DIOSA/XTP V3.1

データストア 利用の手引

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 データストアの利用の手引です。

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。

2022 年 12 月 第 2 版

本書の関連説明書としては次のものがあります。

- DIOSA/XTP 導入の手引き
- 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

DIOSA/XTP データストア基盤は、ユーザプログラムから要求のあったデータを大量、かつ高速に保証型転送/処理するシステムを容易に構築することを目的としています。

DIOSA/XTP データストア基盤は以下の特長があります。

- リアルタイムトランザクションと非同期に実行可能なデータ処理を、異なる論理システムに転送して転送先で実行したり、同一論理システム内で遅延型トランザクションとして実行したりすることができるため、リアルタイムトランザクションのレスポンスが向上します。
- リアルタイムトランザクションで要求を受け付けたデータは、要求順に、確実に処理されることが保証されるため、転送先論理システムの運用状態や、遅延型トランザクションの処理状態などを意識せずにデータ処理要求を行うことができます。
- 転送先論理システムとの通信管理をデータストア基盤が行うため、ユーザプログラムは送信経路や送受信制御などを意識する必要がありません。また、データ転送時に発生する欠送/重送に対応し、データの順序性を保証します。
- 転送先となった論理システムから、さらに別の論理システムへ転送をおこなって複数の論理システム上で遅延型トランザクションを実行させたり、1 つの論理システム上で同一データに対して複数の遅延型トランザクションを実行させたりといった、様々なシステム構成にも対応可能です。
- 遅延型トランザクション処理が異常終了となっても、データ再処理を容易に行うことができます。
- データ圧縮、データ分割をおこなうことで、リソースの消費を抑えることができます。また、データサイズを縮小することで、ファイルアクセス量やデータ転送量も削減でき、効率良く転送やデータ処理をおこなうことができます。

1.2　位置づけ

DIOSA/XTP データストア基盤は、DIOSA/XTP の 1 つの有償プログラムプロダクト(PP)であり、DIOSA/XTP の基本機能、メモリキャッシュを利用して、データ転送/遅延型トランザクション処理環境を提供します。

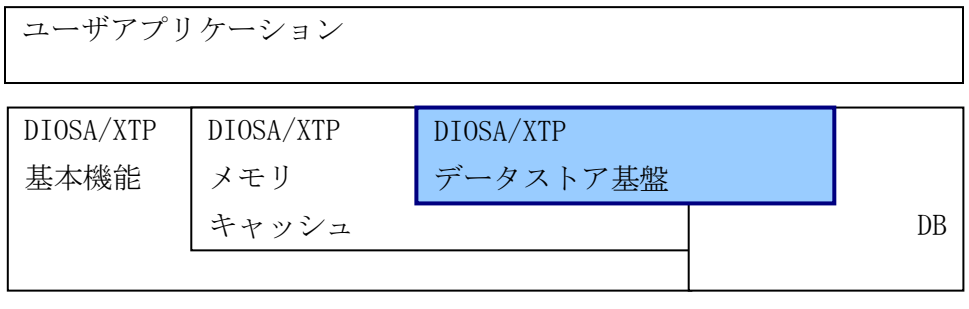


図 1-1 DIOSA/XTP データストア基盤の位置付け

1.3　諸概念

DIOSA/XTP データストア基盤の諸概念について説明します。

- 遅延型トランザクション
- ログデータ
- プールファイル
- スワップ
- スーパーストリーム
- ユニット
- ディビジョン

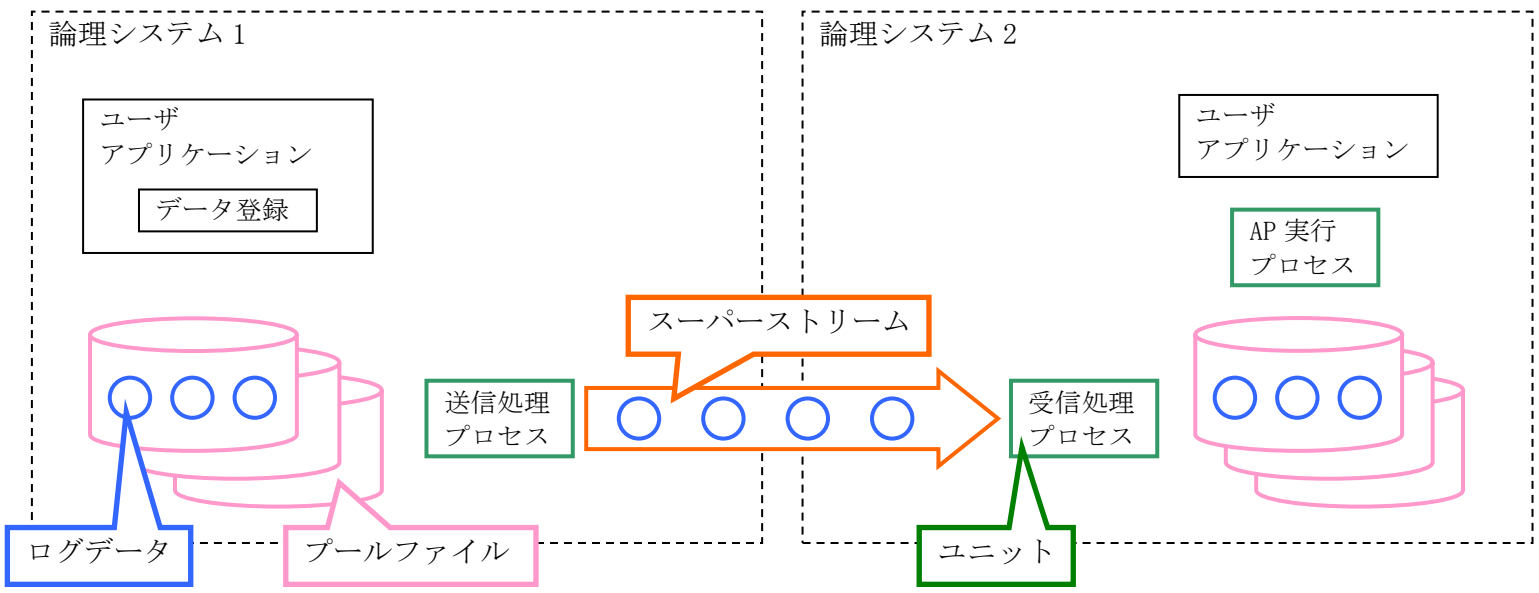


図 1-2 DIOSA/XTP データストア基盤 諸概念

(1)　遅延型トランザクション

遅延型トランザクションとは、リアルタイムトランザクションからユーザデータをデータベースに格納し、格納したユーザデータを処理するユーザアプリケーションをリアルタイムトランザクションと非同期に実行するトランザクション処理方式です。

ユーザアプリケーションを実行する方式としては、ユーザデータを格納した論理システム上で非同期に実行する方式と、ユーザデータを他の論理システムに転送して、転送先論理システム上で実行する方式があります。この方式は、即時性を要求されないが、高速/確実におこないたいデータ処理に有効です。

遅延型トランザクションは、リアルタイムトランザクションと、バッチ処理の中間に位置付けられ、それぞれに対して以下のようなメリットがあります。

- リアルタイム処理は、システムの運用状況/負荷状況の影響を受けますが、遅延型トランザクションは、システムの運用状況/負荷状況の影響を受けずにユーザプログラムからのデータ処理要求を受け付ける事ができます。
- バッチ処理は、リアルタイムトランザクションと、データの移動や処理が切り離されてしまいますが、遅延型トランザクションは、リアルタイムトランザクションに連動した処理として、システムを構築することができます。

(2)　ログデータ

遅延型トランザクション処理を要求されたユーザデータのことをログデータと呼びます。

個々のログデータは、後述するスーパーストリームごとに、ディビジョン ID、ログデータ通番といった制御情報が割り当てられます。遅延型トランザクション処理では、これらの制御情報を利用して、連続性や順序性、また一意性を保証します。

(3)　プールファイル

ユーザプログラムから処理要求されたログデータや、他論理システムからデータ転送されてきたログデータはデータベースに格納しますが、その格納先をプールファイルと呼びます。プールファイルはログデータの転送経路となる論理システム上に必ず存在する必要があります。

プールファイルは後述するスーパーストリームごとに存在し、スーパーストリームごとに定義されるログデータ格納先に合わせて、メモリキャッシュまたはDBに配置されます。また、スーパーストリームごとに、複数のデータ領域から構成されていて、個々のデータ領域をスタックファイルと呼びます。

(4)　スワップ

プールファイルへのログデータ格納は、格納先のスタックファイルが一定のサイズに到達すると、次のスタックファイルに出力先を変更します。このように格納先のスタックファイルを切り替えることをスワップと呼びます。

また、格納されたログデータを処理する際も、スタックファイル内の全てのログデータが処理されると次のスタックファイルに移動して処理を継続する制御をおこなっています。

データストア基盤では、データ処理状況を監視し、全ての処理が完了したスタックファイルを初期化して再利用することで、一定数のスタックファイルをサイクリックに使用します。

(5)　スーパーストリーム

スーパーストリームとはログデータが転送される経路、およびログデータの順序性を保証する単位となります。

ユーザプログラムが遅延型トランザクション処理要求時にストリームを指定することで、そのログデータに対する処理内容が決定されます。この時、ログデータの属性であるディビジョン IDとログデータ通番が付与されます。

ログデータは、スーパーストリームに対して定義された処理経路に従って遅延型トランザクション処理されます。

(6)　ユニット

ユニットとはログデータの転送経路、データ処理形態を決定する機能部品です。

スーパーストリームに対して 1 つ以上のユニットを定義することで、さまざまな遅延型トランザクション処理を実現することが可能です。

ユニットには、センダユニット、レシーバユニット、ログリーダユニットの 3 つのユニットが存在します。

- センダユニット

  プールファイルに格納されたログデータを別論理システムのプールファイルに転送します。

- レシーバユニット

  別論理システムのセンダユニットから転送されてきたログデータをプールファイルに格納します。

- ログリーダユニット

  プールファイルに格納されたログデータを読み込んで、ユーザアプリケーションを呼び出します。

環境定義にはユニットグループという概念がありますが、これはスーパーストリームに対して定義するユニットの構成をパターン化して、定義量を削減するためのもので、定義上にのみ存在する概念です。

(7)　ディビジョン

ディビジョンとは、連続して発生するログデータを時系列で仕切る概念です。具体的には、24 時間運転などで、本日分データと翌日分データの区別をするためなどに用います。

ディビジョンはディビジョン ID というログデータ通番とは異なる連番によって識別管理されます。ディビジョン ID は、後述するディビジョン切り替えのオペレーションを実行するごとに 1 ずつ増加します。次のディビジョン切り替えまでに登録されるログデータには、全て同一のディビジョン ID が割り当てられ、ログデータ通番のみが登録ごとに増加していきます。

ログデータのディビジョンによる仕切りは次のように行います。ログデータの発生元論理システムにおいて現ディビジョンのログデータの発生が終了したら、オペレーションによってディビジョンを切り替えます。このオペレーションをディビジョン切り替えと言います。ディビジョン切り替え後に発生するログデータには、次のディビジョン ID が付与され、ログデータ通番は新たに 1 から割り当てられます。

ディビジョンは運用面では以下の意味を持ちます。

データストア基盤の各機能は、処理中のディビジョン内の最終ログデータを常に意識しており、最終ログデータの処理が完了すると、自動的に処理を停止します。

このため、ディビジョンのデータ処理完了は、ログデータ発生元論理システムにディビジョン切り替えのオペレーションを行うだけで、全て完了します。

次ディビジョンのログデータ処理を開始する場合には各機能の開始オペレーションを実施する必要があります。

1.4　機能構成

データストア基盤は、以下の機能より構成されます。

また、下記以外にもいくつかの運用支援機能が存在します。

- ログデータ転送制御

  センダユニット、レシーバユニット、通信制御機能から構成され、ログデータの転送を行います。

- ログデータ実行制御

  ログリーダユニットがログデータを処理するユーザプログラムの制御を行います。

- プールファイル制御

  プールファイルへの入出力制御を行います。

下記は 2 つの論理システム上で登録されたログデータを実行する場合の例ですが、ログデータ転送制御、ログデータ実行制御を組み合わせることで、さまざまな構成でシステムを構築することができます。

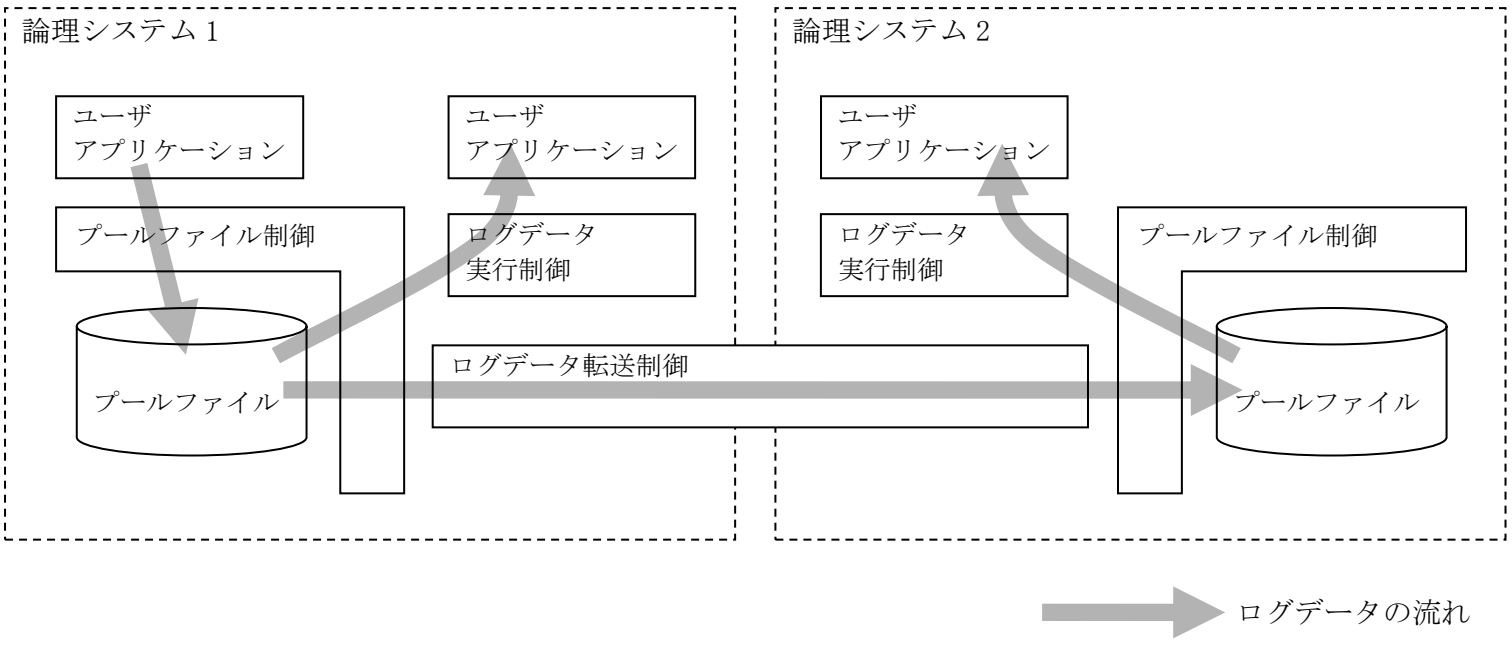


図 1-3 DIOSA/XTP データストア基盤の機能構成

第2章　機能

本章では、データストア基盤が提供する各機能内容について記述します。

2.1　ログデータ転送制御

(1)　概要

ログデータ転送制御は、センダ機能、レシーバ機能、通信制御機能により構成され、転送元論理システムから転送先論理システムへのログデータの転送を行います。

センダは、プールファイルからログデータを読み込み、DIOSA 環境定義に指定されたサイズ単位にブロッキングしたログデータ電文を、通信制御機能を用いて転送先論理システムへ送信します。転送先論理システムでは、レシーバがログデータ電文を受信し、ログデータにデブロッキングしてプールファイルへ書き込みます。このとき転送元と転送先のプールファイルは、同じ種類のデータベースである必要はありませんので、転送元で DB に格納したログデータを転送先でメモリキャッシュのプールファイルに格納するような構成も可能です。

センダ-レシーバ間の送受信処理は、スーパーストリーム単位で多重動作が可能です。また、環境定義によって、スーパーストリーム単位に宛先論理システムを変えたり、複数の論理システムをまたがってログデータを転送したりすることも可能です。

センダとレシーバが連携してデータ転送を行うことで、ログデータ電文の欠送、追い越しに対応し、ログデータの順序性、連続性を保証し、通信制御機能が通信資源利用の有効性、耐障害性を保証しています。

以降ではログデータ転送制御の提供する機能について説明します。

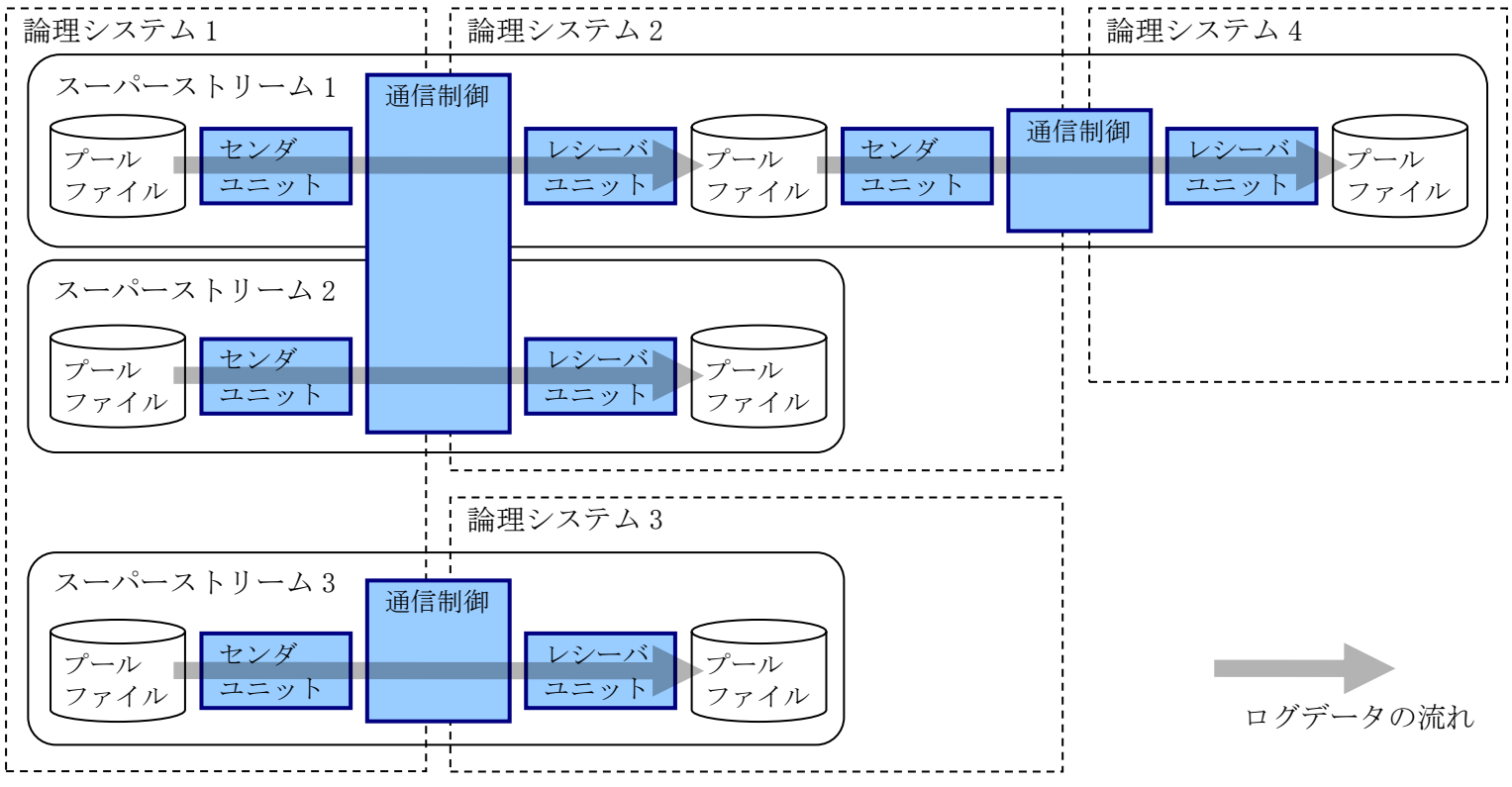


図 2-1 ログデータ転送制御

(2)　ペーシング制御機能

センダ-レシーバ間のログデータ転送は、ペーシング方式によるフロー制御を行っています。

センダは、一定量のログデータを送信するごとに、ペーシング要求電文を送信し、レシーバからの応答を受信するまで、ログデータ電文の送信を抑制します。これにより、障害や遅延が発生したときでも、通信経路上のログデータを一定サイズ以下に保つことができ、通信上の各バッファサイズやトラフィック量制御することができます。一回のペーシングで送信するデータ量は、DIOSA 環境定義で指定することができます。

(3)　ログデータ保証機能

ログデータ転送制御では、登録されたデータを確実に相手論理システムに転送するため、以下の制御を行っています。

- ログデータ再送制御

レシーバは、ログデータの欠送を検出すると、センダに再送要求を送信します。センダは再送要求を受信したら、送信済みのデータを再度送信します。

- ログデータ二重受信チェック

レシーバは、通信経路上で遅延していた等の理由で再送要求したログデータを二回受信した場合、後から受信したログデータを破棄します。

- ログデータ送達確認

プールファイルはサイクリックに使用するため、送信先論理システムのプールファイルに格納したログデータは送信元から削除します。センダは、後述するプールファイル制御と連携し、レシーバが受信済みと確認できたログデータのみをプールファイルから削除するため、通信経路障害などで再送要求を受信しても、ログデータを再送することが可能です。

