションはGet Unique(GU)呼び出しで端末から入力されたメッセージを取得できます。複数のユーザー・サーバーを起動して複数のMPPアプリケーションを処理することができ、データベースに同時接続することも可能です。LTERMだけでなく、トランザクション間にも、ALT-PCBを使用してMPPアプリケーションを開始できます。

3.2.1. MPPユーザー・サーバーの準備

MPPユーザー・サーバーの起動には、Tmax環境ファイルの設定、MPPサーバー起動用のJCLファイルの準備が必要です。OSIでは、Tmaxのtargetオプションを使用することで、OSIMPPSVRバイナリ・ファイル1つで、相互異なるクラス情報を処理する複数のMPPサーバーを起動することができます。

• Tmax環境ファイルの設定

OSIはTmax環境で動作するため、ユーザー・サーバーもTmaxサーバーに登録する必要があります。サーバーの登録方法は、Tmaxでサーバーに登録する一般的な方法と同じです。Tmaxサーバーに登録するときは、関連サービスも一緒に登録する必要があります。

```
#####
#   OpenFrame OSI Dependent Region Servers                               #
#####
OSIMPPSVR      SVGNAME = svg_node1, MIN = 1, MAX = 10, RESTART = NO
#####
#   OSI, example in which IMSID is IMSA                                 #
#####
IMSASCHD      SVGNAME = svg_node1, MAX = 1, SVRTYPE = UCS, RESTART = NO,
              TARGET = osisschd
IMSACMMD      SVGNAME = svg_node1, MAX = 1, SVRTYPE = UCS, RESTART = NO,
              TARGET = osicmsdv
IMSAMPP_TCL1  SVGNAME = svg_node1, MIN = 1, MAX = 10, RESTART = NO,
              TARGET = OSIMPPSVR
IMSAMPP_TCL2  SVGNAME = svg_node1, MIN = 1, MAX = 10, RESTART = NO,
              TARGET = OSIMPPSVR
IMSAMPP_TCL3  SVGNAME = svg_node1, MIN = 1, MAX = 10, RESTART = NO,
              TARGET = OSIMPPSVR
IMSAMPP_TCL4  SVGNAME = svg_node1, MIN = 1, MAX = 10, RESTART = NO,
              TARGET = OSIMPPSVR

*SERVICE
#####
#   OSI USER APPLICATION SERVER DEFAULT                               #
#####
OSIMPPSVRSVC  SVRNAME = OSIMPPSVR, SVCTIME=60
```

• MPPサーバー起動用のJCLファイルの準備

以下は、システム・サーバー(制御リージョン)名がIMSAの場合、管理者が作成したユーザー・サーバー(従属リージョン)に該当するMPPユーザー・サーバーを起動するJCLファイルです。クラスを指定しない場合は「000」で設定可能であり、最初のクラスの指定は必須です。(例で処理するクラスは、1、2、3、4です)

```
//IMSMSG JOB
//STEP1 EXEC PGM=DFSRR00,
```

```
//          PARM='MSG,001002003004,W00099000,,,R,,,,,IMSA,,,,,D2PA'  
//STEPLIB DD DISP=SHR,DSN=OSI.IMSA.STEPLIB  
//SYSPRINT DD SYSOUT=*  
//SYSOUT DD SYSOUT=*  
//SYSDBOU DD SYSOUT=*  
//
```

3.2.2. MPPプログラミング

基本的なMPPプログラミングは、GUとISRT呼び出しで構成されます。

メッセージをGU処理する場合は、IO-PCBを使用してDL/I関数を呼び出します。メッセージが1つ以上のセグメントで構成されている場合は、2番目以降のセグメントは、GN呼び出しでメッセージを読み込む必要があります(SPAを使用するトランザクションの場合には、GUでSPAを読み込み、その後、GNでメッセージを読み込みます)。GUとGN呼び出しの結果は、IO-PCBのステータス・コード・フィールドに設定されます。MPPプログラムは呼び出し処理後に必ずステータス・コードをチェックする必要があります。メッセージ・セグメントを端末に送るか、他のトランザクションに送る場合は、データ領域にメッセージ・セグメントを指定のフォーマットに設定して、ISRT呼び出しでDL/I関数を呼び出します。

