HITACHI
Inspire the Next

Hitachi Streaming Data Platform
システム構築ガイド

3000-3-E23-31

前書き

■ 著作権

■ 輸出時の注意

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

なお、不明な場合は、弊社担当営業にお問い合わせください。

■ 商標類

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

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

This product includes software developed by IAIK of Graz University of Technology.

This product includes software developed by the Java Apache Project for use in the Apache JServ servlet engine project (http://java.apache.org/).

This product includes software developed by Daisuke Okajima and Kohsuke Kawaguchi (http://relaxngcc.sf.net/).

This product includes software developed by Andy Clark.

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

■ 発行

2020 年 7 月

変更内容

変更内容(3000-3-E23-31)　Hitachi Streaming Data Platform 03-30，Hitachi Streaming Data Platform software development kit 03-30

追加・変更内容	変更個所
記述不良を見直した。	-

単なる誤字・脱字などはお断りなく訂正しました。

はじめに

このマニュアルは，Hitachi Streaming Data Platform （Streaming Data Platform）の設計・構築・運用方法，および構築時に設定できる機能の詳細について説明したものです。Hitachi Streaming Data Platform （Streaming Data Platform）の設計・構築・運用をすることで，ストリームデータの分析ができるようになることを目的としています。

■ 対象製品

- P-9W64-9F31 Hitachi Streaming Data Platform 03-30 （適用 OS：Red Hat(R) Enterprise Linux(R) Server 6（64-bit x86_64），Red Hat(R) Enterprise Linux(R) Server 7（64-bit x86_64））
- P-9W64-9K31 Hitachi Streaming Data Platform software development kit 03-30 （適用 OS：Red Hat(R) Enterprise Linux(R) Server 6（64-bit x86_64），Red Hat(R) Enterprise Linux(R) Server 7（64-bit x86_64））

■ 対象読者

このマニュアルは，ソリューションプロジェクトマネージャー，ソリューションデベロッパー，インテグレーションデベロッパー，デプロイメントエンジニア，カスタマイズエンジニア，オペレーター，およびサポートエンジニアを対象にしています。

■ このマニュアルで使用する記号

このマニュアルで使用する記号を次に示します。

記号	意味
斜体	可変値，強調，呼び出しを示します。
＜ ＞	可変値（斜体で変数を表現しきれない場合に使用）
[ ]	任意の値であることを示します。
{ }	必要な値，または期待される値を示します。
| （ストローク）	複数の項目または引数から選択することを示します。
… （点線）	この記号の直前に示された項目を繰り返して指定できることを示します。

■ このマニュアルで使用する略語

このマニュアルで使用する製品名の表記

このマニュアルでは，製品名を次のように表記しています。

表記	製品名
HSDP	Hitachi Streaming Data Platform
Streaming Data Platform	
SDP	
HSDP software development kit	Hitachi Streaming Data Platform software development kit
Streaming Data Platform software development kit	
SDP SDK	
JavaVM	Java Virtual Machine
Red Hat Enterprise Linux Server	Red Hat(R) Enterprise Linux(R) Server

このマニュアルで使用する英略語

このマニュアルで使用する英略語を次に示します。

英略語	正式名称
AP	Application Program
API	Application Programming Interface
ASCII	American Standard Code for Information Interchange
BOM	Byte Order Mark
CPU	Central Processing Unit
CQL	Continuous Query Language
CSV	Comma Separated Value
EADs	Hitachi Elastic Application Data Store
EUC	Extended UNIX Code
EUC-JP	Extended UNIX Code Packed Format for Japanese
G1GC	Garbage First Garbage Collection
GC	Garbage Collection
GCC	GNU Compiler Collection

英略語	正式名称
GMT	Greenwich Mean Time
HTTP	Hyper Text Transfer Protocol
ID	Identifier
IP	Internet Protocol
JDBC	Java Database Connectivity
JIS	Japanese Industrial Standards
JRE	Java Runtime Environment
JST	Japan Standard Time
JVM	Java Virtual Machine
NIC	Network Interface Card
OS	Operating System
SJIS	Shift JIS
SMTP	Simple Mail Transfer Protocol
SNMP	Simple Network Management Protocol
TCP	Transmission Control Protocol
TCP/IP	Transmission Control Protocol/Internet Protocol
UDP/IP	User Datagram Protocol/Internet Protocol
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
UTF	UCS Transformation Format
VM	Virtual Machine
VTL	Velocity Template Language
XML	Extensible Markup Language

■ ¥の表記について

本文中で使用されている¥は半角のバックスラッシュを意味しています。

■ このマニュアルで使用する KB（キロバイト）などの単位表記

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

■ 関連マニュアル

関連マニュアルを次に示します。必要に応じてお読みください。

- 『Hitachi Streaming Data Platform 入門』 (3000-3-E21)
- 『Hitachi Streaming Data Platform 概説』 (3000-3-E22)
- 『Hitachi Streaming Data Platform アプリケーション開発ガイド』(3000-3-E24)
- 『Hitachi Streaming Data Platform メッセージ』 (3000-3-E25)

目次

1 Streaming Data Platform を設定する

この章では，Hitachi Streaming Data Platform（Streaming Data Platform）を設定する手順を説明します。ソリューションデベロッパーとインテグレーションデベロッパーは，Streaming Data Platform をインストールして，ユーザー環境を作成します。

1.1 Hitachi Streaming Data Platform をインストールする（full インストールおよび runtime インストール）

ソリューションデベロッパーおよびインテグレーションデベロッパーは，開発サーバで Streaming Data Platform を full インストールまたは runtime インストールする必要があります。HSDP に基づくユーザーパッケージをデプロイするために，またはすでに準備されているパッケージのシステム構成を変更するために Streaming Data Platform をインストールして，ユーザー環境を準備する必要があります。Streaming Data Platform と外部アダプター（オプション）を同じホストに設定する必要がある場合，HSDP の full インストールが必要です。full インストールが実行されていない別のホストに外部アダプターの設定だけを実施すればよい場合，ユーザーは HSDP を runtime インストールすることもできます。

前提条件

前提条件を次に示します。

システム

- ディスクに十分な空き容量があること。
- メモリに十分な空き容量があること。
- 前提ソフトウェアがホストにインストールされていること。

システム要件については，『Hitachi Streaming Data Platform リリースノート』を参照してください。

メモ

- Streaming Data Platform software development kit を使用する場合，Streaming Data Platform がインストールされているホストに Streaming Data Platform software development kit がインストールされている必要があります。そのため，Streaming Data Platform をインストールする前に，Streaming Data Platform software development kit のシステム要件も確認する必要があります。
- 仮想 OS に開発用ホストを設定することを推奨します。

ユーザー

- `root` ユーザーとしてホストにログインすること。
- 開発用ホストが設定されていること。

操作手順

1. 対応するドライブにインストール用 CD を挿入します。

   インストール用 CD は自動的にマウントされます。

   CD が自動的にマウントされない場合，次のコマンドを実行して手動でマウントします。

```
mount -t iso9660 /dev/cdrom CD-ROMマウントディレクトリ
```

2. 次のコマンドを実行して，インストーラーのあるディレクトリに移動します。

```
cd CD-ROMマウントディレクトリ/HSDP/
```

3. full インストールまたは runtime インストールを実行します。

full インストールの場合，次のインストーラーを実行します。

```
./install.sh
```

runtime インストールの場合，次のインストーラーを実行します。

```
./install_runtime.sh
```

- HSDP のインストールに成功すると，インストーラーが次のメッセージを標準出力（stdout）に出力します。

  full インストールの場合：

  ```
  Hitachi Streaming Data Platform installation completed successfully.
  ```

  runtime インストールの場合：

  ```
  Hitachi Streaming Data Platform runtime installation completed successfully.
  ```

- Streaming Data Platform は/opt/hitachi/hsdp/ディレクトリにインストールされます。
- Streaming Data Platform base は/opt/hitachi/hsdpbase/ディレクトリにインストールされます。
- Common Trace Library は/opt/hitachi/HNTRLib2/ディレクトリおよび/opt/hitachi/common/ディレクトリにインストールされます。
- runtime のディレクトリ/var/opt/hitachi/HNTRLib2/，/var/lib/hitachi/hsdp/，/var/log/hitachi/hsdp/が作成されます。
- Hitachi Streaming Data Platform の full インストールに失敗すると，インストーラーが次のメッセージを標準エラー出力（stderr）に出力します。

  ```
  Hitachi Streaming Data Platform full installation failed.
  ```

- Hitachi Streaming Data Platform の runtime インストールに失敗すると，インストーラーが次のメッセージを標準エラー出力（stderr）に出力します。

  ```
  Hitachi Streaming Data Platform runtime installation failed.
  ```

  エラーの対処方法については，マニュアル『Hitachi Streaming Data Platform メッセージ』を参照してください。

- Hitachi Streaming Data Platform のインストールに失敗した場合，システムおよびユーザーの前提条件が完全に満たされていることを確認し，このインストールを再実行する必要があります。

HSDP のインストーラーについては，『Hitachi Streaming Data Platform リリースノート』を参照してください。

4. インストールの終了後，インストール用 CD を取り外します。

```
# umount /dev/cdrom
```

次の作業

Streaming Data Platform のインストール後，次の操作を実行します。

full インストールの場合：

- SDP マネージャーを起動します。
- 外部アダプターのアーカイブファイルをインポートします（外部アダプターを使用する場合）。

runtime インストールの場合：

- 外部アダプターのアーカイブファイルをインポートします。

1.2 SDP マネージャーを起動する

Streaming Data Platform のインストール後，ソリューションデベロッパーまたはインテグレーションデベロッパーは SDP マネージャーを起動する必要があります。SDP マネージャーの起動時に，同じホストで利用可能な SDP ブローカーと SDP コーディネーターも自動的に起動します。

前提条件

前提条件を次に示します。

システム

Streaming Data Platform がインストールされていること。

ユーザー

ソリューションデベロッパーまたはインテグレーションデベロッパーが root ユーザーとして OS にログインしていること。

操作手順

1. SDP マネージャーを起動するには，次のコマンドを実行します。

```
/opt/hitachi/hsdp/bin/hsdpmanager -start
```

このコマンドを実行すると，同じホストの SDP ブローカーと SDP コーディネーターも自動的に起動します。SDP マネージャーは，正常に起動した場合にはメッセージKFHD92010-I を標準出力（stdout）に出力し，起動に失敗した場合にはエラーメッセージを標準エラー出力（stderr）に出力します。

次の作業

SDP マネージャーの起動後，運用ディレクトリを作成します。

1.3 運用ディレクトリを作成する

ソリューションデベロッパーおよびインテグレーションデベロッパーは，HSDP に基づくパッケージをデプロイするためのユーザー環境として運用ディレクトリを作成する必要があります。ソリューションデベロッパーおよびインテグレーションデベロッパーがシステム構成を変更する場合，運用ディレクトリがまだ作成されていないときは作成する必要があります。

前提条件

前提条件を次に示します。

システム

- Streaming Data Platform がインストールされていること。
- 通常ユーザーのユーザーアカウントが，Streaming Data Platform のインストール先の OS に登録されていること。このアカウントは，分析シナリオの開発とアダプターの定義に使用されます。

ユーザー

作成した運用ディレクトリを使用するユーザーが OS にログインすること。

操作手順

1. 登録済みのアカウントを使用して OS にログインします。
2. 運用ディレクトリを作成するには，次のコマンドを実行します。

```
/opt/hitachi/hsdp/bin/hsdpsetup -dir 運用ディレクトリ
```

メモ

- -dir オプションで指定したディレクトリを最上位のディレクトリとして，運用ディレクトリが作成されます。SDP の実行に必要な権限は，作成した運用ディレクトリ（サブディレクトリおよびファイルを含む）に，各ディレクトリまたはファイルの属性として自動的に設定されます。
- SDP サーバ用定義ファイル，クエリ定義ファイル，およびその他のファイルのテンプレートファイルは運用ディレクトリに格納されます。

- 運用ディレクトリの設定に成功すると，hsdpsetup コマンドはメッセージKFHD91009-I を標準出力（stdout）に出力します。

  hsdpsetup コマンドは，次のディレクトリ構成を含む運用ディレクトリを作成します。

```
運用ディレクトリ/
+- inadaptor/
    +- conf/
        +- xml/
+- bin/
```

```
+- conf/
    +- xml/
+- exadaptor/
    +- inputadaptor/
        +- conf/
        +- bin/
    +- outputadaptor/
        +- conf/
        +- bin/
+- lib/
+- logs/
+- query/
+- spool/
+- trc/
    +- adaptor/
    +- tuplelog/
```

- 作成された運用ディレクトリのアクセス権限は，hsdpsetup コマンドを実行したユーザーに与えられます。
- 次の情報が*運用ディレクトリ*/logs/hsdpsetup.log に格納されます。

  運用ディレクトリの作成中に設定された値とhsdpsetup コマンドを実行したユーザーのアカウント情報
- 運用ディレクトリの設定に失敗すると，hsdpsetup コマンドはエラーメッセージを標準エラー出力（stderr）に出力します。

次の作業

運用ディレクトリの作成後，必要に応じて Streaming Data Platform software development kit をインストールします。

1.4 Hitachi Streaming Data Platform software development kit をインストールする

SDP に基づくユーザーパッケージを開発するために，またはすでに準備されているパッケージのシステム構成を変更するために Hitachi Streaming Data Platform software development kit が必要です。ソリューションデベロッパーまたはインテグレーションデベロッパーが分析シナリオ（クエリ）を CQL で開発する，オプションのカスタム関数を（Java/C 言語で）開発する，CQL デバッグツールを使用して分析シナリオをテストする，またはオプションのカスタムアダプターを（Java で）開発する場合，Streaming Data Platform がインストールされているホストに Hitachi Streaming Data Platform software development kit をインストールする必要があります。インテグレーションデベロッパーが内部標準提供アダプターだけを使用する場合，カスタムアダプターを開発する必要はありません。

前提条件

前提条件を次に示します。

システム

- Streaming Data Platform がインストールされていること。
- ディスクに十分な空き容量があること。
- メモリに十分な空き容量があること。

システム要件については，『Hitachi Streaming Data Platform software development kit リリースノート』を参照してください。

ユーザー

- 開発用ホストが設定されていること。

> **メモ**
>
> 仮想 OS に開発用ホストを設定することを推奨します。

操作手順

1. root ユーザーとして OS にログインします。
2. 対応するドライブにインストール用 CD を挿入します。

   CD は自動的にマウントされます。

   CD が自動的にマウントされない場合，次のコマンドを実行して手動でマウントします。

```
mount -t iso9660 /dev/cdrom CD-ROMマウントディレクトリ
```

3. 次のコマンドを実行して，インストールディレクトリに移動します。

```
cd CD-ROMマウントディレクトリ/HSDP-SDK/
```

4. 次のコマンドを実行します。

```
./install.sh
```

- Streaming Data Platform software development kit のインストール処理が開始されます。Streaming Data Platform software development kit のインストールが完了すると，次のメッセージが出力されます。

  ```
  Hitachi Streaming Data Platform software development kit installation completed successfully.
  ```

- Streaming Data Platform software development kit は，Streaming Data Platform のインストールディレクトリ（/opt/hitachi/hsdp/）の下位にインストールされます。
- Hitachi Streaming Data Platform software development kit のインストールに失敗すると，インストーラーが次のメッセージを標準エラー出力（stderr）に出力します。

  ```
  Hitachi Streaming Data Platform software development kit installation failed.
  ```

  システムおよびユーザーの前提条件が完全に満たされていることを確認し，このインストールを再実行してください。

次の作業

Streaming Data Platform software development kit のインストール後，次の作業を実施します。

- 入力アダプター，出力アダプター，および分析シナリオの詳細を設計します。

2 分析シナリオを設計，開発，テストする

この章では，分析シナリオの開発に関する情報について説明します。ソリューションプロジェクトマネージャーは，顧客の要件に基づいて分析シナリオの概要を作成します。この分析シナリオの概要に基づいて，ソリューションデベロッパーは分析シナリオを開発し，テストします。

2.1　分析シナリオの設計

ソリューションデベロッパーは使用するストリームデータ処理エンジンの種類，クエリの構成，およびタイムスタンプモードを決定後，分析シナリオを設計します。

ソリューションデベロッパーは分析シナリオを開発する前に，入出力データに関する情報を収集し，次に示す項目を検討しておく必要があります。

- クエリの構成

  Streaming Data Platform にどのようなデータを入力して，どのように処理，および出力するかを決定します。クエリの構成は次のとおりです。

  - クエリグループの数
  - クエリグループの名前
  - クエリグループの入力ストリーム
  - クエリの内容
  - クエリグループの出力ストリーム

  クエリの実装については，マニュアル『Hitachi Streaming Data Platform アプリケーション開発ガイド』を参照してください。
- タイムスタンプモードの決定

  Streaming Data Platform でサポートされているタイムスタンプモードは，サーバモードとデータソースモードです。サーバモードの場合は SDP サーバへのタプル到着時刻，データソースモードの場合はデータソースから入力したデータに入っている時刻情報が，タプルの時刻情報としてそれぞれ設定されます。デフォルトでは，タイムスタンプモードはサーバモードに設定されます。サーバモードの場合，タプルに設定される時刻情報の精度はミリ秒です。ログを分析する場合など，データソース内の時刻情報に基づいてストリームデータを順次処理したい場合，タイムスタンプモードをデータソースモードに変更してください。

2.1.1　タイムスタンプモードをデータソースモードに変更するための設定

ソリューションデベロッパーはパラメーター`stream.timestampMode`，`stream.timestampPosition`，および`stream.timestampAccuracy`の値を変更して，タイムスタンプモードをデータソースモードに変更します。

説明

すべてのクエリグループ，特定のクエリグループ，および特定のストリームに対してする必要があるパラメーター設定に関する情報を次の表に示します。

表 2-1　タイムスタンプモードをデータソースモードに変更するためのパラメーターの設定

パラメーター	アクション	すべてのクエリグループ	特定のクエリグループ	特定のストリーム
stream.timestampMode	パラメーターの値を DataSource に変更します。	SDP サーバ上のすべてのクエリグループのタイムスタンプモードを変更する必要がある場合，システムコンフィグプロパティファイルの stream.timestampMode パラメーターの値を変更する必要があります。	特定のクエリグループのタイムスタンプモードを変更する必要がある場合，クエリグループ用プロパティファイルの stream.timestampMode パラメーターの値を変更する必要があります。	該当なし
stream.timestampPosition	タプルの時刻データ列名を指定します。時刻データの場合，データ型をTIMESTAMP として指定する必要があります。	SDP サーバ上のすべてのクエリグループのタイムスタンプモードを変更する必要がある場合，システムコンフィグプロパティファイルの stream.timestampPosition パラメーターの値を変更する必要があります。	特定のクエリグループのタイムスタンプモードを変更する必要がある場合，クエリグループ用プロパティファイルの stream.timestampPosition パラメーターの値を変更する必要があります。	該当なし
stream.timestampAccuracy	タイムスタンプの調整に使用する時刻単位および時刻調整範囲を次のように指定します。 stream.timestampAccuracy={{sec|msec|usec},*時刻調整範囲*|unuse} • sec：秒 • msec：ミリ秒 • usec：マイクロ秒 • unuse：時刻調整は発生しません。 • *時刻調整範囲*:sec を指定する場合，0 から59 までの範囲を指定します。msec またはusec を指定する場合，0 から999 までの範囲を指定します。0 を指定すると，基準時刻だけが時刻調整範囲として使用されます。	SDP サーバ上のすべてのクエリグループのタイムスタンプモードを変更する必要がある場合，システムコンフィグプロパティファイルの stream.timestampAccuracy パラメーターの値を変更する必要があります。	特定のクエリグループのタイムスタンプモードを変更する必要がある場合，クエリグループ用プロパティファイルの stream.timestampAccuracy パラメーターの値を変更する必要があります。	特定のストリームのタイムスタンプモードを変更する必要がある場合，ストリーム用プロパティファイルを作成し，stream.timestampAccuracy パラメーターの値を設定する必要があります。

2.2　CQL を使用して分析シナリオを記述するための設定

ソリューションデベロッパーはクエリ定義ファイルに分析シナリオを記述します。CQL を使用して分析シナリオを記述するには，クエリ定義ファイル，ストリーム用プロパティファイル，およびクエリグループ用プロパティファイルを作成する必要があります。

説明

ソリューションデベロッパーは，Streaming Data Platform で提供されている該当のサンプルファイルに基づいて，クエリ定義ファイルおよびクエリグループ用プロパティファイルを作成します。ただし，Streaming Data Platform ではストリーム用プロパティファイルのサンプルファイルは提供されていません。ストリーム単位で設定を切り替えたい場合，そのストリームに対して，ストリーム用プロパティファイルを作成します。

CQL を使用して分析シナリオを記述するには，次のファイルを作成します。

クエリ定義ファイル

CQL を使用して分析シナリオを記述します。クエリ定義ファイルのサンプルファイルは，次の場所に格納されています。

運用ディレクトリ/query/QueryGroupTest.cql

ストリーム用プロパティファイル

クエリグループ内のストリーム単位でパフォーマンスをチューニングする必要がある場合，ストリーム用プロパティファイルを作成します。

クエリグループ用プロパティファイル

クエリグループ内のストリーム単位でパフォーマンスをチューニングする必要がある場合，作成したストリーム用プロパティファイルのパスを指定します。通常使用時は，ほかのパラメーターを変更する必要はありません。クエリグループ用プロパティファイルのサンプルファイルは，次の場所に格納されています。

運用ディレクトリ/conf/QueryGroupTest.qg

2.2.1　CQL を使用して分析シナリオを開発する

ソリューションデベロッパーはクエリ定義ファイル，クエリグループ用プロパティファイル，およびストリーム用プロパティファイルを作成して分析シナリオを開発します。

前提条件

前提条件を次に示します。

システム

- Streaming Data Platform がインストールされていること。
- Streaming Data Platform software development kit がインストールされていること。
- 運用ディレクトリが作成されていること。
- SDP サーバが停止していること。

ユーザー

- 分析シナリオを設計していること。

操作手順

1. **クエリ定義ファイルを作成します。ファイル名は*任意の文字列*.cql である必要があります。**

   クエリ定義ファイルに分析シナリオを記述するには，サンプルファイルを参考にします。クエリ定義ファイルには，代数的演算式を使用してパラメーターを定義することもできます。

   Streaming Data Platform でサポートされている CQL 構文および代数的演算式については，マニュアル『Hitachi Streaming Data Platform アプリケーション開発ガイド』を参照してください。

2. **クエリグループ用プロパティファイルを作成し，そのファイルを*運用ディレクトリ*/conf/に保存します。ファイル名は*クエリグループ名*.qg である必要があります。サンプルファイルを参考にして，ストリーム用プロパティファイルの格納先とタイムスタンプモードの設定をクエリグループ用プロパティファイルに指定します。**

   メモ

   クエリグループの名前は 64 文字以内である必要があります。

   次の文字を使用できます。

   - A～Z
   - a～z
   - 0～9
   - アンダーライン（_）

   クエリグループの名前は英字で始まる必要があります。

   - クエリ定義ファイル格納先のパスをquerygroup.cqlFilePath パラメーターに指定します。運用ディレクトリの最上位のディレクトリからの相対パスを使用します。例えば，./query/QueryTest.cql のように指定します。
   - クエリグループで，ストリーム単位でパフォーマンスをチューニングする場合，ストリーム用プロパティファイル格納先のパスをstream.propertyFiles パラメーターに指定します。
   - 特定のクエリグループのタイムスタンプモードを変更する場合，次のパラメーターの値も変更します。

     stream.timestampAccuracy

stream.timestampPosition

3. ストリーム単位でパフォーマンスをチューニングする場合，ストリーム用プロパティファイルを作成します。そのファイルを*運用ディレクトリ*/conf/に保存します。ファイル名は任意の文字列である必要があります。

   - ストリーム名とチューニングパラメーターをストリーム用プロパティファイルに指定します。
   - 特定のストリームのタイムスタンプモードを変更する場合，次のパラメーターの値も変更します。

     stream.timestampAccuracy

     stream.timestampPosition

   ストリーム用プロパティファイルの形式，パラメーター，指定する値については，「(3)　ストリーム用プロパティファイル」を参照してください。

操作結果

クエリ定義ファイル，クエリグループ用プロパティファイル，およびストリーム用プロパティファイルが作成されます。

次の作業

CQL を使用して分析シナリオを開発したあと，次の操作を実行します。

- 外部定義関数を開発します（外部定義関数が必要な場合）。
- ローカル環境で分析シナリオをテストします（外部定義関数が必要ない場合）。

2.3　外部定義関数を開発する

ソリューションデベロッパーは HSDP の提供するサンプルを参考に，分析シナリオで使用する外部定義関数を開発します。

前提条件

前提条件を次に示します。

システム

- Streaming Data Platform が full インストールされていること。
- Hitachi Streaming Data Platform software development kit がホストにインストールされていること。
- 運用ディレクトリが作成されていること。
- 次の分析シナリオのファイルが運用ディレクトリの所定ディレクトリに格納されていること。
  - クエリ定義ファイル
  - クエリグループ用プロパティファイル
  - ストリーム用プロパティファイル※

  注※

  ソリューションデベロッパーがこのファイルを作成していない場合は省略できます。

ユーザー

- ソリューションデベロッパーが運用ディレクトリを使用しているユーザーとしてホストにログインしていること。

操作手順

1. **外部定義関数（*.jar または*.so）を Java/C 言語で作成し，*運用ディレクトリ*/lib/に格納します。**

   外部定義関数の作成の詳細については，マニュアル『Hitachi Streaming Data Platform アプリケーション開発ガイド』を参照してください。

2. **作成した外部定義関数に対応する外部定義関数定義ファイル（.xml）を作成し，*運用ディレクトリ*/conf/xml/に格納します。**

   外部定義関数定義ファイルの作成の詳細については，「15.　外部定義関数定義ファイル」を参照してください。

3. 外部定義関数を使用する分析シナリオのクエリグループ用プロパティファイルのプロパティ querygroup.extFuncDefFilePath に，作成した外部定義関数定義ファイルのファイルパスを運用ディレクトリからの相対パスで指定してください。

   例えば，外部定義関数定義ファイル「myExtDef.xml」を運用ディレクトリからの相対パスで指定する場合，次のように指定します。

```
querygroup.extFuncDefFilePath =./conf/xml/myExtDef.xml
```

操作結果

ファイルが次のとおりに格納されます。

ファイル	格納先ディレクトリ
外部定義関数（*.jar または*.so）	*運用ディレクトリ*/lib/
外部定義関数定義ファイル（*.xml）	*運用ディレクトリ*/conf/xml/

次の作業

分析シナリオをテストします。

2.4　ローカル環境で分析シナリオをテストする

ソリューションデベロッパーは，クエリが必要なデータを期待どおりに出力するかどうかを，Streaming Data Platform software development kit に同梱されている CQL デバッグツールを使用してテストできます。

前提条件

前提条件を次に示します。

システム

- Streaming Data Platform と Streaming Data Platform software development kit がインストールされていること。
- 運用ディレクトリが作成されていること。
- SDP サーバが停止していること。

ユーザー

- ソリューションデベロッパーによって次のファイルが作成されていること。
  - クエリ定義ファイル
  - クエリグループ用プロパティファイル
  - ストリーム用プロパティファイル※
  - 外部定義関数ライブラリ※
  - 外部定義関数定義ファイル※
- クエリ定義ファイルに実装されたクエリグループの名前を確認済みであること。

注※

ソリューションデベロッパーがこのファイルを作成していない場合は省略できます。

操作手順

1. CSV 形式でテストデータを作成します。SDP サーバで複数のクエリグループを実行する場合，クエリグループごとにテストデータを作成します。

   テストデータの形式と格納先については，マニュアル『Hitachi Streaming Data Platform アプリケーション開発ガイド』を参照してください。

2. 分析シナリオをテストするには，次のコマンドを実行します。

```
運用ディレクトリ/bin/hsdpcqldebug クエリグループ名 -i 入力ディレクトリ -o 出力ディレクトリ
```

メモ

SDP サーバで複数のクエリグループを実行する場合，クエリグループごとにhsdpcqldebug コマンドを実行することで分析シナリオをテストします。

操作結果

- 分析シナリオの実行に成功した場合，KFHD80002-I が出力され，hsdpcqldebug コマンドで指定されたディレクトリの CSV 形式のファイルである*クエリグループ名-出力ストリーム名*.csv に結果が出力されます。
- CSV ファイルに出力されたデータが期待どおりの分析結果かどうかを確認します。
- 実行時エラーが発生すると，コンソールにエラーメッセージが表示されます。
- エラーメッセージが出力された場合や，CSV ファイルに出力されたデータが期待どおりの分析結果ではない場合，テストデータ，クエリ定義ファイル，またはクエリグループ用プロパティファイルを修正する必要があります。また，外部定義関数，および，その外部定義関数定義ファイルが使用されている場合は，あわせて修正してください。

2.5　分析シナリオを分割する際の注意事項

ソリューションデベロッパーはクエリ定義ファイルを分割して，新規のクエリ定義ファイルを作成します。

説明

クエリ定義ファイルを複数のファイルに分割したあと，例えば 2 つのファイルに分割した場合は，2 つめのファイルで 1 つめのファイルの出力ストリームのスキーマと同じスキーマの入力ストリームを定義します。次の例では，入力ストリームをQ1’と表しています。

```
register stream Q1(name VARCHAR(10), num BIGINT);
```

分割後のクエリ定義ファイルの最後のクエリには，ISTREAM 句，DSTREAM 句，またはRSTREAM 句の付く出力ストリームを定義してください。次のように終端のクエリをリレーションとして定義しないでください。

```
register query Q2 SELECT * FROM Q1[ROWS 3];
```

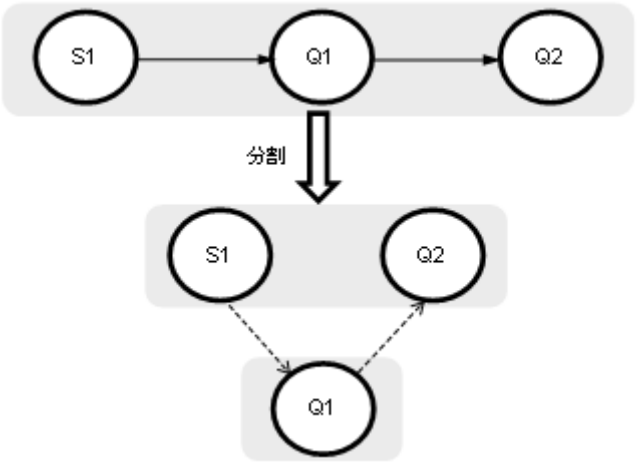
分割後のクエリ定義ファイルが入力ストリームだけになる場合，入力ストリームが入力したデータをそのまま出力する次のようなクエリを追加してください。

```
register stream S1(name VARCHAR(10), num BIGINT);
```

```
register query Q2 ISTREAM (SELECT * FROM S1[now]);
```