(4)　ディビジョン終了機能

ログデータ転送制御では、スーパーストリームごとにディビジョン内の全てのログデータを転送すると、センダ-レシーバ間で送達確認を実施後、自動的にログデータ転送を停止します。

また、レシーバは転送先のプールファイルでもディビジョンを切り替えます。これによって、転送先のプールファイルを処理するセンダやログリーダも、同じようにディビジョン終了の制御を行うことができ、同一スーパーストリーム内の全ての処理をディビジョン終了状態にします。

次のディビジョンの転送処理は、データ転送開始のコマンドを転送元、転送先の論理システムで実行することで開始されます。

(5)　通信パス管理

センダ-レシーバのログデータ送受信は、スーパーストリームごとにデーモンを起動して並列処理を行います。デーモンの動作ノードは、複数のノードが存在する場合は各ノードに分散します。また、センダ-レシーバ間の通信経路となるノードが複数存在する場合、各ノードをラウンドロビンで選択し、システムの負荷を分散します。

(6)　障害対応機能

転送中に、センダ-レシーバの送受信を実行しているノードが障害となった場合、同一ノード種別の正常なノードが存在する場合は、動作ノードを自動的に移動し、中断点からログデータの転送を再開します。

通信経路となるノードや、ネットワークが障害となった場合、転送可能なパスが残っている限りは、選択可能なパスをラウンドロビンで選択しながら転送を継続します。

運用中に障害となったノードやネットワークが復旧した場合、処理ノードの移動や通信経路情報の更新を自動的に行います。

転送先論理システムとのパスが全て切断してしまった場合は転送を中断しますが、障害復旧時には自動的に転送を再開します。

2.2　ログデータ実行制御

(1)　概要

ログデータ実行制御は、ログリーダによる、プールファイルに格納されたログデータを実行するための機能です。

ログリーダはユーザアプリケーション、またはレシーバによってプールファイルに格納されたログデータを、ユーザアプリケーションが登録した順番に読み込み、ログデータごとにユーザアプリケーションを呼び出します。

ログリーダは、プールファイルに対して複数定義することも可能です。複数定義した場合、それぞれのログリーダが一回ずつログデータを実行するため、一つのログデータに対して複数の異なる処理を行うことが可能です。

ユーザアプリケーションが更新対象とするデータベースは、プールファイルと異なるデータベースでも構いませんし、複数のログリーダユニットを定義した場合に、各ユニットが異なるデータベースのユーザデータを処理対象にすることもできますので、メモリキャッシュのプールファイルを読み込んで、DB のユーザデータを更新したり、DB のプールファイルを読み込んで、異なるインスタンスの DB のユーザデータを更新するといった構成も可能です。

以降ではログデータ実行制御の提供する機能について説明します。

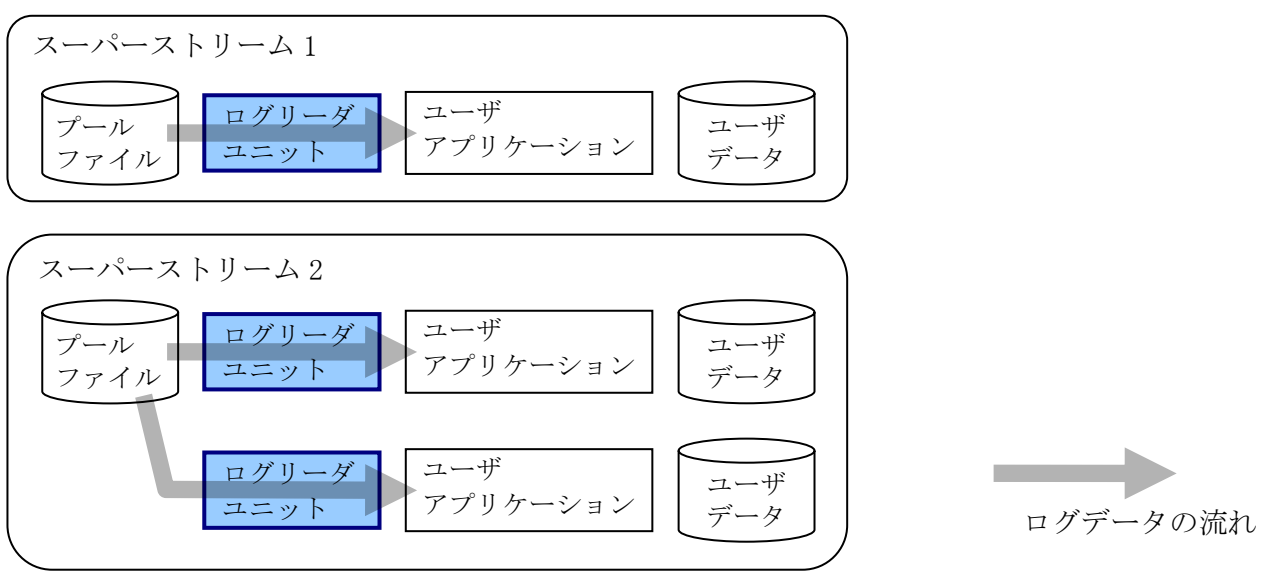


図 2-2 ログデータ実行制御

(2)　ロット化機能

ログリーダでは、複数のログデータ処理を 1 ロットとして扱い、1 ロットの処理が完了するごとにコミットを発行することで、データ処理を高速化します。

1 ロットで処理するログデータ数をコミット係数とよび、環境定義で指定することができます。

(3)　リトライ機能

ユーザアプリケーションでデッドロックや一時障害が発生した場合、ログデータ処理はロールバックされますが、自動的にリトライされます。リトライ回数は環境定義で指定することができます。

リトライオーバーした場合、ログリーダの処理は一時停止します。この場合通常はエラーとなったログデータから処理を再開しますが、原因となったログデータをスキップして処理を継続することも可能です。

(4)　ループ/ストール監視機能

ループ／ストール監視機能は、DIOSA の経過時間監視機能と連携し、長時間実行されているアプリケーションを監視します。ユーザアプリケーションの呼び出しから監視時間が経過すると、コンソールメッセージを表示するか、シグナルによって強制停止させるか、いずれかの動作を選択することができます。また、アプリケーションの CPU 時間も監視し、CPU 時間が監視時間を超えた場合、シグナルによって強制停止させることができます。

監視の実施有無や、監視時間は環境定義で指定することができます。

ある特定のアプリケーション処理だけを長時間動作させたい場合、経過時間監視機能のリセット処理を定期的に実行することで、監視対象のアプリケーションであっても、長時間動作させることが可能です。

(5)　ディビジョン終了機能

ログリーダが処理中ディビジョンの全てのログデータを処理すると、自動的にログデータ処理を停止します。

次のディビジョンの処理は、データ処理開始のコマンドを実行することで開始されます。

(6)　障害対応機能

ログデータの処理実行中のノードが障害となった場合、同一ノード種別の正常なノードが存在する場合は、動作ノードを自動的に移動し、中断点からログデータの処理を再開します。

運用中に障害となったノードが復旧した場合、処理ノードの移動を自動的に行います。

2.3　プールファイル制御

(1)　概要

プールファイル制御は、ログデータの登録、転送、実行のためのプールファイルアクセスを行う機能です。

プールファイルは一定数のスタックファイルをサイクリックに利用しますが、プールファイル制御機能が、スタックファイルの切り替えや、初期化処理を一元管理することで、ユーザアプリケーションや、ログデータ転送制御、ログデータ実行制御がスタックファイルの管理を意識せずにログデータにアクセスできる基盤を提供しています。

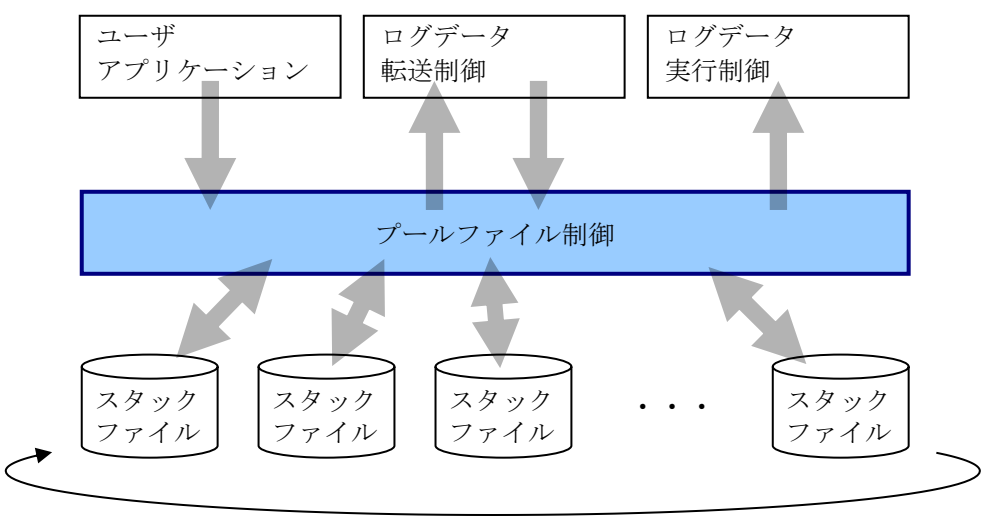


図 2-3 プールファイル制御

(2)　ログデータ登録

プールファイルへのログデータ登録は、ユーザアプリケーションからログデータ登録 API を呼び出しておこないます。登録したいデータの情報と、登録先のスーパーストリーム名を指定すると、ディビジョン ID、通番が割り当てられ、ログデータとしてプールファイルに登録されます。

ログデータの登録時、スーパーストリーム名としてエイリアス名を指定することも可能です。フロント-バックアップの構成をとるシステムにおいて、双方向のデータ転送をおこなう場合、それぞれ異なるスーパーストリーム名を指定する必要がありますが、同一のエイリアス名を付けておくことで、システムごとに異なるログデータ登録用プログラムを用意する必要がなくなります。

(3)　スタックファイルの再利用制御

プールファイルは、複数のスタックファイルで構成されます。また、個々のスタックファイルはサイズの上限(スワップサイズと呼びます)が決まっています。

プールファイル制御では、スタックファイルに書き込んだログデータのサイズがスワップサイズに到達すると、書き込むスタックファイルを移動します。また、書き込まれたログデータを処理するセンダおよびログリーダの処理状態を監視し、全ての処理が完了したスタックファイルは初期化してサイクリックに利用します。これにより、プールファイルのサイズを一定のサイズ未満に抑えながら連続運転が可能となります。

スタックファイルの初期化は、処理が完了したら即時実行することも可能ですが、削除するスタックファイル数を環境定義で指定可能で、処理済みになっても可能な限り保存しておき、書き込めるスタックファイルが少なくなってから削除するといった運用をすることも可能です。

スーパーストリームに割り当てるスタックファイルの数、スワップサイズ、削除スタック数は、スーパーストリームごとに環境定義で指定可能です。

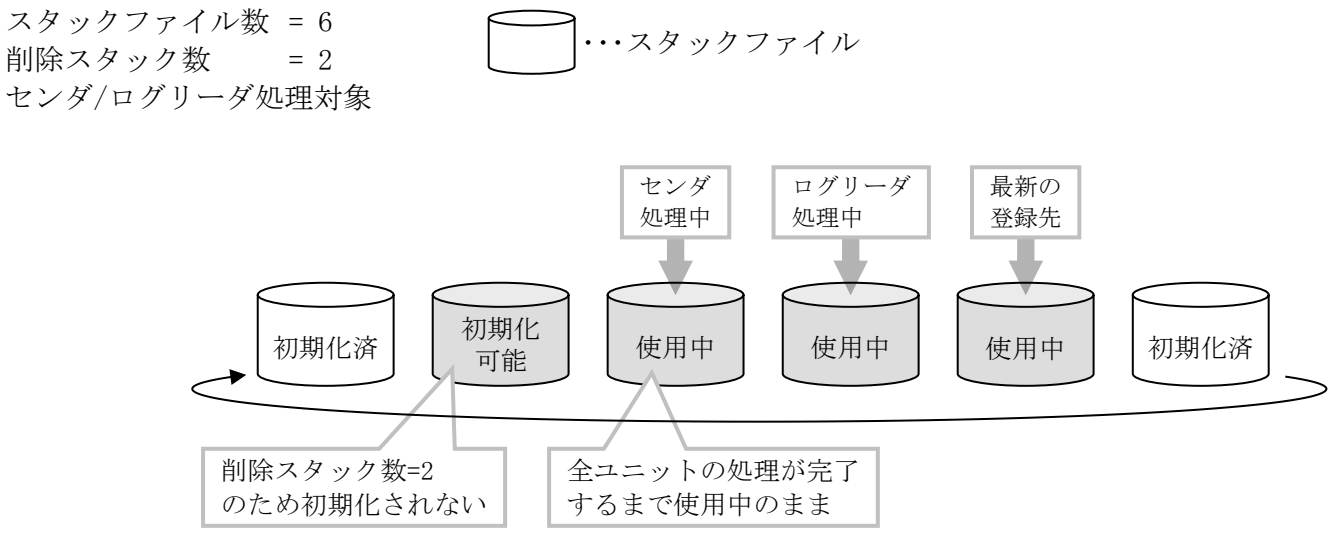


図 2-4 スタックファイルの再利用制御

(4) ログデータの圧縮/解凍、分割/結合の制御

ユーザアプリケーションが登録したログデータは、ログリーダが読み込んでユーザアプリケーションが参照するときには登録データと同じ内容であることが保証されていますが、プールファイルに保存されているときや、論理システム間の転送処理中は、データ圧縮や分割をおこない、データ格納用の領域や、一時バッファのサイズを抑制することが可能です。

データを圧縮するかどうかは、スーパーストリームごとや書き込み要求時に指定可能です。データの分割については、データ転送する電文サイズやプールファイルのテーブルサイズに従って、プールファイル制御で自動的におこないます。

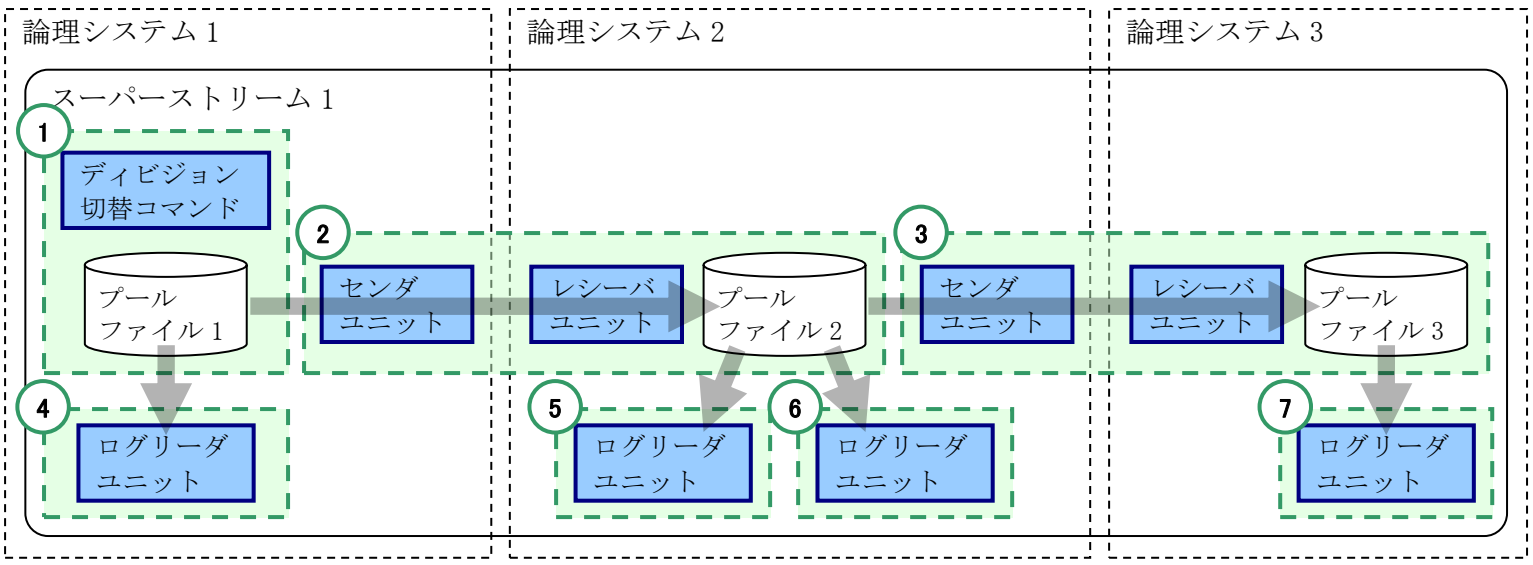
(5) ディビジョン管理

ディビジョン切り替えは利用者がコマンドを実行して行います。

コマンドが実行されると、それ以降に登録要求されたログデータに対しては、別のディビジョン ID を割り当て、ログデータ通番は 1 にリセットされます。

ログデータ登録時のディビジョンを切り替えると、プールファイル制御はログデータ転送制御、ログデータ実行制御と連携し、切り替え前のディビジョンのログデータを全て処理完了したら全体が一時停止するように制御します。

コマンド実行前ディビジョン ID=1 とした場合



① コマンド実行によりプールファイル 1 のディビジョンが 1 から 2 に切り替わります。
② プールファイル 1 のディビジョン 1 のログデータを全て転送すると、プールファイル 2 のディビジョンを 1 から 2 に切り替え、論理システム 1-論理システム 2 間のログデータ転送を停止します。
③ プールファイル 2 のディビジョン 1 のログデータを全て転送すると、プールファイル 3 のディビジョンを 1 から 2 に切り替え、論理システム 2-論理システム 3 間のログデータ転送を停止します。
④～⑦ 処理対象のプールファイルのディビジョン 1 のデータを全て処理したログリーダは、順次処理を停止します。

図 2-5 ディビジョン切り替え

2.4　その他機能

2.4.1　　無効化機能

無効化機能は、スーパーストリームやユニットを一時的に運用対象外にするための機能です。

(1)　スーパーストリーム無効化

スーパーストリームを無効化すると、そのスーパーストリームに対する一切の運用ができなくなります。

- ユーザアプリケーションからのログデータ登録
- センダ/レシーバユニットによるログデータ転送
- ログリーダユニットによるログデータ実行

フロント-バックアップ構成のようなシステムにおいて、フロントシステムがダウンするまで使用しないスーパーストリームがある場合などは、対象のスーパーストリームを無効化することで、誤ったデータ登録を抑止したり、不要なリソース消費を抑えることが可能となります。

スーパーストリームの無効化状態は環境定義で指定可能です。また、起動後にコマンドで有効/無効を切り替えることができます。

(2)　ユニット無効化

ユニットを無効化すると、そのユニットは運用対象外となり、ログデータ処理を停止します。

また、プールファイル制御では無効化したユニットの処理が完了していなくても、その他有効なユニットの処理が完了していれば、ログデータは処理済みと判断し、スタックファイルを初期化します。

プールファイルに、自論理システム内で処理を行うログリーダユニットと、他論理システムに転送するセンダユニットが定義されている場合に、例えば転送先論理システムがダウンしてしまうと、センダはログデータを処理できない状態となります。プールファイル制御では、全てのユニットがログデータ処理を完了しないと、スタックファイルを初期化できないため、転送先論理システムが復旧しない場合はプールファイルが満杯になり、ログデータの登録ができなくなります。

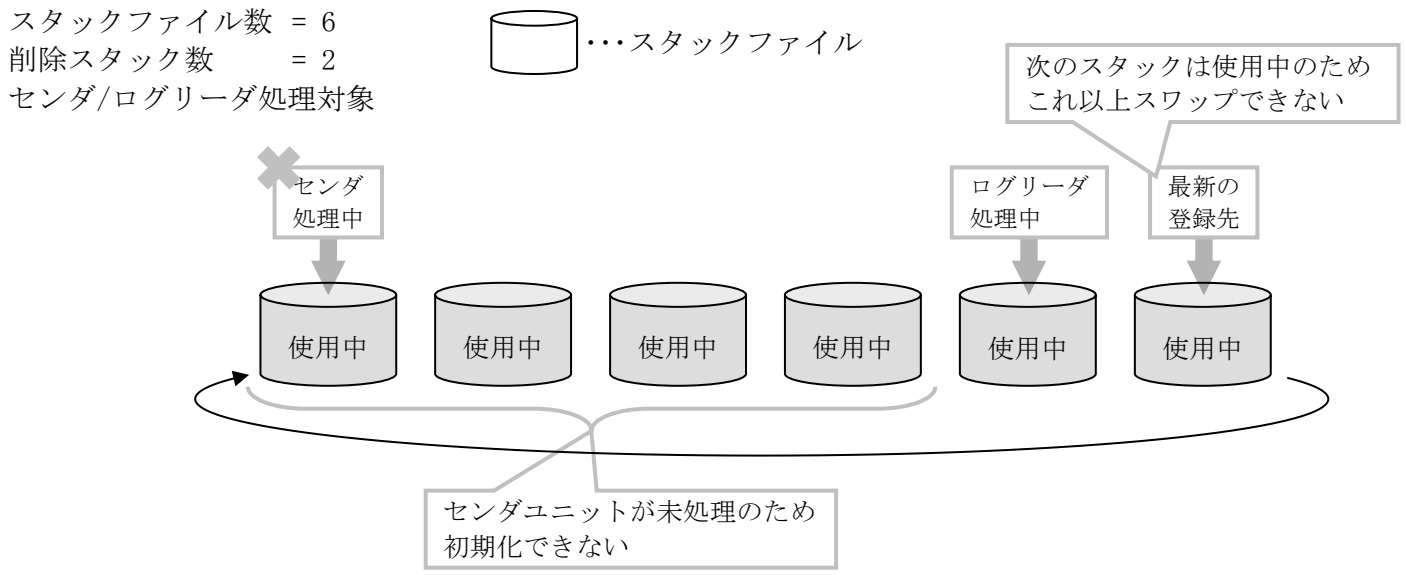


図 2-6 ユニット無効化(無効化しない場合)