他のMPPプログラム、または他の端末に、メッセージを送ることができますが、メッセージを送信する際には、ALT-PCBを使用して宛先を指定する必要があります。

注意事項

以下は、MPPプログラミング時の注意事項です。

- トランザクションが入力されるたびにスケジューリングされ、MPPが動的に空のユーザー・サーバーにロードされます。MPPの処理後、使用していたユーザー・サーバーは他のMPPプログラムに割り当てられるか、IDLE状態になるため、MPPユーザー・サーバー内に情報を保存することは不可能です。データの保存が必要な場合はSPAを使用します。
- MPPの実行のためにOSIが行う事前処理が多いので、一度MPPがロードされると、多数のメッセージを処理するプログラムを作成します。しかし、メッセージの処理に多くの時間を費やしてしまうと、システムの定義時に指定した優先順位に従って均一化した、トランザクション・スケジューリングができなくなります。このような場合は、システム定義でTRANSACTマクロのPROCLIMの指定を活用します。
- MPPでデータベースを更新する場合は、データベースに対する処理も行うため、注意が必要です。アプリケーションを必要以上に大きくしてデータベースのアクセスを独占しないように注意する必要があります。
- 均等なスケジューリング・サービスを行うために、1つのトランザクション処理において多くの時間を要するMPPは作成せずに、複数のMPPに分割するようにプログラミングします。
- MPPは処理中にステータス・コードをチェックし、問題がある場合はそれ以上処理しません。データベースの処理中にデータベースのバックアウトが必要な場合は、異常終了(ABEND)する必要があります。
- OpenFrame環境設定のhidbサブジェクトのHIDB_DEFAULTセクションのUSE_DBD_DBMS_LOCKキーの値がYESに設定されている場合、トランザクションが開始される際、トランザクションが使用するDBD名でロックが生成され、トランザクションが終了されるときに解除されます。詳細については、『OpenFrame 環境設定ガイド』を参照してください。

3.2.3. メッセージ・セグメント・フォーマット

以下は、受信メッセージ・セグメント、送信メッセージ・セグメント、プログラム間のメッセージ・セグメントについての説明です。

3.2.3.1. 受信メッセージ・セグメント

ユーザー・プログラム(アプリケーション)では、端末や他のトランザクションからメッセージが渡されます。この際に、DL/I関数のGUおよびGN呼び出しを使用し、COBOLに定義したデータ領域に読み込んだデータを受信します。

一般的に、アプリケーションが受信したメッセージ・セグメントのフォーマットは、以下のとおりです。

LL ZZ DATA

| 機能 | 説明 |
|------|--|
| LL | LLおよびZZフィールド長を含むメッセージ・セグメントの長さです。2バイトの2進数です。 PL/Iで作成されたアプリケーションの場合は4バイトであり、(実際の長さ-2)の長さで設定されます。 (LL = DATA長+4) |
| ZZ | データ・アクセスのための2バイトの領域です。このフィールドは開発者が設定する必要のないフィールドであり、任意で変更するとエラーが発生します。 |
| DATA | 端末から入力されたメッセージ・セグメントです。メッセージ・セグメントはトランザクション・コードとデータ領域で構成されます。端末に送信されたデータは、先頭の8バイトがトランザクション名になり、残りが実際のデータになります。 複数のメッセージ・セグメントで入力されるメッセージの場合は、最初のメッセージ・セグメントの先頭8バイト内にトランザクション・コードが含まれます。 |

3.2.3.2. 送信メッセージ・セグメント

アプリケーションから、端末や他のトランザクション、あるいは他の端末にメッセージを送る場合は、DL/I関数のISRT呼び出しを使用し、IO-PCB、またはALT-PCBに設定されている宛先にメッセージを送信します。

以下は、送信メッセージ・セグメントのフォーマットです。

LL ZZ DATA

| フィールド | 説明 |
|-------|---|
| LL | LLおよびZZフィールド長を含むメッセージ・セグメントの長さです。2バイトの2進数です。 PL/Iで作成されたアプリケーションの場合、LLは4バイトです。アプリケーションでは、このフィールドを設定する必要があります。(LL = DATA長 + 4) |
| Z1 | データ・アクセスのための1バイトの領域です。アプリケーションでは、このフィールドに2進数で0を入力する必要があります。 |
| Z2 | 1バイトで論理ページングを行うときに、新しいページの開始をMFSに通知する必要がある場合は、X'40'を入力します。それ以外の場合は、2進数で0を入力します。詳細については『OpenFrame OSI MFSリファレンスガイド』を参照してください。 |

| フィールド | 説明 |
|-------|--|
| DATA | 特定の宛先(現在の端末、他のトランザクション、他の端末)に送信するデータ領域です。メッセージ・セグメントの長さは宛先によって異なります。メッセージは、複数のメッセージ・セグメントも問題ありません。 |