分割後のクエリ定義ファイルは，データの流れのループがないようにしてください。ループがあると，分析が実行されなくなります。ループがある不適切な例を次に示します。

図 2-1　ループがある不適切な構造の例

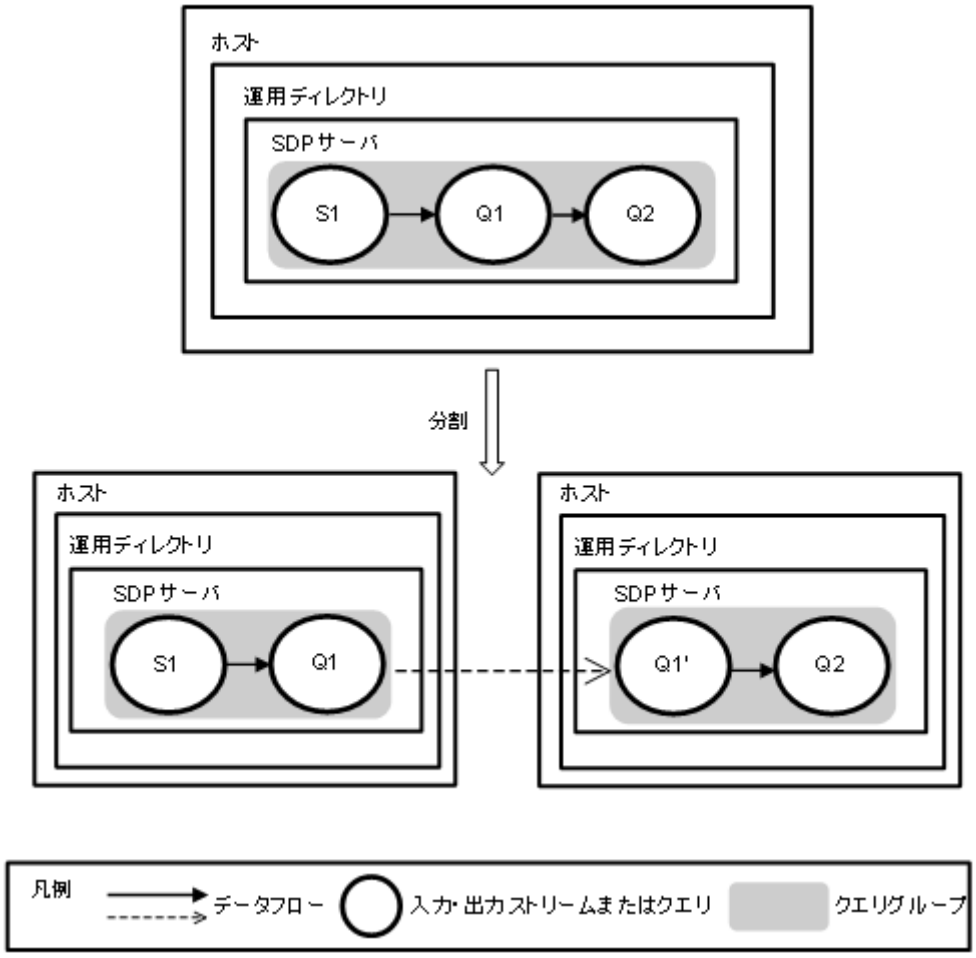


2.5.1　分析シナリオを分割する

ソリューションデベロッパーは分析シナリオを分割します。分析シナリオを分割するとは，単一のクエリグループを，複数のクエリグループに細分化することです。次のように同一または異なるホスト上で複数の運用ディレクトリをセットアップして，細分化したクエリグループを各運用ディレクトリの SDP サーバに登録することで，1 つのクエリグループ当たりの CPU 負荷やメモリ使用量を分散できます。

分析シナリオの分割例

図 2-2　複数の運用ディレクトリ間での分割された分析シナリオの流れ



前提条件

前提条件を次に示します。

システム

- Streaming Data Platform がインストールされていること。
- 運用ディレクトリがセットアップされていること。
- 次の分析シナリオのファイルが運用ディレクトリの所定ディレクトリに格納されていること。
  - クエリ定義ファイル

- クエリグループ用プロパティファイル
- ストリーム用プロパティファイル※
- 外部定義関数※
- 外部定義関数定義ファイル※

注※

ソリューションデベロッパーがこれらのファイルを作成していない場合は省略できます。

ユーザー

- 分割後の分析シナリオのファイルの格納先である運用ディレクトリをセットアップしていること。
- 運用ディレクトリをセットアップしたユーザーでホストにログインしていること。

操作手順

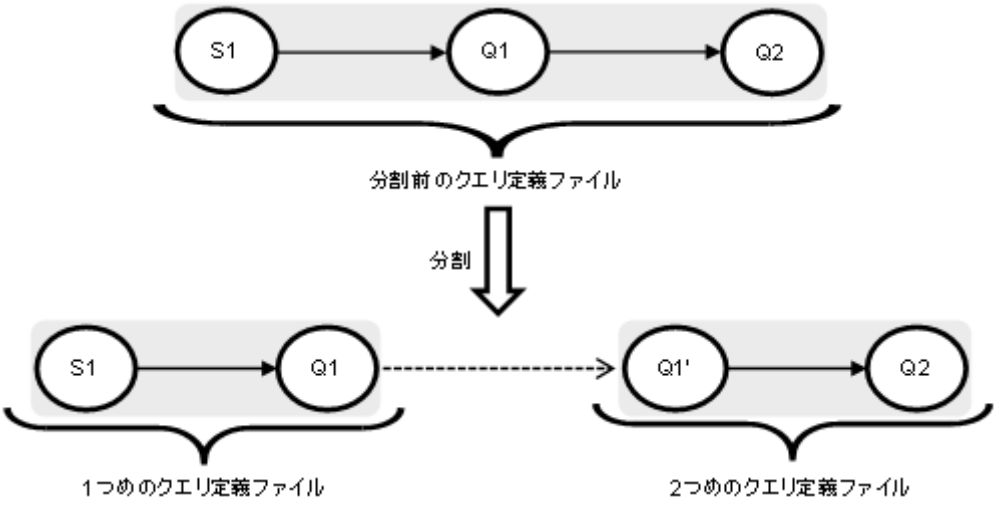
1. 分析シナリオを分割します。

次のルールに従って，クエリ定義ファイルを分割して，新しいクエリ定義ファイルを作成してください。

- 分割後のクエリ定義ファイルが入力ストリームだけになる場合，入力ストリームが入力したデータをそのまま出力するクエリを追加してください。
- 分割したクエリ定義ファイル間でデータのループがない構造にしてください。
- クエリ定義ファイルを分割したあと，2 つめのファイルで 1 つめのファイルの出力ストリームのスキーマと同じスキーマの入力ストリームを定義してください。
- 分割後のクエリ定義ファイルの最後のクエリには，ISTREAM 句，DSTREAM 句，またはRSTREAM 句の付く出力ストリームを定義してください。次のように終端のクエリをリレーションとして定義しないでください。

クエリ定義ファイルの分割例を次に示します。

図 2-3　クエリ定義ファイルの分割例



分割前のクエリ定義ファイルの例を次に示します。

```
register stream S1(name VARCHAR(10), num BIGINT);
```

```
register query Q1 ISTREAM(SELECT * FROM S1[NOW] WHERE S1.num > 0);
```

```
register query Q2 ISTREAM(SELECT name, sum(num) as total FROM Q1[RANGE 60 SECOND] GROUP B
Y Q1.name);
```

分割後の 1 つめのクエリ定義ファイルの例を次に示します。

```
register stream S1(name VARCHAR(10), num BIGINT);
```

```
register query Q1 ISTREAM(SELECT * FROM S1[NOW] WHERE S1.num > 0);
```

分割後の 2 つめのクエリ定義ファイル※の例を次に示します。

```
register stream Q1(name VARCHAR(10), num BIGINT);
```

```
register query Q2 ISTREAM(SELECT name, sum(num) as total FROM Q1[RANGE 60 SECOND] GROUP B
Y Q1.name);
```

注※

上記の図でQ1'と示している入力ストリームを，ここではQ1 として定義しています。

2. 分割後のクエリ定義ファイルを，次のディレクトリに格納します。

```
運用ディレクトリ/query/
```

分割前の分析シナリオで外部定義関数を使用している場合は，外部定義関数と外部定義関数定義ファイルをコピーして，次のディレクトリに格納してください。

ファイル	格納先
外部定義関数	*運用ディレクトリ*/lib/
外部定義関数定義ファイル	*運用ディレクトリ*/conf/xml/

3. 分割後のクエリ定義ファイルを格納したそれぞれの運用ディレクトリで，次の操作を実施します。

   a. クエリグループ用プロパティファイルを作成します。

   分割前のクエリグループ用プロパティファイルをコピーして，次のディレクトリに格納してください。コピーしたファイルの名称は任意です。

   ```
   運用ディレクトリ/conf/
   ```

b. コピーしたクエリグループ用プロパティファイルに，次のプロパティを指定します。

プロパティ	説明
querygroup.cqlFilePath	分割後のクエリ定義ファイルのパスを指定してください。クエリ定義ファイルを別のホストまたは運用ディレクトリに移行することを考慮して，運用ディレクトリからの相対パスで指定することをお勧めします。
stream.output.*ストリーム名*.link	*ストリーム名*には，クエリグループの出力ストリーム名を指定してください。ただし，出力ストリームが分割したクエリグループの最後尾である場合は，このプロパティの指定は不要です。 プロパティの指定例を次に示します。 `stream.output.Q1.link=QG2/Q1`

分析シナリオがデータソースモードで動作する場合（プロパティstream.timestampModeにDataSourceを指定している場合），次のプロパティも指定してください。

プロパティ	説明
stream.timestampPosition	__systemtime__を指定してください。 ただし，クエリグループが分割後の一連のクエリグループのうち，先頭のクエリグループである場合，このプロパティには入力データでタイムスタンプを表す任意のカラム名を指定します。

stream.timestampAccuracy	入力ストリームが複数の入力元からのデータを受け付ける場合，入力データの到着時刻のばらつきの幅に合わせて，タイムスタンプ調整機能の時刻単位と時刻調整範囲を指定してください。
	プロパティの指定例を次に示します。
	`querygroup.cqlFilePath=./query/QG2`
	`stream.timestampPosition=__systemtime__`
	`stream.timestampAccuracy=sec,2`

4. ストリーム用プロパティファイルを作成します。

分割前の分析シナリオでストリーム用プロパティファイルを使用している場合は，分割前のストリーム用プロパティファイルをコピーして，次のディレクトリに格納してください。

運用ディレクトリ/conf/

操作結果

ファイルが次のとおりに格納されます。

ファイル	格納先
分割後のクエリ定義ファイル（*.cql）	*運用ディレクトリ*/query/
クエリグループ用プロパティファイル（*.qg）	*運用ディレクトリ*/conf/
ストリーム用プロパティファイル（*任意の名前*）※	
外部定義関数（*.jar または*.so）※	*運用ディレクトリ*/lib/
外部定義関数定義ファイル（*.xml）※	*運用ディレクトリ*/conf/xml/

注※

ソリューションデベロッパーが同ファイルを作成していない場合は除きます。

次の作業

分割した分析シナリオをテストします。

2.5.2　分割した分析シナリオをテストする

ソリューションデベロッパーは，分割したそれぞれの分析シナリオをテストします。

前提条件

前提条件を次に示します。

システム

- Streaming Data Platform がホストに full インストールされていること。
- Streaming Data Platform software development kit がホストにインストールされていること。
- 運用ディレクトリがセットアップされていること。
- 次の分析シナリオのファイルが運用ディレクトリの所定ディレクトリに格納されていること。
  - クエリ定義ファイル
  - クエリグループ用プロパティファイル
  - ストリーム用プロパティファイル※
  - 外部定義関数※
  - 外部定義関数定義ファイル※

  注※
  ソリューションデベロッパーがこれらのファイルを作成していない場合は除きます。

ユーザー

運用ディレクトリをセットアップしたユーザーでホストにログインしていること。

操作手順

1. CSV 形式でテスト用入力データを作成します。
   SDP サーバで複数のクエリグループを運用している場合は，それぞれのクエリグループ用のテストデータを作成してください。テスト用入力データのフォーマットや格納先の詳細については，マニュアル『Hitachi Streaming Data Platform アプリケーション開発ガイド』を参照してください。

2. 分析シナリオをテストするために，次のコマンドを実行します。

```
運用ディレクトリ/bin/hsdpcqldebug クエリグループ名 -i 入力用ディレクトリ -o 出力用ディレクトリ
```

> **メモ**
>
> SDP サーバで複数のクエリグループを運用している場合は，それぞれのクエリグループで hsdpcqldebug コマンドを実行して分析シナリオをテストしてください。

操作結果

- 分析シナリオが要件に合っていた場合は，0 が返却され，hsdpcqldebug コマンドで指定したディレクトリにある CSV 形式の*クエリグループ名-出力ストリーム名*.csv に結果が出力されます。
- CSV ファイルに出力されたデータが想定どおりの分析結果になっているかどうかを確認します。
- エラーが発生した場合は，エラーメッセージがコンソールに表示されます。
- エラーメッセージが出力された場合，または CSV ファイルに出力されたデータが想定どおりの分析結果でなかった場合は，テストデータ，外部定義関数，外部定義関数定義ファイル，クエリ定義ファイル，またはクエリグループ用プロパティファイルを修正してください。

次の作業

内部標準アダプター，内部カスタムアダプター，または外部アダプターを開発します。

3 アダプターを設計，開発，テストする

この章では，内部アダプターおよび外部アダプターの設定に関する情報について説明します。ソリューションプロジェクトマネージャーは，顧客の要件に基づいて分析シナリオの概要を作成します。この分析シナリオの概要に基づいて，インテグレーションデベロッパーは内部入力アダプター，内部出力アダプター，外部入力アダプター，および外部出力アダプターを設定し，テストします。

3.1　内部入力および出力アダプターの詳細を設計する

Streaming Data Platform には次の 2 種類の内部アダプターがあります。標準提供アダプターとカスタムアダプターです。標準提供アダプターは Streaming Data Platform に組み込まれており，カスタムアダプターはインテグレーションデベロッパーが Streaming Data Platform software development kit に同梱された API を使用して開発します。標準提供アダプターとカスタムアダプターの両方に，それぞれ内部入力アダプターおよび内部出力アダプターを実装できます。内部入力アダプターは CSV ファイルやパケットデータのようなデータを入力し，そのデータをストリームデータ処理エンジンに送信します。内部出力アダプターはストリームデータ処理エンジンから受信したデータを CSV ファイルやダッシュボードに出力します。

アダプターを実装する前に，インテグレーションデベロッパーは使用するアダプターの数と各アダプターが実行する処理（コールバック）を決定します。

標準提供アダプターを使用する場合，内部入力アダプターおよび内部出力アダプターの設計の詳細を決定後，それらのアダプターを設定する必要があります。カスタムアダプターを使用する場合，内部入力および出力アダプターを開発する必要があります。

3.1.1　内部アダプター実装の前提条件

インテグレーションデベロッパーは，内部アダプターを実装する前に，内部アダプターの数と内部アダプターが実行する処理（コールバック）に関する情報を決定する必要があります。

説明

インテグレーションデベロッパーは内部アダプターを実装する前に，次のことを検討する必要があります。

アダプターの数

内部入力および出力アダプターはアダプターグループという個々のグループ単位で扱われます。標準提供アダプターを適用する場合，1 台の SDP サーバではアダプターグループを128 個まで扱うことができます。データ入力元およびデータ出力先ごとにそれぞれ内部入力アダプターおよび内部出力アダプターが必要です。1 つのアダプターグループでは，内部入力アダプターと内部出力アダプターをそれぞれ64 個まで指定できます。ただし，障害が発生した場合にその影響を最小限にするため，アダプターグループごとにアダプターを 1 つだけ実装するという構成を推奨します。

内部入力アダプターが実行する処理（コールバック）

- データ入力（入力対象の形式は，TCP データ，CSV ファイル，HTTP パケットなど）
- データ編集（マッピング，レコードの抽出，フォーマット変換など）
- タプル送信（ストリームデータ処理エンジンにデータを出力）

内部出力アダプターが実行する処理（コールバック）

- データ出力（出力対象の形式は，CSV ファイル，ダッシュボードなど）

- データ編集（マッピング，フィルタリング，フォーマット変換など）
- タプル受信（ストリームデータ処理エンジンからタプルを受信）

3.2　標準提供アダプターに必要なファイルの設定

インテグレーションデベロッパーは，Streaming Data Platform で組み込みアダプター（標準提供アダプター）を内部入力および出力アダプターとして使用するために，特定のファイルを構成する必要があります。これらのファイルには，アダプター構成定義ファイルのテンプレートファイル，パラメーターファイル，およびインプロセス連携用プロパティファイルが含まれます。

説明

Streaming Data Platform で組み込みアダプター（標準提供アダプター）を内部入力および出力アダプターとして使用するために，次のファイルを設定する必要があります。

- アダプター構成定義ファイルのテンプレートファイル

  アダプターグループの構成とアダプターが使用するポート番号を指定します。また，アダプターが使用する入出力コネクターに関する情報を指定します。データ入力元とデータ出力先ごとにテンプレートを作成する必要があります。

  内部入力アダプター

  Streaming Data Platform は次の種類の内部入力アダプターをサポートしています。

  - CQL エンジン用のファイル入力アダプター
  - CQL エンジン用の HTTP パケット入力アダプター
  - CQL エンジン用の TCP データ入力アダプター
  - acceleration CQL エンジン用の TCP データ入力アダプター

  内部出力アダプター

  Streaming Data Platform は次の種類の内部出力アダプターをサポートしています。

  - CQL エンジン用のファイル出力アダプター
  - CQL エンジン用のダッシュボード出力アダプター
  - CQL エンジン用のカスケーディングアダプター
  - acceleration CQL エンジン用のカスケーディングアダプター
- パラメーターファイル
- インプロセス連携用プロパティファイル

  インプロセス連携用プロパティファイルの名前は，`user_app.`*アダプター構成定義ファイルの名前*`.properties` と指定する必要があります。

3.2.1　内部アダプター設定の前提条件（標準提供アダプターを使用する場合）

標準提供アダプターを使用する場合，内部入力および出力アダプターを設定する前に，インテグレーションデベロッパーはソリューションデベロッパーからファイル（テスト済みの分析シナリオに関するもの）

を受け取るタスク，ソリューションデベロッパーに確認してクエリに関する詳細情報を取得するタスク，接続先の開発サーバに関する情報を確認するタスク，および標準提供アダプターを設計するタスクを完了しておく必要があります。

説明

内部アダプター設定の前提条件を次に示します。

- Streaming Data Platform がインストールされていること。
- 運用ディレクトリが作成されていること。
- SDP サーバが停止していること。
- 次に示すテスト済みのファイル（分析シナリオに関するもの）をソリューションデベロッパーから受け取っていること。
  - クエリ定義ファイル
  - クエリグループ用プロパティファイル
  - ストリーム用プロパティファイル（必要な場合）
  - 外部定義関数（必要な場合）
  - 外部定義関数定義ファイル（必要な場合）
- クエリ定義ファイル中のクエリの情報を確認すること。
  ソリューションデベロッパーに確認して，次に示すクエリの詳細情報を取得します。

アダプター	クエリの詳細情報
内部標準入力アダプター	• 入力アダプターがタプルを送信する（送信先の）クエリグループの名前 • アダプターがタプルを送信する（送信先の）入力ストリームの名前 • 入力ストリームのデータ型
内部標準出力アダプター	• アダプターがタプルを受信する（タプル送信元の）クエリグループの名前 • アダプターがタプルを受信する（タプル送信元の）出力ストリームの名前 • 出力ストリームのデータ型

- 標準提供アダプターが設計されていること。
  アダプターグループごとにアダプターを 1 つだけ指定することを推奨します。
  アダプター名とアダプターグループ名を決める必要があります。アダプターおよびアダプターグループの命名規則を次に示します。
  - アダプター名
    アダプターの名前は 100 文字以内である必要があります。次の文字を使用できます。
    A～Z，a～z，0～9，アンダーライン（_）
    また，アダプター名は英字で始まる必要があります。
  - アダプターグループ名
    アダプターグループの名前は 32 文字以内である必要があります。次の文字を使用できます。

A～Z，a～z，0～9，アンダーライン（_）
また，アダプターグループ名は英字で始まる必要があります。

3.2.2 内部アダプターの設定に必要なファイルをデプロイする（標準提供アダプターを使用する場合）

インテグレーションデベロッパーは，ソリューションデベロッパーからテスト済みのファイルを受け取り，それらのファイルをそれぞれのディレクトリにコピーします。

操作手順

1. クエリ定義ファイル，クエリグループ用プロパティファイル，ストリーム用プロパティファイル，外部定義関数，および外部定義関数定義ファイルをソリューションデベロッパーから受け取り，それらのファイルをディレクトリに保存します。
   受け取ったファイルの保存先のディレクトリは次のとおりです。

ファイル	ディレクトリ
クエリ定義ファイル	*運用ディレクトリ*/query/
クエリグループ用プロパティファイル	*運用ディレクトリ*/conf/
ストリーム用プロパティファイル（必要な場合）	*運用ディレクトリ*/conf/
外部定義関数（必要な場合）	*運用ディレクトリ*/lib/
外部定義関数定義ファイル（必要な場合）	*運用ディレクトリ*/conf/xml/

3.2.3 内部入力および出力アダプターを設定する（標準提供アダプターを使用する場合）

インテグレーションデベロッパーは，サンプルのテンプレートファイル，パラメーターファイル，インプロセス連携用プロパティファイルをコピーして，パラメーターファイルを編集します。

操作手順

1. テンプレートファイルとパラメーターファイルのサンプルファイルをコピーして，ファイルの名前を変更します。
   必要なサンプルファイルはアダプターの種類ごとに次の場所に格納されています。
   - アダプターの種類およびサンプルファイル名

アダプターの種類	サンプルファイル
ファイル入力アダプター	アダプター構成定義ファイル：fileInputAdaptor.xml

アダプターの種類	サンプルファイル
ファイル入力アダプター	パラメーターファイル：fileInputAdaptor.param インプロセス連携用プロパティファイル：user_app.fileInputAdaptor.properties
TCP データ入力アダプター	アダプター構成定義ファイル：tcpInputAdaptor.xml パラメーターファイル：tcpInputAdaptor.param インプロセス連携用プロパティファイル：user_app.tcpInputAdaptor.properties
ダッシュボード出力アダプター	アダプター構成定義ファイル：dashboardAdaptor.xml パラメーターファイル：dashboardAdaptor.param インプロセス連携用プロパティファイル：user_app.dashboardAdaptor.properties
ファイル出力アダプター	アダプター構成定義ファイル：fileOutputAdaptor.xml パラメーターファイル：fileOutputAdaptor.param インプロセス連携用プロパティファイル： user_app.fileOutputAdaptor.properties
カスケーディングアダプター	アダプター構成定義ファイル：cascadingAdaptor.xml パラメーターファイル：cascadingAdaptor.param インプロセス連携用プロパティファイル：user_app.cascadingAdaptor.properties

- サンプルファイルの格納先

ファイル	ディレクトリ
アダプター構成定義ファイル	*運用ディレクトリ*/conf/xml/
パラメーターファイル	*運用ディレクトリ*/conf/

操作結果

内部入力および出力アダプターが設定されます。

次の作業

統合テストを実施します。

3.3 内部カスタムアダプターを開発する

インテグレーションデベロッパーは，内部カスタムアダプターを開発します。

前提条件

前提条件を次に示します。

システム

- Streaming Data Platform が full インストールされていること。
- Hitachi Streaming Data Platform software development kit がホストにインストールされていること。
- 運用ディレクトリがセットアップされていること。

ユーザー

- 内部カスタムアダプターに関する次の項目について決定していること。
  - 内部カスタムアダプター名
  - 内部カスタムアダプターのクラス名およびクラスパス
  - 内部カスタムアダプターのクラスに渡す引数の有無
- 運用ディレクトリをセットアップしたユーザーでホストにログインしていること。
- ソリューションデベロッパーから次の分析シナリオのファイルを入手して，運用ディレクトリ以下の所定のディレクトリに格納していること。
  - クエリ定義ファイル
  - クエリグループ用プロパティファイル
  - ストリーム用プロパティファイル※
  - 外部定義関数※
  - 外部定義関数定義ファイル※

  注※
  ソリューションデベロッパーが該当のファイルを作成していない場合は不要です。

操作手順

1. **内部カスタムアダプターを開発します。**
   内部カスタムアダプターのクラスファイル（*.class）またはクラスファイルのアーカイブファイル（*.jar）を作成してください。
   作成したファイルは，次のディレクトリに格納してください。
   運用ディレクトリ/lib/

操作結果

ファイルが次のとおりに格納されます。

ファイル	格納先ディレクトリ
クエリ定義ファイル（*.cql）	*運用ディレクトリ*/query/
クエリグループ用プロパティファイル（*.qg）	*運用ディレクトリ*/conf/
ストリーム用プロパティファイル（*任意の名前*）※	
インプロセス連携用プロパティファイル（user_app.*.properties）	
内部カスタムアダプターのクラスファイル（*.class）またはクラスファイルのアーカイブファイル（*.jar）	*運用ディレクトリ*/lib/
外部定義関数（*.jar または*.so）※	*運用ディレクトリ*/lib/
外部定義関数定義ファイル（*.xml）※	*運用ディレクトリ*/conf/xml/

注※

ソリューションデベロッパーが該当のファイルを作成していない場合は不要です。

次の作業

- 内部標準アダプターを開発する
- 外部アダプターを開発する

> **メモ**
>
> 内部標準アダプターおよび外部アダプターの開発が不要な場合は，統合テスト環境の構築に進んでください。

3.4 外部アダプターを開発する

インテグレーションデベロッパーは，サンプルを基に外部アダプターを開発します。サンプルは Streaming Data Platform software development kit で提供されます。

前提条件

前提条件を次に示します。

システム

- Streaming Data Platform が full インストールされていること。
- Hitachi Streaming Data Platform software development kit がホストにインストールされていること。
- 運用ディレクトリがセットアップされていること。

ユーザー

- 外部アダプターに関する次の項目について決定していること。

外部アダプターの種類	項目の種類
外部入力アダプター	• ストリームデータの出力先クエリグループと入力ストリーム • データソースの入力形式 • カスタムディスパッチャーの使用有無 • 外部アダプターの接続先ブローカーのホスト名または IP アドレスとポート番号
外部出力アダプター	• ストリームデータの入力元クエリグループと出力ストリーム • 外部アダプターの接続先ブローカーのホスト名または IP アドレスとポート番号

- 運用ディレクトリを作成したユーザーでホストにログインしていること。

操作手順

1. 外部アダプターの格納ディレクトリを作成します。
   a. サンプルをコピーして，外部アダプターの格納ディレクトリを*運用ディレクトリ*/exadaptor/の中に作成してください。外部アダプターの格納ディレクトリの名称は任意です。
   b. 開発が必要な外部アダプターそれぞれに対して，格納先ディレクトリを作成してください。
   c. 入力および出力アダプターの外部アダプター格納ディレクトリのサンプルを使用してください。
      外部入力アダプターのサンプルを次に示します。

```
/opt/hitachi/hsdp/sdk/samples/exadaptor/inputadaptor/
```

      外部出力アダプターのサンプルを次に示します。

```
/opt/hitachi/hsdp/sdk/samples/exadaptor/outputadaptor/
```

例えば，外部入力アダプターの格納のために「myexinadaptor」という名前のディレクトリを作成する場合は，次のコマンドを実行して外部入力アダプターのサンプルをコピーしてください。

```
$ cp -r /opt/hitachi/hsdp/sdk/samples/exadaptor/inputadaptor 運用ディレクトリ/exadaptor/myexinadaptor
```

2. 外部アダプター定義ファイルを作成します。

> **メモ**
>
> この手順は，外部アダプターの格納ディレクトリごとに実施してください。

a. 次の例を参考にして，外部アダプター定義ファイル（.cfg）を作成してください。

```
運用ディレクトリ/exadaptor/外部アダプターの格納ディレクトリ/conf/exadp_input.cfg
```

```
運用ディレクトリ/exadaptor/外部アダプターの格納ディレクトリ/conf/exadp_output.cfg
```

b. 外部アダプター定義ファイルでhsdp.adaptor.name プロパティを指定してください。プロパティの詳細については，「12.15　外部アダプター定義ファイル」を参照してください。

> **メモ**
>
> 手順３および手順４での操作は，外部入力アダプターでカスタムディスパッチャーを使用する場合に実施します。カスタムディスパッチャーを使用しない場合は，手順５に進んでください。

3. カスタムディスパッチャーのソースファイルを作成します。

次の例を参考にして，カスタムディスパッチャーのソースファイル（.java）を作成してください。

```
/opt/hitachi/hsdp/sdk/samples/exadaptor/inputadaptor/dispatcher/src/Dispatcher.java
```

4. カスタムディスパッチャーのクラスファイルを作成します。

a. 次のjavac コマンドを実行して，カスタムディスパッチャーのソースファイルをコンパイルしてください。

```
$ /opt/hitachi/hsdpbase/system/psb/jdk/bin/javac -classpath /opt/hitachi/hsdp/lib/hsdp-exadp.jar ソースファイルのパス
```

カスタムディスパッチャーmyDispatcher.java のソースファイルをコンパイルする場合，次のコマンドを実行します。

```
$ /opt/hitachi/hsdpbase/system/psb/jdk/bin/javac -classpath /opt/hitachi/hsdp/lib/hsdp-exadp.jar 運用ディレクトリ/exadaptor/外部アダプターの格納ディレクトリ/dispatcher/src/myDispatcher.java
```

b. 必要に応じて，次の場所にあるjar コマンドを実行して，カスタムディスパッチャーのクラスファイルのアーカイブファイル（.jar）を作成してください。

```
/opt/hitachi/hsdpbase/system/psb/jdk/bin/jar
```

c. 作成したファイルを*運用ディレクトリ*/exadaptor/*外部アダプターの格納ディレクトリ*/dispatcher/bin/に格納してください。

5. 次のサンプルを参考にして，外部アダプターのソースファイル（.java）を作成します。

- 外部入力アダプターの場合

```
/opt/hitachi/hsdp/sdk/samples/exadaptor/inputadaptor/src/ExternalInputAdaptor.java
```

- 外部出力アダプターの場合

```
/opt/hitachi/hsdp/sdk/samples/exadaptor/outputadaptor/src/ExternalOutputAdaptor.java
```

6. 外部アダプターのソースファイルをコンパイルします。

a. 次のjavac コマンドを実行して，外部アダプターのソースファイルをコンパイルしてください。

```
$ /opt/hitachi/hsdpbase/system/psb/jdk/bin/javac -classpath /opt/hitachi/hsdp/lib/hsdp-exadp.jar:運用ディレクトリ/exadaptor/外部アダプターの格納ディレクトリ/bin ソースファイルのパス
```

外部入力アダプターのソースファイルmyexinadaptor.java をコンパイルする場合，次のコマンドを実行します。

```
$ /opt/hitachi/hsdpbase/system/psb/jdk/bin/javac -classpath /opt/hitachi/hsdp/lib/hsdp-exadp.jar:運用ディレクトリ/exadaptor/外部アダプターの格納ディレクトリ/bin 運用ディレクトリ/exadaptor/外部アダプターの格納ディレクトリ/src/myexinadaptor.java
```

b. 必要に応じて，次のjar コマンドを実行して，外部アダプターのクラスファイルのアーカイブファイル（.jar）を作成してください。

```
/opt/hitachi/hsdpbase/system/psb/jdk/bin/jar
```

c. 作成したファイルは，*運用ディレクトリ*/exadaptor/*外部アダプターの格納ディレクトリ*/bin/に格納してください。

7. 外部アダプターの実行スクリプトを作成します。

次のjava コマンドで，外部アダプターを実行してください。必要に応じて，コマンドを実行するスクリプトを作成してください。

```
$ /opt/hitachi/hsdpbase/system/psb/jdk/bin/java -classpath /opt/hitachi/hsdp/lib/hsdp-exadp.jar:/opt/hitachi/HNTRLib2/classes/hntrlib2j64.jar:運用ディレクトリ/exadaptor/外部アダプターの格納ディレクトリ/bin 外部アダプターのクラス名
```

外部アダプターの実行スクリプトの格納場所は任意です。ただし，開発した外部アダプターを構築フェーズに別のホストへ移行する予定の場合，外部アダプターの実行スクリプトは，外部アダプターの格納ディレクトリ下に格納してください。

外部アダプターmyexinadaptor を実行するシェルスクリプトを次に示します。

```
#!/bin/bash
cd `dirname $0`
/opt/hitachi/hsdpbase/system/psb/jdk/bin/java -classpath /opt/hitachi/hsdp/lib/hsdp-exadp
.jar:/opt/hitachi/HNTRLib2/classes/hntrlib2j64.jar:./bin myexinadaptor
```

メモ

外部アダプター格納ディレクトリをカレントディレクトリとして，このスクリプトを起動することを想定しています。

操作結果

ファイルが次のとおりに格納されます。

ファイル	格納先ディレクトリ
外部アダプター定義ファイル（*.cfg）	*運用ディレクトリ*/exadaptor/*外部アダプターの格納ディレクトリ*/conf/
外部アダプターのクラスファイル（*.class）またはクラスファイルのアーカイブファイル（*.jar）	*運用ディレクトリ*/exadaptor/*外部アダプターの格納ディレクトリ*/bin/
カスタムディスパッチャーのクラスファイル（*.class）またはクラスファイルのアーカイブファイル（*.jar）	
外部アダプターの実行スクリプト	任意のディレクトリ

次の作業

- 内部標準アダプターを開発する
- 内部カスタムアダプターを開発する

メモ

内部標準アダプターおよび内部カスタムアダプターの開発が不要な場合は，統合テスト環境の構築に進んでください。

3.5 統合テスト

統合テストでは，インテグレーションデベロッパーはアダプターに関連する設定ファイルが各アダプターグループで正しく設定されているかどうか，およびアダプターが正常に動作しているかどうかを確認する必要があります。分析シナリオとアダプターを同じ運用ディレクトリで開発してテストを実施する場合，統合テストを別途実施する必要はありません。

