DIOSA/XTP V2.1

導入の手引

輸出する際の注意事項

本製品（ソフトウェア）は、外国為替及び外国貿易法で規制される規制貨物（または役務）に該当することがあります。

その場合、日本国外へ輸出する場合には日本国政府の輸出許可が必要です。

なお、輸出許可申請手続にあたり資料等が必要な場合には、お買い上げの販売店またはお近くの当社営業拠点にご相談下さい。

はしがき

本書は、DIOSA/XTP プログラム製品の導入の手引です。

本書の読者としては、業務アプリケーション開発を担当し、OS、TPBASE、TAM、Oracle、その他関連 PP の使用法を一通り心得ているシステム技術者を想定しています。

2019 年 9 月 5 版

本書の関連説明書としては次のものがあります。

- DIOSA/XTP 利用の手引き
- DIOSA/XTP メモリキャッシュ 利用の手引き
- DIOSA/XTP データストア 利用の手引き
- DIOSA/XTP データ変換・通信オプション 導入の手引き
- DIOSA/XTP データ変換・通信オプション 利用の手引き
- DIOSA/XTP API リファレンス
- DIOSA/XTP コマンドリファレンス
- DIOSA/XTP 環境定義リファレンス
- DIOSA/XTP メッセージリファレンス

目次

第1章　概要

1.1　目的と特徴

1.1.1　目的

DIOSA/XTP は、大規模アプリケーションシステム構築の汎用基盤を提供するソフトウェアです。

社会インフラの重要性が増す中、大規模アプリケーションシステムにおいても、高信頼、高性能、高運用・高稼動性、そしてアプリケーション開発の高生産性・即応性への要求が益々高まっています。

DIOSA/XTP はこれらの要件を満足すべく以下の機能を実現します。

- メモリ DB を利用して高速なデータアクセスを実現しつつ、障害時の高速な復旧による業務継続を可能とします。
- システム運行の自動化・省力化、24 時間運用システムを実現するための機能により、高運用・高稼動性を実現します。

1.1.2　特徴

DIOSA/XTP は大きく分けて以下の特徴を備えています。

- アプリケーションプログラムの開発生産性向上
- 24 時間運転システムの実現
- 拡張性の高い大規模・高信頼性・高可用性システムの構築支援

(1)　アプリケーションシステムの開発を容易にします。

- オンライントランザクションプログラムの基本処理構造を規定することにより、アプリケーションプログラムの独立性を高め、標準化された開発が行えます。
- 大規模・分散システムを達成するための所在管理や電文のルーティング、電文送受信の制御作業から解放されます。
- アプリケーション開発のためのトレース、性能解析といったプログラム開発の下流工程を支援する機能群により、アプリケーション開発および本番時の運用作業が軽減されます。
- メモリ DB を複数のアプリケーションから同時利用可能にし、高速大量処理を可能とします。
- データを分散してメモリ DB に配置し、アクセスのための所在管理とルーティング制御を行い、全データへの透過的なアクセスを可能とします。

(2)　24 時間運転システムの稼働を支援します。

- オンライン業務の稼動中に、動作中プログラムの置換を瞬時に行うことが可能であり、業務の追加・変更、プログラム障害時の緊急対応を容易に行うことができます。
- ノードの所在を意識することなく、任意のノードに対し運用指示や状態照会のコマンドを投入することができます。
- オンライン業務の稼動中に、ノード追加などシステム構成の変更や動作環境の変更を行うことが可能であり、柔軟なシステムの保守作業を実現します。

(3)　　**拡張性の高い大規模・高信頼性・高可用性システムを容易に構築することが可能です。**

- OLTP レベルあるいはノードレベルでの分散形態を多様に構成することにより、アプリケーションプログラムへの影響無く、大規模かつ高信頼なシステムを構築することができます。
- メモリ DB の稼動状態を管理し、障害時はマスタ/スレーブ切り替えを自動的に行うことを可能とします。
- Oracle Real Application Clusters を利用したクラスタ構成に対応しており、アプリケーションプログラムは切り替えを意識せずにアクセスするサーバを変更することが可能です。

1.2　構成

1.2.1　位置づけ

DIOSA/XTP は、UNIX オペレーティングシステム、および OLTP や DB などミドルウェアとアプリケーションプログラムの間に立ち、分散オンライントランザクション処理システムのミッションクリティカル性を向上させるアプリケーション実行環境として位置付けられます。

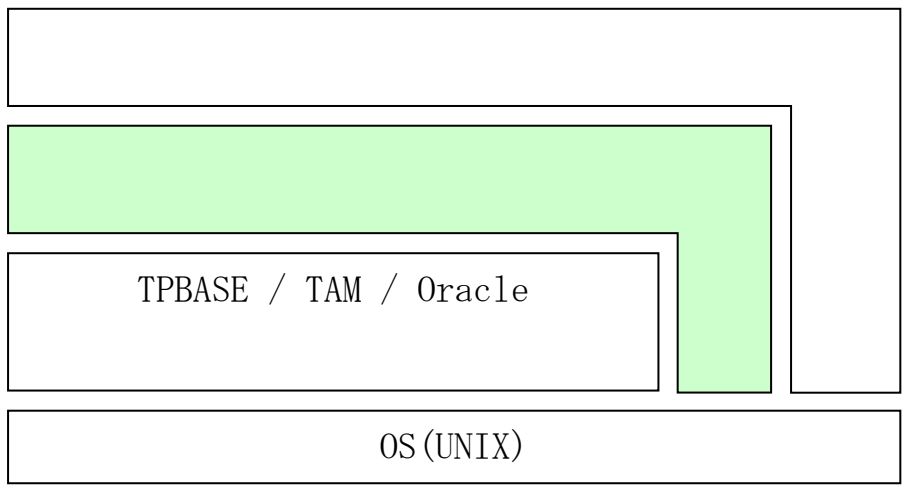
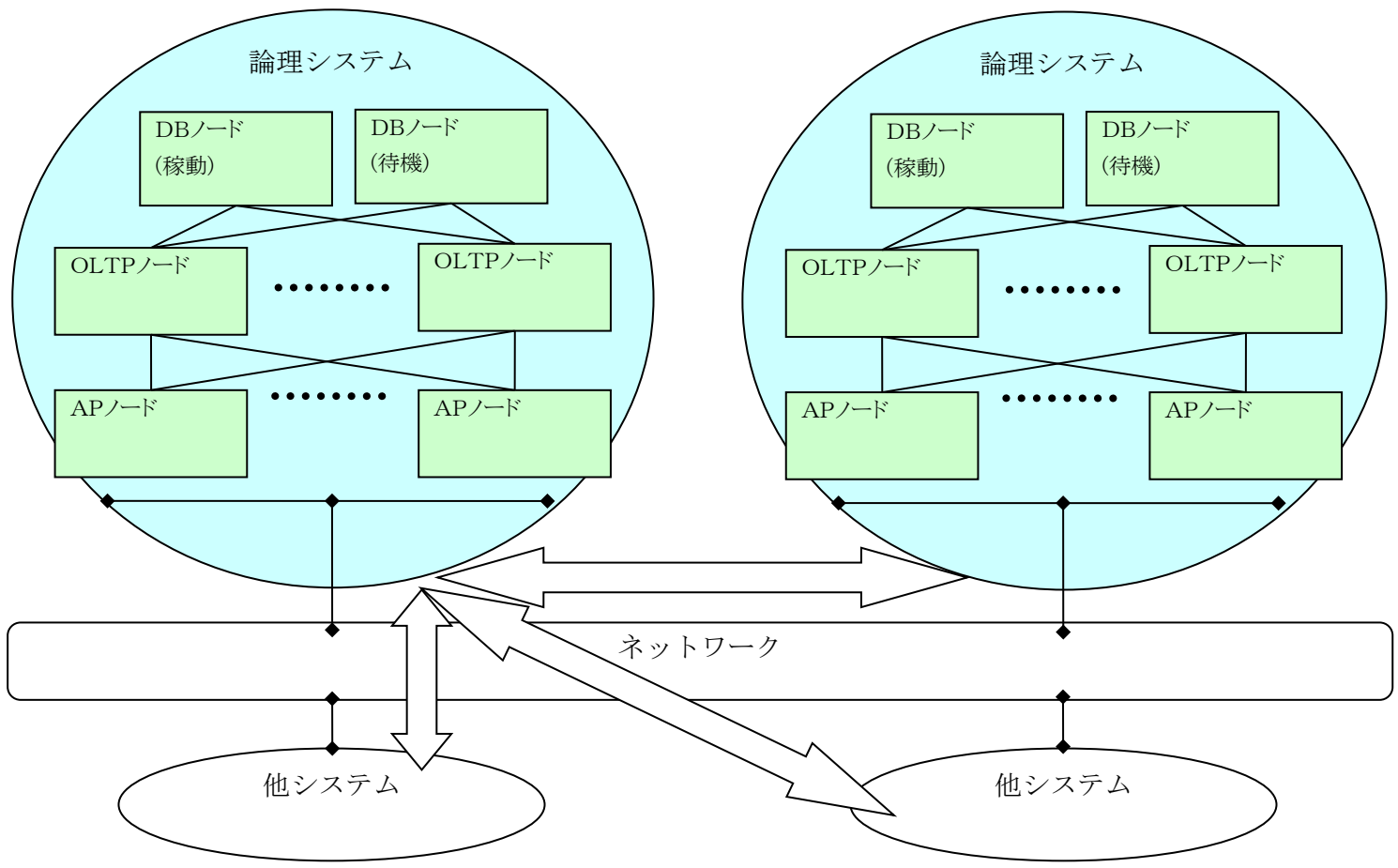


図 1-1 DIOSA/XTP の位置付け

1.2.2　システム構成

DIOSA/XTP の適用可能なシステム構成の例を以下に示します。



1.2.3　機能構成

DIOSA/XTP は大きく以下の有償プログラムプロダクト(PP)群に分けて構成されています。

ライセンス	機能	関連 PP
アプリケーション実行制御	CO 制御機能 バッチアプリケーション制御機能 タイマ制御機能 メモリ管理機能 ロック制御機能 メッセージ出力機能 アプリケーショントレース機能 アプリケーション動的置換機能 経過時間監視機能 稼働統計機能 閉塞管理機能 コマンド配信機能	TPBASE TAM Oracle
通信制御	論理ノード間通信管理機能 論理システム間通信管理機能 データベース管理機能	TPBASE Oracle
メモリキャッシュ	インメモリサーバ インメモリサーバ所在管理機能	TAM
データストア	ディレード転送機能	TAM Oracle

1.3　諸概念

DIOSA/XTP を理解するために、予め理解しておくべき概念・用語について説明します。

なお、メモリキャッシュ、データストアに関しては、以下の利用の手引の記述を参照してください。

・ メモリキャッシュ 利用の手引

・ データストア 利用の手引

(1)　論理システム

巨大なシステムを構成する独立したシステムの単位であり、運用の単位、通信の単位です。

論理システムは、1 つ以上の論理ノード(AP ノード、OLTP ノード、DB ノード)から構成されます。

(2)　論理ノード

論理システムを構成するサーバ上で動作する特定の機能群です。

1 つの物理サーバ上で、複数の論理ノードを動作させることが可能です。

ノード種別により動作可能な機能群が変わります。

- DB ノード

  Oracle Database が動作するノードです。

  主に DB の状態を管理する機能が動作します。

- OLTP ノード

  アプリケーションが動作するノードです。

  メモリ DB を使った高速処理のアプリケーションを実現できます。

- AP ノード

  OLTP ノードと外部の通信を中継するノードです。

  このノードで外部/内部のプロトコル変換などを実現します。

(3)　論理ノード間パス

同一論理システム内の論理ノード間で確立する論理的な通信パスの事を指します。

(4)　論理システム間パス

論理システム間で確立する論理的な通信パスの事を指します。論理システム間パスには、接続方法(タイミング)により、常時接続と都度接続の 2 つがあります。

(a)　常時接続

システムの立ち上げ時などに接続し、立ち下げ時などに切断します。一度接続したパスを使って、複数の電文を送受信します。

(b)　都度接続

論理システム間の通信が必要になるたびに接続・切断します。電文の送信時に送信先に接続し、送信後、または応答を受信後に切断します。

(5)　CO(制御)/出口

CO 制御機能(サーバアプリケーションの制御用フレームワーク)、バッチアプリケーション制御機能から呼び出される業務プログラムのことを CO(Control Object)と呼びます。CO 制御機能では、CO をイベント(メッセージ)により連結して処理することが可能です。

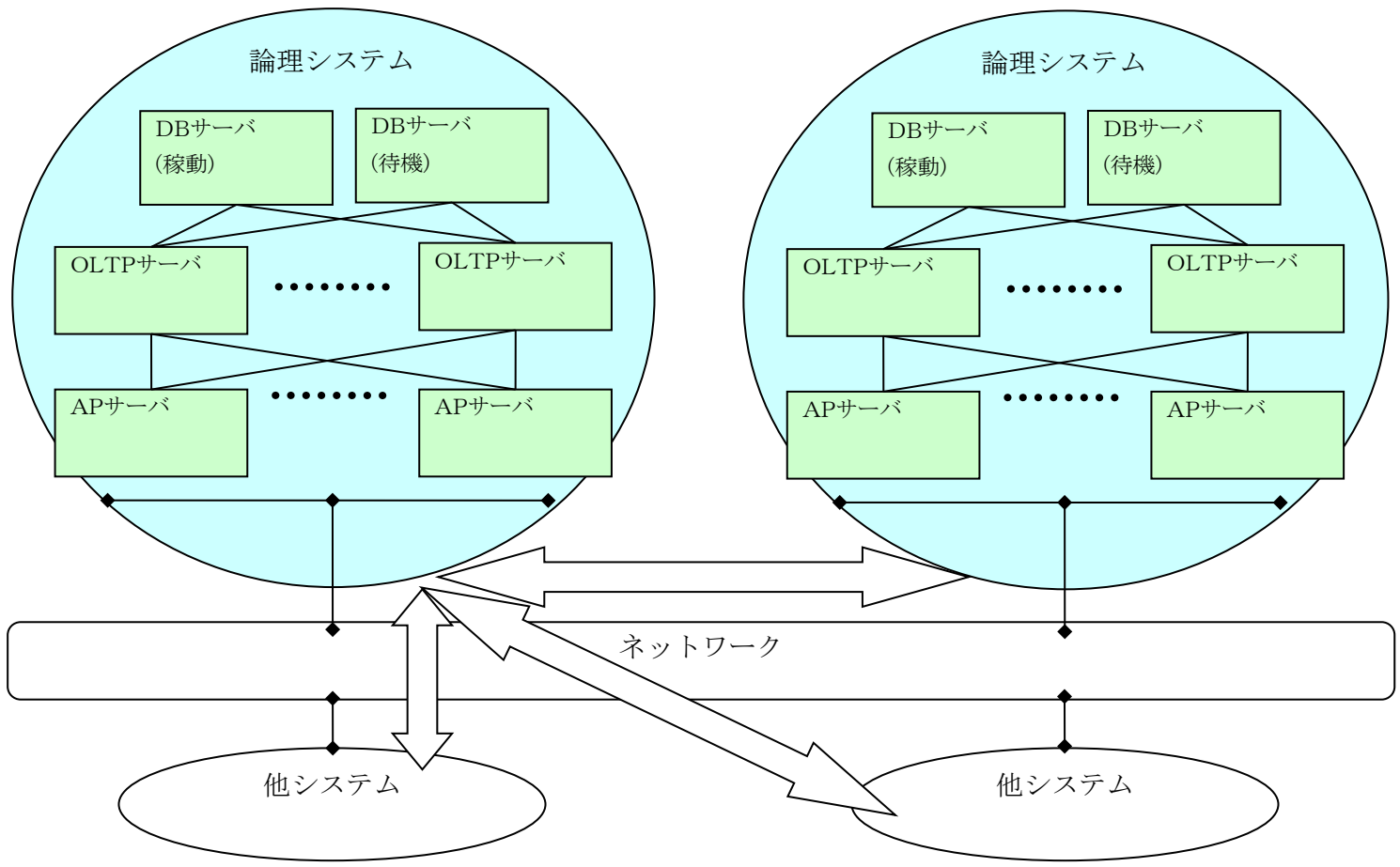
CO 制御サーバ上で動作する業務プログラムのためのプロセス初期化処理など、定型的な処理をするための処理を出口と呼びます。DIOSA/XTP の各機能では、決められたタイミングで出口(業務プログラム)を呼び出してい

ます。

第2章　システムの構築

2.1　システム概要

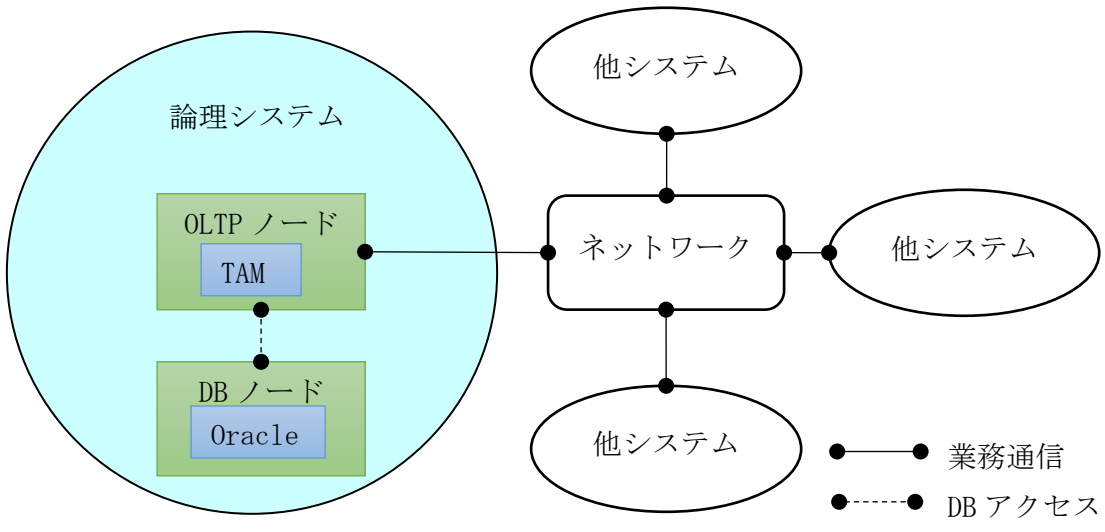
DIOSA/XTP が適用可能なシステムの構成を示します。



2.1.1　　1ノード構成例

トランザクション処理を 1 ノードで行う。

求められる要求性能を 1 ノードで満せる満たすことが可能な場合や、TAM のテーブルサイズが 1 ノードのメモリで賄える場合に選択します。

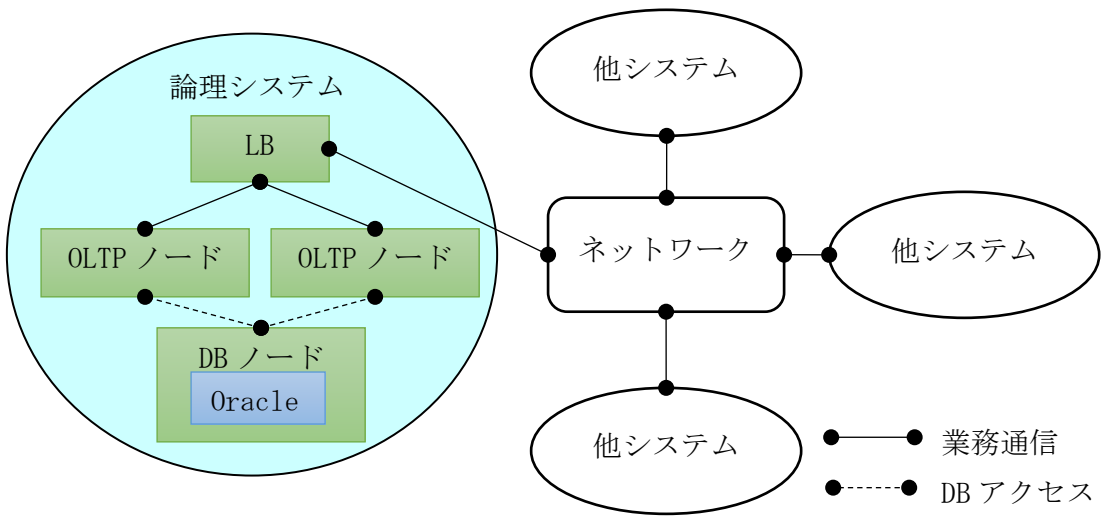


2.1.2　　2ノード構成例

要求性能を満たすために2ノードが必要となる場合や、TAMのテーブルサイズが1ノードに搭載可能なメモリを超える場合などに選択します。

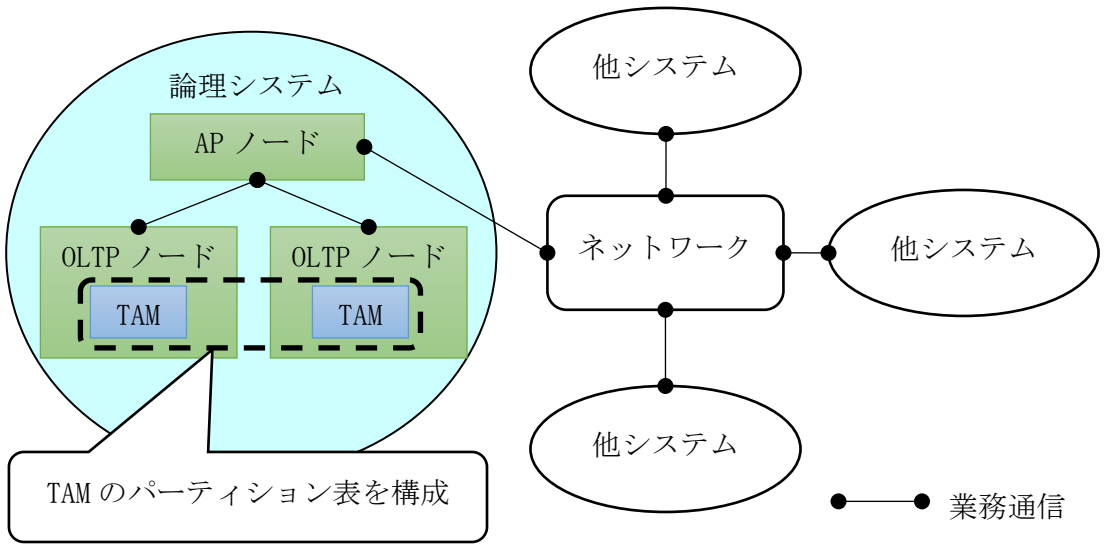
(1)　　DB(Oracle)

2つのOLTPノードによりトランザクション処理を行います。他システムとの通信はLBを介して行います。



(2)　　IM(TAM)

2つのOLTPノードによりトランザクション処理を行います。他システムとの通信はAPノードを介して行います。2つのOLTPノードに載せたTAMによりパーティション表を構成し、APノードでアクセスするパーティションを選択して、OLTPノードへの業務通信を振り分けます行います。

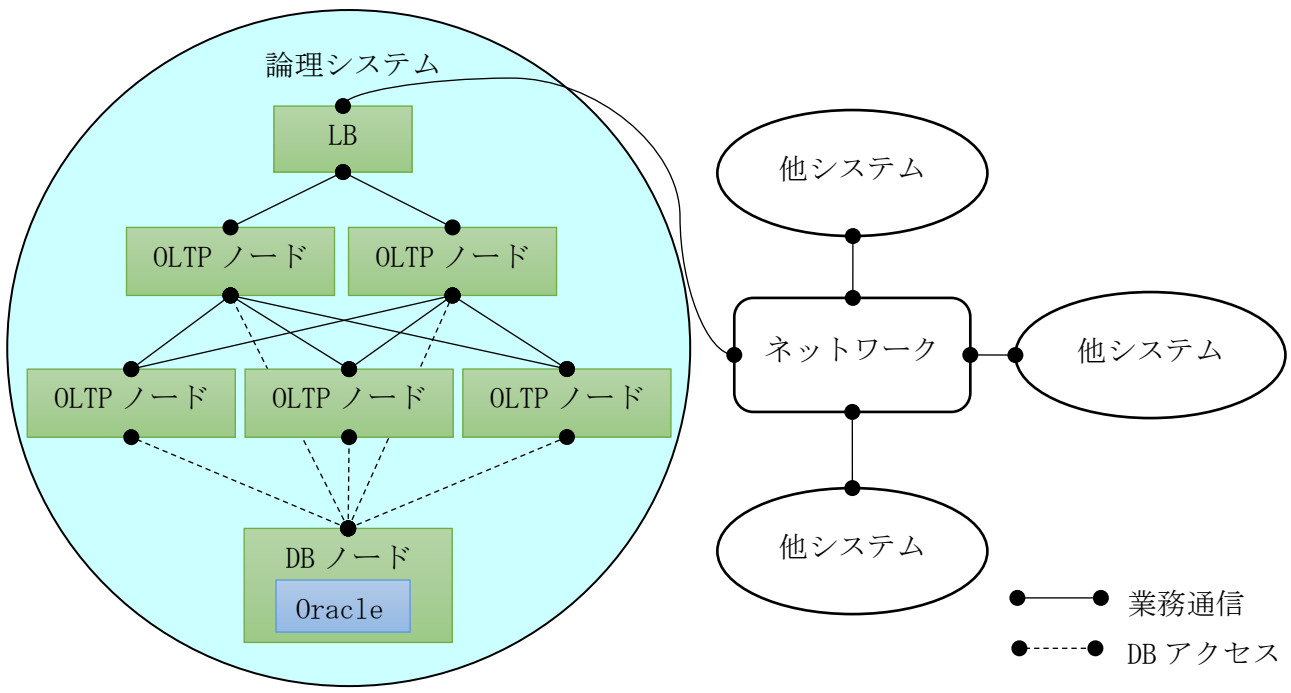


2.1.3　　高負荷システム構成例

より高い性能を満たすためや、TAMのテーブルサイズが大きい場合などに選択します。

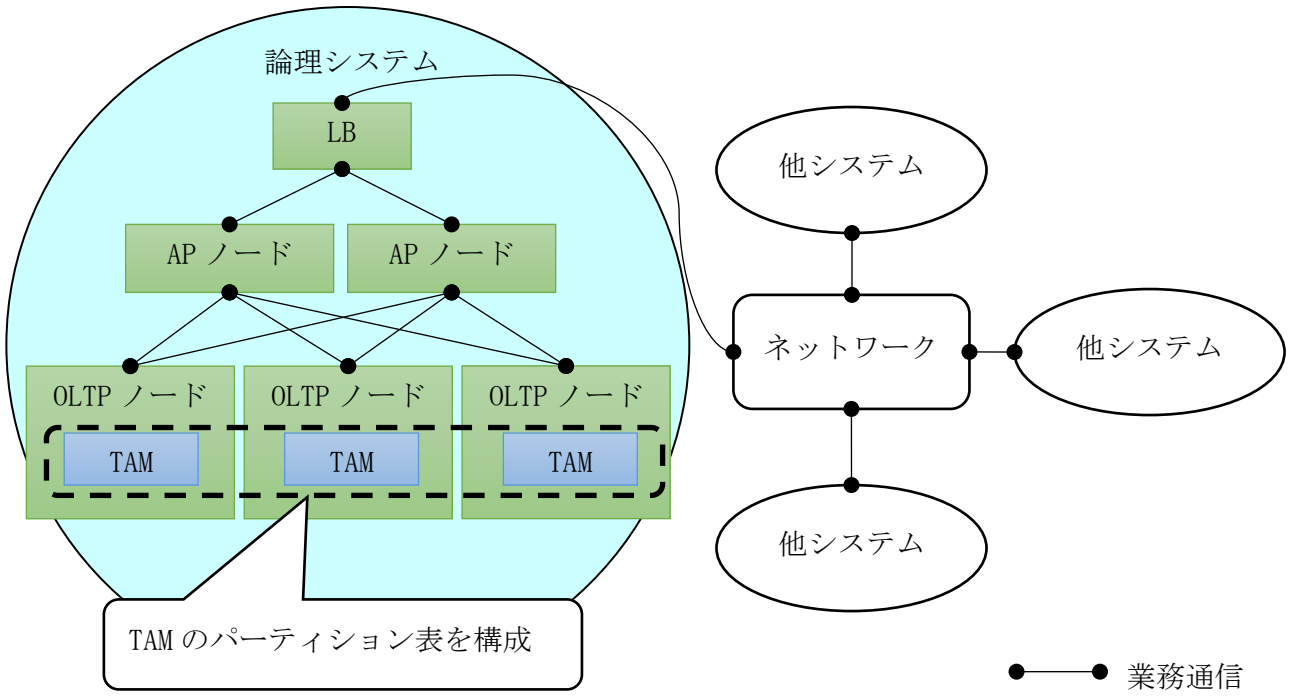
(1)　　DB(Oracle)

3 つの OLTP ノードによりトランザクション処理を行います。他システムと送受信する電文の自システム形式への変換を別ノードで処理します。



(2)　　IM(TAM)

3 つの OLTP ノードによりトランザクション処理を行います。AP ノードではアクセスするパーティションを選択して OLTP ノードへ業務通信の振り分け、他システムと送受信する電文の変換します。



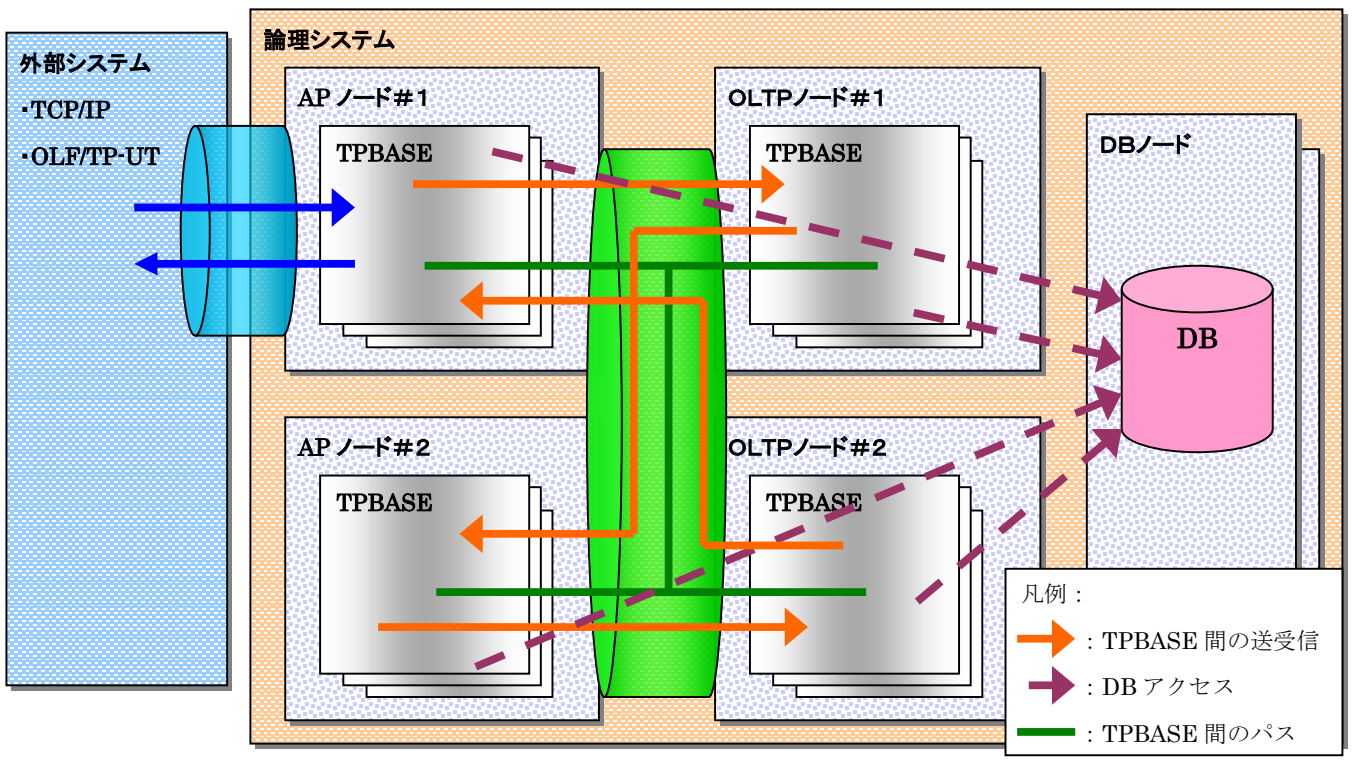
2.2　環境設計

2.2.1　ノード・ネットワーク構成

AP ノードと OLTP ノードの間では TPBASE を用いて通信します。外部システムとは AP ノードまたは OLTP ノードの TPBASE に対して TCP/IP、OLF/TP-UT を利用して通信できます。

DB ノードには TPBASE を配置していないため、他ノードとの TPBASE を利用した通信はできませんが AP,OLTP ノード上からの DB へのアクセスは DIOSA/XTP が処理を隠蔽し、自動的に行います。

TPBASE の設定方法については 2.3.7 TPBASE の環境定義を参照してください。



2.2.2　　DIOSA に関する設計

(1)　　共通設計

(a)　DIOSA/XTP インストールディレクトリ

DIOSA/XTP をインストールしたディレクトリを、環境変数(DIOSA_ROOT)に設定します。

既定値が”/opt/diosa_xtp”のため、インストール先が同ディレクトリの場合は設定不要です。

(b)　DIOSA/XTP 作業用ディレクトリ

DIOSA/XTP の各機能が共通的に利用するファイルの格納先ディレクトリを、環境変数(DIOSA_TMP)に設定します。

ディレクトリ配下には、以下の情報が格納されます。

{DIOSA_TMP}/{論理ノード名}　/lock -------排他制御用ロックファイル

/log --------常駐プロセスからの標準出力、標準エラー出力

/socket -----ソケットファイル

(2)　　メッセージ出力機能

(a)　DIOSA/XTP メッセージログファイル

DIOSA/XTP の運用メッセージや、エラー発生時のメッセージは、メッセージログファイルに出力されます。

ファイルの出力先は環境変数(DIOSA_MSG_LOG_PATH)で指定します。また、ファイルは指定されたファイルサイズごとにスワップして複数のファイルにローテーションしながら出力されますが、ファイルサイズやファイル数を環境変数(DIOSA_MSG_LOG_ROTATE)で指定します。

指定されたディレクトリ配下に、「diosa_{論理ノード名}_msg_log_{連番}」というファイルが作成されます。

(b)　ユーザ用メッセージ原型ファイル

利用者が作成した任意のメッセージを DIOSA/XTP メッセージログファイルに出力することができます。出力のためには、メッセージを記述したテキストファイルから、ユーザ用メッセージ原型ファイルを作成する必要があります。