3.2.3.3. プログラム間のメッセージ・セグメント

特定のプログラムから他のプログラムにメッセージが送信できます。プログラム間でメッセージを送信する場合、上記の送信メッセージ・セグメントと同じ方法で送信します。

以下は、プログラム間で送信するメッセージ・セグメントのフォーマットです。

```
LL ZZ DATA
```

| フィールド | 説明 |
|-------|----------------------------|
| LL ZZ | 送信メッセージ・セグメントのフォーマットと同じです。 |
| DATA | ターゲット・プログラムにメッセージを送信します。 |

注意事項

以下は、プログラム間で送信されるメッセージ・セグメントを使用する場合の注意事項です。アプリケーションで使用するALT-PCBは、事前にPSBに定義されている必要があります。また、アプリケーションで使用するPCB(LINKAGE SECTIONおよびENTRY文)は、PSBに定義されたPCBと同じ順序で定義する必要があります。

- アプリケーションAP1から他のアプリケーションAP2にメッセージを送信する場合はALT-PCBを使用し、ALT-PCBの宛先をAP2の名前として設定した後、CHNG、ISRT、PURGの順にDL/I関数を使用します。
- AP2でメッセージを受信する場合は、IO-PCBを指定してメッセージを受信できます。正常にデータ(データ構造の整合性)を取得するには、IO-PCBにAP1のALT-PCBの内容が設定されている必要があります。
- AP2で受信メッセージを処理した後、メッセージをAP1に再送信する場合はAP1のALT-PCBを指定します。この場合、AP1はAP2からメッセージを受信して処理します。メッセージを端末に出力する場合は、IO-PCBを指定してメッセージを送信します。AP2がメッセージを受信できるのは、MPPまたはBMPの場合です。

3.3. BMPアプリケーション

MPPとは異なり、ユーザーはデータベースにアクセスするだけでなく、データ管理を使用してOSIからユーザー・データセットにアクセスすることができます。データベースまたはメッセージを処理する方法は、MPPと同じです。MPPプログラミングとの違いは、MQにロードされているデータをリアルタイムで処理するのではなく、ユーザーが必要なときにJCLを介してBMPアプリケーションを起動し、処理することです。

たとえば、MPPアプリケーションによって処理された結果を、ALT-PCB(Alternative Program PCB)を使用してBMPアプリケーションに送信すると、OSIはその結果をMQ表にロードして待機し、ユーザーがJCLでそのBMPアプリケーションを実行すると、MQ表に蓄積されているすべてのメッセージを宛先のBMPアプリケーションに送信します。

以下は、BMPプログラムを起動するJCLの例です。

<OSIBMP.jcl>

```
//OSIBMPT JOB CLASS=A,MSGCLASS=X,MSGLEVEL=(1,1)
//TSTEP1 EXEC PGM=DFSRRRC00,
// PARM=(BMP,OIVPIL04,,OIVPBMP4,,,,,,,,,IMSA,)
//SYSOUT DD SYSOUT=*
```

以下は、上記のJCLを実行するtjesmgrユーティリティの実行例です。

```
$ tjesmgr run OSIBMP.jcl
> Command : [run OSIBMP.jcl]
Node name : NODE1
/home/oframe/OpenFrame/volume_default/SYS1.JCLLIB/OSIBMP.jcl is submitted as OSIBMPT(JOB00001).
$ tjesmgr ps
> Command : [ps]
JOBNAME JOBID CLASS STATUS RC NODE JCL
-----
OSIBMPT JOB00001 A Done R00000 NODE1 OSIBMP.jcl
```

付録 A: DL/Iステータス・コード

本付録では、DL/Iステータス・コードの内容と対応方法について説明します。

DL/I呼び出しの処理結果はPCBステータス・コード・フィールドのコード値で確認できます。アプリケーション・プログラムの作成者は、ステータス・コード・フィールドを参照し、処理結果の原因や対応方法をチェックしてください。