このような場合に、センダユニットを無効化することで、ログデータ登録とログリーダによるデータ処理は正常運用を継続することができ、一部の障害が全体に波及することを防ぐことができます。

ユニットの無効化状態は環境定義で指定可能です。また、起動後にコマンドで有効/無効を切り替えることができます。

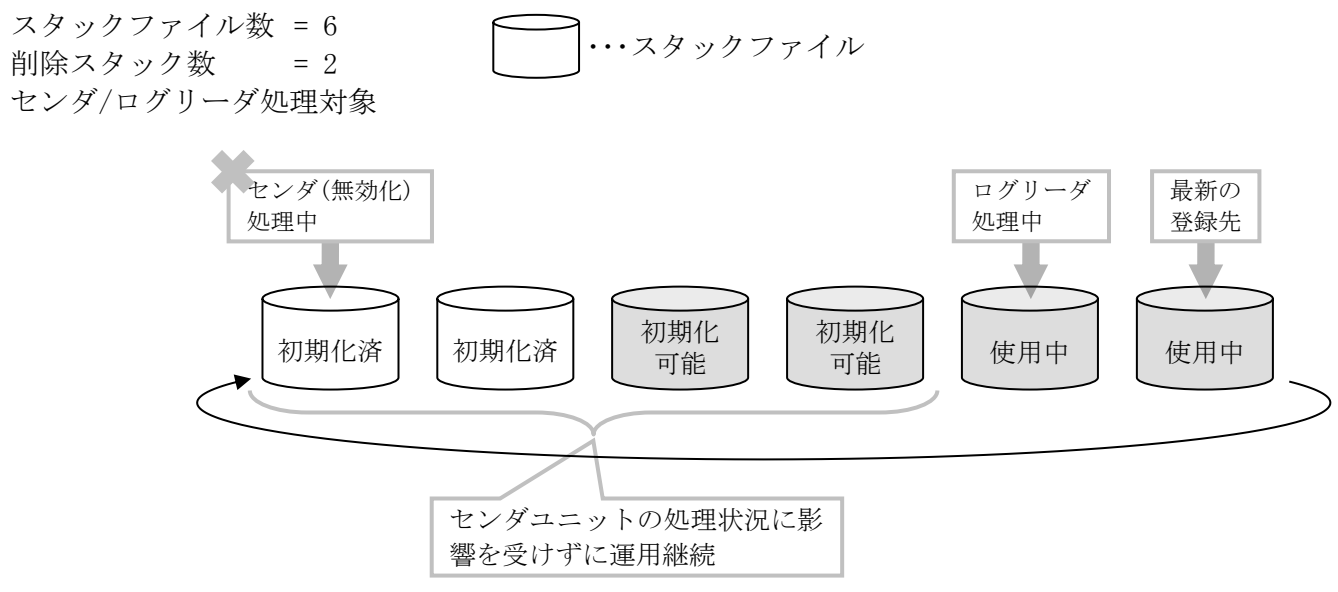


図 2-7 ユニット無効化(無効化した場合)

2.4.2 動作ノード管理機能

プールファイルに登録されたログデータを処理するノードは、スーパーストリームごとに論理システム内の特定のノードに決定されます。複数のスーパーストリームが存在する場合、処理するノードを分散することで、システムの負荷分散を実現します。

プールファイルがメモリキャッシュ上にある場合、スーパーストリームごとにプールファイルを配置する MAPID を定義します。DIOSA/XTP メモリキャッシュ機能によって、MAPID(レプリケーショングループ)ごとにマスタノードが決

定されるため、それに合わせてスーパーストリームの動作ノードも決定します。

(1)　ストリーム所在管理機能

ストリーム所在管理機能は、プールファイルがDB上にあるスーパーストリームの動作ノードを管理する機能です。動作ノードのノード種別は、APノードかOLTPノードを環境定義で指定します。

ストリーム所在管理機能は、指定されたノード種別の全ノードを監視し、各ノードで動作するスーパーストリーム数が均等になるように配置します。ノードダウンを検出した場合、該当ノードで動作していたスーパーストリームは、正常動作中のノードに分散するように再配置されます。

(a)　スーパーストリームグループ

ストリーム所在管理機能では、各ノードで動作するスーパーストリーム数が均等になるように振り分けをおこないますが、スーパーストリームグループを指定すると、同一のスーパーストリームグループに属するスーパーストリームが均等になるように振り分けをおこないます。処理形態の異なるスーパーストリーム群ごとに、異なるスーパーストリームグループを指定することで、より細かいノード振り分けが可能となります。

2.4.3　　メンテナンス機能

ログデータ転送制御、ログデータ実行制御、プールファイル制御の動作に必要な制御テーブルを生成、更新したり、動作パラメータを変更したりするための運用コマンドを提供しています。

(1)　定義生成機能

プールファイルやセンダ/レシーバ/ログリーダの動作に必要な制御表を、環境定義情報をもとに生成します。

(2)　定義変更機能

スーパーストリームやユニットの追加等、環境定義を変更した場合に、定義生成で生成した制御表を、変更後の環境定義情報に従って更新します。

(3)　動作変更機能

一部の動作パラメータの変更について、定義変更よりも簡単に一時変更することが可能です。

以下のパラメータが動作変更できます。

センダ	ログデータ電文送信遅延時間
	連続電文送信数
	ペーシングあたりの電文数
レシーバ	１コミットあたりの電文受信数
ログリーダ	コミット係数

変更した動作パラメータを、全て環境定義に定義されている値に戻すことも可能です。

第3章　アプリケーションの開発

3.1　ログデータ登録

ログデータの登録は、CO 制御、バッチ AP 制御、ユーザアプリケーションのいずれからも実行可能です。

ログデータの登録先スーパーストリームのプールファイルがメモリキャッシュ上にあるか、DB 上にあるかによって、必要な DB 接続の処理がそれぞれありますが、それらの詳細については各機能のマニュアルを参照してください。

3.1.1　コーディング例

ログデータ登録をおこなうプログラムのサンプルを以下に示します。

```
#include <diosa.h>

int PutLogData(){
    int             Ret;
    char            LogData[1000];
    t_diosa_dloguca  DlogUca;

    memset( LogData, 0, sizeof( LogData ) );
    memset( &DlogUca, 0, sizeof( t_diosa_dloguca ) );

    strcpy( DlogUca.SpstName, "SPST1" );       /* スーパーストリーム名 */
    DlogUca.TextLen = sizeof( LogData );       /* ログデータサイズ */
    DlogUca.CompressFlg = DIOSA_DATACOMP_ON;   /* 圧縮指定(圧縮する) */

    Ret = diosadpuglog( &DLogUca, LogData);
    if ( Ret != DIOSA_DONE ){
        printf( "diosadputlog error. return=%d, dstatus=%d¥n", Ret, DlogUca.Dstatus );
        return DIOSA_ERROR;
    }

    return DIOSA_DONE;
}
```

3.2　ログデータ実行

ログリーダのログデータ処理実行デーモンでは、以下の利用者プログラムを作成し、デーモンから呼び出させることができます。

- ログデータ実行メイン処理
- プロセス初期化処理
- プロセス終了処理
- トランザクション初期化処理
- トランザクション終了処理

ログデータ実行メイン処理は、作成必須ですが、それ以外の処理については、不要ならば作成しなくてもログデータ処理は可能です。各処理を呼び出すかどうかや、呼び出す関数名はログリーダユニット単位に環境定義で指定します。

ログデータ実行メイン処理、および各初期化/終了処理では、DIOSA/XTP の各種 API を利用して処理を実装可能です。各関数から呼び出し可能な API については、DIOSA/XTP API リファレンスの付録を参照してください。

ログリーダから呼び出される利用者プログラムが標準出力をおこなった場合、以下のファイルに出力されます。

出力先ディレクトリ‥‥‥‥‥‥ 環境変数(DIOSA_TMP)指定ディレクトリ/論理ノード名/log
標準出力ファイル名‥‥‥‥‥‥ didtlexecd.std
標準エラー出力ファイル名‥‥‥ didtlexecd.err

ログデータ実行メイン処理が返却する以下 4 つの戻り値でログリーダは以下の動作を行います。

・DIOSA_DONE(0)
ログデータ処理を正常終了し、次のログデータの処理へ進む。

・DIOSA_RETRY(5)
ロールバックを行い、リトライを行う。

・DIOSA_ETRAN(-19)
ロールバックを行い、ログリーダユニットの処理を停止する(ユニットステータスを INACT へ変更する)。

・DIOSA_EFATAL(-30)
ロールバックを行い、プロセス再起動後、ログリーダユニットの処理を停止する(ユニットステータスを INACT へ変更する)。

3.3　アプリケーションの生成

アプリケーションの生成について、データストア基盤固有の指定等はありません。
DIOSA/XTP 利用の手引きを参照して、アプリケーションを生成してください。

データ変換・通信オプションの更新ログによる非同期レプリケーションを行う場合は、データ変換・通信オプションのアプリケーションを使用するため生成する必要はありません。その場合に指定するアプリケーションについては「4.1.4　　ログデータ実行制御 (3) ユーザアプリケーション関数名」を参照してください。

第4章　システムの構築

4.1　環境設計

4.1.1　全体構成の設計

データストア基盤の環境構築する場合、まず遅延型トランザクション処理の形態や、プールファイルを配置するデータベースの種類等を決定し、転送先を含めてどのような処理システムを構築するのかを決定する必要があります。

(1)　プールファイル配置決定

ログデータを登録する論理システム、転送先の論理システムには必ずプールファイルを準備する必要があります。

スーパーストリームを複数定義して、データ処理を多重化したい場合は、その分のプールファイルが必要です。

またプールファイルは、ログデータ格納先をメモリキャッシュにするのか、DB にするのかをそれぞれに対して決定する必要があります。

下図は、論理システム 1→論理システム 2→論理システム 3 とログデータ転送する、2 多重で動作するスーパーストリーム 1、スーパーストリーム 2 と、論理システム 1→論理システム 3 とログデータ転送する、1 多重で動作するスーパーストリーム 3 を定義する例です。

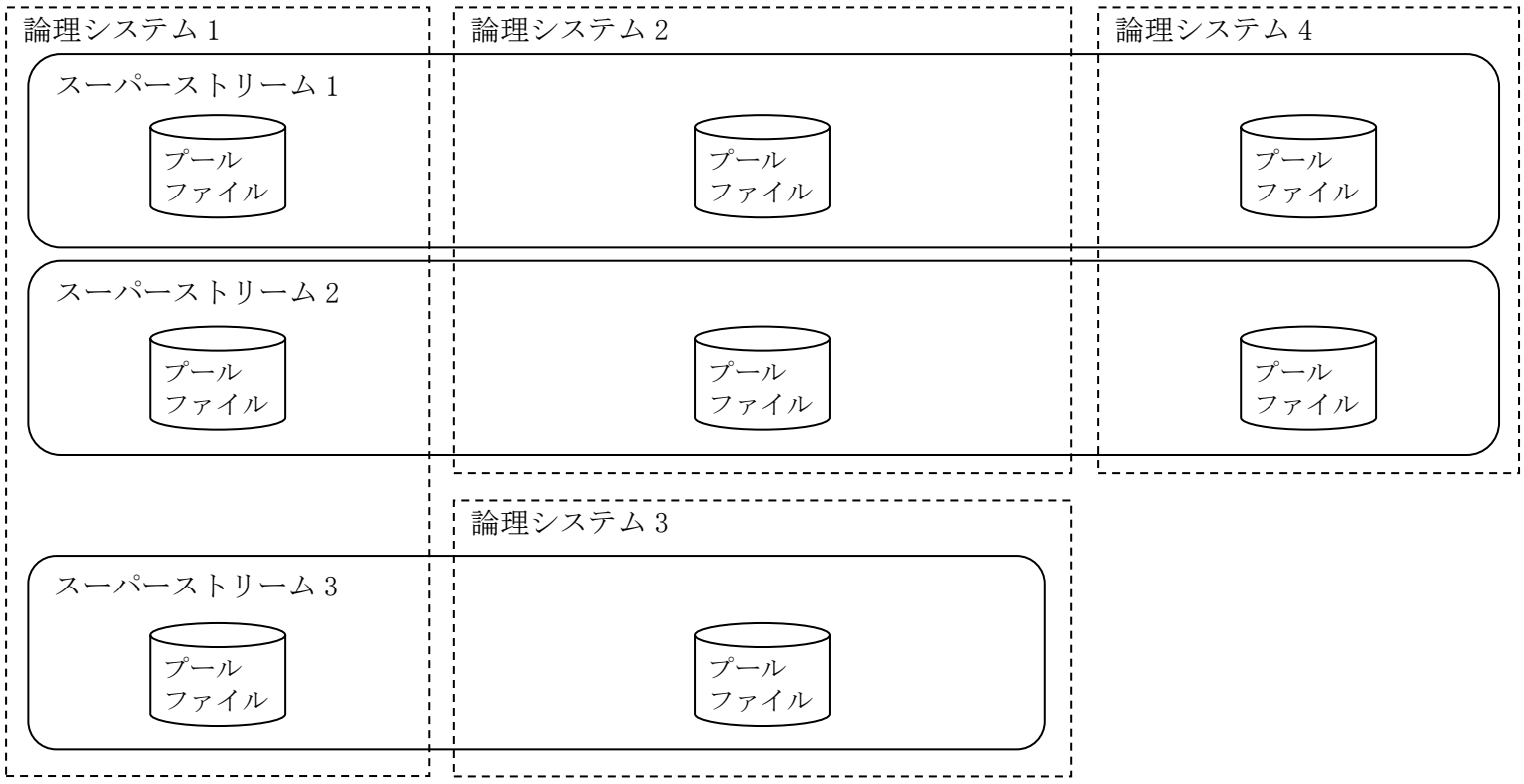


図 4-1 プールファイル配置決定

ログデータ格納先と、各処理が動作する論理ノード関係は以下の通りです。

	メモリキャッシュ	DB
ログデータ書き込み	OLTP ノード	AP ノード/OLTP ノード(※1)
センダ/レシーバ/ログリーダユニット	OLTP ノード	AP ノード/OLTP ノード(※2)

※1 DB アクセス可能であること(両方アクセス可能な場合、いずれのノード種別からも書き込み可能)

※2 どちらの論理ノード種別を動作ノードとするかは、環境定義で指定します。

(2) ユニット構成決定

ログデータを複数の論理システム上で転送しながら処理するためには、転送元にセンダユニット、転送先にレシーバユニットを必ず定義しなければなりません。また、ログデータを処理するためには、必ずログリーダユニットを定義しなければなりません。

下図は、スーパーストリーム 1、スーパーストリーム 2 は論理システム 2、論理システム 4 上でログデータ処理をおこない、スーパーストリーム 3 は論理システム 1 で 1 種類、論理システム 3 で 2 種類のログデータ処理をおこなう場合の例です。

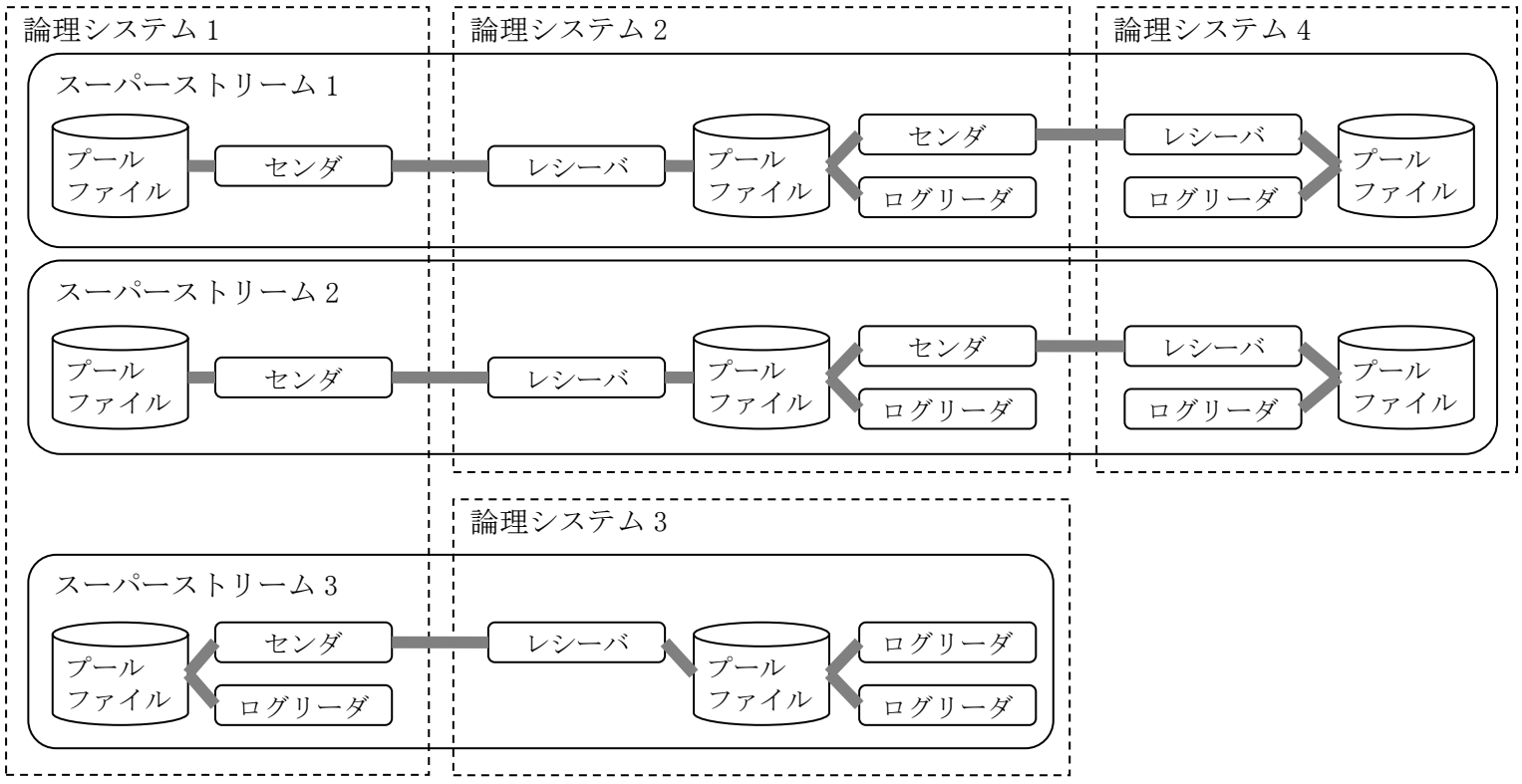


図 4-2 ユニット構成決定

(3) ユニット構成決定

構成の概要が決定したら、それぞれの構成に合わせてディレードの環境定義を記述します。

論理システムごとに自論理システム上で動作するスーパーストリーム、ユニットの定義のみを記述するため、ディレードの環境定義は、論理システムごとに異なる内容となります。

上記の例を記述する場合以下のようになります。(概要イメージです)

```
[論理システム 1]
$DELAYED
```

```
    %UNITGROUP
        NAME = UNITGROUP1
        POOLTYPE = WRITER
        %SNDUNIT
            NAME=SENDER
            DLSNAME=論理システム 2
        ;
    ;
    %SUPERSTREAM
        NAME = スーパーストリーム 1
        UNITGROUP = UNITGROUP1
    ;
    %SUPERSTREAM
        NAME = スーパーストリーム 2
        UNITGROUP = UNITGROUP1
    ;
    %UNITGROUP
        NAME = UNITGROUP2
        POOLTYPE = WRITER
        %SNDUNIT
            NAME=SENDER
            DLSNAME=論理システム 3
        ;
        %LRDUNIT
            NAME=LOGREADER
        ;
    ;
    %SUPERSTREAM
        NAME = スーパーストリーム 3
        UNITGROUP = UNITGROUP2
    ;
;
```

```
[論理システム 2]
$DELAYED
    %UNITGROUP
        NAME = UNITGROUP1
        POOLTYPE = RECEIVER
        %RCVUNIT
            NAME=RECEIVER
            DLSNAME=論理システム 1
        ;
        %SNDUNIT
            NAME=SENDER
            DLSNAME=論理システム 4
        ;
        %LRDUNIT
            NAME=LOGREADER
        ;
    ;
    %SUPERSTREAM
        NAME = スーパーストリーム 1
        UNITGROUP = UNITGROUP1
    ;
    %SUPERSTREAM
        NAME = スーパーストリーム 2
        UNITGROUP = UNITGROUP1
    ;
;
```

```
[論理システム 3]
$DELAYED
    %UNITGROUP
        NAME = UNITGROUP1
        POOLTYPE = RECEIVER
        %RCVUNIT
            NAME=RECEIVER
            DLSNAME=論理システム 1
        ;
        %LRDUNIT
            NAME=LOGREADER1
        ;
        %LRDUNIT
            NAME=LOGREADER2
        ;
    ;
    %SUPERSTREAM
        NAME = スーパーストリーム 3
        UNITGROUP = UNITGROUP1
    ;
;
```

```
[論理システム 4]
$DELAYED
    %UNITGROUP
        NAME = UNITGROUP1
        POOLTYPE = RECEIVER
        %RCVUNIT
            NAME=RECEIVER
            DLSNAME=論理システム 2
        ;
        %LRDUNIT
            NAME=LOGREADER
        ;
    ;
    %SUPERSTREAM
        NAME = スーパーストリーム 1
        UNITGROUP = UNITGROUP1
    ;
    %SUPERSTREAM
        NAME = スーパーストリーム 2
        UNITGROUP = UNITGROUP1
    ;
;
```