[作成手順]

1. メッセージを記述したテキストファイルを準備する。1 行に 1 メッセージを以下の形式で記述する。
   「メッセージ ID¥t メッセージレベル¥t メッセージ種別¥t 英語メッセージ¥t 日本語メッセージ」
   ※¥t はタブ文字
2. ユーザ用メッセージ原型ファイル名を環境変数(DIOSA_MSG_FORM_FILEPATH)に指定します。
3. 原型メッセージファイル・メンテナンスコマンド(dimsgmtn)を実行します。
   > dimsgmtn -i {1 で作成したテキストファイル}

2.2.3　　TPBASE に関する設計

TPBASE の基本的な諸概念として、クラス、プロセス、トランザクションの 3 種類があります。

クラスはプロセスをまとめた属性定義であり、起動プロセス数、再起動回数、プロセスに渡すアーギュメント等を定義します。プロセスは OS で管理されるプロセスと同じで、クラス毎に管理されます。トランザクションは端末から 1 件の入力メッセージに対する一連の業務処理をいい、その識別子をトランザクション ID と呼びます。トランザクション ID はクラスに紐付けられクラス配下のプロセス群で処理されます。

TPBASE の優先度制御は、クラス内のトランザクション ID 毎に定義することができます。それ以外の優先度制御はできません。(クラス間の優先度制御などはできません)

クラスはプロセス数を管理しますので、クラスで起動できるプロセス数を分けることで、クラス間の違いを付けることが可能です。発生頻度の高い電文のトランザクションを集めたクラスを定義してプロセス数を多く定義する、また、発生頻度の少ないトランザクションを集めたクラスを定義してプロセス数を少なく定義するなどして、運用に合わせる設計をすることが可能です。

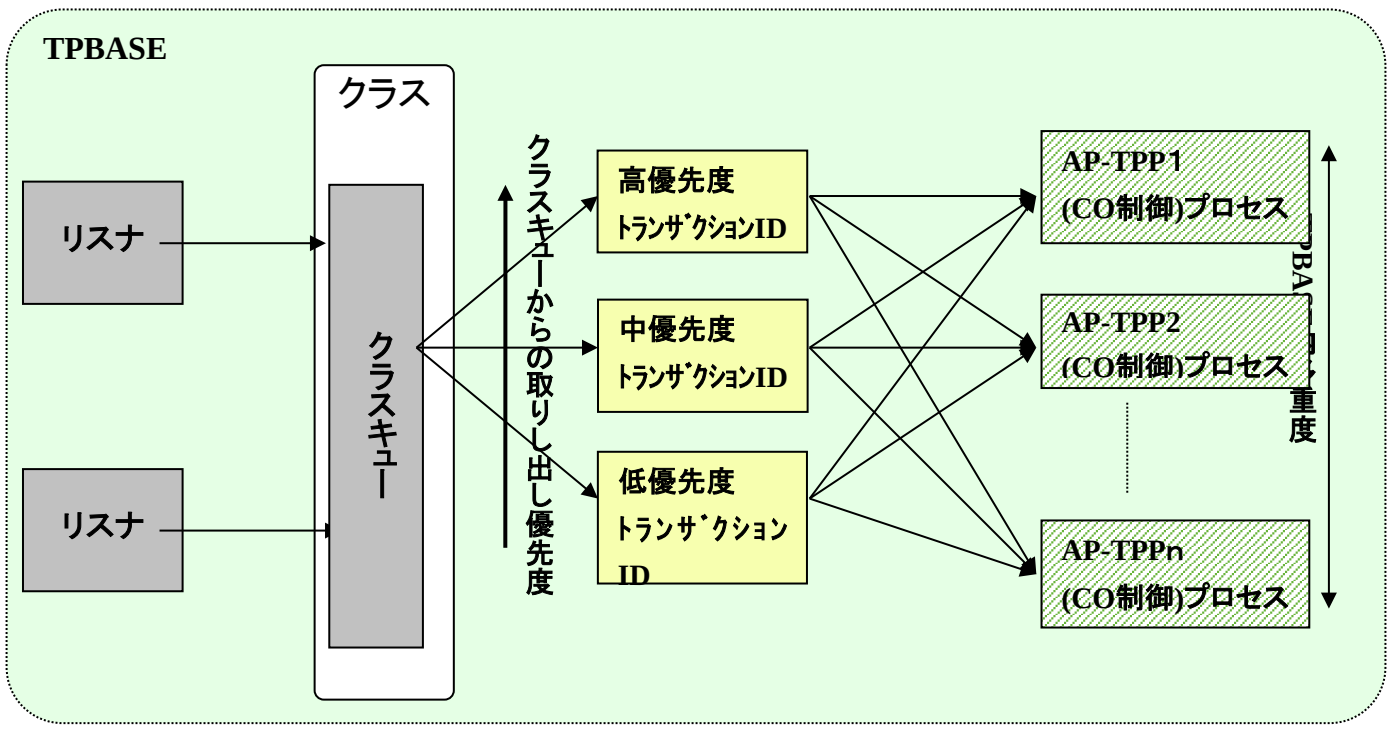
ここでは、TPBASE と DIOSA/XTP の関係から考慮すべき TPBASE 環境を述べます。

(1)　　1 クラス方式

クラスを 1 つだけ定義する。

本方式では優先度の高いトランザクションを優先的に処理します。優先度の低いトランザクションは高優先度の処理の合間で動作します。

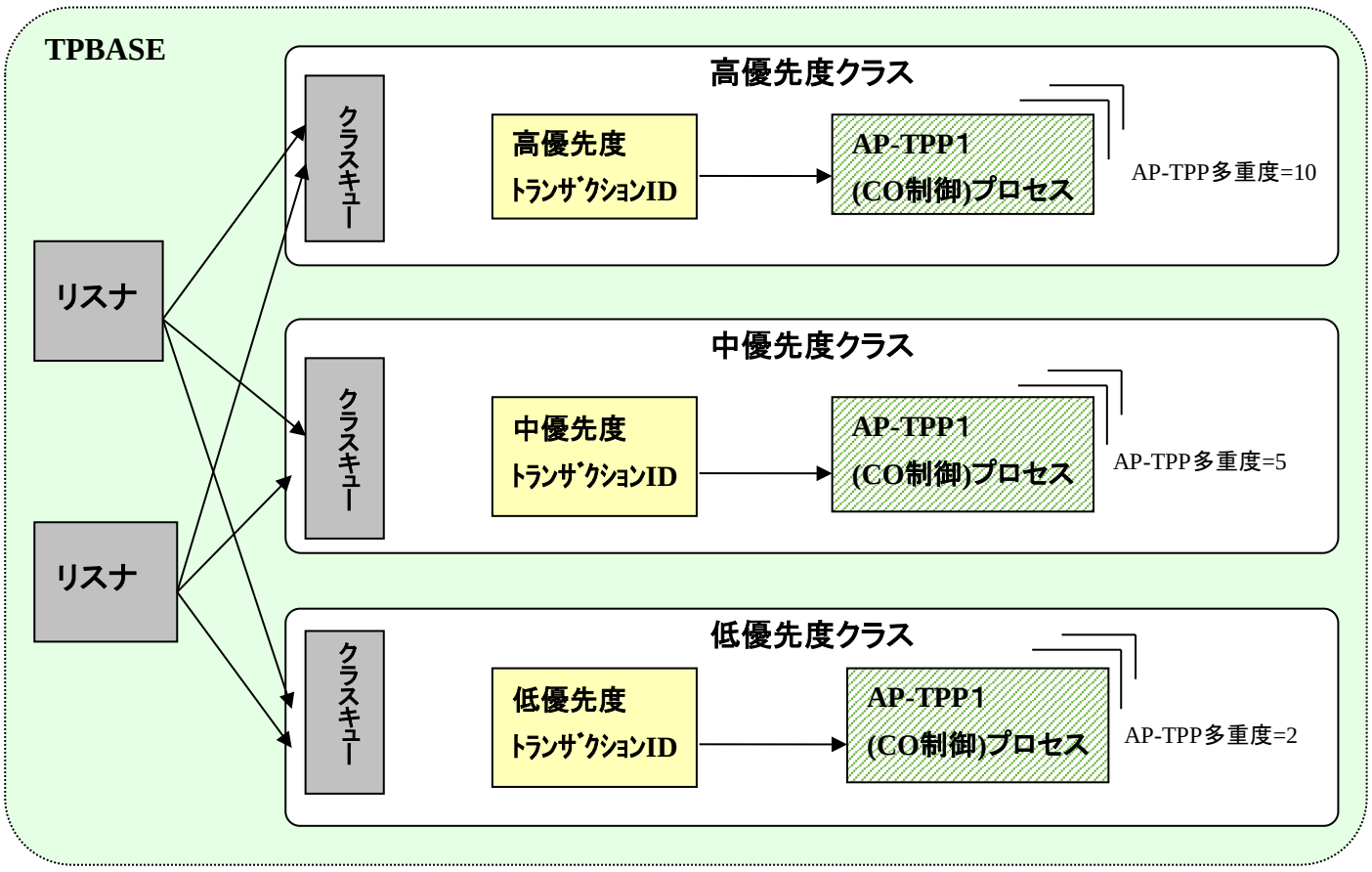
優先度の高いトランザクションが多い状況と、あまり多くない状況で優先度毎の多重度をダイナミックに切り替えます、優先度の高い、または中のトランザクションがない状況では優先度低のトランザクションが全多重度で動作できます、また、中、低優先度がプロセスを占めている割合により、高優先度トランザクションが到着しても即時動作できない場合があります。



(2)　　n クラス方式

n クラスを定義して、クラス毎に属性を持たせることができます。

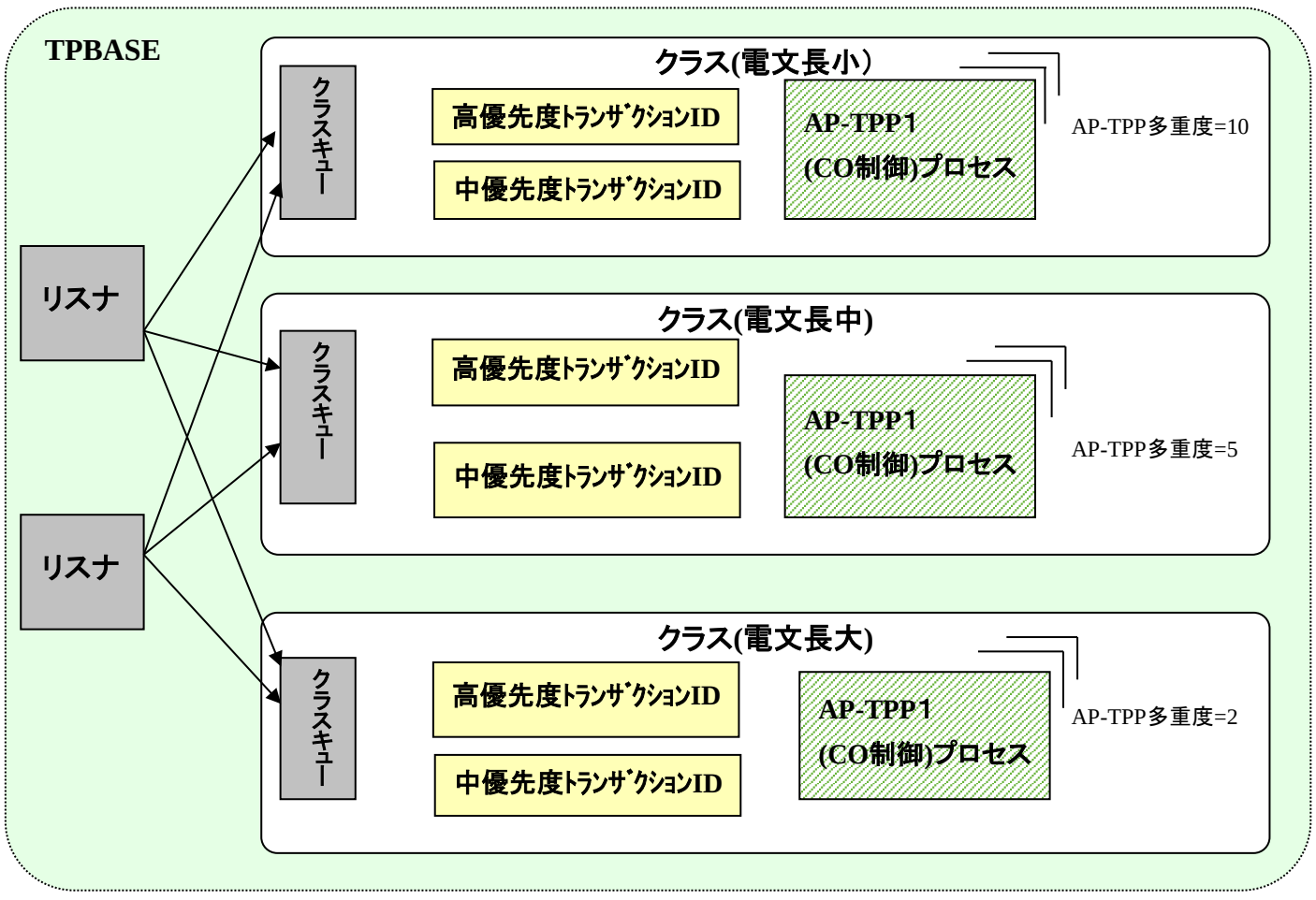
下図では便宜上高、中、低優先度クラスとしていますが TPBASE 上クラス間の優先度は制御できません。本方式ではクラスと優先度を対応させ TPP 多重度を変えることによりプロセスを制御します。　本方式では高優先度クラス以外の中、低優先度クラスに対応した業務も常に多重度分は動作が可能となります。優先度の低い業務が後回しになる危険性は少なくなります。しかし、高優先度のトランザクションが低優先度のトランザクションと CPU を分け合う等、影響される可能性があります。また、多重度数以上の業務を動作させることができないため、高優先度が少ない状況では低優先度の業務を効率良く動作させることはできません。



(3)　　n クラス、受信電文長

クラス毎に受信電文長を定義する。DIOSA/XTP ではクラス毎に受信可能な電文長を設定することができます。業務の最大電文長が 100MB のように大きい場合、全てのクラス、トランザクションで最大電文長を扱えるようにすると、メモリを圧迫する危険性があるます。DIOSA/XTP では電文長によるクラス分けができるようになっています。

最大電文長に近い業務電文の発生率が低い場合等は、クラスのプロセス多重度を絞るように定義します。また、発生率が高い電文長クラスは、プロセス多重度を上げる等を考慮することで、メモリ使用量を抑えることができます。



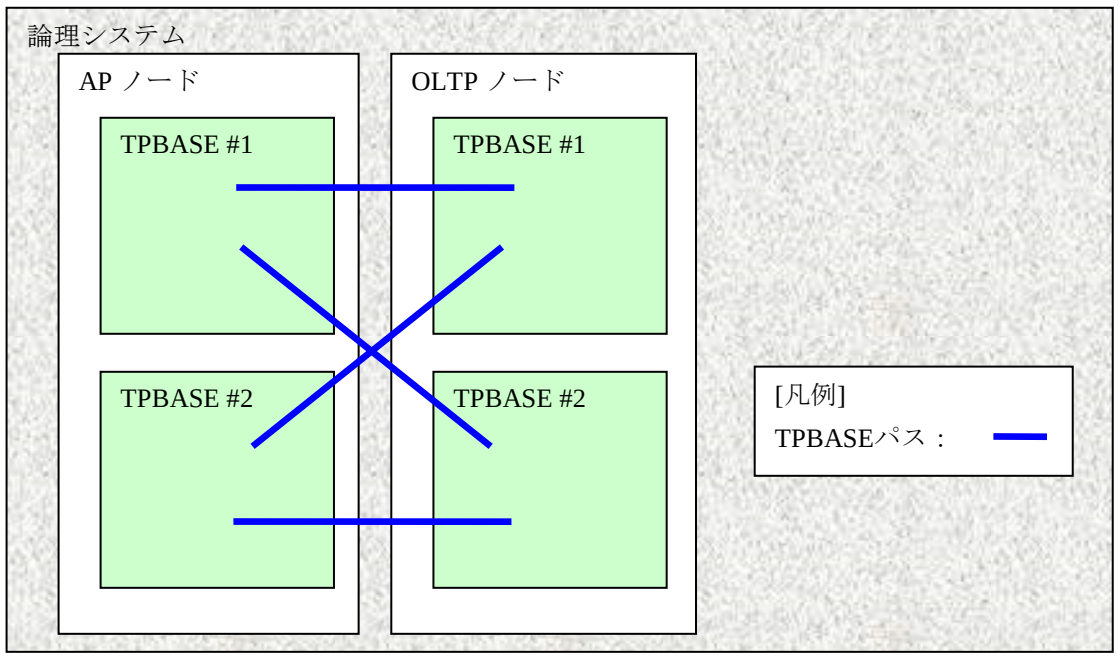
(a)　TPBASE の配置

TPBASE はクラス数、プロセス数を増やしても比例してスループットが上がるわけではありません。必要に応じて 1 ノードにマルチ TPBASE の配置を考慮してください。

(4)　　マルチ TPBASE

TPBASE の性能が求める性能に及ばない場合は 1 ノードに TPBASE を複数配置することができます。

ノード内に複数の TPBASE を配置した場合、AP ノード、OLTP ノードの各 TPBASE は全てが互いに接続するように設定してください。



(5) DIOSA/XTP が提供する物件

DIOSA/XTP を利用するために TPBASE に以下の設定を行う必要があります。

(a) 制御用 TPP

- CO 制御 TPP(CO 制御機能)
- 通信制御 TPP(パス制御機能)
- 電文再送 TPP(電文保証機能)
- 応答電文受信 TPP(電文保証機能)
- 受信情報削除 TPP(電文保証機能)

(b) 制御用ライブラリ

- 通信リスナ出口ルーチン(論理ノード間通信)(パス制御機能)
- 通信リスナ出口ルーチン(論理システム間通信)(パス制御機能)

設定方法の詳細は、2.3.8 TPBASE の環境定義を参照してください。

2.2.4　　TAMに関する設計

TAMの環境定義については、DIOSA/XTP　メモリキャッシュ利用の手引 第4章 システムの構築を参照してください。

2.2.5　　Oracle データベースに関する設計

DIOSA/XTP ではリレーショナル・データベースとして Oracle DB を使用しています。アプリケーションプログラムが OracleDB にアクセスしデータを操作するためには、SQL コンテキストを使用しデータベースへ接続することが必要です。また、OracleDB はデータを格納するデータベース・ファイルを管理するにあたり、データベース・インスタンスを生成します。

DIOSA/XTP は OracleDB とアプリケーションプログラムの間に立ち、アプリケーションプログラムにおける OracleDB に関する管理の負担を軽減します。本節では、DIOSA/XTP が管理するにあたり Oracle に関する DIOSA/XTP 固有の概念について説明し、合わせてユーザが設計する内容を述べます。

(1)　　接続方式設計

OracleDB に接続するにあたって、DIOSA/XTP では 2 つの方式を選ぶことが出来ます。1 つはデータベース接続出口によるデータベース接続、もう 1 つは環境定義によるデータベース接続です。「表 2-1 データベース接続方式」に長所と短所を記載しますので、適切な方法を選んでください。

実際の環境定義方法は「第 I 編 2.3.3 DB 関連(DBCTRL)」を参照してください。

表 2-1 データベース接続方式

方式	詳細	長所	短所
データベース接続出口	データベース接続を行うための出口ルーチンを用意し、ルーチン内で接続を行う方式	ユーザ ID、パスワードを環境定義に記載することなく、接続を行うことが出来ます	利用者が接続のためのルーチンを用意する必要があります
環境定義	環境定義にユーザ ID とパスワードを設定し、DIOSA/XTP が接続を行う方式	利用者が接続のためのルーチンを用意する必要がありません	ユーザ ID、パスワードを環境定義に平文で記載するため、環境定義を見られる人であれば誰でも参照することができます

(2)　　リソースグループ設計

DIOSA/XTP はデータベース・インスタンスが 2 台の Oracle RAC に対応しています。Oracle RAC 機能は、複数の Oracle DB サーバを経由して同一の DB の整合性を保ちながらアクセスする機能ですが、無秩序に Oracle DB サーバを選択して DB にアクセスした場合、各サーバ上のキャッシュを同期させる処理(キャッシュフュージョン)が多発し、全体的なスループットを低下させてしまいます。利用者は「図 2-1 データベース・インスタンスとリソースグループの関連性」のように Oracle DB のデータベース・インスタンスとリソースグループの関連性を設計することで、DIOSA/XTP がリソースグループごとに通常利用する DB ノードを 1 台に限定させることにより、キャッシュフュージョンの発生を抑えることが可能です。

また DIOSA/XTP では、一方のデータベース・インスタンスに障害が発生した場合、もう一方のデータベース・インスタンスへと自動的に接続先を切り替えることで、高可用性を実現しています。

なお、Oracle RAC を導入しないシステムで運用を行いたい場合は、データベース・インスタンス 1 台に全てのリソースグループを関連付けることで対応することが可能です。

実際の環境定義方法は「2.3.3 DB 関連(DBCTRL)」を参照してください。

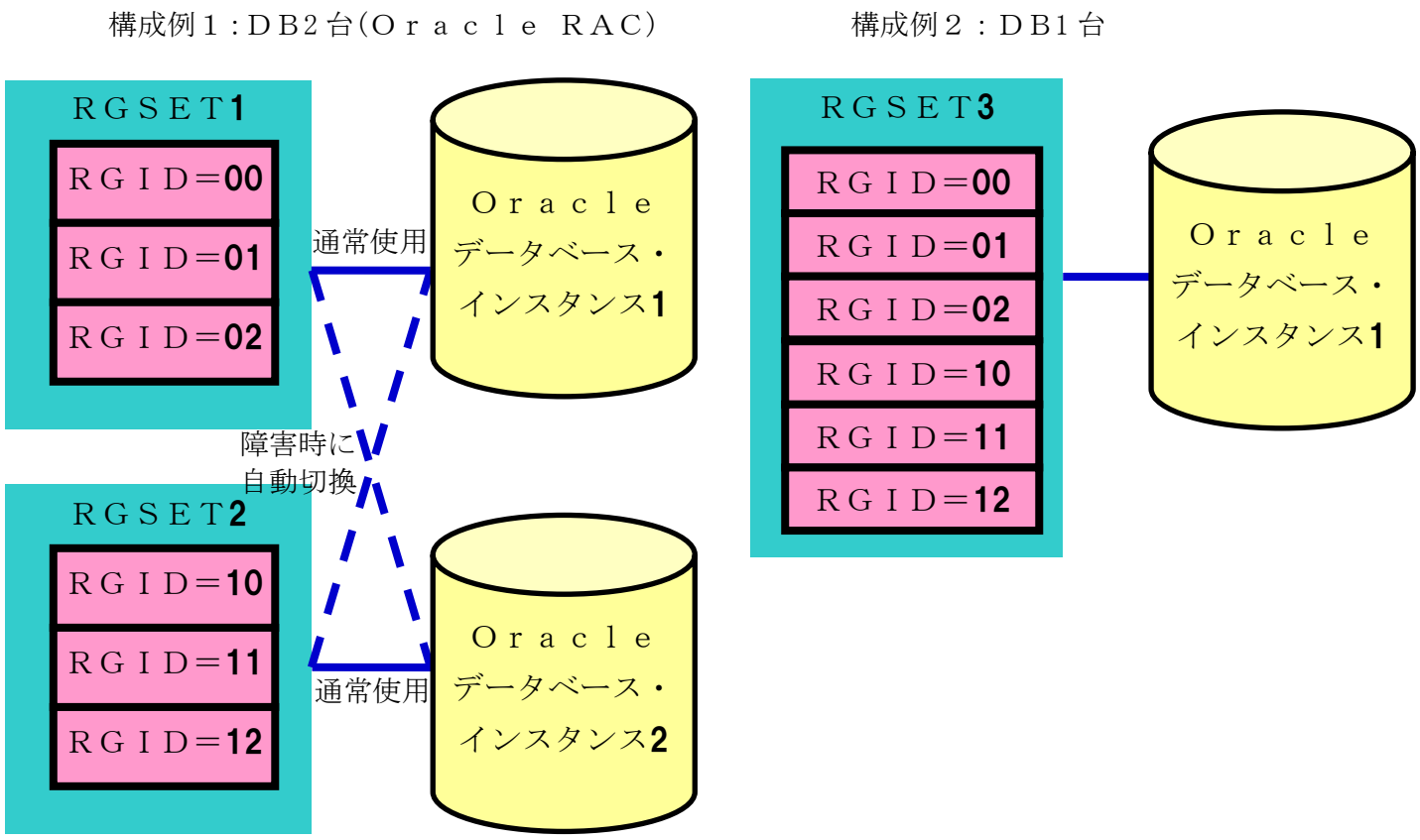


図 2-1 データベース・インスタンスとリソースグループの関連性

(3)　　インスタンスグループ設計

DIOSA/XTP では、1 台または 2 台のデータベース・インスタンスをセットとして取り扱い、これをインスタンスグループと呼んでいます。インスタンスグループは複数セット用意できるため、「図 2-2 データベース・インスタンスとインスタンスグループの関連性」のように、同一システム内に稼動系データベース・インスタンスを併設することが可能です。データベース・インスタンスに障害が発生した場合の接続管理は、インスタンスグループ毎に行われます。

利用者は Oracle DB に保存するデータ領域の設計を行った上で、データベース・インスタンスとインスタンスグループの設計を行ってください。

実際の環境定義方法は「2.3.3 DB 関連(DBCTRL)」を参照してください。

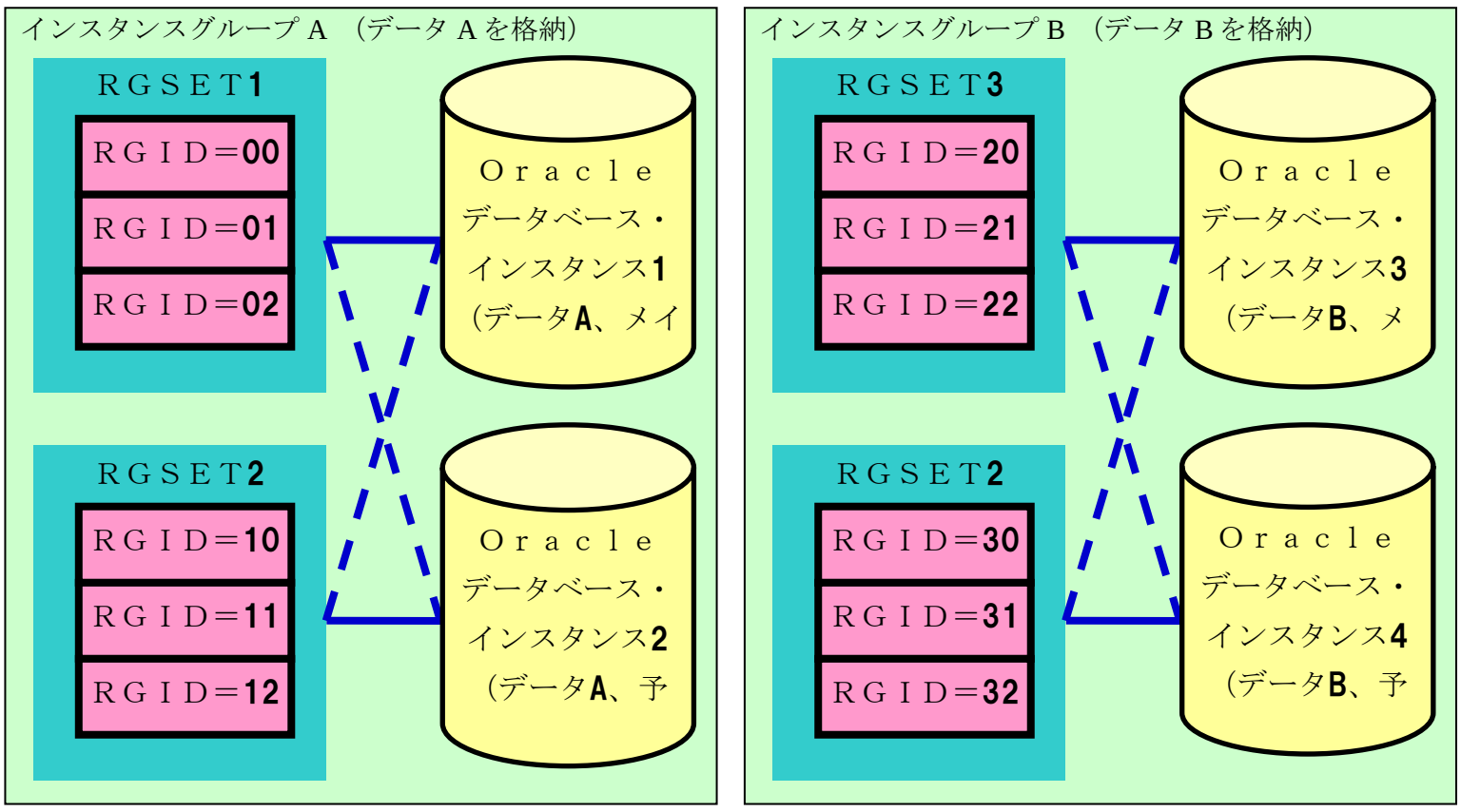


図 2-2 データベース・インスタンスとインスタンスグループの関連性

(4)　　ネット・サービス名とインスタンス識別子(SID)の設計について

DIOSA/XTP では OracleDB に接続するためのネット・サービス名を環境定義(DBCTRL 節の DBNAME パラメータ)に指定しますが、ネット・サービス名は該当するインスタンスのインスタンス識別子(SID)と同一になるように設計してください。

これは、指定されたネット・サービス名が DB ノードで SMON プロセスの監視処理でプロセス名の特定のために使用され、インスタンス識別子(SID)としての意味も持つためです。

2.2.6　　電文保証

電文保証では以下の 4 項目の設計を行います。

- 電文保証機能自体の動作設計
- 保証電文送信有無判定出口
- リトライオーバ CO
- DB(Oracle,TAM)

(1)　　電文保証機能自体の動作設計

電文保証では登録された電文の再送を行う管理プロセスがいます。このプロセスが定期的DBをチェックして、再送が必要な電文を宛先に送信します。DB をチェックする間隔や受信側のチェック履歴の保持期間などを設計します。設計する対象項目については、「環境定義リファレンス」の「第Ⅱ編 環境定義」「2.2 APMGRNT 電文保証機能)」を参照してください。

電文保証の管理プロセスはプロセス多重度 1 以上で起動できます。多重度を上げることで、再送処理の効率が上がりますが、DB へのアクセス率も上がります。システム要件やシステムリソースに合わせて多重度を決定指定ください。多重度の設定は TPBASE のプロセス定義(ped ファイル)で行います。「2.3.8 TPBAE の環境定義」「(3)DISOA 制御プロセスのプロセス環境定義(*.ped)」を参照してください。

(2)　　保証電文送信有無判定出口

電文保証の DB に登録されたメッセージを再送する際に、宛先システムの状況などにより、送信を保留する、電文を破棄することができます。これらの判断は保証電文送信有無判定出口で行います。

この機能を利用するためには、DIOSA/XTP 環境定義「APMGRNT」の「JUDGEEXIT」に出口名を設定します。出口の仕様はAPIリファレンスを参照してください。出口を利用しない場合は、再送回数の上限まで再送を行います。

(3)　　リトライオーバ CO

管理プロセスによる再送が上限回数まで行われても、相手からの応答がない場合に、再送した電文について後処理を行うための CO を呼び出すことができます。

この機能を利用するためには、DIOSA/XTP 環境定義「APMGRNT」の「OVERCONAME」に出口名、「OVERTXID」TXID を設定します。出口の仕様は通常の CO と同じです。出口を利用しない場合は管理プロセスにより電文が削除されます。

(4)　　DB(Oracle,TAM)

到達保証や順序性保証を行う場合、電文保証では送信する電文の情報やその制御情報を Oracle 表または TAM に保持します。電文保証が使用する表は送信管理表、送信電文保存表、順序性保証グループ管理表、受信管理表の 4 つです。

送信管理表のサイズについては、「付録 C データベース一覧」を参照してください。

TAM 上の表はシステムで更新を行う全ての MAPID に作成してください。

2.3　環境定義

2.3.1　環境変数

DIOSA が動作するために必要となる環境変数を設定します。

(1)　システム環境変数

DIOSA/XTP のコマンドやライブラリ、また内部実行するシステムコマンドへのパスを設定します。

PATH	/opt/diosa_xtp/bin:/usr/bin
LD_LIBRARY_PATH	/opt/diosa_xtp/lib:/usr/lib64

2.3.2　システム構成関連(DIOSAMAP,SYSMAP,OTCENV)

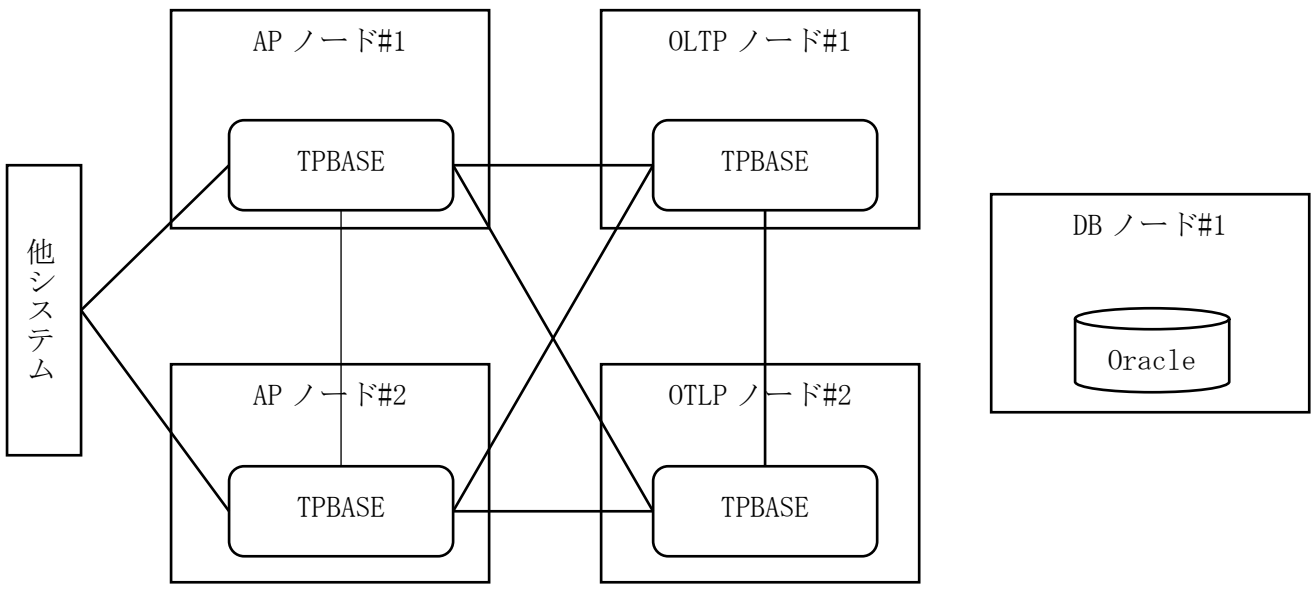
DIOSAMAP 節、SYSMAP 節および OTCENV 節では、DIOSA の各機能が認識可能な論理システム内の論理ノードの構成や論理システムの構成を定義します。