• AB

| | |
|-------------|--|
| 原因 | I/O領域が必要な関数呼び出しでI/O領域がパラメータとして指定されていません。 |
| 対応方法 | 関数呼び出しに適切な正しいデータ領域を指定します。 |

• AD

| | |
|-------------|--|
| 原因 | 関数呼び出しに不適切なPCBをパラメータとして指定しました。 1. システム・サービス呼び出しで、IO-PCBを使用しませんでした。 2. メッセージGU呼び出しまたはGN呼び出しで、IO-PCBではなくALT-PCBが使用されました。 3. BMP JCLでIN=trancodeを定義をせずにメッセージGU呼び出しがIO-PCBに使用されました。 |
| 対応方法 | 該当の関数呼び出しに適切なPCBであるかどうかを確認します。 |

• AZ

| | |
|-------------|---|
| 原因 | 1. PURG呼び出しをSPAG送信に使用した場合です。 2. ISRT呼び出し時に、ALT-PCBの宛先が会話型トランザクションであるにもかかわらず、最初のメッセージ・セグメントがSPAでない場合です。 |
| 対応方法 | PURG、ISRT呼び出しを正しく使用してください。 |

• A2

| | |
|-------------|---|
| 原因 | CHNG呼び出しが正しく使用されていません。 1. ALT-PCBでない場合にCHNG呼び出しが使用されました。 2. ALT-PCBまたは変更できないPCBに対して、CHNG呼び出しが使用されました。 3. メッセージ・セグメントの送信が完了していないPCBに対して、CHNG呼び出しが使用されました。 |
| 対応方法 | 1. CHNG呼び出しは、変更可能なALT-PCBを使用します。 2. ISRT呼び出しでメッセージ・セグメントを送信した後、宛先を変更する場合は、メッセージをパーージしてからCHNG呼び出しで宛先を変更します。 |

• A3

| | |
|-------------|--|
| 原因 | 宛先を設定されていない変更可能な代替PCBに対してISRTまたはPURG呼び出しが発行されました。PURG呼び出しは、1つのI/O領域のパラメータとして処理します。 |
| 対応方法 | CHNG呼び出しで指定した宛先で使用されるPCBに対してISRTまたはPURG呼び出しを発行します。 |

- A4

| | |
|-------------|---|
| 原因 | CHNG呼び出しで指定した宛先へのアクセス権限がありません。 |
| 対応方法 | CHNG呼び出しで指定した宛先が正しいかどうかを確認し、宛先への権限をチェックします。 |

- A5

| | |
|-------------|---|
| 原因 | メッセージISRT呼び出しで正しくないパラメータ・リストが使用されました。出力メッセージの最初のセグメントでないのに、4番目のパラメータ(MOD名)が指定されました。 |
| 対応方法 | メッセージの最初のセグメントでない場合は、4番目のパラメータを指定しないでください。 |

- A6

| | |
|-------------|---|
| 原因 | メッセージISRT呼び出しまたはPURG呼び出しで送信したメッセージ・セグメントのサイズが、TRANSACTマクロのSEGSIZE定義より大きい場合です。 |
| 対応方法 | 送信メッセージ・セグメントのサイズは、TRANSACTマクロのSEGSIZEを超えないように設定します。 |

- QC

| | |
|-------------|--|
| 原因 | 入力キューに該当のプログラムに関するメッセージが存在しない場合、GU呼び出しを使用しました。 |
| 対応方法 | このコードは、アプリケーション・プログラムが終了する際、ステータス情報を通知するために出力されます。 |

- QD

| | |
|-------------|--|
| 原因 | GN呼び出しを使用しましたが、そのメッセージにセグメントが存在しない場合です。 |
| 対応方法 | セグメントがないため、アプリケーション・プログラムでメッセージを適切に処理してください。 |

- QE

| | |
|-------------|--|
| 原因 | メッセージGU呼び出しを使用せずに、メッセージGN呼び出しを使用した場合です。 |
| 対応方法 | メッセージGN呼び出しを使用する前に、メッセージGU呼び出しを使用してください。 |

- QH

| | |
|-------------|---|
| 原因 | 出力論理端末の名前、またはトランザクション・コードがIMSに登録されていません。メッセージISRT呼び出しまたはPURG呼び出しで送信したメッセージ・セグメントの長さが5バイト未満です。 |
| 対応方法 | 論理端末の名前とトランザクション・コードを確認します。 |