前提条件

前提条件を次に示します。

システム

- Streaming Data Platform がインストールされていること。
- SDP マネージャーが起動していること。

ユーザー

- 統合テスト環境を構築していること。
  運用ディレクトリを作成していること。
  統合テストに必要な分析シナリオおよびアダプターのファイルを運用ディレクトリに指定していること。
- テストに使用する入力データが作成されていること。
  内部入力アダプターおよび外部入力アダプターのテストデータが，顧客環境で想定されるデータに基づいて作成されていること。
  外部入力アダプターおよび外部出力アダプターを使用する場合，開始するために必要なコマンドをすでに用意していること。

操作手順

1. SDP サーバを起動し，クエリグループを登録して開始します。
   a. SDP サーバを起動するには，各運用ディレクトリで次のコマンドを実行します。

   ```
   運用ディレクトリ/bin/hsdpstart
   ```

   b. クエリグループを登録するために，次のコマンドを実行します。

   ```
   運用ディレクトリ/bin/hsdpcql -start クエリグループ名
   ```

   分析シナリオを実行するために CQL で記述されたクエリ定義ファイルに対して，文法チェックが実施されます。クエリ定義ファイルの場所は，クエリグループ用プロパティファイルで設定されています。クエリ定義ファイルに問題がない場合，0 が返されます。

2. 内部アダプターを起動するには，運用ディレクトリでhsdpstartinpro コマンドを実行します。

アダプター構成定義ファイルがスキーマを自動解決する書式になっている場合

アダプター構成定義ファイル*myadp*.xml と，アダプターが接続するクエリグループ名*myQueryGroup* およびストリーム名*myStream* とを指定して，内部アダプター*myAdaptor* を起動します。

```
運用ディレクトリ/bin/hsdpstartinpro -qg myQueryGroup -st myStream -file myadp.xml myAdaptor
```

アダプター構成定義ファイルがスキーマを自動解決する書式になっていない場合

アダプター構成定義ファイル*myadp*.xml を指定して，内部アダプター*myAdaptor* を起動します。

```
運用ディレクトリ/bin/hsdpstartinpro -file myadp.xml myAdaptor
```

3. アダプターおよびアダプターグループに関する情報を確認するには，次のコマンドを実行します。

```
hsdpstatusshow -adp [アダプターグループ名 | カスタムアダプター名]...
```

- 設定されているすべてのアダプターおよびアダプターグループに関する情報が表示されているかどうかを確認します。
- アダプターの状態とアダプターグループの状態にRunning が表示されているかどうかを確認します。

> **メモ**
>
> アダプターの状態がRunning ではない場合，またはアダプターおよびアダプターグループに関する情報が表示されていない場合は，テンプレートファイルまたはパラメーターファイルを変更します。

4. 外部出力アダプターを使用する場合は，すべての外部出力アダプターを開始してください。

5. 外部入力アダプターを使用する場合は，すべての外部入力アダプターを開始してください。

操作結果

統合テストが成功した場合，結果は次のとおりです。

- hsdpstart コマンドは，標準出力（stdout）にメッセージKFHD92010-I を出力します。
- hsdpcql コマンドは，標準出力（stdout）にメッセージKFHD92010-I を出力します。
- hsdpcqlstart コマンドは，標準出力（stdout）にメッセージKFHD92010-I を出力します。
- 外部出力アダプターはストリームデータを出力します。

> **メモ**
>
> 出力ストリームデータの内容が期待どおりかどうかを確認します。データが期待どおりではなかった場合，分析シナリオおよびアダプターの開発時に作成したファイルを見直して修正します。該当ファイルを見直して修正したあと，統合テストを再度実施します。

統合テストが失敗した場合，結果は次のとおりです。

- コマンドと外部アダプターはエラーメッセージを標準エラー出力（stderr）に出力します。
- *運用ディレクトリ*/logs/ディレクトリのログファイルにエラーメッセージが出力されます。
- /var/log/hitachi/hsdp/ディレクトリのログファイルにエラーメッセージが出力されます。

エラーの対処方法については，マニュアル『Hitachi Streaming Data Platform メッセージ』を参照してください。

次の作業

統合テストの終了後，次の操作を実施してください。

- 次のファイルのパラメーター値を定義してください。
  - SDP マネージャーの定義ファイル
  - 分析シナリオに関するファイル
    クエリ定義ファイル，クエリグループ用プロパティファイル，ストリーム用プロパティファイル，および外部定義関数定義ファイル
  - 内部アダプターに関するファイル
    アダプター構成定義ファイル

4 パラメーター値を指定し，分析シナリオおよびアダプターのファイルをエクスポートする

Streaming Data Platform のインストールが完了したら，SDP マネージャーの定義ファイルのパラメーター値，分析シナリオ，および内部アダプターのファイルを準備してテストします。分析シナリオとアダプターファイルがエクスポートされます。

4.1　分析シナリオのパラメーターを指定する

ソリューションデベロッパーは，分析シナリオのパラメーター値を指定します。パラメーター値の指定が完了したら，テストを実施します。

前提条件

前提条件を次に示します。

システム

- Streaming Data Platform が full インストールされていること。
- Streaming Data Platform software development kit がホストにインストールされていること。
- 統合テストに使用する運用ディレクトリがセットアップされていること。

ユーザー

- ソリューションデベロッパーが，統合テストに使用する運用ディレクトリをセットアップしたユーザーとしてホストにログインしていること。

操作手順

1. 必要に応じて，パラメーターファイル（*.param）を作成して，クエリ定義ファイル，クエリグループ用プロパティファイル，ストリーム用プロパティファイル，および外部定義関数定義ファイルに分析シナリオのパラメーター値を設定します。

   a. 作成したファイルを次のディレクトリに格納してください。

ファイル	格納先ディレクトリ
パラメーター化したクエリ定義ファイル（*.cql）	*運用ディレクトリ*/query/
パラメーター化したクエリグループ用プロパティファイル（*.qg）	*運用ディレクトリ*/conf/
クエリグループ用プロパティファイルのパラメーターファイル（*.param）	
クエリ定義ファイルのパラメーターファイル（*.param）	
外部定義関数定義ファイルのパラメーターファイル（*.param）	
パラメーター化した外部定義関数定義ファイル（*.xml）	*運用ディレクトリ*/conf/xml/

2. hsdpcqldebug コマンドを実行して，指定したパラメーター値をテストします。

   hsdpcqldebug コマンドで，テンプレートファイルとパラメーターファイルを統合して，指定したパラメーター値を適用します。

操作結果

- 分析シナリオファイルのパラメーター値が正しく指定されていた場合は，hsdpcqldebug コマンドがメッセージKFHD80002-I を標準出力（stdout）に出力します。また，テンプレートファイルとパラメーターファイルが統合され，次のファイルに出力されます。

ファイル	格納先ディレクトリ
クエリ定義ファイル（*.cql.out）	*運用ディレクトリ*/query/
クエリグループ用プロパティファイルのテンプレートファイル（*.qg.out）	*運用ディレクトリ*/conf/
外部定義関数定義ファイル（*.xml.out）	*運用ディレクトリ*/conf/xml/

> **メモ**
>
> 統合されたファイルの内容が想定どおりの結果となっていることを確認してください。ファイルの内容が正しくない場合，テンプレートファイルとパラメーターファイルを見直して修正してください。その後，手順 1 および手順 2 を再度実施してください。

- 分析シナリオファイルのパラメーター値が正しく指定されていなかった場合は，hsdpcqldebug コマンドを実行した際に標準エラー出力にエラーメッセージが出力されます。

  エラーの対処方法については，マニュアル『Hitachi Streaming Data Platform メッセージ』を参照してください。

次の作業

内部アダプターのファイルのパラメーター値を指定します。

4.2　内部アダプターのファイルのパラメーターを指定する

インテグレーションデベロッパーは，内部アダプターのファイルのパラメーター値を指定します。パラメーター値の指定が完了したら，テストを実施します。

前提条件

前提条件を次に示します。

システム

- Streaming Data Platform がインストールされていること。
- 統合テストに使用する運用ディレクトリがセットアップされていること。

ユーザー

- インテグレーションデベロッパーが，統合テストに使用する運用ディレクトリを作成したユーザーとしてホストにログインしていること。

操作手順

1. アダプター構成定義ファイルのテンプレートファイル（*.xml）およびパラメーターファイル（*.param）を編集して，内部アダプターのファイルにパラメーター値を設定します。

2. Streaming Data Platform は，SDP ブローカーを使用している場合，外部アダプターとの通信，分割した分割シナリオ間の通信に使う内部アダプターを自動生成できます（詳細は「14.10　自動生成アダプターテンプレートファイル」を参照してください）。自動生成の内部アダプターを利用する場合，次のテンプレートファイルを編集して，自動生成するアダプターの設定値をパラメーター化してください。パラメーター化方法の詳細は，「11.　定義ファイルのパラメーターへの変換とパラメーターファイルの形式」を参照してください。

ファイル	格納先ディレクトリ
TCP データ入力アダプターのアダプター構成定義ファイル（tcpinput.xml）	*運用ディレクトリ*/inadaptor/conf/xml/
外部アダプターに接続するカスケーディングアダプターのアダプター構成定義ファイル（cascading_out.xml）	
内部 TCP データ入力アダプターに接続するカスケーディングアダプターのアダプター構成定義ファイル（cascading_link.xml）	

3. hsdpstartinpro コマンドを実行して，指定したパラメーター値をテストします。

   hsdpstartinpro コマンドで，表に一覧化されたテンプレートファイルおよびパラメーターファイルを統合して，指定したパラメーター値を適用します。

   すでに内部アダプターのファイルが自動で生成されるようにパラメーター値を設定している場合は，「3.5　統合テスト」を実施してください。

操作結果

- 内部アダプターのファイルのパラメーター値が正しく指定されていた場合は，hsdpstartinpro コマンドがメッセージKFHD92010-I を標準出力（stdout）に出力します。また，テンプレートファイルとパラメーターファイルが統合され，次のファイルに出力されます。

ファイル	格納先ディレクトリ
アダプター構成定義ファイル（*.xml.out）	*運用ディレクトリ*/conf/xml/

> **メモ**
>
> 統合されたファイルの内容が想定どおりの結果となっていることを確認してください。ファイルの内容が正しくない場合，テンプレートファイルとパラメーターファイルを見直して修正してください。その後，手順 1 および手順 2 を再度実施してください。

  エラーの対処方法については，マニュアル『Hitachi Streaming Data Platform メッセージ』を参照してください。

- 内部アダプターのファイルのパラメーター値が正しく指定されていなかった場合は，hsdpstartinpro コマンドを実行した際に標準エラー出力（stderr）にエラーメッセージが出力されます。

次の作業

分析シナリオおよび内部アダプターのファイルをエクスポートします。

4.3　プロパティファイルの設定値を変更する

登録したクエリグループを削除して，プロパティファイルの設定値を変更します。

操作手順

1. hsdpstopinpro コマンドでアダプターグループを停止します。
   詳細については，「10.14　hsdpstopinpro」を参照 してください。
2. hsdpcqlstop コマンドでクエリグループを停止します。
   詳細については，「10.12　hsdpcqlstop 」を参照 してください。
3. hsdpcqldel コマンドで登録済みのクエリグループを削除します。
   詳細については，「10.10　hsdpcqldel」を参照 してください。
4. プロパティファイルの内容を変更します。
   クエリグループ用プロパティファイルの詳細については「12.9　クエリグループ用プロパティファイル」を，ストリーム用プロパティファイルの詳細については「12.10　ストリーム用プロパティファイル」を参照 してください。
5. hsdpcql コマンドでクエリグループを再登録します。
   詳細については，「10.9　hsdpcql」を参照 してください。
6. hsdpcqlstart コマンドでクエリグループを開始します。
   詳細については，「10.11　hsdpcqlstart」を参照 してください。
7. hsdpstartinpro コマンドでアダプターグループを起動します。
   詳細については，「10.13　hsdpstartinpro」を参照 してください。

操作結果

- パラメーター値が正しく指定されていた場合は，hsdpcqldebug コマンドがメッセージKFHD80002-I を標準出力（stdout）に出力します。
- パラメーター値が正しく指定されていなかった場合は，hsdpcqldebug コマンドを実行した際に標準エラー出力にエラーメッセージが出力されます。
  エラーの対処方法については，マニュアル『Hitachi Streaming Data Platform メッセージ』を参照してください。

4.4　分析シナリオおよび内部アダプターのファイルを出力する

インテグレーションデベロッパーは，分析シナリオおよび内部アダプターのファイルを出力します。

前提条件

前提条件を次に示します。

システム

- Streaming Data Platform が full インストールされていること。
- 統合テストに使用する運用ディレクトリがセットアップされていること。

ユーザー

- インテグレーションデベロッパーが，統合テストに使用する運用ディレクトリを作成したユーザーとしてホストにログインしていること。

操作手順

1. 分析シナリオおよび内部アダプターのファイルを出力するために，統合テスト用の運用ディレクトリごとに，-dir または-all オプションを指定して，hsdpexport コマンドを実行してください。
   a. 分析シナリオおよび内部アダプターのファイルだけを出力する（SDP マネージャーのファイルを除いて出力する）場合は，次のコマンドを実行してください。
      $ *運用ディレクトリ*/bin/hsdpexport -dir
   b. SDP マネージャーのファイルも含めて出力する場合は，次のコマンドを実行してください。
      $ *運用ディレクトリ*/bin/hsdpexport -all

操作結果

- 分析シナリオおよび内部アダプターのファイルが正しく出力されると，hsdpexport コマンドを実行した際に標準出力（stdout）にメッセージ KFHD99100-I が出力されます。また，アーカイブファイルであるhsdpexport_all.tar.gz※1 およびhsdpexport_dir.tar.gz※2 も*運用ディレクトリ*/export/に出力されます。
  注※1
  -all オプションが指定されている場合。
  注※2
  -dir オプションが指定されている場合。
- 分析シナリオおよび内部アダプターのファイルが正しく出力されなかった場合は，hsdpexport コマンドを実行した際に標準エラー出力（stderr）にエラーメッセージが出力されます。
  エラーの対処方法については，マニュアル『Hitachi Streaming Data Platform メッセージ』を参照してください。

次の作業

外部アダプターのファイルを出力します。

4.5　外部アダプターのファイルを出力する

インテグレーションデベロッパーは，外部アダプターのファイルを出力します。

前提条件

前提条件を次に示します。

システム

- Streaming Data Platform が full インストールされていること。
- 統合テストに使用する運用ディレクトリがセットアップされていること。

ユーザー

インテグレーションデベロッパーが，統合テストに使用する運用ディレクトリを作成したユーザーとしてホストにログインしていること。

> **メモ**
>
> 外部アダプターの実行に必要なファイルも出力したい場合は，外部アダプターの格納ディレクトリ下に格納してください。

操作手順

1. 統合テストに使用する運用ディレクトリごとに-exadp オプションを指定しながらhsdpexport コマンドを実行して，外部アダプターのファイルを出力します。

```
$ 運用ディレクトリ/bin/hsdpexport -exadp
```

操作結果

- 外部アダプターのファイルが正しく出力されると，hsdpexport コマンドを実行した際に標準出力（stdout）にメッセージ KFHD99100-I が出力されます。また，アーカイブファイルである*外部アダプターの格納ディレクトリ*_exadp.tar.gz が*運用ディレクトリ*/export/に出力されます。
- 外部アダプターのファイルが正しく出力されなかった場合は，hsdpexport コマンドを実行した際に，標準エラー出力（stderr）にエラーメッセージが出力されます。

  エラーの対処方法については，マニュアル『Hitachi Streaming Data Platform メッセージ』を参照してください。

次の作業

- HSDP をインストールする（構築フェーズ）

5 Streaming Data Platform をセットアップする（構築フェーズ）

この章では，構築フェーズでの Streaming Data Platform のセットアップ手順について説明します。デプロイメントエンジニアは，関連ファイルをインポートしたり，SDP マネージャーを開始したりして，Streaming Data Platform（構築フェーズ）をセットアップします。

5.1 分析シナリオおよび内部アダプターのファイルをインポートする（構築フェーズ）

デプロイメントエンジニアは，開発フェーズにエクスポートした分析シナリオおよび内部アダプターのファイルをインポートします。

前提条件

前提条件を次に示します。

システム

Streaming Data Platform がホストに full インストールされていること。

ユーザー

インテグレーションデベロッパーが開発フェーズに出力した分析シナリオおよび内部アダプターのアーカイブファイルを入手して，Streaming Data Platform の構築用ホストに格納していること。

操作手順

1. 分析シナリオおよび内部アダプターのファイルをインポートします。

   アーカイブファイルを展開するためのプログラム（アーカイバ）を使用してください。アーカイブファイルを展開することで作成されるディレクトリが運用ディレクトリとして使用されます。

   例として，アーカイブファイルの展開にtar コマンドを使用した場合の例を次に示します。

```
$ tar -zPxvf アーカイブ名
```

> **メモ**
>
> SDP マネージャーのファイルもアーカイブファイルに含まれている場合は，tar コマンドを実行してアーカイブファイルを展開する際，root 権限が必要です。root 権限でアーカイブファイルを展開した場合，運用ディレクトリの所有者がユーザーではなくroot になります。そうすると，一般ユーザーは運用ディレクトリにファイルを作成できません。この状態を回避するためには，次のコマンドをroot 権限で実行して，運用ディレクトリの権限をroot からユーザーに書き換えてください。
>
> ```
> # chown -R ユーザー名:グループ名 運用ディレクトリ
> ```

操作結果

開発フェーズにhsdpexport コマンドで作成されたアーカイブファイルが展開されます。

hsdpexport コマンドでアーカイブされるファイルの一覧についての詳細は，「10.16　hsdpexport」を参照してください。

次の作業

分析シナリオおよび内部アダプターのファイルのインポートが完了したら，構築フェーズとして SDP マネージャーを開始します。

5.2　SDP マネージャーを開始する（構築フェーズ）

デプロイメントエンジニアは，SDP マネージャー，SDP ブローカー，および SDP コーディネーターを開始します。

前提条件

前提条件を次に示します。

システム

Streaming Data Platform がホストに full インストールされていること。

ユーザー

- root ユーザーとしてホストにログインしていること。
- デフォルト値に関しての詳細は「12.14　SDP マネージャー定義ファイル」の「表 12-17　SDP マネージャー定義ファイルのプロパティ」を参照してください。デフォルト値は構築するシステムに応じて修正する必要があります。

操作手順

1. SDP マネージャー定義ファイルをテキストエディターで編集します。

> **メモ**
>
> デフォルト値から変更しない場合は，この操作は不要です。プロパティの編集に関する詳細については，「12.14　SDP マネージャー定義ファイル」の「表 12-17　SDP マネージャー定義ファイルのプロパティ」を参照してください。

SDP プロセスに割り当てる CPU を変更したい場合

次の例では，SDP プロセスに割り当てる CPU の番号を0 から1 に変更しています。

変更前

```
hsdp_cpu_no_list=0
```

変更後

```
hsdp_cpu_no_list=1
```

SDP の再起動設定を有効にしたい場合

次の例では，再起動設定をfalse からtrue に変更しています。

変更前

```
hsdp_restart=false
```

変更後

```
hsdp_restart=true
```

SDP のプロセス間通信に失敗した場合の最大リトライ回数を変更したい場合

次の例では，リトライ回数を30 から40 に変更しています。

変更前

```
hsdp_retry_times=30
```

変更後

```
hsdp_retry_times=40
```

SDP のプロセス間通信に失敗した場合にリトライする間隔（単位：秒）を変更したい場合

次の例では，リトライする間隔を3 秒から4 秒に変更しています。

変更前

```
hsdp_retry_interval=3
```

変更後

```
hsdp_retry_interval=4
```

連携するすべての SDP コーディネーターが開始するまでのタイムアウト時間（単位：秒）を変更したい場合

次の例では，タイムアウト時間を60 秒から100 秒に変更しています。

変更前

```
hsdp_coordinator_boot_timeout=60
```

変更後

```
hsdp_coordinator_boot_timeout=100
```

SDP マネージャーのログ設定を変更したい場合

次の例では，ログ面数を4 から3，ログサイズを16,777,216 から4,096，ログレベルを1 から2 に変更しています。

変更前

```
hsdp_manager_log_filecount=4
hsdp_manager_log_filesize=16777216
hsdp_manager_log_level=1
```

変更後

```
hsdp_manager_log_filecount=3
hsdp_manager_log_filesize=4096
hsdp_manager_log_level=2
```

SDP ブローカーのログ設定を変更したい場合

次の例では，ログ面数を4 から3，ログサイズを16,777,216 から4,096，ログレベルを1 から2 に変更しています。

変更前

```
hsdp_broker_log_filecount=4
hsdp_broker_log_filesize=16777216
hsdp_broker_log_level=1
```

変更後

```
hsdp_broker_log_filecount=3
hsdp_broker_log_filesize=4096
hsdp_broker_log_level=2
```

SDP コーディネーターのログ設定を変更したい場合

次の例では，ログ面数を4 から3，ログサイズを16,777,216 から4,096，ログレベルを1 から2 に変更しています。

変更前

```
hsdp_coordinator_log_filecount=4
hsdp_coordinator_log_filesize=16777216
hsdp_coordinator_log_level=1
```

変更後

```
hsdp_coordinator_log_filecount=3
hsdp_coordinator_log_filesize=4096
hsdp_coordinator_log_level=2
```

2. hsdpmanager コマンドを実行します。

```
# /opt/hitachi/hsdp/bin/hsdpmanager -start
```

メモ

SDP コーディネーターに接続する場合は，このコマンドをすべてのホストで実行する必要があります。

操作結果

SDP マネージャーが次のどちらかのメッセージを出力します。

- SDP マネージャーが正しく開始すると，メッセージKFHD92010-I が出力されます。
- SDP マネージャーが開始に失敗すると，エラーメッセージが標準エラー出力（stderr）に出力されます。エラーの対処方法については，マニュアル『Hitachi Streaming Data Platform メッセージ』を参照してください。

5.3 構築，チューニング後に動作確認する（構築フェーズ）

デプロイメントエンジニアは，構築，チューニングした SDP マネージャーの動作を確認します。

前提条件

前提条件を次に示します。

システム

- HSDP が full インストールされていること。
- SDP マネージャーが開始されていること。

ユーザー

チューニングが完了していること。

操作手順

1. SDP サーバを開始します。

   各運用ディレクトリで次のコマンドを実行して，SDP サーバを開始します。

   ```
   $ 運用ディレクトリ/bin/hsdpstart
   ```

2. クエリグループを登録します。

   各運用ディレクトリで次のコマンドを実行して，クエリグループを登録します。

   ```
   $ 運用ディレクトリ/bin/hsdpcql クエリグループ名
   ```

   クエリグループの登録後，hsdpstatusshow -qg コマンドを実行して，次のことを確認します。

   - スケールアップ数／スケールアウト数に対応した数の SDP サーバが開始されていること。
   - 開始した SDP サーバにクエリグループが登録されていること。

3. クエリグループを開始します。

   分析シナリオの出力側の運用ディレクトリから順に，次のコマンドを実行してクエリグループを開始します。

   ```
   $ 運用ディレクトリ/bin/hsdpcqlstart クエリグループ名
   ```

   クエリグループの開始後，hsdpstatusshow -qg コマンドを実行して，クエリグループが正常に開始しているか確認します。

4. 内部アダプターを開始します。

   各運用ディレクトリで次のコマンドを実行して，内部アダプターを開始します。

   ```
   $ 運用ディレクトリ/bin/hsdpstartinpro -file アダプター定義ファイル -qg クエリグループ名 -st ストリーム名 アダプターグループ名
   ```

内部アダプターの開始後，hsdpstatusshow -adp コマンドを実行して，内部アダプターのアダプターグループが正常に開始していることを確認します。

5. 外部アダプターを開始します。

各運用ディレクトリで，「7.9　外部アダプターを開始する（運用フェーズ）」を参照して外部アダプターを開始します。

外部アダプターの開始後，次のことを確認します。

- /var/log ディレクトリ下に出力される外部アダプターのログファイルで正常に開始されていること。
- /var/log ディレクトリ下に出力される外部アダプターのログファイルで入力ストリーム，または出力ストリームに正常に接続されていること。

6. 外部入力アダプターにテスト用データを入力します。

開始したそれぞれの外部入力アダプターに，テスト用のデータを入力します。

テスト用データの入力後，次のことを確認します。

- 外部出力アダプターが取得するデータが，分析シナリオに従って解析された正しい結果であること。
- 各運用ディレクトリ内のlogs/ディレクトリのログファイルにエラーメッセージが出力されていないこと。
- /var/log/hitachi/hsdp/ディレクトリ下のログファイルにエラーメッセージが出力されていないこと。

7. 外部アダプターを停止します。

「7.11　外部アダプターを停止する（運用フェーズ）」を参照して，外部アダプターを停止します。

外部アダプターの停止後，/var/log/ディレクトリ下に出力される外部アダプターのログファイルに，エラーメッセージが出力されていないことを確認します。

8. ストリームデータの分析を停止します。

「7.17　ストリームデータの分析を停止する（運用フェーズ）」を参照して，ストリームデータの分析を停止します。

ストリームデータの分析の停止後，各運用ディレクトリのlogs/ディレクトリの HSDP コマンドログファイルに，エラーメッセージが出力されていないことを確認します。

9. SDP サーバを停止します。

「7.18　SDP サーバを停止する（運用フェーズ）」を参照して，SDP サーバを停止します。

SDP サーバの停止後，各運用ディレクトリのlogs/ディレクトリの HSDP コマンドログファイルに，エラーメッセージが出力されていないことを確認します。

メモ

手順 1 から手順 9 のそれぞれの確認内容がどれか 1 つでも不正だった場合，メッセージ，およびログから問題の個所を特定して対策してください。対策の完了後，再び動作確認をしてください。

操作結果

動作確認が成功します。

次の作業

ネットワークパラメーターを設定します。

6 SDP コンポーネントを設定する

この章では，SDP コンポーネントの設定について説明します。

6.1　HSDP を runtime インストールする(構築フェーズ)

デプロイメントエンジニアが，SDP がインストールされていないホストに外部アダプターの構築環境を準備する場合，HSDP を runtime インストールします。HSDP の runtime インストールの詳細については，「1.1　Hitachi Streaming Data Platform をインストールする（full インストールおよび runtime インストール)」を参照してください。

7

SDP コンポーネントを操作する（運用フェーズ）

この章では，SDP コンポーネントの起動と操作について説明します。カスタマイズエンジニアは，SDP サーバ，SDP マネージャー，アダプターなどのさまざまなコンポーネントを起動して操作します。

7.1　ネットワークパラメーターを設定する（運用フェーズ）

カスタマイズエンジニアが HSDP システムを準備するために，顧客先のホストのネットワーク環境に合わせて，HSDP のネットワークパラメーターを設定します。

前提条件

前提条件は次のとおりです。

システム

- ホストに HSDP が full インストールされていること。
- SDP マネージャー，SDP ブローカー，SDP コーディネーター，およびすべての SDP サーバが停止していること。

ユーザー

- 構築フェーズで構築した HSDP システムを顧客先のホストに納入していること。
- root ユーザーでホストにログインしていること。

操作手順

1. **外部アダプターまたは内部カスケーディングアダプターがストリームに接続する際に使用するホスト名，または IP アドレスを設定します。**

   各ホスト上で，次のコマンド引数を指定して，hsdpsetup コマンドを実行してください。

   - -mgr オプション
   - -host オプションと，そのオプション引数としてホスト名，または IP アドレス

   外部アダプターまたは内部カスケーディングアダプターがストリームに接続する際に使用するホスト名を「myHost1」に設定する例を次に示します。

   ```
   # /opt/hitachi/hsdp/bin/hsdpsetup -mgr -host myHost1
   ```

   指定方法の詳細については，「10.5　hsdpsetup」の引数を参照してください。

2. **HSDP が使用するポート群の先頭ポート番号を設定します。**

   各ホスト上で，次のコマンド引数を指定してhsdpsetup コマンドを実行してください。

   - -mgr オプション
   - -port オプションと，そのオプション引数として，ポート番号

   HSDP が使用するポート群の先頭ポート番号を「20425」に設定する例を次に示します。

   ```
   # /opt/hitachi/hsdp/bin/hsdpsetup -mgr -port 20425
   ```

3. コーディネーターグループに所属する各ホスト上で，ホスト名または IP アドレスを設定します。

コーディネーターグループに所属するホストに対して，次のコマンド引数を指定してhsdpsetup コマンドを実行してください。

- -mgr オプション
- -chosts オプションと，そのオプション引数としてホスト名，または IP アドレス

コーディネーターグループに所属する，３つのホストのホスト名をそれぞれ「myHost1」，「myHost2」，「myHost3」に設定する例を次に示します。

```
# /opt/hitachi/hsdp/bin/hsdpsetup -mgr -chosts myHost1,myHost2,myHost3
```

4. コーディネーターグループに所属するホストの SDP コーディネーターが互いの生存監視をするためのマルチキャストアドレスを設定します。

コーディネーターグループに所属する各ホスト上で，次のコマンド引数を指定してhsdpsetup コマンドを実行してください。

- -mgr オプション
- -cmaddr オプションと，そのオプション引数として，マルチキャストアドレス

コーディネーターグループが使用するマルチキャストアドレスを「239.255.2.2」に設定する例を次に示します。

```
/opt/hitachi/hsdp/bin/hsdpsetup -mgr -cmaddr 239.255.2.2
```

操作結果

次のメッセージは，ネットワークパラメーターの設定に基づいて出力されます。

- ネットワークパラメーターの設定に成功した場合

  hsdpsetup コマンドがメッセージKFHD91009-I を標準出力（stdout）に出力します。

  ネットワークパラメーターの設定内容は，次のファイルに出力されます。

  ```
  /var/log/hitachi/hsdp/hsdpsetupmanager.log
  ```

  例：ネットワークパラメーターの設定

  ```
      :
  hsdp_host=myHost1
  hsdp_port=20425-20432
  hsdp_coordinator_group=myHost1,myHost2,myHost3
  hsdp_coordinator_multiplicity=2
  hsdp_coordinator_multicast_address=239.255.2.2
      :
  ```

  上記ファイルの変更内容に誤りがないか確認してください。変更内容に誤りがある場合は，ネットワークパラメーターの設定をし直してください。

- ネットワークパラメーターの設定に失敗した場合

hsdpsetup コマンドがエラーメッセージを標準エラー出力（stderr）に出力します。

エラーの対処方法については，マニュアル『Hitachi Streaming Data Platform メッセージ』を参照してください。

次の作業

ネットワークパラメーターを設定したあと，SDP マネージャーを開始します。（運用フェーズ）

7.2　コーディネーターグループを作成する（運用フェーズ）

オペレーターは，HSDP システムを冗長構成で運用するために，コーディネーターグループを作成します。

前提条件

前提条件を次に示します。

システム

- ホストに HSDP が full インストールされていること。
- コーディネーターグループに所属するホストの SDP マネージャー，SDP ブローカー，SDP コーディネーター，およびすべての SDP サーバーが停止していること。

ユーザー

root ユーザーでホストにログインすること。

操作手順

1. コーディネーターグループを作成します。

   作成するコーディネーターグループに所属するホストごとに，次のオプションと引数を指定して`hsdpsetup`コマンドを実行します。

   指定方法の詳細については「10.5　hsdpsetup」を参照してください。

   - `-host` オプションと，その引数として，外部アダプター，または内部カスケーディングアダプターがストリームに接続する際に使用するホストのホスト名，または IP アドレス。
   - `-port` オプションと，その引数として，HSDP が使用するポート群の先頭のポート番号。
   - `-chosts` オプションと，その引数として，作成するコーディネーターグループに属するホストのホスト名，または IP アドレス。

     > **メモ**
     >
     > 自ホストを含めて複数のホストをコーディネーターグループに所属させる場合は，自ホストのホスト名に`localhost` と指定しないでください。また，IP アドレスとしてループバックアドレスを指定しないでください。

   - `-cmulti` オプションと，その引数として，コーディネーターグループで保持するデータの多重度。

     > **メモ**
     >
     > コーディネーターグループで保持するデータの多重度は，「コーディネーターグループに属するホストの数≥(多重度×2)-1」を満たす最大値を推奨します。

- -cmaddr オプションと，その引数として，コーディネーターグループに所属するホストの SDP コーディネーターが，お互いの生存監視をするためのマルチキャストアドレス。

次のオプションと引数を指定する場合

- -host オプション
  外部アダプター，または内部カスケーディングアダプターがストリームに接続する際に使用するホスト名がmyHost1。
- -port オプション
  HSDP が使用するポート群の先頭のポート番号が20425。
- -chosts オプション
  コーディネーターグループに所属する 3 つのホストのホスト名がmyHost1，myHost2，myHost3。
- -cmulti オプション
  コーディネーターグループで保持するデータの多重度が2。
- -cmaddr オプション
  コーディネーターグループに所属するホストの SDP コーディネーターが，お互いの生存監視をするためのマルチキャストアドレスが239.255.2.1。

```
# /opt/hitachi/hsdp/bin/hsdpsetup -mgr -host myHost1 -port 20425 -chosts myHost1,myHost2,
myHost3 -cmulti 2 -cmaddr 239.255.2.1
```