(1)　DIOSAMAP 節

DIOSAMAP 節では DIOSA の各機能が認識可能な論理システム内の論理ノードの構成を定義します。

以下の例では、論理システムは AP ノード 2 台、OLTP ノード 2 台、DB ノード 1 台、DB ノード 1 台から構成されています。AP ノード、OLTP ノードにはそれぞれ 1 つの TPBASE が置かれています。AP ノードと OLTP ノードは TPBASE により接続され業務通信が行われます。DB ノードとは業務通信は行われません。

論理ノードと TPBASE の配置関係を DIOSAMAP に記述します。



さらに、DIOSA の各機能がノード間の通信で使用する IP アドレスとポート番号を記述します。

IP アドレス/ポート番号と機能の対応は以下の通りです。

IP アドレスと機能

	メモリキャッシュ	ディレード転送	閉塞	コマンド配信
IPADDR	○			
IPADDR2	○			
IPADDR_CTL	○	○		
IPADDR_CTL2	○			
IPADDR_OP			○	○
IPADDR_EXT		○		○

ポートの指定

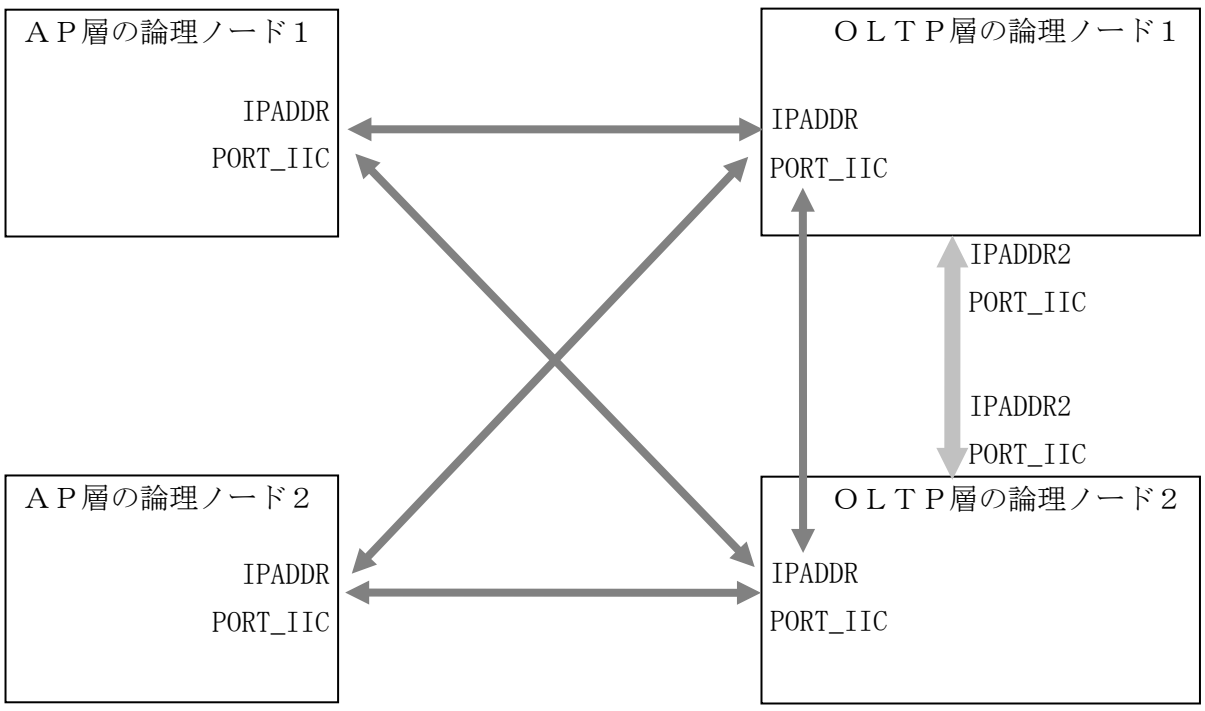
	メモリキャッシュ	ディレード転送	閉塞	コマンド配信
PORT_BCM			○	
PORT_CDD				○
PORT_DLT		○		
PORT_IIC	○			
PORT_IMS	○			

各機能が参照する IP アドレスとポートを以降で説明します。

(a)　IPADDR、IPADDR2

メモリキャッシュ機能が使用し、論理システム内の各論理ノードとの接続を行います。

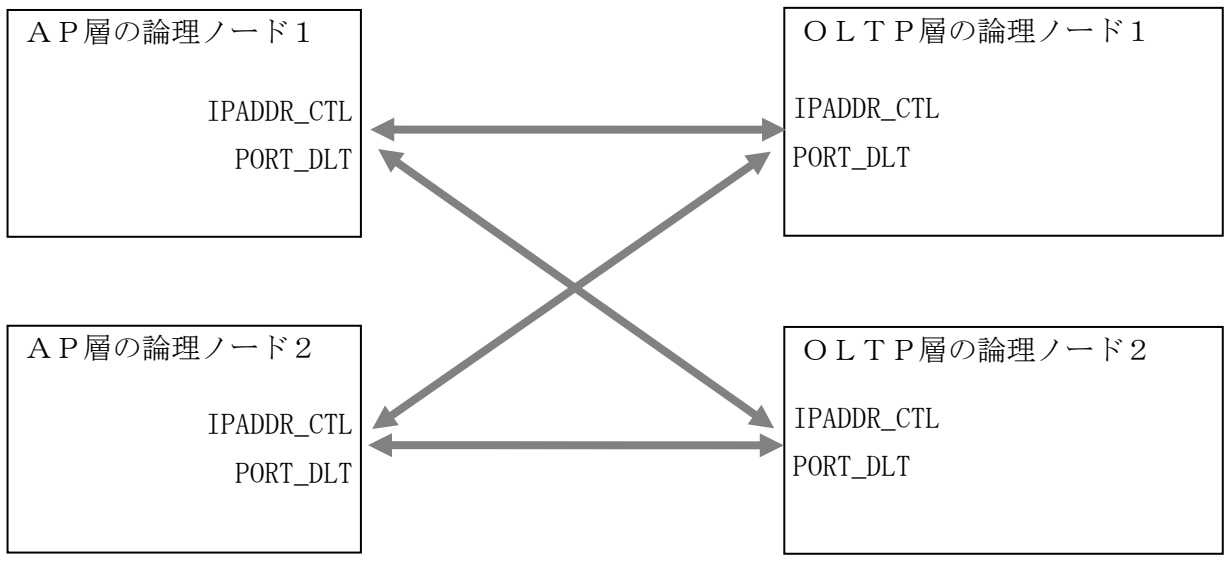
メモリキャッシュ機能の IPADDR/IPADDR2 を使用した接続を以下に示します。



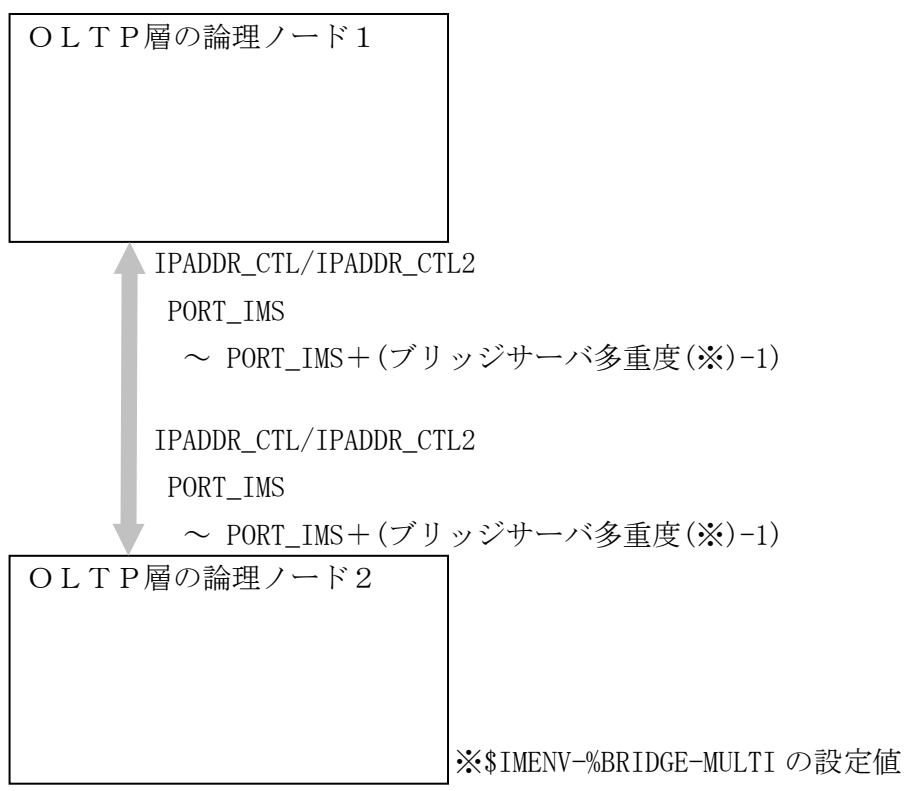
(b)　IPADDR__CTL、IPADDR__CTL2

ディレード転送機能、メモリキャッシュ機能が使用し、論理システム内の各論理ノードとの接続を行います。

ディレード転送機能の IPADDR__CTL を使用した接続を以下に示します。



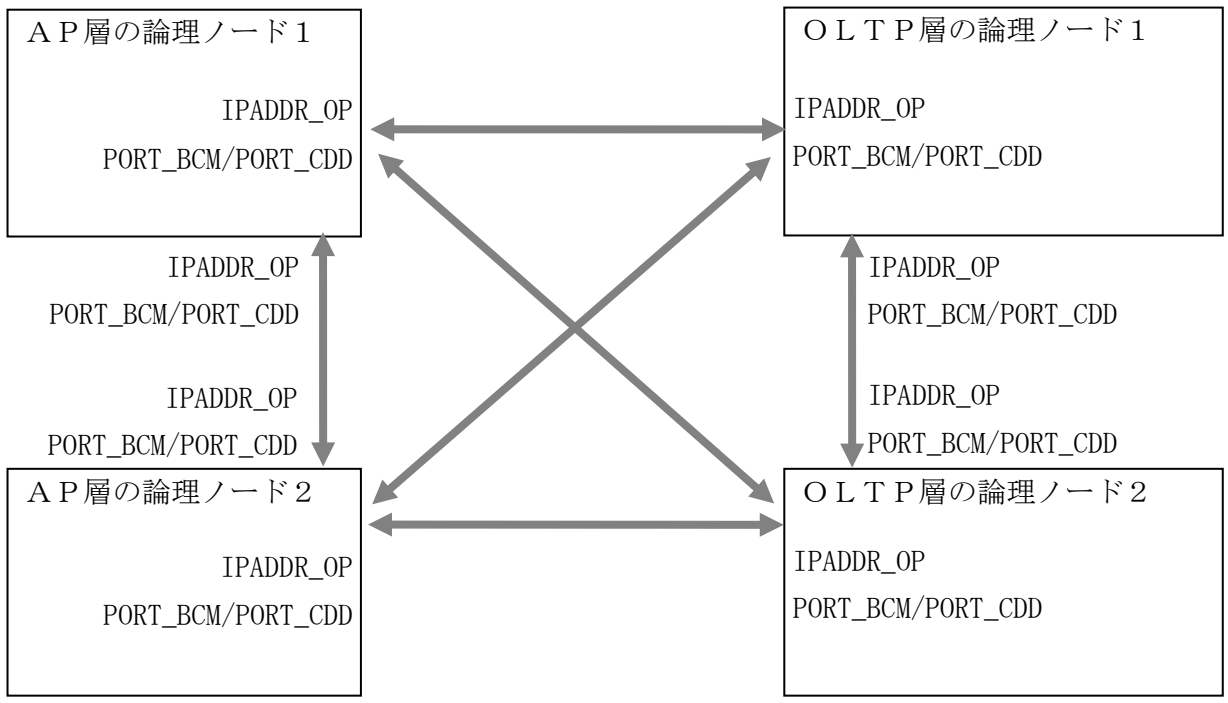
メモリキャッシュ機能の IPADDR__CTL、IPADDR__CTL2 を使用した接続を以下に示します。



(c)　IPADDR__OP

閉塞管理機能及びコマンド配信機能が使用し、論理システム内の各論理ノードとの接続を行います。

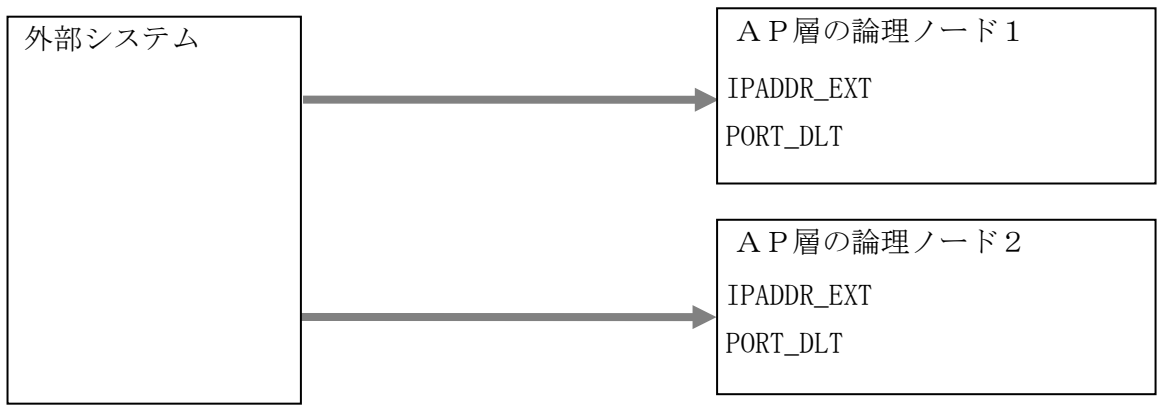
閉塞管理機能及びコマンド配信機能の IPADDR__OP を使用した接続を以下に示します。



(d)　IPADDR__EXT

ディレード転送機能及びコマンド配信機能が使用し、外部システムとの接続を行います。本 IP アドレスは、外部システムからの要求を受信するために使用します。

使用例として、ディレード転送機能の IPADDR__EXT を使用した接続を以下に示します。



(2)　SYSMAP 節

SYSMAP 節には、論理システムの構成を定義します。

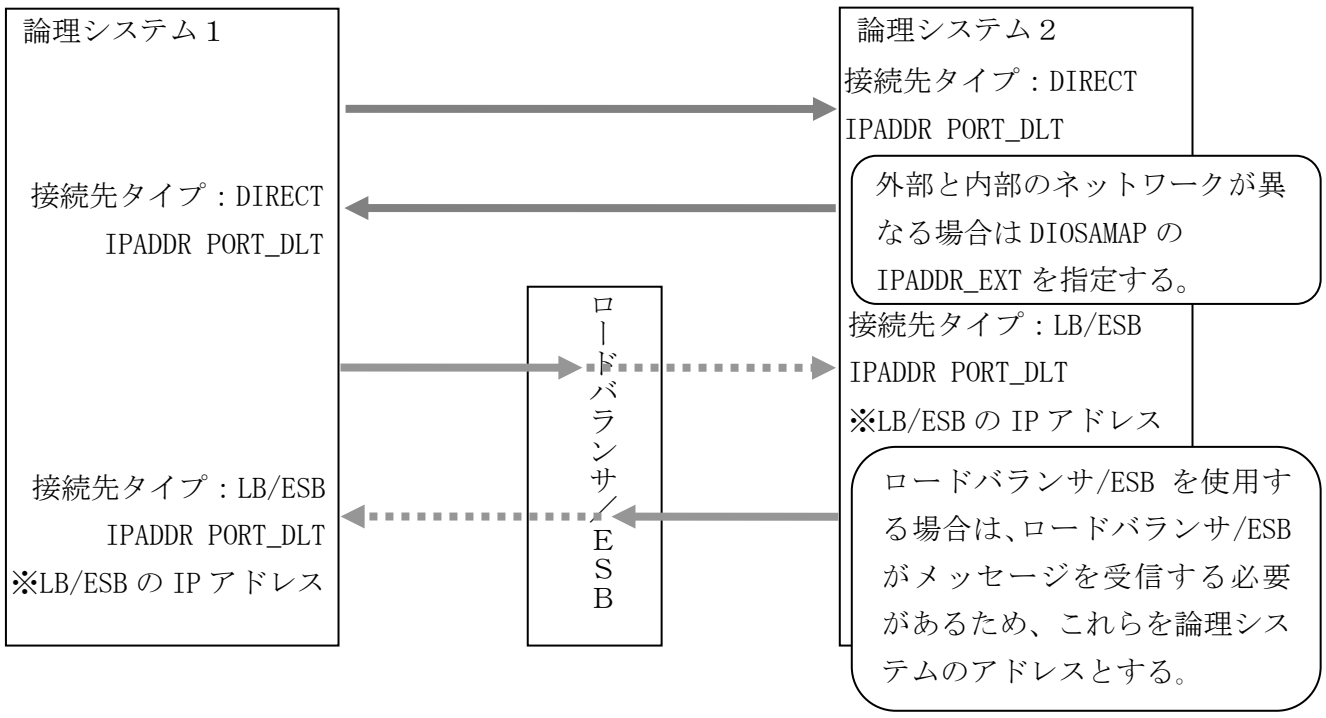
SYSMAP 節に定義された自論理システムの IP アドレスは、コマンド配信機能及びディレード転送機能が外部システム(相手論理システム)からの要求を受信するために使用します。

また、外部システム(相手論理システム)の IP アドレスについては、コマンド配信機能及びディレード転送機能が外部システム(相手論理システム)に要求を送信するために使用します。

なお、SYSMAP の IPADDR に DIOSAMAP の IPADDR と異なるアドレスを指定する場合(論理システム内と論理システム外のネットワークが異なる場合)は、DIOSAMAP の IPADDR__EXT に SYSMAP の IPADDR を指定する必要があります。

例として、ディレード転送機能が定義された IP アドレスを使用して接続する際の動作イメージを以下に示し

ます。

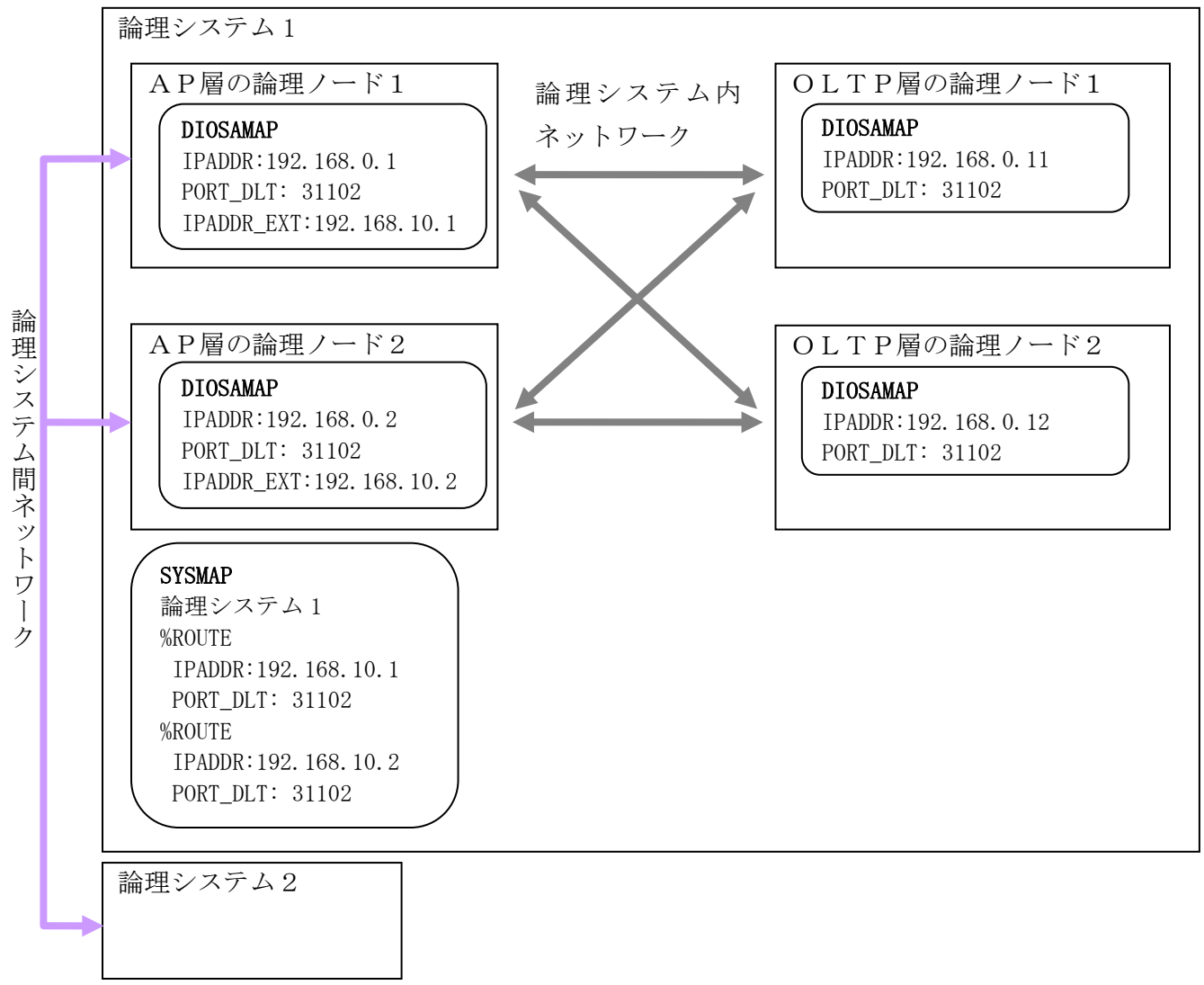


(3)　定義構成例

(a)　他論理システムとの接続

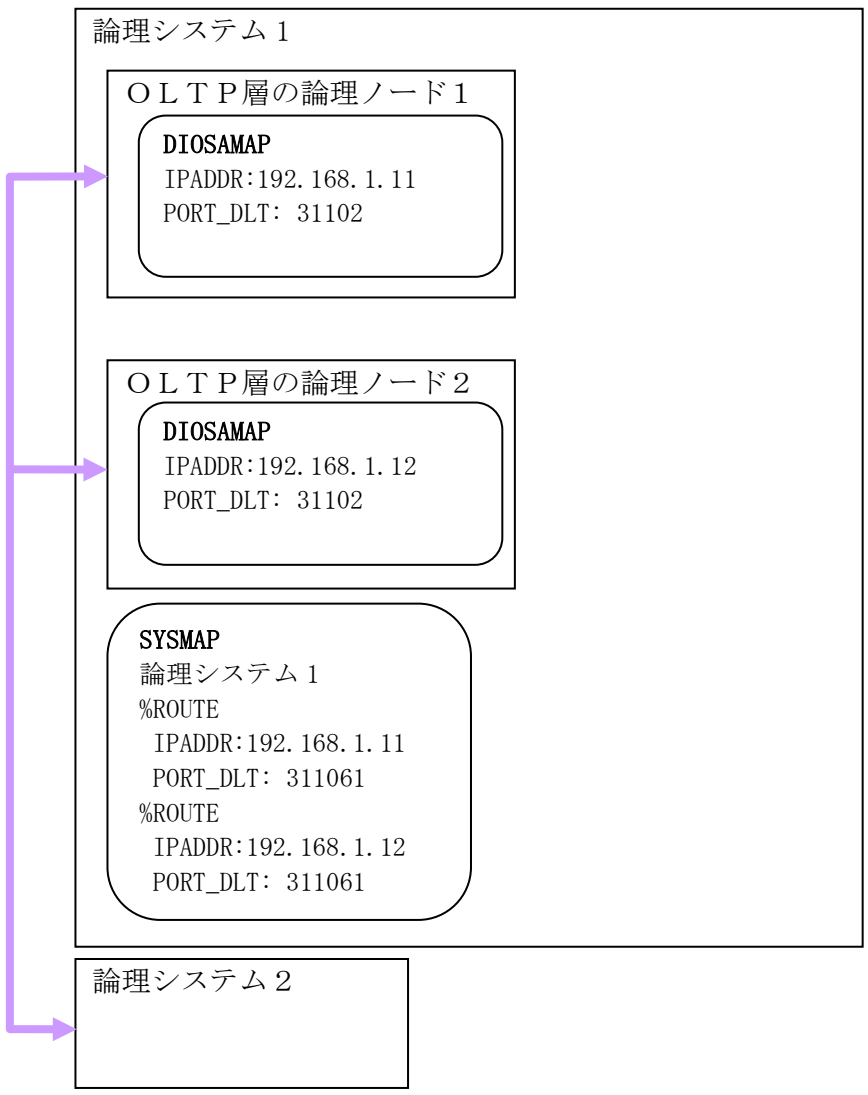
他論理システムと接続するための IP アドレスを SYSMAP の IPADDR に定義します。この IP アドレスが論理システム内で使用するネットワークと別になる場合は、DIOSAMAP の IPADDR_EXT にも同じアドレスを指定する必要があります。

具体的には下図の様に設定します。



(b)　APノードを使用しない場合

APノードを使用しない場合は、SYSMAPのIPADDRにOLTPノードのIPアドレスを指定します。また、論理システム内で使用するネットワークと外部ネットワークが同一であれば、SYSMAPのIPADDRとDIOSAMAPのIPADDRは同一のIPアドレスを指定します(IPADDR__EXTは使用しません)。


論理システム１
ＯＬＴＰ層の論理ノード１
DIOSAMAP
IPADDR:192.168.1.11
PORT_DLT: 31102
ＯＬＴＰ層の論理ノード２
DIOSAMAP
IPADDR:192.168.1.12
PORT_DLT: 31102
SYSMAP
論理システム１
%ROUTE
IPADDR:192.168.1.11
PORT_DLT: 311061
%ROUTE
IPADDR:192.168.1.12
PORT_DLT: 311061
論理システム２

2.3.3 DB 関連(DBCTRL)

(1) DB 監視機能用環境定義(DBCTRL 節)

DB 監視機能用環境定義(DBCTRL 節)では、Oracle DB のヘルスチェック機能、および Oracle DB への接続に関する以下の項目を定義します。

- DB 監視機能制御用定義(CONTROL 項)
- データベース・インスタンスの定義(INSTANCE 項)
- リソースグループセットの定義(RGSET 項)
- リソースグループ ID の定義(RGIDSET 項)
- インスタンスグループの定義(INSTANCEGRP 項)(省略可)

データベースのヘルスチェックは、INSTANCE 項に定義されている全てのデータベース・インスタンスに対して実施されます。データベース・インスタンスのヘルスチェックを制御するための情報(ヘルスチェック間隔、障害と判定するまでの SQL 発行リトライ回数等)を CONTROL 項で定義します。

データベースのヘルスチェック時、および diosadbconnect 等の DB 接続関数を実行した際に、データベースへの接続を行います。データベースへの接続には、データベース接続出口関数を使用する方法と、INSTANCE 項に定義されたユーザ ID/パスワードを使用して接続する方法があります。

データベース接続出口関数を使用する場合は、CONTROL 項の CONNECTEXIT パラメータを定義します。

データベース接続出口関数を使用しない場合は、CONTROL 項の CONNECTEXIT パラメータを定義せず、代わりに INSTANCE 項の USERID、PASSWORD パラメータを定義します。

なお、環境変数「DIOSA_NCM_DBEXIT」が定義されている場合は環境変数が優先されるため、環境定義の内容に依らず、環境変数で指定されたデータベース接続出口関数を使用してデータベースへの接続が行われます。

以下、各項の説明を行います。詳細については、環境定義リファレンスを参照してください。

(a) DB 監視機能制御用定義(CONTROL 項)

ヘルスチェック間隔などの制御に関する定義を行います。

データベースへの接続にデータベース接続出口関数を使用する場合は、本項に CONNECTEXIT パラメータを定義します。

(i) 稼動 DB ヘルスチェック処理間隔時間(INTERVAL1)

正常稼働中のデータベース・インスタンスをヘルスチェックする間隔を定義します。

(ii) 障害 DB ヘルスチェック処理間隔時間(INTERVAL2)

障害中のデータベース・インスタンスを復旧検出するためのヘルスチェック間隔を定義します。

(iii) SQL 発行のリトライ回数(RETRY)

データベース・インスタンスを障害と判定するまでの SQL 発行リトライ回数を定義します。

(iv) デフォルトで使用するリソースグループセット名(DEFAULTRGSET)

バッチ AP 制御においてリソースグループセット名が省略された場合、またはリソースグループに依存しない処理の場合に、デフォルトで使用するリソースグループセットを指定します。

(v)　データベース接続出口関数名(CONNECTEXIT)

データベースに接続するためのデータベース接続出口関数名を指定します。

(vi) データベース接続多重度(MULTI)

データベースに同時に接続できる最大多重度を定義します。

(vii)DB ヘルスチェックデーモンが使用するポート番号(PORTNUM)

DB ヘルスチェックデーモンがソケット通信の際に使用するポート番号を定義します。

(b)　データベース・インスタンスの定義(INSTANCE 項)

データベース・インスタンスの情報に関する定義を行います。

データベースへの接続にデータベース接続出口関数を使用しない場合は、本項に USERID と PASSWORD パラメータを定義します。

(i)　ネット・サービス名(DBNAME)

ネット・サービス名を指定します。

(ii) データベースノード名(DBLNODENAME)

データベースノード名(論理ノード名)を指定します。

(iii)ユーザ名(USERID)

ユーザ名を指定します。

ユーザ名と DB 監視機能制御用定義(CONTROL 項)のデータベース接続出口関数名の両方が定義されている場合は、データベース接続出口関数の設定が優先されます。

(iv) パスワード(PASSWORD)

パスワードを指定します。

パスワードと DB 監視機能制御用定義(CONTROL 項)のデータベース接続出口関数名の両方が定義されている場合は、データベース接続出口関数の設定が優先されます。

(v)　データベース・インスタンスの初期状態(INITSTATUS)

COLD モード起動時のデータベース・インスタンスの初期状態(稼動/待機)を定義します。

- ACTIVE：稼動系データベース・インスタンス
- STANDBY：待機系データベース・インスタンス

(c)　リソースグループセットの定義(RGSET 項)

リソースグループが使用するデータベース・インスタンスを定義します。

本項は､diosadbconnect 等の DB 接続関数で指定するリソースグループ ID/リソースグループセット名と、INSTANCE 項で定義されたデータベース・インスタンスとを紐付けるために使用します。

(i)　リソースグループセット名(NAME)

リソースグループセットの名称を定義します。

(ii) 初期接続先データベース・インスタンス(INITINSTANCE)

COLD モード起動時の初期接続先データベース･インスタンスをネット･サービス名(INSTANCE 項の DBNAME パラメータで指定した名前)で指定します。初期接続先は、稼動系データベース・インスタンス(INSTANCE 項の INITSTATUS パラメータが ACTIVE)である必要があります。

(d)　リソースグループ ID の定義(RGIDSET 項)

リソースグループに紐付くリソースグループ ID を定義します。本項は、RGSET 項内に定義する必要があります。

リソースグループ ID の定義方法には、RGID パラメータを使用して 1 つずつ定義する方法と、RGIDMIN/RGIDMAX パラメータを使用して範囲指定する方法の二種類があります。

(i)　リソースグループ ID(RGID)

リソースグループセットに含まれるリソースグループ ID を定義します。

(ii) 範囲指定するリソースグループ ID の下限値(RGIDMIN)

リソースグループセットに含まれるリソースグループ ID を範囲指定する際の下限値を定義します。この定義は範囲指定するリソースグループ ID の上限値とセットで指定します。

(iii)範囲指定するリソースグループ ID の上限値(RGIDMAX)

リソースグループセットに含まれるリソースグループ ID を範囲指定する際の上限値を定義します。この定義は範囲指定するリソースグループ ID の下限値とセットで指定します。

(e)　インスタンスグループの定義(INSTANCEGRP 項)

データベース・インスタンスのペアをインスタンスグループとして定義します。

接続中のデータベース・インスタンスで障害が発生した場合は、同一インスタンスグループ内に定義されている、もう一方のデータベース・インスタンスへと自動的に接続先が切り替わり、動作を続けることが可能です。

本項を定義しない場合は、INSTANCE 項の先頭 2 つ(INSTANCE 項が 1 つしか定義されていない場合は、その 1 つ)のデータベース・インスタンスがデフォルトインスタンスグループとして取り扱われます。

(i)　インスタンスグループ名(NAME)

インスタンスグループの名前を定義します。

(ii) ネット・サービス名(DBNAME1、DBNAME2)

インスタンスグループに含まれるネット・サービス名(INSTANCE 項の DBNAME パラメータで指定した名前)を 2 つまで定義します。

2.3.4　　CO 制御関連(COCENV)

CO 制御用環境定義(COCENV 節)では、TPBASE の TPP として動作する CO 制御の実行環境を定義します。

(1)　　COCENV 節

(a)　共通項定義(COMMON 項)

CO 制御全体の共通項目を定義します。CO が異常終了要求を返した際のプロセス停止有無、経過時間超過時の所作、プロセス開始、終了時のメッセージ出力有無等を定義することができます。

(b)　利用者出口項定義(EXIT 項)

利用者出口有無と出口名を定義します。出口名が定義されているものは出口有りと判断します。

CO 制御全体に対する共通的な定義となります。出口にはプロセス初期化、プロセス終了、受信電文解析、トランザクション初期化、トランザクション終了、アボート＃1、そしてアボート＃2 出口があります。

(c)　既定クラス項定義(DEF_CLASS 項)

この定義は実際存在するクラス定義ではなく、以降定義されるクラス(CLASS)項の既定値となります。SHARE_CLASS 項、CLASS 項定義で省略されたパラメータは本項の定義が採用されます。

受信電文最大長、更新 DB のタイプ、電文保留トランザクション等を定義します。

(d)　既定トランザクション項定義(DEF_TRNS 項)