以降では各機能に関連する定義パラメータの詳細について説明します。

4.1.2　　プールファイル制御

プールファイルには、ユニットグループ配下のパラメータで定義する、プールファイル種別、およびログデータ格納先という必須パラメータがあります。

(1)　プールファイル種別

プールファイルにデータを格納する機能を指定します。

WRITER　　ユーザアプリケーションがログデータを登録します。

RECEIVER　　他論理システムから転送されたログデータをレシーバが登録します。

ログデータを登録する論理システムでは WRITER、他論理システムから転送されるログデータを処理する論理システムでは RECEIVER を指定してプールファイルを定義してください。

RECEIVER を指定した場合、RCVUNIT 項の定義が必須となります。

[パラメータ名(最優先パラメータ)]

$DELAYED-%UNITGROUP-POOLTYPE

(2)　ログデータ格納先

プールファイルを配置するデータベースを指定します。

IM　　メモリキャッシュ

DB　　Oracle または PostgreSQL

プールファイル種別が WRITER の場合、ユーザアプリケーションのデータ更新先に合わせてください。プールファイル種別が RECEIVER の場合、ログリーダ上で動作させたいユーザアプリケーションのデータ更新先がメモリキャッシュの場合は、ログデータ格納先には IM を指定してください。データ更新先が DB の場合、ログデータ格納先には IM/DB のいずれも指定可能です。

ログデータ格納先に DB を指定するプールファイルが存在する場合、ログデータの転送や実行をおこなうノード種別を、AP ノード/OLTP ノードのいずれかに決定し、%COMMON-DBLAYER パラメータに指定します。

ログデータ格納先を決定したら、メモリキャッシュの場合は、スーパーストリームごとにプールファイルを配置する MAPID、DB の場合は、データにアクセスするためのリソースグループセット名を合わせて指定する必要があります。

なお、ログデータ格納先に指定可能なリソースグループセットは、デフォルトリソースグループセットに指定したリソースグループセットか、それと同一インスタンスグループ配下のリソースグループセットにしてください。(デフォルトリソースグループと RAC 構成となっている必要があります。)

[パラメータ名(最優先パラメータ)]

$DELAYED-%UNITGROUP-POOLFILE

$DELAYED-%COMMON-DBLAYER

$DELAYED-%SUPERSTREAM-MAPID

$DELAYED-%SUPERSTREAM-RGSETNAME

(a) プールファイル ID について

メモリキャッシュのプールファイルは、スタック番号ごとに論理表を作成する必要がありますが、同一の MAPID を指定した複数のスーパーストリームを定義する場合は、それらのプールファイルを識別するための情報として、さらにプールファイル ID を指定する必要があります。

プールファイルの論理表名は、下記の命名規則で決定しますので、このように定義して作成してください。

[プールファイル論理表命名規則]

DIOSA_DELAYED_POOL_{プールファイル ID(0 埋め 2 桁)}{スタック番号(0 埋め 2 桁)}

[パラメータ名(最優先パラメータ)]

$DELAYED-%SUPERSTREAM-POOLFILEID

[例 1] MAPID が異なる場合

スーパーストリーム名	MAPID	プールファイル ID	スタック数	プールファイル論理表
SPST1	1	指定不要(0)	5	DIOSA_DELAYED_POOL_0001～
SPST2	2	指定不要(0)	5	DIOSA_DELAYED_POOL_0005

[例 2] MAPID が重複する場合

スーパーストリーム名	MAPID	プールファイル ID	スタック数	プールファイル論理表
SPST1	1	指定不要(0)	5	DIOSA_DELAYED_POOL_0001～
SPST2	2	指定不要(0)	5	DIOSA_DELAYED_POOL_0005
SPST11	1	1	5	DIOSA_DELAYED_POOL_0101～
SPST12	2	1	5	DIOSA_DELAYED_POOL_0105

(3) その他プールファイル関連パラメータ

構成を決定したら、システムのリソース使用量に影響するその他のパラメータを決定します。

(a) スタックファイル数、スワップサイズ

スーパーストリームごとに、プールファイルをいくつのスタックファイルに分割するかをスタックファイル数、1 スタックファイルあたりのサイズをスワップサイズに指定します。

スタックファイル数×スワップサイズが、蓄積できるログデータ合計サイズになりますので、ログデータ発生量や、障害時の復旧時間などを考慮して決定する必要があります。

[パラメータ名(最優先パラメータ)]

$DELAYED-%SUPERSTREAM-STACKNUM

$DELAYED-%SUPERSTREAM-SWAPSIZE

(b) 削除スタック数

処理済みのスタックファイルを初期化する数を指定します。処理済みになったスタックファイルをできるだけ残しておきたい場合、小さな数を指定します。また、合計サイズが同じ場合、スタックファイル数を大きくして、スワップサイズを小さくした方が、削除するログデータ量を抑えた運用が可能となります。

[パラメータ名(最優先パラメータ)]

$DELAYED-%DSAM-%DEFAULT-DELSTACKCNT

(c)　ログデータ圧縮有無

ログデータを圧縮するか指定します。ログデータを圧縮することで、より多くのデータをプールファイルに格納することができます。また、ログデータ転送は圧縮データのままおこなうため、データ転送の効率も良くなります。

[パラメータ名(最優先パラメータ)]

$DELAYED-%SUPERSTREAM-COMPRESS

(d)　ログデータ削除監視間隔

処理済みとなったログデータを監視し、最新のスタックファイルから削除スタック数分のスタックファイルを初期化する処理を実施する間隔を指定します。

ログデータ格納先が IM と DB のそれぞれに対して個別の間隔を指定可能です。

[パラメータ名(最優先パラメータ)]

$DELAYED-%DSAM-DELINVL_IM

$DELAYED-%DSAM-DELINVL_DB

4.1.3　　ログデータ転送制御

プールファイルの配置が決定したら、ログデータを転送する経路に従って、センダユニット、レシーバユニットを配置します。異なる論理システムのプールファイル間には、データ転送元にセンダユニット、データ転送先にレシーバユニットを必ず配置します。

結果的に、ログデータ発生元の論理システム(プールファイル種別=WRITER)ではセンダユニットのみ、最終処理論理システムではレシーバユニットのみ、それ以外の論理システムでは、センダユニットとレシーバユニットの両方を定義することになります。

(1)　相手論理システム

転送をおこなう論理システム間では、相互の論理システムを定義します。

センダユニットには、受信側論理システム、レシーバユニットには、送信側論理システムをそれぞれ定義します。ここで指定する論理システムは、DIOSA/XTP 環境定義の SYSMAP 節に定義されている必要があります。

[パラメータ名(最優先パラメータ)]

$DELAYED-%UNITGROUP-%SNDUNIT-DLSNAME

$DELAYED-%UNITGROUP-%RCVUNIT-DLSNAME

(2)　電文サイズ

ログデータを送受信する電文サイズを指定します。送受信をおこなうセンダユニット、レシーバユニットでは同じサイズを指定することを推奨します。センダユニットの方が大きな値を指定した場合、ログデータ転送は実施することができません。

[パラメータ名(最優先パラメータ)]

$DELAYED-%SUPERSTREAM-%SNDUNIT-MSGSIZE

$DELAYED-%SUPERSTREAM-%RCVUNIT-MSGSIZE

(3)　データ発生監視間隔

プールファイルに未処理のログデータがなくなったときに、データの発生を定期的にチェックする処理をおこなう間隔を定義します。既定値定義は、ログデータ格納先ごとに異なる値を定義可能です。

[パラメータ名(最優先パラメータ)]

$DELAYED-%SUPERSTREAM-%SNDUNIT-CHKINVL

(4)　ログデータ転送性能関連項目

(a)　1 ペーシングあたりの送信電文数

1 ペーシング処理でセンダが送信するログデータ電文数を指定します。この値が大きいほど転送速度は上がりますが、滞留が発生した場合などに転送経路でのリソース消費が大きくなります。

[パラメータ名(最優先パラメータ)]

$DELAYED-%SUPERSTREAM-%SNDUNIT-PSENDCNT

(b) 電文送信遅延電文数、電文送信遅延間隔

1 ペーシング処理内で、電文送信遅延電文数に指定した電文数を処理するごとに、電文送信遅延間隔に指定した時間だけ送信処理を休止します。ログデータ発生量に対して、転送速度に余裕がある場合、ログデータ送信処理によるシステムへの最大負荷を下げることが可能です。

[パラメータ名(最優先パラメータ)]

$DELAYED-%SUPERSTREAM-%SNDUNIT-MSGDLYCNT

$DELAYED-%SUPERSTREAM-%SNDUNIT-MSGDLYTIME

(c) 1 コミットあたりの受信電文数

レシーバユニットが、受信処理中にコミットを実行するまでの処理電文数を指定します。この値が大きいほど受信処理の速度は上がりますが、メモリキャッシュや DB によるリソース消費が増大します。

[パラメータ名(最優先パラメータ)]

$DELAYED-%SUPERSTREAM-%RCVUNIT-MSGPERCMT

(5) 通信制御用定義

論理システム内のノード間や、論理システム間の通信をおこなうための定義は、DIOSA/XTP の DIOSAMAP 節や SYSMAP 節におこないます。ログデータ転送をおこなうためには、IP アドレスやポート番号の定義が必須です。

通信制御機能の詳細な動作パラメータは、$DELAYED 節-%PATHCTRL 項に定義しますが、センダユニット、レシーバユニットが定義されていれば、PATHCTRL 項は特に定義をしなくても動作可能です。

(a) 振り分けデーモン、通信デーモン最大/最小スレッド数

通信制御機能の電文送受信をおこなうデーモン内のスレッド数を指定します。

最小スレッド数で起動し、処理要求が滞留すると、最大スレッド数まではスレッドを起動して多重度を上げていきます。最大スレッド数を越える要求は同時に処理できませんので、最低でも全スーパーストリーム配下のセンダユニット数とレシーバユニット数の合計以上の値を指定することを推奨します。

[パラメータ名(最優先パラメータ)]

$DELAYED-%PATHCTRL-MAXTHR_C

$DELAYED-%PATHCTRL-MINTHR_C

$DELAYED-%PATHCTRL-MAXTHR_D

$DELAYED-%PATHCTRL-MINTHR_D

(b) 復旧監視間隔

通信できなかったノードや相手論理システムとの通信パスの復旧を確認する間隔を指定します。

[パラメータ名(最優先パラメータ)]

$DELAYED-%PATHCTRL-RCHKINVL

(c) ヘルスチェック間隔

通信可能と判断しているノードや相手論理システムのヘルスチェックを実行する間隔を指定します。

[パラメータ名(最優先パラメータ)]

$DELAYED-%PATHCTRL-HCHKINVL

(d) 相手論理システム通信対象外判定時間

本パラメータで指定した時間以上通信が発生しない論理システムは、使用しないと判断し、ヘルスチェックや復旧監視の処理を停止します。ただし、使用しないと判断された論理システムであっても、通信の宛先として指定された場合は、データ送信処理をおこないます。

本パラメータを定義する場合、(6)その他定義項目で定義するペーシング要求再送間隔、制御電文再送間隔に対して十分大きな(最低でも 2 倍以上の)時間を指定するようにしてください。

[パラメータ名(最優先パラメータ)]

$DELAYED-%PATHCTRL-DLSIDLETIME

(e) 通信制御用環境変数

通信制御では以下の環境変数で送受信時の動作を指定可能です。

環境変数名	説明
DIOSA_DELAYED_CONNECTTIMEOUT	電文送信時のソケット接続タイムアウト値(秒)
DIOSA_DELAYED_SELECTTIMEOUT	電文送受信時のソケット監視タイムアウト値(秒)
DIOSA_DELAYED_SENDTIMEOUT	電文送信時の送信タイムアウト値(秒)

(6) その他定義項目

(a) ユニット名

ユニットを識別するための名前です。

SUPERSTREAM 項配下に SNDUNIT、RCVUNIT を定義する場合、UNITGROUP 項配下で指定したのと同じユニット名を指定する必要があります。

[パラメータ名(最優先パラメータ)]

$DELAYED-%UNITGROUP-%SNDUNIT-NAME

$DELAYED-%SUPERSTREAM-%SNDUNIT-NAME

$DELAYED-%UNITGROUP-%RCVUNIT-NAME

$DELAYED-%SUPERSTREAM-%RCVUNIT-NAME

(b) ペーシング要求再送間隔、ペーシング要求再送回数

センダユニット固有の項目です。ペーシング要求に対するレシーバからの応答が返却されない場合に、要求を再送する間隔をペーシング要求再送間隔に指定します。ペーシング要求再送回数に指定した回数だけ要求しても応答が返却されない場合、ログデータの転送を停止します。

(5)通信制御用定義(c)ヘルスチェック間隔を指定していない、もしくは非常に長く設定する場合、再送回数はAPノード数以上の値を定義するようにしてください。

[パラメータ名(最優先パラメータ)]

$DELAYED-%SUPERSTREAM-%SNDUNIT-PRTYINVL

$DELAYED-%SUPERSTREAM-%SNDUNIT-PRTYMAX

(c) 制御電文再送間隔、制御電文再送回数

センダユニット、レシーバユニットそれぞれに定義可能です。開局や閉局などのために送信する制御電文に対する相手からの応答が返却されない場合に、要求を再送する間隔を制御電文再送間隔に指定します。制御電文再送回数に指定した回数だけ送信しても応答が返却されない場合、制御電文ごとに規定した状態に遷移します。

(5)通信制御用定義(c)ヘルスチェック間隔を指定していない、もしくは非常に長く設定する場合、再送回数はAPノード数以上の値を定義するようにしてください。

[パラメータ名(最優先パラメータ)]

$DELAYED-%SUPERSTREAM-%SNDUNIT-CRTYINVL

$DELAYED-%SUPERSTREAM-%SNDUNIT-CRTYMAX

$DELAYED-%SUPERSTREAM-%RCVUNIT-CRTYINVL

$DELAYED-%SUPERSTREAM-%RCVUNIT-CRTYMAX

(d) ログデータ転送制御用環境変数

ログデータ転送制御では以下の環境変数を指定可能です。

環境変数名	説明
DIOSA_DELAYED_DATADBGINVL	センダが転送するログデータ電文の運行ログを出力する通番の間隔(件数)
DIOSA_DELAYED_SENDRCVRYINVL	センダ実行デーモンがDB アクセス障害発生時にリトライする間隔(秒)

4.1.4　　ログデータ実行制御

ログデータ処理をおこないたい論理システム上では、プールファイルにログリーダユニットを定義します。

ログリーダユニットはスーパーストリームごとに最大 16 個まで定義可能で、1 つのログデータに対して複数の異なるアプリケーションを実行することもできます。

(1)　ユニット名

ユニット名を指定します。

スーパーストリーム配下に複数のログリーダユニットを定義する場合、個々のログリーダユニットを識別するための情報です。ユニットごとに異なる名前を指定してください。

[パラメータ名(最優先パラメータ)]

$DELAYED-%UNITGROUP-%LRDUNIT-NAME

$DELAYED-%SUPERSTREAM-%LRDUNIT-NAME

(2)　ユーザデータ更新先

ログリーダが呼び出すアプリケーションが更新するデータの格納先を指定します。

IM　　メモリキャッシュ

DB　　Oracle または PostgreSQL

指定しない場合はログデータの格納先と同じ接続先と判断します。

ログデータ格納先が IM の場合、ユーザデータ更新先に IM、DB のいずれも指定可能です。ログデータ格納先が DB の場合、ユーザデータ更新先には DB しか指定できません。

ユーザデータ更新先に DB を指定した場合、データにアクセスするためのリソースグループセット名を合わせて指定することができます。リソースグループセットはユニットごとに指定可能なので、ユニットごとに異なるデータベースを更新するようなシステムも構築可能です。

[パラメータ名(最優先パラメータ)]

$DELAYED-%UNITGROUP-%LRDUNIT-USERDATA

$DELAYED-%SUPERSTREAM-%LRDUNIT-RGSETNAME

(3)　ユーザアプリケーション関数名

ログデータ実行制御が呼び出すユーザアプリケーション関数名を指定します。本パラメータで指定する関数名の関数は、アプリケーション動的置換機能を利用して呼び出すため、アプリケーション動的置換機能の環境定義(APLIB 節)、または環境変数の設定を合わせておこなう必要があります。

以下の関数が指定可能です。

(a)　メイン処理

実際にログデータ処理をおこなう関数名を指定します。

メイン処理は必ず指定してください。

データ変換・通信オプションの更新ログによる非同期レプリケーションを利用する場合は、以下を指定してください。

%LRDUNIT-USERDATA が DB のログリーダユニットには、LogUpdOraExit を指定する。

%LRDUNIT-USERDATA が IM のログリーダユニットには、LogUpdTamExit を指定する。

[パラメータ名(最優先パラメータ)]

$DELAYED-%SUPERSTREAM-%LRDUNIT-MAINAP

(b) ユーザ用プロセス初期化/終了処理

ログリーダの処理実行プロセスの起動時、および停止時に一度ずつ呼び出されるユーザ関数です。プロセス初期化、終了に相当する処理を実装したい場合に指定します。指定を省略した場合関数は呼び出されません。

データ変換・通信オプションの更新ログによる非同期レプリケーションを利用する場合は、以下を指定してください。

プロセス初期化には、UpdPrcInitExit を指定する。

プロセス終了には、UpdPrcTermExit を指定する。

[パラメータ名(最優先パラメータ)]

$DELAYED-%SUPERSTREAM-%LRDUNIT-PRCINIT

$DELAYED-%SUPERSTREAM-%LRDUNIT-PRCTERM

(c) ユーザ用トランザクション初期化/終了処理

ログリーダの処理実行プロセスが、1 ロット処理をおこなう前後に呼び出されるユーザ関数です。トランザクション初期化、終了に相当する処理を実装したい場合に指定します。指定を省略した場合関数は呼び出されません。トランザクション終了処理は、コミット/ロールバックの前後で 1 回ずつ呼び出されます。

データ変換・通信オプションの更新ログによる非同期レプリケーションを利用する場合は、以下を指定してください。

トランザクション初期化には、UpdTrnInitExit を指定する。

トランザクション終了には、UpdTrnTermExit を指定する。

[パラメータ名(最優先パラメータ)]

$DELAYED-%SUPERSTREAM-%LRDUNIT-TRNINIT

$DELAYED-%SUPERSTREAM-%LRDUNIT-TRNTERM

(4) ログデータ処理性能関連項目

(a) コミット係数

1 ロット(1 コミット区間内)で処理するログデータ数を指定します。

この値が大きいほどログデータ実行処理の速度は上がりますが、メモリキャッシュや DB によるリソース消費が増大します。

[パラメータ名(最優先パラメータ)]

$DELAYED-%SUPERSTREAM-%LRDUNIT-DATAPERLOT

(b) ログデータ格納領域サイズ

ログデータを読み込むバッファのサイズを指定します。1 ログデータのサイズが、ここで指定したサイズを超える(1 ログデータも読み込めない)ような場合、領域は自動拡張されますが、コミット係数に指定した数のログデータが格納できない場合は、読み込めた数のログデータを 1 ロットとして処理します。

したがって、コミット係数の指定を有効にするためには、平均ログデータサイズ×コミット係数以上のサイズを指定する必要があります。

[パラメータ名(最優先パラメータ)]

$DELAYED-%SUPERSTREAM-%LRDUNIT-DATABUFSIZE

(c) 処理遅延間隔

1 ロット処理するごとに処理を休止する時間を指定します。ログデータ発生量に対して、処理速度に余裕がある場合、ログデータ実行処理によるシステムへの最大負荷を下げることが可能です。

[パラメータ名(最優先パラメータ)]

$DELAYED-%SUPERSTREAM-%LRDUNIT-DLYTIME

(5) ログデータ処理監視関連項目

(a) 性能情報収集有無

ログデータ処理関数およびコミット処理の処理時間統計情報を収集するかどうかを指定します。

収集すると、最大処理時間、最小処理時間、平均処理時間を参照できるようになります。

[パラメータ名(最優先パラメータ)]

$DELAYED-%SUPERSTREAM-%LRDUNIT-PERF

(b) 経過時間制限値

アプリケーションの処理時間が一定の時間を超過した場合に、警告メッセージを出したり、シグナルを送信して処理を中断させたりすることができます。

また、大量のデータ処理をおこなうアプリケーションでは、指定した経過時間を超過しないように、監視時間をリセットする API を実行することが可能ですが、そのリセット API を実行する上限回数も指定可能です。

[パラメータ名(最優先パラメータ)]

$DELAYED-%SUPERSTREAM-%LRDUNIT-ELPOVER

$DELAYED-%SUPERSTREAM-%LRDUNIT-ELPTIME

$DELAYED-%SUPERSTREAM-%LRDUNIT-ELPRESET

(c) CPU 時間制限値

アプリケーションの CPU 時間が一定の時間を超過した場合に、シグナルを送信して処理を中断させることができます。

[パラメータ名(最優先パラメータ)]

$DELAYED-%SUPERSTREAM-%LRDUNIT-CPUTIME

(6) リトライ回数

ロールバックリトライを行う最大回数を指定します。

指定した回数を超過した場合、ログデータ処理を停止します。

なお、DB アクセス障害時にも 5 秒のインターバル後にリトライをおこないますが、インターバル時間を変更したい場合は、環境変数(DIOSA_DELAYED_EXECRCVRYINVL)を設定してください。