操作結果

- コーディネーターグループの作成に成功した場合

  hsdpsetup コマンドがメッセージKFHD91009-I を標準出力（stdout）に出力します。ホストの追加後のコーディネーターグループの設定は，次のファイルに出力されます。

  ```
  /var/log/hitachi/hsdp/hsdpsetupmanager.log
  ```

  設定内容の例は次のとおりです。

  ```
      :
  hsdp_host=myHost1
  hsdp_port=20425-20432
  hsdp_coordinator_group=myHost1,myHost2,myHost3
  hsdp_coordinator_multiplicity=2
  hsdp_coordinator_multicast_address=239.255.2.1
      :
  ```

  上記ファイルの変更内容に誤りがないか確認してください。変更内容に誤りがある場合は，コーディネーターグループを作成し直してください。

- コーディネーターグループの作成に失敗した場合

  hsdpsetup コマンドが失敗の内容に応じたメッセージを標準エラー出力（stderr）に出力します。エラーメッセージに従って対処してください。

エラーの対処方法については，マニュアル『Hitachi Streaming Data Platform メッセージ』を参照してください。

次の作業

SDP マネージャーを開始します。

7.3 SDP マネージャーを開始する（運用フェーズ）

オペレーターは SDP マネージャーを起動して HSDP システムを開始します。

前提条件

前提条件は次のとおりです。

システム

ホストに HSDP が full インストールされていること。

ユーザー

root ユーザーでホストにログインしていること。

操作手順

1. SDP マネージャーを起動します。

   各ホスト上で，-start オプションを指定して，hsdpmanager コマンドを実行してください。

   ```
   # /opt/hitachi/hsdp/bin/hsdpmanager -start
   ```

   実行すると，SDP ブローカーと SDP コーディネーターも自動的に起動されます。

操作結果

SDP マネージャーは，次のどちらかを出力します。

- SDP マネージャーの開始に成功した場合，hsdpmanager コマンドがメッセージKFHD92010-I を標準出力（stdout）に出力します。
- SDP マネージャーの開始に失敗した場合，hsdpmanager コマンドが失敗内容に応じたメッセージを標準エラー出力（stderr）に出力します。

  エラーの対処方法については，マニュアル『Hitachi Streaming Data Platform メッセージ』を参照してください。

次の作業

SDP マネージャーを起動したあと，SDP マネージャーの稼働状態を確認します（運用フェーズ）。

7.4　SDP マネージャーの稼働状態を確認する（運用フェーズ）

オペレーターは，SDP マネージャーで問題が発生していないか確認するために，SDP マネージャーの稼働状態を確認します。

前提条件

前提条件は次のとおりです。

システム

- ホストに HSDP が full インストールされていること。
- SDP マネージャーが開始していること。

ユーザー

運用ディレクトリのアクセス権を持つユーザーでホストにログインしていること。

操作手順

1. SDP マネージャーの稼働状態を確認します。

   各ホストの運用ディレクトリで，-mgr オプションを指定して，hsdpstatusshow コマンドを実行してください。：

   ```
   $ 運用ディレクトリ/bin/hsdpstatusshow -mgr
   ```

操作結果

hsdpstatusshow コマンドが SDP マネージャーおよび SDP マネージャーの管理する SDP ブローカーと SDP コーディネーターの稼働状態を標準出力（stdout）に出力します。

コマンドが出力する次の項目のどれかの値が「Stopped」になっている場合，問題が発生しています。エラーメッセージを確認して問題に対処してください。変数については，「10.19　hsdpstatusshow」の「出力形式」を参照してください。

- Manager Status（変数「*aa...aa*」）
- Broker Status（変数「*dd...dd*」）
- Coordinator Status（変数「*gg...gg*」）

次の作業

SDP マネージャーの稼働状態を確認したあと，動作フェーズで SDP サーバを開始します。

7.5 SDP サーバを開始する（運用フェーズ）

オペレーターが HSDP システムを開始するために，SDP サーバを開始します。

前提条件

前提条件は次のとおりです。

システム

- ホストに HSDP が full インストールされていること。
- ホストに構築フェーズで構築した運用ディレクトリ，分析シナリオ，および内部アダプターのファイルがインポートされていること。
- SDP マネージャーが開始していること。

ユーザー

運用ディレクトリのアクセス権を持つユーザーでホストにログインしていること。

操作手順

1. SDP サーバを開始します。

   各ホストの運用ディレクトリでhsdpstart コマンドを実行してください。

   ```
   $ 運用ディレクトリ/bin/hsdpstart
   ```

操作結果

SDP サーバは，次のどちらかを出力します。

- SDP サーバの開始に成功した場合，hsdpstart コマンドがメッセージKFHD92010-I を標準出力（stdout）に出力します。
- SDP サーバの開始に失敗した場合，hsdpstart コマンドがエラーメッセージを標準エラー出力（stderr）に出力します。

  エラーの対処方法については，マニュアル『Hitachi Streaming Data Platform メッセージ』を参照してください。

次の作業

SDP サーバを起動したあと，ストリームデータの分析を開始します。（運用フェーズ）

7.6 ストリームデータの分析を開始する（運用フェーズ）

オペレーターは，HSDP システムを開始するために，ストリームデータの分析を開始します。

前提条件

前提条件は次のとおりです。

システム

- ホストに HSDP が full インストールされていること。
- 構築フェーズで構築した運用ディレクトリ，分析シナリオ，および内部アダプターのファイルがホストにインポートされていること。
- SDP マネージャー，および SDP サーバが開始していること。

ユーザー

運用ディレクトリのアクセス権を持つユーザーでホストにログインしていること。

操作手順

1. **クエリグループを登録します。**

   各ホストの運用ディレクトリで，登録するクエリグループ名をコマンド引数に指定して，hsdpcql コマンドを実行してください。

   クエリグループ名「myQueryGroup1」を登録する例を次に示します。

   ```
   $ 運用ディレクトリ/bin/hsdpcql myQueryGroup1
   ```

2. **登録したクエリグループを開始して，ストリームデータの分析を開始します。**

   各ホストの運用ディレクトリで，開始するクエリグループ名をコマンド引数に指定して，hsdpcqlstart コマンドを実行してください。

   クエリグループ名「myQueryGroup1」を開始する例を次に示します。

   ```
   $ 運用ディレクトリ/bin/hsdpcqlstart myQueryGroup1
   ```

操作結果

ストリームデータ分析結果によって，コマンドは次のどちらかを出力します。

- ストリームデータの分析の開始に成功した場合，hsdpcql コマンドおよびhsdpcqlstart コマンドがメッセージKFHD92010-I を標準出力（stdout）に出力します。
- ストリームデータの分析の開始に失敗した場合は，hsdpcql コマンドまたはhsdpcqlstart コマンドがエラーメッセージを標準エラー出力（stderr）に出力します。

エラーの対処方法については，マニュアル『Hitachi Streaming Data Platform メッセージ』を参照してください。

次の作業

ストリームデータ解析を開始したあと，内部アダプターを開始します（運用フェーズ）。

7.7 内部アダプターを開始する（運用フェーズ）

オペレーターは，HSDP システムを開始するために，内部アダプターを開始します。

前提条件

前提条件は次のとおりです。

システム

- ホストに HSDP が full インストールされていること。
- 構築フェーズで構築した運用ディレクトリ，内部アダプターのファイルがホストにインポートされていること。
- ストリームデータの分析が開始していること。

ユーザー

運用ディレクトリのアクセス権を持つユーザーでホストにログインしていること。

操作手順

1. 内部アダプターを開始します。

   各ホストの運用ディレクトリで，hsdpstartinpro コマンドを実行して，開発した内部アダプターを開始してください。

   > **メモ**
   >
   > アダプターを開始する際は，出力アダプターをすべて開始してから，入力アダプターを開始してください。

   内部アダプターが接続するクエリグループ名「myQuery」，ストリーム名「myStream」と，アダプター構成定義ファイル「myadp.xml」を指定して，内部アダプター「myAdaptor」を開始する例を次に示します。

   ```
   $ 運用ディレクトリ/bin/hsdpstartinpro -qg myQuery -st myStream -file myadp.xml myAdaptor
   ```

操作結果

内部アダプターの開始を実行すると，コマンドは次のどちらかのメッセージを出力します。

- 内部アダプターの開始に成功した場合，hsdpstartinpro コマンドがメッセージKFHD92010-I を標準出力（stdout）に出力します。
- 内部アダプターの開始に失敗した場合，hsdpstartinpro コマンドがメッセージKFHD92022-E を標準エラー出力（stderr）に出力します。

エラーの対処方法については，マニュアル『Hitachi Streaming Data Platform メッセージ』を参照してください。

次の作業

内部アダプターを起動したあと，SDP マネージャーの稼働状態を確認します（運用フェーズ）。

7.8 SDP サーバの稼働状態を確認する（運用フェーズ）

オペレーターは，SDP サーバで問題が発生していないか確認するために，SDP サーバの稼働状態を確認します。

前提条件

前提条件は次のとおりです。

システム

- ホストに HSDP が full インストールされていること。
- ホストに構築フェーズで構築した運用ディレクトリ，分析シナリオ，および内部アダプターのファイルがインポートされていること。
- SDP マネージャー，および SDP サーバが開始していること。
- SDP サーバでストリームデータの分析が開始していること。

ユーザー

運用ディレクトリのアクセス権を持つユーザーでホストにログインしていること。

操作手順

1. SDP サーバの稼働状態を確認します。

   各ホストの運用ディレクトリで，-svr オプションを指定して，hsdpstatusshow コマンドを実行してください。

   ```
   $ 運用ディレクトリ/bin/hsdpstatusshow -svr
   ```

2. クエリグループの稼働状態を確認します。

   各ホストの運用ディレクトリで，-qg オプションを指定して，hsdpstatusshow コマンドを実行してください。

   ```
   $ 運用ディレクトリ/bin/hsdpstatusshow -qg
   ```

3. 内部アダプターの稼働状態を確認します。

   各ホストの運用ディレクトリで，-adp オプションを指定して，hsdpstatusshow コマンドを実行してください。

   ```
   $ 運用ディレクトリ/bin/hsdpstatusshow -adp
   ```

操作結果

hsdpstatusshow コマンドが SDP サーバ，クエリグループ，および内部アダプターの稼働状態をそれぞれ標準出力（stdout）に出力します。

コマンドが出力する次の項目のどれかの値が「Stopped」になっている場合，問題が発生しています。エラーメッセージを確認して問題に対処してください。変数については，「10.19　hsdpstatusshow」の「出力形式」を参照してください。

- SDP サーバ（変数「*jj...jj*」） のSTATUS（変数「*kk...kk*」）
- クエリグループ（変数「*uu...uu*」）のSTATUS（変数「*vv...vv*」）
- アダプターグループ（変数「*XX...XX*」）の STATUS（変数「*YY...YY*」）

次の作業

SDP サーバのステータスを確認したあと，外部アダプターを開始します（運用フェーズ）。

7.9 外部アダプターを開始する（運用フェーズ）

オペレーターは，HSDP システムを開始するために，外部アダプターを開始します。

前提条件

前提条件は次のとおりです。

システム

- ホストに HSDP が full インストールまたは runtime インストールされていること。
- ホストに構築フェーズで構築した運用ディレクトリ，外部アダプターのファイルがインポートされていること。
- 外部アダプターの接続先の SDP サーバで，ストリームデータの分析が開始していること。

ユーザー

外部アダプターのアクセス権を持つユーザーでホストにログインしていること。

操作手順

1. 外部出力アダプターを開始します。

   外部出力アダプターのファイルをインポートした運用ディレクトリごとに，外部出力アダプターの実行を開始してください。

   外部アダプター格納ディレクトリ「output」に格納されている外部出力アダプターを同ディレクトリの実行コマンド「script.sh」で開始する例を次に示します。なお，「script.sh」は，外部アダプターのプログラムを実行するために，ユーザーが任意に作成するコマンドです。

   ```
   $ 運用ディレクトリ/exadaptor/output/script.sh
   ```

2. 外部入力アダプターを開始します。

   外部入力アダプターのファイルをインポートした運用ディレクトリごとに，外部入力アダプターの実行を開始してください。

   外部アダプター格納ディレクトリ「input」に格納されている外部入力アダプターを同ディレクトリの実行コマンド「script.sh」で開始する例を次に示します。なお，「script.sh」は，外部アダプターのプログラムを実行するために，ユーザーが任意に作成するコマンドです。

   ```
   $ 運用ディレクトリ/exadaptor/input/script.sh
   ```

操作結果

外部アダプターは，次のどちらかを出力します。

- 外部アダプターの開始に成功した場合，外部アダプターがメッセージKFHD13016-I を外部アダプターのログファイルに出力します。
- 外部アダプターの開始に失敗した場合，外部アダプターが標準エラー出力（stderr）または外部アダプターのログファイルに，警告メッセージまたはエラーメッセージを出力します。

  エラーの対処方法については，マニュアル『Hitachi Streaming Data Platform メッセージ』を参照してください。

次の作業

外部アダプターを起動したあと，外部アダプターの稼働状態を確認します（運用フェーズ）。

7.10　外部アダプターの稼働状態を確認する（運用フェーズ）

オペレーターは，外部アダプターで問題が発生していないか確認するために，外部アダプターの稼働状態を確認します。

前提条件

前提条件は次のとおりです。

システム

- ホストに HSDP が full インストールまたは runtime インストールされていること。
- ホストに構築フェーズで構築した運用ディレクトリ，外部アダプターのファイルがインポートされていること。
- 外部アダプターが開始していること。
- 外部アダプターの接続先の SDP サーバで，ストリームデータの分析が開始していること。

ユーザー

外部アダプターのアクセス権を持つユーザーでホストにログインしていること。

操作手順

1. 外部アダプターの稼働状態を確認します。

   次の外部アダプターのログファイルの内容を確認してください。

   ```
   /var/log/hitachi/hsdp/exadaptor/外部アダプター名/ExAdaptorMessageN※.log
   ```

   注※

   N はログファイルの通し番号を示します。

   *外部アダプター名*は，外部アダプター定義ファイルのプロパティ「hsdp.adaptor.name」に指定した名称です。

操作結果

外部アダプターで問題が発生している場合，外部アダプターがエラーメッセージをログファイルに出力します。

エラーメッセージの詳細については， マニュアル『Hitachi Streaming Data Platform メッセージ』を参照し てください。

次の作業

外部アダプターの状態を確認したあと，外部アダプターを停止します（運用フェーズ）。

7.11　外部アダプターを停止する（運用フェーズ）

オペレーターは，HSDP システムを停止するために，外部アダプターを停止します。

前提条件

前提条件は次のとおりです。

システム

- ホストに HSDP が full インストールまたは runtime インストールされていること。
- ホストに構築フェーズで構築した運用ディレクトリ，外部アダプターのファイルがインポートされていること。
- 外部アダプターが開始していること。
- 外部アダプターの接続先の SDP サーバで，ストリームデータの分析が開始していること。

ユーザー

外部アダプターのアクセス権を持つユーザーでホストにログインしていること。

操作手順

1. 外部入力アダプターを停止します。

   外部入力アダプターのファイルをインポートした運用ディレクトリごとに，「3.4　外部アダプターを開発する」で，インテグレーションデベロッパーが開発した外部入力アダプターの仕様に従い，外部入力アダプターを停止してください。

   外部アダプター格納ディレクトリ「input」に格納されている外部入力アダプターを同ディレクトリの実行コマンド「script_stop.sh」で停止する例を次に示します。なお，「script_stop.sh」は，外部アダプターのプログラムを実行するために，ユーザーが任意に作成するコマンドです。

   ```
   $ 運用ディレクトリ/exadaptor/input/script_stop.sh
   ```

2. 外部出力アダプターを停止します。

   外部出力アダプターのファイルをインポートした運用ディレクトリごとに，「3.4　外部アダプターを開発する」で，インテグレーションデベロッパーが開発した外部出力アダプターの仕様に従い，外部出力アダプターを停止してください。

   外部アダプター格納ディレクトリ「output」に格納されている外部出力アダプターを同ディレクトリの実行コマンド「script_stop.sh」で停止する例を次に示します。なお，「script_stop.sh」は，外部アダプターのプログラムを実行するために，ユーザーが任意に作成するコマンドです。

   ```
   $ 運用ディレクトリ/exadaptor/output/script_stop.sh
   ```

操作結果

外部アダプターは，次のどちらかを出力します。

- 外部アダプターの停止に成功した場合，外部アダプターがメッセージKFHD13016-I を外部アダプターのログファイルに出力します。
- 外部アダプターの停止に失敗した場合，外部アダプターが標準エラー出力（stderr）または外部アダプターのログファイルに，警告メッセージまたはエラーメッセージを出力します。

  エラーの対処方法については，マニュアル『Hitachi Streaming Data Platform メッセージ』を参照してください。

次の作業

外部アダプターを停止したあと，内部アダプターを停止します（運用フェーズ）。

7.12 HSDP がインストールされていないサーバに外部アダプターのアーカイブファイルをインポートする

デプロイメントエンジニアが，開発フェーズでエクスポートした外部アダプターのアーカイブファイルを構築環境のホストにインポートします。

前提条件

前提条件は次のとおりです。

システム

Java Runtime Environment(JRE) 1.8.0_92 以上が構築環境にインストールされていること。

ユーザー

- 外部アダプターのアーカイブファイルを構築環境のホストに転送していること。
- アーカイブファイルの展開先ディレクトリのアクセス権を持つユーザーでホストにログインしていること。

操作手順

1. 実行に必要なライブラリをインポートします。

   外部アダプターの実行に必要なライブラリを HSDP がインストールされているサーバから取得します。必要なライブラリとライブラリの格納先を次の表に示します。

   表 7-1 HSDP がインストールされていないサーバ上の外部アダプターの動作に必要なライブラリ

項番	ライブラリ	格納場所
1	hsdp-exadp.jar	/opt/hitachi/hsdp/lib/
2	hntrlib2j64.jar	/opt/hitachi/HNTRLib2/classes/

2. 外部アダプターのアーカイブファイルをインポートします。

   アーカイバを使用して，外部アダプターのアーカイブファイルを任意のディレクトリに展開します。

   tar コマンドを使用して，カレントディレクトリにアーカイブファイルを展開する例を次に示します。

```
$ tar -zxvf アーカイブ名
```

3. 外部アダプターが接続するブローカーのホストを設定します。

   展開した運用ディレクトリに格納されている外部アダプター定義ファイルのプロパティ「hsdp.broker.address」に，外部アダプターが接続するブローカーのホスト名または IP アドレスを指定します。

   外部アダプターが接続するブローカーのホスト名「myHost1」を指定する例を次に示します。

```
hsdp.broker.address=myHost1:20425
```

操作結果

アーカイブファイルの展開先に，次のディレクトリが作成されます。展開されるディレクトリには，外部アダプターのファイル一式が格納されます。

任意のディレクトリ/外部アダプター名/

次の作業

HSDP がインストールされていないサーバで，外部アダプターのアーカイブファイルを構築環境のホストにインポートしたあと，外部アダプターを起動します（運用フェーズ）。

7.13 HSDP がインストールされていないサーバで外部アダプターを開始する（運用フェーズ）

オペレーターは，HSDP がインストールされていないサーバで，外部アダプターを開始します。

前提条件

前提条件は次のとおりです。

システム

- Java Runtime Environment(JRE) 1.8.0_92 以上が構築環境にインストールされていること。
- ホストに外部アダプターのファイルがインポートされていること。
- 外部アダプターの実行に必要なライブラリがインポートされていること。
- 外部アダプターの接続先の SDP サーバで，ストリームデータの分析が開始されていること。

ユーザー

外部アダプターのアクセス権を持つユーザーでホストにログインしていること。

操作手順

1. 外部出力アダプターを開始します。

   外部出力アダプターのファイルをインポートしたディレクトリごとに，外部出力アダプターを開始してください。

   外部アダプター収納ディレクトリ「output」に格納されている外部出力アダプターを，同ディレクトリの実行コマンド「script.sh」で開始する例を次に示します。なお，「script.sh」は，外部アダプターのプログラムを実行するために，ユーザーが任意に作成するコマンドです。

   ```
   $ 任意のディレクトリ/exadaptor/output/script.sh
   ```

2. 外部入力アダプターを開始します。

   外部入力アダプターのファイルをインポートしたディレクトリごとに，外部入力アダプターを開始してください。

   外部アダプター収納ディレクトリ「input」に格納されている外部入力アダプターを，同ディレクトリの実行コマンド「script.sh」で開始する例を次に示します。なお，「script.sh」は，外部アダプターのプログラムを実行するために，ユーザーが任意に作成するコマンドです。

   ```
   $ 任意のディレクトリ/exadaptor/input/script.sh
   ```

操作結果

外部アダプターは，次のどちらかを出力します。

- 外部アダプターの開始に成功した場合，外部アダプターがメッセージKFHD13016-I を外部アダプターのログファイルに出力します。
- 外部アダプターの開始に失敗した場合，外部アダプターが標準エラー出力（stderr）または外部アダプターのログファイルに，警告メッセージまたはエラーメッセージを出力します。

  エラーの対処方法については，マニュアル『Hitachi Streaming Data Platform メッセージ』を参照してください。

次の作業

HSDP がインストールされていないサーバで，外部アダプターを起動したあと，外部アダプターの稼働状態を確認します（運用フェーズ）。

7.14 HSDP がインストールされていないサーバで起動した外部アダプターの稼働状態を確認する（運用フェーズ）

オペレーターは，外部アダプターで問題が発生していないか確認するために，外部アダプターの稼働状態を確認します。

前提条件

前提条件は次のとおりです。

システム

- Java Runtime Environment(JRE) 1.8.0_92 以上が構築環境にインストールされていること。
- 外部アダプターの実行に必要なライブラリがインポートされていること。
- ホストに外部アダプターのファイルがインポートされていること。
- 外部アダプターが開始されていること。
- 外部アダプターの接続先の SDP サーバで，ストリームデータの分析が開始されていること。

ユーザー

外部アダプターのアクセス権を持つユーザーでホストにログインしていること。

操作手順

1. 外部アダプターの稼働状態を確認します。

   次の外部アダプターのログファイルの内容を確認してください。

   ```
   /var/log/hitachi/hsdp/exadaptor/外部アダプター名/ExAdaptorMessageN※.log
   ```

   注※

   N はログファイルの通し番号を示します。

   *外部アダプター名*は，外部アダプター定義ファイルのプロパティ「hsdp.adaptor.name」に指定した名称です。

操作結果

外部アダプターで問題が発生している場合，外部アダプターがエラーメッセージをログファイルに出力します。

エラーメッセージの詳細については，マニュアル『Hitachi Streaming Data Platform メッセージ』を参照してください。

次の作業

外部アダプターの稼働状態を確認したあと，外部アダプターを停止します（運用フェーズ）。

7.15　HSDP がインストールされていないサーバで起動した外部アダプターを停止する（運用フェーズ）

オペレーターは，HSDP がインストールされていないサーバで起動した外部アダプターを停止します。

前提条件

前提条件は次のとおりです。

システム

- Java Runtime Environment(JRE) 1.8.0_92 以上が構築環境にインストールされていること。
- ホストに外部アダプターのファイルがインポートされていること。
- 外部アダプターの実行に必要なライブラリがインポートされていること。
- 外部アダプターが開始されていること。
- 外部アダプターの接続先の SDP サーバで，ストリームデータの分析が開始されていること。

ユーザー

外部アダプターのアクセス権を持つユーザーでホストにログインしていること。

操作手順

1. 外部入力アダプターを停止します。

   外部入力アダプターのファイルをインポートしたディレクトリごとに，「3.4　外部アダプターを開発する」で，インテグレーションデベロッパーが開発した外部入力アダプターの仕様に従い，外部入力アダプターを停止してください。

   外部アダプター格納ディレクトリ「input」に格納されている外部入力アダプターを，同ディレクトリの実行コマンド「script_stop.sh」で停止する例を次に示します。なお，「script_stop.sh」は，外部アダプターのプログラムを実行するために，ユーザーが任意に作成するコマンドです。

   ```
   $ 任意のディレクトリ/exadaptor/input/script_stop.sh
   ```

2. 外部出力アダプターを停止します。

   外部出力アダプターのファイルをインポートしたディレクトリごとに，「3.4　外部アダプターを開発する」で，インテグレーションデベロッパーが開発した外部出力アダプターの仕様に従い，外部出力アダプターを停止してください。

   外部アダプター格納ディレクトリ「output」に格納されている外部出力アダプターを，同ディレクトリの実行コマンド「script_stop.sh」で停止する例を次に示します。なお，「script_stop.sh」は，外部アダプターのプログラムを実行するために，ユーザーが任意に作成するコマンドです。

   ```
   $ 任意のディレクトリ/exadaptor/output/script_stop.sh
   ```

操作結果

外部アダプターは，次のどちらかを出力します。

- 外部アダプターの停止に成功した場合，外部アダプターがメッセージKFHD13019-I を外部アダプターのログファイルに出力します。
- 外部アダプターの停止に失敗した場合，外部アダプターが標準エラー出力（stderr）または外部アダプターのログファイルに，警告メッセージまたはエラーメッセージを出力します。

  エラーメッセージの詳細については，マニュアル『Hitachi Streaming Data Platform メッセージ』を参照してください。

次の作業

HSDP がインストールされていないサーバで起動した外部アダプターを停止したあと，内部アダプターを停止します（運用フェーズ）。

7.16　内部アダプターを停止する（運用フェーズ）

オペレーターは，HSDP システムを停止するために，内部アダプターを停止します。

前提条件

前提条件は次のとおりです。

システム

- ホストに HSDP が full インストールされていること。
- ホストに構築フェーズで構築した運用ディレクトリ，内部アダプターのファイルがインポートされていること。
- SDP サーバで，ストリームデータの分析が開始していること。
- 内部アダプターが開始していること。

ユーザー

運用ディレクトリのアクセス権を持つユーザーでホストにログインしていること。

操作手順

1. 内部アダプターを停止します。

   各ホストの運用ディレクトリで，hsdpstopinpro コマンドを実行して，開発した内部アダプターを停止してください。

   内部アダプターmyAdaptor を停止する例を次に示します。

```
$ 運用ディレクトリ/bin/hsdpstopinpro myAdaptor
```

> **メモ**
>
> アダプターを停止する際は，入力アダプターをすべて停止してから，出力アダプターを停止してください。

操作結果

hsdpstopinpro コマンドは，次のメッセージを出力します。

- 内部アダプターの停止に成功した場合，hsdpstopinpro コマンドがメッセージKFHD92010-I を標準出力（stdout）に出力します。
- 内部アダプターの停止に失敗した場合，hsdpstopinpro コマンドがメッセージKFHD92022-E を標準エラー出力（stderr）に出力します。

エラーの対処方法については，マニュアル『Hitachi Streaming Data Platform メッセージ』を参照してください。

次の作業

内部アダプターを停止したあと，ストリームデータの分析を停止します（運用フェーズ）。

7.17　ストリームデータの分析を停止する（運用フェーズ）

オペレーターは，HSDP システムを停止するために，ストリームデータの分析を停止します。

前提条件

前提条件は次のとおりです。

システム

- ホストに HSDP が full インストールされていること。
- SDP マネージャーおよび SDP サーバが開始していること。
- SDP サーバでストリームデータの分析が開始していること。
- 外部アダプターが停止していること。

ユーザー

運用ディレクトリのアクセス権を持つユーザーでホストにログインしていること。

操作手順

1. クエリグループを停止して，ストリームデータの分析を停止します。

   各ホストの運用ディレクトリで，停止するクエリグループ名をコマンド引数に指定して，hsdpcqlstop コマンドを実行してください。実行すると，停止するクエリグループに接続している内部アダプターも停止されます。

   クエリグループ名「myQueryGroup1」を停止する例を次に示します。

   ```
   $ 運用ディレクトリ/bin/hsdpcqlstop myQueryGroup1
   ```

2. 停止したクエリグループを削除します。

   各ホストの運用ディレクトリで，削除するクエリグループ名をコマンド引数に指定して，hsdpcqlstop コマンドを実行してください。

   クエリグループ名「myQueryGroup1」を削除する例を次に示します。

   ```
   $ 運用ディレクトリ/bin/hsdpcqldel myQueryGroup1
   ```

操作結果

hsdpcqlstop コマンドは，次のどちらかを出力します。

- ストリームデータの分析の停止に成功した場合，hsdpcqlstop コマンドがメッセージKFHD92010-I を標準出力（stdout）に出力します。

- ストリームデータの分析の停止に失敗した場合，hsdpcqlstop コマンドがエラーメッセージを標準エラー出力（stderr）に出力します。

  エラーの対処方法については，マニュアル『Hitachi Streaming Data Platform メッセージ』を参照してください。

次の作業

ストリームデータ解析を停止したあと，SDP サーバを停止します（運用フェーズ）。

7.18 SDP サーバを停止する（運用フェーズ）

オペレーターは，HSDP システムを停止するために，SDP サーバを停止します。

前提条件

前提条件は次のとおりです。

システム

- ホストに HSDP が full インストールされていること。
- SDP マネージャーおよび SDP サーバが開始していること。
- SDP サーバでストリームデータの分析が停止していること。

ユーザー

運用ディレクトリのアクセス権を持つユーザーでホストにログインしていること。

操作手順

1. SDP サーバを停止します。

   各ホストの運用ディレクトリで，hsdpstop コマンドを実行してください。

   ```
   $ 運用ディレクトリ/bin/hsdpstop
   ```

操作結果

SDP サーバの停止を実行すると，コマンドは次のどちらかを出力します。

- SDP サーバの停止に成功した場合，hsdpstop コマンドがメッセージKFHD92010-I を標準出力（stdout）に出力します。
- SDP サーバの停止に失敗した場合，hsdpstop コマンドがエラーメッセージを標準エラー出力（stderr）に出力します。

  エラーの対処方法については，マニュアル『Hitachi Streaming Data Platform メッセージ』を参照してください。

次の作業

SDP サーバを停止したあと，SDP マネージャーを停止します（運用フェーズ）。

7.19 SDP マネージャーを停止する（運用フェーズ）

オペレーターは，HSDP システムを停止するために，SDP マネージャーを停止します。

前提条件

前提条件は次のとおりです。

システム

- ホストに HSDP が full インストールされていること。
- SDP マネージャーが開始していること。
- SDP サーバが停止していること。

ユーザー

root ユーザーでホストにログインしていること。

操作手順

1. SDP マネージャーを停止します。

   各ホストの運用ディレクトリで，-stop オプションを指定して，hsdpmanager コマンドを実行してください。

```
# /opt/hitachi/hsdp/bin/hsdpmanager -stop
```

操作結果

SDP マネージャーの停止を実行すると，コマンドは次のどちらかを出力します。

- SDP マネージャーの停止に成功した場合，hsdpmanager コマンドがメッセージKFHD92010-I を標準出力（stdout）に出力します。
- SDP マネージャーの停止に失敗した場合，hsdpmanager コマンドが失敗内容に応じたメッセージを標準エラー出力（stderr）に出力します。

  エラーの対処方法については，マニュアル『Hitachi Streaming Data Platform メッセージ』を参照してください。

次の作業

SDP マネージャーを停止したあと，次の操作を実行します。

- HSDP システムのネットワーク設定を変更する場合，ネットワークパラメーターを変更する（運用フェーズ）。
- HSDP システムを冗長構成で運用する場合，コーディネーターグループを作成する（運用フェーズ）。

- HSDP システムの冗長構成を変更する場合
  - コーディネーターグループにホストを静的に追加する（運用フェーズ）。
  - コーディネーターグループにホストを動的に追加する（運用フェーズ）。
  - コーディネーターグループからホストを削除する（運用フェーズ）。
  - コーディネーターグループで保持するデータの多重度を変更する（運用フェーズ）。
- HSDP システムのスケーラブル構成を変更する場合
  - スケールアップ構成に変更する。
  - スケールアウト構成に変更する。
- HSDP システムを消去する場合
  - 運用ディレクトリをアンセットアップする。
  - HSDP をアンインストールする。