- XA

| | |
|-------------|-----------------------------------|
| 原因 | 発信元の端末に応答した後、別のプログラムにSPAを送信した場合は。 |
| 対応方法 | アプリケーション・プログラムを修正します。 |

- XB

| | |
|-------------|--|
| 原因 | 別のプログラムにSPA送信した後、発信元の端末に出力メッセージを送信した場合は。 |
| 対応方法 | アプリケーション・プログラムを修正します。 |

- X2

| | |
|-------------|---|
| 原因 | ALT-PCBの宛先が会話型トランザクションであるにもかかわらず、最初のISRT呼び出しがSPAに送信されなかった場合は。 |
| 対応方法 | SPAをISRTした後、メッセージをISRTしてください。 |

- X3

| | |
|-------------|---|
| 原因 | SPAの先頭の6バイト・フィールドが変更されたため、SPAメッセージとして定義されません。 |
| 対応方法 | 最初に取得したSPAの先頭の6バイトは変更しないでください。 |

- X4

| | |
|-------------|-----------------------------------|
| 原因 | 会話処理が定義されていないトランザクションでSPAを送信しました。 |
| 対応方法 | SPAでないデータ・メッセージのみISRT呼び出しを行います。 |

- X5

| | |
|-------------|------------------------------------|
| 原因 | ISRT呼び出しを使用してSPAを2回以上送信しようとしていました。 |
| 対応方法 | 1つのメッセージに1つのSPAを送信します。 |

- X6

| | |
|-------------|--|
| 原因 | ISRT呼び出しを使用してSPAを送信するための無効なトランザクション・コードです。 |
| 対応方法 | IMSシステムに登録されている会話型トランザクション・コードで宛先を指定します。 |

付録 B: IO-PCBマスク

IBMメインフレームのIMS/DCが提供しているIO-PCBマスクのフィールド項目です。フィールド長とOSI製品でのサポート可否に関する情報を含みます。

| 記述子(Descriptor) | 長さ(Byte) | 対応の可否 |
|--------------------------------|--------------------------------|-------|
| Logical terminal name | 8 | 対応 |
| Reserved for IMS | 2 | 対応 |
| Status code | 2 | 対応 |
| Local date and time | 4(Date)+4(Time) | 対応 |
| Input message sequence number | 4 | 非対応 |
| Message output descriptor name | 8 | 対応 |
| Userid | 8 | 非対応 |
| Group name | 8 | 非対応 |
| Time Stamp | 4(Date)+6(Time6)+2(UTC Offset) | 非対応 |
| Userid Indicator | 1 | 非対応 |
| Reserved for IMS | 3 | 対応 |

付録 C: AIBマスク

以下は、IBMメインフレームのIMS/DCで提供されているAIBマスクのフィールド項目です。各フィールドの長さおよびOSI製品でのサポート可否に関する情報を含みます。

| 記述子(Descriptor) | 長さ(Byte) | 対応の可否 |
|----------------------------|----------|-------|
| AIB identifier | 8 | 対応 |
| DFSAIB allocated length | 2 | 非対応 |
| Subfunction code | 8 | 対応 |
| Resource name 1 | 8 | 対応 |
| Resource name 2 | 8 | 非対応 |
| Reserved | 8 | 対応 |
| Maximum output area length | 4 | 対応 |
| Output area length used | 4 | 対応 |
| Resource field | 4 | 非対応 |
| Optional area length | 4 | 非対応 |
| Reserved | 4 | 非対応 |
| Return code | 4 | 対応 |
| Reason code | 4 | 対応 |
| Error code extension | 4 | 非対応 |
| Resource address 1 | 4 | 対応 |
| Resource address 2 | 4 | 非対応 |
| Resource address 3 | 4 | 非対応 |
| User defined token | 16 | 非対応 |
| Return token | 8 | 非対応 |
| Reserved | 16 | 非対応 |