[パラメータ名(最優先パラメータ)]

$DELAYED-%SUPERSTREAM-%LRDUNIT-RETRY

(7) データ発生監視間隔

プールファイルに未処理のログデータがなくなったときに、データの発生を定期的にチェックする処理をおこなう間隔を定義します。既定値定義は、ログデータ格納先(ユーザデータ更新先ではありません)ごとに異なる値を定義可能です。

[パラメータ名(最優先パラメータ)]

$DELAYED-%SUPERSTREAM-%LRDUNIT-CHKINVL

(8) その他定義項目

(a) ログデータ実行制御用環境変数

ログデータ実行制御では以下の環境変数を指定可能です。

環境変数名	説明
DIOSA_DELAYED_EXECRCVRYINVL	ログリーダ実行デーモンが DB アクセス障害発生時にリトライする間隔(秒)

4.1.5　　ストリーム所在管理機能関連定義

ストリーム所在管理機能は、プールファイル種別が DB のプールファイルを定義すれば、特に定義をしなくても動作可能です。ストリーム所在管理機能に関する定義項目は、%LOCATIONCTRL 項に定義します。

(1)　ストリーム所在管理機能エージェント受付時間

DB ノードで起動するマネージャが、エージェントからの接続を待つ時間を指定します。

指定時間以内にエージェントが起動しなかったノードは、動作ノードとしてスーパーストリームの振り分け対象外となります。

[パラメータ名(最優先パラメータ)]

$DELAYED-%LOCATIONCTRL-INITWAIT

(2)　ヘルスチェック間隔、ヘルスチェックリトライ回数

監視対象ノードのヘルスチェックをおこなう間隔を指定します。ヘルスチェック間隔×ヘルスチェックリトライ回数時間経過しても通信がないノードは、スーパーストリームの動作対象外となり、該当ノードで動作していたスーパーストリームは、他の正常稼働中のノードに振り分けられます。

[パラメータ名(最優先パラメータ)]

$DELAYED-%LOCATIONCTRL-CHKINVL

$DELAYED-%LOCATIONCTRL-CHKRETRY

(3)　スーパーストリームグループ

スーパーストリームグループを指定します。

[パラメータ名(最優先パラメータ)]

$DELAYED-%SUPERSTREAM-GROUP

(4)　ストリーム所在管理機能用環境変数

ストリーム所在管理機能では以下の環境変数を指定可能です。

環境変数名	説明
DIOSA_DELAYED_SLMLOCKID	ストリーム所在管理機能にて使用する DBMS ロック識別子 テスト実施のため、同一の DB インスタンス上に複数の論理システムを構築する必要がある場合、本環境変数に異なる値を設定する必要があります。

4.1.6　　初期無効化状態

スーパーストリームやユニットは、それぞれ無効化することができますが、環境定義で初期無効化状態を指定することで、データストア基盤の起動時から無効状態にすることができます。

[パラメータ名(最優先パラメータ)]

$DELAYED-%SUPERSTREAM-BLOCKSTATUS

$DELAYED-%SUPERSTREAM-%SNDUNIT-BLOCKSTATUS

$DELAYED-%SUPERSTREAM-%RCVUNIT-BLOCKSTATUS

$DELAYED-%SUPERSTREAM-%LRDUNIT-BLOCKSTATUS

4.1.7　　定義簡略化

DELAYED 節に定義する項目は、既定値が存在するため定義しなくても動作可能な項目や、全体に共通の既定値定義が可能な項目、後述するユニットグループに定義可能な項目、スーパーストリーム単位に定義可能な項目と、同一のパラメータが何ヶ所かで定義可能なものがあります。

これらは定義量を削減するための定義であり、個々に定義する必要のない項目については、定義を省略したり、既定値を定義したりすることで、最小限の定義でシステムを構築することができます。

(1)　既定値定義

ディレード全体で同じ定義値を使用するような項目については、既定値定義のみおこなえば、各スーパーストリームはそこに定義された値を使用します。また、ほとんどの既定値定義には、それ自体が省略された場合の既定値があり、パラメータがどこにも記述されなかった場合には、その値で動作します。

既定値定義と同じパラメータ項目が、個々のスーパーストリームや後述するユニットグループ単位に指定された場合、既定値定義より優先されます。

項	パラメータ名	説明
%DSAM-%DEFAULT	STACKNUM	プールファイルのスタックファイル数
	SWAPSIZE	スワップサイズ
	DELSTACKCNT	削除スタック数
	COMPRESS	ログデータ圧縮有無
%SENDER-%DEFAULT	MSGSIZE	電文サイズ
	CHKINVL_IM	データ格納先が IM のスーパーストリームのデータ発生監視間隔
	CHKINVL_DB	データ格納先が DB のスーパーストリームのデータ発生監視間隔
	PSENDCNT	1 ペーシングあたりの送信電文数
	PRTYINVL	ペーシング要求電文再送間隔
	PRTYMAX	ペーシング要求電文再送回数
	MSGDLYCNT	電文送信遅延電文数
	MSGDLYTIME	電文送信遅延間隔
	CRTYINVL	制御電文再送間隔
	CRTYMAX	制御電文再送回数
%RECEIVER-%DEFAULT	MSGSIZE	電文サイズ
	MSGPERCMT	1 コミットあたりの電文受信数
	CRTYINVL	制御電文再送間隔
	CRTYMAX	制御電文再送回数
%LOGREADER-%DEFAULT	DATAPERLOT	コミット係数
	CHKINVL_IM	データ格納先が IM のスーパーストリームのデータ発生監視間隔
	CHKINVL_DB	データ格納先が DB のスーパーストリームのデータ発生監視間隔
	DLYTIME	処理遅延間隔
	PERF	性能情報収集有無
	ELPOVER	経過時間制限値超過時動作
	ELPTIME	経過時間制限値
	ELPRESET	経過時間リセット最大回数
	CPUTIME	CPU 時間制限値
	RETRY	リトライ最大回数
	DATABUFSIZE	ログデータ格納領域サイズ
	PRCINIT	ユーザ用プロセス初期化処理
	PRCTERM	ユーザ用プロセス終了処理
	TRNINIT	ユーザ用トランザクション初期化処理
	TRNTERM	ユーザ用トランザクション終了処理
	MAINAP	ユーザ用ログデータ実行メイン処理

表 4-1 既定値定義可能パラメータ

(2)　ユニットグループ

同じ処理をおこなう、多重化されたスーパーストリームの定義を簡略化するため、ユニットグループという定義があります。ユニットグループに、スーパーストリーム配下のユニット構成を定義すれば、同じ構成のスーパーストリームについては、ユニットグループ名だけ指定すれば、共通の定義項目は個々に記述する必要がありません。スーパーストリームとユニットグループは、$DELAYED-%UNITGROUP-NAME に定義した名前を、$DELAYED-%SUPERSTREAM-UNITGROUP に指定することで対応付けられます。

多くの定義パラメータや配下のユニット情報は、ユニットグループ項、スーパーストリーム項の両方に定義可能ですが、下記の項目については、いずれかでのみ定義可能です。

ユニットグループに記述した定義の一部のみを変更したい場合、SUPERSTREAM 項配下で、変更したいユニットの、変更したいパラメータのみを定義すれば、その部分だけ上書きされた定義となります。

項	パラメータ名	説明
%UNITGROUP	NAME	ユニットグループ名
	POOLTYPE	プールファイル種別
	POOLFILE	ログデータ格納先
%UNITGROUP-%SNDUNIT	DLSNAME	相手論理システム名
%UNITGROUP-%RCVUNIT	DLSNAME	相手論理システム名

表 4-2 ユニットグループ項配下のみ定義可能なパラメータ

項	パラメータ名	説明
%SUPERSTREAM	NAME	スーパーストリーム名
	ALIAS	エイリアス名
	MAPID	MAPID(プールファイル種別が IM の場合)
	GROUP	スーパーストリームグループ

表 4-3 スーパーストリーム項配下のみ定義可能なパラメータ

4.1.8　　その他定義項目

ここまでの説明で出ていない定義パラメータについて説明します。

(1)　共通定義

(a)　フェールオーバー指定

実行デーモン以外のディレード転送で管理しているデーモンが、死活監視機能による再起動でリトライオーバーした場合に、論理ノードをフェールオーバーするかどうかを指定します。

[パラメータ名(最優先パラメータ)]

$DELAYED-%COMMON-FAILOVER

(b)　実行デーモン再起動処理制御用定義

センダ、レシーバ、ログリーダの管理デーモンが、実行デーモン異常終了時に再起動処理をおこなう間隔と、リトライを中止して処理を停止すると判断するまでの再起動回数を指定します。

DMNDOWNTIME に指定した時間内に、DMNDOWNCNT 回再起動に失敗すると、再起動を停止します。

[パラメータ名(最優先パラメータ)]

$DELAYED-%COMMON-DMNRESTINVL

$DELAYED-%COMMON-DMNDOWNCNT

$DELAYED-%COMMON-DMNDOWNTIME

(c)　動作ノードチェック間隔

自ノードで動作対象外となった実行デーモンを停止する処理の自動チェック間隔を指定します。通常は動作対象外となった場合は即座に通知処理をおこなうため、自動チェックによって検出する前にノード移動がおこなわれます。

[パラメータ名(最優先パラメータ)]

$DELAYED-%COMMON-RCVRYCHKINVL

(2)　エイリアス名

ログデータ登録用に、スーパーストリームに別名を付けたい場合に指定します。

[パラメータ名(最優先パラメータ)]

$DELAYED-%SUPERSTREAM-ALIAS

4.2　環境定義

環境定義には、データストア基盤以外にも、いくつかの定義をおこなう必要があります。

4.2.1　データストア基盤

(1)　DIOSA/XTP 環境定義(DELAYED 節)

4.1 で説明した定義内容については、DELAYED 節に記述します。

例として、4.1.1 で例とした論理システム構成の、論理システム 2 でおこなう定義を記述します。

```
$DELAYED
    %UNITGROUP
        NAME = UNITGROUP1
        POOLTYPE = RECEIVER
        POOLFILE = IM
        %RCVUNIT
            NAME = RECEIVER
            DLSNAME = DLS1
        ;
        %SNDUNIT
            NAME = SENDER
            DLSNAME = DLS4
        ;
        %LRDUNIT
            NAME = LOGREADER
            USERDATA = IM
            MAINAP = AP1
        ;
    ;
    %SUPERSTREAM
        NAME = SPST1
        UNITGROUP = UNITGROUP1
        MAPID = 1
    ;
    %SUPERSTREAM
        NAME = SPST2
        UNITGROUP = UNITGROUP1
        MAPID = 2
    ;
;
```

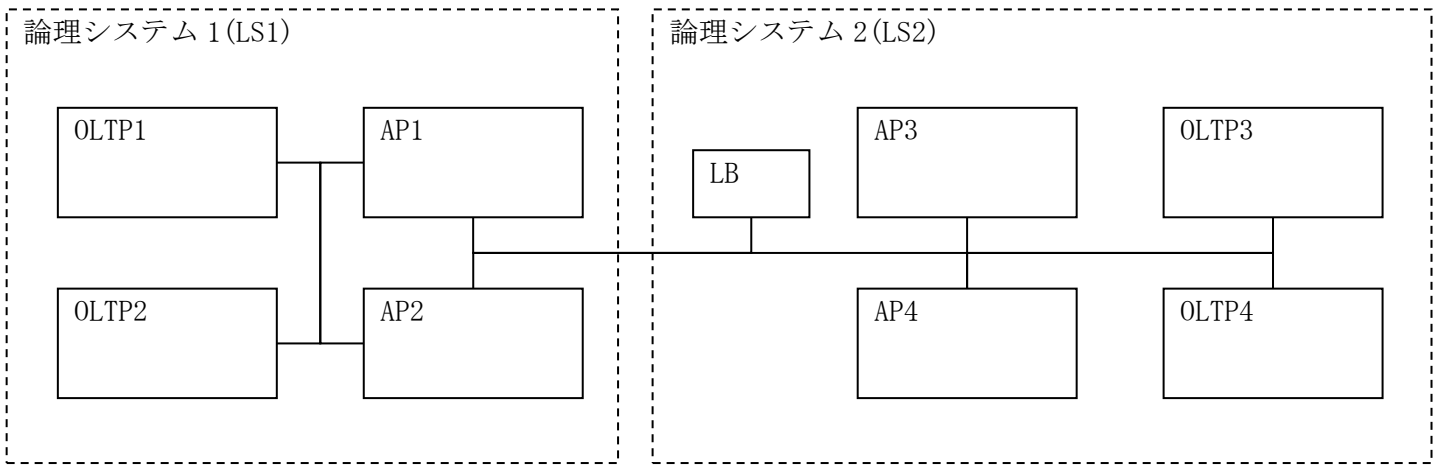
(2)　DIOSA/XTP 環境定義(DIOSAMAP 節、SYSMAP 節)

論理システムの構成や、論理システム内の論理ノード構成は、DIOSA/XTP の環境定義(DIOSAMAP 節、SYSMAP 節)に記述し、ディレードの通信で使用する IP アドレスやポート番号を指定する必要があります。

センダユニット、レシーバユニットに指定する相手論理システム名は、SYSMAP 節に記述されている必要があります。

(a)　AP ノードを構築する構成の場合

論理システム 1(LS1)と論理システム 2(LS2)で転送をおこなう場合。(論理システム 1 側は外部向け IP アドレスを持つ AP ノードで直接受信し、論理システム 2 側は負荷分散装置(LB)が受信し、外部向け IP アドレスを持たない AP ノードに振り分ける)



SYSMAP 定義(論理システム 1/2 共通)

定義パラメータ	定義値
LOGSYSTEM 項(LS1) -ROUTE 項-TYPE	DIRECT
-ROUTE 項-IPADDR	AP ノード 1 の外部向け IP アドレス(①)
-ROUTE 項-PORT_DLT	AP ノード 1 の外部向けディレード転送用ポート番号(②)
-ROUTE 項-TYPE	DIRECT
-ROUTE 項-IPADDR	AP ノード 2 の外部向け IP アドレス(③)
-ROUTE 項-PORT_DLT	AP ノード 2 の外部向けディレード転送用ポート番号(④)
LOGSYSTEM 項(LS2) -ROUTE 項-TYPE	LB
-ROUTE 項-IPADDR	LB の IP アドレス
-ROUTE 項-PORT_DLT	LB のディレード転送用ポート番号

DIOSAMAP 定義(論理システム 1)

定義パラメータ	定義値
LNODE 項(AP1) -IPADDR_CTL	AP1 の内部向け IP アドレス
-IPADDR_EXT	AP1 の外部向け IP アドレス(①と一致させる)
-PORT_DLT	AP1 のディレード転送用ポート番号(②と一致させる)
LNODE 項(AP2) -IPADDR_CTL	AP2 の内部向け IP アドレス
-IPADDR_EXT	AP2 の外部向け IP アドレス(③と一致させる)
-PORT_DLT	AP2 のディレード転送用ポート番号(④と一致させる)
LNODE 項(OLTP1) -IPADDR_CTL	OLTP1 の IP アドレス
-PORT_DLT	OLTP1 のディレード転送用ポート番号
LNODE 項(OLTP2) -IPADDR_CTL	OLTP2 の IP アドレス
-PORT_DLT	OLTP2 のディレード転送用ポート番号

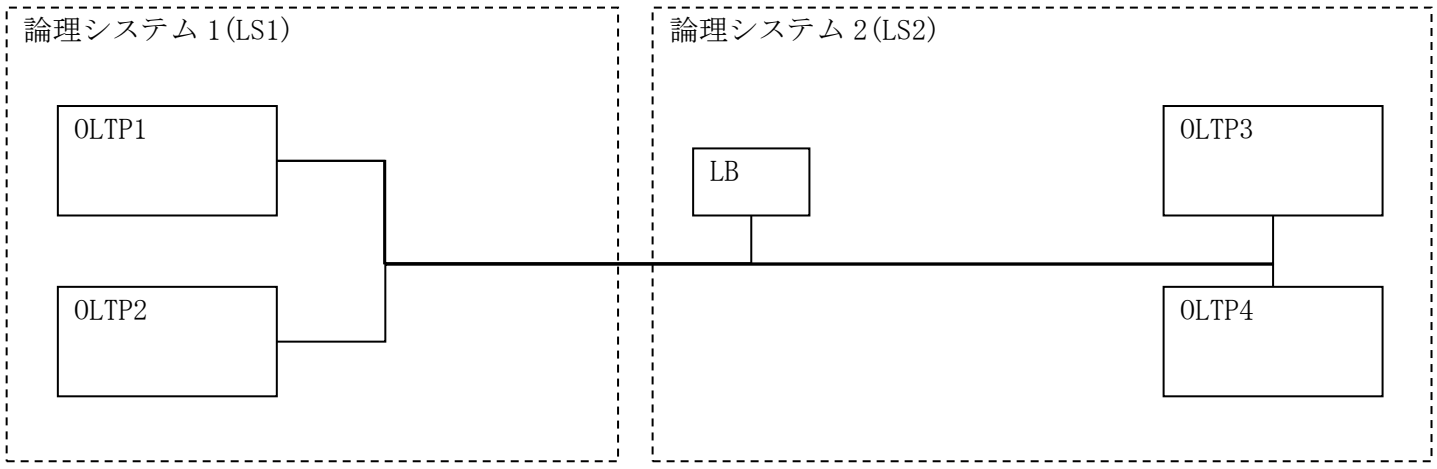
DIOSAMAP 定義(論理システム 2)

定義パラメータ	定義値
LNODE 項(AP3) -IPADDR_CTL	AP3 の内部向け IP アドレス(LB の分散先に設定)
-PORT_DLT	AP3 のディレード転送用ポート番号
LNODE 項(AP4) -IPADDR_CTL	AP4 の内部向け IP アドレス(LB の分散先に設定)
-PORT_DLT	AP4 のディレード転送用ポート番号
LNODE 項(OLTP3) -IPADDR_CTL	OLTP3 の IP アドレス
-PORT_DLT	OLTP3 のディレード転送用ポート番号
LNODE 項(OLTP4) -IPADDR_CTL	OLTP4 の IP アドレス
-PORT_DLT	OLTP4 のディレード転送用ポート番号

ROUTE や LNODE は、IP アドレスとポートの組み合わせで、通信経路やノードが識別できれば良い。

(b) AP ノードを構築しない構成の場合

論理システム 1(LS1)と論理システム 2(LS2)で転送をおこなう場合。(論理システム 1 側は外部向け IP アドレスを持つ OLTP ノードで直接受信し、論理システム 2 側は負荷分散装置(LB)が受信し、外部向け IP アドレスを持たない OLTP ノードに振り分ける)



SYSMAP 定義(論理システム 1/2 共通)

定義パラメータ	定義値
LOGSYSTEM 項(LS1) -ROUTE 項-TYPE	DIRECT
-ROUTE 項-IPADDR	OLTP ノード 1 の外部向け IP アドレス(①)
-ROUTE 項-PORT_DLT	OLTP ノード 1 の外部向けディレード転送用ポート番号(②)
-ROUTE 項-TYPE	DIRECT
-ROUTE 項-IPADDR	OLTP ノード 2 の外部向け IP アドレス(③)
-ROUTE 項-PORT_DLT	OLTP ノード 2 の外部向けディレード転送用ポート番号(④)
LOGSYSTEM 項(LS2) -ROUTE 項-TYPE	LB
-ROUTE 項-IPADDR	LB の IP アドレス
-ROUTE 項-PORT_DLT	LB のディレード転送用ポート番号

DIOSAMAP 定義(論理システム 1)

定義パラメータ	定義値
LNODE 項(OLTP1) -IPADDR_CTL	OLTP1 の内部向け IP アドレス
-IPADDR_EXT	OLTP1 の外部向け IP アドレス(①と一致させる)
-PORT_DLT	OLTP1 のディレード転送用ポート番号(②と一致させる)
LNODE 項(OLTP2) -IPADDR_CTL	OLTP2 の内部向け IP アドレス
-IPADDR_EXT	OLTP2 の外部向け IP アドレス(③と一致させる)
-PORT_DLT	OLTP2 のディレード転送用ポート番号(④と一致させる)

DIOSAMAP 定義(論理システム 2)

定義パラメータ	定義値
LNODE 項(OLTP3) -IPADDR_CTL	OLTP3 の内部向け IP アドレス(LB の分散先に設定)
-PORT_DLT	OLTP3 のディレード転送用ポート番号
LNODE 項(OLTP4) -IPADDR_CTL	OLTP4 の内部向け IP アドレス(LB の分散先に設定)
-PORT_DLT	OLTP4 のディレード転送用ポート番号
-PORT_DLT	OLTP4 のディレード転送用ポート番号

ROUTE や LNODE は、IP アドレスとポートの組み合わせで、通信経路やノードが識別できれば良い。

(3) DIOSA/XTP 環境定義(APLIB 節)

ログリーダが呼び出すユーザアプリケーションや、DB への接続に、接続出口関数を定義している場合の関数呼び出しには、DIOSA/XTP のアプリケーション動的置換機能を利用するため、環境定義(APLIB 節)記述、もしくは環境変数設定のいずれかをおこなう必要があります。

詳細は DIOSA/XTP 利用の手引きの記述を参照してください。

利用条件	アプリケーション動的置換機能利用プロセス
DB 接続出口関数を定義している	didltblock, didltchg, didltcreate, didltdivchg, didltinit, didltmodreset, didltpooldeld, didltpoolmod, didltpoolref, didltrefdb, didltslmmgrd, didltslmmv, didltslmref, didltupd, didtlexecd, didtlref, didtrexecd, didtsexecd, didtsref
ログリーダを利用する	didtlexecd