8 HSDP をメンテナンスする

この章では，HSDP をメンテナンスする手順を説明します。HSDP の導入から運用までの各フェーズで，それぞれの目的に応じてメンテナンスを実施します。

8.1　メンテナンスの作業一覧

ここでは，HSDP の導入から運用までのメンテナンスの作業一覧を示します。

表 8-1　メンテナンスの作業一覧

項番	目的	作業
1	ホストを HSDP の導入前の状態に戻す	運用ディレクトリをアンセットアップする
2		HSDP SDK をアンインストールする
3		HSDP をアンインストールする
4	HSDP の環境をバックアップ・リストアする	運用ディレクトリをバックアップする
5		運用ディレクトリをリストアする
6	HSDP の環境を他ホストへ移行する	運用ディレクトリをエクスポートする
7		エクスポートしたファイルを転送する
8		運用ディレクトリをインポートする
9	HSDP に設定されている IP アドレス情報を変更する	HSDP に設定されている IP アドレス情報を変更する
10	HSDP に設定されているホスト名を変更する	HSDP に設定されているホスト名を変更する
11	HSDP が使用するポート番号を変更する	HSDP が使用するポート番号を変更する
12	HSDP の環境をバージョン 2 からアップグレードする	HSDP をアップグレードする
13		運用ディレクトリを移行する
14	HSDP のログ出力先を変更する	HSDP のログ出力先を変更する
15	HSDP の保守情報を詳細化する	HSDP のログレベルを上げる
16	HSDP で集計・分析したタプルを確認する	タプルログを取得する
17		タプルログの内容を表示する
18	HSDP に登録されたクエリにタプルログの内容を再投入する	HSDP に登録されたクエリにタプルログの内容を再投入する

8.2　ホストを HSDP の導入前の状態に戻す

ホストを HSDP の導入前の状態に戻すには，次の順に操作を実施します。

- 「運用ディレクトリをアンセットアップする」
- 「HSDP SDK をアンインストールする」
- 「HSDP をアンインストールする」

8.2.1　運用ディレクトリをアンセットアップする

HSDP システムから HSDP の運用ディレクトリをアンセットアップします。

前提条件

前提条件は次のとおりです。

システム

- ホストに HSDP が full インストールされていること。
- ホストに運用ディレクトリがセットアップされていること。
- SDP サーバが停止していること。

  hsdpstatusshow コマンドに-svr オプションを指定して実行し，運用ディレクトリ内の SDP サーバが停止していることを確認してください。なお，hsdpstatusshow コマンドは，SDP マネージャーが起動した状態で実行してください。

ユーザー

- 運用ディレクトリをセットアップしたユーザーが，手順を実行すること。
- 運用ディレクトリの絶対パスを確認すること。
- 運用ディレクトリをアンセットアップするため，必要に応じて運用ディレクトリのバックアップを取ってあること。
- カレントディレクトリとして，運用ディレクトリ以外のディレクトリを使用していること。

操作手順

1. 運用ディレクトリをアンセットアップします。

   各ホストの運用ディレクトリに対して，アンセットアップする運用ディレクトリの絶対パスをコマンド引数に指定して，hsdpunsetup コマンドを実行します。

   運用ディレクトリ「/home/user/hsdp」をアンセットアップする例を次に示します。

```
$ /opt/hitachi/hsdp/bin/hsdpunsetup -dir /home/user/hsdp
```

操作結果

次のメッセージが出力されます。

- 運用ディレクトリのアンセットアップに成功した場合，hsdpunsetup コマンドがメッセージKFHD91056-I を標準出力（stdout）に出力します。
- 運用ディレクトリのアンセットアップに失敗した場合，hsdpunsetup コマンドが失敗の内容に応じたメッセージを標準エラー出力（stderr）に出力します。エラーメッセージに従って対処してください。

  エラーの対処方法については，マニュアル『Hitachi Streaming Data Platform メッセージ』を参照してください。

次の作業

HSDP SDK をアンインストールします。

8.2.2 HSDP SDK をアンインストールする

開発用の HSDP システムから HSDP SDK をアンインストールします。

前提条件

前提条件は次のとおりです。

システム

なし。

ユーザー

- root ユーザーでホストにログインしていること。
- カレントディレクトリを HSDP のインストールディレクトリ「/opt/hitachi/hsdp/sdk/」以外のディレクトリに変更していること。

操作手順

1. HSDP SDK をアンインストールします。

   HSDP SDK のアンインストーラー「uninstall.sh」コマンドを実行して，HSDP SDK をアンインストールします。

```
# /opt/hitachi/hsdp/sdk/uninstall/uninstall.sh
```

操作結果

- HSDP SDK のアンインストールに成功した場合，アンインストーラーが次のメッセージを標準出力（stdout）に出力します。

```
Hitachi Streaming Data Platform software development kit uninstallation completed success
fully.
```

- HSDP SDK のアンインストールに失敗した場合，アンインストーラーが失敗の内容に応じたメッセージを標準エラー出力（stderr）に出力します。エラーメッセージに従って対処してください。

  エラーの対処方法については，マニュアル『Hitachi Streaming Data Platform メッセージ』を参照してください。

次の作業

HSDP をアンインストールします。

8.2.3 HSDP をアンインストールする

HSDP システムから HSDP をアンインストールします。

前提条件

前提条件は次のとおりです。

システム

- ホストに HSDP が full インストール，または runtime インストールされていること。
- すべての SDP サーバが停止していること。
- SDP マネージャーが停止していること。

ユーザー

- root ユーザーでホストにログインしていること。
- カレントディレクトリを HSDP のインストールディレクトリ「/opt/hitachi/hsdp/」以下以外のディレクトリに変更していること。

操作手順

1. HSDP をアンインストールします。

   HSDP のアンインストーラーuninstall.sh コマンドを実行して，HSDP をアンインストールします。

```
# /opt/hitachi/hsdp/uninstall/uninstall.sh
```

操作結果

- HSDP のアンインストールに成功した場合，アンインストーラーが次のメッセージを標準出力（stdout）に出力します。
  - full インストールの場合

    ```
    Hitachi Streaming Data Platform full uninstallation completed successfully.
    ```

  - runtime インストールの場合

    ```
    Hitachi Streaming Data Platform runtime uninstallation completed successfully.
    ```

- HSDP のアンインストールに失敗した場合，アンインストーラーが次のメッセージを標準エラー出力（stderr）に出力します。
  - full インストールの場合

    ```
    Hitachi Streaming Data Platform full uninstallation failed.
    ```

  - runtime インストールの場合

    ```
    Hitachi Streaming Data Platform runtime uninstallation failed.
    ```

  エラーの対処方法については，マニュアル『Hitachi Streaming Data Platform メッセージ』を参照してください。

次の作業

なし。

8.3 HSDP の環境をバックアップ・リストアする

HSDP の環境をバックアップ，およびリストアするには，次の順に操作を実施します。

- 「運用ディレクトリをバックアップする」
- 「運用ディレクトリをリストアする」

8.3.1 運用ディレクトリをバックアップする

運用ディレクトリをバックアップします。運用ディレクトリのバックアップには，運用ディレクトリだけをバックアップする場合と，運用ディレクトリと同時に SDP マネージャー関連ファイルをバックアップする場合の 2 種類があります。SDP マネージャー関連ファイルとは次のファイルを示します。

- SDP マネージャー定義ファイル

  ```
  /opt/hitachi/hsdp/conf/hsdp_setup_manager.cfg
  ```

- /var/log/hitachi/hsdp/以下のすべてのファイル

前提条件

前提条件は次のとおりです。

システム

- 運用ディレクトリをバックアップする場合，SDP サーバが停止していること。

  hsdpstatusshow コマンドに-svr オプションを指定して実行し，運用ディレクトリ内の SDP サーバが停止していることを確認してください。なお，hsdpstatusshow コマンドは，SDP マネージャーが起動した状態で実行してください。

- 運用ディレクトリをバックアップする場合，対象の運用ディレクトリのセットアップ，およびアンセットアップ（hsdpsetup -dir コマンド，またはhsdpunsetup コマンド）が実行中でないこと。
- 運用ディレクトリと SDP マネージャー関連ファイルを同時にバックアップする場合，SDP マネージャーのセットアップ（hsdpsetup -mgr コマンド）が実行中でないこと。また，SDP マネージャー，SDP ブローカー，および SDP コーディネーターが停止していること。

ユーザー

- 運用ディレクトリだけをバックアップする場合，root ユーザー，または運用ディレクトリに対するアクセス権限を持つ一般ユーザーで実行すること。
- 運用ディレクトリと SDP マネージャー関連ファイルを同時にバックアップする場合，root ユーザーで実行すること。

操作手順

1. *運用ディレクトリ*/export/にhsdpexport_dir.tar.gz ファイル，またはhsdpexport_all.tar.gz ファイルが格納されている場合，運用ディレクトリ以外のディレクトリに移動します。

   運用ディレクトリ/export/にhsdpexport_dir.tar.gz ファイル，またはhsdpexport_all.tar.gz ファイルが格納されている場合，手順 2 でhsdpexport コマンドを実行するとユーザーへの確認なしでファイルが上書きされます。

2. hsdpexport コマンドを実行し，運用ディレクトリをバックアップします。

   - 運用ディレクトリだけをバックアップする場合

     ```
     $ 運用ディレクトリ/bin/hsdpexport -dir
     ```

   - 運用ディレクトリと SDP マネージャー関連ファイルを同時にバックアップする場合

     ```
     # 運用ディレクトリ/bin/hsdpexport -all
     ```

3. *運用ディレクトリ*/export/に出力されたhsdpexport_dir.tar.gz ファイル，または hsdpexport_all.tar.gz ファイルを，運用ディレクトリ以外の任意のディレクトリにバックアップファイルとして格納します。

   - 運用ディレクトリだけをバックアップする場合

     ```
     hsdpexport_dir.tar.gz
     ```

   - 運用ディレクトリと SDP マネージャー関連ファイルを同時にバックアップする場合

     ```
     hsdpexport_all.tar.gz
     ```

操作結果

- hsdpexport コマンドが正常終了し，バックアップに成功した場合，KFHD99100-I メッセージが標準出力（stdout）に出力されます。
- 運用ディレクトリだけをバックアップした場合は，運用ディレクトリ，および運用ディレクトリ以下に格納されているすべてのファイルがhsdpexport_dir.tar.gz に格納されます。
- 運用ディレクトリと SDP マネージャー関連ファイルを同時にバックアップした場合は，運用ディレクトリ，SDP マネージャー関連ファイル，および運用ディレクトリ以下に格納されているすべてのファイルがhsdpexport_all.tar.gz に格納されます。
- hsdpexport コマンドが異常終了し，バックアップに失敗した場合，失敗の内容に応じたメッセージが標準エラー出力（stderr）に出力されます。エラーメッセージに従って対処してください。

  エラーの対処方法については，マニュアル『Hitachi Streaming Data Platform メッセージ』を参照してください。

次の作業

運用ディレクトリをリストアします。

8.3.2　運用ディレクトリをリストアする

運用ディレクトリ，および SDP マネージャー関連ファイルをリストアします。

前提条件

前提条件は次のとおりです。

システム

- HSDP が full インストールされていること。
- 運用ディレクトリをリストアする場合，SDP サーバが停止していること。

  hsdpstatusshow コマンドに-svr オプションを指定して実行し，運用ディレクトリ内の SDP サーバが停止していることを確認してください。なお，hsdpstatusshow コマンドは，SDP マネージャーが起動した状態で実行してください。
- SDP マネージャー関連ファイルをリストアする場合，SDP マネージャー，SDP ブローカー，SDP コーディネーター，およびすべての SDP サーバが停止していること。

  hsdpstatusshow コマンド（オプション指定なし）を実行して，SDP マネージャー，SDP ブローカー，および SDP コーディネーターが停止していることを確認してください。また，セットアップ済みの各運用ディレクトリでhsdpstatusshow コマンドに-svr オプションを指定して実行し，SDP サーバが停止していることを確認してください。

ユーザー

- 事前に運用ディレクトリ，または SDP マネージャー関連ファイルのバックアップを実行していること。
- 運用ディレクトリと SDP マネージャー関連ファイルを同時にリストアする場合，root ユーザーで実行すること。
- 運用ディレクトリだけをリストアする場合，root ユーザー，または運用ディレクトリに対するアクセス権限を持つ一般ユーザーで実行すること。

操作手順

1. バックアップファイルを展開します。

```
$ tar -zPxvf アーカイブ名 -C 運用ディレクトリの格納先ディレクトリ
```

- *アーカイブ名*

  バックアップファイルを指定します。
- *運用ディレクトリの格納先ディレクトリ*

  運用ディレクトリを格納する，運用ディレクトリの上位ディレクトリを指定します。
- なお，SDP マネージャー関連ファイルもバックアップファイルに含まれている場合（hsdpexport_all.tar.gz），root 権限でこの操作を実行します。バックアップファイルの展開後，次のコマンドを実行して，運用ディレクトリの権限を root からユーザーに書き換えます。

```
# chown -R ユーザー名:グループ名運用ディレクトリ
```

2. SDP マネージャー関連ファイルをリストアする場合，次のコマンドを実行して SDP マネージャーを開始します。

```
# /opt/hitachi/hsdp/bin/hsdpmanager -start
```

3. SDP マネージャー関連ファイルをリストアする場合，次のコマンドを実行して SDP サーバを開始します。

```
$ 運用ディレクトリ/bin/hsdpstart
```

操作結果

- 運用ディレクトリをリストアした場合，運用ディレクトリ，および運用ディレクトリ以下に格納されているすべてのファイルとディレクトリがリストアされます。
- バックアップファイルに SDP マネージャー関連ファイルが含まれている場合，運用ディレクトリのほか，SDP マネージャー関連ファイルもリストアされます。

次の作業

なし。

8.4 HSDP の環境を他ホストへ移行する

HSDP の環境を他ホストへ移行するには，次の順に操作を実施します。

- 「運用ディレクトリをエクスポートする」
- 「エクスポートしたファイルを転送する」
- 「運用ディレクトリをインポートする」

8.4.1 運用ディレクトリをエクスポートする

運用ディレクトリをエクスポートします。運用ディレクトリのエクスポートには，運用ディレクトリだけをエクスポートする場合と，運用ディレクトリと同時に SDP マネージャー関連ファイルをエクスポートする場合の 2 種類があります。SDP マネージャー関連ファイルとは次のファイルを示します。

- SDP マネージャー定義ファイル

  ```
  /opt/hitachi/hsdp/conf/hsdp_setup_manager.cfg
  ```

- /var/log/hitachi/hsdp/以下のすべてのファイル
- SDP マネージャーのセットアップで指定した情報（hsdpsetup -mgr，またはhsdpsetup -mgr の各オプションで設定した情報）

  /opt/hitachi/hsdp/system/以下の一部のファイル（内部管理用ファイル）

前提条件

前提条件は次のとおりです。

システム

- SDP サーバが停止していること。

  hsdpstatusshow コマンドに-svr オプションを指定して実行し，運用ディレクトリ内の SDP サーバが停止していることを確認してください。なお，hsdpstatusshow コマンドは，SDP マネージャーが起動した状態で実行してください。

- 対象の運用ディレクトリのセットアップ，およびアンセットアップ（hsdpsetup -dir コマンド，またはhsdpunsetup コマンド）が実行中でないこと。
- 運用ディレクトリと SDP マネージャー関連ファイルを同時にエクスポートする場合，SDP マネージャーのセットアップ（hsdpsetup -mgr コマンド）が実行中でないこと。また，SDP マネージャー，SDP ブローカー，および SDP コーディネーターが停止していること。

ユーザー

- 運用ディレクトリだけをエクスポートする場合，root ユーザー，または運用ディレクトリに対するアクセス権限を持つ一般ユーザーで実行すること。

- 運用ディレクトリと SDP マネージャー関連ファイルを同時にエクスポートする場合，root ユーザーで実行すること。

操作手順

1. *運用ディレクトリ*/export/にhsdpexport_dir.tar.gz ファイル，またはhsdpexport_all.tar.gz ファイルが格納されている場合，運用ディレクトリ以外のディレクトリに移動します。

   運用ディレクトリ/export/にhsdpexport_dir.tar.gz ファイル，またはhsdpexport_all.tar.gz ファイルが格納されている場合，手順 2 でhsdpexport コマンドを実行するとユーザーへの確認なしでファイルが上書きされます。

2. hsdpexport コマンドを実行し，運用ディレクトリ，および SDP マネージャー関連ファイルをエクスポートします。

   - 運用ディレクトリだけをエクスポートする場合

     ```
     $ 運用ディレクトリ/bin/hsdpexport -dir
     ```

   - 運用ディレクトリと SDP マネージャー関連ファイルを同時にエクスポートする場合

     ```
     # 運用ディレクトリ/bin/hsdpexport -all
     ```

   > **メモ**
   >
   > 運用ディレクトリと SDP マネージャー関連ファイルを同時にエクスポートする場合に，SDP マネージャーのファイルもアーカイブファイルに含まれているときは，root 権限で上記のコマンドを実行してアーカイブファイルを展開したあと，次のコマンドを root 権限のまま実行して運用ディレクトリの権限を root からユーザーに書き換えます。
   >
   > ```
   > # chown -R 変更後のユーザーID:変更後のグループID運用ディレクトリ
   > ```

3. *運用ディレクトリ*/export/に出力されたhsdpexport_dir.tar.gz ファイル，または hsdpexport_all.tar.gz ファイルを，運用ディレクトリ以外の任意のディレクトリにエクスポートファイルとして格納します。

   - 運用ディレクトリだけをエクスポートする場合

     ```
     hsdpexport_dir.tar.gz
     ```

   - 運用ディレクトリと SDP マネージャー関連ファイルを同時にエクスポートする場合

     ```
     hsdpexport_all.tar.gz
     ```

操作結果

- hsdpexport コマンドが正常終了し，エクスポートに成功した場合，KFHD99100-I メッセージが標準出力（stdout）に出力されます。

- 運用ディレクトリだけをエクスポートした場合は，運用ディレクトリ，および運用ディレクトリ以下に格納されているすべてのファイルがhsdpexport_dir.tar.gz に格納されます。
- 運用ディレクトリと SDP マネージャー関連ファイルを同時にエクスポートした場合は，運用ディレクトリ，SDP マネージャー関連ファイル，および運用ディレクトリ以下に格納されているすべてのファイルがhsdpexport_all.tar.gz に格納されます。
- hsdpexport コマンドが異常終了し，エクスポートに失敗した場合，失敗の内容に応じたメッセージが標準エラー出力（stderr）に出力されます。エラーメッセージに従って対処してください。

  エラーの対処方法については，マニュアル『Hitachi Streaming Data Platform メッセージ』を参照してください。

次の作業

エクスポートしたファイルを転送します。

8.4.2　エクスポートしたファイルを転送する

エクスポートした運用ディレクトリのファイル，および SDP マネージャー関連ファイルを転送します。

前提条件

前提条件は次のとおりです。

システム

なし

ユーザー

- 事前に運用ディレクトリ，または SDP マネージャー関連ファイルのエクスポートを実行していること。
- 転送先のホストのユーザー ID，ホスト名または IP アドレス，ホスト内の格納ディレクトリを確認すること。

操作手順

1. scp コマンドを実行して，運用ディレクトリをエクスポートしたファイル，または運用ディレクトリと SDP マネージャー関連ファイルをエクスポートしたファイルを転送先のホストに転送し，任意のディレクトリに格納します。

   メモ

   インストール対象の運用ディレクトリパス下には格納しないでください。

   ```
   $ scp エクスポートファイル名 転送先ホストのユーザーID@転送先ホストのホスト名またはIPアドレス:転送先ホストの格納先ディレクトリ
   ```

操作結果

- エクスポートファイルがscp コマンドで指定した*転送先ホストの格納先ディレクトリ*に格納されます。

次の作業

運用ディレクトリをインポートします。

8.4.3　運用ディレクトリをインポートする

転送した運用ディレクトリをインポートします。運用ディレクトリのインポートには，運用ディレクトリだけをインポートする場合と，運用ディレクトリと同時に SDP マネージャー関連ファイルをインポートする場合の 2 種類があります。

前提条件

前提条件は次のとおりです。

システム

- HSDP が full インストールされていること。
- SDP サーバが停止していること。

  hsdpstatusshow コマンドに-svr オプションを指定して実行し，運用ディレクトリ内の SDP サーバが停止していることを確認してください。なお，hsdpstatusshow コマンドは，SDP マネージャーが起動した状態で実行してください。
- 運用ディレクトリと SDP マネージャー関連ファイルを同時にインポートする場合，SDP マネージャー，SDP ブローカー，SDP コーディネーター，およびすべての SDP サーバが停止していること。

  hsdpstatusshow コマンド（オプション指定なし）を実行して，SDP マネージャー，SDP ブローカー，および SDP コーディネーターが停止していることを確認してください。また，セットアップ済みの各運用ディレクトリでhsdpstatusshow コマンドに-svr オプションを指定して実行し，SDP サーバが停止していることを確認してください。

ユーザー

- 事前に運用ディレクトリのエクスポートファイル，または運用ディレクトリと SDP マネージャー関連ファイルのエクスポートファイルをインポート先のホストの任意のディレクトリに格納していること。
- 運用ディレクトリだけをインポートする場合，root ユーザー，または運用ディレクトリに対するアクセス権限を持つ一般ユーザーで実行すること。
- 運用ディレクトリと SDP マネージャー関連ファイルをインポートする場合，root ユーザーで実行すること。

操作手順

1. 転送先のホストの任意のディレクトリに格納されたエクスポートファイルを展開します。

```
# tar -zPxvf エクスポートファイル名-C 運用ディレクトリの格納先ディレクトリ
```

2. 運用ディレクトリと SDP マネージャーの関連ファイルをインポートする場合，展開した運用ディレクトリの権限を変更するときは，chown コマンドで権限を変更します。

```
# chown -R 変更後のユーザーID:変更後のグループID運用ディレクトリ
```

3. 運用ディレクトリと SDP マネージャーの関連ファイルをインポートする場合，インポート先のホストの情報を SDP マネージャーのセットアップで設定します。

次のコマンドを実行して，インポート先のホストの情報を設定します。

```
# /opt/hitachi/hsdp/bin/hsdpsetup -mgr インポート先のホストの情報
```

*インポート先のホストの情報*は，hsdpsetup コマンドの次のオプションを使って指定します。

-host オプション

外部アダプター，または内部カスケーディングアダプターがストリームに接続する際に使用するホスト名，または IP アドレス。

-port オプション

外部アダプター，または内部カスケーディングアダプターがストリームに接続する際に使用するポート番号。

-chosts オプション

接続するコーディネーターのホストのホスト名，または IP アドレス。

-cmulti オプション

連携中のコーディネーター内でのデータの多重度。

-cmaddr オプション

コーディネーターグループに所属する SDP コーディネーターが，お互いの生存監視を行うためのマルチキャストアドレス。

> **メモ**
>
> オプションの詳細は，「10.5　hsdpsetup」を参照してください。

4. 運用ディレクトリと SDP マネージャーの関連ファイルをインポートする場合，SDP マネージャーを開始します。

次のコマンドを実行して SDP マネージャーを開始します。

```
# /opt/hitachi/hsdp/bin/hsdpmanager -start
```

5. SDP サーバを開始します。

次のコマンドを実行して SDP サーバを開始します。

```
# 運用ディレクトリ/hsdpstart
```

操作結果

- SDP マネージャーの開始，または SDP サーバの開始が成功した場合，KFHD92010-I メッセージが標準出力（stdout）に出力されます。
- 運用ディレクトリだけをインポートした場合は，運用ディレクトリ，および運用ディレクトリ以下に格納されているすべてのファイルとディレクトリがインポートされます。
- 運用ディレクトリと SDP マネージャー関連ファイルを同時にインポートした場合は，運用ディレクトリ，SDP マネージャー関連ファイル，および運用ディレクトリ以下に格納されているすべてのファイルがインポートされます。
- SDP マネージャーの開始，または SDP サーバの開始が失敗した場合，失敗の内容に応じたメッセージが標準エラー出力（stderr）に出力されます。エラーメッセージに従って対処してください。

  エラーの対処方法については，マニュアル『Hitachi Streaming Data Platform メッセージ』を参照してください。

次の作業

HSDP に設定されている IP アドレス情報を変更します。

8.5　HSDP に設定されている IP アドレス情報を変更する

サポートエンジニアは，HSDP に設定されている IP アドレス情報を変更します。

前提条件

前提条件は次のとおりです。

システム

- ホストに HSDP が full インストールされていること。
- すべての SDP サーバが停止していること。

ユーザー

root ユーザーでホストにログインすること。

操作手順

1. SDP マネージャーが起動している場合はhsdpmanager コマンドを実行し，SDP マネージャーを停止します。

```
# /opt/hitachi/hsdp/bin/hsdpmanager -stop
```

2. hsdpsetup コマンドを実行して，IP アドレス情報を変更します。

```
# /opt/hitachi/hsdp/bin/hsdpsetup -mgr -host IPアドレス
```

> **メモ**
>
> IP アドレスを変更するホストがコーディネーターグループに所属している場合，「7.2　コーディネーターグループを作成する（運用フェーズ）」を参照して，コーディネーターグループの設定も再度実行してください。

3. 外部アダプターが接続する SDP ブローカーの宛先を変更します。

   外部アダプター定義ファイルのhsdp.broker.address に，変更後の IP アドレスを指定します。

```
hsdp.broker.address=IPアドレス:ポート番号
```

操作結果

- hsdpsetup コマンドが正常終了し，IP アドレスの変更に成功した場合，KFHD91009-I メッセージが標準出力（stdout）に出力されます。
- SDP マネージャーのセットアップ内容は，次のファイルに出力されます。

  /var/log/hitachi/hsdp/hsdpsetupmanager.log

- hsdpsetup コマンドが異常終了し，IP アドレスの変更に失敗した場合，失敗の内容に応じたメッセージが標準エラー出力（stderr）に出力されます。エラーメッセージに従って対処してください。

  エラーの対処方法については，マニュアル『Hitachi Streaming Data Platform メッセージ』を参照してください。

次の作業

HSDP に設定されているホスト名を変更します。

8.6　HSDP に設定されているホスト名を変更する

サポートエンジニアは，HSDP に設定されているホスト名を変更します。

前提条件

前提条件は次のとおりです。

システム

- ホストに HSDP が full インストールされていること。
- すべての SDP サーバが停止していること。

ユーザー

root ユーザーでホストにログインすること。

操作手順

1. SDP マネージャーが起動している場合はhsdpmanager コマンドを実行し，SDP マネージャーを停止します。

```
# /opt/hitachi/hsdp/bin/hsdpmanager -stop
```

2. hsdpsetup コマンドを実行して，ホスト名を変更します。

```
# /opt/hitachi/hsdp/bin/hsdpsetup -mgr -host ホスト名
```

> **メモ**
>
> ホスト名を変更するホストがコーディネーターグループに所属している場合，「7.2　コーディネーターグループを作成する（運用フェーズ）」を参照して，コーディネーターグループの設定も再度実行してください。

3. IP アドレスが変わる場合は，外部アダプターが接続する SDP ブローカーの宛先を変更します。

外部アダプター定義ファイルのhsdp.broker.address に，変更後の IP アドレスを指定します。

```
hsdp.broker.address=IPアドレス:ポート番号
```

操作結果

- hsdpsetup コマンドが正常終了し，ホスト名の変更に成功した場合，KFHD91009-I メッセージが標準出力（stdout）に出力されます。
- SDP マネージャーのセットアップ内容は，次のファイルに出力されます。

  /var/log/hitachi/hsdp/hsdpsetupmanager.log

- hsdpsetup コマンドが異常終了し，ホスト名の変更に失敗した場合，失敗の内容に応じたメッセージが標準エラー出力（stderr）に出力されます。エラーメッセージに従って対処してください。

  エラーの対処方法については，マニュアル『Hitachi Streaming Data Platform メッセージ』を参照してください。

次の作業

HSDP が使用するポート番号を変更します。

8.7　HSDP が使用するポート番号を変更する

サポートエンジニアは，HSDP が使用するポート番号を変更します。

前提条件

前提条件は次のとおりです。

システム

- ホストに HSDP が full インストールされていること。
- すべての SDP サーバが停止していること。

ユーザー

root ユーザーでホストにログインすること。

操作手順

1. SDP マネージャーが起動している場合hsdpmanager コマンドを実行し，SDP マネージャーを停止します。

> **重要**
>
> 連携している SDP コーディネーターがある場合は，すべての SDP コーディネーターのホストの SDP マネージャーを停止させる必要があります。

```
# /opt/hitachi/hsdp/bin/hsdpmanager -stop
```

2. hsdpsetup コマンドを実行し，ポート番号を変更します。

連携させるすべての SDP コーディネーターのホストで実行する必要があります。

```
# /opt/hitachi/hsdp/bin/hsdpsetup -mgr -port ポート番号
```

3. 外部アダプターが接続する SDP ブローカーの宛先を変更します。

外部アダプター定義ファイルのhsdp.broker.address に，変更後の IP アドレスを指定します。

```
hsdp.broker.address=IPアドレス:ポート番号
```

4. 動作を確認します。

「5.3　構築，チューニング後に動作確認する（構築フェーズ）」を参照し，動作を確認します。

操作結果

- hsdpsetup コマンドが正常終了し，ポート番号の変更に成功した場合，KFHD91009-I メッセージが標準出力（stdout）に出力されます。
- SDP マネージャーのセットアップ内容は，次のファイルに出力されます。

/var/log/hitachi/hsdp/hsdpsetupmanager.log

- 動作確認の結果は，「5.3　構築，チューニング後に動作確認する（構築フェーズ）」を参照してください。
- hsdpsetup コマンドが異常終了し，ポート番号の変更に失敗した場合，失敗の内容に応じたメッセージが標準エラー出力（stderr）に出力されます。エラーメッセージに従って対処してください。

  エラーの対処方法については，マニュアル『Hitachi Streaming Data Platform メッセージ』を参照してください。

次の作業

なし。

8.8　HSDP の環境をバージョン 2 からアップグレードする

HSDP の環境をバージョン 2 からアップグレードするには，次の順に操作を実施します。

- 「HSDP をアップグレードする」
- 「運用ディレクトリを移行する」

8.8.1　HSDP をアップグレードする

HSDP の環境をバージョン 2 からバージョン 3 にアップグレードします。

前提条件

前提条件は次のとおりです。

システム

バージョン 2 の HSDP がインストールされていること。

ユーザー

バージョン 2 の HSDP のインストールディレクトリ/opt/hitachi/hsdp/以下のファイルを削除するため，必要に応じてファイルをバックアップすること。

操作手順

1. root ユーザーで OS にログインします。
2. バージョン 2 の HSDP をアンインストールします。
   カレントディレクトリを/opt/hitachi/hsdp/以外のディレクトリへ移動してから，次のコマンドを実行してバージョン 2 の HSDP をアンインストールしてください。

   ```
   # /opt/hitachi/hsdp/uninstall/uninstall.sh
   ```

   コマンドを実行すると，インストールディレクトリ/opt/hitachi/hsdp/以下のファイルが削除されます。
3. バージョン 3 の HSDP のインストール用 CD を挿入します。
   インストール用 CD は自動的にマウントされます。
   CD が自動的にマウントされない場合，次のコマンドを実行して手動でマウントします。

   ```
   mount -t iso9660 /dev/cdrom CD-ROMマウントディレクトリ
   ```
4. 次のコマンドを実行して，インストーラーのあるディレクトリにカレントディレクトリを移動します。

   ```
   # cd CD-ROMマウントディレクトリHSDP/
   ```

5. バージョン 3 の HSDP をインストールします。

install.sh コマンドを実行して，バージョン 3 の HSDP をインストールします。

```
# install.sh
```

操作結果

- アップグレードに成功した場合は，次のメッセージが標準出力（stdout）に出力されます。

```
Hitachi Streaming Data Platform full installation completed successfully.
```

- アップグレードで作成されるディレクトリは次のとおりです。なお，ディレクトリ構成の詳細については，「1.3　運用ディレクトリを作成する」を参照してください。

```
運用ディレクトリ
/opt/
  |_ hitachi/
    |_ hsdp/
    |_ hsdpbase/
    |_ HNTRLib2/
/var/
  |_ opt/
    |_ hitachi/
      |_ hsdp/
      |_ HNTRLib2/
```

- アップグレードに失敗した場合は，失敗の内容に応じたメッセージが標準エラー出力（stderr）に出力されます。エラーメッセージに従って対処してください。

  エラーの対処方法については，マニュアル『Hitachi Streaming Data Platform メッセージ』を参照してください。

次の作業

運用ディレクトリを移行します。

8.8.2　運用ディレクトリを移行する

バージョン 2 の運用ディレクトリを，バージョン 3 の運用ディレクトリに移行します。

前提条件

前提条件は次のとおりです。

システム

- バージョン 3 の HSDP がインストールされていること。
- SDP サーバが停止していること。

hsdpstatusshow コマンドに-svr オプションを指定して実行し，運用ディレクトリ内の SDP サーバが停止していることを確認してください。なお，hsdpstatusshow コマンドは，SDP マネージャーが起動した状態で実行してください。