この定義は実際存在するトランザクション定義ではなく、以降定義されるトランザクション(TRNS)項の既定値となります。TRNS 項定義で省略されたパラメータは本項の定義が採用されます。

リトライ回数、CPU 時間制限値、経過時間制限値、稼働統計情報採取有無、CO グループ名等を定義します。

(e)　CO グループ項(COGROUP 項)

CO 名をグループ化する CO グループ名を定義します。本項を省略すると CO 名指定の電文送信 API(diosasendtx)が利用できません。

(f)　CO 名項(CO 項)

グループ化する CO 名を定義します。

(g)　ノードタイプ項(NODETYPE 項)

TPBASE 情報をノードタイプ別にを定義します。ノードタイプは、AP ノード、OLTP ノード、ノードタイプ無しが定義できます。ノードタイプ無しは、AP、OLTP ノード全てに共通する TPBASE 情報となります。AP、OLTP ノードを指定した場合は、各ノードタイプ配下に共通するの TPBASE 情報、または個別ノード毎の TPBASE 情報となります。

(h)　共有クラス項(SHARE_CLASS 項)

ノードタイプで定義した全てのノードに共通するクラス定義となります。本項で定義されなかったパラメータは DEF_CLASS 項のパラメータ値が採用されます。

(i)　共有トランザクション項(SHARE_TRNS 項)

SHARE_CLASS 項に共通するトランザクション情報定義となります。本項で定義されなかったパラメータは DEF_TRNS 項のパラメータ値が採用されます。

(j)　ノード項(NODE 項)

NODE 毎に任意の TPBASE 情報を定義します。

(k)　TPBASE モニタ項(TPM 項)

TPBASE 毎の任意情報を定義します。

(l)　クラス項定義(CLASS 項)

TPBASE モニタ項のクラスに対応します。TPBASE 定義のクラス名と合わせます。

クラスで受信できる最大電文長を定義することができます。受信電文長を超える電文は受信が拒否されますので注意してください。

また、電文保留機能を使う場合は、電文保留用トランザクション ID を定義することが可能です。電文保留用トランザクション未定義では、そのクラスから電文保留機能は使うことができません。

本項で定義されなかったパラメータ値は、SHARE_CLASS 項のパラメータ値が採用されます。SHARE_CLASS 項が存在しない場合は、DEF_CLASS 項のパラメータ値が採用されます。

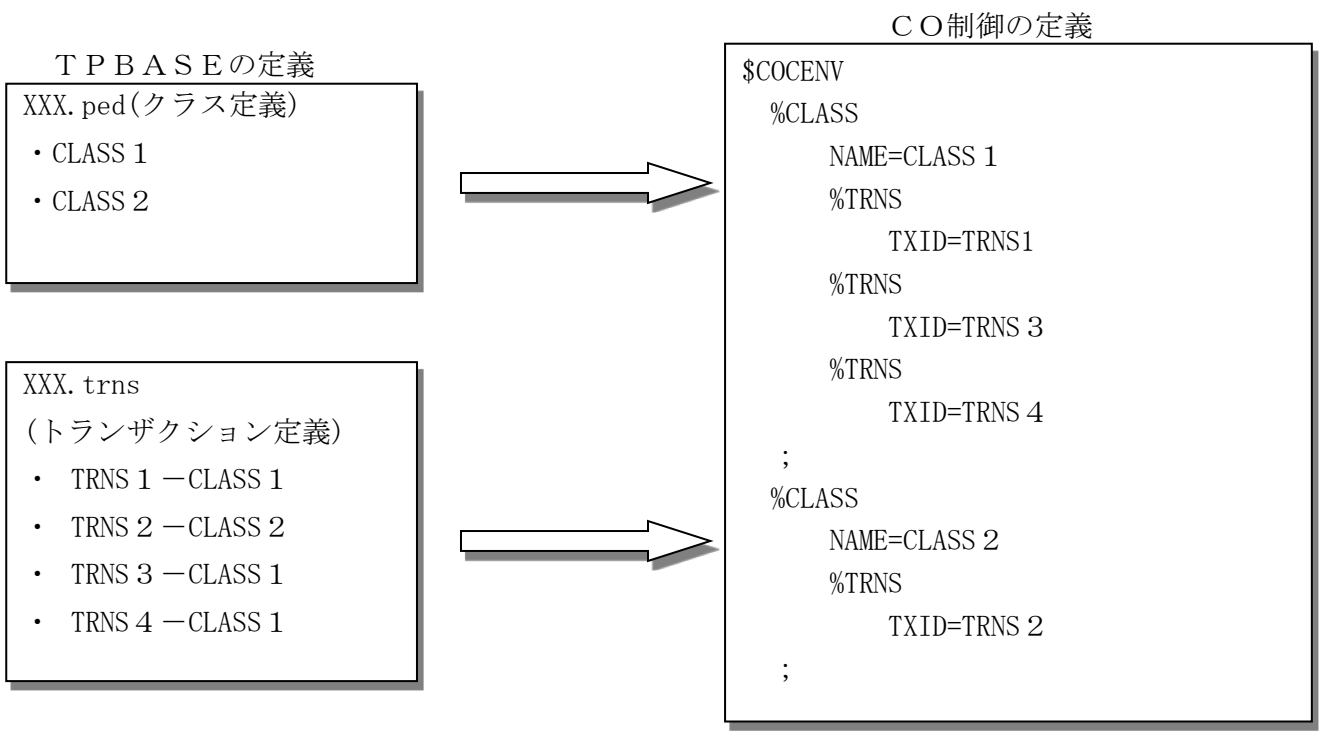
(m)　トランザクション項定義(TRNS 項)

CLASS 項配下のトランザクション情報を定義します。ここでは、CO 制御機能としての環境定義をおこないます。リトライ回数、経過監視時間、CPU 時間、稼動統計採取有無等の定義をおこないます。また、トランザクションで動作可能な CO グループ名の定義を行います。

本項で定義されなかったパラメータ値は、SHARE_TRNS 項のパラメータ値が採用されます。SHARE_TRNS 項が存在しない場合は、DEF_TRNS 項のパラメータ値が採用されます。

(n)　TPBASE 定義と COCENV 節

TPBASE の定義は、クラス、トランザクションのようにカテゴリで分別して定義しますが、COCENV は関係性を階層化して定義します。

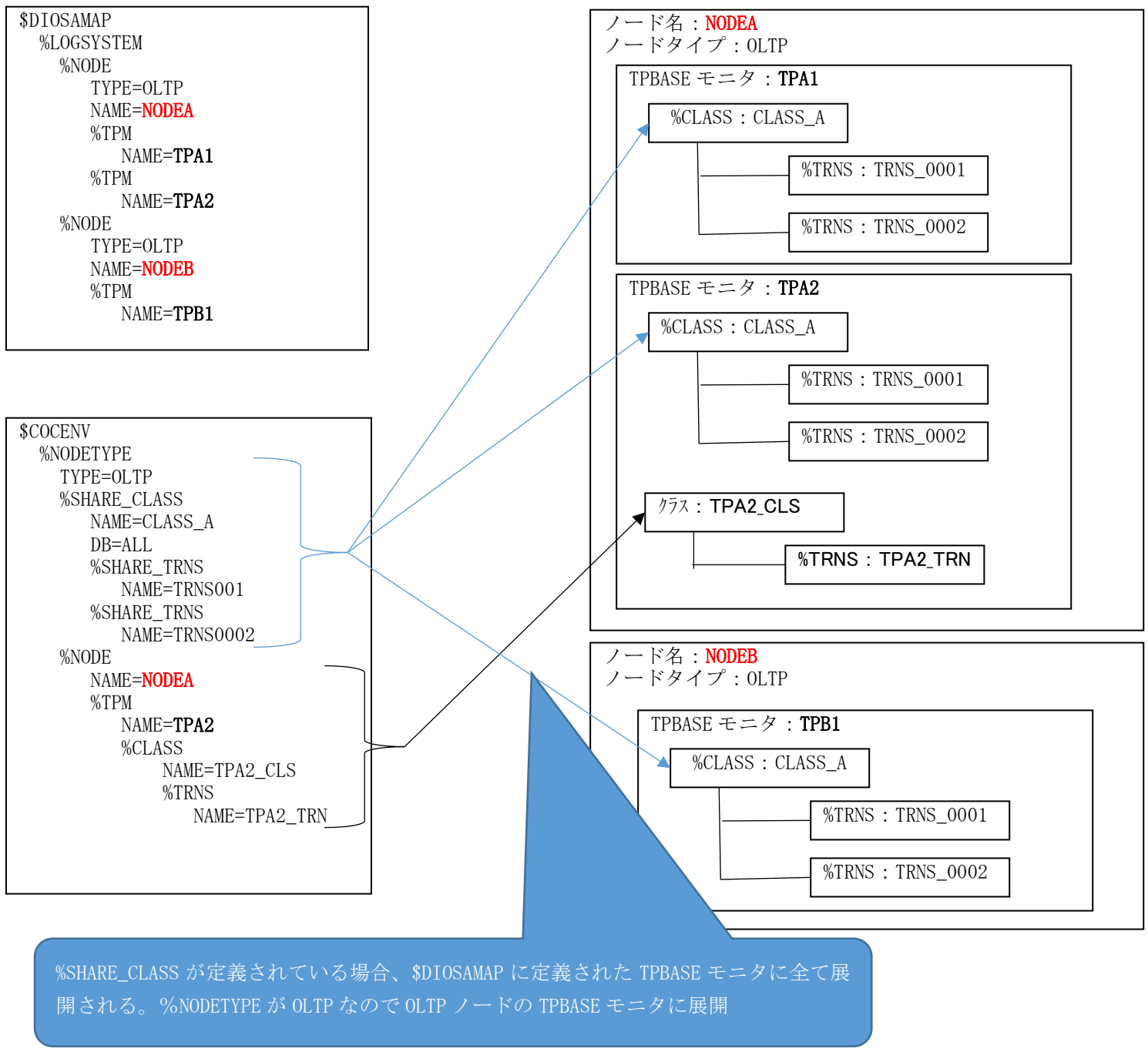


(o)　SHARE_xxx 項と＄DIOSAMAP 節

SHARE_CLASS 項、SHARE_TRNS 項は、複数 TPBASE に共通の情報となります。NODETYPE 項に TYPE＝OLTP が指定された場合、%NODETYPE 配下に定義された SHARE_xxx が OLTP ノードで動作可能な TPBASE モニタ全てに定義されていると解釈されます。論理ノードと TPBASE モニタの配置情報は＄DIOSAMAP に定義されていなければなりません。CO 制御は初期化コマンド(dicocinit)が実行された論理ノードの TPBASE モニタ名を＄DIOSAMAP から取得し、SHARE_xxx を展開します。

%NODE－%TPM 項(個別の TPBASE モニタ情報)が定義されている場合は、SHARE_xxx(未定義でも可)に加えて、%TPM の定義も展開されます。

SHARE_xxx と＄DIOSAMAP の関係を示します。



2.3.5　　部品関連(APLIB、MMG、OPSENV、TMCENV)

(1)　　APLIB(アプリケーション動的置換機能)

アプリケーション動的置換機能を利用するには以下のいずれかをおこなう必要があります。

・環境定義 APLIB 節を定義する

・環境変数 DIOSA_LIBNAME を設定する

・環境変数 DIOSA_LIBNAME_NODR を設定する

これらの呼び出し手段は 1 つの LM から複数利用可能ですが、関数呼び出し時の処理順序としては、まず環境定義の定義情報を参照して関数呼び出しをおこない、呼び出せなかった場合に環境変数(DIOSA_LIBNAME_NODR, DIOSA_LIBNAME の順)による呼び出しをおこないます。

(a)　環境定義 APLIB 節を定義する

APLIB 節に LM 項、LLIB 項、LIBRARY 項、FUNC 項を定義することで関数呼び出しが可能となります。

FUNC 項を定義している関数のみが呼び出し可能で、LIBRARY 項に定義したライブラリに含まれている関数でも、FUNC 項が定義されていない場合は、呼び出すことはできません。

DIOSA/XTP で動作する各プロセスを APLIB 節に定義する場合の LM 名、関数名は以下のようになります。(環境変数 DIOSA_LIBNAME, DIOSA_LIBNAME_NODR に%F で指定する文字列も、下記の関数名になります。)

プロセスの種類	LM 名	関数名
DIOSA/XTP コマンド	コマンド名	出口関数名(※1)
DIOSA/XTP デーモン	デーモンプロセス名	出口関数名(※1)
CO 制御サーバ	TPBASE の TPP 名	出口関数名 CO 名
バッチアプリケーション	バッチ AP 制御実行コマンド名 (dibacexec)	コマンドパラメータで指定する CO 名、出口関数名
ユーザバッチ	プロセス名(※2)	呼び出し関数名
ユーザデーモン	プロセス名(※2)	呼び出し関数名

(※1) 出口の呼び出しをおこなわないコマンド、デーモンでは定義不要です。

(※2) シンボリックリンクを作成して LM を実行する場合、リンク先ではなく、作成したシンボリックリンクの名前を LM 名として定義する必要があります。

環境定義の詳細については、リファレンスマニュアルを参照してください。

(b)　環境変数 DIOSA_LIBNAME(DIOSA_LIBNAME_NODR)を設定する

各プロセスの起動時に、環境変数 DIOSA_LIBNAME(DIOSA_LIBNAME_NODR)に指定したライブラリ内の関数は呼び出し可能です。ライブラリはコロン(:)で区切って二つの環境変数の合計で 100 ライブラリまで指定することが可能で、複数指定した場合は先頭のライブラリから順にロード、関数検索処理を実行します。

ライブラリ名が関数名を含む場合、%F を記述することで、その部分は関数名に置換されます。

環境変数の詳細については、リファレンスマニュアルを参照してください。

(2)　　MMG(メモリ管理機能)

メモリ管理機能の利用有無に関わらず、環境定義 MMG 節を定義する必要があります。メモリ管理機能を利用しない場合、MMG 節のみを定義します。

メモリ管理機能を利用する場合、MMG 節に利用する領域種別に応じて初期サイズ、拡張サイズ、最大サイズの設定、及びプログラムアボート時のメモリダンプ出力設定を行います。

プロセスメモリに関しては、環境変数 DIOSA_MEMBUF で初期サイズ、拡張サイズ、最大サイズを設定することができます。環境定義と環境変数の両方を設定した場合、環境変数の値が有効となります。

(a)　メモリサイズ設定

メモリ管理で取り扱うメモリは、アプリケーションからのメモリ割り当て要求時に既に確保済みのメモリを切り出して利用します。以下に初期サイズ、拡張サイズ、最大サイズの設定について説明します。

(i)　初期サイズ(WRINIT、RDINIT)

領域種別毎に必要なメモリの初期サイズを設定します。

以下のように領域種別毎にメモリを確保するタイミングが異なります。

・更新可共有メモリ/保護属性共有メモリ

DIOSA/XTP 起動時に初期サイズで指定したサイズの共有メモリを確保します。

・一括解放対象外プロセス内メモリ/一括解放対象外スレッド内メモリ

初回のアプリケーションからのメモリ割り当て要求時に初期サイズで指定したサイズのプロセスメモリを確保します。

・一括解放対象サービス内メモリ

DIOSA/XTP のプロセス初期化時に初期サイズで指定したサイズのプロセスメモリを確保します。

DIOSA/XTP のプロセス初期化処理を呼び出す全プロセス(メモリ管理機能を利用しないプロセス含む)で、初期サイズで設定したプロセスメモリが確保されます。

＜注意事項＞

初期サイズで設定したサイズのプロセスメモリが全 CO 制御サーバプロセスで確保されるため、初期サイズの設定には考慮が必要となります。例えば、初期サイズを小さく設定し、拡張サイズを大きく設定することで必要なメモリが効率よく確保できます。

(ii)　拡張サイズ(WRINC、RDINC)

領域種別毎に必要なメモリの拡張サイズを定義します。アプリケーションからのメモリ割り当て要求時、セグメント内の空き領域がなくなった場合に動的に拡張サイズのメモリを確保します。

＜注意事項＞

拡張領域は既存セグメントとは別の領域にメモリを確保する(既存の領域と連続領域とはならない)ため、アプリケーションからの 1 回のメモリ割り当てで要求する最大のメモリサイズを、拡張サイズで設定する必要があります。

(iii)最大サイズ(WRMAX、RDMAX)

領域種別毎に必要なメモリの最大サイズを定義します。

このサイズを超えるメモリの拡張は行われません。この場合、メモリ割り当て API は DIOSA_ENOBUFS(-7)が返却されます。

(b)　プログラムアボート時のメモリダンプ出力設定

以下にプログラムアボート時のメモリダンプ出力設定について説明します。

(i)　利用者用メモリアボートダンプ出力可否(APWR、APRD、PRC、THR、SVC)

プログラムアボート時の領域種別毎にアボートダンプ出力可否を設定します。

アボートダンプを出力できるプロセスは、CO 制御サーバプロセスとバッチ AP 制御プロセスが対象となります。

(ii) 利用者用メモリアボートダンプ出力先(DIR)

アボートダンプファイルの出力ディレクトリを設定します。省略した場合、プロセス起動時のカレントディレクトリとなります。

＜注意事項＞

設定するディレクトリ(省略時はプロセス起動時のカレントディレクトリ)のパーミッションは、メモリ管理の使用有無に関わらず、プロセス実行ユーザの書込み権を付与する必要があります。

(iii)内部用メモリアボートダンプ出力可否、出力先(PRC、DIR)

プログラムアボート時のアボートダンプ出力可否とアボートダンプ出力ディレクトリを設定します。

省略した場合、プロセス起動時のカレントディレクトリとなります。

＜注意事項＞

設定するディレクトリのパーミッションは、利用者用メモリアボートダンプ出力先と同様にプロセス実行ユーザの書込み権を付与する必要があります。

環境変数・環境定義の詳細については、リファレンスマニュアルを参照してください。

(3)　　OPSENV(稼動統計機能)

稼動統計機能を利用するためには環境定義 OPSENV 節を定義する必要があります。

(a)　OPSPARAM 項

稼動統計機能の動作環境として稼動情報ファイル関連の定義を行います。

以下に各設定値について説明します。

(i)　FTP ユーザ名、パスワード

稼動統計収集コマンドにて稼動情報ファイルを収集時に使用する FTP のログインユーザ名とパスワードを指定します。省略した場合、収集コマンドでのファイル転送は利用できません。

(ii) 収集ディレクトリパス

稼動統計収集コマンドにて稼動情報ファイルを収集するディレクトリパスを絶対パスで指定します。

(iii)出力ディレクトリパス

稼動情報ファイルを格納するディレクトリパスを絶対パスで指定します。

(iv) プレフィックス

稼動情報ファイル名のプレフィックス(接頭語)を指定します。

省略時は”OPS”が設定されます。

(v)　稼動統計ファイルのサイズ

稼動情報ファイルの 1 ファイルのサイズを指定します。

単位は M バイトで、1～100 まで指定可能です。省略時は 50 が設定されます。

(vi) 稼動統計ファイルの個数

稼動情報ファイルの個数を指定します。

2～100 まで指定可能で、省略時は 3 が設定されます。

(vii)稼動統計ファイルセットの個数

稼動情報ファイルのセット数を指定します。

1～99 まで指定可能で、省略時は 1 が設定されます。

(viii) 強制出力間隔

メモリバッファ中の稼動統計情報をファイルに強制出力する最大経過時間を指定します。

秒単位で 0～3600 まで指定可能です。0 の場合には強制出力は行われません。

省略時は 90 が設定されます。

(4)　　TMCENV(タイマ制御機能環境定義)

SG タイマ登録を行うためには環境定義 TMCENV 節を定義する必要があります。

DIOSA 起動時に ditmcsgset を実行することで、SG に定義された情報でタイマ登録することができます。

タイマ制御機能の動作自体に影響はありません。

ただし SG に定義するタイマ情報の最大数は、環境変数 DIOSA_TMCTBLMAX と整合性を取る必要があります。(最大 2048 個、環境変数省略時のデフォルト値 512 個)

環境変数・環境定義の詳細については、リファレンスマニュアルを参照してください。

2.3.6　　電文保証(APMGRNT)

電文法相の環境定義は、DIOSA/XTP の APMGRNT 節と DB(Oracle,TAM)の 2 つを定義します。

(1)　　APMGRNT 節

電文保証機能の基本設定を行います。設定項目については「環境定義リファレンス」の「第Ⅱ編 環境定義」「2.2 APMGRNT(電文保証機能)」を参照してください。APMGRNT 節の定義を省略した場合は、既定値で動作します。

(2)　　DB(Oracle,TAM)

電文保証機能が使用する以下の 4 つの表を DB 上に作成します。()内は Oracle の表名および TAM(メモリキャッシュ)の論理表名です。

- 送信管理表(DIOSA_MSGGNT_SNDCTL)
- 送信電文保存表(DIOSA_MSGGNT_SNDMSG)
- 順序性保証グループ管理表(DIOSA_MSGGNT_SEQGRP)
- 受信管理表(DIOSA_MSGGNT_RCVCTL)

Oracle 表の定義は /opt/diosa_xtp/samples/sql/create_table_gnt.sql を参照してください。

TAM 表の定義は TAM の table.conf および DIOSA/XTP のメモリキャッシュ IMTABLECONF 節の定義を行います。定義サンプルが/opt/diosa_xtp/sample/tam にあります。table.conf については table.conf を、IMTABLECONF 節については imtableconf.edl を参照してください。メモリキャッシュの定義方法については「メモリキャッシュ 利用の手引」を参照してください。

2.3.7　　運用関連(CMDSEND)

(1)　　CMDSEND 節(コマンド配信機能)

CMDSEND 節には、コマンド配信機能の動作環境を定義します。

(a)　コマンド配信動作環境定義(CMDSENDINFO 項)

CMDSENDINFO 項では、コマンド配信機能の動作環境を定義します。

(i)　コマンド配信動作情報

コマンド配信時の動作情報として、タイムアウト時間や接続リトライ指定等を定義します。

タイムアウト時間は、APITIMEOUT(コマンド配信 API の応答待ち合わせ時間)と EXECTIMEOUT(ノード間の応答待ち合わせ時間)の 2 つについて定義します。また、配信先ノードへの接続失敗時にリトライする場合、RTRYCNT(接続リトライ回数)と RTRYINTVL(リトライインターバル時間)を定義します。

コマンド配信の結果を API 側で取得したい場合、以下の条件を満たすように各値を指定してください。

- APITIMEOUT 値 > EXECTIMEOUT 値＋(RTRYCNT 値×RTRYINTV 値)

なお、これらの値についてはコマンド配信を行う際のパラメータとして指定された場合、パラメータで指定された値が優先されます。

(ii)　コマンド配信履歴ファイル情報

コマンド配信履歴を採取する場合、HSTTYPE に”NO”以外の値を指定します。コマンド配信履歴ファイルは、「HSTFLNAME で指定した絶対パスファイル名_論理ノード名.ファイル番号」として生成されます。生成さ

れたコマンド配信履歴ファイルは、HSTFLMAXSIZE で指定されたファイルサイズに達するまで出力され、ファイル番号が HSTFLMAXCNT で指定した値でサイクリックに利用されます。

(b)　サーバグループ構成定義(SRVGRPINFO 項)

SRVGRPINFO 項では、複数の論理ノードの集合をサーバグループとして定義します。

本項で定義したサーバグループは、コマンド配信する際の配信先として指定することが可能となります。

(c)　コマンドルーティング定義(CMDROUTING 項)

CMDROUTING 項では、コマンドに対する配信先情報を定義します。これにより、コマンド配信を行う際の配信先パラメータを省略することができ、配信先を意識することなくコマンド配信を利用することができます。

コマンドルーティングを利用する場合、対象とするコマンド毎に、以下の配信先情報を定義します。省略した場合既定値で配信先を決定します。

- 配信宛先名(論理ノード名、論理システム名、サーバグループ名)
- 論理ノード属性種別(全ノード、AP ノード、OLTP ノード、DB ノード)
- 配信対象種別(全ノード、任意の 1 ノード)
- 配信元対象(配信元ノードを配信対象に含める、含めない)

なお、上記の配信先情報を定義しているコマンドについてコマンド配信要求する際に、配信先パラメータとしてコマンドルーティングと異なる配信先情報が指定された場合、配信先パラメータで指定された情報が優先されます。

(d)　コマンド実行権限定義(CMDPERM 項)

ユーザ及びユーザグループ毎に配信可能なコマンドを制限する場合、CMDPERM 項を定義します。

本項では、制限対象とするコマンド群を CMDGRP 項に定義し、ユーザ及びユーザグループ毎に各コマンド群に対する実行権限(実行可・実行不可)を EXPERM 項に定義します。

EXPERM 項に定義されていないコマンドの実行権限については、CDMPERM 項-DFLTPERM の定義に従い、実行可・不可が決定されます。

なお、CMDPERM 項を省略した場合、全てのコマンドが実行可能となります。

2.3.8　　TPBASE の環境定義

パス制御機能、CO 制御機能を使用するための定義を行います。ここでは、DIOSA/XTP のインストール先が"/opt/diosa_xtp"、TPBASE のインストール先が"/usr/OLTP"であると仮定して、各ファイルパスに関する設定値を提示しています。

以降に記載のないパラメータは既定値を想定しています。システム要件に合わせて変更する場合は、十分な評価の上、システムへ適用してください

(1)　　TPBASE のシステム設定

(a)　自動起動定義(autoup.cnf)

TPBASE 起動時に自動組み込みを行う環境として、下記を定義します。()内は SG 定義ファイルのファイル名であり、＊は TPBASE の命名規則に準じた任意の名前です。

- システム TPP のプロセス環境定義ファイル名(systpp.ped)
- タイマデーモンのプロセス環境定義ファイル名(timer.ped)
- VD サーバのプロセス環境定義ファイル名(vdserver.ped)
- DIOSA 制御用プロセスのプロセス環境定義ファイル名(*.ped)
- DIOSA 制御用プロセスの AP 環境定義(*.ap)
- DIOSA 制御用 VD の VD 環境定義ファイル名(*.vd)
- 論理ノード間通信用リスナプロセスのプロセス環境定義ファイル名(*.ped)
- 論理ノード間通信用の端末定義ファイル名(*.term)
- 論理ノード間通信用の VD 定義ファイル名(*.vd)
- 論理システム間通信用リスナプロセスのプロセス環境定義ファイル名(*.ped)
- 論理システム間通信(常時接続)用の外部接続用の端末定義ファイル名(*.term)
- 論理システム間通信(常時接続)用の外部接続用の VD 定義ファイル名(*.vd)
- 論理システム間通信(都度接続)用の外部接続用の端末定義ファイル名(DEFAULT.term)

(b)　TPBASE 構成定義(tpbase.cnf)

TPBASE 構成定義に関する制限事項はありません。

(c)　通信制御機能定義(mcs)

キーワード	設定値	補足
EXCEPTSIG	QUIT ILL FPE BUS SEGV XCPU ABRT	
RESPCONNECT	YES	DIOSA が通信パスを制御するために必要です。
RESPTROUBLE	YES	DIOSA が通信パスを制御するために必要です。
CONNECTVD	DXTP_NOTIFYVD	端末環境定義(*.term)で NOTIFYVD2 を省略する場合に必要です。
TROUBLEVD	DXTP_NOTIFYVD	端末環境定義(*.term)で NOTIFYVD1 を省略する場合に必要です。
DISCERR	PURGE	端末環境定義(*.term)で同キーワードを省略する場合に必要です。
SYSMSGLANG	ENGLISH	

VDRECVBUF	32768～2147483647	送受信電文サイズが 32K を超える場合に設定します。

(d)　システム TPP のプロセス環境定義(systpp.ped)

システム TPP のプロセス環境定義に関する制限事項はありません。

(e)　タイマデーモンのプロセス環境定義(timer.ped)

タイマデーモンのプロセス環境定義に関する制限事項はありません。

(f)　VD サーバのプロセス環境定義(vdserver.ped)

キーワード	設定値	補足
PATH	/usr/OLTP/bin/vdserver	
ARGS	-T 0	

(2)　DIOSA 制御用プロセスの AP 環境定義(*.ap)

キーワード	設定値	補足
APID	任意の値	

(3)　DIOSA 制御用プロセスのプロセス環境定義(*.ped)

(a)　CO 制御 TPP

キーワード	設定値	補足
CLASS	任意の値	
PATH	/opt/diosa_xtp/bin/dicoctpp	フルパスで指定する
PROEPILIB	/opt/diosa_xtp/lib/libdxtpcocprotpb.so	フルパスで指定する
PROEPILANG	C	
PROENTRY	di_coc_CocPrologue	
EPIENTRY	di_coc_CocEpilogue	

(b)　パス制御 TPP

キーワード	設定値	補足
CLASS	任意の値	
PATH	/opt/diosa_xtp/bin/dincmctrltpp	フルパスで指定する
PROEPILIB	/opt/diosa_xtp/lib/libdxtpncmtpppproepi.so	フルパスで指定する
PROEPILANG	C	
PROENTRY	di_ncm_TppPrologue	
EPIENTRY	di_ncm_TppEpilogue	

(c)　電文保証 電文再送 TPP

キーワード	設定値	補足
CLASS	任意の値	
PATH	/opt/diosa_xtp/bin/digntrsdtpp	フルパスで指定する

PROEPILIB	/opt/diosa_xtp/lib/libdxtpgntresendproepi.so	フルパスで指定する
PROEPILANG	C	
PROENTRY	di_gnt_TppResendInit	
EPIENTRY	di_gnt_TppResendTerm	

(d)　電文保証 応答電文受信 TPP

キーワード	設定値	補足
CLASS	任意の値	
PATH	/opt/diosa_xtp/bin/digntfintpp	フルパスで指定する
PROEPILIB	/opt/diosa_xtp/lib/libdxtpgntrcvfinproepi.so	フルパスで指定する
PROEPILANG	C	
PROENTRY	di_gnt_TppRcvFinInit	
EPIENTRY	di_gnt_TppRcvFinTerm	

(e)　電文保証 受信情報削除 TPP

キーワード	設定値	補足
CLASS	任意の値	
PATH	/opt/diosa_xtp/bin/digntdeltpp	フルパスで指定する
PROEPILIB	/opt/diosa_xtp/lib/libdxtpgntdelrcvproepi.so	フルパスで指定する
PROEPILANG	C	
PROENTRY	di_gnt_TppDelRcvInit	
EPIENTRY	di_gnt_TppDelRcvTerm	

(4)　DIOSA 制御用プロセスのトランザクション環境定義(*.trns)

(a)　CO 制御 TPP

キーワード	設定値	補足
TRNSID	任意の値	COCENV 節で指定した値
TPPLIB	/opt/diosa_xtp/lib/libdxtpcoctrntpb.so	フルパスで指定する
TPPENTRY	di_coc_CocTrnsEnt	
TPPLANG	C	
CLASS	*.ped 内 (a) CO 制御 TPP で指定した CLASS	
TRTYPE	IN	
ABORTEXITLIB	/opt/diosa_xtp/lib/libdxtpcocabttpb.so	フルパスで指定する
ABORTEXIT	di_coc_CocExceptMain	
ABORTLANG	C	
ABORT	CONT	
TIMEOVER	CONT	

(b)　パス制御 TPP

キーワード	設定値	補足
TRNSID	DXTPCT	
TPPLIB	/opt/diosa_xtp/lib/libdxtpncmtpptrns.so	フルパスで指定する
TPPENTRY	di_ncm_TppCtrlTx	
TPPLANG	C	
CLASS	*.ped 内の（b）パス制御 TPP で指定した CLASS	
TRTYPE	IN	
ABORTEXITLIB	/opt/diosa_xtp/lib/libdxtpncmtppabort.so	フルパスで指定する
ABORTEXIT	di_ncm_TppAbort	
ABORTLANG	C	
ABORT	CONT	

(c)　電文保証 電文再送 TPP

キーワード	設定値	補足
TRNSID	GNTRSD	
TPPLIB	/opt/diosa_xtp/lib/libdxtpgntresendtrns.so	フルパスで指定する
TPPENTRY	di_gnt_TppResendMain	
TPPLANG	C	
CLASS	*.ped 内の（c）電文保証 電文再送 TPP で指定した CLASS	
TRTYPE	IN	
ABORTEXITLIB	/opt/diosa_xtp/lib/libdxtpgntresendabort.so	フルパスで指定する
ABORTEXIT	di_gnt_TppResendAbort	
ABORTLANG	C	
ABORT	CONT	

(d)　電文保証 応答電文受信 TPP

キーワード	設定値	補足
TRNSID	GNTFIN	
TPPLIB	/opt/diosa_xtp/lib/libdxtpgntrcvfintrns.so	フルパスで指定する
TPPENTRY	di_gnt_TppRcvFinMain	
TPPLANG	C	
CLASS	*.ped 内の（d）電文保証 応答電文受信 TPP で指定した CLASS	
TRTYPE	IN	
ABORTEXITLIB	/opt/diosa_xtp/lib/libdxtpgntrcvfinabort.so	フルパスで指定する
ABORTEXIT	di_gnt_TppRcvFinAbort	
ABORTLANG	C	
ABORT	CONT	

(e)　電文保証 受信情報削除 TPP

キーワード	設定値	補足
TRNSID	GNTDEL	
TPPLIB	/opt/diosa_xtp/lib/libdxtpgntdelrcvtrns.so	フルパスで指定する
TPPENTRY	di_gnt_TppDelRcvMain	
TPPLANG	C	
CLASS	*.ped 内の（e）電文保証 受信情報削除 TPP で指定した CLASS	
TRTYPE	IN	
ABORTEXITLIB	/opt/diosa_xtp/lib/libdxtpgntdelrcvabort.so	フルパスで指定する
ABORTEXIT	di_gnt_TppDelRcvAbort	
ABORTLANG	C	
ABORT	CONT	

(5)　DIOSA 制御用プロセスの VD 環境定義(*.vd)

(a)　CO 制御 TPP の制御電文 VD

キーワード	設定値	補足
VD	COCENV 節 TNRS 項または SHARE_TRNS 項で指定した VD 名を指定する	
VDTYPE	TRNS	
TRNS	COCENV 節 TNRS 項または SHARE_TRNS 項で指定した TXID を指定する	

(b)　パス制御 TPP の制御電文 VD

キーワード	設定値	補足
VD	DXTP_NOTIFYVD	
VDTYPE	TRNS	
TRNS	DXTPCT	

(c)　電文保証 電文再送 TPP の制御電文 VD

キーワード	設定値	補足
VD	DXTP_GNTRSD	
VDTYPE	TRNS	
TRNS	GNTRSD	

(d)　電文保証 応答受信 TPP の制御電文 VD

キーワード	設定値	補足
VD	DXTP_GNTFIN	
VDTYPE	TRNS	
TRNS	GNTFIN	

(e)　電文保証 受信情報削除 TPP の制御電文 VD

キーワード	設定値	補足
VD	DXTP_GNTDEL	
VDTYPE	TRNS	
TRNS	GNTDEL	

(6)　パス制御 論理ノード間通信用の設定

(a)　リスナプロセスのプロセス環境定義(*.ped)