(4) その他 DIOSA/XTP 環境定義

(a) コマンド配信機能

データストア基盤の動作には、論理システム内でコマンド配信機能を実行するための環境定義(DIOSAMAP 節、CMDSEND 節)が必要です。

詳細は DIOSA/XTP 利用の手引きの記述を参照してください。

4.2.2　　メモリキャッシュ関連定義

プールファイルのデータ格納先にメモリキャッシュを指定する場合、メモリキャッシュや TAM に関する定義をする必要があります。

定義が必要となる条件は以下の通りです。

論理表名	定義が必要となる条件
DIOSA_DSAM	POOLTYPE=IM のスーパーストリームが定義されている
DIOSA_DELAYED_SPST	
DIOSA_DELAYED_STRM	
DIOSA_DELAYED_STACK	
DIOSA_DELAYED_STACKDATANO	
DIOSA_DELAYED_UNIT	
DIOSA_DELAYED_POOL_xxxx	
DIOSA_SENDER_UNIT	POOLTYPE=IM のスーパーストリームにセンダユニットが定義されている
DIOSA_SENDER_STRM	
DIOSA_RECEIVER_UNIT	POOLTYPE=IM のスーパーストリームにレシーバユニットが定義されている
DIOSA_RECEIVER_STRM	
DIOSA_LOGREADER_UNIT	ユーザデータ更新先が IM のログリーダユニットが定義されている
DIOSA_LOGREADER_STRM	

詳細は DIOSA/XTP メモリキャッシュ 利用の手引きの記述を参照してください。

(1)　DIOSA/XTP 環境定義(IMTABLECONF 節)

データストア基盤で使用するプールファイルや制御表の定義をおこないます。

以下に記述例を示します。PTABLE 項については、スーパーストリームに割り当てた MAPID 数分の記述をおこなう必要がありますが、DIOSA_DSAM 表だけは、任意の 1MAPID 分の記述のみをおこないます。

また、プールファイルについては、各スーパーストリームに定義した中で、最大のスタックファイル数分の LTABLE 項を定義する必要があります。

PTABLE 項に記述する TAM 表名は任意ですが、「論理表名_MAPID」といった名称を推奨します。

(a)　DIOSA_DSAM 表

```
%LTABLE
    TYPE        = 0
    NAME        = DIOSA_DSAM
    ID          = 任意の ID
    %RECORDCONF
        TYPE  = FIXED
        %PRIMARYKEY
            NAME = DIOSA_DSAM_00_IDX
            %KEYDEF
                OFFSET = 0
                LENGTH = 4
            ;
        ;
    ;
    %PTABLE
        NAME    = 任意の TAM 表名
        MAPID   = 任意の MAPID
    ;
;
```

(b) DIOSA_DELAYED_SPST 表

```
%LTABLE
    TYPE        = 0
    NAME        = DIOSA_DELAYED_SPST
    ID          = 任意の ID
    %RECORDCONF
        TYPE  = FIXED
        %PRIMARYKEY
            NAME = DIOSA_DELAYED_SPST_00_IDX
            %KEYDEF
                OFFSET = 0
                LENGTH = 16
            ;
        ;
    ;

    %PTABLE                                          定義した MAPID 数分繰り返す
        NAME    = 任意の TAM 表名
        MAPID   = スーパーストリームに割り当てた MAPID
    ;
;
```

(c) DIOSA_DELAYED_STRM

```
%LTABLE
    TYPE        = 0
    NAME        = DIOSA_DELAYED_STRM
    ID          = 任意の ID
    %RECORDCONF
        TYPE  = FIXED
        %PRIMARYKEY
            NAME = DIOSA_DELAYED_STRM_00_IDX
            %KEYDEF
                OFFSET = 0
                LENGTH = 16
            ;
            %KEYDEF
                OFFSET = 16
                LENGTH = 16
            ;
            %KEYDEF
                OFFSET = 32
                LENGTH = 4
            ;
        ;
        %SECONDARYKEY
            NAME = DIOSA_DELAYED_STRM_01_IDX
            %KEYDEF
                OFFSET = 0
                LENGTH = 16
            ;
        ;
        %SECONDARYKEY
            NAME = DIOSA_DELAYED_STRM_04_IDX
            %KEYDEF
                OFFSET = 0
                LENGTH = 16
            ;
            %KEYDEF
                OFFSET = 68
                LENGTH = 4
            ;
            %KEYDEF
                OFFSET = 32
                LENGTH = 4
            ;
        ;
        %SECONDARYKEY
            NAME = DIOSA_DELAYED_STRM_05_IDX
            %KEYDEF
                OFFSET = 0
                LENGTH = 16
            ;
            %KEYDEF
                OFFSET = 68
                LENGTH = 4
            ;
            %KEYDEF
                OFFSET = 128
                LENGTH = 8
            ;
        ;
        %SECONDARYKEY
```

```
            NAME = DIOSA_DELAYED_STRM_06_IDX
            %KEYDEF
                OFFSET = 0
                LENGTH = 16
            ;
            %KEYDEF
                OFFSET = 32
                LENGTH = 4
            ;
        ;
    ;

    %PTABLE                                                    定義した MAPID 数分繰り返す
        NAME    = 任意の TAM 表名
        MAPID   = スーパーストリームに割り当てた MAPID
    ;
;
```

(d) DIOSA_DELAYED_STACK

```
%LTABLE
    TYPE        = 0
    NAME        = DIOSA_DELAYED_STACK
    ID          = 任意の ID
    %RECORDCONF
        TYPE  = FIXED
        %PRIMARYKEY
            NAME = DIOSA_DELAYED_STACK_00_IDX
            %KEYDEF
                OFFSET = 0
                LENGTH = 16
            ;
            %KEYDEF
                OFFSET = 16
                LENGTH = 4
            ;
        ;
        %SECONDARYKEY
            NAME = DIOSA_DELAYED_STACK_01_IDX
            %KEYDEF
                OFFSET = 0
                LENGTH = 16
            ;
        ;
        %SECONDARYKEY
            NAME = DIOSA_DELAYED_STACK_02_IDX
            %KEYDEF
                OFFSET = 0
                LENGTH = 16
            ;
            %KEYDEF
                OFFSET = 20
                LENGTH = 4
            ;
        ;
    ;

    %PTABLE                                          定義した MAPID 数分繰り返す
        NAME    = 任意の TAM 表名
        MAPID   = スーパーストリームに割り当てた MAPID
    ;
;
```

(e) DIOSA_DELAYED_STACKDATANO

```
%LTABLE
    TYPE        = 0
    NAME        = DIOSA_DELAYED_STACKDATANO
    ID          = 任意の ID
    %RECORDCONF
        TYPE  = FIXED
        %PRIMARYKEY
            NAME = DIOSA_DELAYED_STKDATANO_00_IDX
            %KEYDEF
                OFFSET = 0
                LENGTH = 16
            ;
            %KEYDEF
                OFFSET = 16
                LENGTH = 16
            ;
            %KEYDEF
                OFFSET = 32
                LENGTH = 4
            ;
            %KEYDEF
                OFFSET = 36
                LENGTH = 4
            ;
        ;
        %SECONDARYKEY
            NAME = DIOSA_DELAYED_STKDATANO_01_IDX
            %KEYDEF
                OFFSET = 0
                LENGTH = 16
            ;
            %KEYDEF
                OFFSET = 16
                LENGTH = 16
            ;
            %KEYDEF
                OFFSET = 36
                LENGTH = 4
            ;
    ;
    %SECONDARYKEY
        NAME = DIOSA_DELAYED_STKDATANO_02_IDX
        %KEYDEF
            OFFSET = 0
            LENGTH = 16
        ;
        %KEYDEF
            OFFSET = 16
            LENGTH = 16
        ;
        %KEYDEF
            OFFSET = 32
            LENGTH = 4
        ;
    ;
    %SECONDARYKEY
        NAME = DIOSA_DELAYED_STKDATANO_03_IDX
        %KEYDEF
            OFFSET = 0
```

```
            LENGTH = 16
        ;
    ;
    %SECONDARYKEY
        NAME = DIOSA_DELAYED_STKDATANO_04_IDX
        %KEYDEF
            OFFSET = 0
            LENGTH = 16
        ;
        %KEYDEF
            OFFSET = 16
            LENGTH = 16
        ;
        %KEYDEF
            OFFSET = 32
            LENGTH = 4
        ;
        %KEYDEF
            OFFSET = 40
            LENGTH = 8
        ;
    ;
    %SECONDARYKEY
        NAME = DIOSA_DELAYED_STKDATANO_05_IDX
        %KEYDEF
            OFFSET = 0
            LENGTH = 16
        ;
        %KEYDEF
            OFFSET = 36
            LENGTH = 4
        ;
                ;
            ;
        ;

    %PTABLE                                                    定義した MAPID 数分繰り返す
        NAME    = 任意の TAM 表名
        MAPID   = スーパーストリームに割り当てた MAPID
    ;
;
```

(f) DIOSA_DELAYED_UNIT

```
%LTABLE
    TYPE        = 0
    NAME        = DIOSA_DELAYED_UNIT
    ID          = 任意の ID
    %RECORDCONF
        TYPE  = FIXED
        %PRIMARYKEY
            NAME = DIOSA_DELAYED_UNIT_00_IDX
            %KEYDEF
                OFFSET = 0
                LENGTH = 16
            ;
            %KEYDEF
                OFFSET = 16
                LENGTH = 1
            ;
            %KEYDEF
                OFFSET = 17
                LENGTH = 16
            ;
        ;
        %SECONDARYKEY
            NAME = DIOSA_DELAYED_UNIT_01_IDX
            %KEYDEF
                OFFSET = 0
                LENGTH = 16
            ;
        ;
    ;
```

定義した MAPID 数分繰り返す

```
    %PTABLE
        NAME     = 任意の TAM 表名
        MAPID    = スーパーストリームに割り当てた MAPID
    ;
;
```

(g) DIOSA_DELAYED_POOL_xxxx

プールファイルの論理表名は以下の命名規則で決定します。

DIOSA_DELAYED_POOL_{プールファイル ID(0 埋め 2 桁)}{スタック番号(0 埋め 2 桁)}

プールファイル ID を定義していない場合 0 が既定値ですので、例えば MAPID の重複のないスーパーストリームを定義して、全てのスタックファイル数が 10 の場合、DIOSA_DELAYED_POOL_0001～DIOSA_DELAYED_POOL_0010 の 10 個の論理表を定義する必要があります。

定義したスタックファイル数(最大)分繰り返す

```
%LTABLE
    TYPE         = 0
    NAME         = DIOSA_DELAYED_POOL_xxxx
    ID           = 任意の ID
    %RECORDCONF
        TYPE  = VARLEN
        %PRIMARYKEY
            NAME = DIOSA_DELAYED_POOL_00_IDX
            %KEYDEF
                OFFSET = 0
                LENGTH = 28
            ;
        ;
    ;
```

定義した MAPID 数分繰り返す

```
    %PTABLE
        NAME     = 任意の TAM 表名
        MAPID    = スーパーストリームに割り当てた MAPID
    ;
;
```

(h) DIOSA_SENDER_UNIT

```
%LTABLE
    TYPE        = 0
    NAME        = DIOSA_SENDER_UNIT
    ID          = 任意の ID
    %RECORDCONF
        TYPE  = FIXED
        %PRIMARYKEY
            NAME = DIOSA_SENDER_UNIT_00_IDX
            %KEYDEF
                OFFSET = 0
                LENGTH = 32
            ;
        ;
    ;

    %PTABLE                                    センダユニットを定義した MAPID 数分繰り返す
        NAME    = 任意の TAM 表名
        MAPID   = スーパーストリームに割り当てた MAPID
    ;
;
```

(i) DIOSA_SENDER_STRM

```
%LTABLE
    TYPE        = 0
    NAME        = DIOSA_SENDER_STRM
    ID          = 任意の ID
    %RECORDCONF
        TYPE  = FIXED
        %PRIMARYKEY
            NAME = DIOSA_SENDER_STRM_00_IDX
            %KEYDEF
                OFFSET = 0
                LENGTH = 48
            ;
        ;
    ;

    %PTABLE                                    センダユニットを定義した MAPID 数分繰り返す
        NAME    = 任意の TAM 表名
        MAPID   = スーパーストリームに割り当てた MAPID
    ;
;
```

(j) DIOSA_RECEIVER_UNIT

```
%LTABLE
    TYPE        = 0
    NAME        = DIOSA_RECEIVER_UNIT
    ID          = 任意の ID
    %RECORDCONF
        TYPE  = FIXED
        %PRIMARYKEY
            NAME = DIOSA_RECEIVER_UNIT_OO_IDX
            %KEYDEF
                OFFSET = 0
                LENGTH = 32
            ;
        ;
    ;

    %PTABLE                                    レシーバユニットを定義した MAPID 数分繰り返す
        NAME    = 任意の TAM 表名
        MAPID   = スーパーストリームに割り当てた MAPID
    ;
;
```

(k) DIOSA_RECEIVER_STRM

```
%LTABLE
    TYPE        = 0
    NAME        = DIOSA_RECEIVER_STRM
    ID          = 任意の ID
    %RECORDCONF
        TYPE  = FIXED
        %PRIMARYKEY
            NAME = DIOSA_RECEIVER_STRM_OO_IDX
            %KEYDEF
                OFFSET = 0
                LENGTH = 48
            ;
        ;
    ;

    %PTABLE                                    レシーバユニットを定義した MAPID 数分繰り返す
        NAME    = 任意の TAM 表名
        MAPID   = スーパーストリームに割り当てた MAPID
    ;
;
```

(l) DIOSA_LOGREADER_UNIT

```
%LTABLE
    TYPE        = 0
    NAME        = DIOSA_LOGREADER_UNIT
    ID          = 任意の ID
    %RECORDCONF
        TYPE  = FIXED
        %PRIMARYKEY
            NAME = DIOSA_LOGREADER_UNIT_00_IDX
            %KEYDEF
                OFFSET = 0
                LENGTH = 32
            ;
        ;
    ;

    %PTABLE                                    ログリーダユニットを定義した MAPID 数分繰り返す
        NAME    = 任意の TAM 表名
        MAPID   = スーパーストリームに割り当てた MAPID
    ;
;
```

(m) DIOSA_LOGREADER_STRM

```
%LTABLE
    TYPE        = 0
    NAME        = DIOSA_LOGREADER_STRM
    ID          = 任意の ID
    %RECORDCONF
        TYPE  = FIXED
        %PRIMARYKEY
            NAME = DIOSA_LOGREADER_STRM_00_IDX
            %KEYDEF
                OFFSET = 0
                LENGTH = 48
            ;
        ;
    ;

    %PTABLE                                    ログリーダユニットを定義した MAPID 数分繰り返す
        NAME    = 任意の TAM 表名
        MAPID   = スーパーストリームに割り当てた MAPID
    ;
;
```

(2) DIOSA/XTP 環境定義(IMENV 節)

IMENV 節には、IMTABLECONF 節に定義した MAPID についてのハッシュ値定義や、MAPID の配置等を定義する必要があります。

(3) TAM 環境定義

IMTABLECONF 節に記述した TAM 表定義は、TAM のテーブル定義(table.conf)にも記述する必要があります。

見出しは論理表名で記述しますが、各論理表名配下の PTABLE 節に記述したテーブル名が、実際に定義する必要がある TAM 表名で、1 つの論理表に対して複数の PTABLE 項を記述した場合、PTABLE 項の定義数分の TAM 表定義が必要です。太字の項目については、スーパーストリームの構成等によらず変更不可ですので、定義例に書かれてい

る通りに記述してください。

各項目や、記述されていないパラメータの詳細につきましては、InfoFrame Table Access Method利用の手引きの記述を参照してください。

(a) DIOSA_DSAM表

```
[table]
    tam_table_name          = TAMテーブル名
    tam_table_file          = TAMファイル名
    backup                  = yes
    share                   = yes
    auto_extend             = yes
    auto_extend_size        = 1
    max_auto_extend_size    = 5
    record_size             = 96
    init_record_num         = 10
    index_number            = 1
    index_name1             = DIOSA_DSAM_00_IDX
    keys1                   = 0:4
    index_entry_num1        = 2
    auto_extend_key_num1    = 20
    max_auto_extend_count1  = 10
    init_key_num1           = 10
    auto_release_index1     = yes
```

(b) DIOSA_DELAYED_SPST表

```
[table]
    tam_table_name          = TAMテーブル名
    tam_table_file          = TAMファイル名
    backup                  = yes
    share                   = yes
    auto_extend             = yes
    auto_extend_size        = 1
    max_auto_extend_size    = 5
    record_size             = 144
    init_record_num         = 1000
    index_number            = 1
    index_name1             = DIOSA_DELAYED_SPST_00_IDX
    keys1                   = 0:16
    index_entry_num1        = 10
    auto_extend_key_num1    = 20
    max_auto_extend_count1  = 10
    init_key_num1           = 1000
    auto_release_index1     = yes
```

(c) DIOSA_DELAYED_STRM

```
[table]
    tam_table_name          = TAM テーブル名
    tam_table_file          = TAM ファイル名
    backup                  = yes
    share                   = yes
    auto_extend             = yes
    auto_extend_size        = 1
    max_auto_extend_size    = 5
    record_size             = 176
    init_record_num         = 1000
    index_number            = 5
    index_name1             = DIOSA_DELAYED_STRM_00_IDX
    keys1                   = 0:16,16:16,32:4
    index_entry_num1        = 10
    auto_extend_key_num1    = 20
    max_auto_extend_count1  = 10
    init_key_num1           = 1000
    auto_release_index1     = yes
    index_name2             = DIOSA_DELAYED_STRM_01_IDX
    keys2                   = 0:16,0:16,16:16,32:4
    index_entry_num2        = 10
    auto_extend_key_num2    = 20
    max_auto_extend_count2  = 10
    init_key_num2           = 1000
    auto_release_index2     = yes
    index_name3             = DIOSA_DELAYED_STRM_04_IDX
    keys3                   = 0:16,68:4,32:4,0:16,16:16,32:4
    index_entry_num3        = 10
    auto_extend_key_num3    = 20
    max_auto_extend_count3  = 10
    init_key_num3           = 1000
    auto_release_index3     = yes
    index_name4             = DIOSA_DELAYED_STRM_05_IDX
    keys4                   = 0:16,68:4,128:8,0:16,16:16,32:4
    index_entry_num4        = 10
    auto_extend_key_num4    = 20
    max_auto_extend_count4  = 10
    init_key_num4           = 1000
    auto_release_index4     = yes
    index_name5             = DIOSA_DELAYED_STRM_06_IDX
    keys5                   = 0:16,32:4,0:16,16:16,32:4
    index_entry_num5        = 10
    auto_extend_key_num5    = 20
    max_auto_extend_count5  = 10
    init_key_num5           = 1000
    auto_release_index5     = yes
```

(d) DIOSA_DELAYED_STACK

```
[table]
    tam_table_name          = TAM テーブル名
    tam_table_file          = TAM ファイル名
    backup                  = yes
    share                   = yes
    auto_extend             = yes
    auto_extend_size        = 1
    max_auto_extend_size    = 5
    record_size             = 48
    init_record_num         = 1000
    index_number            = 3
    index_name1             = DIOSA_DELAYED_STACK_00_IDX
    keys1                   = 0:16,16:4
    index_entry_num1        = 10
    auto_extend_key_num1    = 20
    max_auto_extend_count1  = 10
    init_key_num1           = 1000
    auto_release_index1     = yes
    index_name2             = DIOSA_DELAYED_STACK_01_IDX
    keys2                   = 0:16,0:16,16:4
    index_entry_num2        = 10
    auto_extend_key_num2    = 20
    max_auto_extend_count2  = 10
    init_key_num2           = 1000
    auto_release_index2     = yes
    index_name3             = DIOSA_DELAYED_STACK_02_IDX
    keys3                   = 0:16,20:4,0:16,16:4
    index_entry_num3        = 10
    auto_extend_key_num3    = 20
    max_auto_extend_count3  = 10
    init_key_num3           = 1000
    auto_release_index3     = yes
```

(e) DIOSA_DELAYED_STACKDATANO

```
[table]
    tam_table_name          = TAM テーブル名
    tam_table_file          = TAM ファイル名
    backup                  = yes
    share                   = yes
    auto_extend             = yes
    auto_extend_size        = 1
    max_auto_extend_size    = 5
    record_size             = 88
    init_record_num         = 1000
    index_number            = 6
    index_name1             = DIOSA_DELAYED_STKDATANO_00_IDX
    keys1                   = 0:16,16:16,32:4,36:4
    index_entry_num1        = 10
    auto_extend_key_num1    = 20
    max_auto_extend_count1  = 10
    init_key_num1           = 1000
    auto_release_index1     = yes
    index_name2             = DIOSA_DELAYED_STKDATANO_01_IDX
    keys2                   = 0:16,16:16,36:4,0:16,16:16,32:4,36:4
    index_entry_num2        = 10
    auto_extend_key_num2    = 20
    max_auto_extend_count2  = 10
    init_key_num2           = 1000
    auto_release_index2     = yes
    index_name3             = DIOSA_DELAYED_STKDATANO_02_IDX
    keys3                   = 0:16,16:16,32:4,0:16,16:16,32:4,36:4
    index_entry_num3        = 10
    auto_extend_key_num3    = 20
    max_auto_extend_count3  = 10
    init_key_num3           = 1000
    auto_release_index3     = yes
    index_name4             = DIOSA_DELAYED_STKDATANO_03_IDX
    keys4                   = 0:16,0:16,16:16,32:4,36:4
    index_entry_num4        = 10
    auto_extend_key_num4    = 20
    max_auto_extend_count4  = 10
    init_key_num4           = 1000
    auto_release_index4     = yes
    index_name5             = DIOSA_DELAYED_STKDATANO_04_IDX
    keys5                   = 0:16,16:16,32:4,40:8,0:16,16:16,32:4,36:4
    index_entry_num5        = 10
    auto_extend_key_num5    = 20
    max_auto_extend_count5  = 10
    init_key_num5           = 1000
    auto_release_index5     = yes
    index_name6             = DIOSA_DELAYED_STKDATANO_05_IDX
    keys6                   = 0:16,36:4,0:16,16:16,32:4,36:4
    index_entry_num6        = 10
    auto_extend_key_num6    = 20
    max_auto_extend_count6  = 10
    init_key_num6           = 1000
    auto_release_index6     = yes
```