ユーザー

運用ディレクトリに対するアクセス権限を持つユーザーが実行すること。

操作手順

1. **移行に必要なファイルを，バージョン 2 の HSDP の運用ディレクトリから抽出します。**

   バージョン 2 の HSDP の運用ディレクトリから，次のファイルを抽出してバックアップします。

   - クエリグループ用プロパティファイル
   - クエリ定義ファイル
   - 外部定義関数定義ファイル
   - アダプター構成定義ファイル
   - インプロセス連携用プロパティファイル
   - E メールグループ設定ファイル
   - E メールテキストテンプレートファイル
   - SNMP トラッププロパティファイル
   - cascading 用プロパティファイル
   - 外部定義関数が格納されたライブラリ
   - カスタムアダプターが格納されたライブラリ
   - その他，運用ディレクトリに格納されているユーザー独自のファイル

2. **バージョン 2 の HSDP の運用ディレクトリを退避します。**

   バージョン 3 の HSDP の運用ディレクトリに入れ替えるため，次のコマンドを実行してバージョン 2 の HSDP の運用ディレクトリの名称を変更し，退避します。運用ディレクトリは，任意の名称を指定できます。

   ```
   $ mv 退避前のバージョン2の運用ディレクトリ 退避後のバージョン2の運用ディレクトリ.bk
   ```

3. **運用ディレクトリをセットアップします。**

   次のコマンドを実行し，手順 2 で退避した運用ディレクトリのパスにバージョン 3 の運用ディレクトリを新規に作成します。

   ```
   $ /opt/hitachi/hsdp/bin/hsdpsetup {-dir 退避したバージョン2の運用ディレクトリ[-sc サーバクラスター名] | -help}
   ```

   必要に応じて，-sc オプションにサーバクラスタ名を指定します。なお，このオプションへの指定は省略できます。

4. **手順 1 でバージョン 2 の運用ディレクトリから抽出したファイルを，バージョン 3 の運用ディレクトリに格納します。**

   抽出したファイルは，次の格納場所に格納します。

   - 定義ファイル

     定義ファイルごとに指定された格納場所
   - ユーザー独自のファイル

     運用ディレクトリの任意の場所

5. **手順 4 で格納した定義ファイルに記載されているパスが正しいかを確認して，必要に応じて修正します。**

   バージョン 3 の運用ディレクトリの構造は，バージョン 2 の構造と異なるため，定義ファイルに記載されている相対パスがずれている場合があります。このため，定義ファイルの内容と，定義の値としてパスを指定している個所を確認してください。 特に，次の定義の値は必ず見直してください。

   - クエリグループ用プロパティファイルの`querygroup.cqlFilePath`
   - インプロセス連携用プロパティファイルの`user_app.classpath_dir`
   - 外部定義関数定義ファイルに指定する，外部定義関数定義が格納されたライブラリのパス
   - アダプター構成定義ファイルに指定する，各種ファイルへのパス

6. **分析シナリオやアダプターが，正常に動作することを確認します。**

   確認方法は「3.5　統合テスト」を参照してください。

7. **バージョン 2 の HSDP の運用ディレクトリを削除します。**

   次のコマンドを実行して，退避しておいたバージョン 2 の運用ディレクトリを削除します。

   ```
   $ rm -r 退避後のバージョン2の運用ディレクトリ.bk
   ```

操作結果

- バージョン 2 の HSDP の運用ディレクトリに格納されていた各定義ファイルとユーザー独自のファイルが，バージョン 3 の HSDP の運用ディレクトリに格納されます。バージョン 3 の HSDP の運用ディレクトリ構成の詳細は，「1.3　運用ディレクトリを作成する」を参照してください。

次の作業

なし。

8.9 HSDP のログ出力先を変更する

オペレーターまたはサポートエンジニアは，ログファイルを任意のディレクトリに格納するために，ログファイルの出力先を変更します。

前提条件

前提条件は次のとおりです。

システム

SDP マネージャー，SDP ブローカー，SDP コーディネーターのログファイル，および SDP マネージャーコマンドログファイルの出力先を変更する場合

- ホストに HSDP が full インストールされていること。
- SDP マネージャー，SDP ブローカー，および SDP コーディネーターが停止していること。

SDP マネージャー，SDP ブローカー，および SDP コーディネーターの日立 JavaVM ログファイルの出力先を変更する場合

- ホストに HSDP が full インストールされていること。
- SDP マネージャー，SDP ブローカー，および SDP コーディネーターが停止していること。

共通コンポーネント（コマンド・ロガー）のログファイル，SDP サーバのログファイルの出力先を変更する場合

- ホストに HSDP が full インストールされていること。
- SDP サーバが停止していること。

SDP サーバの日立 JavaVM ログファイルの出力先を変更する場合

- ホストに HSDP が full インストールされていること。
- SDP サーバが停止していること。

外部アダプターのログファイルの出力先を変更する場合

- 外部アダプターが停止していること。

ユーザー

特になし。

操作手順

1. 各種定義ファイルに，ログファイルの出力先を設定します。

   SDP マネージャー，SDP ブローカー，SDP コーディネーターのログファイルの出力先を変更する場合

root ユーザーで次のパラメーターを変更してください。パラメーターに指定する出力先は，root ユーザーに書き込み権限がある出力先を指定してください。

- SDP マネージャー定義ファイルのhsdp_logfile_dir パラメーター

各パラメーターの詳細については，次の個所を参照してください。

- 「12.14　SDP マネージャー定義ファイル」

SDP マネージャー，SDP ブローカー，SDP コーディネーターの日立 JavaVM ログファイルの出力先を変更する場合

root ユーザーで次のパラメーターを変更してください。パラメーターに指定する出力先は，root ユーザーに書き込み権限がある出力先を指定してください。

- SDP マネージャー用 JavaVM オプションファイルのMANAGER_JVM_LOG パラメーター
- SDP ブローカー用 JavaVM オプションファイルのBROKER_JVM_LOG パラメーター
- SDP コーディネーター用 JavaVM オプションファイルのCOORDINATOR_JVM_LOGGER_DIR パラメーター

各パラメーターの詳細については，次の個所を参照してください。

- 「12.5　SDP マネージャー用 JavaVM オプションファイル（manager_jvm.cfg）」
- 「12.6　SDP ブローカー用 JavaVM オプションファイル（broker_jvm.cfg）」
- 「12.7　SDP コーディネーター用 JavaVM オプションファイル（coordionator_jvm.cfg）」

SDP マネージャーコマンドログファイルの出力先を変更する場合

root ユーザーで次のパラメーターを変更してください。パラメーターに指定する出力先は，root ユーザーに書き込み権限がある出力先を指定してください。

- SDP マネージャーコマンドのログ定義ファイルのhsdpLogfileDir パラメーター

パラメーターの詳細については，「12.16　SDP マネージャーコマンドのログ定義ファイル」を参照してください。

共通コンポーネント（コマンド・ロガー）のログファイルの出力先を変更する場合

運用ユーザーで次のパラメーターを変更してください。パラメーターに指定する出力先は，運用ユーザーに書き込み権限がある出力先を指定してください。

- ログファイル出力用プロパティファイル（hsdplogger.properties）のhsdpLogfileDir パラメーター

パラメーターの詳細については，「12.13　共通コンポーネント（コマンド・ロガー）のログファイル出力用プロパティファイル（hsdplogger.properties）」を参照してください。

SDP サーバのログファイルの出力先を変更する場合

運用ユーザーで次のパラメーターを変更してください。パラメーターに指定する出力先は，運用ユーザーに書き込み権限がある出力先を指定してください。

- ログファイル出力用プロパティファイル（logger.properties）のlogger.logfileDir パラメーター

各パラメーターの詳細については，次の個所を参照してください。

- 「12.12　SDP サーバのログファイル出力用プロパティファイル（logger.properties）」

SDP サーバの日立 JavaVM ログファイルの出力先を変更する場合

運用ユーザーで次のパラメーターを変更してください。パラメーターに指定する出力先は，運用ユーザーに書き込み権限がある出力先を指定してください。

- SDP サーバ用 JavaVM オプションファイルのSDP_JVM_LOG パラメーター

パラメーターの詳細については，「12.4　SDP サーバ用 JavaVM オプションファイル（jvm_options.cfg）」を参照してください。

外部アダプターのログファイルの出力先を変更する場合

外部アダプターを起動する運用ユーザーで，次のパラメーターを変更してください。パラメーターに指定する出力先は，同じ運用ユーザーに書き込み権限がある出力先を指定してください。

- 外部アダプター定義ファイルのhsdp.logfile.dir パラメーター

パラメーターの詳細については，「12.15　外部アダプター定義ファイル」を参照してください。

SDP 用ログファイルの出力先を変更する例として，ログファイル出力用プロパティファイル（logger.properties）のlogger.logfileDir パラメーターの指定例を次に示します。

変更前

```
# logger.logfileDir=
```

変更後

```
logger.logfileDir=/tmp/hsdp/logs
```

2. ログ収集定義ファイルを作成します。

手順 1.で変更したログファイルの出力先に合わせてログ収集定義ファイルを作成してください。

作成したログ収集定義ファイルは，hsdplogcollect コマンドで保守情報を収集するときに，-file オプションに指定してください。

手順 1.で SDP 用ログファイルの出力先を変更した場合の，ログ収集定義ファイルの記載例を次に示します。

```
/tmp/hsdp/logs/*
/tmp/hsdp/logs/.*.log
```

ログ収集定義ファイルについては，「12.17　ログ収集定義ファイル」を参照してください。

操作結果

なし。

次の作業

なし。

8.10　HSDP のログレベルを上げる

オペレーター，またはサポートエンジニアは，開発フェーズ，構築フェーズで問題が発生したときに，詳細なメッセージがログに出力されるようにログレベルを上げます。これによって，開発フェーズ，構築フェーズで発生する問題に対処しやすくなります。

前提条件

前提条件は次のとおりです。

システム

なし。

ユーザー

なし。

操作手順

1. SDP サーバのログファイル出力用プロパティファイルに，ログレベルを設定します。

   ログに出力するメッセージのレベルを SDP サーバのログファイル出力用プロパティファイルに設定します。

   SDP サーバのログファイル出力用プロパティファイルについては，「12.12　SDP サーバのログファイル出力用プロパティファイル（logger.properties）」を参照してください。

   - コマンドに関する詳細なメッセージをログに出力する場合，次のように指定します。

     ```
     commandMessageLogLevel=2
     ```

   - SDP サーバに関する詳細なメッセージをログに出力する場合，次のように指定します。

     ```
     serverMessageLogLevel=2
     ```

操作結果

なし。

次の作業

なし。

8.11　HSDP で集計・分析したタプルを確認する

HSDP で集計，分析したタプルログを確認するには，次の順に操作を実施します。

- 「タプルログを取得する」
- 「タプルログの内容を表示する」

8.11.1　タプルログを取得する

オペレーターは，タプルログを取得するための設定をします。

前提条件

前提条件は次のとおりです。

システム

- ホストに HSDP が full インストールされていること。
- SDP サーバが停止していること。

ユーザー

運用ディレクトリのアクセス権を持つユーザーでホストにログインしていること。

操作手順

1. システムコンフィグプロパティファイルをテキストエディタで編集し，タプルログを取得する設定に変更します。

   `engine.useTupleLog` プロパティの値を false（デフォルト）から true に変更します。

   ```
   engine.useTupleLog=true
   ```

   システムコンフィグプロパティファイルの詳細については，「12.8　システムコンフィグプロパティファイル（system_config.properties）」を参照してください。

   > **重要**
   >
   > タプルログを取得する設定に変更した場合，ストリームデータ処理の性能が低下します。

操作結果

タプルログを取得する設定に成功した場合，`hsdpcqlstart` コマンドを実行したあとに，*運用ディレクトリ*`/trc/tuplelog` にタプルログファイルが作成されます。

次の作業

なし。

8.11.2　タプルログの内容を表示する

オペレーターは，タプルログを確認するために，タプルログの内容を表示します。

前提条件

前提条件は次のとおりです。

システム

- ホストに HSDP が full インストールされていること。
- SDP サーバが停止していること。

ユーザー

運用ディレクトリのアクセス権を持つユーザーでホストにログインしていること。

操作手順

1. **hsdptplls コマンドを実行して，タプルログの内容を表示します。**
   各ホストの運用ディレクトリで，次のコマンドを実行します。

   ```
   $ 運用ディレクトリ/bin/hsdptplls -file タプルログファイル
   ```

   hsdptplls コマンドの詳細は，「10.21　hsdptplls」を参照してください。

操作結果

- タプルログの内容の表示に成功した場合は，コンソールにタプルログの内容が出力されます。
- タプルログの内容の表示に失敗した場合は，失敗の内容に応じたメッセージがhsdptplls コマンドで標準エラー出力されます。

次の作業

なし。

8.12　HSDP に登録されたクエリにタプルログの内容を再投入する

オペレーターは，タプルログを使用してタプルを再投入し，クエリを再実行します。

前提条件

前提条件は次のとおりです。

システム

- ホストに HSDP が full インストールされていること。
- 再投入用のタプルログファイルがあること。

ユーザー

運用ディレクトリのアクセス権を持つユーザーでホストにログインしていること。

操作手順

1. タイムスタンプモードがサーバモードの場合に，システムコンフィグプロパティファイルをテキストエディタで編集して，タプルの再投入を許可する設定をします。

   stream.tupleLogMode プロパティの値を false（デフォルト）から true に変更します。

   ```
   stream.tupleLogMode=true
   ```

   システムコンフィグプロパティファイルの詳細については，「12.8　システムコンフィグプロパティファイル（system_config.properties）」を参照してください。

2. SDP サーバを開始します。

   各運用ディレクトリで，次のコマンドを実行して SDP サーバを開始します。

   ```
   $ 運用ディレクトリ/bin/hsdpstart
   ```

3. 再実行するクエリグループを登録します。

   各運用ディレクトリで，次のコマンドを実行して再実行するクエリグループを登録します。

   ```
   $ 運用ディレクトリ/bin/hsdpcql クエリグループ名
   ```

4. 再実行するクエリグループを開始します。

   各運用ディレクトリで，次のコマンドを実行して再実行するクエリグループを開始します。

   ```
   $ 運用ディレクトリ/bin/hsdpcqlstart クエリグループ名
   ```

5. タプルログファイルからタプルを再投入します。

   各運用ディレクトリで，次のコマンドを実行して，タプルログファイルからタプルを再投入します。

   ```
   $ 運用ディレクトリ/bin/ hsdptplput -file タプルログファイル
   ```

hsdptplls コマンドの詳細は，「10.20　hsdptplput」を参照してください。

操作結果

なし。

次の作業

なし。

9 トラブルシュート

この章では，HSDP で発生した問題を解決する手順を説明します。ユーザーは，HSDP の導入から運用までの各フェーズで，目的に応じてトラブルシュートを実施します。

9.1　トラブルシュートの作業一覧

ここでは，HSDP の導入から運用までのトラブルシュートの作業一覧を示します。

表 9-1　トラブルシュートの作業一覧

項番	目的	作業
1	問題に対処する	エラーメッセージを確認して対処する
2		トラブルシュート情報の収集対象を追加する
3		トラブルシュート情報を収集する

9.2　問題に対処する

HSDP で発生した問題を解決するには，次の順に操作を実施します。

- 「エラーメッセージを確認して対処する」
- 「トラブルシュート情報の収集対象を追加する」
- 「トラブルシュート情報を収集する」

9.2.1　エラーメッセージを確認して対処する

オペレーター，またはサポートエンジニアは，HSDP で発生した問題を解決するために，エラーメッセージを確認して対処します。

前提条件

前提条件は次のとおりです。

システム

なし。

ユーザー

- SDP サーバのログファイル出力用プロパティファイル（logger.properties）を作成している場合，SDP サーバ用ログファイル，内部アダプター用ログファイル，内部カスタムアダプター用ログファイルの出力先ディレクトリを確認すること。
- 共通コンポーネント（コマンド・ロガー）のログファイル出力用プロパティファイル（hsdplogger.properties）を作成している場合，共通コンポーネント用ログファイルの出力先ディレクトリを確認すること。
- SDP マネージャー定義ファイルを作成している場合，SDP マネージャー用ログファイル，SDP ブローカー用ログファイル，SDP コーディネーター用ログファイルの出力先ディレクトリを確認すること。
- 外部アダプター定義ファイルを作成している場合，外部アダプターのログファイルの格納先ディレクトリを確認すること。

操作手順

1. 各種ログファイルを参照して，エラーメッセージが出力されているかどうかを確認します。
   - 共通コンポーネント用ログファイル

hsdpcommandmessage*N*※1.log

hsdpservermessage*N*※1.log

- 外部アダプターのログファイル

  ExAdaptorMessage*N*※1.log

- SDP ブローカー用ログファイル

  BrokerMessage*N*※1.log

- SDP コーディネーター用ログファイル

  CoordinatorMessage*N*※1.log

- SDP マネージャー用ログファイル

  ManagerMessage*N*※1.log

- SDP サーバ用ログファイル

  SDPServerMessage*N*※1.log

  SDPServerCMessage*N*※1.log

- 内部アダプター用ログファイル

  ADP_*XXX*※2-AdaptorMessage*N*※1.log

  ADP_*XXX*※2-AdaptorCMessage*N*※1.log

- 内部カスタムアダプター用ログファイル

  CADP_*YYY*※3-AdaptorMessage*N*※1.log

  CADP_*YYY*※3-AdaptorCMessage*N*※1.log

- エクスポートログファイル（hsdpexport.log）

  hsdpexport.log

- SDP マネージャーコマンドのログファイル

  hsdpmanagercommandmessage*N*※1.log

注※1

メッセージログファイルの番号

注※2

アダプターグループ名

注※3

SDPClientLogger インタフェースのinitialize()メソッドで指定した任意の名前

重要

HSDP に何らかの問題が発生していても，HSDP のエラーメッセージが出力されていない場合は「9.2.3　トラブルシュート情報を収集する」を参照してトラブルシュート情報を収集してください。また，サポートエンジニアに連絡してください。

2. マニュアル『Hitachi Streaming Data Platform メッセージ』を参照して，出力されているエラーメッセージの対処方法を調査します。
3. 調査した対処方法に従って，問題を解決します。
4. 問題が発生した操作を再実行し，問題が解決されていることを確認します。

操作結果

なし。

次の作業

なし。

9.2.2　トラブルシュート情報の収集対象を追加する

オペレーター，またはサポートエンジニアは，hsdplogcollect コマンドで収集できるトラブルシュート情報の対象にファイルを追加します。

前提条件

前提条件は次のとおりです。

システム

HSDP が full インストールされていること。

ユーザー

なし。

操作手順

1. ログ収集定義ファイルを作成します。
   「12.17　ログ収集定義ファイル」を参照して，ログ収集定義ファイルを作成します。

操作結果

なし。

次の作業

トラブルシュート情報を収集します。

9.2.3　トラブルシュート情報を収集する

オペレーター，またはサポートエンジニアは，トラブルシュート情報を収集します。

前提条件

前提条件は次のとおりです。

システム

ホストに HSDP が full インストールされていること。

ユーザー

- hsdplogcollect コマンドで収集できるトラブルシュート情報にファイルを追加したい場合，あらかじめ「9.2.2　トラブルシュート情報の収集対象を追加する」を実行しておくこと。
- SDP サーバの情報を含むトラブルシュート情報を取得する場合は，運用ディレクトリをセットアップしたユーザー権限で手順を実行すること。
- SDP サーバの情報を含まないトラブルシュート情報を取得する場合は，root ユーザーで手順を実行すること。

操作手順

1. **hsdplogcollect コマンドを実行して，トラブルシュート情報を収集します。**

   SDP サーバの情報を含むトラブルシュート情報を取得する場合

   トラブルシュート情報を収集したい SDP サーバの運用ディレクトリ内のhsdplogcollect コマンドを，運用ディレクトリをセットアップしたユーザーで実行します。

   - ログ収集定義ファイルで収集対象のファイルを追加していない場合，次のコマンドを実行します。

     ```
     # 運用ディレクトリ/bin/hsdplogcollect
     ```

   - 「9.2.2　トラブルシュート情報の収集対象を追加する」を実行してログ収集定義ファイルを作成している場合，-file オプションで指定して，次のコマンドを実行します。

     ```
     # 運用ディレクトリ/bin/hsdplogcollect -file ログ収集定義ファイル
     ```

- 収集したトラブルシュート情報のアーカイブファイルの出力先を変更したい場合は，-dir オプションにログ収集定義ファイルを指定して，次のコマンドを実行します。

```
# 運用ディレクトリ/bin/hsdplogcollect -dir 出力先ディレクトリパス
```

SDP サーバの情報を含まないトラブルシュート情報を取得する場合

トラブルシュート情報に SDP サーバの情報を含める必要がない場合は，*インストールディレクトリ*/bin/のhsdplogcollect コマンドを root 権限で実行します。

- ログ収集定義ファイルで収集対象のファイルを追加していない場合，次のコマンドを実行します。

```
# インストールディレクトリ/bin/hsdplogcollect
```

- 「9.2.2　トラブルシュート情報の収集対象を追加する」を実行してログ収集定義ファイルを作成している場合，-file オプションにログ収集定義ファイルを指定して，次のhsdplogcollect コマンドを実行します。

```
# インストールディレクトリ/bin/hsdplogcollect -file ログ収集定義ファイル
```

- 収集したトラブルシュート情報のアーカイブファイルの出力先を変更したい場合は，-dir オプションを指定して，次のhsdplogcollect コマンドを実行します。

```
# インストールディレクトリ/bin/hsdplogcollect -dir 出力先ディレクトリ
```

操作結果

- トラブルシュート情報の収集に成功した場合，収集ファイルのアーカイブファイルが出力されます。出力されるファイルは次のとおりです。
  - -dir オプションを省略した場合

```
hsdplogcollectコマンドを実行したカレントディレクトリ/hsdplogcollect.tar.gz
```

  - -dir オプションを指定した場合

```
-dirオプションに指定したディレクトリ/hsdplogcollect.tar.gz
```

- トラブルシュート情報の収集に失敗した場合，メッセージが次のログファイルに出力されます。
  - SDP サーバの情報を含むトラブルシュート情報を取得する場合

```
運用ディレクトリ/logs/hsdplogcollect.log
```

  - SDP サーバの情報を含まないトラブルシュート情報を取得する場合

```
hsdplogcollectコマンドを実行したカレントディレクトリ/hsdplogcollect.log
```

次の作業

なし。

10 コマンドリファレンス

この章では，Streaming Data Platform の操作に使用するコマンドについて説明します。

10.1 コマンド一覧

項番	コマンド名	説明
1	hsdpsetup	運用ディレクトリ，SDP マネージャー，SDP ブローカー，および SDP コーディネーターをセットアップします。
2	hsdpunsetup	運用ディレクトリを削除します。
3	hsdpstart	SDP サーバを起動します。
4	hsdpstop	SDP サーバを停止します。
5	hsdpcql	SDP サーバにクエリグループを登録します。
6	hsdpcqldel	SDP サーバのクエリグループを削除します。
7	hsdpcqlstart	SDP サーバのクエリグループを開始します。
8	hsdpcqlstop	SDP サーバのクエリグループを停止します。
9	hsdpstartinpro	インプロセス連携の標準提供アダプター，またはインプロセス連携のカスタムアダプターを起動します。
10	hsdpstopinpro	インプロセス連携の標準提供アダプター，またはインプロセス連携のカスタムアダプターを停止します。
11	hsdpversion	HSDP のバージョンを表示します。
12	hsdpexport	運用ディレクトリや，SDP マネージャー，SDP ブローカー，および SDP コーディネーターのセットアップ構成のファイルを収集し，アーカイブファイルを作成します。
13	hsdplogcollect	HSDP の障害情報を収集して保存します。
14	hsdpmanager	SDP マネージャー，SDP ブローカー，および SDP コーディネーターを，開始したり停止したりします。
15	hsdpstatusshow	SDP マネージャー，SDP ブローカー，SDP コーディネーター，SDP サーバ，クエリグループ，アダプターグループ，およびカスタムアダプターの稼働情報を表示します。
16	hsdptplput	タプルログファイルに取得したタプルを入力ストリームキューに再投入します。
17	hsdptplls	タプルログファイルに取得したタプルの情報を表示します。

10.2　排他の関係

コマンド間の排他の関係を次の表に示します。

表 10-1　コマンド間の排他の関係

実行するコマンド	hsdpsetup	hsdpunsetup	hsdpstart	hsdpstop	hsdpstop-force	hsdpcql	hsdpcqldel	hsdpcqlstart	hsdpcqlstop	hsdpstartinpro	hsdpstopinpro	hsdplogcollect	hsdpversion	hsdpexport	hsdpmanager	hsdpstatusshow	hsdptplput	hsdptplls
hsdpsetup	×	×	×	×	×	×	×	×	×	×	×	×	○	×	○	×	×	×
hsdpunsetup	×	×	×	×	×	×	×	×	×	×	×	×	○	×	○	×	×	×
hsdpstart	×	×	×	×	×	×	×	×	×	×	×	○	○	○	×	○	×	○
hsdpstop	×	×	×	×	×	×	×	×	×	×	×	○	○	○	×	○	×	○
hsdpstop -force	×	×	○	○	×	○	○	○	○	○	○	○	○	○	×	○	×	○
hsdpcql	×	×	×	×	×	×	×	×	×	×	×	○	○	○	×	○	×	○
hsdpcqldel	×	×	×	×	×	×	×	×	×	×	×	○	○	○	×	○	×	○
hsdpcqlstart	×	×	×	×	×	×	×	×	×	×	×	○	○	○	×	○	×	○
hsdpcqlstop	×	×	×	×	×	×	×	×	×	×	×	○	○	○	×	○	×	○
hsdpstartinpro	×	×	×	×	×	×	×	×	×	×	×	○	○	○	×	○	×	○
hsdpstopinpro	×	×	×	×	×	×	×	×	×	×	×	○	○	○	×	○	×	○
hsdplogcollect	×	×	○	○	○	○	○	○	○	○	○	×	○	○	×	○	○	○
hsdpversion	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
hsdpexport	×	×	○	○	○	○	○	○	○	○	○	○	○	×	×	○	○	○
hsdpmanager	○	○	×	×	×	×	×	×	×	×	×	×	○	×	×	×	×	○
hsdpstatusshow	×	×	○	○	○	○	○	○	○	○	○	○	○	○	×	○	○	○
hsdptplput	×	×	×	×	×	×	×	×	×	×	×	○	○	○	×	○	×	○
hsdptplls	×	×	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○

（凡例）

- ○：実行できる
- ×：実行できない

10.3　コマンドを実行する上での注意事項

ここでは，コマンドを実行する上での注意事項について説明します。

実行権限を持つユーザーについて

運用ディレクトリ/bin/に格納される次のコマンドは，運用ディレクトリをセットアップしたユーザー以外にも実行権限が付与されます。コマンドを実行できるユーザーを限定したい場合は，運用ディレクトリへのアクセス権限で設定してください。

- hsdpstart
- hsdpstop
- hsdpcql
- hsdpcqldel
- hsdpcqlstart
- hsdpcqlstop
- hsdpstartinpro
- hsdpstopinpro
- hsdpexport
- hsdplogcollect
- hsdpstatusshow
- hsdptplput
- hsdptplls

また，*運用ディレクトリ*/bin/および*インストールディレクトリ*/bin/に格納されているコマンドは，各コマンドの説明で記載している実行権限を持ったユーザーが実行してください。

環境変数 PATH について

運用ディレクトリ以下のパスおよび*インストールディレクトリ*以下のパスを環境変数PATH に指定してコマンドを実行しないでください。

コマンド実行時のリトライについて

次の場合，30 秒間コマンドをリトライします。リトライしても実行できなかった場合，メッセージ KFHD90031-E が出力され，エラー終了します。また，コマンドのリトライ中に，ほかのコマンドを追加で実行した場合，実行順序は保証されません。

- 外部アダプターの起動による自動生成アダプターの起動中に，コマンドを実行した場合
- 任意のコマンドを実行中に，同時実行できないコマンドを実行した場合

10.4　コマンドリファレンスの説明項目

この章で説明する項目は次のとおりです。ただし，コマンドによっては説明しない項目もあります。

形式

コマンドの形式を説明しています。

説明

コマンドの機能を説明しています。

実行権限を持つユーザー

コマンドの実行に必要なユーザーの条件について説明しています。

コマンド実行の前提条件

コマンドの実行に必要な前提条件について説明しています。

格納先ディレクトリ

コマンドの格納先ディレクトリについて説明しています。

引数

コマンドの引数について説明しています。

注意事項

コマンドを実行する上での注意事項について説明しています。

戻り値

コマンドの戻り値を説明しています。

出力形式

コマンドの出力形式について説明しています。

10.5 hsdpsetup

形式

```
hsdpsetup {
-mgr [-host ホスト名またはIPアドレス] [-port ポート番号]
    [-chosts ホスト名またはIPアドレス-1[, ホスト名またはIPアドレス-2] ...]
    [-cmulti データの多重度]
    [-cmaddr SDPコーディネーターのマルチキャストアドレス]
|-dir 運用ディレクトリ [-sc サーバクラスタ名]

|-help
}
```

説明

hsdpsetup コマンドで，次のことを実施します。

- SDP マネージャー，SDP ブローカー，および SDP コーディネーターのセットアップ
- 運用ディレクトリの作成およびセットアップ

SDP マネージャー，SDP ブローカー，および SDP コーディネーターのセットアップでは，Streaming Data Platform が使用するホストやポート番号などを設定します。すでにセットアップが完了している場合にこのコマンドを実行すると，すべての設定内容が上書きされます。上書きは，指定を省略した引数に対しても実施されます。

運用ディレクトリの作成およびセットアップでは，SDP サーバを動作させるための運用ディレクトリを，任意の場所に作成します。すでに作成されている運用ディレクトリを指定してこのコマンドを実行すると，すべての設定内容が上書きされます。上書きは，指定を省略した引数に対しても実施されます。ただし，既存の運用ディレクトリ下に，セットアップ時に配布されるファイルと同名のファイルが存在していた場合，そのファイルは上書きされません。

メモ

ほかのユーザーが作成した運用ディレクトリを指定してhsdpsetup コマンドを実行した場合は，エラーになります。

セットアップ後の運用ディレクトリのディレクトリ構造の詳細については，『Hitachi Streaming Data Platform リリースノート』を参照してください。

正しい引数を指定してコマンドを実行した場合，セットアップを続行するかどうかを確認するメッセージが表示されます。y を入力すると，コマンドはセットアップを続行します。n を入力すると，セットアップをキャンセルします。

セットアップが正常に終了すると，セットアップ時に指定した引数の内容が次のログファイルに出力されます。

- SDP マネージャー，SDP ブローカー，および SDP コーディネーターをセットアップした場合

```
/var/log/hitachi/hsdp/hsdpsetupmanager.log
```

- 運用ディレクトリを作成およびセットアップした場合

```
運用ディレクトリ/logs/hsdpsetup.log
```

各ログファイルのフォーマットの詳細については，「13.6　運用ディレクトリのセットアップログファイル」または「13.7　SDP マネージャーセットアップログファイル（hsdpsetupmanager.log)」を参照してください。

実行権限を持つユーザー

- SDP マネージャー，SDP ブローカー，および SDP コーディネーターのセットアップ

  root 権限を持っているユーザー
- 運用ディレクトリのセットアップ
  - 運用ディレクトリを新規作成する場合

    すべてのユーザー
  - 作成済みの運用ディレクトリを再セットアップする場合

    対象の運用ディレクトリを作成したユーザー

コマンド実行の前提条件

- SDP マネージャー，SDP ブローカー，および SDP コーディネーターをセットアップする場合
  - HSDP が full インストールされていること
  - SDP マネージャー，SDP ブローカー，SDP コーディネーター，およびすべての SDP サーバが停止していること
- 運用ディレクトリを作成およびセットアップする場合
  - HSDP が full インストールされていること
  - 作成済の運用ディレクトリを再セットアップする場合，セットアップ先の運用ディレクトリで SDP サーバが起動していないこと

格納先ディレクトリ

```
インストールディレクトリ/bin/
```

引数

-dir *運用ディレクトリ*

作成する運用ディレクトリのパスを指定します。

メモ

- 運用ディレクトリは，128 バイト以下の絶対パスで指定してください。また，絶対パスは，英数字，ピリオド（.），アンダーライン（_），コロン（:），スラッシュ（/）のどれかの文字で構成されている必要があります。
- 運用ディレクトリには，*インストールディレクトリ*下のディレクトリは指定できません。

-sc *サーバクラスタ名*

運用ディレクトリで動作する SDP サーバが所属するサーバクラスタ名を指定します。

*サーバクラスタ名*には，英数字またはアンダーライン（_）で構成された 1 文字～64 文字の文字列を指定してください。先頭に使用できる文字は半角の英字だけです。英字の大文字と小文字は区別されます。

このオプションの指定を省略した場合，サーバクラスタ名にはMy_server_cluster が設定されます。

-mgr

SDP マネージャー，SDP ブローカー，および SDP コーディネーターをセットアップする場合に指定します。このオプションを指定してコマンドを実行するには，root 権限が必要です。

-host *ホスト名またはIPアドレス*

自ホストのホスト名または IP アドレスを指定します。

指定したホスト名または IP アドレスは，外部アダプターまたは内部カスケーディングアダプターが，hsdpsetup コマンドを実行したホスト上のストリームに接続する際に使用されます。

コーディネーターグループに属するほかのホストの SDP ブローカーに接続した外部アダプターまたは内部カスケーディングアダプターが，-host オプションに指定したホスト名を名前解決できなかった場合，hsdpsetup を実行したホストのストリームへの接続に失敗します。

localhost を指定した場合は，NIC を使用せず，同じホスト内でだけ通信します。ほかのホストの SDP サーバとの通信はできません。

ホスト名を指定する場合は，英数字，ハイフン（-），ピリオド（.）で構成された 1 文字～63 文字の文字列で指定してください。IP アドレスを指定する場合は，数字，ピリオド（.）で構成された IPv4 形式の文字列で指定してください。

ホスト名を指定する場合は，hsdpsetup コマンドを実行したホストで名前解決できる必要はありません。

このオプションの指定を省略した場合，localhost となります。

-port *ポート番号*

HSDP が使用するポート群の先頭のポート番号を指定します。HSDP では，指定したポート番号から始まる 8 個のポートを連番で使用します。

-chosts オプションを使用して SDP コーディネーターをほかのホストと連携させる場合は，すべてのホストで同じ値を指定してください。

それぞれのポートは，次の用途で使われます。表中の N は，-port オプションに指定したポート番号を表します。

項番	ポート番号	用途
1	*N*	SDP ブローカーが外部アダプターや内部アダプターからの要求を受け付けるための，待ち受けをするポートとして使用します。
2	*N*+1	HSDP の内部通信に使用します。
3	*N*+2	SDP コーディネーター間の通信（TCP/IP 通信）に使用します。
4	*N*+3	
5	*N*+4	
6	*N*+5	
7	*N*+6	
8	*N*+7	SDP コーディネーターが互いの生存を監視するための通信（UDP/IP 通信）に使用します。

ポート番号は，1024～60000 の範囲で指定してください。

このオプションの指定を省略した場合，20425 が設定され，SDP ブローカーは20425 番のポートで要求の待ち受けをします。20426 番のポートは HSDP の内部通信に使用され，20427 番～20432 番のポートは SDP コーディネーター間の通信に使用されます。

-chosts *ホスト名またはIPアドレス-1*[，*ホスト名またはIPアドレス-2*]...