付録 D: DL/I呼び出し

アプリケーション・プログラムでシステム・サービス関数を使用するにはDL/I呼び出しが必要です。

本付録では、OSIがサポートするシステム・サービス呼び出しについて説明します。



DL/I呼び出しについては、『IMS V7 Application Programming: Transaction Manager』を参照してください。

D.1. システム・サービスDL/I呼び出し

D.1.1. CHKP(基本)呼び出し

基本CHKP呼び出しは、アプリケーション・プログラムが変更を適用するか、または異常終了した場合にリカバリを目的として使用します。

- フォーマット

```
>>-CHKP--+i/o pcb-----i/o area-----<<
```

- パラメータ

| パラメータ | 説明 |
|----------|---|
| i/o pcb | 入出力パラメータです。プログラムの最初のPCBとして渡される入出力PCBを指定します。 |
| i/o area | 入力パラメータです。8バイトのチェックポイントIDを指定します。 |

D.1.2. CHKP(シンボリック)呼び出し

シンボリックCHKP呼び出しは、アプリケーション・プログラムが変更を適用するか、または異常終了した場合にリカバリを目的として使用します。異常終了した場合は、拡張再始動(XRST)呼び出しを使用してプログラムを再起動することができます。最大7つのデータを保管することが可能であり、プログラムの再起動時に復元されます。

- フォーマット

```
>>-CHKP--+i/o pcb-----i/o area length--i/o area--+-----+<<
                                     | .----- . |
                                     | V           | |
                                     |---area length--area-+-'
```

- パラメータ

| パラメータ | 説明 |
|---------|---|
| i/o pcb | 入出力パラメータです。プログラムの最初のPCBとして渡される入出力PCBを指定します。 |

| パラメータ | 説明 |
|-------------|----------------------------------|
| i/o area | 入力パラメータです。8バイトのチェックポイントIDを指定します。 |
| area length | 入力パラメータです。エリア長を4バイトで指定します。 |
| area | 入力パラメータです。エリアの長さだけデータを保存します。 |

D.1.3. INIT呼び出し

IBMメインフレームのIMS/DCでINIT(初期設定)呼び出しを使用すると、アプリケーション・プログラムは各DB PCBのデータ可用性を検査することにより、データ可用性状態コードを受け取ることができます。

OSIでのINIT呼び出しは、アプリケーション・プログラムのINIT呼び出しの可否を確認するフラグを設定する目的のみ使用します。

- フォーマット

```
>>-INIT--+i/o pcb---+---i/o area-----<<
      '---aib-----'
```

- パラメータ

| パラメータ | 説明 |
|----------|--|
| i/o pcb | 入出力パラメータです。プログラムの最初のPCBとして渡される入出力PCBを指定します。 |
| aib | 入出力パラメータです。呼び出しに使用するAIB(アプリケーション・インターフェース・ブロック)を指定します。 |
| i/o area | 入出力パラメータです。セグメントの送信に十分なサイズの入出力域を指定します。 |

D.1.4. INQY呼び出し

INQY(照会)呼び出しは、実行環境、宛先のタイプと状態およびセッションの状態に関する情報を要求するために使用します。OSIでは、AIBSFUNCとしてENVIRONとFINDを使用できます。

- フォーマット

```
>>-INQY--+aib---+---i/o area-----<<
```

- パラメータ

| パラメータ | 説明 |
|----------|--|
| aib | 入出力パラメータです。呼び出しに使用するAIBを指定します。 |
| i/o area | 入出力パラメータです。セグメントの送信に十分なサイズの入出力域を指定します。 |

D.1.4.1. ENVIRON

ENVIRON副次機能は、アプリケーションの実行環境に関する情報を取得します。

- 出力

| タイプ | 長さ | 値 | 説明 |
|-------------------------------|----|---------|------------------------|
| IMS ID | 8 | | IMS IDを示します。 |
| リリース・レベル | 4 | | 現在サポートされていません。 |
| 制御リージョン・タイプ | 8 | DB/DC | 制御リージョンのタイプを示します。 |
| アプリケーション・リージョン ・タイプ | 8 | MPP、BMP | 実行中のアプリケーションのタイプを示します。 |
| リージョンID | 4 | | 現在サポートされていません。 |
| アプリケーション・プログラム 名 | 8 | | 実行中のアプリケーションの名前を示します。 |
| PSB名 | 8 | | 実行中のPSBの名前を示します。 |
| トランザクション名 | 8 | | 実行中のトランザクションの名前を示します。 |
| ユーザーID | 8 | | 現在サポートされていません。 |
| グループ名 | 8 | | 現在サポートされていません。 |
| 状況グループ標識 | 4 | | 現在サポートされていません。 |
| リカバリ・トークンのアドレス | 4 | | 現在サポートされていません。 |
| アプリケーション・パラメータ ・ストリングのアドレス | 4 | | 現在サポートされていません。 |
| 共用キュー標識 | 4 | | 現在サポートされていません。 |
| アドレス・スペースのユーザ ーID | 8 | | 現在サポートされていません。 |
| ユーザーID標識 | 1 | | 現在サポートされていません。 |
| リソース・リカバリ・サービ ス(RRS)標識 | 3 | | 現在サポートされていません。 |
| カタログ使用可能化標識 | 8 | | 現在サポートされていません。 |