キーワード	設定値	補足
PATH	/usr/OLTP/bin/tcpiplsn	
ARGS	SG_FNAME 〈 (b) の SG パラメータファイルパス〉	フルパスで指定する

(b)　リスナプロセスの SG パラメータ(*.sg)

キーワード	設定値	補足
TPSEXIT_CALL	1	
TPSEXIT_LIBNAME	libdxtpncmlsnr.so	
AUTOTRMGEN	OFF	

(c)　端末環境定義(*.term)

キーワード	設定値	補足
TERMID	DXTP_LNxxxxx_TPMyy_Tzzzz	※1
PROTOCOL	TCPIP	
INITAP	(2)DIOSA 制御用プロセスの AP 環境定義(*.ap)で指定した APID	
INITVD	DXTP_LNxxxxx_TPMyy_Vzzzz	
CONNECTMODE	CONCURRENT	
INITTX	DXTPCT	
INITMSG	NO	
FREEFORMAT	NO	
RESPCONNECT	YES	mcs と端末環境定義の両方に必須
RESPTROUBLE	YES	mcs と端末環境定義の両方に必須
NOTIFYVD1	DXTP_NOTIFYVD	mcs で TROUBLEVD を定義していない場合必須
NOTIFYVD2	DXTP_NOTIFYVD	mcs で CONNECTVD を定義していない場合必須

※1 TERMID の命名規則は以下の通り

- xxxxx は DIOSAMAP 節 LNODE 項-ID で指定した値を、10 進数 5 桁で指定する。(00001～99999)。通信相手のノードの ID を指定する。
- yy は DIOSAMAP 節 LNODE 項-TPM 項-ID で指定した値を 10 進数 2 桁で指定する。(01-16) 通信相手の TPM の ID を指定する。

- zzzz は 0001～9999 の通番を指定する。0001 から始めて、間をあけてはいけない。間が空いた場合、0001 から連続した端末のみが使用される。

(d) VD 環境定義(*.vd)

キーワード	設定値	補足
VD	DXTP_LNxxxxx_TPMyy_Vzzzz	命名規則は端末環境定義の TERMID と同じ
VDSTAT	ENABL	
VDTYPE	TERM	
DISCERR	PURGE	

(7) パス制御 論理システム間通信用(常時接続)の設定

(a) TCP リスナプロセスのプロセス環境定義(*.ped)

キーワード	設定値	補足
PATH	/usr/OLTP/bin/tcpiplsn	
ARGS	SG_FNAME 〈 (b) の SG パラメータファイルパス〉	フルパスで指定する

(b) TCP リスナプロセスの SG パラメータ(*.sg)

キーワード	設定値	補足
TPSEXIT_CALL	1	
TPSEXIT_LIBNAME	libdxtplpmlsnr.so	
TPSEXIT_SEND_WORK	4	
AUTOTRMGEN	OFF	

(c) OLFTP リスナプロセスのプロセス環境定義(*.ped)

キーワード	設定値	補足
PATH	/usr/OLTP/bin/olftplsn	
ARGS	SG_FNAME 〈 (d) SG パラメータファイルパス〉	フルパスで指定する

(d) OLFTP リスナプロセスの SG パラメータ(*.sg)

キーワード	設定値	補足
TPSEXIT_CALL	1	
TPSEXIT_LIBNAME	libdxtplpmlsnr.so	
TPSEXIT_SEND_WORK	4	
TPSEXIT_RECV_WORK	10	
AUTOTRMGEN	OFF	

(e) 端末環境定義(*.term)

キーワード	設定値	補足
TERMID	任意の名前	

PROTOCOL	TCPIP または OLFTP または MQSR	
TYPE	PROTOCOL が MQSR の場合のみ、IN(入力型) または OU(出力型) のどちらかの指定が必須	MQ リスナでは端末属性に IO(入出力型)は未サポート
INITAP	(2)DIOSA 制御用プロセスの AP 環境定義(*.ap)で指定した APID	
INITVD	(f) VD 環境定義(*.vd)の VD	
CONNECTMODE	CONCURRENT	
INITTX	DXTPCT	
INITMSG	NO	
FREEFORMAT	YES	PROTOCOL が TCPIP の場合のみ必須
SYSMSG	NO	
RESPCONNECT	YES	mcs と端末環境定義の両方に必須
RESPTROUBLE	YES	mcs と端末環境定義の両方に必須
NOTIFYVD1	DXTP_NOTIFYVD	mcs で TROUBLEVD を定義していない場合必須
NOTIFYVD2	DXTP_NOTIFYVD	mcs で CONNECTVD を定義していない場合必須

(f) VD 環境定義(*.vd)

キーワード	設定値	補足
VD	任意の名前	
VDTYPE	TERM	
DISCERR	PURGE	

(8) パス制御 論理システム間通信用(都度接続)の設定

(a) TCP リスナプロセスのプロセス環境定義(*.ped)

キーワード	設定値	補足
PATH	/usr/OLTP/bin/tcpiplsn	
ARGS	SG_FNAME 〈 (b) SG パラメータファイルパス〉	フルパスで指定する

(b) TCP リスナプロセスの SG パラメータ(*.sg)

キーワード	設定値	補足
TPSEXIT_CALL	1	
TPSEXIT_LIBNAME	libdxtplpmlsnr.so	
TPSEXIT_SEND_WORK	4	
AUTOTRMGEN	ON	

(c) 端末環境定義(DEFAULT.term)

キーワード	設定値	補足
TERMID	DEFAULT	

PROTOCOL	TCPIP	
INITAP	(2)DIOSA 制御用プロセスの AP 環境定義(*.ap)で指定した APID	
PORTNO_NOCHK	NO	
FREEFORMAT	YES	
SYSMSG	NO	

2.3.9 TAM の環境定義

TAM の環境定義については、以下のマニュアルを参照してください。

・DIOSA/XTP メモリキャッシュ利用の手引　第 4 章 システムの構築

・DIOSA/XTP データストア利用の手引　第 4 章 システムの構築

2.3.10　Oracle データベースの環境定義

DIOSA/XTP を Oracle を利用して動作させるために、Oracle DB の制御表やストアドプロシージャの生成が必要です。なお、データストアの Oracle DB 環境定義についてはデータストア 利用の手引の記述を参照してください。

(1)　Oracle DB 表生成

Oracle DB 表を作成するため、サンプルとして提供している SQL ファイル({DIOSA/XTP インストールディレクトリ}/sample/sql 参照)を実行してください。

(a)　表の生成

SQL ファイルを実行して表を生成します。

それぞれの表の生成用 SQL ファイルは、サンプルの下記ファイルに記述されています。

表名	生成用ファイル	説明
DIOSA_NCM_DBHC_01～32	create_table_ncm.sql	Oracle の死活監視に利用します。
DIOSA_MSGGNT_SNDCTL DIOSA_MSGGNT_SNDMSG DIOSA_MSGGNT_SEQGRP DIOSA_MSGGNT_RCVCTL	create_table_gnt.sql	電文保証の管理および電文の保持に利用します。

(2)　ストアドプロシージャの生成

SQL ファイルを実行してストアドプロシージャを生成します。

```
> cd /opt/diosa_xtp/sql/{環境の文字コード}
> sqlplus Oracle ログイン情報 @COM/create_com.sql
```

2.4　監視設計

2.4.1　プロセス監視

ほとんどの DIOSA/XTP の常駐プロセスは、DIOSA/XTP のプロセス監視機能によって監視されています。同一の常駐プロセスが一定時間内に一定回数異常終了した場合、監視機能の常駐プロセス自体が停止するため、外部のプロセス監視ツールで DIOSA/XTP の常駐プロセスを監視する場合は、監視機能の常駐プロセスとメッセージ出力をおこなうための常駐プロセスのみを監視対象としてください。

デーモンプロセスは起動パラメータとして、「-N 論理ノード名」を指定されているため、同一マシン上で複数の論理ノードを起動している場合でも識別することが可能です。

プロセス名	説明
dimsgd	メッセージ出力をおこなう
didamd	DIOSA/XTP の常駐プロセス監視をおこなう

表 2-2 外部からの監視が必要な常駐プロセス

DIOSA/XTP のプロセス監視機能のパラメータ設定は環境変数でおこないます。(詳細は環境定義リファレンスを参照してください。)

環境変数名	説明
DIOSA_DAM_DOWNCNT	再起動回数
DIOSA_DAM_MONINVL	監視間隔
DIOSA_DAM_RESINVL	再起動監視時間
DIOSA_DAM_RTYCNT	再起動失敗時のリトライ回数
DIOSA_DAM_RTYINVL	再起動失敗時のリトライ間隔

表 2-3 DIOSA/XTP プロセス監視条件設定用環境変数

また、一部の常駐プロセスについては、各機能の独自の監視機能によって監視されているため、上記環境変数とは異なる条件で監視を実施しています。

機能名	監視条件設定箇所
データベース管理機能	環境定義 DBCTRL 節
IM サーバ所在管理機能	環境定義 IMENV 節
ディレード機能	環境定義 DELAYED 節

表 2-4 DIOSA/XTP プロセス監視機能以外による常駐プロセス監視

DIOSA/XTP で動作するすべての常駐プロセスの監視方法、動作ノード、起動停止方法については、「付録 B プロセス一覧」を参照してください。

2.4.2　　ログ管理

以下のファイルは定期的なメンテナンス作業が必要です。

(1)　　メッセージログファイル

メッセージログファイルは複数のファイルをサイクリックに使用しますが、古いファイルは順次上書きされます。メッセージを履歴として全て残したい場合は、ファイルスワップ時に実行するコマンドを環境変数(DIOSA_MSG_CMD)に指定し、ファイルをコピーする等の運用が必要です。

(2)　　常駐プロセスからの標準出力

常駐プロセスから標準出力や標準エラー出力に出力されたメッセージは、{DIOSA_TMP}/{論理ノード名}/log/{常駐プロセス名}.[std|err]というファイルに格納されます。

本ファイルはファイルローテーション等の制御は実施していないため、定期的にファイルをコピーして初期化する等の運用が必要です。

第3章　システムの運用

3.1　定義生成

3.1.1　DIOSA 環境定義

作成した環境定義は、DIOSA の起動前にオブジェクトの生成を行う必要があります。

生成したオブジェクトファイルの内容にしたがい動作します。

(a)　環境定義オブジェクトファイル作成

はじめて環境定義オブジェクトを生成するときに以下のコマンドを実行します。

```
diirmadd -E 環境定義ファイル名 ...
```

(b)　環境定義オブジェクトファイル更新

すでに環境定義オブジェクトファイルを生成しているときに以下のコマンドで更新します。

```
diirmrep -E 環境定義ファイル名 ...
```

3.1.2　TPBASE 環境定義

作成した環境定義は、変換操作は必要ありません。

環境定義の詳細については、TPBASE のマニュアルを参照してください。

3.1.3　TAM 環境定義

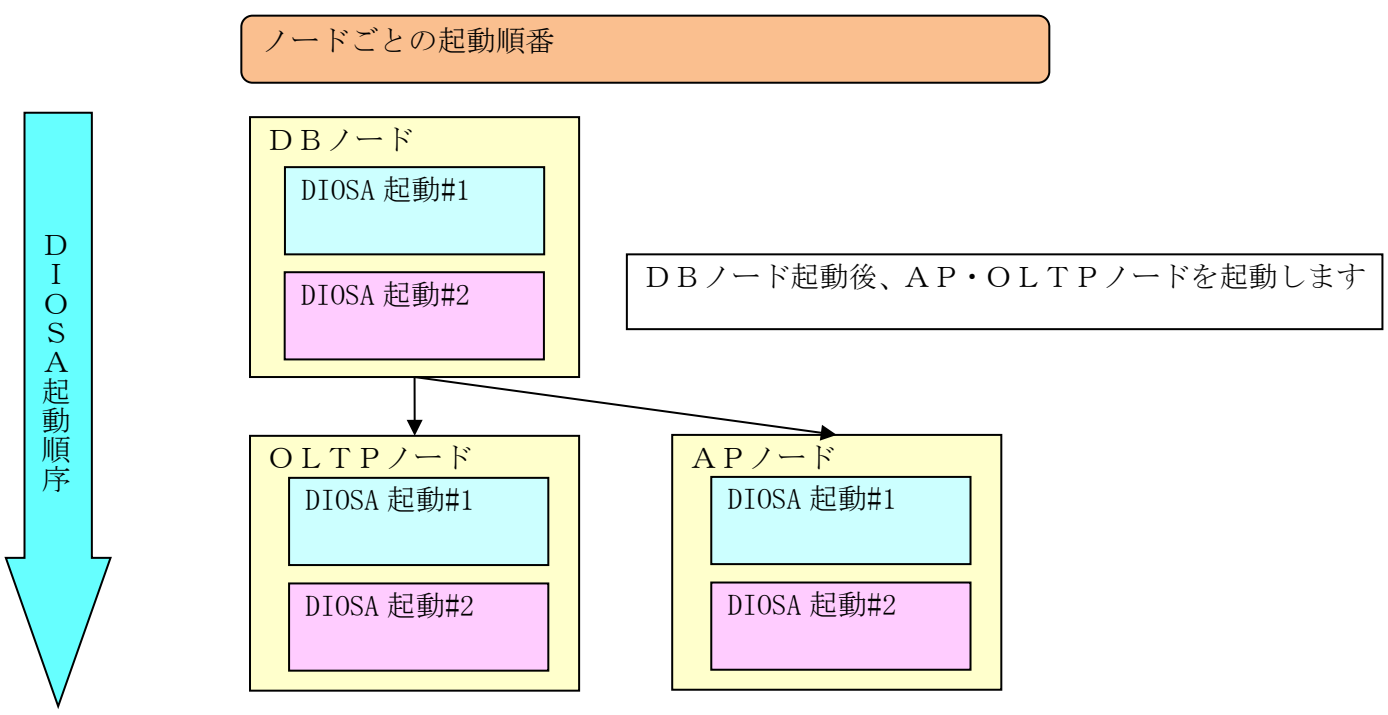
作成した環境定義は、tamcfgmaint コマンドや tamcreate を使って変換や反映する操作が必要となります。

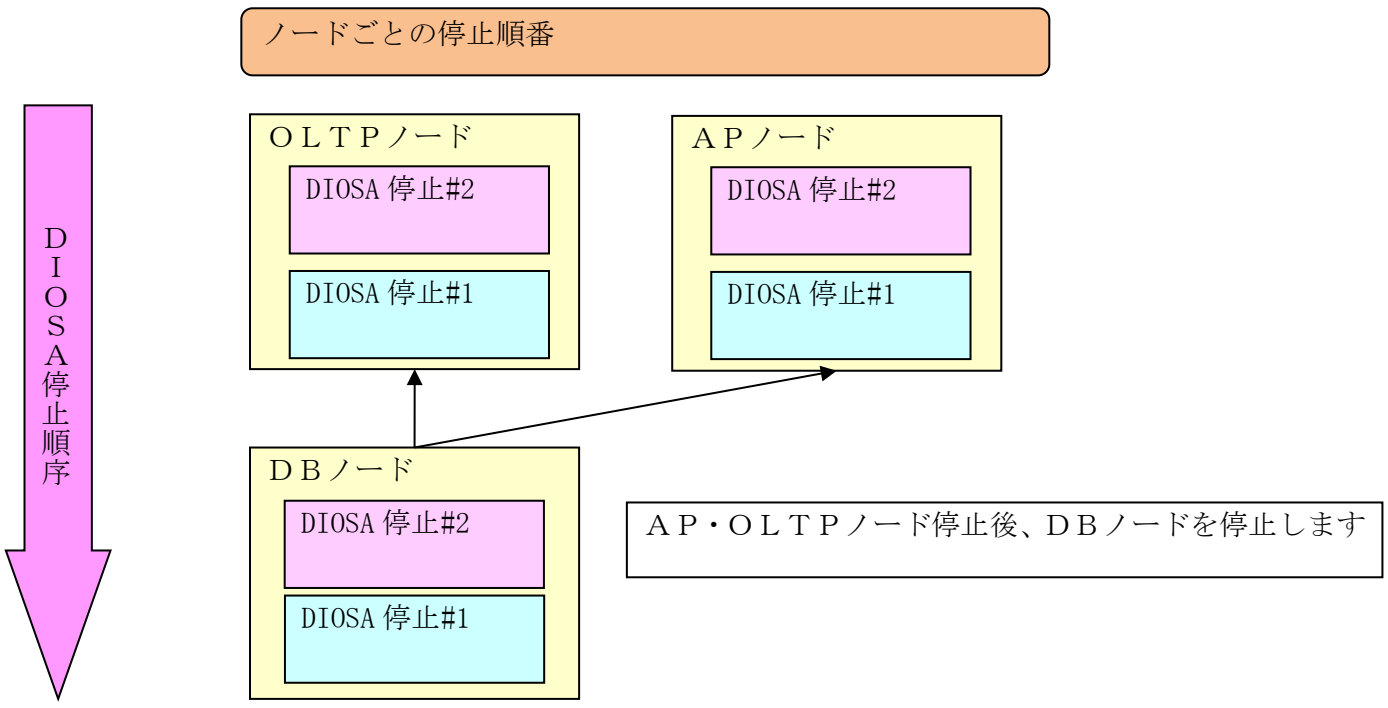
環境定義の詳細については、TAM のマニュアルを参照してください。

3.2　起動・停止

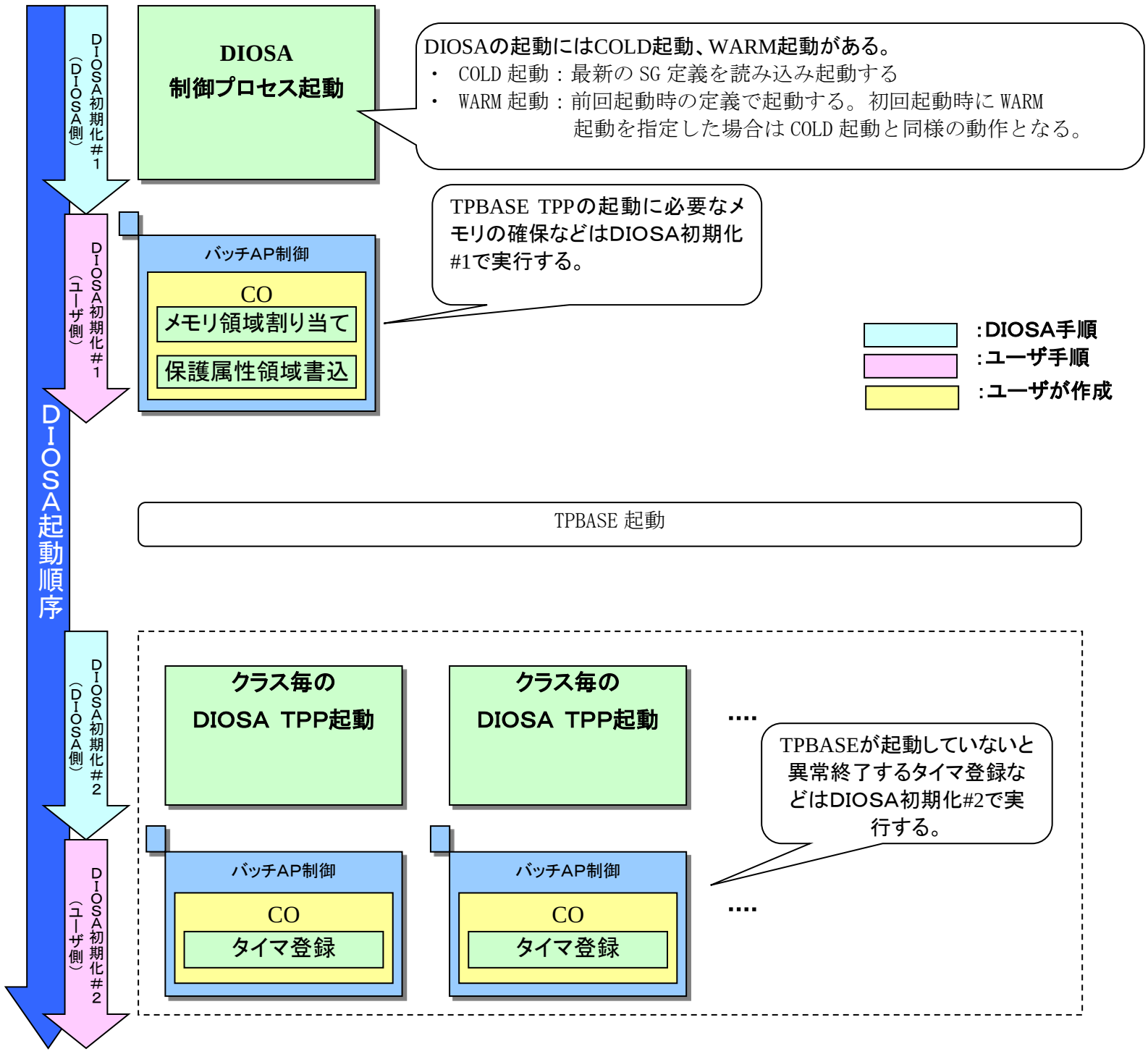
3.2.1　ノードごとの起動・停止順番

以下の図に DIOSA の起動停止順を記述します。





3.2.2　　DIOSA の起動・停止フロー



(1)　　DIOSA 初期化＃1

状態：TPBASE 起動前

説明：CO などの TPBASE TPP 稼動前にできる必要な初期化作業を行う。

たとえば、各 CO で共通に使用する共有メモリの確保とデータの設定等に用いる。

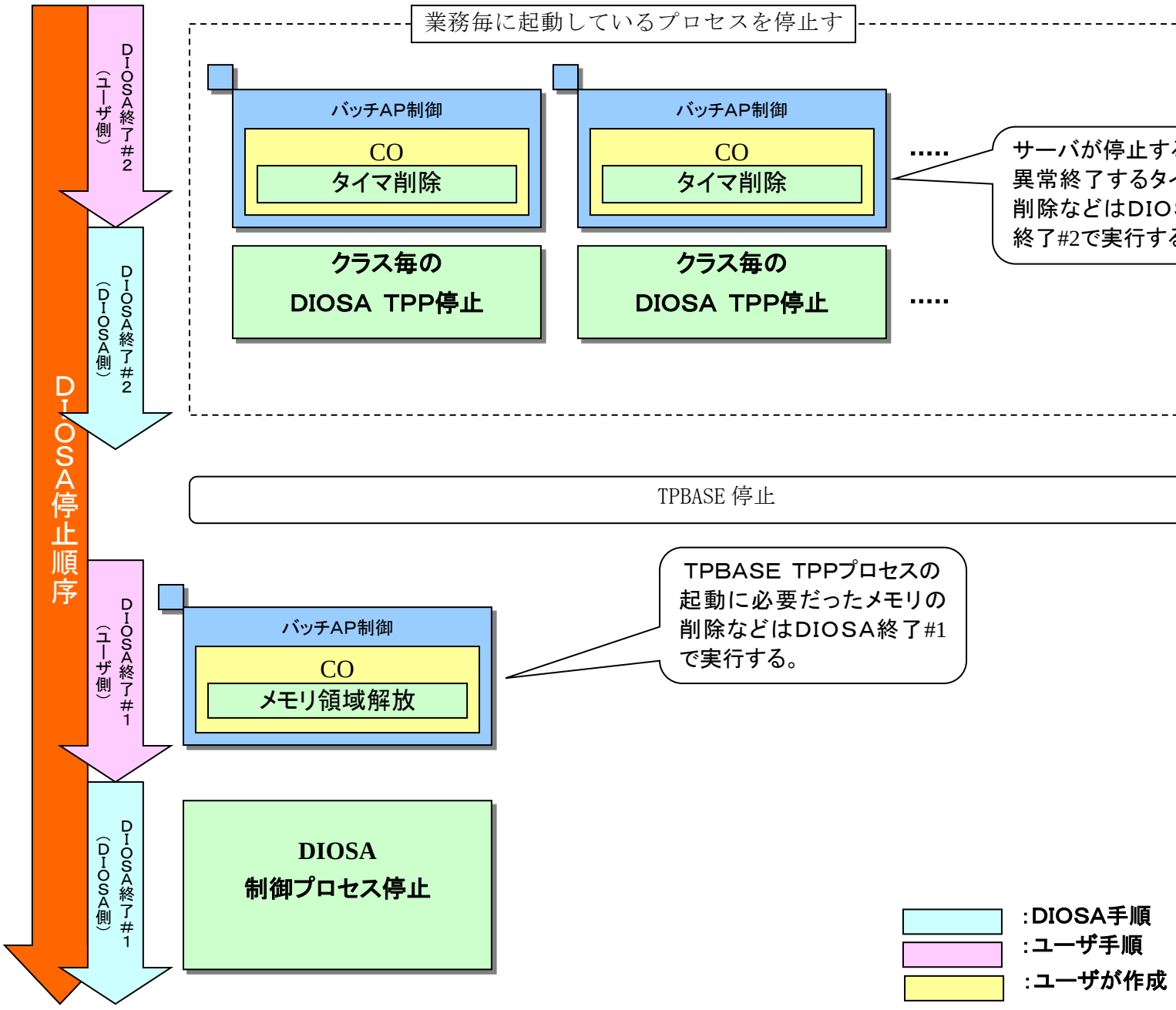
(2)　　DIOSA 初期化＃2

状態：TPBASE 起動後

説明：CO などの TPBASE TPP が稼動していないとできない必要な初期化作業、業務毎の初期化作業を行う。

たとえば、CO を呼び出すタイマの登録等に用いる。

なお、TPBASE は起動するまで時間がかかる場合があるため、必要に応じて待ち合わせを行う。



(3) **DIOSA 終了#2**

状態：TPBASE 停止前

説明：CO などの TPBASE TPP が稼動していないとできない必要な後始末作業、業務毎の後始末作業を行う。

たとえば、CO を呼び出すタイマの削除等に用いる。

(4) **DIOSA 終了#1**

状態：TPBASE 停止後

説明：CO などの TPBASE TPP 停止後に必要な後始末作業を行う。

たとえば、各 CO で使用していた共有メモリの解放等に用いる。

3.2.3 DIOSA 起動コマンド一覧

以降の表にノード毎の DIOSA 起動コマンド一覧を記します。

(1) DB ノード

	コマンド名	オプション	実行有無	概要
DIOSA#1	didbginit	didbginit	必須	デバッグトレース機能の初期化処理、ログ情報ファイル出力デーモン(デーモン名: didbglogd)及び障害情報ファイル出力デーモン(デーモン名: didbgerrd)を起動します ただし、次の環境変数を設定した場合は各デーモンを起動しません DIOSA_ DBGLOGPAGE =1 (または 省略)の場合、ログ情報ファイル出力デーモンは起動しません。 DIOSA_DBGERRPAGE =1 (または 省略)の場合、障害情報ファイル出力デーモンは起動しません。 環境変数の詳細については環境定義リファレンスを参照してください
	dimsgdctrl	dimsgdctrl -b	必須	メッセージ出力デーモンの起動、停止を行います
	distart_com	distart_com [-c|-w]	必須	DIOSA を起動するために必要な初期化処理と、各サブコンポーネントを初期化する API を実行します
	didlrinit	didlrinit [-c|-w]	必須	アプリケーション動的置換機能の初期化をおこないます
	didbglogswap	didbglogswap -f [-P 絶対パス]	選択	ログ情報保存領域を強制スワップし、ログ情報出力ファイルへ出力後、ユーザ指定によりログ情報出力ファイルに対しても強制スワップを行います。ログ情報ファイル出力デーモンがデーモン死活監視に登録されていない場合は、登録要求を行います 環境変数 DIOSA_DBGLOGPAGE =1 (または 省略)の場合は本コマンドの実行は不要です
	didbgerrswap	didbgerrswap -f [-P 絶対パス]	選択	障害情報保存領域を強制スワップし、障害情報出力ファイルへ出力後、ユーザ指定により障害情報出力ファイルに対しても強制スワップを行います。障害情報ファイル出力デーモンがデーモン死活監視に登録されていない場合は、登録要求を行います 環境変数 DIOSA_DBGERRPAGE =1 (または 省略)の場合は本コマンドの実行は不要です
	dibcmctrl	dibcmctrl –b	必須	閉塞管理デーモンを起動または停止します
	dicddctrl	dicddctrl –b	必須	コマンド配信コマンドを実行するデーモンを起動します
	dincmdbinit	dincmdbinit [-c|-w]	必須	DB 管理機能の共有メモリを作成し、環境定義から情報をセットする処理を行います
	dincmdbhchkctrlinit	dincmdbhchkctrlinit	必須	DB のヘルスチェックデーモンの起動を行います

※各コマンドの詳細は、コマンドリファレンスを参照してください。

起動確認は、distatref コマンドと各機能の状態照会コマンドを使って行います。

なお、distatref コマンドで「Running」と表示された機能は、機能の起動コマンドが実行された状態であることを表していますが、各機能が現在正常に動作しているかどうかは、各機能の照会コマンド等で確認する必要があります。

状態確認の実行例

```
# distatref -–v
NODE STARTUP STATUS    : Running
LNODE TYPE             : DB
LNODE NAME             : db1

<INFORMATION ON EACH FUNCTION>

Start-End Function                    : Running
Dynamic Library Replace Function      : Running
Daemon Alive Monitoring               : Running
Blockade Management Function          : Running
Command Delivery Function             : Running
Data Base Monitoring                  : Running
Delayed Transfer                      : Running
```

(2)　　APノード

	コマンド名	オプション	実行有無	概要
DIOSA#1	didbginit	didbginit	必須	デバッグトレース機能の初期化処理、ログ情報ファイル出力デーモン(デーモン名: didbglogd)及び障害情報ファイル出力デーモン(デーモン名: didbgerrd)を起動します ただし、次の環境変数を設定した場合は各デーモンを起動しません DIOSA_ DBGLOGPAGE =1 (または 省略)の場合、ログ情報ファイル出力デーモンは起動しません。 DIOSA_DBGERRPAGE =1 (または 省略)の場合、障害情報ファイル出力デーモンは起動しません。 環境変数の詳細については環境定義リファレンスを参照してください
	dimsgdctrl	dimsgdctrl -b	必須	メッセージ出力デーモンの起動、停止を行います
	distart_com	distart_com [-c|-w]	必須	DIOSA を起動するために必要な初期化処理と､各サブコンポーネントを初期化する API を実行します
	diapptrcinit	diapptrcinit	選択	アプリケーショントレース出力デーモンを起動します
	didlrinit	didlrinit [-c|-w]	必須	アプリケーション動的置換機能の初期化をおこないます
	didbglogswap	didbglogswap -f [-P 絶対パス]	選択	ログ情報保存領域を強制スワップし、ログ情報出力ファイルへ出力後、ユーザ指定によりログ情報出力ファイルに対しても強制スワップを行います。ログ情報ファイル出力デーモンがデーモン死活監視に登録されていない場合は、登録要求を行います 環境変数 DIOSA_DBGLOGPAGE =1 (または 省略)の場合は本コマンドの実行は不要です
	didbgerrswap	didbgerrswap -f [-P 絶対パス]	選択	障害情報保存領域を強制スワップし、障害情報出力ファイルへ出力後、ユーザ指定により障害情報出力ファイルに対しても強制スワップを行います。障害情報ファイル出力デーモンがデーモン死活監視に登録されていない場合は、登録要求を行います 環境変数 DIOSA_DBGERRPAGE =1 (または 省略)の場合は本コマンドの実行は不要です
	dibcmctrl	dibcmctrl –b	必須	閉塞管理デーモンを起動または停止します
	dicddctrl	dicddctrl –b	必須	コマンド配信コマンドを実行するデーモンを起動します
	dincmdbinit	dincmdbinit [-c|-w]	選択	DB 管理機能の共有メモリを作成し、環境定義から情報をセットする処理を行います ※Oracle DB を使用しない場合は不要です。
	dincmdbhchkctrlinit	dincmdbhchkctrlinit	選択	DB のヘルスチェックデーモンの起動を行います ※Oracle DB を使用しない場合は不要です。
	dincmpathinit	dincmpathinit	必須	パス制御機能の初期化を行います。
	diopsctrl	diopsctrl -b	選択	CO 制御・バッチアプリケーション制御からの稼動情報を受信してファイルへ出力する稼動統計デーモンの起動、停止を行います
	dietgctrl	dietgctrl -b [-t 監視間隔時間]	選択	経過時間を監視するデーモンの起動、停止をおこないます
	dicocinit	dicocinit [-c|-w]	選択	CO 制御の初期化処理を行います
#2	dincmpathdmnctrl	dincmpathdmnctrl -b	必須	パス制御機能のデーモンを起動します。
	ditmcdmn	ditmcdmn [-t インターバル時間]	必須	登録されたタイマを監視するデーモンを起動します
	ditmcsgset	ditmcsgset	選択	SG で指定されたタイマを登録します

※各コマンドの詳細は、コマンドリファレンスを参照してください。

起動確認は、distatref コマンドと各機能の状態照会コマンドを使って行います。

なお、distatref コマンドで「Running」と表示された機能は、機能の起動コマンドが実行された状態であることを表していますが、各機能が現在正常に動作しているかどうかは、各機能の照会コマンド等で確認する必要があります。

状態確認の実行例

```
# distatref -v
NODE STARTUP STATUS    : Running
LNODE TYPE             : AP
LNODE NAME             : ap1

<INFORMATION ON EACH FUNCTION>

Start-End Function                      : Running
Dynamic Library Replace Function        : Running
Daemon Alive Monitoring                 : Running
Timer Control Function                  : Running
Elapsed Time Guard Function             : Running
Blockade Management Function            : Running
Command Delivery Function               : Running
Data Base Monitoring                    : Running
Inter-Node Communication Path Management: Running
Message Flow Control                    : Stop
Message Guarantee                       : Running
In-Memory Server Information Control    : Running
Operation Statistics Function           : Running
Control-Object Controller               : Running
Delayed Transfer                        : Running
Delayed Transfer / Sender               : Running
Delayed Transfer / Receiver             : Running
Delayed Transfer / Log Reader           : Running
Online Maintenance                      : Running
Database Access Control                 : Running
```