SDP コーディネーターがコーディネーターグループに所属するかどうか，または，自ホストに SDP コーディネーターを起動させないでほかのホストの SDP コーディネーターを使用するかどうかを指定します。

- 自ホストの SDP コーディネーターをコーディネーターグループに所属させる場合

  コーディネーターグループを形成させたい SDP コーディネーターが存在するホストのホスト名，または IP アドレスを指定します。自ホストのホスト名または IP アドレスを含め，コーディネーターグループを形成させたいすべてのホストのホスト名または IP アドレスを指定する必要があります。それぞれのホストの情報をコンマでつないで指定します。コンマの前後にはスペースを入れないでください。

  ホスト名を指定する場合は，英数字，ハイフン（-），ピリオド（.）で構成された 1 文字～63 文字の文字列で指定してください。IP アドレスを指定する場合は，数字，ピリオド（.）で構成された IPv4 形式の文字列で指定してください。ホストの情報は，96 個まで指定できます。

  > **メモ**
  >
  > 同じコーディネーターグループに所属するホスト間で，同じ内容を指定する必要があります。
  >
  > また，2 台以上の SDP コーディネーターでコーディネーターグループを形成する場合，ホスト名としてlocalhost を指定する，または IP アドレスとしてループバックアドレスを指定するとエラーになります。

- 自ホストの SDP コーディネーターだけで動作させる場合

localhost，またはNICに設定されている自ホストのホスト名またはIPアドレスを指定してください。

- 自ホストのSDPコーディネーターは起動しないで，ほかのホストのSDPコーディネーターを使用する場合

  SDPコーディネーターが起動している任意のホストのホスト名またはIPアドレスを指定してください。指定値に自ホストのホスト名またはIPアドレスが存在しなければ，自ホストのSDPコーディネーターは起動しません。指定したホストのSDPコーディネーターがコーディネーターグループに所属している場合は，そのコーディネーターグループに所属するすべてのホストの，ホスト名またはIPアドレスを指定してください。

このオプションの指定を省略した場合，localhostが設定されます。-hostオプションとともに指定を省略した場合は，SDPサーバはどのコーディネーターグループにも所属しないで，自ホストのSDPコーディネーターだけで動作します。

-cmulti *データの多重度*

コーディネーターグループで保持するデータの多重度を指定します。データの多重度は，コーディネーターグループ内で設定を一致させる必要があります。

データの多重度には，1～5の値を指定してください。

データの多重度を1以外にする場合，-chostsには(*データの多重度*×2)-1以上の台数のホストを指定する必要があります。

このオプションの指定を省略した場合，1が設定されます。

-cmaddr *SDPコーディネーターのマルチキャストアドレス*

コーディネーターグループに所属するSDPコーディネーターが互いの生存を監視するためのマルチキャストアドレスを指定します。コーディネーターグループ内で設定を一致させる必要があります。

SDPコーディネーターのマルチキャストアドレスは，数字，ピリオド（.）で構成されたIPv4の形式の文字列で指定してください。

このオプションの指定を省略した場合，239.255.2.1が設定されます。

> **メモ**
>
> 同一LAN内でコーディネーターグループを複数動作させる場合は，アドレスが競合しないように設定してください。同一LAN内でマルチキャストアドレスの設定が競合している場合，SDPコーディネーターの動作は保証されません。

-help

コマンドの使用方法を表示します。このオプションを指定すると，コマンドはセットアップを実施しないで終了します。

注意事項

特になし。

戻り値

値	説明
0	セットアップが正常に終了しました。
1	セットアップの実行中にエラーが発生しました。
2	コマンドを実行したユーザーによって，セットアップがキャンセルされました。

10.6 hsdpunsetup

形式

```
hsdpunsetup {-dir 運用ディレクトリ | -help}
```

説明

hsdpunsetup コマンドで，運用ディレクトリを削除します。コマンドを実行すると，運用ディレクトリ下にある，すべてのサブディレクトリおよびファイルも削除されます。ただし，ほかのユーザーがセットアップした運用ディレクトリは削除できません。

このコマンドを実行すると，運用ディレクトリの削除を続行するかどうかを確認するメッセージが表示されます。y を入力すると，コマンドは削除を続行します。n を入力すると，削除がキャンセルされます。

実行権限を持つユーザー

root ユーザーまたは運用ディレクトリをセットアップしたユーザー

コマンド実行の前提条件

削除する運用ディレクトリの SDP サーバが停止していること

格納先ディレクトリ

```
インストールディレクトリ/bin/
```

引数

-dir 運用ディレクトリ

削除する運用ディレクトリを指定します。指定されたディレクトリが存在しない場合，または運用ディレクトリではない場合，コマンドはエラーを返します。

-help

コマンドの使用方法を表示します。このオプションを指定すると，コマンドは運用ディレクトリの削除を実施しないで終了します。

注意事項

運用ディレクトリを削除すると，指定した運用ディレクトリに格納されているすべてのディレクトリおよびファイルが削除されます。バックアップが必要なディレクトリまたはファイルは，削除する前に退避してください。

戻り値

値	説明
0	運用ディレクトリの削除が正常に終了しました。
1	運用ディレクトリの削除中にエラーが発生しました。
2	コマンドを実行したユーザーによって，運用ディレクトリの削除がキャンセルされました。

10.7 hsdpstart

形式

```
hsdpstart {[-init] | -help}
```

説明

hsdpstart コマンドで，HSDP サーバを起動します。

HSDP サーバの起動には，次の種類があります。

- 通常起動
  運用ディレクトリ内の SDP サーバを起動します。
- 再起動
  異常終了した SDP サーバを起動し，異常終了時に動作していたクエリグループおよびアダプターを再登録します。

hsdpstart コマンドを引数なしで実行すると，運用ディレクトリ内の異常終了した SDP サーバが再起動されます。運用ディレクトリ内に異常終了した HSDP サーバがない場合は，HSDP サーバが通常起動されます。 すでに SDP サーバが起動している状態でこのコマンドを実行すると，起動しないで，正常に終了します。

実行権限を持つユーザー

運用ディレクトリをセットアップしたユーザー

コマンド実行の前提条件

特になし。

格納先ディレクトリ

運用ディレクトリ/bin/

引数

-init
: SDP サーバを通常起動します。運用ディレクトリ内の異常終了した SDP サーバを再起動したくない場合に，このオプションを指定してください。

-help
: コマンドの使用方法を表示します。このオプションを指定すると，コマンドは HSDP サーバを起動しないで終了します。

注意事項

SDP サーバの起動後に，運用ディレクトリ下に格納されている次のファイルを変更しないでください。

- system_config.properties
- jvm_options.cfg
- jvm_cengine_options.cfg
- クエリグループ用プロパティファイル
- ストリーム用プロパティファイル
- クエリ定義ファイル
- アダプター構成定義ファイル

戻り値

値	説明
0	HSDP サーバの起動が正常に終了しました。
1	HSDP サーバの起動中にエラーが発生しました。

10.8 hsdpstop

形式

```
hsdpstop {[-force] | [-help]}
```

説明

hsdpstop コマンドで，運用ディレクトリで動作する SDP サーバを停止します。SDP サーバを停止すると，停止前に SDP サーバへ登録されていたクエリグループは削除されます。

実行権限を持つユーザー

運用ディレクトリをセットアップしたユーザー

コマンド実行の前提条件

SDP サーバが稼働状態であること

格納先ディレクトリ

運用ディレクトリ/bin/

引数

-force

SDP サーバを強制停止します。このオプションを指定すると，ストリームエンジンおよび内部アダプターに滞留している分析中のタプルはすべて破棄されます。

ほかのコマンドを実行中でも，SDP サーバを強制停止します。

-help

コマンドの使用方法を表示します。このオプションを指定すると，コマンドは SDP サーバを停止しないで終了します。

注意事項

-force オプションを省略してhsdpstop コマンドを実行すると，ストリームエンジンおよび内部アダプターがすべてのタプルを処理するまで，戻り値が返却されません。そのため，SDP サーバも停止しません。

戻り値

値	説明
0	SDP サーバの停止が正常に終了しました。
1	SDP サーバの停止中にエラーが発生しました。

10.9 hsdpcql

形式

```
hsdpcql {[-start] [-thread 並列実行数]
クエリグループ名 | -help}
```

説明

hsdpcql コマンドで，指定したクエリグループを SDP サーバに登録します。

このコマンドを実行した際には，登録対象のクエリグループに対応するクエリグループ用プロパティファイルに指定したプロパティが反映されます。また，クエリグループ用プロパティファイルで指定したクエリ定義ファイル内のすべてのクエリの構文が解析されます。ファイル内の構文のエラーを検出すると，ログファイルとコンソールに構文のエラーメッセージが出力されます。

メモ

すでに登録されているクエリグループと同名のクエリグループを登録しようとした場合，エラーになります。

実行権限を持つユーザー

運用ディレクトリをセットアップしたユーザー

コマンド実行の前提条件

次のファイルが必要です。

- クエリグループ用プロパティファイル
- クエリ定義ファイル

格納先ディレクトリ

```
運用ディレクトリ/bin/
```

引数

-start

このオプションを指定すると，クエリグループの登録後，クエリグループが開始状態になり，入力タプルの受付が開始されます。

このオプションの指定を省略した場合，クエリグループの登録だけが実施され，開始は実施されません。

-thread *並列実行数*

スケールアップ構成の場合に，クエリグループに割り当てるスレッドの並列実行数を指定します。*並列実行数*には1 から999 の値を指定してください。

このオプションの指定を省略した場合，スケールアップ構成でのクエリグループの並列化は実施されません。

クエリグループ名

SDP サーバに登録するクエリグループの名前を指定します。

-help

コマンドの使用方法を表示します。このオプションを指定すると，コマンドは SDP サーバへのクエリグループの登録を実施しないで終了します。

注意事項

特になし。

戻り値

値	説明
0	クエリグループの登録が正常に終了しました。
1	クエリグループの登録中，またはクエリグループの起動中にエラーが発生しました。

10.10 hsdpcqldel

形式

```
hsdpcqldel {クエリグループ名 | -help}
```

説明

hsdpcqldel コマンドで，SDP サーバに登録されている，停止状態または閉塞状態にあるクエリグループを削除します。

実行権限を持つユーザー

運用ディレクトリをセットアップしたユーザー

コマンド実行の前提条件

削除対象のクエリグループが停止状態または閉塞状態であること

格納先ディレクトリ

運用ディレクトリ/bin/

引数

クエリグループ名

SDP サーバから削除するクエリグループの名前を指定します。

-help

コマンドの使用方法を表示します。このオプションを指定すると，コマンドは SDP サーバからのクエリグループの削除を実施しないで終了します。

注意事項

特になし。

戻り値

値	説明
0	クエリグループの削除が正常に終了しました。
1	クエリグループの削除中にエラーが発生しました。

10.11 hsdpcqlstart

形式

```
hsdpcqlstart {クエリグループ名 | -help}
```

説明

hsdpcqlstart コマンドで，SDP サーバに登録された，停止状態または閉塞状態のクエリグループを開始します。

実行権限を持つユーザー

運用ディレクトリをセットアップしたユーザー

コマンド実行の前提条件

開始対象のクエリグループが停止状態または閉塞状態であること

格納先ディレクトリ

運用ディレクトリ/bin/

引数

クエリグループ名

開始するクエリグループの名前を指定します。

-help

コマンドの使用方法を表示します。このオプションを指定すると，コマンドはクエリグループの開始を実施しないで終了します。

注意事項

特になし。

戻り値

値	説明
0	クエリグループの開始が正常に終了しました。
1	クエリグループの開始中にエラーが発生しました。

10.12 hsdpcqlstop

形式

```
hsdpcqlstop { クエリグループ名 | -help}
```

説明

hsdpcqlstop コマンドで，HSDP サーバで開始状態になっているクエリグループを停止状態にします。

実行権限を持つユーザー

運用ディレクトリをセットアップしたユーザー

コマンド実行の前提条件

対象のクエリグループが開始状態であること

格納先ディレクトリ

運用ディレクトリ/bin/

引数

クエリグループ名

停止するクエリグループの名前を指定します。

-help

コマンドの使用方法を表示します。このオプションを指定すると，コマンドはクエリグループの停止を実施しないで終了します。

注意事項

特になし。

戻り値

値	説明
0	クエリグループの停止が正常に終了しました。
1	クエリグループの停止中にエラーが発生しました。

10.13 hsdpstartinpro

形式

インプロセス連携の標準提供アダプターを起動する場合

```
hsdpstartinpro {[-file アダプター構成定義ファイル][-qg クエリグループ名 -st ストリーム名 ] アダプターグループ名 | -help}
```

インプロセス連携のカスタムアダプターを起動する場合

```
hsdpstartinpro {インプロセス連携のカスタムアダプター名 [ カスタムアダプターに渡す引数]... | -help}
```

説明

hsdpstartinpro コマンドで，指定したインプロセス連携の標準提供アダプター，またはインプロセス連携のカスタムアダプターを起動します。

インプロセス連携のカスタムアダプターを指定してこのコマンドを実行すると，インプロセス連携用プロパティファイルに指定されたメインクラスをロードして HSDP サーバに登録したあと，StreamInprocessUP インタフェース実装クラスのexecute メソッドを実行します。

実行権限を持つユーザー

運用ディレクトリをセットアップしたユーザー

コマンド実行の前提条件

HSDP サーバが稼働状態であること

格納先ディレクトリ

運用ディレクトリ/bin/

引数

-file *アダプター構成定義ファイル*

アダプターの構成ファイルのパスを相対パスまたは絶対パスで指定します。ファイルに指定されたアダプター構成で，標準提供アダプターのアダプターグループが起動します。

このオプションの指定を省略した場合，デフォルトファイルに定義されたアダプターの構成で，標準提供アダプターのアダプターグループが起動します。

相対パスを使用する場合に基準となる場所は，次のディレクトリです。

運用ディレクトリ/conf/xml/

-qg *クエリグループ名*

アダプターが接続するクエリググループ名を指定します。アダプター構成定義ファイルがスキーマを自動解決する書式になっている場合にだけ，このオプションを指定する必要があります。それ以外の場合は，指定を省略してください。

アダプター構成定義ファイルがスキーマを自動解決する書式になっている場合，このオプションで指定したクエリグループ内にある，-st オプションで指定したストリームからスキーマ情報を取得してアダプターを動作させます。

-st *ストリーム名*

アダプターが接続するストリーム名を指定します。アダプター構成定義ファイルがスキーマを自動解決する書式になっている場合にだけ，このオプションを指定する必要があります。それ以外の場合は，指定を省略してください。

アダプター構成定義ファイルがスキーマを自動解決する書式になっている場合，-qg オプションで指定したクエリグループ内にある，このオプションで指定したストリームからスキーマ情報を取得してアダプターを動作させます。

アダプターグループ名

起動するアダプターグループの名前を指定します。

アダプター構成定義ファイル（-file オプションで指定されたファイル）にあるプロセスグループ定義（InprocessGroupDefinition タグのname 属性）で指定された，アダプターグループ名を指定してください。

インプロセス連携のカスタムアダプター名

起動するインプロセス連携のカスタムアダプターの名前を指定します。次のファイルで指定します。

user_app.*インプロセス連携のカスタムアダプター名*.properties

カスタムアダプターに渡す引数

起動するインプロセス連携のカスタムアダプターに渡す引数を指定します。指定した引数は，インプロセス連携のカスタムアダプターで，String 型の可変長引数を持つコンストラクタに渡されます。引数は，半角スペース区切りで複数指定できます。

-help

コマンドの使用方法を表示します。このオプションを指定すると，コマンドはアダプターを起動しないで終了します。

注意事項

- 引数の指定を省略したり，誤った引数を指定したりした場合は，コマンドの使い方が表示され，コマンドが終了します。インプロセス連携の標準提供アダプターを起動している場合は，仮定されたアダプターグループ名InprocessApName がメッセージ中に表示されます。
- hsdpstartinpro コマンドは，HSDP サーバが起動しているときだけ実行できます。
- 同時に実行できる最大のアダプターグループ数は 128 です。カスタムアダプター，および HSDP が自動生成する内部アダプターも，アダプターグループの 1 つにカウントされます。

- アダプター構成定義ファイルがスキーマを自動解決する書式になっている場合に，-st オプションおよび-qg オプションを省略すると，アダプター構成定義ファイルの形式が不完全になるおそれがあります。この場合，ログファイルにメッセージKFSP56101-E が出力されます。
- スケールアップ構成で標準提供アダプターを起動する場合は，hsdpstatusshow コマンドでアダプターが接続するクエリグループを確認した上で，アダプター構成定義ファイルにクエリグループ名を指定してください。

戻り値

値	説明
0	アダプターの起動が正常に終了しました。
1	アダプターの起動中にエラーが発生しました。

10.14 hsdpstopinpro

形式

インプロセス連携の標準提供アダプターを停止する場合

```
hsdpstopinpro {アダプターグループ名 | -help}
```

インプロセス連携のカスタムアダプターを停止する場合

```
hsdpstopinpro {インプロセス連携のカスタムアダプター名 | -help}
```

説明

hsdpstopinpro コマンドで，インプロセス連携の標準提供アダプターまたはカスタムアダプターを停止します。

対象がインプロセス連携のカスタムアダプターの場合，HSDP サーバに登録されているインプロセス連携のカスタムアダプターが実装しているStreamInprocessUP インタフェースのstop メソッドを実行します。

hsdpstopinpro コマンドは，hsdpstartinpro コマンドで起動したアダプターに対して実行できます。

実行権限を持つユーザー

運用ディレクトリをセットアップしたユーザー

コマンド実行の前提条件

内部アダプターがすでに起動していること

格納先ディレクトリ

```
運用ディレクトリ/bin/
```

引数

アダプターグループ名

停止するアダプターグループの名前を指定します。アダプター構成定義ファイルでインプロセスグループ定義(<InprocessGroupDefinition>タグのname 属性)に指定されているアダプターグループ名を指定してください。

インプロセス連携のカスタムアダプター名

停止するインプロセス連携のカスタムアダプターの名前を指定します。

-help

コマンドの使用方法を表示します。このオプションを指定すると，コマンドはアダプターを停止しないで終了します。

注意事項

特になし。

戻り値

値	説明
0	アダプターの停止が正常に終了しました。
1	アダプターの停止中にエラーが発生しました。

10.15 hsdpversion

形式

```
hsdpversion [-help]
```

説明

hsdpversion コマンドで，現在インストールされている HSDP のバージョンを表示します。バージョンは，*V1V2.R1.R2-MM* の形式で表示されます。

V1V2：バージョン番号 （*V1* が0 の場合，*V1* は省略されます）

R1：リリース番号

R2：リビジョン番号

MM：保守レベル（*MM* は省略されません）

実行権限を持つユーザー

すべてのユーザー

コマンド実行の前提条件

特になし。

格納先ディレクトリ

インストールディレクトリ/bin/

引数

-help

コマンドの使用方法を表示します。このオプションを指定すると，コマンドは HSDP のバージョンを表示しないで終了します。

注意事項

特になし。

戻り値

値	説明
0	コマンドが正常終了しました。

値	説明
1	コマンドの処理中にエラーが発生しました。

10.16 hsdpexport

形式

```
hsdpexport {{-all | {-file エクスポート定義ファイル|-dir|-qg} [-mgr]|-exadp} |-help}
```

説明

hsdpexport コマンドで，各運用ディレクトリ内の定義ファイルや製品の設定ファイルなどを収集して，それらをまとめたアーカイブファイルを作成します。アーカイブファイルは，*運用ディレクトリ*/export/下に作成されます。

収集するファイルは，コマンドオプションで指定します。

このコマンドが作成しようとするアーカイブファイルと同名のファイルが*運用ディレクトリ*/export/下にすでに格納されている場合，ファイルは自動で上書きされます。

運用ディレクトリ以下にあるファイルをアーカイブする場合，運用ディレクトリからの相対パスで，ディレクトリ構造を維持したままアーカイブします。

運用ディレクトリ以外のディレクトリにデプロイされているファイルをアーカイブする場合，ルートディレクトリから始まる絶対パスで，ディレクトリ構造を維持したままアーカイブします。

実行権限を持つユーザー

運用ディレクトリをセットアップしたユーザー

コマンド実行の前提条件

特になし。

格納先ディレクトリ

運用ディレクトリ/bin/

引数

-all

運用ディレクトリと，SDP マネージャー，SDP ブローカー，および SDP コーディネーターに関連するファイルをアーカイブファイルに収集します。作成されるアーカイブファイルは，hsdpexport_all.tar.gz という名称になります。

このオプションを指定してhsdpexport コマンドを実行する際は，hsdpmanager -stop を実行して，SDP マネージャーを停止しておく必要があります。SDP マネージャーが停止していない状態でhsdpexport コマンドを実行した場合，アーカイブファイルの内容は保証されません。

運用ディレクトリにユーザーが任意に作成しているディレクトリも収集対象になります。収集対象になるファイルおよびディレクトリを次に示します。

- *運用ディレクトリ*/inadaptor/*
- *運用ディレクトリ*/bin/*
- *運用ディレクトリ*/conf/*
- *運用ディレクトリ*/exadaptor/*
- *運用ディレクトリ*/lib/*
- *運用ディレクトリ*/query/*
- *運用ディレクトリ*/spool/*
- *運用ディレクトリ*/*ユーザーディレクトリ*/*
- /opt/hitachi/hsdp/conf/*
- /var/log/hitachi/hsdp/hsdpsetupmanager.log

このオプションを指定して作成したアーカイブファイルを展開するには，root 権限が必要です。アーカイブファイルを展開後，次のコマンドを実行して，運用ディレクトリの所有者，グループを root から運用ディレクトリを使用するユーザーとそのグループに書き換えてください。

chown -R *ユーザー名*:*グループ名* *運用ディレクトリ*

-file *エクスポート定義ファイル*

エクスポート定義ファイルに定義したファイルをアーカイブファイルに収集します。このオプションを指定した場合，作成されるアーカイブファイルの名称は，hsdpexport_file.tar.gz になります。

また，-mgr オプションも指定していた場合はhsdpexport_file_mgr.tar.gz になります。

エクスポート定義ファイルは，絶対パス，またはコマンドを実行するカレントディレクトリからの相対パスで指定してください。

エクスポート定義ファイルに指定されているファイルが存在しない場合，そのファイルのパスをコンソールと*運用ディレクトリ*/logs/hsdpexport.log に出力します。

*インストールディレクトリ*下のファイルを定義ファイルに指定した場合，このオプションを指定して作成したアーカイブファイルを展開するには，root 権限が必要です。アーカイブファイルを展開後，次のコマンドを実行して，運用ディレクトリの所有者とグループを root から運用ディレクトリを使用するユーザーとそのグループに書き換えてください。

chown -R *ユーザー名*:*グループ名* *運用ディレクトリ*

-dir

運用ディレクトリ以下のファイルをアーカイブファイルに収集します。このオプションを指定した場合，作成されるアーカイブファイルの名称は，hsdpexport_dir.tar.gz になります。

また，-mgr オプションも指定していた場合はhsdpexport_dir_mgr.tar.gz になります。

運用ディレクトリにユーザーが任意に作成しているディレクトリも収集対象になります。収集対象になるファイルおよびディレクトリを次に示します。

- *運用ディレクトリ*/inadaptor/*
- *運用ディレクトリ*/bin/*
- *運用ディレクトリ*/conf/*
- *運用ディレクトリ*/exadaptor/*
- *運用ディレクトリ*/lib/*
- *運用ディレクトリ*/query/*
- *運用ディレクトリ*/spool/*
- *運用ディレクトリ*/*ユーザーディレクトリ*/*

-qg

運用ディレクトリに格納されているクエリグループごとに，クエリグループの動作に必要となるファイルをアーカイブします。作成されるアーカイブファイルの名称は，*クエリグループ名*_qg.tar.gz になります。

また，-mgr オプションを指定していた場合は*クエリグループ名*_qg_mgr.tar.gz になります。

クエリグループが複数存在する場合，クエリグループの数だけアーカイブファイルが作成されます。それぞれのアーカイブファイルには，クエリグループ用プロパティファイルとクエリ定義ファイルが 1 つずつと，そのほかの関連するファイルが格納されます。

クエリグループ用プロパティファイルのquerygroup.cqlFilePath に運用ディレクトリ外のパスが指定されている場合，メッセージを出力して処理を終了します。

収集対象になるファイルおよびディレクトリを次に示します。

- *運用ディレクトリ*/inadaptor/*
- *運用ディレクトリ*/bin/*
- *運用ディレクトリ*/conf/*
- *運用ディレクトリ*/conf/*クエリグループプロパティファイル*
- *運用ディレクトリ*/lib/*
- *運用ディレクトリ*/spool/*
- *任意のディレクトリ*/*クエリ定義ファイル*

-mgr

SDP マネージャー，SDP ブローカー，および SDP コーディネーターに関連するファイルをアーカイブファイルに含めます。このオプションは，単独では指定できません。

このオプションを指定すると，作成されるアーカイブファイル名の末尾に_mgr が付きます。

このオプションを指定してhsdpexport コマンドを実行する場合，SDP マネージャーが停止している必要があります。

このオプションを指定することで，次のファイルがアーカイブファイルに追加されます。

- /opt/hitachi/hsdp/conf/*

- /var/log/hitachi/hsdp/hsdpsetupmanager.log

このオプションを指定して作成したアーカイブファイルを展開するには，root 権限が必要です。アーカイブファイルを展開後，次のコマンドを実行して，運用ディレクトリの所有者，グループを root から運用ディレクトリを使用するユーザーとそのグループに書き換えてください。

chown -R *ユーザー名*:*グループ名* *運用ディレクトリ*

-exadp

運用ディレクトリ/exadaptor/直下に格納されている外部アダプターの格納ディレクトリごとに，アーカイブファイルを作成します。作成されるアーカイブファイルの名称は，*外部アダプターディレクトリ名*_exadp.tar.gz になります。

運用ディレクトリ/exadaptor/直下にディレクトリが複数存在した場合，アーカイブファイルはディレクトリの数だけ作成されます。

-help

コマンドの使用方法を表示します。このオプションを指定すると，コマンドはアーカイブファイルを作成しないで終了します。

注意事項

- いったんコマンドを実行したらキャンセルしないでください。コマンドをキャンセルすると，次のディレクトリとファイルが削除されないで残ります。この場合，手動で削除してください。
  - ディレクトリ

    運用ディレクトリ/*出力ファイル名*_work/
  - ファイル

    運用ディレクトリ/export/*出力ファイル名*.tar.gz
- hsdpexport コマンドを実行している間は，hsdpsetup コマンドまたはhsdpunsetup コマンドを実行しないでください。

戻り値

値	説明
0	コマンドが正常終了しました。
1	コマンドの処理中にエラーが発生しました。
2	コマンドがキャンセルされました。
3	コマンドを実行したユーザーには，ディレクトリの書き込み権限がありません。
4	一時ディレクトリを作成できませんでした。

10.17 hsdplogcollect

形式

```
hsdplogcollect {[-file ログ収集定義ファイル] [-dir 出力先ディレクトリパス] | -help}
```

説明

hsdplogcollect コマンドで，HSDP の障害情報と関連情報のファイルを収集してアーカイブします。アーカイブファイルは，HSDP の障害の解析のための保守資料として使用できます。

hsdplogcollect コマンドは，-dir オプションで指定したディレクトリに，アーカイブファイルとして次のファイルを出力します。

```
hsdplogcollect.tar.gz
```

メモ

- 運用ディレクトリが存在するホストでは，*運用ディレクトリ*/bin/に格納されている hsdplogcollect コマンドを実行してください。
- 運用ディレクトリが存在しないホストでは，*インストールディレクトリ*/bin/に格納されているコマンドを実行してください。

ファイルの収集に失敗した場合，次のログファイルにメッセージを出力します。

- *運用ディレクトリ*/bin/に格納されているコマンドを実行した場合
  運用ディレクトリ/logs/hsdplogcollect.log
- *インストールディレクトリ*/bin/に格納されているコマンドを実行した場合
  カレントディレクトリ/hsdplogcollect.log

hsdplogcollect コマンドの実行時，指定したディレクトリに同名のファイルがすでに存在していた場合，そのファイルを上書きします。

実行権限を持つユーザー

- *運用ディレクトリ*/bin/に格納されているコマンドを実行する場合
  運用ディレクトリをセットアップしたユーザー
- *インストールディレクトリ*/bin/に格納されているコマンドを実行する場合
  root 権限を持っているユーザー

コマンド実行の前提条件

次のファイルは，クエリ定義ファイルとして使用できません。また，ログ収集定義ファイル中にも指定できません。

- /snapshotlog.zip
- /osinfo.log

次のファイルまたはディレクトリが存在する場合，スナップショットのログおよびOS情報のログは収集できません。

- /snapshotlog.zip
- /osinfo.log
- /snapshotlog.zip/
- /osinfo.log/

格納先ディレクトリ

運用ディレクトリ/bin/

インストールディレクトリ/bin/

引数

-file *ログ収集定義ファイル*

収集対象のファイルを記述しているログ収集定義ファイルのパスを指定します。
パスは，絶対パスまたはコマンド実行時のカレントディレクトリからの相対パスで指定します。
このオプションの指定を省略すると，製品で定義されている規定のファイルが収集されます。

-dir *出力先ディレクトリパス*

アーカイブファイルの出力先ディレクトリのパスを指定します。
パスは，絶対パスまたはコマンド実行時のカレントディレクトリからの相対パスで指定します。
指定したパスが存在しない場合は，自動的にディレクトリが作成されます。
指定したパスにアクセス権限（書き込み権限）がない場合はエラーになります。
このオプションの指定を省略すると，カレントディレクトリにアーカイブファイルが出力されます。