(f) DIOSA_DELAYED_UNIT

```
[table]
    tam_table_name          = TAM テーブル名
    tam_table_file          = TAM ファイル名
    backup                  = yes
    share                   = yes
    auto_extend             = yes
    auto_extend_size        = 1
    max_auto_extend_size    = 5
    record_size             = 96
    init_record_num         = 1000
    index_number            = 2
    index_name1             = DIOSA_DELAYED_UNIT_00_IDX
    keys1                   = 0:16,16:1,17:16
    index_entry_num1        = 10
    auto_extend_key_num1    = 20
    max_auto_extend_count1  = 10
    init_key_num1           = 1000
    auto_release_index1     = yes
    index_name2             = DIOSA_DELAYED_UNIT_01_IDX
    keys2                   = 0:16,0:16,16:1,17:16
    index_entry_num2        = 10
    auto_extend_key_num2    = 20
    max_auto_extend_count2  = 10
    init_key_num2           = 1000
    auto_release_index2     = yes
```

(g) DIOSA_DELAYED_POOL_xxxx

```
[table]
    tam_table_name          = TAM テーブル名
    tam_table_file          = TAM ファイル名
    backup                  = yes
    share                   = yes
    auto_extend             = yes
    auto_extend_size        = 2
    max_auto_extend_size    = 2000
    record_size             = 4152
    init_record_num         = 1000
    index_number            = 1
    index_name1             = DIOSA_DELAYED_POOL_00_IDX
    keys1                   = 24:28,6:2
    index_entry_num1        = 20
    auto_extend_key_num1    = 500
    max_auto_extend_count1  = 0
    init_key_num1           = 1000
    auto_release_index1     = yes
```

(h) DIOSA_SENDER_UNIT

```
[table]
    tam_table_name          = TAMテーブル名
    tam_table_file          = TAMファイル名
    backup                  = yes
    share                   = yes
    auto_extend             = yes
    auto_extend_size        = 1
    max_auto_extend_size    = 5
    record_size             = 248
    init_record_num         = 1000
    index_number            = 1
    index_name1             = DIOSA_SENDER_UNIT_00_IDX
    keys1                   = 0:32
    index_entry_num1        = 10
    auto_extend_key_num1    = 20
    max_auto_extend_count1  = 10
    init_key_num1           = 1000
    auto_release_index1     = yes
```

(i) DIOSA_SENDER_STRM

```
[table]
    tam_table_name          = TAMテーブル名
    tam_table_file          = TAMファイル名
    backup                  = yes
    share                   = yes
    auto_extend             = yes
    auto_extend_size        = 1
    max_auto_extend_size    = 5
    record_size             = 168
    init_record_num         = 1000
    index_number            = 1
    index_name1             = DIOSA_SENDER_STRM_00_IDX
    keys1                   = 0:48
    index_entry_num1        = 10
    auto_extend_key_num1    = 20
    max_auto_extend_count1  = 10
    init_key_num1           = 1000
    auto_release_index1     = yes
```

(j) DIOSA_RECEIVER_UNIT

```
[table]
    tam_table_name          = TAMテーブル名
    tam_table_file          = TAMファイル名
    backup                  = yes
    share                   = yes
    auto_extend             = yes
    auto_extend_size        = 1
    max_auto_extend_size    = 5
    record_size             = 200
    init_record_num         = 1000
    index_number            = 1
    index_name1             = DIOSA_RECEIVER_UNIT_00_IDX
    keys1                   = 0:32
    index_entry_num1        = 10
    auto_extend_key_num1    = 20
    max_auto_extend_count1  = 10
    init_key_num1           = 1000
    auto_release_index1     = yes
```

(k) DIOSA_RECEIVER_STRM

```
[table]
    tam_table_name          = TAMテーブル名
    tam_table_file          = TAMファイル名
    backup                  = yes
    share                   = yes
    auto_extend             = yes
    auto_extend_size        = 1
    max_auto_extend_size    = 5
    record_size             = 240
    init_record_num         = 1000
    index_number            = 1
    index_name1             = DIOSA_RECEIVER_STRM_00_IDX
    keys1                   = 0:48
    index_entry_num1        = 10
    auto_extend_key_num1    = 20
    max_auto_extend_count1  = 10
    init_key_num1           = 1000
    auto_release_index1     = yes
```

(l) DIOSA_LOGREADER_UNIT

```
[table]
    tam_table_name          = TAMテーブル名
    tam_table_file          = TAMファイル名
    backup                  = yes
    share                   = yes
    auto_extend             = yes
    auto_extend_size        = 1
    max_auto_extend_size    = 5
    record_size             = 360
    init_record_num         = 1000
    index_number            = 1
    index_name1             = DIOSA_LOGREADER_UNIT_00_IDX
    keys1                   = 0:32
    index_entry_num1        = 10
    auto_extend_key_num1    = 20
    max_auto_extend_count1  = 10
    init_key_num1           = 1000
    auto_release_index1     = yes
```

(m) DIOSA_LOGREADER_STRM

```
[table]
    tam_table_name          = TAMテーブル名
    tam_table_file          = TAMファイル名
    backup                  = yes
    share                   = yes
    auto_extend             = yes
    auto_extend_size        = 1
    max_auto_extend_size    = 5
    record_size             = 160
    init_record_num         = 1000
    index_number            = 1
    index_name1             = DIOSA_LOGREADER_STRM_00_IDX
    keys1                   = 0:48
    index_entry_num1        = 10
    auto_extend_key_num1    = 20
    max_auto_extend_count1  = 10
    init_key_num1           = 1000
    auto_release_index1     = yes
```

4.2.3　　Oracle 環境定義

プールファイルのログデータ格納先、またはログリーダユニットのユーザデータ更新先に DB を指定した場合、対応する Oracle の制御表や、ストアドプロシージャの生成が必要です。

生成が必要となる条件は以下の通りです。

対象	定義が必要となる条件
ストアドプロシージャ	POOLTYPE=DB のスーパーストリームが定義されている
DIOSA_DSAM	
DIOSA_DELAYED_SPST	
DIOSA_DELAYED_UNIT	
DIOSA_DELAYED_STRM	
DIOSA_DELAYED_HISTORY	
DIOSA_DELAYED_STACK	
DIOSA_DELAYED_STACKDATANO	
DIOSA_DELAYED_FAIL_BKUP	
DIOSA_DELAYED_POOL	
DIOSA_DELAYED_STRMVIEW	
DIOSA_DELAYED_LOCATION	
DIOSA_DELAYED_LOCATION_SPST	
DIOSA_SENDER_UNIT	POOLTYPE=DB のスーパーストリームにセンダユニットが定義されている
DIOSA_SENDER_STRM	
DIOSA_RECEIVER_UNIT	POOLTYPE=DB のスーパーストリームにレシーバユニットが定義されている
DIOSA_RECEIVER_STRM	
DIOSA_LOGREADER_UNIT	ユーザデータ更新先が DB のログリーダユニットが定義されている
DIOSA_LOGREADER_STRM	

生成は、デフォルト RGSET として定義されているインスタンスに対しておこなってください。ただし、ログリーダの制御表(DIOSA_LOGREADER_UNIT、DIOSA_LOGREADER_STRM)については、ユーザデータ更新先として定義されている全てのインスタンス上に生成する必要があります。

(1)　Oracle 表生成

Oracle 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)　パーティション定義の記述修正

プールファイル制御に必要な制御表生成ファイル(create_table_dtd.sql)には、パーティション名が仮の定義で記述されています。(SPST01, SPST02 というスーパーストリーム名にそれぞれ 5 個のスタックファイルが割り当てられています。)

テーブル作成前に、ファイルを定義したスーパーストリーム名、スタックファイル数に合わせて修正してください。

命名規則は、パーティション名が、DIOSA_POOL_{スーパーストリーム名}_{スタック番号(0 埋め 2 桁)}、パーティション ID が、{スーパーストリーム名}_{スタック番号(0 埋め 2 桁)}です。

［サンプルのプールファイル定義］

```
CREATE TABLE DIOSA_DELAYED_POOL
  (
   STRM_NAME     VARCHAR2(16),
   DATA_NO       NUMBER(19)    DEFAULT 0,
   DIV_ID        NUMBER(10)    DEFAULT 1,
   PART_ID       VARCHAR2(19),
   USER_DATA_NO  NUMBER(19)    DEFAULT 0,
   DATA_LEN      NUMBER(6)     DEFAULT 0,
   USER_DATA_LEN NUMBER(10)    DEFAULT 0,
   SPLIT_TOTAL   NUMBER(7)     DEFAULT 0,
   SPLIT_NO      NUMBER(7)     DEFAULT 0,
   USER_AREA     RAW(8),
   UP_DATE       DATE,
   LOG_HEAD      RAW(400),
   LOG_DATA01    RAW(2000),

     (中略)

   LOG_DATA16    RAW(2000),
   CONSTRAINT DIOSA_DELAYED_POOL_PK
     PRIMARY KEY (STRM_NAME, DATA_NO, DIV_ID, PART_ID) USING INDEX LOCAL
      (                                                    赤字部分が編集対象箇所
       PARTITION DIOSA_POOL_SPST01_01,
       PARTITION DIOSA_POOL_SPST01_02,
       PARTITION DIOSA_POOL_SPST01_03,
       PARTITION DIOSA_POOL_SPST01_04,
       PARTITION DIOSA_POOL_SPST01_05,
       PARTITION DIOSA_POOL_SPST02_01,
       PARTITION DIOSA_POOL_SPST02_02,
       PARTITION DIOSA_POOL_SPST02_03,
       PARTITION DIOSA_POOL_SPST02_04,
       PARTITION DIOSA_POOL_SPST02_05
      )
  )
  PARTITION BY LIST (PART_ID)
  (
   PARTITION DIOSA_POOL_SPST01_01 VALUES ('SPST01_01')   INITRANS 1   MAXTRANS 255,
   PARTITION DIOSA_POOL_SPST01_02 VALUES ('SPST01_02')   INITRANS 1   MAXTRANS 255,
   PARTITION DIOSA_POOL_SPST01_03 VALUES ('SPST01_03')   INITRANS 1   MAXTRANS 255,
   PARTITION DIOSA_POOL_SPST01_04 VALUES ('SPST01_04')   INITRANS 1   MAXTRANS 255,
   PARTITION DIOSA_POOL_SPST01_05 VALUES ('SPST01_05')   INITRANS 1   MAXTRANS 255,
   PARTITION DIOSA_POOL_SPST02_01 VALUES ('SPST02_01')   INITRANS 1   MAXTRANS 255,
   PARTITION DIOSA_POOL_SPST02_02 VALUES ('SPST02_02')   INITRANS 1   MAXTRANS 255,
   PARTITION DIOSA_POOL_SPST02_03 VALUES ('SPST02_03')   INITRANS 1   MAXTRANS 255,
   PARTITION DIOSA_POOL_SPST02_04 VALUES ('SPST02_04')   INITRANS 1   MAXTRANS 255,
   PARTITION DIOSA_POOL_SPST02_05 VALUES ('SPST02_05')   INITRANS 1   MAXTRANS 255
  )
  NOCACHE
;
```

(b) 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、サンプルの下記のファイルに記述されています。

create_table_dtd.sql については、プールファイルのパーティション定義を更新したファイルを実行します。

対象	生成用ファイル
DIOSA_DSAM	create_table_dtd.sql
DIOSA_DELAYED_SPST	
DIOSA_DELAYED_UNIT	
DIOSA_DELAYED_STRM	
DIOSA_DELAYED_HISTORY	
DIOSA_DELAYED_STACK	
DIOSA_DELAYED_STACKDATANO	
DIOSA_DELAYED_FAIL_BKUP	
DIOSA_DELAYED_POOL	
DIOSA_DELAYED_STRMVIEW	
DIOSA_DELAYED_LOCATION	create_table_dlt.sql
DIOSA_DELAYED_LOCATION_SPST	
DIOSA_SENDER_UNIT	create_table_dts.sql
DIOSA_SENDER_STRM	
DIOSA_RECEIVER_UNIT	create_table_dtr.sql
DIOSA_RECEIVER_STRM	
DIOSA_LOGREADER_UNIT	create_table_dtl.sql
DIOSA_LOGREADER_STRM	

(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 {DIOSA/XTP インストールディレクトリ}/sql/{環境の文字コード}
> sqlplus Oracle ログイン情報 @DTD/create_dtd.sql
```

(3) その他 DIOSA/XTP 関連定義

(a) DB 接続管理機能

データストア基盤の DB へのアクセスには、BD 接続管理機能の環境定義(DBCTRL 節)が必要です。DELAYED 節に記述するリソースグループセット名や、コマンドパラメータで指定するインスタンスグループ名は、DBCTRL 節に記述されている必要があります。

詳細は DIOSA/XTP 利用の手引きの記述を参照してください。

4.2.4　PostgreSQL の環境定義

DBCTRL 節で PostgreSQL を使用する設定を行って、ログリーダユニットのユーザデータ更新先に DB を指定した場合、対応する PostgreSQL の制御表が必要です。

生成が必要となる条件は以下の通りです。

対象	定義が必要となる条件
DIOSA_LOGREADER_UNIT	ユーザデータ更新先が DB のログリーダユニットが定義されている
DIOSA_LOGREADER_STRM	

ユーザデータ更新先として定義されている全てのインスタンス上に生成する必要があります。

(1)　PostgreSQL 表生成

Oracle 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_pg 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、サンプルの下記のファイルに記述されています。

対象	生成用ファイル
DIOSA_LOGREADER_UNIT	create_table_dtl_pg.sql
DIOSA_LOGREADER_STRM	

(2)　その他 DIOSA/XTP 関連定義

(a)　DB 接続管理機能

データストア基盤の DB へのアクセスには、BD 接続管理機能の環境定義(DBCTRL 節)が必要です。DELAYED 節に記述するリソースグループセット名や、コマンドパラメータで指定するインスタンスグループ名は、DBCTRL 節に記述されている必要があります。

詳細は DIOSA/XTP 利用の手引きの記述を参照してください。

4.2.5　　Java 環境設定

Java アプリケーションで、ログデータ出力をおこなう場合、以下の設定が必要です。

(1)　クラスパスの設定

以下のファイルをクラスパスに追加してください。

diosadelayed.jar

commons-logging-1.1.1.jar

ojdbc6.jar

(2)　送信電文サイズ指定

Java アプリケーションから登録するログデータを送信するセンダの送信電文最小サイズを環境変数に設定します。

```
setenv DIOSA_DTDMSGSIZE 最小送信電文長
```

(3)　その他設定

(a)　log4j を使用する場合の設定

commons-logging.properties ファイル、log4j.xml ファイルを記述します。

commons-logging.properties ファイル

```
org.apache.commons.logging.Log=org.apache.commons.logging.impl.Log4JLogger
```

log4j.xml ファイル

ログファイルの絶対パスには任意のディレクトリを指定可能です。

ログ出力レベルは、出力レベルの高い順に、fatal/error/warn/info/debug のいずれかを指定します。

下記では info を指定しているので、debug 以外の全てのログが出力されます。

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE log4j:configuration SYSTEM "log4j.dtd">
<log4j:configuration xmlns:log4j="http://jakarta.apache.org/log4j/" >
  <appender name="ApplicationLog" class="org.apache.log4j.FileAppender">
    <param name="File" value="(ログファイルの絶対パス)"/>
    <param name="Append" value="true"/>
    <layout class="org.apache.log4j.PatternLayout">
      <param name="ConversionPattern" value="%d %-5p %-52c [%-12t] %m (%F:%L)%n"/>
    </layout>
  </appender>
  <logger name="com.nec.jp.diosa" >
    <level value ="info" />
    <appender-ref ref="ApplicationLog" />
  </logger>
</log4j:configuration>
```

※log4j の詳細は、Apache log4j のホームページをご参照ください。

(b)　simplelog を使用する場合の設定

simplelog.properties ファイルを記述します。

simplelog.properties ファイル

ログ出力レベルは、出力レベルの高い順に、fatal/error/warn/info/debug/trace のいずれかを指定します。

下記では info を指定しているので、trace/debug 以外の全てのログが出力されます。

```
org.apache.commons.logging.simplelog.defaultlog=info
```

※simpleog の詳細は、Apache commons logging のホームページをご参照ください。

4.2.6　　WebOTX の環境定義

WebOTX 上で動作するアプリケーションで、ログデータ出力をおこなう場合、Java 環境設定に加え、以下の設定が必要です。

(a)　セキュリティポリシーファイル(server.policy)の更新

対象ドメイン配下の config/server.policy ファイルに、以下の記述を追加してください。

```
grant {
      permission java.lang.RuntimePermission "getenv. DIOSA_DTDMSGSIZE";
};
```

(b)　メッセージ出力用設定

log4j を使用する場合

対象ドメイン/config ディレクトリ配下に、commons-logging.properties ファイル、および log4j.xml ファイルを格納します。

simplelog を使用する場合

対象ドメイン/config ディレクトリ配下に、simplelog.properties ファイルを格納します。

第5章　システムの運用

5.1　起動・停止

5.1.1　データストア基盤の起動

データストア基盤の起動は、DIOSA/XTP の基本機能やメモリキャッシュ機能を起動後におこないます。
それぞれの起動については、各機能の利用の手引きを参照してください。

(1)　環境定義オブジェクト作成

環境構築で作成した、DIOSA/XTP の環境定義ファイルは、環境定義オブジェクトに変換することで使用可能となります。各ノードで最初のオブジェクト作成時のみ diirmadd コマンド、それ以降のオブジェクト更新は diirmrep コマンドを実行して、環境定義オブジェクトを作成してください。

```
> diirmadd -E 環境定義ファイル名
> diirmrep -E 環境定義ファイル名
```

コマンドはディレード起動をおこなう全てのノードで実行する必要があります。

1 回オブジェクトを作成したら、定義内容を変更するまでは DIOSA/XTP を停止したりした場合でも、再作成する必要はありません。

(2)　制御データの初期化

制御データの初期化は、環境定義情報に合わせて制御テーブル上に必要な情報を作成します。論理システム運用開始時に 1 回だけ実施してください。

```
> didltcreate
```

以降は DIOSA/XTP を再起動しても、DB やメモリキャッシュ上のデータを引き継いで動作しますので、制御データの初期化は不要です。

コマンドは、利用するデータベースに全てアクセス可能なノードで実行する場合は、論理システム内で 1 回実行すれば全ての制御データを初期化します。DB を使用するが、OLTP ノード上ではメモリキャッシュへのアクセスしかできない等、全てのデータベースにアクセスできるノードが存在しない場合は、それぞれのノードで対象データベースを指定して実行する必要があります。

```
> didltcreate -D im       (メモリキャッシュ上の制御データのみ初期化)
> didltcreate -D db       (DB 上の制御データのみ初期化)
```

制御データの初期化は、初回については後述するデータストア基盤起動後でも実行可能ですが、2 回目以降の初期化をデータストア基盤起動後に実行する場合、ストリーム所在管理機能を停止させて実行する必要があります。

[停止(初期化前におこなう)]

```
> didltctrl -e -F locationm       (DB ノードで実行)
> didltctrl -e -F locationa       (AP ノード、および DBLAYER に指定したノードで実行)
```

［再起動(初期化後におこなう)］

```
> didltctrl -b -F locationm        (DB ノードで実行)
> didltctrl -b -F locationa        (AP ノード、および DBLAYER に指定したノードで実行)
```

(3) データストア基盤起動

起動処理は、プールファイル制御、通信制御機能、ストリーム所在管理機能を起動させるディレード起動コマンドと、センダ/レシーバ/ログリーダの各ユニットを開始するコマンドがあります。

(a) ディレード起動

ディレード起動は、初回起動時のみ共有メモリを環境定義ファイルから作成し、以降は前回起動時の情報を引き継いで起動します。

```
> didltinit -M create     (初回起動時)
> didltinit               (前回起動時情報引き継ぎ)
```

引き継ぎ情報がない場合、自動的に環境定義ファイルを読み込んで起動するため、常にオプションなしで起動する運用でも構いません。

```
> didltinit
```

ディレード起動コマンドまで実行すれば、プールファイルへのログデータ登録が可能となります。

ディレード起動コマンドの実行が必要な論理ノード種別と、必要となる条件は下記の通りです。該当論理ノード種別の全てのノードでコマンドを実行してください。(必要ないノードで起動コマンドを実行しても構いません。)

論理ノード種別	実行条件
DB	プールファイル種別=DB のプールファイルを定義している
AP	センダ/レシーバユニットを定義している または、$DELAYED-%COMMON-DBLAYER=AP と定義している または、AP ノード上でログデータ登録処理をおこなう
OLTP	プールファイル種別=IM のプールファイルを定義している または、$DELAYED-%COMMON-DBLAYER=OLTP と定義している または、OLTP ノード上でログデータ登録処理をおこなう