(3) OLTP ノード

	コマンド名	オプション	実行有無	概要
DIOSA#1	didbginit	didbginit	必須	デバッグトレース機能の初期化処理、ログ情報ファイル出力デーモン(デーモン名: didbglogd)及び障害情報ファイル出力デーモン(デーモン名: didbgerrd)を起動します ただし、次の環境変数を設定した場合は各デーモンを起動しません DIOSA_ DBGLOGPAGE =1 (または 省略)の場合、ログ情報ファイル出力デーモンは起動しません。 DIOSA_DBGERRPAGE =1 (または 省略)の場合、障害情報ファイル出力デーモンは起動しません。 環境変数の詳細については環境定義リファレンスを参照してください
	dimsgdctrl	dimsgdctrl -b	必須	メッセージ出力デーモンの起動、停止を行います
	distart_com	distart_com [-c|-w]	必須	DIOSA を起動するために必要な初期化処理と､各サブコンポーネントを初期化する API を実行します
	diapptrcinit	diapptrcinit	選択	アプリケーショントレース出力デーモンを起動します
	didlrinit	didlrinit [-c|-w]	必須	アプリケーション動的置換機能の初期化をおこないます
	didbglogswap	didbglogswap -f [-P 絶対パス]	選択	ログ情報保存領域を強制スワップし、ログ情報出力ファイルへ出力後、ユーザ指定によりログ情報出力ファイルに対しても強制スワップを行います。ログ情報ファイル出力デーモンがデーモン死活監視に登録されていない場合は、登録要求を行います 環境変数 DIOSA_DBGLOGPAGE =1 (または 省略)の場合は本コマンドの実行は不要です
	didbgerrswap	didbgerrswap -f [-P 絶対パス]	選択	障害情報保存領域を強制スワップし、障害情報出力ファイルへ出力後、ユーザ指定により障害情報出力ファイルに対しても強制スワップを行います。障害情報ファイル出力デーモンがデーモン死活監視に登録されていない場合は、登録要求を行います 環境変数 DIOSA_DBGERRPAGE =1 (または 省略)の場合は本コマンドの実行は不要です
	dibcmctrl	dibcmctrl –b	必須	閉塞管理デーモンを起動または停止します
	dicddctrl	dicddctrl –b	必須	コマンド配信コマンドを実行するデーモンを起動します
	dincmdbinit	dincmdbinit [-c|-w]	選択	DB 管理機能の共有メモリを作成し、環境定義から情報をセットする処理を行います ※Oracle DB を使用しない場合は不要です。
	dincmdbhchkctrlinit	dincmdbhchkctrlinit	選択	DB のヘルスチェックデーモンの起動を行います。 ※Oracle DB を使用しない場合は不要です。
	dincmpathinit	dincmpathinit	必須	パス制御機能の初期化を行います。
	diopsctrl	diopsctrl -b	選択	CO 制御・バッチアプリケーション制御からの稼動情報を受信してファイルへ出力する稼動統計デーモンの起動、停止を行います
	dietgctrl	dietgctrl -b [-t 監視間隔時間]	選択	経過時間を監視するデーモンの起動、停止をおこないます
	dicocinit	dicocinit [-c|-w]	選択	CO 制御の初期化処理を行います
#2	dincmpathdmnctrl	dincmpathdmnctrl -b	必須	パス制御機能のデーモンを起動します
	ditmcdmn	ditmcdmn [-t インターバル時間]	必須	登録されたタイマを監視するデーモンを起動します
	ditmcsgset	ditmcsgset	選択	SG で指定されたタイマを登録します

※各コマンドの詳細は、コマンドリファレンスを参照してください。

起動確認は、distatref コマンドと各機能の状態照会コマンドを使って行います。

なお、distatref コマンドで「Running」と表示された機能は、機能の起動コマンドが実行された状態であることを表していますが、各機能が現在正常に動作しているかどうかは、各機能の照会コマンド等で確認する必要があります。

状態確認の実行例

```
# distatref -—v
NODE STARTUP STATUS    : Running
LNODE TYPE             : OLTP
LNODE NAME             : oltp1

<INFORMATION ON EACH FUNCTION>

Start-End Function                      : Running
Dynamic Library Replace Function        : Running
Daemon Alive Monitoring                 : Running
Timer Control Function                  : Running
Elapsed Time Guard Function             : Running
Blockade Management Function            : Running
Command Delivery Function               : Running
Data Base Monitoring                    : Running
T-Path Management                       : Running
Message Flow Control                    : Stop
In-Memory Server Information Control    : Running
Operation Statistics Function           : Running
Control-Object Controller               : Running
Delayed Transfer                        : Running
Delayed Transfer / Sender               : Running
Delayed Transfer / Receiver             : Running
Delayed Transfer / Log Reader           : Running
Online Maintenance                      : Stop
Database Access Control                 : Stop
```

3.2.4　　DIOSA 停止コマンド一覧

以降の表にノード毎の DIOSA 停止コマンド一覧を記します。

(1)　　DB ノード

	コマンド名	オプション	実行有無	概要
DIOSA#1	dincmdbhchkctrlterm	dincmdbhchkctrlterm	必須	DB のヘルスチェックデーモンの停止を行います
	dincmdbterm	dincmdbterm	必須	DB 管理機能の共有メモリを削除する処理を行います
	dicddctrl	dicddctrl –e [-f force]	必須	コマンド配信コマンドを実行するデーモンを起動します
	dibcmctrl	dibcmctrl -e	必須	閉塞管理デーモンを起動または停止します
	distop	distop	必須	DIOSA を停止します
	dimsgdctrl	dimsgdctrl -e	必須	メッセージ出力デーモンの起動、停止を行います
	didbgterm	didbgterm	必須	デバッグトレース機能で使用したデバッグ共有メモリを解放します。この時、障害情報ファイル出力デーモンも終了します

※各コマンドの詳細は、コマンドリファレンスを参照してください。

(2)　　AP ノード

	コマンド名	オプション	実行有無	概要
#2	ditmcdmnstop	ditmcdmnstop	必須	タイマデーモンを停止します
	dincmpathdmnctrl	dincmpathdmnctrl -e	必須	パス制御機能のデーモンを停止します。
DIOSA#1	dicocterm	dicocterm	選択	CO 制御の終了処理を行います
	dietgctrl	dietgctrl -e	選択	経過時間を監視するデーモンの起動、停止をおこないます
	diopsctrl	diopsctrl –e	選択	CO 制御・バッチアプリケーション制御からの稼動情報を受信してファイルへ出力する稼動統計デーモンの起動、停止を行います
	dincmpathterm	dincmpathterm	必須	パス制御機能の共有メモリを削除する処理を行います
	dincmdbhchkctrlterm	dincmdbhchkctrlterm	選択	DB のヘルスチェックデーモンの停止を行います
	dincmdbterm	dincmdbterm	選択	DB 管理機能の共有メモリを削除する処理を行います
	dicddctrl	dicddctrl –e [-f force]	必須	コマンド配信コマンドを実行するデーモンを起動します
	dibcmctrl	dibcmctrl -e	必須	閉塞管理デーモンを起動または停止します
	diapptrcterm	diapptrcterm	選択	アプリケーショントレース出力デーモンを停止します。
	distop	distop	必須	DIOSA を停止します
	dimsgdctrl	dimsgdctrl -e	必須	メッセージ出力デーモンの起動、停止を行います
	didbgterm	didbgterm	必須	デバッグトレース機能で使用したデバッグ共有メモリを解放します。この時、障害情報ファイル出力デーモンも終了します

※各コマンドの詳細は、コマンドリファレンスを参照してください。

(3)　　OLTP ノード

	コマンド名	オプション	実行有無	概要
#2	ditmcdmnstop	ditmcdmnstop	必須	タイマデーモンを停止します
	dincmpathdmnctrl	dincmpathdmnctrl -e	必須	パス制御機能のデーモンを停止します。
DIOSA#1	dicocterm	dicocterm	選択	CO 制御の終了処理を行います
	dietgctrl	dietgctrl -e	選択	経過時間を監視するデーモンの起動、停止をおこないます
	diopsctrl	diopsctrl –e	選択	CO 制御・バッチアプリケーション制御からの稼動情報を受信してファイルへ出力する稼動統計デーモンの起動、停止を行います
	dincmpathterm	dincmpathterm	必須	パス制御機能の共有メモリを削除する処理を行います
	dincmdbhchkctrlterm	dincmdbhchkctrlterm	選択	DB のヘルスチェックデーモンの停止を行います
	dincmdbterm	dincmdbterm	選択	DB 管理機能の共有メモリを削除する処理を行います
	dicddctrl	dicddctrl –e [-f force]	必須	コマンド配信コマンドを実行するデーモンを起動します
	dibcmctrl	dibcmctrl -e	必須	閉塞管理デーモンを起動または停止します
	diapptrcterm	diapptrcterm	選択	アプリケーショントレース出力デーモンを停止します。
	distop	distop	必須	DIOSA を停止します
	dimsgdctrl	dimsgdctrl -e	必須	メッセージ出力デーモンの起動、停止を行います
	didbgterm	didbgterm	必須	デバッグトレース機能で使用したデバッグ共有メモリを解放します。この時、障害情報ファイル出力デーモンも終了します

※各コマンドの詳細は、コマンドリファレンスを参照してください。

3.3　環境変更

3.3.1　サブコン固有変更

(1)　アプリケーション動的置換定義変更

プロセスの追加や、リンクするライブラリ/関数の追加、削除等を実施する場合、以下の手順で置換をおこないます。

(a)　APLIB 節更新

変更後の定義に合わせて、APLIB 節の内容を修正します。

(b)　環境定義オブジェクトファイル作成

環境定義オブジェクトファイルを更新します。

```
diirmrep -E 環境定義ファイル名
```

(c)　アプリケーション動的置換コマンド実行

アプリケーション動的置換コマンドを実行し、変更後の定義を有効にします。

```
didlrchg
```

コマンド実行以降に開始するトランザクション、起動するプロセスについては、全て置換後の定義に従ってアプリケーションの呼び出しをおこなうようになります。

3.4　障害時対応

3.4.1　ノード障害

DIOSA/XTP が動作するノードが異常終了した場合の復旧方法について説明します。

(1)　AP ノード障害からの復旧

AP ノードが意図せず停止した場合、DIOSA は障害状態を認識して該当ノードへの通信を行わなくなります。また、メモリキャッシュやデータストアなどの製品が動作していた場合も同様に他のノードで処理が継続されます。

出力されているメッセージ等を元に障害の原因を特定、原因を取り除いた後、AP ノードを再起動します。

再起動後は、必要に応じて論理ノード間パスを構成する端末を再接続してください。

(2)　OLTP ノード障害からの復旧

OLTP ノードが意図せず停止した場合、DIOSA は障害状態を認識して該当ノードへの通信を行わなくなります。また、メモリキャッシュやデータストアなどの製品が動作していた場合も同様に他のノードで処理が継続されます。

出力されているメッセージ等を元に障害の原因を特定、原因を取り除いた後、OLTP ノードを再起動します。

再起動後は、必要に応じて論理ノード間パスを構成する端末を再接続してください。

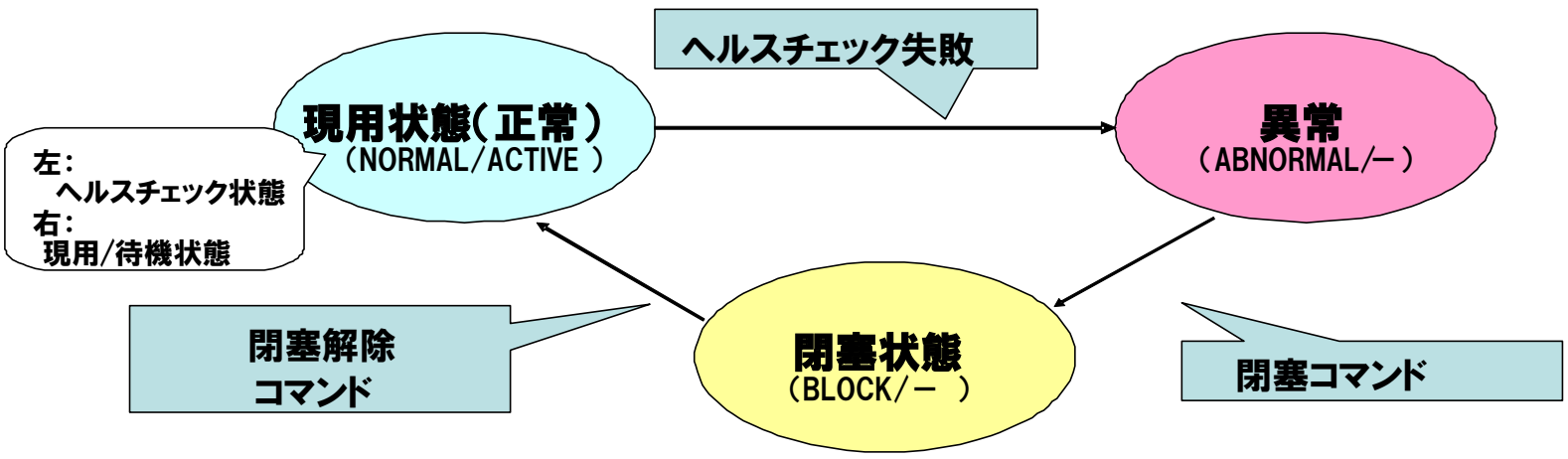
OLTP ノードが再起動した後に、TAM マスタを復旧したノードに移動する場合は、AP ノードで以下のコマンドを実行して、復旧したノードとの論理ノード間パスが活性である(STATUS が NORMAL となっている)ことを確認してください。論理ノード間パスの活性前に TAM マスタを移動した場合、AP ノードからのデータ通知が失敗する場合があります。

```
$ dinodepathref
+====== T-PATH STATUS DISPLAY ======+  2017/01/23 04:56:07
LSNAME    = LSYS1
LNODENAME = ap1
SOURCE___ DESTINATION____________________
TPM______ TYPE LNODENAME______ TPM______ STATUS__
ap1tpm    OLTP oltp1          oltp1tpm  OPEN
          OLTP oltp2          oltp2tpm  OPEN
+====== END OF DISPLAY  ======+
```

(3)　DB ノード障害からの復旧

出力されているメッセージ等を元に障害の原因を特定し、原因を取り除いた後、障害 DB ノードを再起動します。DB ノード再起動後、閉塞状態参照コマンド(dibcmref)を実行し、閉塞状態の確認を行います。当該 DB ノードが閉塞状態に移行していた場合は、何れかのノードで閉塞状態変更コマンド(dibcmupd)を実行し、当該 DB ノードの閉塞を解除します。

なお、環境変数 DIOSA_DBINTEGRATE_AUTO により閉塞を抑止していた場合は、障害 DB ノードが閉塞状態に移行することがないため、障害 DB ノードを再起動後、AP/OLTP ノードからのヘルスチェックが成功した時点で稼働系として組み込まれます。DB ノードの状態管理については下図を参照してください。



現用状態(正常)	通常にDBサーバが利用されている(利用可能)な状態です。
閉塞状態	閉塞状態の場合は、当該DBサーバのヘルスチェックは実行されません。 閉塞されたDBノードの利用を開始する場合は、閉塞解除コマンドを実行します。
異常	ヘルスチェックが失敗すると、異常状態になりますが、自動的に閉塞コマンドが実行され、閉塞状態に移行します。ただし、閉塞を抑止している場合には、ヘルスチェックが成功すると自動的に、現用状態(正常)に移行します。

図 3-1 DBノードの状態管理

3.4.2　　DIOSA 障害

通常の停止手順で DIOSA/XTP が停止できず強制停止させたい場合、起動中の常駐プロセスの停止、作成した IPC 資源の削除が必要です。

以下のコマンドを実行することにより DIOSA/XTP のプロセスの停止、IPC 資源の削除を行うことができます。

distop -f

コマンド実行後は、下記の常駐プロセスの停止及び、IPC 資源が削除されていることを確認してください。

(1)　常駐プロセスの停止

DIOSA の常駐プロセスは、パラメータに「-N 論理ノード名」を指定して起動されています。

「付録 B.1 常駐プロセス一覧」に記載されているプロセス名と論理ノード名を ps コマンドで検索し起動されていないことを確認してください。

(2)　IPC 資源の削除

DIOSA で作成する IPC 資源は、ID の上位 4 桁が以下の定義で指定した値で作成されます。

・環境変数 DIOSA_IPCKEY で指定した値

ipcs コマンドで対象の IPC 資源を検索し、削除されていることを確認してください。

3.4.3　　TAM 障害

TAM 障害時の復旧については、DIOSA/XTP メモリキャッシュ利用の手引を参照してください。

3.4.4　　Oracle データベース障害

DIOSA/XTP で使用しているデータベース表や OracleDB に障害が発生した場合の復旧方法と処理の再開方法について説明します。

(1)　　データベース表障害

DIOSA/XTP では、データベース表に様々な制御情報を保有しています。表に障害が発生した場合は、OracleDB の機能を利用してデータリストアおよびロールフォワード/ロールバックを行うことで障害直前の状態に復旧します。表の復旧により DIOSA/XTP は処理実行可能な状態となるため、障害により停止していた機能を再開することでサービスを再開します。

(2)　　OracleDB 障害

OracleDB が障害となった場合は、DIOSA/XTP では全 DB ノード障害となるため、下記の手順によりデータベースを復旧してください。

1. DB ノードを停止する。
2. OracleDB の障害原因を調査し、取り除く。
3. OracleDB を起動する。
4. DB ノードを起動する。
5. DB ノードが閉塞状態に移行している場合は閉塞を解除する。(閉塞解除手順については、「3.4.1(3) DB ノード障害からの復旧」を参照してください)
6. DB ノード障害により停止したデーモンを再起動する。(詳細は、DIOSA/XTP データストア 利用の手引に記載されている「データベース障害」の項を参照してください)

付録A　リソース一覧

A.1　共有メモリ

DIOSA/XTP で使用する共有メモリ一覧です。

キーの上位 4 桁は、環境変数(DIOSA_IPCKEY)に定義した値です。

また、「-」となっているキーの最下位 1 桁は、0 か 1 のいずれかを使用しています。

A.1.1　アプリケーション実行制御、通信制御

キー	サイズ(※)	備考
0x----0000	備考参照	DIOSA/XTP 起動時に作成 [サイズ見積] 4284 + 8 ×${環境変数 DIOSA_PRIVATEIPC_MAX の値} DIOSA_PRIVATEIPC_MAX 未定義時は以下となる 4284 + 8 ×10000
0x----0200	備考参照	DIOSA/XTP 起動時に作成 [サイズ見積] 32 + 32 × DIOSAMAP 節-LOGSYSTEM 項数 + 1576 × DIOSAMAP 節-LNODE 項数 + 32 + 416 × SYSMAP 節-LOGSYSTEM 項数 + 144 × SYSMAP 節-ROUTE 項数 + 8272 × SYSMAP 節-ACCESSPOINT 項数 + 328 × SYSMAP 節-PROTOCOL 項数
0x----0d00	備考参照	DIOSA/XTP 起動時に作成 [サイズ見積] 40＋184×${環境変数 DIOSA_ETGMAXENTRY の値}
0x----1301	1KB	稼動統計機能起動時に作成
0x----1302	1KB	稼動統計機能起動時に作成
0x----1303	1KB	稼動統計機能起動時に作成
0x----1304	1KB	稼動統計機能起動時に作成
0x----2810	1KB	DIOSA/XTP 起動時に作成
0x----282-	備考参照	DIOSA/XTP 起動時に作成 キーの最下位 1 桁は、0 か 1 のいずれかを使用する [サイズ見積] 704＋32×(論理システム数×5＋CMDSEND 節-SRVGRP 項数) ＋296×CMDSEND 節-ENV 項数＋32×CMDSEND 節-LNODECMD 項数 ＋32×CMDSEND 節-SRVGRP 項数＋16×CMDSEND 節-LNODESRV 項数 ＋88×CMDSEND 節-CMDRT 項数＋8×CMDSEND 節-CMDPERM 項数 ＋48×CMDSEND 節-CMDGRP 項数＋64×CMDSEND 節-CMDTEXT 項数 ＋56×CMDSEND 節-EXPERM 項数＋32×CMDSEND 節-CMD 項数
0x----0a00	備考参照	DIOSA/XTP 起動時に作成 [サイズ見積] 192＋(336＋112＋${環境変数 DIOSA_TMCLENMAX の値}) ×((${環境変数 DIOSA_TMCTBLMAX の値}×1.1 を切り上げ) ＋(${環境変数 DIOSA_TMCDITBLMAX の値}×1.1 を切り上げ)＋2)
0x----0a01	備考参照	タイマ制御機能起動時に作成 [サイズ見積] 16＋120×(${環境変数 DIOSA_TMCPROC の値} ＋${環境変数 DIOSA_TMCDIPROC の値})

キー	サイズ(※)	備考
0x----fe00	備考参照	アプリケーショントレース機能起動時に作成 [サイズ見積] 2048＋(16×${環境変数 DIOSA_APPTRCMIDNUM の値}) ＋(${環境変数 DIOSA_APPTRCMPAGE_1 の値} ×${環境変数 DIOSA_APPTRCMSIZE_1 の値}) ＋(${環境変数 DIOSA_APPTRCMPAGE_2 の値} ×${環境変数 DIOSA_APPTRCMSIZE_2 の値}) ＋(${環境変数 DIOSA_APPTRCMPAGE_3 の値} ×${環境変数 DIOSA_APPTRCMSIZE_3 の値}) 注意)加算する DIOSA_APPTRCMPAGE_N(1～3)×DIOSA_APPTRCMSIZE_N(1～3)は ${環境変数 DIOSA_APPTRCMIDNUM の値}数分とする。 ${環境変数 DIOSA_APPTRCMPAGE の値}は DIOSA_APPTRCMPAGE_N(1～3)のデフォルト値とする ${環境変数 DIOSA_APPTRCMSIZE の値}は DIOSA_APPTRCMSIZE_N(1～3)のデフォルト値とする
0x----ffff	備考参照	デバッグトレース機能起動時に作成 [サイズ見積] 3296 ＋(${環境変数 DIOSA_DBGERRPAGE の値}×${環境変数 DIOSA_DBGERRSIZE の値}) ＋(${環境変数 DIOSA_DBGLOGPAGE の値}×${環境変数 DIOSA_DBGLOGSIZE の値})
0x----0900	5.25KB	メモリ管理機能起動時に作成
0x00000000	備考参照	メモリ管理機能起動時、及びメモリ拡張(アプリケーションからメモリ確保の要求でメモリ不足)時に作成 [サイズ見積り] 起動時:(288＋MMG 節-SHM 項-WRINIT の値) ＋ (288＋MMG 節-SHM 項-RDINIT の値) 拡張時 : (288＋MMG 節-SHM 項-WRINC の値) ＋ (288＋MMG 節-SHM 項-RDINC の値) 注意)メモリ種別毎に共有メモリのセグメントが作成される。 メモリ拡張の最大サイズは、メモリ種別毎の最大サイズ(WRMAX,RDMAX)で決まる。
0x----0801	1KB	閉塞管理機能起動に作成する
0x----0802	1KB	DIOSA/XTP 起動時に作成
0x----080a	60KB	DIOSA/XTP 起動時に作成
0x----0814	109.4KB	DIOSA/XTP 起動時に作成
0x----0c00	備考参照	メッセージ出力機能起動時に作成するメッセージキュー ${環境変数 DIOSA_MSG_QUEUE_SIZE の値}(省略時 : カーネルパラメータの設定値)
0x----2d01	備考参照	DIOSA/XTP 起動時に作成 [サイズ見積] (592 ＋(8×(${環境変数 DIOSA_DIOSA_DAM_DOWNCNT の値} －1))) ×50 ＋ 28
0x----2e01	128.5KB	DIOSA/XTP 起動時に作成

※ 1KB 未満のサイズは 1KB と表記

A.1.2 メモリキャッシュ(AP ノード)

キー	サイズ	備考
0x----4101	備考参照	メモリキャッシュ起動時に作成 初期サイズとして、サイズ見積の計算式でメモリを確保します。 領域が足らなくなると 10416 バイトずつメモリを確保します。 [サイズ見積] 48＋(ブリッジサーバの多重度×4＋定義した MAP 数＋100)×104
0x----4200	64 バイト	メモリキャッシュ起動時に作成
0x----421-	備考参照	メモリキャッシュ起動時に作成 [サイズ見積]

		48＋(自論理システム内 OLTP ノード数×48)
0x----422-	備考参照	メモリキャッシュ起動時に作成 [サイズ見積] 304＋32＋(IMENV 節に定義したマスタの数×96) ＋(IMENV 節に定義したスレーブの数×96) ＋32＋(定義した MAP の数×88) ＋32＋(定義したハッシュ範囲の数×8)
0x----4240 ～ 0x----42ff	131072 バイト	メモリキャッシュ起動時に作成

A.1.3 メモリキャッシュ(OLTP ノード)

キー	サイズ	備考
0x----4101	備考参照	メモリキャッシュ起動時に作成 初期サイズとして、サイズ見積の計算式でメモリを確保します。 領域が足らなくなると 10416 バイトずつメモリを確保します。 [サイズ見積] 48＋(ブリッジサーバの多重度×4＋定義した MAP 数＋100)×104
0x----4200	64 バイト	メモリキャッシュ起動時に作成
0x----421-	備考参照	メモリキャッシュ起動時に作成 [サイズ見積] 48＋(自論理システム内 OLTP ノード数×48)
0x----422-	備考参照	メモリキャッシュ起動時に作成 [サイズ見積] 304＋32＋(IMENV 節に定義したマスタの数×96) ＋(IMENV 節に定義したスレーブの数×96) ＋32＋(定義した MAP の数×88) ＋32＋(定義したハッシュ範囲の数×8)
0x----423-	備考参照	メモリキャッシュ起動時に作成 [サイズ見積] 168＋24＋(IMENV 節に定義した MAP の数×16400) ＋144＋(IMTABLECONF 節に定義した論理表の数×328)＋8208＋8208 ＋24＋(IMTABLECONF 節に定義した物理表の数×272) ＋24＋(IMTABLECONF 節に定義したプライマリキーの数×56) ＋(IMTABLECONF 節に定義したセカンダリキーの数×56) ＋24＋(IMTABLECONF 節に定義したキーの位置、サイズ情報の数×16) ＋24＋(IMENV 節に定義したノードの数×16) ＋24＋(IMENV 節に定義したレプリケーショングループの数×16) ＋24＋(IMENV 節に定義した MAP の数×16)
0x----4240 ～ 0x----42ff	131072 バイト	メモリキャッシュ起動時に作成
0x00000000	備考参照	メモリキャッシュ起動時に作成 [サイズ見積] 5416＋120＋(IMENV 節に定義したノードの数×24) ＋120＋(IMENV 節に定義したレプリケーショングループの数×64) ＋120＋(IMENV 節に定義した MAP の数×8752)

キー	サイズ	備考
0x00000000	備考参照	メモリキャッシュ起動時に作成 [サイズ見積] IMENV 節 MAP 項(DEF_MAP 項)に定義した IMS キューバッファサイズ
0x00000000	備考参照	メモリキャッシュ起動時に作成 下記サイズの共有メモリセグメントが IMENV 節に定義した MAP の数分作成される [サイズ見積] 142944＋IMTABLECONF 節に定義したトランザクション管理テーブルサイズ×224
0x00000000	273240 バイト	メモリキャッシュ起動時に作成
0x00000000	備考参照	メモリキャッシュ起動時に作成 [サイズ見積] 524288＋IMENV 節 BRIDGE 項に定義した IMS キューバッファサイズ×2
0x00000000	備考参照	利用者プロセス起動時に作成 [サイズ見積] IMENV 節 USERAP 項に定義した IMS キューバッファサイズ

A.1.4 データストア

キー	サイズ(※)	備考
0x----20f0	4KB	DIOSA/XTP 起動時に作成
0x----201-	40KB	ディレード転送起動時に作成
0x----202-	1KB	ログデータ転送をおこなう論理ノードでディレード転送起動時に作成
0x----203-	備考参照	[サイズ見積①] 100＋64×ログデータ転送相手の論理システム数 ＋32×ログデータ転送相手の接続先合計 ＋48×自論理システム内のログデータ転送制御動作ノード数 [サイズ見積②] 150＋32×ログデータ転送相手の論理システム数 ＋4×相手論理システム数×AP ノード数 AP ノード ： 動作ノードの場合は①＋②、非動作ノードの場合は① OLTP ノード ： 動作ノードの場合は②、非動作ノードの場合は作成しない ログデータ転送をおこなう論理ノードでディレード転送起動時に作成
0x----204-	備考参照	[サイズ見積] 32KB デフォルト RGSET を含むインスタンスグループの DB ノードでディレード転送起動時に作成 [サイズ見積] 600＋64×ログデータ格納先が Oracle のスーパーストリーム数 DBLAYER に指定した論理ノード、および AP ノードでディレード転送起動時に作成
0x----205-	32KB	DBLAYER に指定した論理ノード、および AP ノードでディレード転送起動時に作成
0x----220-	備考参照	[サイズ見積] 160＋950×センダユニットを定義したスーパーストリーム数 センダ機能開始時に作成
0x----230-	備考参照	[サイズ見積] 160＋950×レシーバユニットを定義したスーパーストリーム数 レシーバ機能開始時に作成
0x----241-	備考参照	[サイズ見積] 100＋550×ログリーダユニットを定義したスーパーストリーム数 ＋600×ログリーダユニット定義数 ログリーダ機能開始時に作成
0x----250-	備考参照	[サイズ見積] 280×スーパーストリーム定義数＋120×ユニット定義数 ディレード転送起動時に作成
0x----251-	備考参照	[サイズ見積] 24×Oracle に定義したスタック(パーティション)数 ログデータ格納先が Oracle の場合、ディレード転送起動時に作成
0x----252-	1KB	ディレード転送起動時に作成

※ 1KB 未満のサイズは 1KB と表記

A.2　メッセージキュー

DIOSA/XTP で使用するメッセージキュー一覧です。

A.2.1　メモリキャッシュ

キー	備考
0x00000000	メモリキャッシュ起動時に作成 [個数見積] 　IMENV 節に定義した MAP の数＋ブリッジサーバの多重度×4＋利用者プロセス数

A.3　ソケットファイル

DIOSA/XTP で使用するソケットファイル一覧です。

ファイルは${環境変数 DIOSA_TMP 指定ディレクトリ}/{論理ノード名}/socket 配下に作成します。

A.3.1　アプリケーション実行制御

ファイル名	備考
etg_etg_dmn	経過時間監視機能で使用
ops_xxx	稼動統計機能で使用(xxx は論理システム名)
cdd_cmdsend	コマンド配信機能で使用
cdd_dmnctl	コマンド配信機能で使用
cdd_hstctl	コマンド配信機能で使用
BCM	閉塞管理機能で使用
MMG_17	メモリ管理機能で使用
MDR_nnn	マルチスレッド動的置換機能で使用(nnn はプロセス ID)
msg_dmn	メッセージ出力機能で使用

A.3.2　メモリキャッシュ

ファイル名	備考
iic_dmn_main	インメモリサーバ所在管理で使用
iic_repg_xxx	インメモリサーバ所在管理で使用(xxx にはレプリケーショングループ ID が入ります)
IMS_CL_A_xxx	アクセスサーバで使用(xxx にはレプリケーショングループ ID が入ります)
IMS_CT_A_xxx	アクセスサーバで使用(xxx にはレプリケーショングループ ID が入ります)
IMS_CL_B_xxx	ブリッジサーバで使用(xxx にはブリッジサーバ ID が入ります)
IMS_CT_B_xxx	ブリッジサーバで使用(xxx にはブリッジサーバ ID が入ります)

A.3.3　データストア

ファイル名	備考
dlt_pcc	ログデータ転送制御で使用
dlt_pcd	ログデータ転送制御で使用
dlt_slm	ストリーム所在管理で使用
dlt_sla	ストリーム所在管理で使用
dtd_del	プールファイル制御で使用
dts_cmd	センダで使用
dts_comm_xxx	センダで使用(xxx には、スーパーストリーム定義数分の通番が入ります)
dtr_cmd	レシーバで使用
dtr_comm_xxx	レシーバで使用(xxx には、スーパーストリーム定義数分の通番が入ります)
dtl_cmd	ログリーダで使用

付録B　プロセス一覧

B.1　常駐プロセス一覧

プロセス名	説明	起動コマンド	停止コマンド	監視タイプ(*1)	動作ノード		
					AP	OLTP	DB
diapptrcd	アプリケーションログをファイルへ出力する。	diapptrcinit	diapptrcterm	①-B	○	○	○
dibcmdmn	ノードおよびCOの閉塞情報を管理する。	dibcmctrl -b	dibcmctrl -e	①-B	○	○	○
dicdddmn	要求のあったコマンドを適切なノードへ配信し実行する。	dicddctrl -b	dicddctrl -e	①-B	○	○	○
dicddhstdmn	コマンド配信履歴を出力する	dicddhstctl -s	dicddhstctl -e	⑤	○	○	○
dicocdmn	CO制御TPPの死活監視をおこなう。	dicocinit	dicocterm	①-A	○	○	
didamd	DIOSAの常駐プロセスを監視し、障害時に再起動を行う。	distart_com	distop	なし	○	○	○
didbgerrd	エラーログをファイルへ出力する。	didbginit	didbgterm	①-B	○	○	○
didbglogd	動作ログをファイルへ出力する。	didbginit	didbgterm	①-B	○	○	○
didltpathcommd	他論理システムへデータを送信する。	didltinit didltctrl(*3)	didltterm didltctrl -e(*3)	①-C	○(*2)		
didltpathdispd	他論理システムから受信したデータを適切なスーパーストリームへ転送する。	didltinit didltctrl(*3)	didltterm didltctrl -e(*3)	①-C	○(*2)	○(*2)	
didltpooldeld	プールファイルの不要データを削除する。	didltinit didltctrl(*3)	didltterm didltctrl -e(*3)	①-C	○(*2)	○(*2)	
didltslmagtd	スーパーストリームが処理するノードを管理する。	didltinit didltctrl(*3)	didltterm didltctrl -e(*3)	①-C	○(*2)	○(*2)	
didltslmmgrd	スーパーストリームが処理するノードを管理する。	didltinit didltctrl(*3)	didltterm didltctrl -e(*3)	①-C			○(*2)
didtldmnmngd	ディレードログリーダ機能の動作状態を管理する。	didtlinit didltctrl(*3)	didtlterm didltctrl -e(*3)	①-C	○(*2)	○(*2)	
didtlexecd	ログデータを処理する。	didtlinit didtlctrl	didtlterm	②	○(*2)	○(*2)	
didtrdmnmngd	ディレードレシーバ機能の動作状態を管理する。	didtrinit didltctrl(*3)	didtrterm didltctrl -e(*3)	①-C	○(*2)	○(*2)	
didtrexecd	他論理システムからログデータを受信する。	didtrinit didtrctrl	didtrterm	②	○(*2)	○(*2)	
didtsdmnmngd	ディレードセンダ機能の動作状態を管理する。	didtsinit didltctrl(*3)	didtsterm didltctrl -e(*3)	①-C	○(*2)	○(*2)	
didtsexecd	ログデータを他論理システムへ送信する。	didtsinit didtsctrl	didtsterm	②	○(*2)	○(*2)	
dietgdmn	プロセス稼動時間を監視する。	dietgctrl -b	dietgctrl -e	①-B	○	○	
diiicdmn	メモリDBのマスタ情報の管理と切り替え制御を行う。	diiminit	diimterm	①-B	○	○	
diimacssrv	メモリDBへアクセスする。	diiminit diimctrl -b	diimterm diimctrl -e	③		○	