D.1.4.2. FIND

IBMメインフレームのIMS/DCにおけるFIND副次機能は、要求されたPCB名のPCBアドレスを返します。

OSIのFIND副次機能は、アプリケーションの実行環境に関する情報を取得します。

D.1.5. LOG

IBMメインフレームのIMS/DCにおけるLOG呼び出しは、IMS システム・ログに情報を書き込むために使用されます。

OSIのLOG呼び出しは、OFM_OSI_LOG TABLEに情報を保存します。現在、OSIではAIBのみをサポートしていません。

- フォーマット

```
>>-LOG--+aib---+-----<<
```

- パラメータ

| パラメータ | 説明 |
|-------|---|
| aib | 入出力パラメータです。プログラムの最初のPCBとして渡されるIO-PCBを指定します。 |

D.1.6. ROLB

ROLB(ロールバック)呼び出しは、アプリケーション・プログラムで出力メッセージおよびデータベースの変更をキャンセルするために使用されます。

- フォーマット

```
>>-ROLB--+i/o pcb---+-----<<
```

- パラメータ

| パラメータ | 説明 |
|---------|---|
| i/o pcb | 入出力パラメータです。プログラムの最初のPCBとして渡されるIO-PCBを指定します。 |

- 制限事項

- OSCメッセージは、ROLB呼び出しを使用できません。

D.1.7. SYNC

SYNC(同期点)呼び出しは、アプリケーション・プログラムがコミット処理を行うために使用されます。BMPプログラムにのみ適用されます。

- フォーマット

```
>>-SYNC--+i/o pcb---+-----<<  
      '-aib-----'
```

- パラメータ

| パラメータ | 説明 |
|---------|--|
| i/o pcb | 入出力パラメータです。プログラムの最初のPCBとして渡されるIO-PCBを指定します。 |
| aib | 入出力パラメータです。呼び出しに使用するAIB(アプリケーション・インターフェース・ブロック)を指定します。 |

D.1.8. XRST

XRST(拡張再始動)呼び出しは、以前のアプリケーション・プログラム動作の記号チェックポイントから再始動するために使用されます。

- フォーマット

```
>>-XRST--+i/o pcb+---i/o area length--i/o area+-----+---<
                                     | .-----|
                                     | V       |
                                     '---area length--area+--'
```

- パラメータ

| パラメータ | 説明 |
|-----------------|---|
| i/o pcb | 入出力パラメータです。プログラムの最初のPCBとして渡されるIO-PCBを指定します。 |
| i/o area length | 入出力パラメータです。現在使用されていません。 |
| i/o area | 入力パラメータです。チェックポイントIDを指定します。 |
| area length | 入力パラメータです。領域の長さを4バイトで指定します。 |
| area | 入出力パラメータです。領域の長さだけのデータを復元します。 |

D.2. トランザクション管理 DL/I呼び出し

D.2.1. CHNG

CHNG(変更)呼び出しは、メッセージ・セグメントの宛先を変更するために使用します。CHNG呼び出しを使用して変更可能なALT-PCBの宛先を特定の論理端末、LU 6.2記述子、またはトランザクション・コードに指定することができます。

- フォーマット

```
>>-CHNG--+alternate pcb+---destination name-----<
```

- パラメータ

| パラメータ | 説明 |
|------------------|---|
| alternate pcb | 入出力パラメータです。CHNG呼び出しを使用する変更可能なALT-PCBを指定します。 |
| destination name | 変更するメッセージ・セグメントの宛先名を指定します。 |