(b) ユニット開始

センダ/レシーバ/ログリーダユニットの開始は、それぞれの開始コマンドを実行する必要があります。

```
> didtsinit               (センダ開始)
> didtrinit               (レシーバ開始)
> didtlinit               (ログリーダ開始)
```

各ユニットを開始すると、それぞれのデータ処理開始コマンド等を実行可能となります。

ユニット開始コマンドは、各論理ノードで動作するスーパーストリームに、該当ユニットが定義されている場合に実行する必要があります。該当論理ノード種別の全てのノードでコマンドを実行してください。

スーパーストリームの動作ノードは下記の条件で決定します。

論理ノード種別	動作するスーパーストリーム
AP	・ プールファイル種別=DB のプールファイルを処理する かつ、$DELAYED-%COMMON-DBLAYER=AP と定義している
OLTP	・ プールファイル種別=DB のプールファイルを処理する かつ、$DELAYED-%COMMON-DBLAYER=OLTP と定義している ・ プールファイル種別=IM のプールファイルを処理する

(4) ログデータ処理開始

ユニットを開始後、それぞれのログデータ処理開始をおこなうと、ログデータ転送、ログデータ処理を開始します。

(a) ログデータ転送制御

ログデータ転送制御は、転送をおこなうセンダ/レシーバの両方に、転送開始コマンドを実行することで開始状態となります。-A は、無効化されていない、コマンド実行ノードで動作しているスーパーストリーム全てが対象となります。論理ノード種別の全てのノードでコマンドを実行してください。

```
> didtsstart -A          (センダ転送開始)
> didtrstart -A          (レシーバ転送開始)
```

ログデータ転送制御は、転送元のセンダユニット、転送先のレシーバユニットに転送開始コマンドを実行し、論理システム間で、制御データをやり取りして初めて開始状態となります。

通信制御機能が、論理システム間・論理システム内の通信パスが全て通信可能と判断するまで、制御データのやり取りができないため、起動直後に転送開始を実行した場合、すぐには転送開始できない場合があります。その場合でも、ネットワーク障害等が発生していなければ、自動的にリトライして開始状態になるため、特にコマンド再実行等の操作は不要です。

(b) ログデータ実行制御

ログデータ実行制御は、ログリーダユニットにデータ処理開始コマンドを実行すれば開始します。-A は、無効化されていない、コマンド実行ノードで動作しているスーパーストリーム全てが対象となります。論理ノード種別の全てのノードでコマンドを実行してください。

```
> didtlstart -A          (ログリーダデータ処理開始)
```

5.1.2　　データストア基盤の停止

データストア基盤の停止は、DIOSA/XTP の基本機能やメモリキャッシュ機能の停止前におこないます。
それぞれの停止については、各機能の利用の手引きを参照してください。

(1)　ログデータ処理停止

ログデータ転送、ログデータ処理を停止します。

(a)　ログデータ転送制御

ログデータ転送制御は、転送をおこなうセンダ/レシーバの両方に、転送停止コマンドを実行することで停止状態となります。-A は、無効化されていない、コマンド実行ノードで動作しているスーパーストリーム全てが対象となります。論理ノード種別の全てのノードでコマンドを実行してください。

```
> didtsstop -A          (センダ転送停止)
> didtrstop -A          (レシーバ転送停止)
```

ログデータ転送制御は、転送元のセンダユニット、転送先のレシーバユニットに転送停止コマンドを実行すると転送相手と制御データをやり取りして停止を通知します。

(b)　ログデータ実行制御

ログデータ実行制御は、ログリーダユニットにデータ処理停止コマンドを実行すれば停止します。-A は、無効化されていない、コマンド実行ノードで動作しているスーパーストリーム全てが対象となります。論理ノード種別の全てのノードでコマンドを実行してください。

```
> didtlstop -A          (ログリーダデータ処理停止)
```

(2)　データストア基盤停止

停止処理は、プールファイル制御、通信制御機能、ストリーム所在管理機能を起動させるディレード停止コマンドと、センダ/レシーバ/ログリーダの各ユニットを終了するコマンドがあります。
全ユニットが終了状態でないと、ディレード停止を実行することはできません。

(a)　ユニット終了

センダ/レシーバ/ログリーダユニットの終了は、それぞれの終了コマンドを実行する必要があります。

```
> didtsterm             (センダ終了)
> didtrterm             (レシーバ終了)
> didtlterm             (ログリーダ終了)
```

ログデータ処理停止を含めて実行したい場合は、ログデータ処理停止のオプションをつけてコマンドを実行すると、ログデータ処理停止＋ユニット終了をおこないます。

```
> didtsterm -M stop     (センダ終了(ログデータ転送停止オプション))
> didtrterm -M stop     (レシーバ終了(ログデータ転送停止オプション))
> didtlterm -M stop     (ログリーダ終了(ログデータ処理停止オプション))
```

ユニット終了コマンドは、ユニット開始コマンドを実行した全てのノードでコマンドを実行してください。

(b) ディレード停止

ディレード停止コマンドを実行します。

```
> didltterm
```

ディレード起動を実行した全てのノードでコマンドを実行してください。

5.2　ディビジョン切り替え

5.2.1　　ディビジョン切り替え

ディビジョン切り替えは、プールファイル種別=WRITER のプールファイルに対してディビジョン切り替えコマンドを実行することでおこないます。コマンドはスーパーストリーム単位に実行します。

(a) ディビジョン切り替え

```
> didltdivchg -s スーパーストリーム名
```

コマンド実行により、ディビジョン ID が 1 増加し、それ以降のログデータ登録は新しいディビジョンに対しておこなわれます。

また、切り替え前のディビジョンのログデータを全て転送/処理完了すると、センダ/レシーバ/ログリーダの各ユニットは、ディビジョン終了状態となり、転送/処理が停止します。

5.2.2　　次ディビジョン開始

ディビジョン終了状態となったセンダ/レシーバ/ログリーダの各ユニットは、ログデータ処理開始をおこなうことで、次ディビジョンの処理を開始します。

次ディビジョンの開始は、「5.1.1 (4) ログデータ処理開始」と同様の手順でおこないます。

5.3　環境変更

5.3.1　データストア基盤

DELAYED 節の環境定義変更は以下の手順でおこないます。

(1)　センダ、レシーバ、ログリーダ停止

センダ、レシーバ、ログリーダが起動している状態では環境定義変更コマンドは実行できません。

全ての動作ノードで機能停止コマンドを実行してください。

(2)　環境定義ファイル修正

DELAYED 節の定義内容を記述したファイルを、変更後の内容に修正します。

(3)　環境定義オブジェクトファイル更新

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
```

コマンドはディレード起動をおこなう全てのノードで実行する必要があります。

(4)　制御テーブル準備

定義変更の実施内容によっては、制御テーブルを追加/更新する必要があります。

(a)　スーパーストリーム追加、スタックファイル追加

追加スーパーストリームが使用するプールファイルを追加する必要があります。

ログデータ格納先が IM の場合はプールファイル用 TAM 表追加、DB の場合はプールファイルにパーティション追加をおこないます。

スタックファイル数を増やす場合も同様に追加スタック分のプールファイルを追加する必要があります。

(5)　定義変更実施

環境定義オブジェクトファイルを更新したら、定義変更コマンドを実行します。

```
> didltupd
```

コマンドは定義を変更したスーパーストリームのアクセス先データベースが更新可能なノードで実行してください。

定義変更コマンドは、論理システム内の 1 ノードで実行すれば完了です。また、各ノードの共有メモリを更新する処理は、定義変更コマンドから各ノードにコマンド配信して自動的におこないますので、定義反映処理の完了メッセージまで出力されていれば、各ノードでのコマンド実行は不要です。

コマンド配信処理のエラー等で、定義変更は完了しているが、定義反映に失敗している場合、定義反映コマンドを全ノード配信のオプションで別途実行する必要があります。

```
> didltchg -A
```

［注意］

通信制御機能の定義(DELAYED節-PATHCTRL項)、およびストリーム所在管理機能の定義(DELAYED節-LOCATIONCTRL項)のみを変更した場合、定義変更コマンドでは定義変更されません。その場合は、定義反映コマンドを全ノード配信のオプションで実行してください。

```
> didltchg -A
```

(6) 定義変更後処理

既存のスーパーストリームにユニットを追加した場合や、ログリーダユニットのユーザデータ更新先変更、異なるインスタンスグループへのリソースグループセット変更をおこなった場合、ユニットの処理対象ログデータ通番を、プールファイルのディビジョン ID、通番に合わせて更新する必要があります。

```
> didtsmod -T divid=ディビジョン ID -T datano=通番 -T userdatano=ユーザ通番
          -s スーパーストリーム名                                    (センダユニット通番変更)
> didtlmod -T divid=ディビジョン ID -T datano=通番 -T userdatano=ユーザ通番
          -s スーパーストリーム名［-u ユニット名］                    (ログリーダユニット通番変更)
```

(7) センダ、レシーバ、ログリーダ開始

センダ、レシーバ、ログリーダを開始します。

全ての動作ノードで機能開始コマンドを実行してください。

5.3.2 DIOSA/XTP 基盤機能

DIOSAMAP 節､SYSMAP 節の定義を変更した場合､定義反映コマンドを-o オプションで全ノードに配信してください。

コマンドは、DIOSAMAP/SYSMAP のローリングアップデートにより全ノード変更後に実行してください。

```
> didltchg -A -o
```

論理ノードを削除する場合、削除対象の論理ノード閉塞後、DIOSA/XTP またはディレード転送機能を停止してください。

```
> didtsterm -M stop
> didtrterm -M stop
> didtlterm -M stop
> didltterm
```

DB ノードを追加する場合、上記定義反映コマンド実行後、AP ノードおよび$DELAYED 節-COMMON 項-DBLAYER パラメータに指定したノード種別のノードで、ストリーム所在管理機能のエージェントデーモンを再起動してください。

```
> didltctrl -e -F locationa
> didltctrl -b -F locationa
```

5.3.3 環境変更に関する注意事項

(1) データストア基盤

- $DELAYED-$COMMON-FAILOVER パラメータを変更した場合、定義反映後に各ノードのディレード制御デーモン

を再起動してください。

```
> didltctrl -e
> didltctrl -b
```

- 定義変更コマンド実行前に、変更後の定義を配布したノードで定義反映コマンドを手動で実行した場合、以下の対処が必要です。
  - DB ノードで実行した場合は、プールファイル格納先が DB のスーパーストリームを定義している場合は、ストリーム所在管理機能のマネージャデーモンを再起動してください。

```
> didltctrl -e -F locationm
> didltctrl -b -F locationm
```

- ユニットのステータスが INIT, INACT, DIVEND 以外の場合、定義を変更することはできません。
- 以下のパラメータは、条件によらず定義変更することができません。
  - $DELAYED-$SUPERSTREAM-GROUP
  - $DELAYED-$SUPERSTREAM-ALIAS
- 以下の状態でスーパーストリームやユニットを削除することはできません。
  - ユニットのステータスが INIT, DIVEND 以外
  - 未処理データがある(ただし、ステータスが INIT, DIVEND の場合は強制モードを指定すれば削除可能)
- 下記の定義を変更した場合、スーパーストリームの制御情報は全て初期化されるため、定義生成直後と同じ状態になります。また、これらの変更は、上記削除可能条件と同じ条件を満たしている必要があります。
  - プールファイル種別(WRITER/RECEIVER)
  - ログデータ格納先、ユーザデータ更新先(IM/DB)
  - スタックファイル数
  - MAPID
  - RGSET(ログリーダユニットの RGSET を異なるインスタンスグループ配下の RGSET へ変更した場合)

5.4　障害対応

5.4.1　　ノード障害

(1)　APノード / OLTPノード

AP ノード/OLTP ノードが停止した場合、同一論理ノード種別のノードが 1 つ以上稼働していれば、処理を実行するノードを移動したり、データ転送経路を変更したりして処理を継続することができます。

また、障害だったノードが復旧した場合、復旧したノードにデータ処理を分散したり、データ転送経路に組み込む制御は自動的におこなわれるため、特別なオペレーションは不要です。

同一論理ノード種別の全てのノードが障害となった場合、OLTP ノードでメモリキャッシュのプールファイルを使用していた場合は、制御情報やプールファイルが消失してしまいます。データを復旧できる場合は DIOSA/XTP 再起動後に、起動手順を実施してください。復旧できない場合は制御データの初期化からやり直す必要があります。

(2)　DBノード

DB ノードが停止した場合、停止したノードのインスタンスグループが RAC 構成で、もう一方の DB ノードが稼働していれば、処理を継続します。

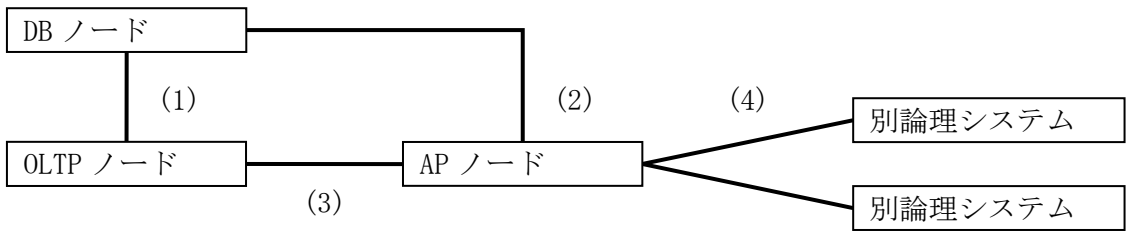
RAC 構成でない場合や、RAC 構成の全ての DB ノードが停止してしまった場合、DB アクセスできなくなるため、ログデータ格納先が DB のスーパーストリーム、およびユーザデータ更新先が DB のログリーダユニットは処理を停止します。

障害が復旧したタイミングで、ユニットが動作しているノード種別内の全論理ノードで、管理デーモンを再起動してください。

```
> didltctrl -e -F sender -M force   (センダ管理デーモン停止)
> didltctrl -b -F sender            (センダ管理デーモン起動)
> didltctrl -e -F receiver -M force (レシーバ管理デーモン停止)
> didltctrl -b -F receiver          (レシーバ管理デーモン起動)
> didltctrl -e -F reader -M force   (ログリーダ管理デーモン停止)
> didltctrl -b -F reader            (ログリーダ管理デーモン起動)
```

5.4.2　　通信パス障害

論理システム内のノード間、あるいは論理システム間の通信パスが切断された場合、以下のような動作となります。



(1)　OLTP ノード-DB ノード間

特定の DB ノード-OLTP ノード間の通信パスが障害となった場合、ログデータ格納先が DB のスーパーストリームが OLTP ノードで動作していた場合は、別の OLTP ノードに移動して処理を継続します。(ログデータ格納先がメモリキャッシュで、ユーザデータ更新先が DB のスーパーストリームは、メモリキャッシュの障害に連動して移動等をおこなうため、DB ノードとの通信パス障害では動作ノードは移動せず、データベース障害として処理を一時停止します。)

通信パスの復旧を検知すると、一部のスーパーストリームを復旧した OLTP ノードに移動して負荷を均等にします。

全てのパスが障害となると、DB ノード障害や後述するデータベース障害と同じ扱いとなるため、ログデータ格納先や、ユーザデータ更新先が DB の処理は一時停止します。

障害が復旧したタイミングで、ユニットが動作しているノード種別内の全論理ノードで、実行デーモン起動/停止コマンドを実行してください。

(2)　AP ノード-DB ノード間

特定の DB ノード-AP ノード間の通信パスが障害となった場合、ログデータ格納先が DB のスーパーストリームが AP ノードで動作していた場合は、別の AP ノードに移動して処理を継続します。

通信パスの復旧を検知すると、一部のスーパーストリームを復旧した AP ノードに移動して負荷を均等にします。

全てのパスが障害となると、DB ノード障害や後述するデータベース障害と同じ扱いとなるため、ログデータ格納先や、ユーザデータ更新先が DB の処理は一時停止します。

障害が復旧したタイミングで、ユニットが動作しているノード種別内の全論理ノードで、実行デーモン起動/停止コマンドを実行してください。

(3)　OLTP ノード-AP ノード間

特定の OLTP ノード-AP ノード間の通信パスが障害となった場合、メモリキャッシュを利用していると、DIOSA/XTP メモリキャッシュ機能によってノード障害と判定されるケースがあります。その場合、MAPID のマスタノードが移動するため、それに合わせてログデータ格納先がメモリキャッシュのスーパーストリームは、動作ノードを移動します。

マスタノードが移動しない場合、ログデータ転送制御は該当 AP ノードを経由せずにデータ転送を継続します。

障害復旧を検知すると、自動的に該当 AP ノードを経由したデータ転送を再開します。

(4)　論理システム間

論理システム間の通信パスが障害となった場合、複数の通信パスの一部だった場合は、通信可能なパスを使用して継続してログデータ転送をおこないます。

全ての通信パスが障害となった場合、ログデータ転送は一時停止しますが、障害復旧を検知すると、障害時にログデータ転送中であったスーパーストリームは自動的に転送を再開します。

5.4.3　　データベース障害

(1)　メモリキャッシュ

メモリキャッシュの障害は、DIOSA/XTP メモリキャッシュ機能で監視しています。

レプリケーショングループが複数の OLTP ノードに分散している場合、1 つの論理ノードで障害となっても、別の論理ノードに処理ノードを移動して処理を継続可能です。分散先の全ての論理ノードが障害となった場合は、該当レプリケーショングループに属する MAPID は、処理を継続できなくなります。該当 MAPID をログデータ格納先とするスーパーストリームは処理を停止します

障害が復旧したタイミングで、停止したユニットが動作している全論理ノードで、実行デーモン起動/停止コマンドを実行してください。

```
> didtsctrl （センダユニット起動）
> didtrctrl （レシーバユニット起動）
> didtlctrl （ログリーダユニット起動）
```

(2)　DB

DB の障害は、DIOSA/XTP データベース管理機能で監視しています。

デフォルトリソースグループセットの属するインスタンスグループが障害となった場合、ログデータ格納先が DB のスーパーストリーム全て、およびユーザデータ更新先に該当インスタンスグループを指定しているログリーダユニットが処理を停止します。

それ以外のインスタンスグループが障害となった場合、ユーザデータ更新先に該当インスタンスグループを指定しているログリーダユニットが処理を停止します。

障害が復旧したタイミングで、停止したユニットが動作しているノード種別内の全論理ノードで、管理デーモンを再起動してください。

```
> didltctrl -e -F sender -M force   （センダ管理デーモン停止）
> didltctrl -b -F sender            （センダ管理デーモン起動）
> didltctrl -e -F receiver -M force （レシーバ管理デーモン停止）
> didltctrl -b -F receiver          （レシーバ管理デーモン起動）
> didltctrl -e -F reader -M force   （ログリーダ管理デーモン停止）
> didltctrl -b -F reader            （ログリーダ管理デーモン起動）
```

データベース管理機能では、インスタンスグループの障害復旧のタイミングで、ユーザ作成のシェルスクリプトを呼び出す機能を提供しています。上記コマンドをシェルスクリプト内に記述することで、障害復旧時の自動処理再開が可能です。

(3)　メモリキャッシュの全障害からの復旧について

メモリキャッシュは全 OLTP ノードが障害になると、データ自体が消えてしまいます。この場合の復旧は、定義生成を実行してください。

メモリキャッシュのみ使用している環境や、DB と併用している場合でも、ログデータ格納先がメモリキャッシュ、ユーザデータ更新先が DB のログリーダユニットが存在しない場合は、任意の OLTP ノードで制御データの初期化をおこないます。

```
> didltcreate -D im
```

ログデータ格納先がメモリキャッシュ、ユーザデータ更新先が DB のログリーダユニットが存在する場合、上記手順の後で、該当スーパーストリームについてスーパーストリーム指定で定義生成を実行する必要があります。

コマンドは対象となるスーパーストリーム数分実行してください。

```
> didltcreate -s スーパーストリーム名
```

5.4.4　　無効化について

スーパーストリームに複数の処理ユニット(センダユニットまたはログリーダユニット)が定義されている場合、障害によって一部のユニットが動作不可となると、未処理データが蓄積し、プールファイルが満杯になることで、正常動作中のユニットまで処理継続不可となってしまいます。

このような場合、ユニットを無効化することで、障害となったユニットを切り離し、正常なユニットのみで処理を継続することが可能です。

```
> didltblock -b -s スーパーストリーム名 -F sender -u ユニット名　(センダユニット無効化)
> didltblock -b -s スーパーストリーム名 -F reader -u ユニット名　(ログリーダユニット無効化)
```

障害が復旧したら、無効化解除することで制御対象に組み込まれます。

```
> didltblock -a -s スーパーストリーム名 -F sender -u ユニット名　(センダユニット無効化解除)
> didltblock -a -s スーパーストリーム名 -F reader -u ユニット名　(ログリーダユニット無効化解除)
```

この際無効化していたユニットの処理対象データは既に削除されてしまっている可能性が高いため、プールファイル照会コマンドで現在残っているログデータのディビジョン ID や通番を確認し、存在するデータから処理を再開するように、動作変更コマンドでディビジョン ID、通番の情報を更新する必要があります。

```
> didtsmod -T divid=ディビジョン ID -T datano=通番 -T userdatano=ユーザ通番
          -s スーパーストリーム名                                    (センダユニット通番変更)
> didtlmod -T divid=ディビジョン ID -T datano=通番 -T userdatano=ユーザ通番
          -s スーパーストリーム名 [-u ユニット名]                    (ログリーダユニット通番変更)
```

付録A　リソース一覧
付録B　プロセス一覧
付録C　データベース一覧
付録D　諸元一覧

「DIOSA/XTP 導入の手引き」を参照してください。

DIOSA/XTP V3.1
データストア 利用の手引

2022年 12月 第2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)