プロセス名	説明	起動コマンド	停止コマンド	監視タイプ(*1)	動作ノード		
					AP	OLTP	DB
diimbrgsrv	他ノードのメモリ DB へのアクセスを中継する。	diiminit diimctrl -b	diimterm diimctrl -e	③		○	
dimdrnotifyd	DIOSA の SG 変更を管理する。	distart_com	distop	①-A	○	○	○
dimmgd	アプリケーションで使用するメモリを管理する。	distart_com	distop	①-A	○	○	○
dimsgd	API から要求があったメッセージをファイルへ出力する。	dimsgdctrl -b	dimsgdctrl -e	なし	○	○	○
dincmdbhcdmnap	Oracle Database の状態を管理する。	dincmdbhchkctrlinit	dincmdbhchkctrlterm	①-B	○	○	
dincmdbhcdmndb	Oracle Database の状態を管理する。	dincmdbhchkctrlinit	dincmdbhchkctrlterm	①-B			○
dincmdbhcdmnmn	Oracle Database の状態を管理する。	dincmdbhchkctrlinit	dincmdbhchkctrlterm	④	○	○	
dincmdbhcdmnsc	Oracle Database の状態を管理する。	dincmdbhchkctrlinit	dincmdbhchkctrlterm	①-B			○
diopsdmn	稼動統計情報をファイルへ出力する。	diopsctrl -b	diopsctrl -e	①-B	○	○	
diotcsndd	外部のアクセスポイントへ都度接続プロトコルで電文を送信する。	dincmpathdmnctrl -b	dincmpathdmnctrl -e	⑥	○	○	
diotcwatchd	diotcsndd のプロセス死活監視および再起動する。	dincmpathdmnctrl -b	dincmpathdmnctrl -e	①-A	○	○	
ditmcdmn	登録されたタイマを監視し、指定時刻または指定時間後にタイマを実行する。	ditmcdmn	ditmcdmnstop	①-B	○	○	
ditmccmdexeccmd	コマンドタイマを実行する。	- (*4)	ditmcdmnstop(*4)	⑦	○	○	
ditmccmdexecco	CO タイマを実行する。	- (*4)	ditmcdmnstop(*4)	⑦	○	○	
ditmccmdexectpb	TPBASE タイマを実行する。	- (*4)	ditmcdmnstop(*4)	⑦	○	○	

*1 監視タイプについて

なし : 監視対象外

① : DIOSA/XTP のプロセス監視機能(didamd)による監視

①-A: リトライオーバーした場合監視デーモン停止

①-B: リトライオーバーしても監視デーモンは継続

①-C: ①-A / ①-B の動作を環境定義等で指定

② : ディレード機能の各制御デーモン(didtldmnmngd、didtrdmnmngd、didtsdmnmngd)による監視、リトライオーバー時の監視プロセスの振る舞いは環境定義(DELAYED 節)で指定可能

③ : IM サーバ所在管理機能による監視、リトライオーバーしても監視プロセスは継続

④ : データベース管理機能による監視

⑤ : コマンド配信機能(dicdddmn)による監視、リトライオーバーしても監視プロセスは継続

⑥ : 都度接続管理機能(diotcwatchd)による監視、リトライオーバー時の監視プロセスの振る舞いは環境定義(OTCENV 節)で指定可能

⑦ : タイマ管理機能(ditmcdmn)による監視、異常終了した場合でも次回タイマ実行時に起動される

*2 DELAYED 節の定義にしたがい、必要な場合のみ起動する。

*3 個別の常駐プロセスを指定した起動/停止には、-F パラメータの指定が必要

*4 タイマ管理機能により自動起動され、タイマ処理が完了すると自動停止する。

B.2　TPP一覧

プロセス名	説明	補足
dicoctpp	CO の起動、停止、実行、エラー処理を制御する。	
dincmctrltpp	論理システム間通信管理機能、論理ノード間通信管理機能の端末状態監視・制御処理をおこなう。	
digntrsdtpp	処理が完了していない保証電文の再送を行う。	
digntdeltpp	保持時間を超過した受信電文情報の削除を行う。	
digntfintpp	処理完了した保証電文の削除、順序性保証電文削除時の次電文送信を行う。	

付録C　データベース一覧

C.1　TAM 表

C.1.1　データストア

ディレードプールファイル							
概要	ディレードのログデータを格納する						
論理表名	DIOSA_DELAYED_POOL_{プールファイル ID(2 桁)}{スタック ID(2 桁)}				物理表名	{論理表名}_{MAPID(10 桁)}	
データ/制御	データ	分割	あり	レコードサイズ	データ長	サイズ計算式	格納件数×データ長

ディレード書込み制御ファイル							
概要	ディレードのストリームの通番などを管理する 過去ディビジョン分の履歴情報も持つ						
論理表名	DIOSA_DELAYED_STRM				物理表名	{論理表名}_{MAPID(10 桁)}	
データ/制御	制御	分割	MAPID	レコードサイズ	176 byte	サイズ計算式	ストリーム数×ディビジョン数 × レコードサイズ

ディレードスーパーストリーム管理ファイル							
概要	ディレードのスーパーストリームの情報を管理する						
論理表名	DIOSA_DELAYED_SPST				物理表名	{論理表名}_{MAPID(10 桁)}	
データ/制御	制御	分割	MAPID	レコードサイズ	144 byte	サイズ計算式	スーパーストリーム数 × レコードサイズ

ディレードスタック管理ファイル							
概要	ディレードのプールファイルを構成するスタック情報を管理する						
論理表名	DIOSA_DELAYED_STACK				物理表名	{論理表名}_{MAPID(10 桁)}	
データ/制御	制御	分割	MAPID	レコードサイズ	48 byte	サイズ計算式	スタック数 × スーパーストリーム数 × レコードサイズ

ディレードスタック・通番管理ファイル							
概要	ディレードの各スタックに格納しているログデータ通番を管理する						
論理表名	DIOSA_DELAYED_STACKDATANO				物理表名	{論理表名}_{MAPID(10 桁)}	
データ/制御	制御	分割	MAPID	レコードサイズ	88 byte	サイズ計算式	スタック数 × スーパーストリーム数 × レコードサイズ

ディレードユニット管理ファイル							
概要	ディレードのユニットの情報を管理する						
論理表名	DIOSA_DELAYED_UNIT				物理表名	{論理表名}_{MAPID(10 桁)}	
データ/制御	制御	分割	MAPID	レコードサイズ	96 byte	サイズ計算式	総ユニット数 × レコードサイズ

DSAM システム制御ファイル							
概要	DSAM の環境定義情報を管理する						
論理表名	DIOSA_DSAM					物理表名	{論理表名}_{MAPID(10 桁)}
データ/制御	制御	分割	無	レコードサイズ	96 byte	サイズ計算式	レコードサイズ

センダユニット管理ファイル							
概要	センダのスーパーストリーム、ユニット情報を管理する						
論理表名	DIOSA_SENDER_UNIT					物理表名	{論理表名}_{MAPID(10 桁)}
データ/制御	制御	分割	MAPID	レコードサイズ	248 byte	サイズ計算式	センダユニット数 × レコードサイズ

センダストリーム管理ファイル							
概要	センダのストリーム情報を管理する						
論理表名	DIOSA_SENDER_STRM					物理表名	{論理表名}_{MAPID(10 桁)}
データ/制御	制御	分割	MAPID	レコードサイズ	168 byte	サイズ計算式	センダユニット数 × レコードサイズ

レシーバユニット管理ファイル							
概要	レシーバのスーパーストリーム、ユニット情報を管理する						
論理表名	DIOSA_RECEIVER_UNIT					物理表名	{論理表名}_{MAPID(10 桁)}
データ/制御	制御	分割	MAPID	レコードサイズ	200 byte	サイズ計算式	レシーバユニット数 × レコードサイズ

レシーバストリーム管理ファイル							
概要	レシーバのストリーム情報を管理する						
論理表名	DIOSA_RECEIVER_STRM					物理表名	{論理表名}_{MAPID(10 桁)}
データ/制御	制御	分割	MAPID	レコードサイズ	232 byte	サイズ計算式	レシーバユニット数 × レコードサイズ

ログリーダユニット管理ファイル							
概要	ログリーダのスーパーストリーム、ユニット情報を管理する						
論理表名	DIOSA_LOGREADER_UNIT					物理表名	{論理表名}_{MAPID(10 桁)}
データ/制御	制御	分割	MAPID	レコードサイズ	360 byte	サイズ計算式	ログリーダユニット数× レコードサイズ

ログリーダストリーム管理ファイル							
概要	ログリーダのストリーム情報を管理する						
論理表名	DIOSA_LOGREADER_STRM					物理表名	{論理表名}_{MAPID(10 桁)}
データ/制御	制御	分割	MAPID	レコードサイズ	160 byte	サイズ計算式	ログリーダユニット数× レコードサイズ

C.1.2　　電文保証

送信電文情報管理表							
概要	再送電文の一覧						
論理表名	DIOSA_MSGGNT_SNDCTL					物理表名	{論理表名}_{MAPID(10 桁)}
データ/制御	制御	分割	MAPID	レコードサイズ	128 byte	サイズ計算式	再送対象電文数×レコードサイズ

送信電文保存表							
概要	再送電文そのもの						
論理表名	DIOSA_MSGGNT_SNDMSG					物理表名	{論理表名}_{MAPID(10 桁)}
データ/制御	制御	分割	MAPID	レコードサイズ	1176+電文サイズ byte	サイズ計算式	再送対象電文数×レコードサイズ

順序性保証グループ管理表							
概要	順序性保証グループの一覧						
論理表名	DIOSA_MSGGNT_SEQGRP					物理表名	{論理表名}_{MAPID(10 桁)}
データ/制御	制御	分割	MAPID	レコードサイズ	56 byte	サイズ計算式	順序性保証グループ数×レコードサイズ

受信電文情報管理表							
概要	保証電文の受信 ID の一覧						
論理表名	DIOSA_MSGGNT_RCVCTL					物理表名	{論理表名}_{MAPID(10 桁)}
データ/制御	制御	分割	MAPID	レコードサイズ	40 byte	サイズ計算式	受信電文数×レコードサイズ

C.2　Oracle 表

C.2.1　　通信制御

DB ヘルスチェック更新確認用テーブル									
概要	Oracle DB が生存していることを確認する。								
表名	DIOSA_NCM_DBHC_01～32								
データ/制御	制御	レコードサイズ	138 byte	サイズ計算式	32 × レコードサイズ				

C.2.2　　データストア

ストリーム所在管理 制御テーブル									
概要	ディレードのストリーム所在管理機能 制御情報を管理する								
表名	DIOSA_DELAYED_LOCATION								
データ/制御	制御	分割	無	バッファ	ー	レコードサイズ	2048 byte	サイズ計算式	レコードサイズ

ストリーム所在管理 スーパーストリーム情報テーブル									
概要	ディレードのストリーム所在管理機能 スーパーストリーム所在情報を管理する								
表名	DIOSA_DELAYED_LOCATION_SPST								
データ/制御	制御	分割	無	バッファ	常駐	レコードサイズ	2048 byte	サイズ計算式	スーパーストリーム数 × レコードサイズ

ディレードプールファイル									
概要	ディレードのログデータを格納する								
表名	DIOSA_DELAYED_POOL								
データ/制御	データ	分割	スーパー×スタック	バッファ	ー	レコードサイズ	32768 byte	サイズ計算式	格納件数 × レコードサイズ ※PRIMARY KEY の INDEX 領域も考慮する必要あり

ディレード書込み制御ファイル									
概要	ディレードのストリームの通番などを管理する								
表名	DIOSA_DELAYED_STRM								
データ/制御	制御	分割	無	バッファ	常駐	レコードサイズ	2048 byte	サイズ計算式	スーパーストリーム数 × レコードサイズ

ディレード書込み制御履歴ファイル									
概要	ディレードのストリームの書込履歴情報を管理する								
表名	DIOSA_DELAYED_HISTORY								
データ/制御	制御	分割	無	バッファ	常駐	レコードサイズ	2048 byte	サイズ計算式	スーパーストリーム数 × ディビジョン数 × レコードサイズ

ディレード書込制御ビュー									
概要	ディレード書込み制御ファイルと書込み制御履歴ファイルを結合したビュー								
表名	DIOSA_DELAYED_STRMVIEW								
データ/制御	制御	分割	無	バッファ	－	レコードサイズ	－	サイズ計算式	－

ディレードスーパーストリーム管理ファイル									
概要	ディレードのスーパーストリームの情報を管理する								
表名	DIOSA_DELAYED_SPST								
データ/制御	制御	分割	無	バッファ	常駐	レコードサイズ	2048 byte	サイズ計算式	スーパーストリーム数 ×レコードサイズ

ディレードスタック管理ファイル									
概要	ディレードのプールファイルを構成するスタック情報を管理する								
表名	DIOSA_DELAYED_STACK								
データ/制御	制御	分割	無	バッファ	常駐	レコードサイズ	2048 byte	サイズ計算式	スタック数 × スーパーストリーム数 × レコードサイズ

ディレードスタック・通番管理ファイル									
概要	ディレードの各スタックに格納しているログデータ通番を管理する								
表名	DIOSA_DELAYED_STACKDATANO								
データ/制御	制御	分割	無	バッファ	常駐	レコードサイズ	2048 byte	サイズ計算式	スタック数 × スーパーストリーム数 × レコードサイズ

ディレードスタック初期化情報ファイル									
概要	スタック(パーティション)の追加削除(DROP/ADD)で障害が発生した際の障害情報を管理する								
表名	DIOSA_DELAYED_FAIL_BKUP								
データ/制御	制御	分割	無	バッファ	－	レコードサイズ	2048 byte	サイズ計算式	スタック数 × スーパーストリーム数 × レコードサイズ

ディレードユニット管理ファイル									
概要	ディレードのユニットの情報を管理する								
表名	DIOSA_DELAYED_UNIT								
データ/制御	制御	分割	無	バッファ	常駐	レコードサイズ	2048 byte	サイズ計算式	総ユニット数 × レコードサイズ

DSAM システム制御ファイル									
概要	DSAM の環境定義情報を管理する								
表名	DIOSA_DSAM								
データ/制御	制御	分割	無	バッファ	ー	レコードサイズ	2048 byte	サイズ計算式	2048 byte

センダユニット管理ファイル									
概要	センダのスーパーストリーム、ユニット情報を管理する								
表名	DIOSA_SENDER_UNIT								
データ/制御	制御	分割	無	バッファ	常駐	レコードサイズ	2048 byte	サイズ計算式	センダユニット数 × レコードサイズ

センダストリーム管理ファイル									
概要	センダのストリーム情報を管理する								
表名	DIOSA_SENDER_STRM								
データ/制御	制御	分割	無	バッファ	常駐	レコードサイズ	2048 byte	サイズ計算式	センダユニット数 × レコードサイズ

レシーバユニット管理ファイル									
概要	レシーバのスーパーストリーム、ユニット情報を管理する								
表名	DIOSA_RECEIVER_UNIT								
データ/制御	制御	分割	無	バッファ	常駐	レコードサイズ	2048 byte	サイズ計算式	レシーバユニット数 × レコードサイズ

レシーバストリーム管理ファイル									
概要	レシーバのストリーム情報を管理する								
表名	DIOSA_RECEIVER_STRM								
データ/制御	制御	分割	無	バッファ	常駐	レコードサイズ	2048 byte	サイズ計算式	レシーバユニット数 × レコードサイズ

ログリーダユニット管理ファイル									
概要	ログリーダのスーパーストリーム、ユニット情報を管理する								
表名	DIOSA_LOGREADER_UNIT								
データ/制御	制御	分割	無	バッファ	常駐	レコードサイズ	2048 byte	サイズ計算式	ログリーダユニット数 × レコードサイズ

ログリーダストリーム管理ファイル									
概要	ログリーダのストリーム情報を管理する								
表名	DIOSA_LOGREADER_STRM								
データ/制御	制御	分割	無	バッファ	常駐	レコードサイズ	2048 byte	サイズ計算式	ログリーダユニット数 × レコードサイズ

C.2.3 電文保証

送信電文情報管理表							
概要	再送電文の一覧						
論理表名	DIOSA_MSGGNT_SNDCTL						
データ/制御	制御	分割	無	レコードサイズ	199 byte	サイズ計算式	再送対象電文数×レコードサイズ

送信電文保存表							
概要	再送電文そのもの						
論理表名	DIOSA_MSGGNT_SNDMSG						
データ/制御	データ	分割	無	レコードサイズ	2032+電文サイズ byte	サイズ計算式	再送対象電文数×レコードサイズ

順序性保証グループ管理表							
概要	順序性保証グループの一覧						
論理表名	DIOSA_MSGGNT_SEQGRP						
データ/制御	制御	分割	無	レコードサイズ	124 byte	サイズ計算式	順序性保証グループ数×レコードサイズ

受信電文情報管理表							
概要	保証電文の受信 ID の一覧						
論理表名	DIOSA_MSGGNT_RCVCTL						
データ/制御	制御	分割	無	レコードサイズ	84 byte	サイズ計算式	受信電文数×レコードサイズ

付録D　諸元一覧

D.1　共通

項目名称	単位	諸元	備考
論理システム数	個数	1～255	
論理ノード数	個数	1～256	1 論理システム内は 256 ノードまで
論理ライブラリ数	個数	0～32767	1 プロセスに対しては最大 5120 個まで

D.2　アプリケーション実行制御

項目名称	単位	諸元	備考
CO 制御電文長	K バイト	3～2097151	制御情報を含む TPBASE の最大電文長以下にすること
CO 制御が動作する TPBASE TPP 数	個数	1～2147483647	
CO 制御が動作する TPBASE クラス数	個数	1～1000	

D.3　通信制御

項目名称	単位	諸元	備考
1 論理ノード内の TPBASE 数	個数	1～16	
DB インスタンス数 (インスタンスグループ数)	個数	0～16	
RAC 構成 DB インスタンス数	個数	2	

D.4　メモリキャッシュ

項目名称	単位	諸元	備考
レプリケーショングループ数	個数	1 ～ 256	
MAP 数	個数	1 ～ 256	レプリケーショングループ内の定義数 総数は最大 65536
論理表数	個数	1 ～ 1023	
論理表名	文字	1 ～ 255	
物理表数	個数	1 ～ 65536	論理表内の定義数
物理表名	文字	1 ～ 255	
プライマリキー名	文字	1 ～ 30	
セカンダリキー名	文字	1 ～ 30	
プライマリキー数	個数	1	論理表内の定義数
セカンダリキー数	個数	0 ～ 62	論理表内の定義数
キー情報数	個数	1 ～ 100	論理表内のプライマリキー、セカンダリキー内の定義数
同時接続アプリケーション数	個数	1 ～ 2000	1 レプリケーショングループあたりの接続数
メインキー	バイト	32	固定長
スレーブ TAM インスタンス数	個数	0～10	

D.5　データストア

項目名称	単位	諸元	備考
最大送信電文長	バイト	524288	制御情報を含む
ログデータサイズ	バイト	1～2147483647	
スーパーストリーム名	文字	1～15	
スーパーストリーム数	個数	1～1024	同一 MAPID 配下には 100 個まで
センダユニット数	個数	0～1	1 スーパーストリーム配下の定義数
ログリーダユニット数	個数	0～16	1 スーパーストリーム配下の定義数
スタック数	個数	2～99	
ディビジョン ID	数値	1～2147483647	

項目名称	単位	諸元	備考
ログデータ通番	数値	1～ 9223372036854 775806	ログデータ分割後通番

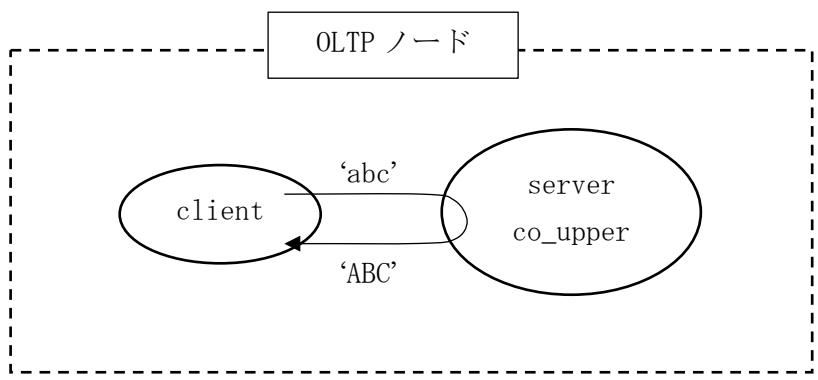
付録E　サンプルについて

サンプルは /opt/diosa_xtp/samples に置かれています。このディレクトリに置かれているサンプルは以下の通りです。

- sql　　Oracle 用の SQL
- tam　　TAM 用の DIOSA/XTP および TAM の設定ファイル
- sample1　　クライアントから送られた英字を大文字小文字変換して返すアプリケーション
- sample2　　データベースアクセスアプリケーション Oracle 版
- sample3　　データベースアクセスアプリケーション TAM 版
- sample4　　データベースアクセスアプリケーション Oracle 版 複数ノード構成
- sample5　　データベースアクセスアプリケーション TAM 版　　複数ノード構成

E.1　sample1 簡単なアプリケーション

クライアントから送られてきた英字文字列を、大文字にして返します。アプリケーションとノードの構成は以下の図のとおりです。



E.1.1　使用方法

1) sample1 配下をコピーする。この例では現在のユーザのホームディレクトリにコピーします。

```
% cp -r /opt/diosa_xtp/samples/sample1 ~/
```

2) config.base を編集します。

```
% cd ~/sample1
% vi config.base
```

編集する項目は以下のとおりです。

項目	説明	例
WORK_DIR	sample1 のコピー先を指定します。	/home/user1/sample1
IPCKEY	サンプルが使用する IPC リーソスのキー上位 2 バイトを指定します。	FFFF
TPMONITOR_PREFIX	TPBASE を識別する文字列の先頭 2 文字を指定します。	xx
IP_ADDRESS	サンプルが通信で使用する IP アドレスを指定します。	192.168.0.1
PORT	サンプルが使用するポートを指定します。 指定されたポート番号から 50 のポートを使用します。	32768

3) 設定ファイルを作成します。

```
% make config
```

4) 環境変数の反映

```
% cd lsys1.oltp1
% . ./test.env
```

5) アプリケーションの作成、設定ファイルの反映をします。

```
% make
```

6) 起動

```
% distart_OLTP -c
```

7) アプリケーションを呼び出します。

```
% ./app/cl abc
send message:"abc"
recv message:"ABC"

% ./app/cl XYZ
send message:"XYZ"
recv message:"xyz"
```

8) 停止

```
% distop_OLTP -f
```

E.1.2 ディレクトリ構成

sample1/lsys1.oltp1

app	アプリケーション
bin	起動・停止スクリプト
diosa	DIOSA/XTP の環境定義
log	ログファイル
tmp	一時ファイル
tpbase	TPBASE の環境定義

E.1.3 アプリケーションの説明

(1) クライアントアプリケーション

cl.c

1. 37-52 行目 サーバにソケットを用いて接続
2. 53-60 行目 パメータで指定された英字文字列をサーバに送信
3. 62-81 行目 サーバからの応答を受信
4. 83 行目 応答を表示

(2)　　サーバアプリケーション

(a)　lsnrexit.c

1. 12-17 行目 電文受信時の動作 電文長と電文を処理する TXID を指定する
2. 19-20 行目 電文受信時の動作 このアプリケーションでは必要な処理はない
3. 22-23 行目 不正電文 電文を破棄する

(b)　co_upper.c

1. 26-33 行目 電文を受信 diosarecvtx()
2. 35-43 行目 応答電文用の領域確保 diosamsgbufalloc()
3. 44-50 行目 応答電文の作成 大/小文字変換
4. 52-67 行目 応答の送信 diosasendtx()

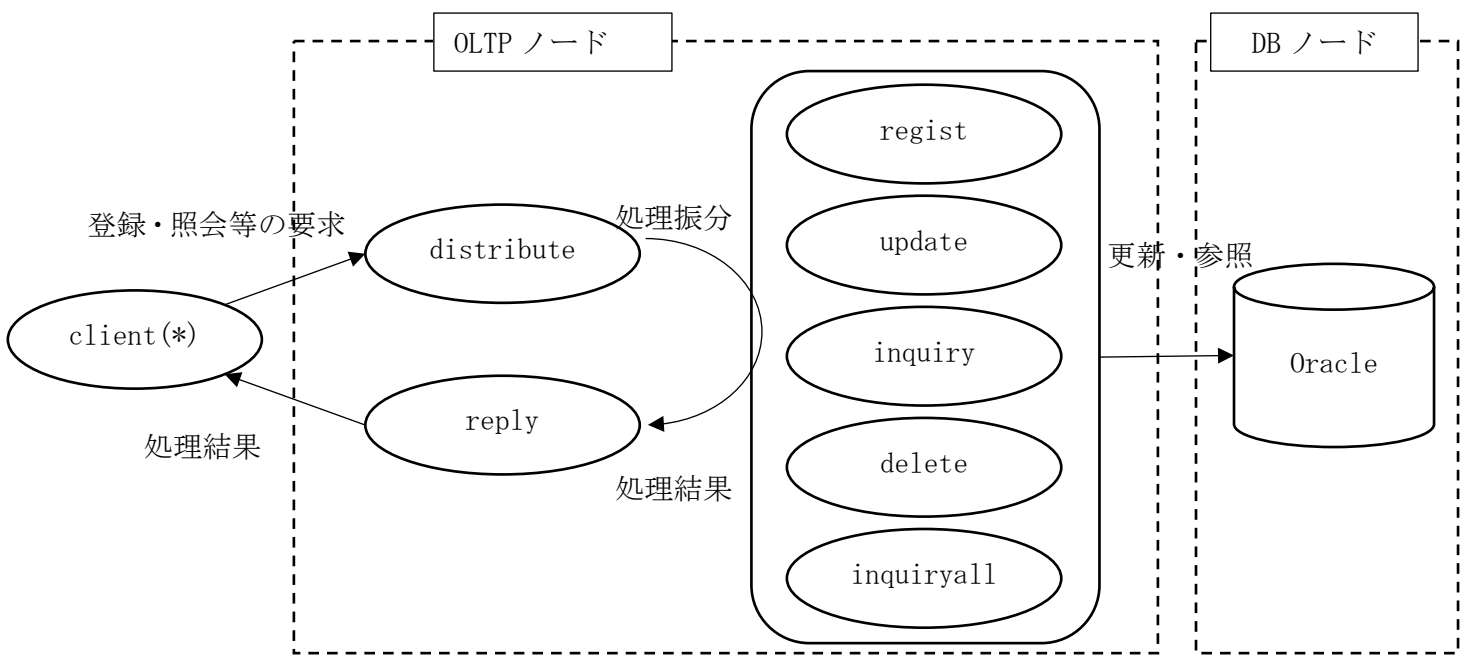
(3)　　共通

(a)　message.h

サーバ/クライアント間で送受信する電文形式(構造体)を定義する

E.2　sample2 データベースアクセスアプリケーション Oracle 版

ユーザ ID とユーザ情報の登録・更新・削除・照会・全ユーザ照会を行います。データの格納先として OracleDB を利用します。アプリケーションとノードの構成は以下の図のとおりです。



(*)サンプル構成の都合上、クライアントアプリケーションは OLTP ノードに格納していますが、
他のノードでも動作します。

E.2.1　使用方法

(1) sample2 配下をコピーする（OLTP ノード、DB ノードで行う）

```
% cp -r /opt/diosa_xtp/samples/sample2 ~/
```

(2) config.base を編集する（OLTP ノード、DB ノードで行う）

OLTP,DB ノードで同じ設定にしてください。

```
% cd ~/sample2
% vi config.base
```

項目	説明	例
WORK_DIR	sample1 のコピー先を指定します。	/home/user1/sample2
IPCKEY	サンプルが使用する IPC リーソスのキー上位 2 バイトを指定します。	FFFF
TPMONITOR_PREFIX	TPBASE を識別する文字列の先頭 2 文字を指定します。	xx
IP_ADDRESS11	アプリケーションが動くサーバの IP アドレスを指定します。	192.168.0.1
IP_ADDRESS101	Oracle が動作するサーバの IP アドレスを指定します。	192.168.0.2
PORT	サンプルが使用するポート番号を指定します。指定されたポート番号から 50 のポートを使用します。	32768
ORACLE_HOME	Oracle がインストールされたディレクトリを指定します。	/u01/app/oracle/product/12.1/dbhome_1
DB_NAME	Oracle へアクセスするためにネット・サービス名を指定	sampledb

	します。	
DB_USER	Oralce へアクセスするためのユーザ名を指定します。	user
DB_PASSWORD	Oralce へアクセスするためのパスワードを指定します。	passwd

(3) OracleDB に DB 死活監視およびアプリケーションが使う表を作成する

表を作成する表領域や他のパラメータは必要に応じて修正してください。

```
% sqlplus ユーザ名/パスワード@DB 名 @ /opt/diosa_xtp/samples/sql/create_table_ncm.sql
% sqlplus ユーザ名/パスワード@DB 名 ￥
        @ /opt/diosa_xtp/samples/sample2/lsys2.oltp11/app/create_userlist.sql
```

(4) 設定ファイルを作成する(OLTP ノード、 DB ノードで行う)

```
% make config
```

(5) 環境変数の反映を行う (OLTP ノード、DB ノードで行う)

OLTP ノード

```
% cd lsys2.oltp11
% . ./test.env
```

DB ノード

```
% cd lsys2.db01
% . ./test.env
```

(6) アプリケーションの作成　設定ファイルの反映 (OLTP ノード、DB ノードで行う)

```
% make
```

(7) 起動

DB ノード -> AP ノードの順に起動します。

A) DB ノード

```
% distart_DB -c
```

B) OLTP ノード

```
% distart_OLTP -c
```

(8) アプリケーション呼び出し (OLTP ノード で行う)

全データ照会

```
% ./app/cl inquiryall
Result:0
NumOfRecord:0
```

データ登録

```
% ./app/cl regist NEC "Nippon Denki"
```

```
Result:0
NumOfRecord:1
"NEC" "Nippon Denki"
```

全データ照会

```
% ./app/cl inquiryall
Result:0
NumOfRecord:1
"NEC" "Nippon Denki"
```

データ登録

```
% ./app/cl regist DIOSA "Distributed Integrated Operating System for Applications"
Result:0
NumOfRecord:1
"DIOSA" "Distributed Integrated Operating System for Applications"
```

全データ照会

```
% ./app/cl inquiryall
Result:0
NumOfRecord:2
"NEC" "Nippon Denki"
"DIOSA" "Distributed Integrated Operating System for Applications"
```

(9) 停止

OLTP ノード -> DB ノードの順に停止します。

A) OLTP ノード

```
% distop_OLTP -f
```

B) DB ノード停止

```
% distop_DB -f
```

E.2.2 ディレクトリ構成

sample2/lsys2.oltp11

app	アプリケーション
bin	起動・停止スクリプト
diosa	DIOSA/XTP の環境定義
log	ログファイル
tmp	一時ファイル
tpbase	TPBASE の環境定義

sample2/lsys2.db01

app	アプリケーション
bin	起動・停止スクリプト
diosa	DIOSA/XTP の環境定義
log	ログファイル
tmp	一時ファイル

E.2.3 アプリケーションの説明

(1) クライアントアプリケーション

(a) cl.c (cl.c.template)

1. 162-168 行目 コマンドのパラメータチェック
2. 176-190 行目 サーバにソケットを用いて接続　サーバが冗長化されている場合はラウンドロビンで接続
3. 191-195 行目 サーバに要求電文を送信
4. 197-201 行目 サーバからの応答を受信(ヘッダのみ)
5. 203-207 行目 応答全体を受け取るための領域確保
6. 209-212 行目 応答全体を受信
7. 214-217 行目 応答を表示

SendRecv.c SendRecv.h 送受信用のサブルーチン 送受信のタイムアウトチェックを行う

(2) サーバアプリケーション

(a) lsnrexit.c

1. 12-17 行目 電文受信時の動作　電文長と電文を処理する TXID を指定する
2. 19-20 行目 電文送信時の動作　このアプリケーションでは何もしない

(b) analyze.c

受信した電文を処理するアプリケーション(CO,関数)を決定します。

1. 14-17 行目 クライアントから送られた電文の場合は distribute を呼ぶ
2. 19-23 行目 システム内の電文では、送信する際に指定したアプリケーションを呼ぶ
3. 25-28 行目 不正な電文は処理を中止する

(c) distribute.c

クライアントから受信した電文を処理内容により振り分けてアプリケーションに転送します。

1. 38-44 行目 電文を受信 diosarecvtx()
2. 46-52 行目 クライアントからの電文をチェック
3. 55-61 行目 電文を処理するアプリケーションへの送る電文領域の確保 diosamsgbufalloc()
3. 65-89 行目 送信電文の作成 送り先のアプリケーション名を決定、送り先のノード選択(ラウンドロビン)
4. 91-99 行目 クライアントからの要求を処理するアプリケーションに電文を送信 diosasendtx()

(d)　regist.pc

ユーザ情報を登録します。

1. 38-44 行目　要求電文を受け取る diosarecvtx()
2. 53-61 行目　処理結果を返す電文領域の確保 diosamsgbufalloc()
3. 64-74 行目　DB アクセスするためのコンテキストを取得 diosagetdbctx()
4. 78-95 行目　ユーザ情報を登録 EXEC SQL INSERT
5. 98-112 行目　処理結果の電文を作成
6. 109-114 行目 クライアントへ結果を送信するアプリケーション reply に電文を送信

(e)　update.pc

ユーザ情報を更新します。

1. 38-44 行目　要求電文を受け取る diosarecvtx()
2. 53-61 行目　処理結果を返す電文領域の確保 diosamsgbufalloc()
3. 62-72 行目　DB アクセスするためのコンテキストを取得 diosagetdbctx()
4. 75-92 行目　ユーザ情報を更新 EXEC SQL UPDATE
5. 94-102 行目　処理結果の電文を作成
6. 119-119 行目 クライアントへ結果を送信するアプリケーション reply に電文を送信

(f)　delete.pc

ユーザ情報を削除します

1. 38-45 行目　要求電文を受け取る diosarecvtx()
2. 51-59 行目　処理結果を返す電文領域の確保 diosamsgbufalloc()
3. 62-71 行目　DB アクセスするためのコンテキストを取得 diosagetdbctx()
4. 74-91 行目　ユーザ情報を削除 EXEC SQL DELETE
5. 94-107 行目　処理結果の電文を作成
6. 109-114 行目 クライアントへ結果を送信するアプリケーション reply に電文を送信

(g)　inquiry.pc

ユーザ情報を照会します

1. 38-45 行目　要求電文を受け取る diosarecvtx()
2. 50-58 行目　処理結果を返す電文領域の確保 diosamsgbufalloc()
3. 61-69 行目　DB アクセスするためのコンテキストを取得 diosagetdbctx()
4. 71-92 行目　ユーザ情報を取得 EXEC SQL SELECT
5. 94-108 行目　処理結果の電文を作成
6. 110-115 行目 クライアントへ結果を送信するアプリケーション reply に電文を送信

(h)　inquiryall.pc

全ユーザ情報を照会します

1. 45-51 行目　要求電文を受け取る diosarecvtx()
2. 56-64 行目　処理結果の一時保存領域の確保 diosamalloc()
3. 66-75 行目　DB アクセスするためのコンテキストを取得 diosagetdbctx()
4. 94-101 行目　情報取得のカーソルオープン EXEC SQL DECLARE ...