D.2.2. CMD

CMD(コマンド)呼び出しは、アプリケーション・プログラムがIMSコマンドを入力するために使用します。GCMD呼び出しとともに使用してOSIコマンド・サーバーにコマンドを送り、その応答を受け取ることができます。コマンドの応答メッセージの最初のセグメントを取り出します。

- フォーマット

```
>>-CMD--+-i/o pcb+---i/o area-----<<
```

- パラメータ

| パラメータ | 説明 |
|----------|---|
| i/o pcb | 入出力パラメータです。プログラムの最初のPCBとして渡されるIO-PCBを指定します。 |
| i/o area | 入出力パラメータです。セグメントの送信に十分なサイズの入出力域を指定します。 |

D.2.3. GCMD

CMD(コマンド結果の取り出し)呼び出しは、CMD呼び出しを使用して処理したコマンドの応答メッセージの次のセグメントを取り出します。

- フォーマット

```
>>-GCMD--+-i/o pcb+---i/o area-----<<
```

- パラメータ

| パラメータ | 説明 |
|----------|---|
| i/o pcb | 入出力パラメータです。プログラムの最初のPCBとして渡されるIO-PCBを指定します。 |
| i/o area | 出力パラメータです。セグメントの送信に十分なサイズの入出力域を指定します。 |

D.2.4. GN

入力されたメッセージが1つ以上のセグメントを含む場合、GU呼び出しを使用してメッセージの最初のセグメントを取得した後、GN呼び出しを使用してセグメントを取得します。

- フォーマット

```
>>-GN---i/o pcb---i/o area-----<
```

- パラメータ

| パラメータ | 説明 |
|----------|---|
| i/o pcb | 入出力パラメータです。プログラムの最初のPCBとして渡されるIO-PCBを指定します。 |
| i/o area | 出力パラメータです。セグメントの送信に十分なサイズの入出力域を指定します。 |

D.2.5. GU

GU(後続取り出し)呼び出しは、メッセージの最初のセグメントを取得します。

- フォーマット

```
>>-GU---i/o pcb---i/o area-----<
```

- パラメータ

| パラメータ | 説明 |
|----------|---|
| i/o pcb | 入出力パラメータです。プログラムの最初のPCBとして渡されるIO-PCBを指定します。 |
| i/o area | 出力パラメータです。セグメントの送信に十分なサイズの入出力域を指定します。 |

D.2.6. ISRT

ISRT(挿入)呼び出しは、特定の宛先に1つのメッセージ・セグメントを送信するために使用します。宛先は、入出力PCB、代替PCBの指定に従います。

- フォーマット

```
>>-ISRT---i/o pcb-----i/o area---+-----+-----<
      '-alternate pcb-'          '-mod name-'
```

- パラメータ

| パラメータ | 説明 |
|---------------|---|
| i/o pcb | 入出力パラメータです。プログラムの最初のPCBとして渡されるIO-PCBを指定します。 |
| alternate pcb | 入出力パラメータです。PCBを指定します。 |
| i/o area | 出力パラメータです。セグメント送信に十分なサイズの入出力域を指定します。 |

| パラメータ | 説明 |
|----------|---|
| mod name | 出力メッセージを送信するための入力パラメータです。MOD名は8バイトで指定します。指定したMODに従って出力メッセージがフォーマットされます。 |

D.2.7. PURG

PURG(パージ)呼び出しは、ISRT呼び出しで送信したメッセージ・セグメントを出力します。アプリケーション・プログラムで1つ以上のメッセージ・セグメントを送信する場合、前のメッセージをパージするか、メッセージの送信を完了するために使用されます。

- フォーマット

```
>>-PURG--+i/o pcb-----+-----+-----+-----<
      '-alternate pcb-' '-i/o area--+-----+-'
                                '-mod name-'
```

- パラメータ

| パラメータ | 説明 |
|---------------|--|
| i/o pcb | 入出力パラメータです。プログラムの最初のPCBとして渡されるIO-PCBを指定します。 |
| alternate pcb | 入出力パラメータです。PCBを指定します。 |
| i/o area | 出力パラメータです。セグメント送信に十分なサイズの入出力域を指定します。 |
| mod name | 出力メッセージを送信するための入力パラメータです。MOD名は8バイトで指定します。指定したMODに従って出力メッセージがフォーマットされます。 PURG呼び出しは、出力メッセージの最初のセグメントのためのMOD名を指定します。 |