5. 103-139 行目 ユーザ情報を 1 件ずつ取得 EXEC SQL FETCH ...
6. 141-131 行目 カーソルクローズ EXEC SQL CLOSE
7. 146-154 行目 取得した件数文の応答電文領域を確保 diosamsgbufalloc()
8. 156-175 行目 処理結果の電文を作成
9. 177-182 行目 クライアントへ結果を送信するアプリケーション reply に電文を送信

(i) reply.c

クライアントへ応答を返します。

1. 24-30 行目 要求処理結果を受け取る diosarecvtx()
2. 32-39 行目 クライアントへ返す電文領域の確保 diosamsgbufalloc()
3. 43-53 行目 応答電文の作成
4. 55-60 行目 クライアントへ処理結果を送信 diosasendtx()

(j) dbconnect.pc (dbconnect.pc.template)

DB への接続処理を行います。

DIOSA/XTP はこの関数により確保されたコンテキストを使って DB へアクセスします。

コンテキストのクローズは DIOSA/XTP が行います。

1. 20 行目 DB へのコンテキスト確保 EXEC SQL CONTEXT ALLOCATE ...
2. 30 行目 DB へ接続 EXEC SQL CONNECT
3. 39 行目 コンテキストのアドレスを返却領域に設定

(3) 共通

(a) message.h

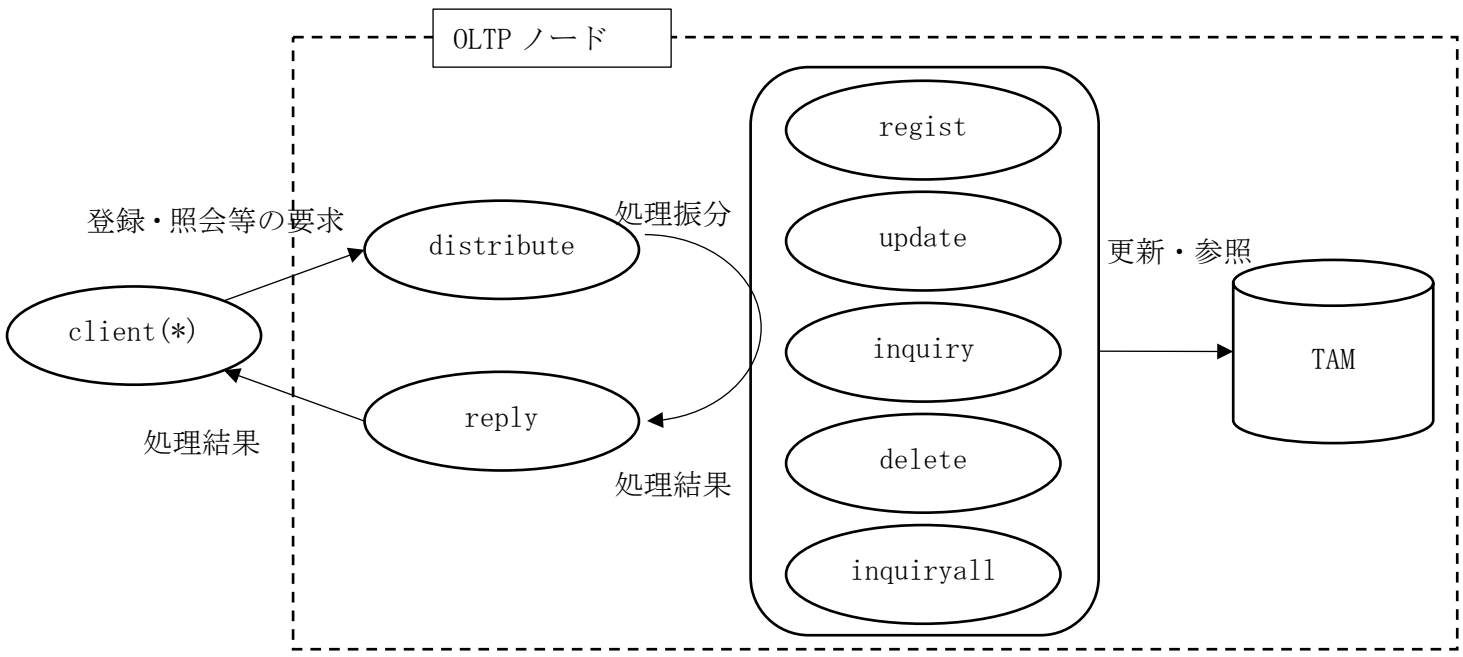
サーバ/クライアント間で送受信する電文形式(構造体)を定義します。

(b) create_userlist.sql

ユーザ情報を格納する Oracle 表の定義

E.3　sample3 データベースアクセスアプリケーション TAM 版

ユーザ ID とユーザ情報の登録・更新・削除・照会・全ユーザ照会を行います。データの格納先として TAM を利用します。アプリケーションとノードの構成は以下の図のとおりです。



(*)サンプル構成の都合上、クライアントアプリケーションは OLTP ノードに格納していますが、
他のノードでも動作します。

E.3.1　使用方法

(1) sample3 配下をコピーする

```
% cp -r /opt/diosa_xtp/samples/sample3 ~/
```

(2) config.base を編集する

```
% cd ~/sample3
% vi config.base
```

項目	説明	例
WORK_DIR	sample1 のコピー先を指定します。	/home/user1/sample3
IPCKEY	サンプルが使用する IPC リーソスのキー上位 2 バイトを指定します。	FFFF
TPMONITOR_PREFIX	TPBASE を識別する文字列の先頭 2 文字を指定します。	xx
NODE11	アプリケーションが動くサーバ名を指定します。	server1
IP_ADDRESS11	アプリケーションが動くサーバの IP アドレスを指定します。	192.168.0.1
PORT	サンプルが使用するポート番号を指定します。指定されたポート番号から 50 のポートを使用します。	32768

(3) DIOSA/XTP TPBASE TAM の設定ファイルを作成する

```
% make config
```

(4) 環境変数の反映

```
% cd lsys3.oltp11
```

```
% . ./test.env
```

(5) アプリケーションの作成　設定ファイルの反映

```
% make
```

(6) TAM の設定

```
% cd ../lsys3.tam
```

TAM 環境定義のチェック

```
% make check
```

TMA 環境定義の反映

```
% make update
```

テーブルの作成

```
% make create
```

(7) 起動

```
% distart_OLTP -c
```

(8) アプリケーション呼び出し

全データ照会

```
% ./app/cl inquiryall
Result:0
NumOfRecord:0
```

ユーザ情報登録

```
% ./app/cl regist NEC "Nippon Denki"
Result:0
NumOfRecord:1
"NEC" "Nippon Denki"
```

全データ照会

```
% ./app/cl inquiryall
Result:0
NumOfRecord:1
"NEC" "Nippon Denki"
```

ユーザ情報登録

```
% ./app/cl regist DIOSA "Distributed Integrated Operating System for Applications"
Result:0
NumOfRecord:1
"DIOSA" "Distributed Integrated Operating System for Applications"
```

全データ照会

```
% ./app/cl inquiryall
Result:0
NumOfRecord:2
"DIOSA" "Distributed Integrated Operating System for Applications"
"NEC" "Nippon Denki"

(9) 停止
% distop_OLTP -f
```

E.3.2　　ディレクトリ構成

sample3/lsys3.oltp11

app	アプリケーション
bin	起動・停止スクリプト
diosa	DIOSA/XTP の環境定義
log	ログファイル
tmp	一時ファイル
tpbase	TPBASE の環境定義

sample3/lsys3.tam

conf	TAM の環境定義
tablefile	

var/lsys3oltp11

E.3.3　　アプリケーションの説明

(1)　　クライアントアプリケーション

(a)　cl.c (cl.c.template)

1. 162-168 行目 コマンドのパラメータチェック
2. 176-190 行目 サーバにソケットを用いて接続　サーバが冗長化されている場合はラウンドロビンで接続
3. 191-195 行目 サーバに要求電文を送信
4. 197-201 行目 サーバからの応答を受信(ヘッダのみ)
5. 203-207 行目 応答全体を受け取るための領域確保
6. 209-212 行目 応答全体を受信
7. 214-217 行目 応答を表示

SendRecv.c SendRecv.h 送受信用のサブルーチン 送受信のタイムアウトチェックを行う

(2)　　サーバアプリケーション

(a)　lsnrexit.c

1. 12-17 行目 電文受信時の動作　電文長と電文を処理する TXID を指定する
2. 19-20 行目 電文送信時の動作　このアプリケーションでは何もしない

(b)　analyze.c

受信した電文を処理するアプリケーション(CO,関数)を決定します。

1. 14-17 行目 クライアントから送られた電文の場合は distribute を呼ぶ
2. 19-23 行目 システム内の電文では、送信する際に指定したアプリケーションを呼ぶ
3. 25-28 行目 不正な電文は処理を中止する

(c)　distribute.c

クライアントから受信した電文を処理内容により振り分けてアプリケーションを呼び出します。

1. 38-45 行目　電文を受信 diosarecvtx()
2. 47-53 行目　クライアントからの電文をチェック
3. 56-62 行目　電文を処理する処理への送る電文領域の確保 diosamsgbufalloc()
4. 66-96 行目　送信電文の作成 アプリケーション名を決定
   送り先のノード選択
   登録・更新・削除処理は名前をキーに送信先ノードを決定する
   照会・全照会処理はラウンドロビンで決定する
5. 98-106 行目 電文の送信 diosasendtx()

(d)　regist.c

ユーザ情報を登録します。

1. 33-39 行目　要求電文を受け取る diosarecvtx()
2. 41-47 行目　処理結果を返す電文領域の確保 diosamsgbufalloc()
3. 52-67 行目　ユーザ情報を登録する IM のパーティションを選択する diosagetmap() diosaimsetmap()
4. 69-74 行目　ユーザ情報テーブル ID の取得 diosaimgettblid()
5. 77-114 行目　ユーザ情報を登録 diosaimwrite()
6. 117-131 行目 処理結果の電文を作成
7. 133-138 行目 クライアントへ結果を送信するアプリケーション reply に電文を送信

(e)　update.c

ユーザ情報を更新します。

1. 34-40 行目　要求電文を受け取る diosarecvtx()
2. 42-48 行目　処理結果を返す電文領域の確保 diosamsgbufalloc()
3. 53-68 行目　更新するユーザ情報が属する IM のパーティションを選択する diosagetmap() diosaimsetmap()
4. 70-75 行目　ユーザ情報テーブル ID を取得 diosaimgettblid()
5. 77-102 行目　更新対象のユーザ情報を読み込み&ロック diosaimcondsetkey() diosaimread1()
6. 104-128 行目 ユーザ情報を更新 diosaimrewrite()
7. 131-144 行目 処理結果の電文を作成
8. 146-152 行目 クライアントへ結果を送信するアプリケーション reply に電文を送信

(f)　delete.c

ユーザ情報を削除します

1. 33-39 行目　要求電文を受け取る diosarecvtx()
2. 41-47 行目　処理結果を返す電文領域の確保 diosamsgbufalloc()
3. 52-67 行目　削除するユーザ情報が属する IM のパーティションを選択する diosagetmap() diosaimsetmap()
4. 70-75 行目　ユーザ情報テーブル ID を取得 diosaimgettblid()
5. 77-106 行目　ユーザ情報を削除 diosimcondsetkey() diosaimdeletex1()
6. 109-122 行目 処理結果の電文を作成
8. 124-129 行目 クライアントへ結果を送信するアプリケーション reply に電文を送信

(g)　inquiry.c

1. 34-40 行目　要求電文を受け取る diosarecvtx()
2. 42-48 行目　処理結果を返す電文領域の確保 diosamsgbufalloc()
3. 53-68 行目　照会するるユーザ情報が属する IM のパーティションを選択する diosagetmap() diosaimsetmap()
4. 70-75 行目　ユーザ情報テーブル ID を取得 diosaimgettblid()
5. 77-105 行目　ユーザ情報を読込 diosimcondsetkey() diosaimread1()
6. 108-122 行目 処理結果の電文を作成
8. 124-129 行目 クライアントへ結果を送信するアプリケーション reply に電文を送信

(h)　inquiryall.c

全ユーザ情報を照会します

1. 43-50 行目　要求電文を受け取る diosarecvtx()
2. 53-61 行目　処理結果の一時保存領域の確保 diosamalloc()
3. 63-69 行目　ユーザ情報テーブル ID を取得 diosaimgettblid()
4. 71-76 行目　パーティションの一覧を取得 diosagetmaplist()
5. 78-138 行目　全パーティションからユーザ情報を取得
6. 81-86 行目　ユーザ情報を読み込むパーティションを選択 diosaimsetmap()
7. 98-94 行目　検索コンテキストを取得 diosaimctxopen()
8. 96-168 行目　ユーザ情報を 1 件ずつ読み込み diosaimread()
9. 130-136 行目 検索コンテキストを開放 diosaimctxclose()
10. 140-146 行目 取得した件数文の応答電文領域を確保 diosamsgbufalloc()
11. 150-168 行目 処理結果の電文を作成
12. 170-175 行目 クライアントへ結果を送信するアプリケーション reply に電文を送信

(a)　reply.c

クライアントへ応答を返します。

1. 24-31 行目 要求処理結果を受け取る diosarecvtx()
2. 33-40 行目 クライアントへ返す電文領域の確保 diosamsgbufalloc()
3. 44-55 行目 応答電文の作成
4. 57-64 行目 クライアントへ処理結果を送信 diosasendtx()

(b)　imhash.c

ユーザ情報を格納するパーティションを決定するためのハッシュ値を計算します。

DIOSA/XTP は、環境定義 IMENV 節を参照して、この関数が返したハッシュ値が属する MAPID を決定します。

1. 6-7 行目 ユーザ名の先頭が小文字の場合は大文字に変換しハッシュ値とする

2. 8-9 行目 ユーザ名の先頭が大文字の場合はそのままハッシュ値とする

(3)　　共通

(a)　message.h

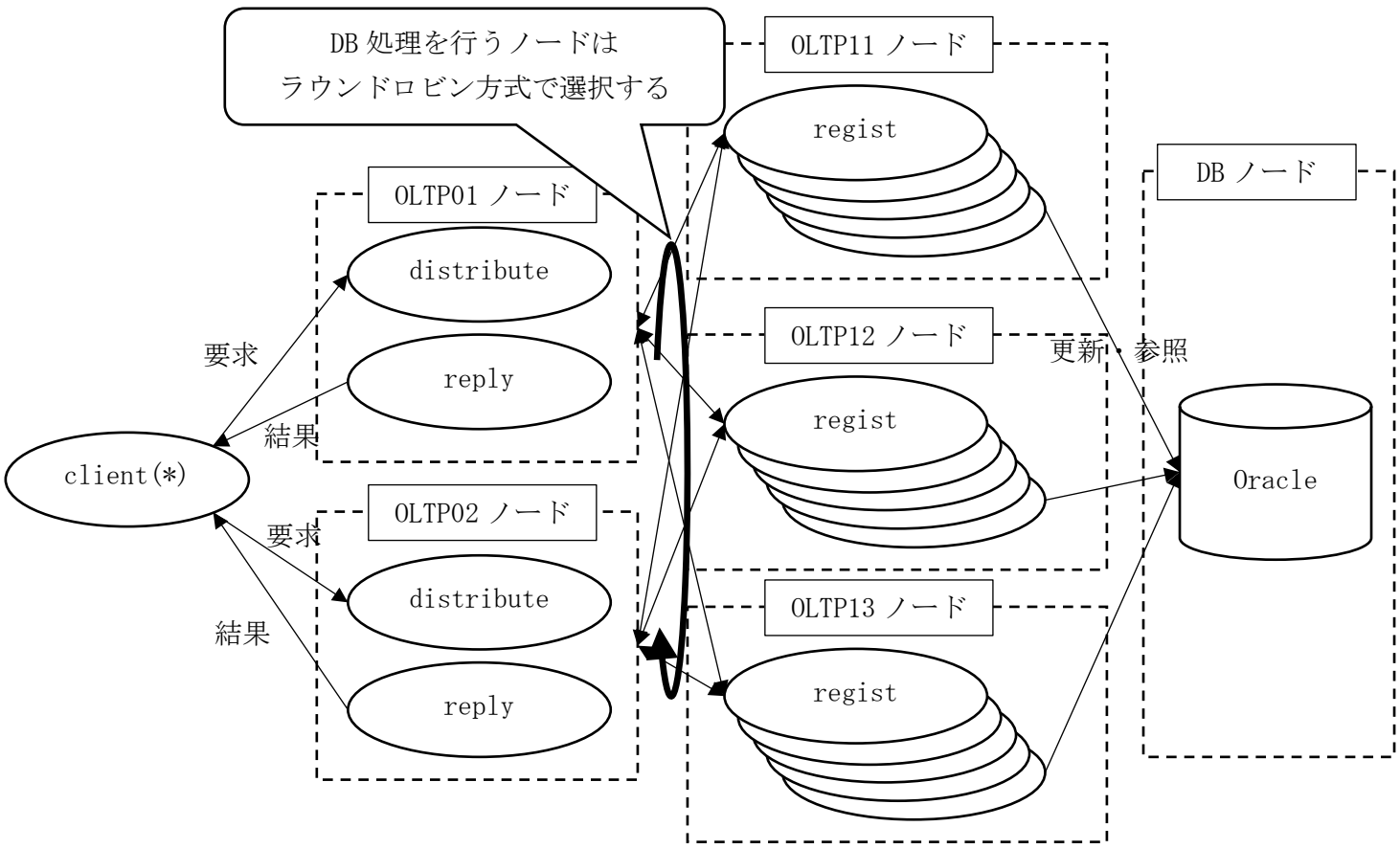
サーバ/クライアント間で送受信する電文形式(構造体)を定義します。

E.4　sample4 データベースアクセスアプリケーション Oracle 版 複数ノード構成

ユーザ ID とユーザ情報の登録・更新・削除・照会・全ユーザ照会を行います。データの格納先として OracleDB を利用します。

サンプル 2 に対して、クライアントとの送受信を行うノードと、DB アクセスを行うノードを分けることで、より多くの処理を行えるようにします。クライアントからの要求を受けた distribute から regsit などの処理を呼び出す際、実行するノードはラウンドロビン方式で選択します。

アプリケーションとノードの構成は以下の図のとおりです。



(*)サンプル構成の都合上、クライアントアプリケーションは OLTP ノードに格納していますが、
他のノードでも動作します。

E.4.1　使用方法

(1) sample4 配下をコピーする

すべてノードので実施します。NFS などによりコピー先を共有しても問題ありません。

```
% cp -r /opt/diosa_xtp/samples/sample4 ~/
```

(2) config.base を編集する

```
% cd ~/sample4
% vi config.base
```

項目	説明	例
WORK_DIR	sample1 のコピー先を指定します。	/home/user1/sample4
IPCKEY	サンプルが使用する IPC リーソスのキー上位 2 バイトを指定します。	FFFF

TPMONITOR_PREFIX	TPBASE を識別する文字列の先頭 2 文字を指定します。	xx
IP_ADDRESS01	アプリケーションが動くサーバの IP アドレスを指定します。	192.168.16.1
IP_ADDRESS02	アプリケーションが動くサーバの IP アドレスを指定します。	192.168.16.2
IP_ADDRESS11	アプリケーションが動くサーバの IP アドレスを指定します。	192.168.16.3
IP_ADDRESS12	アプリケーションが動くサーバの IP アドレスを指定します。	192.168.16.4
IP_ADDRESS13	アプリケーションが動くサーバの IP アドレスを指定します。	192.168.16.5
IP_ADDRESS101	Oracle が動作するサーバの IP アドレスを指定します。	192.168.16.6
PORT	サンプルが使用するポート番号を指定します。指定されたポート番号から 50 のポートを使用します。	32768
ORACLE_HOME	Oracle がインストールされたディレクトリを指定します。	/u01/app/oracle/product/12.1/dbhome_1
DB_NAME	Oracle へアクセスするためにネット・サービス名を指定します。	sampledb
DB_USER	Oralce へアクセスするためのユーザ名を指定します。	user
DB_PASSWORD	Oralce へアクセスするためのパスワードを指定します。	passwd

(3) 設定ファイルを作成する

```
% make config
```

(4) 環境変数の反映

```
% cd lsys4.oltpxx (xx は 01,02,11,12,13) または lsys4.db01
% . ./test.env
```

(5) アプリケーションの作成　設定ファイルの反映

```
% make
```

(6) 起動

DB ノード -> OLTP ノードの順に起動します。

A)　DB ノード起動

```
% distart_DB -c
```

B)　OLTP ノード 起動

```
% distart_OLTP -c
```

(7) ノード間パスの確認

OLTP01 OLTP02 で行う

STATUS 項が「OPEN」となっていることを確認します。

```
% dinodepathref
+====== T-PATH STATUS DISPLAY ======+  2017/08/03 12:59:37
LSNAME    = lsys4
LNODENAME = lsys4oltp01
SOURCE___ DESTINATION____________________
TPM______ TYPE LNODENAME______ TPM______ STATUS__
g4ls4o01  OLTP lsys4oltp02     g4ls4o02  OPEN
          OLTP lsys4oltp11     g4ls4o11  OPEN
          OLTP lsys4oltp12     g4ls4o12  OPEN
          OLTP lsys4oltp13     g4ls4o13  OPEN
+====== END OF DISPLAY  ======+
```

(8) アプリケーション呼び出し いずれかのノードで行う

```
% ./app/cl inquiryall
Result:0
NumOfRecord:0

% ./app/cl regist NEC "Nippon Denki"
Result:0
NumOfRecord:1
"NEC" "Nippon Denki"

% ./app/cl inquiryall
Result:0
NumOfRecord:1
"NEC" "Nippon Denki"

% ./app/cl regist DIOSA "Distributed Integrated Operating System for Applications"
Result:0
NumOfRecord:1
"DIOSA" "Distributed Integrated Operating System for Applications"

% ./app/cl inquiryall
Result:0
NumOfRecord:2
"NEC" "Nippon Denki"
"DIOSA" "Distributed Integrated Operating System for Applications"
```

(9) 停止

OLTP ノード -> DB ノードの順に停止します。

A) OLTP ノード 停止

% distop_OLTP -f

B) DB ノード 停止

% distop_DB -f

E.4.2 ディレクトリ構成

sample4/lsys4.oltpxx

app	アプリケーション
bin	起動・停止スクリプト
diosa	DIOSA/XTP の環境定義
log	ログファイル
tmp	一時ファイル
tpbase	TPBASE の環境定義

sample4/lsys4.db01

app	アプリケーション
bin	起動・停止スクリプト
diosa	DIOSA/XTP の環境定義
log	ログファイル
tmp	一時ファイル

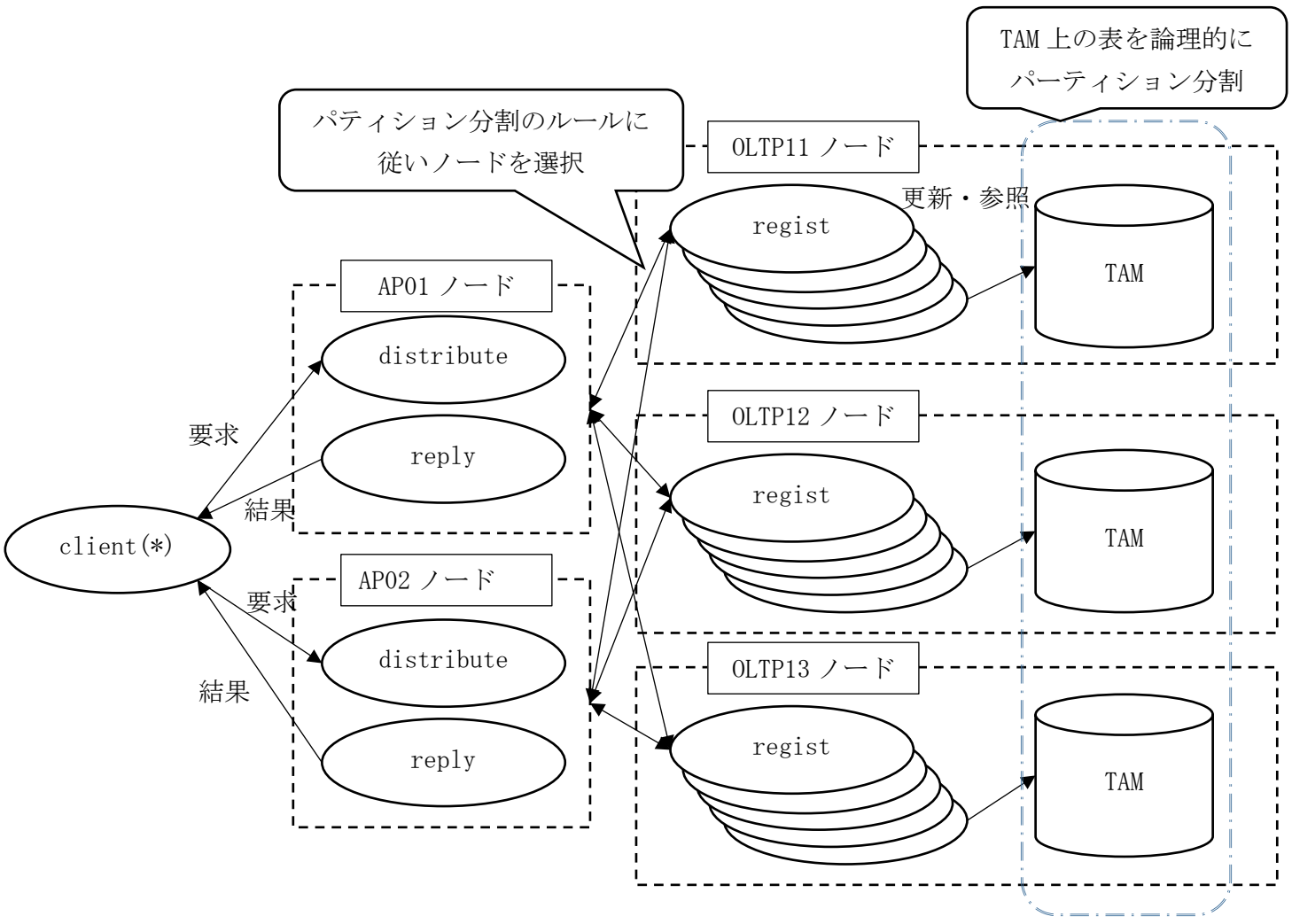
E.4.3 アプリケーションの説明

アプリケーションはサンプル 2 と同じです。サンプル 2 の説明を参照してください。

E.5　sample5 データベースアクセスアプリケーション TAM 版 複数ノード構成

サンプル 3 に対して、クライアントとの送受信を行うノードと、TAM アクセスを行うノードを分けることで、より多くの処理を行えるようにします。クライアントからの要求を受けた distribute から regsit などの処理を呼び出す際、実行するノードは対象のデータが配置されるノードをルールに従い選択します。

アプリケーションとノードの構成は以下の図のとおりです。



(*)サンプル構成の都合上、クライアントアプリケーションは OLTP ノードに格納していますが、
他のノードでも動作します。

E.5.1　使用方法

(1)　sample5 配下をコピーする

すべてノードので実施します。NFS などによりコピー先を共有しても問題ありません。

```
% cp -r /opt/diosa_xtp/samples/sample5 ~/
```

(2) config.base を編集する

```
% cd ~/sample5
% vi config.base
```

項目	説明	例
WORK_DIR	sample1 のコピー先を指定します。	/home/user1/sample5
IPCKEY	サンプルが使用する IPC リーソスのキー上位 2 バイトを指定します。	FFFF

TPMONITOR_PREFIX	TPBASE を識別する文字列の先頭 2 文字を指定します。	xx
NODE01	アプリケーションが動くサーバ名を指定します。	server1
IP_ADDRESS01	アプリケーションが動くサーバの IP アドレスを指定します。	192.168.16.1
NODE02	アプリケーションが動くサーバ名を指定します。	server1
IP_ADDRESS02	アプリケーションが動くサーバの IP アドレスを指定します。	192.168.16.2
NODE11	アプリケーションが動くサーバ名を指定します。	server1
IP_ADDRESS11	アプリケーションが動くサーバの IP アドレスを指定します。	192.168.16.3
NODE12	アプリケーションが動くサーバ名を指定します。	server1
IP_ADDRESS12	アプリケーションが動くサーバの IP アドレスを指定します。	192.168.16.4
NODE13	アプリケーションが動くサーバ名を指定します。	server1
IP_ADDRESS13	アプリケーションが動くサーバの IP アドレスを指定します。	192.168.16.5
PORT	サンプルが使用するポート番号を指定します。指定されたポート番号から 50 のポートを使用します。	32768

(3) 設定ファイルを作成する

```
% make config
```

(4) 環境変数の反映

全ノードで行う

```
% cd lsys5.apxx (xx は 01,02) または lsys5.oltpxx(xx は 11,12,13)
% . ./test.env
```

(5) アプリケーションの作成　設定ファイルの反映

```
% make
```

(6) TAM の設定

OLTP11、OLTP12、OLTP13 で行う。NFS で共有している場合はいずれか 1 台で実施する。

```
% cd ../lsys5.tam
```

TAM 環境定義のチェック

```
% make check
```

TMA 環境定義の反映

```
% make update
```

テーブルの作成

```
% make create
```

(7) 起動

OLTP ノード -> AP ノードの順に起動します。

A) OLTP ノード起動

% distart_OLTP -c

IM サーバ(スレーブ)の起動

% diimctrl -b

エラーとなったら数秒後に再度実行する。

IM 起動確認

STATUS 項が NORMAL となっていることを確認する

% diimref -M

```
+====== IN-MEMORY SERVER CONTROL INFORMATION DISPLAY ======+  2017/08/03 13:52:58
+------ MAP INFORMATION ------+
                                        -----------AT-----------  -----------TT-----------
     MAPID  STATUS        REPGRP  TYPE    SMQID   MSGQNUM  OVERFLOW  SMQID   MSGQNUM  OVERFLOW  MASTER      BLOCK
         1  NORMAL            1  MASTER      5         0         0      4         0         0  NORMAL/UPD  OFF
         2  NORMAL            2  SLAVE       8         0         0      7         0         0  NORMAL/UPD  OFF
         3  NORMAL            3  SLAVE      10         0         0      9         0         0  NORMAL/UPD  OFF
+====== END OF DISPLAY ======+
```

B) AP ノード起動

% distart_AP -c

IM 連携確認

STATUS 項が NORMAL となっていることを確認する

% diimref -N

```
+====== IN-MEMORY SERVER CONTROL INFORMATION DISPLAY ======+  2017/08/03 13:54:53
+------ NODE INFORMATION DISPLAY ------+
 LNODE           STATUS     BRIDGE-SERVER
 lsys5oltp11     NORMAL     NORMAL
 lsys5oltp12     NORMAL     NORMAL
 lsys5oltp13     NORMAL     NORMAL
+====== END OF DISPLAY ======+
```

% diimref -R

```
+====== IN-MEMORY SERVER CONTROL INFORMATION DISPLAY ======+  2017/08/03 13:56:01
+------ REPLICATION-GROUP INFORMATION DISPLAY ------+
    REPGRP  STATUS
         1  NORMAL
         2  NORMAL
         3  NORMAL
+====== END OF DISPLAY ======+
```

(8) アプリケーション呼び出し

いずれかのノードで行う

```
% ./app/cl inquiryall
Result:0
NumOfRecord:0

% ./app/cl regist NEC "Nippon Denki"
Result:0
NumOfRecord:1
"NEC" "Nippon Denki"

% ./app/cl inquiryall
Result:0
NumOfRecord:1
"NEC" "Nippon Denki"

% ./app/cl regist DIOSA "Distributed Integrated Operating System for Applications"
Result:0
NumOfRecord:1
"DIOSA" "Distributed Integrated Operating System for Applications"

% ./app/cl inquiryall
Result:0
NumOfRecord:2
"DIOSA" "Distributed Integrated Operating System for Applications"
"NEC" "Nippon Denki"
```

(9) 停止

AP ノード -> OLTP ノードの順に停止する

A) AP ノード停止

```
% distop_AP -f
```

B) OLTP ノード停止

```
% distop_OLTP -f
```

E.5.2 ディレクトリ構成

sample5/lsys5.oltpxx sample5/lsys5.apxx

app	アプリケーション
bin	起動・停止スクリプト

diosa	DIOSA/XTP の環境定義
log	ログファイル
tmp	一時ファイル
tpbase	TPBASE の環境定義

sample5/lsys5.tam

conf	TAM の環境定義
tablefile	TAM テーブルイメージ
var/lsys5oltp11	OLTP11 用の TAM ワークディレクトリ
var/lsys5oltp12	OLTP12 用の TAM ワークディレクトリ
var/lsys5oltp13	OLTP13 用の TAM ワークディレクトリ

E.5.3 アプリケーションの説明

アプリケーションはサンプル 3 と同じです。サンプル 3 の説明を参照してください。

DIOSA/XTP V2.1
導入の手引

2019年 9月 5版

日本電気株式会社
東京都港区芝五丁目7番1号
TEL (03)3454－1111(大代表)