-help

コマンドの使用方法を表示します。このオプションを指定すると，コマンドはファイルを収集しないで終了します。

注意事項

- このコマンドを実行している間は，hsdpsetup コマンドまたはhsdpunsetup コマンドを実行しないでください。
- [Ctrl] ＋ [C] を入力してコマンドの実行を中断しないでください。
  コマンドの実行を中断すると，次のディレクトリおよびファイルが残ります。この場合，これらを手動で削除してください。

  -dir オプションを指定していない場合

  カレントディレクトリ/hsdplogcollect_work/

  hsdplogcollect.tar.gz

  -dir オプションを指定している場合

  出力先ディレクトリ/hsdplogcollect_work/

  hsdplogcollect.tar.gz
- このコマンドは，同時に複数は実行できません。
- このコマンドを一般ユーザー権限で実行すると，SDP マネージャー，SDP ブローカー，および SDP コーディネーターに関する，一部またはすべての障害情報が取得できません。
- 次のファイルの出力先を変更している場合は変更内容に合わせてログ収集定義ファイルを作成してください。そして，hsdplogcollect コマンドの実行時に，作成したログ収集定義ファイルを-file オプションに指定してください。
  - 共通コンポーネント（コマンド・ロガー）のログファイル
  - SDP マネージャー，SDP ブローカー，SDP コーディネーターのログファイル
  - 外部アダプターのログファイル
  - SDP サーバのログファイル
  - SDP マネージャーコマンドのログファイル
- このコマンドの戻り値やメッセージからコマンドの失敗要因が特定できない場合は，アーカイブファイルの出力先のディスク容量が不足していないか確認してください。
- SDP コーディネーターの起動後は，hsdplogcollect コマンドを実行するたびに SDP コーディネーターに関するスナップショットログが蓄積されるため，アーカイブファイルの容量が約 500 キロバイトずつ増加します。なお，蓄積されたスナップショットログは，hsdpmanager コマンドで SDP コーディネーターを再起動すると削除されます。
- *運用ディレクトリ*/bin/に格納されているコマンドを root 権限を持つユーザーで実行した場合，運用ディレクトリ内の一部のファイルが情報収集のために root 権限で生成されます。そのため，*運用ディレクトリ*/bin/に格納されているコマンドは，運用ディレクトリをセットアップしたユーザーで実行してください。

戻り値

値	説明
0	コマンドが正常終了しました。
1	コマンドの処理中にエラーが発生しました。
2	コマンドを実行したユーザーによってコマンドがキャンセルされました。
3	• コマンドを実行したユーザーにはディレクトリを書き込む権限がありません。 • -dir オプションに指定した出力先ディレクトリパスがディレクトリではありません。
4	コマンドが一時ディレクトリの作成に失敗しました。

10.18 hsdpmanager

形式

```
hsdpmanager {-start | -stop | -help}
```

説明

hsdpmanager コマンドで，SDP マネージャー，SDP ブローカー，および SDP コーディネーターを開始または停止します。

実行権限を持つユーザー

root 権限を持っているユーザー

コマンド実行の前提条件

HSDP が full インストールされていること

格納先ディレクトリ

```
/opt/hitachi/hsdp/bin/
```

引数

-start

自ホスト上の SDP マネージャー，SDP ブローカー，および SDP コーディネーターを開始します。一部の開始に失敗しても処理を続行します。一部の開始に失敗した場合は，出力されたメッセージをもとに対処を実施したあと，コマンドを再実行してください。その際，必要に応じて一度停止してから再実行してください。

-stop

自ホスト上の SDP マネージャー，SDP ブローカー，および SDP コーディネーターを停止します。一部の停止に失敗した場合でも処理を続行します。一部の停止に失敗した場合は，出力されたメッセージをもとに対処を実施したあと，コマンドを再実行してください。

-help

コマンドの使用方法を表示します。このオプションを指定すると，SDP マネージャー，SDP ブローカー，および SDP コーディネーターを開始または停止しないで終了します。ほかのオプションと同時に指定した場合，このオプションの指定が優先されます。

注意事項

特になし。

戻り値

値	説明
0	処理が正常に終了しました。
1	処理が異常終了しました。または，一部のコンポーネントの開始もしくは停止が失敗したあと，処理が終了しました。

10.19 hsdpstatusshow

形式

```
hsdpstatusshow {[-mgr |
-svr |
-qg[ クエリグループ名]... |
-adp[ {アダプターグループ名|カスタムアダプター名}] ...]
[-timeout 秒] |
-help}
```

説明

hsdpstatusshow コマンドで，SDP マネージャー，SDP ブローカー，SDP コーディネーター，SDP サーバ，クエリグループ，およびアダプターグループ（またはカスタムアダプター）の稼働情報を取得して表示します。

取得する情報は，オプションで指定してください。オプションをすべて省略してコマンドを実行した場合は，コマンドを実行した運用ディレクトリを構成する SDP サーバ，クエリグループ，およびアダプターグループ（またはカスタムアダプター）の稼働情報を取得して表示します。

また，コンソールにはすべてJson 形式で表示されます。

実行権限を持つユーザー

運用ディレクトリをセットアップしたユーザー

コマンド実行の前提条件

運用ディレクトリがセットアップ済みであること

格納先ディレクトリ

運用ディレクトリ/bin/

引数

-mgr

次の自ホストに関する稼働情報を取得して表示します。

- SDP マネージャーの状態
- SDP ブローカーのホスト名または IP アドレス，ポート番号，および状態
- SDP コーディネーターのホスト名または IP アドレス，ポート番号，および状態

-svr

SDP サーバの稼働情報を取得して表示します。

-qg[*クエリグループ名*]...

クエリグループ稼働情報と，そのクエリグループが登録されている SDP サーバ名を取得して表示します。このオプションのオプション引数に指定された名前のクエリグループから，稼働情報を取得します。クエリグループ名は，半角スペース区切りで複数指定できます。また，このオプションの引数を省略すると，すべてのクエリグループを指定できます。

引数に指定したクエリグループ名が重複している場合，重複は無視されます。

SDP サーバに未登録のクエリグループ名を指定した場合，エラーになります。

-adp[{*アダプターグループ名*|*カスタムアダプター名*}]...

アダプターグループまたはカスタムアダプターの稼働情報と，そのアダプターを起動した SDP サーバ名を取得して表示します。このオプションのオプション引数に指定された名前のアダプターグループまたはカスタムアダプターから，稼働情報を取得します。アダプターグループ名またはカスタムアダプター名は，半角スペース区切りで複数指定できます。また，このオプションの引数を省略すると，すべてのアダプターグループまたはカスタムアダプターを指定できます。

引数に指定したアダプターグループ名またはカスタムアダプター名が重複している場合，重複は無視されます。

SDP サーバに未登録のアダプターグループ名またはカスタムアダプター名を指定した場合，エラーになります。

-timeout *秒*

このコマンドのタイムアウト時間を指定します。0～60 秒までの整数を指定します。0 を指定した場合，このコマンドはタイムアウトしません。このオプションを省略すると，タイムアウト時間にはデフォルトの30 秒が設定されます。

コマンド実行中にタイムアウト時間が経過した場合，タイムアウト時間内に得られた稼働情報と，コマンドがタイムアウトのために終了したことを示すエラーメッセージを出力して終了します。

-help

コマンドの使用方法を表示します。このオプションを指定すると，コマンドは稼働情報の取得および表示を実施しないで終了します。

注意事項

- -mgr，-svr，-qg，および-adp オプションの指定をすべて省略した場合，すべてのコンポーネントから一括で稼働情報を取得して表示します。
- 次の表に，コマンドオプション間の排他の関係を示します。

表 10-2　コマンドオプション間の排他の関係

オプション名	-mgr	-svr	-qg	-adp	-timeout
-mgr	×	×	×	×	○
-svr	×	×	×	×	○
-qg	×	×	×	×	○
-adp	×	×	×	×	○

オプション名	-mgr	-svr	-qg	-adp	-timeout
-timeout	○	○	○	○	×

（凡例）
○：実行されます。
×：実行時エラーになります。

- 外部アダプターは情報取得の対象外です。

戻り値

値	説明
0	コマンドが正常終了しました。
1	コマンドが異常終了しました。
2	コマンドが［Ctrl］＋［C］で終了しました。
3	コマンドがタイムアウトによって終了しました。
4	コマンドが終了しました。一部の稼働情報またはすべての稼働情報が取得できませんでした。

出力形式

```
{
   "ii...ii" :{
       "MANAGER_STATUS" : "aa...aa",
       "BROKER_HOST_NAME" : "bb...bb",
       "BROKER_PORT" : "ccccc",
       "BROKER_STATUS" : "dd...dd",
       "COORDINATOR_HOST_NAME" : "ee...ee",
       "COORDINATOR_PORT" : "fffff",
       "COORDINATOR_STATUS" : "gg...gg",
       "jj...jj" : {
          "SETUP_USER" : "hh...hh",
          "STATUS" : "kk...kk",
          "CQL_ENGINE_PID" : "lllll",
          "ACCELERATION_CQL_ENGINE_PID" : "mmmmm",
          "SDP_SERVER_COUNT" : "nnnnnnnnn",
          "QUERY_GROUP_COUNT" : "ooooooooo",
          "STREAM_COUNT" : "ppppppppp",
          "QUERY_COUNT" : "qqqqqqqqq",
          "ADAPTOR_GROUP_COUNT" : "rrrrrrrrr",
          "STANDARD_ADAPTOR_COUNT" : "sssssssss",
          "CUSTOM_ADAPTOR_COUNT" : "ttttttttt",
          "QUERY_GROUP" : {
            "uu...uu" : {
              "STATUS" : "vv...vv",
              "MODE" : "ww...ww",
              "TIME_STAMP_MODE" : "xx...xx",
              "STREAM_COUNT" : "yyyyyyyyy",
              "QUERY_COUNT" : "zzzzzzzzz",
              "BB...BB" : {
                "DATA_COUNT" : "CCCCCCCCCCCCCCCCCC",
```

```
        "QUEUE_COUNT" : "DDDDDDDDDD",
        "MAX_COUNT" : "EEEEEEEEEE",
        "ACCURACY_ADJUST_STATUS" : "FF...FF",
        "TUPLE_COUNT" : "GGGGGGGGGG",
        "MAX_TUPLE_COUNT" : "HHHHHHHHHH",
        "REJECT_TUPLE_COUNT" : "IIIIIIIII",
        "FIRST_TUPLE_TIME_STAMP" : "LLLL/LL/LL LL:LL:LL.LLLLLLLLL",
        "END_TUPLE_TIME_STAMP" : "MMMM/MM/MM MM:MM:MM.MMMMMMMMM",
      },
      "NN...NN" :{
        "DATA_COUNT" : "00000000000000000000",
        "QUEUE_COUNT" : "PPPPPPPPPP",
        "MAX_COUNT" : "QQQQQQQQQ",
      }
    }
  },
  "ADAPTOR_GROUP" : {
    "XX...XX" : {
      "STATUS" : "YY...YY",
      "ADAPTOR_TRACE_FEATURE" : "ZZ...ZZ",
      "INPUT_ADAPTOR_COUNT" : "1111111111",
      "OUTPUT_ADAPTOR_COUNT" : "2222222222",
      "33...33" : {
        "STATUS" : "44...44",
        "TYPE" : "55...55",
        "MODE" : "66...66",
        "STREAM_NAME" : "77...77",
        "QUERY_GROUP_NAME" : "88...88",
        "SEND_COUNT" : "99...99",
        "RECEIVE_COUNT" : "1010...1010",
        "CONNECTION_PORT" : "1111...1111",
        "CONNECTION_COUNT" :  "1212...1212",
        "CONNECTION" : [ {
          "1313...1313" : {
            "1414...1414" : {
              "1515...1515" : {
                "1616...1616" : {
                  "1717...1717" : {
                    "CONNECTION_STATUS" : "1818...1818",
                    "WORKING_DIRECTORY_PATH" : "1919...1919"
                  }
                }
              }
            }
          }
        } ]
      }
    }
  },
  "CUSTOM_ADAPTOR" : {
    "2020...2020" : {
      "STATUS" : "2121...2121",
      "MODE" : "2222...2222"
    }
  }
}
```

```
    }
}
```

情報の種類		変数	説明	指定オプション				
				なし	-mgr	-svr	-qg	-adp
SDP マネージャーの情報		*aa…aa*	SDP マネージャーの状態 Starting：開始処理中 Running：稼働状態 Stopping：停止処理中 Stopped：停止状態	○	○	－	－	－
	SDP ブローカーの情報	*bb…bb*	SDP コーディネーターに接続している SDP ブローカーのホスト	○	○	－	－	－
		ccccc	SDP ブローカーのポート番号	○	○	－	－	－
		dd...dd	SDP ブローカーの状態 Starting：開始処理中 Running：稼働状態 Stopping：停止処理中 Stopped：停止状態	○	○	－	－	－
	SDP コーディネーターの情報	*ee…ee*	SDP コーディネーターが起動しているホスト	○	○	－	－	－
		fffff	SDP コーディネーターのポート番号	○	○	－	－	－
		gg…gg	SDP コーディネーターの状態 Starting：開始処理中 Available：稼働状態（使用可能な状態） Unavailable：稼働状態（SDP コーディネーター連携をしている場合で，SDP コーディネーターが多重度数以上停止して使用できなくなっている状態） Stopping：停止処理中 Stopped：停止状態	○	○	－	－	－
運用ディレクトリの情報		－	－	－	－	－	－	－
	SDP サーバの情報	*hh...hh*	運用ディレクトリをセットアップしたユーザー名	○	－	○	－	－
		ii...ii	サーバクラスタ名	○	○	○	○	○
		jj...jj	SDP サーバ名	○	－	○	－	－
		kk…kk	SDP サーバの状態 Initializing：初期化中 Running：稼働状態 Stopping：停止中 Stopped：停止状態	○	－	○	－	－

情報の種類			変数	説明	指定オプション				
					なし	-mgr	-svr	-qg	-adp
	SDP サーバの情報		*kk…kk*	ForceStopping：強制停止中 ForceStopped：強制停止状態 Abnormality：異常状態	○	－	○	－	－
			lllll	CQL エンジンのプロセス ID	○	－	○	－	－
			mmmmm	acceleration CQL のプロセス ID acceleration CQL 登録時以外は-を表示します。	○	－	○	－	－
		総合的な情報	*nnnnnnnnn*	SDP サーバの合計数	○	－	－	－	－
			ooooooooo	登録されているクエリグループ総数（10 桁の 10 進数）	○	－	－	－	－
			ppppppppp	登録されている入力ストリーム総数	○	－	－	－	－
			qqqqqqqqq	登録されているクエリ（出力ストリーム）総数（10 桁の 10 進数）	○	－	－	－	－
			rrrrrrrrr	登録されている標準提供アダプターのアダプターグループの総数（10 桁の 10 進数）	○	－	－	－	－
			sssssssss	登録されている標準提供アダプターの総数（10 桁の 10 進数）	○	－	－	－	－
			ttttttttt	登録されているカスタムアダプターの総数（10 桁の 10 進数）	○	－	－	－	－
		クエリグループの情報	*uu…uu*	クエリグループ名	○	－	－	○	－
			vv…vv	クエリグループの状態 Initialized：初期化中 Starting：開始処理中 Running：実行中 Holding：閉塞処理中 FatalHold：続行不可 Hold：実行時エラーまたはキューあふれによる閉塞中 PreparingForStop：停止要求を受けて入力キューおよび仕掛中のタプルを処理中 Stopping：停止処理中 Stop：停止中 Deleting：削除処理中 AlreadyDeleted：削除済み	○	－	－	○	－
			ww…ww	動作しているクエリグループのモード	○	－	－	○	－

情報の種類			変数	説明	指定オプション				
					なし	-mgr	-svr	-qg	-adp
		クエリグループの情報	*ww…ww*	CQL：Java エンジン用クエリグループ acceleration CQL：C エンジン用クエリグループ	○	－	－	○	－
			xx…xx	クエリグループが動作しているタイムスタンプモード DataSource：データソースモード Server：サーバモード	○	－	－	○	－
			yyyyyyyyyy	クエリグループに所属するストリーム数（10 桁の 10 進数）	○	－	－	○	－
			zzzzzzzzzz	クエリグループに所属するクエリ数（10 桁の 10 進数）	○	－	－	○	－
			BB…BB	ストリーム名	○	－	－	○	－
			CCCCCCCCCCCCCCCCCCCC	ストリームキューに格納されたタプルの合計数（19 桁の 10 進数）	○	－	－	○	－
			DDDDDDDDDD	現在のストリームキューの滞留数（10 桁の 10 進数）	○	－	－	○	－
			EEEEEEEEEE	ストリームキューの最大滞留数（10 桁の 10 進数）	○	－	－	○	－
			FF…FF	タイムスタンプ調整機能の状態 Stop：停止中 Running：実行中	△	－	－	△	－
			GGGGGGGGGG	タイムスタンプ調整機能によって保持されているタプルの数（10 桁の 10 進数）	△	－	－	△	－
			HHHHHHHHHH	タイムスタンプ調整機能によって保持されているタプルの実績最大個数（10 桁の 10 進数）	△	－	－	△	－
			IIIIIIIII	タイムスタンプ調整機能によって破棄されたタプルの数（10 桁の 10 進数）	△	－	－	△	－
			LLLL/LL/LL LL:LL:LL.LLLLLLLLL	入力ストリームに滞留している先頭のタプルの時刻値	○	－	－	○	－
			MMMM/MM/MM MM:MM:MM.MM	入力ストリームに滞留している末端のタプルの時刻値	○	－	－	○	－

情報の種類			変数	説明	指定オプション				
					なし	-mgr	-svr	-qg	-adp
		クエリグループの情報	*MMMMMMMM*	入力ストリームに滞留している末端のタプルの時刻値	○	−	−	○	−
			NN…NN	クエリ名	○	−	−	○	−
			OOOOOOOOOOOOOOOOOOO	出力ストリームキューに格納されたタプルの合計数（19 桁の 10 進数）	○	−	−	○	−
			PPPPPPPPPP	現在，出力ストリームキューに滞留しているタプルの数（10 桁の 10 進数）	○	−	−	○	−
			QQQQQQQQQQ	出力ストリームキューに格納されたタプルの実績値（10 桁の 10 進数）	○	−	−	○	−
		アダプターグループの情報	*XX...XX*	標準提供アダプターのアダプターグループ名	○	−	−	−	○
			YY…YY	標準提供アダプターのアダプターグループの状態 次のどれかが出力されます。 Initializing：開始中 Running：実行中 Stopping：停止処理中 Stopped：停止中 ForceStopping 強制停止中	○	−	−	−	○
			ZZ…ZZ	アダプタートレース機能の状態 次のどちらかが出力されます。 Running：使用可能 Stop：使用不可	○	−	−	−	○
			1111111111	標準提供アダプターのアダプターグループに属している入力アダプターの合計数（10 桁の 10 進数）	○	−	−	−	○
			2222222222	標準提供アダプターのアダプターグループに属している出力アダプターの合計数（10 桁の 10 進数）	○	−	−	−	○
			33...33	アダプター名	○	−	−	−	○
			44...44	アダプターの状態 次のどれかが表示されます。 Initializing：初期化中 Running：実行中 Stopping 停止処理中	○	−	−	−	○

情報の種類				変数	説明	指定オプション				
						なし	-mgr	-svr	-qg	-adp
		アダプターグループの情報		*44...44*	Stopped：停止中 ForceStopping：強制停止中	○	－	－	－	○
				55...55	アダプターの入出力タイプ 次のどちらかが表示されます。 Input：入力タイプ Output：出力タイプ	○	－	－	－	○
				66...66	アダプタータイプ 次のどちらかが表示されます。 C adapter：C アダプター Java adapter：Java アダプター	○	－	－	－	○
				77...77	アダプターが接続するストリーム名※5	○	－	－	－	○
				88...88	アダプターが接続するクエリグループ名※5	○	－	－	－	○
				99...99	アダプターが送信するレコード数※5	○	－	－	－	○
				1010...1010	アダプターが受信したレコード数※5	○	－	－	－	○
				1111...1111	自アダプターのポート番号※5	○	－	－	－	○
				1212...1212	接続数※5 アダプターが接続相手のアダプターと現在確立しているコネクションの数が表示されます。	○	－	－	－	○
			配列	*1313...1313*	接続先のサーバクラスタ名※1	○	－	－	－	○
				1414...1414	接続先のホスト名	○	－	－	－	○
				1515...1515	接続先のアダプター名（SMTP アダプターの場合はポート番号）※1※2	○	－	－	－	○
				1616...1616	接続先のクエリグループ名※1※3	○	－	－	－	○
				1717...1717	接続先のストリーム名※1※3	○	－	－	－	○
				1818...1818	アダプターの接続状況 connecting：接続中	○	－	－	－	○
				1919...1919	接続先の運用ディレクトリパス※4	○	－	－	－	○

情報の種類			変数	説明	指定オプション				
					なし	-mgr	-svr	-qg	-adp
		カスタムアダプターの情報	*2020...2020*	カスタムアダプター名	○	－	－	－	○
			2121...2121	カスタムアダプターの状態 次のどれかが表示されます。 Initializing：起動中 Running：実行中 Stopping：停止処理中 Stopped：停止中 ForceStopping：強制停止中 Abnormality：異常状態	○	－	－	－	○
			2222...2222	カスタムアダプタータイプ 次のどちらかが表示されます。 C adapter：C アダプター Java adapter：Java アダプター	○	－	－	－	○

（凡例）

○：表示される

△：データソースモードが使用されている場合にだけ表示される

－：表示されない

注※1

接続先のカスケーディングアダプターがクエリグループ用プロパティファイルの次のプロパティを指定して起動されている場合に表示されます。

stream.output.*ストリーム名*.link

注※2

接続先が外部アダプターの場合に表示されます。

注※3

接続先が TCP データ入力アダプターの場合に表示されます。

注※4

注※1 のアダプターの場合は，接続先運用ディレクトリパスが表示されます。それ以外のアダプターの場合は-（ハイフン）を表示します。

注※5

C アダプターを使用している場合は-（ハイフン）を表示します。

10.20 hsdptplput

形式

```
hsdptplput {-file タプルログファイル1[△タプルログファイル2...] [-time [開始時刻][,終了時刻]
] [-interval 投入間隔] [-count 投入個数]|-help}
```

説明

hsdptplput コマンドで，タプルログファイルに取得したタプルを入力ストリームキューに再投入します。

実行権限を持つユーザー

運用ディレクトリをセットアップしたユーザー

コマンド実行の前提条件

- SDP サーバが稼働状態であること
- Java エンジンで動作していること
- 再投入対象のストリームのクエリグループが開始していること

格納先ディレクトリ

```
運用ディレクトリ/bin/
```

引数

-file *タプルログファイル1*[△*タプルログファイル2*...]

再投入するタプルログファイルのパスを指定します。

複数のタプルログファイルを指定する場合は，半角スペースで区切って指定します。

ファイルを複数指定する場合は，次のことに注意してください。

- タプルログ取得基準時刻が異なるタプルログファイルを一度に指定すると，ファイル指定エラーになります。

  タプルログ取得基準時刻とは，クエリグループを初期化する契機でタプルログファイルに出力される時刻情報です。この時刻が同じタプルログファイルがクエリの再実行に必要な一連のタプルログとなります。

  各タプルログファイルのタプルログ取得基準時刻は，hsdptplls コマンドで確認できます。
- タプルログが複数ファイルに出力された場合や，複数の入力ストリームを持つクエリグループに再投入する場合，取得したすべてのタプルログファイルを指定しなくてもエラーになりません。指定されたファイルで再投入します。
- 同一のタプルログファイルを複数回指定した場合，ファイル指定エラーになります。

- 同一ストリームキューのタプルログファイルを複数指定した場合，ファイル更新時刻が古いファイルから再投入します。
  各タプルログファイルのファイル更新時刻は，hsdptplls コマンドで確認できます。

-time [*開始時刻*][,*終了時刻*]

再投入するタプルの開始時刻と終了時刻を，次の形式で指定します。
指定された時刻内のタプルだけが再投入されます。

hhmmsslll [*MMDD* [*YYYY*]]

hh：時（00≦*hh*≦23）
mm：分（00≦*mm*≦59）
ss：秒（00≦*ss*≦59）
lll：ミリ秒（000≦*lll*≦999）
MM：月（01≦*MM*≦12）
DD：日（01≦*DD*≦31）
YYYY：年（4 桁の西暦）

> **メモ**
>
> データソースモードでは，タプルログがタイムスタンプの昇順になっていないことがあり，タイムスタンプを使用した時刻によるタプルの抽出が正確に実行できません。そのため，タプルを取得した時刻をタプルログファイルに出力し，その時刻を使用してコマンドで時刻による抽出を実行します。

開始時刻，終了時刻は半角数字で指定してください。
開始，または終了の「年」の指定を省略した場合は，当年の指定月日時刻と見なされます。
「年，月，日」の指定を省略した場合，当年当月当日の指定時刻と見なされます。
「月，日」，「月」，または「日」だけを省略できません。省略した場合はオプションエラーになります。
「月」または「日」を省略したい場合は，「年」，「月」，「日」のすべてを省略してください。
開始時刻，終了時刻はタプルの取得時刻で指定します。タプルの取得時刻は，hsdptplls コマンドで確認できます。
開始時刻と終了時刻は，スペースを挿入しないでコンマで区切って指定します。
時刻の指定方法によって，それぞれ次のように再投入されます。

- 開始時刻を省略した場合
  終了時刻以前のタプルが再投入されます。
- 終了時刻を省略した場合
  開始時刻以降のタプルが再投入されます。
- 開始時刻，終了時刻の両方を省略した場合
  すべてのタプルが再投入されます。
- 開始時刻と終了時刻に同じ値を指定した場合

指定した時刻のタプルだけが再投入されます。

- 終了時刻よりもあとの時刻を開始時刻に指定した場合

  オプションエラーになります。

-interval *投入間隔*（1～600000）

タプルを再投入する間隔をミリ秒単位で指定します。指定を省略した場合，100 ミリ秒が設定されます。

-count オプションで指定した個数のタプルを再投入後，指定した間隔を空けて再投入を開始します。また，入力ストリームキューのキューあふれが発生した場合，再投入に失敗したタプルを指定した間隔を空けて，再度投入します。

-count オプションを指定していない場合，-interval オプションによる投入間隔は，入力ストリームキューのキューあふれが発生したときだけ有効になります。

-count *投入個数*（1～1048576）

一度に再投入するタプルの個数を指定します。

指定した個数のタプルを再投入したあと，-interval オプションで指定した間隔を空けて再投入を開始します。

指定を省略した場合，間隔を空けないですべてのタプルを再投入します。

-help

コマンドの使用方法を表示します。

このオプションが指定された場合，不正チェックをしないでコマンドを終了します。

注意事項

- このコマンドは，複数の入力ストリームキューのタプル入力待ち合わせによるタイムアウトを再現できません。
- デフォルトと異なる文字コードの環境で取得したタプルログファイルを指定した場合，タプルの文字列情報が正しく復元されないことがあります。
- このコマンドを実行した場合，入力アダプターでのタプルの投入と同様にタプルログが取得されます。再投入するときは，使用するタプルログファイルを運用ディレクトリから退避し，退避したファイルを使用して実行してください。
- このコマンドは，最後の入力タプルを再投入してからputEnd メソッドを実行するまでの間隔を再現できません。そのため，サーバモードで取得したタプルログファイルを使用したクエリの再実行では，最後の入力タプルを投入してからputEnd メソッドを実行するまでの間の結果が，入力アダプターを使用してクエリを実行した結果と異なる場合があります。
- このコマンドは，SDP サーバ内でタプルを再投入します。そのため，コマンドプロセスを強制終了した場合，タプルの再投入を継続します。継続中のタプルの再投入を中止するには，hsdpcqlstop コマンドでクエリグループを停止してください。
- -interval オプション，および-count オプションを指定してこのコマンドを実行した場合，クエリグループの状態の変更は，-interval に指定した間隔で検知されます。そのため，このコマンドの実行中にクエリグループを停止させても，すぐに再投入を中止しないことがあります。

- `hsdptplput` コマンドを使用して，Java エンジンで取得したタプルログファイルを acceleration CQL エンジンで動作している SDP サーバに投入しないでください。投入した場合，動作は保証されません。

戻り値

値	説明
0	コマンドが正常に終了しました。
1	コマンドでエラーが発生しました。

10.21 hsdptplls

形式

```
hsdptplls {-file タプルログファイル1[△タプルログファイル2...]
[-time [開始時刻][,終了時刻]] [-info {file|tuple}] [-csv] [-data] [-full]
|-file タプルログファイル1[△タプルログファイル2...] -check|-help}
```

説明

hsdptplput コマンドで，タプルログファイルに取得したタプルの情報を表示します。

表示するタプルログファイルの情報，およびタプル情報の一覧は次のとおりです。

表 10-3 表示するタプルログファイルの情報の一覧

項番	表示内容	意味	備考
1	製品バージョン	タプルログを取得した HSDP のバージョン情報。	なし。
2	クエリグループ名	タプルログを取得したクエリグループ名。	
3	ストリーム名	タプルログを取得したストリームキュー名。	
4	タプルログファイルパス	タプルログファイルのファイルパス。	
5	タプルログ取得基準時刻	HSDP サーバが対象のクエリグループを初期化した時刻。	putEnd()メソッドによってクエリグループが初期化されるときに更新されます。
6	ファイル更新時刻	タプルログファイルの更新を開始した時刻。	なし。
7	タイムスタンプモード	タプルログを取得したクエリグループのタイムスタンプモード。	
8	ストリーム種別	タプルログを取得したストリームキューの入力・出力の種別。	

表 10-4 表示するタプル情報の一覧

項番	表示内容	意味	備考
1	レコード通番	タプルログに割り当てられる通番。	クエリグループの開始時とputEnd()メソッドの実行時に，1に初期化されます。
2	状態	タプルの状態。	なし。

項番	表示内容			意味	備考
3	取得時刻			タプルを取得した時刻。	なし。
4	タプル	タイムスタンプ	日付	タイムスタンプの日付。	
5			時刻	タイムスタンプの時刻。	
6			ナノ秒	タイムスタンプのナノ秒時刻。	-full オプションを指定してこのコマンドを実行した場合に表示されます。
7			通番	同一タイムスタンプの通番。	
8		タプル種別		タプルの種別。	
9		データオブジェクト		タプルのデータ本体。	なし。

実行権限を持つユーザー

運用ディレクトリをセットアップしたユーザー

コマンド実行の前提条件

タプルログファイルの生成元であるクエリグループが停止していること

格納先ディレクトリ

運用ディレクトリ/bin/

引数

-file *タプルログファイル1*[△*タプルログファイル2*...]

表示するタプルログファイルパスを指定します。

複数のタプルログファイルを指定する場合は，半角スペースで区切って指定します。この場合，指定したファイルの順番に情報が表示されます。

-time [*開始時刻*][,*終了時刻*]

表示するタプルの開始時刻と終了時刻を，次の形式で指定します。

指定された時刻内のタプルだけが表示されます。

なお，-info オプションでfile を指定した場合は，このオプションの指定は無効になります。

hhmmsslll [*MMDD* [*YYYY*]]

hh：時（00≦*hh*≦23）

mm：分（00≦*mm*≦59）

ss：秒（00≦*ss*≦59）

lll：ミリ秒（000≦*lll*≦999）

MM：月（01≦*MM*≦12）

DD：日（01≦*DD*≦31）

YYYY：年（4桁の西暦）

> **メモ**
>
> データソースモードでは，タプルログがタイムスタンプの昇順になっていないことがあり，タイムスタンプを使用した時刻によるタプルの抽出が正確に実行できません。そのため，タプルを取得した時刻をタプルログファイルに出力し，その時刻を使用してコマンドで時刻による抽出を実行します。

開始時刻，終了時刻は半角数字で指定してください。

開始，または終了の「年」の指定を省略した場合は，当年の指定月日時刻と見なされます。

「年，月，日」の指定を省略した場合，当年当月当日の指定時刻と見なされます。

「月，日」，「月」，または「日」だけを省略できません。省略した場合はオプションエラーになります。

「月」または「日」を省略したい場合は，「年」，「月」，「日」のすべてを省略してください。

開始時刻，終了時刻はタプルの取得時刻で指定します。タプルの取得時刻は，このコマンドで確認できます。

開始時刻と終了時刻は，スペースを挿入しないでコンマで区切って指定します。

時刻の指定方法によって，それぞれ次のように再投入されます。

- 開始時刻を省略した場合
  終了時刻以前のタプルが表示されます。
- 終了時刻を省略した場合
  開始時刻以降のタプルが表示されます。
- 開始時刻，終了時刻共に省略した場合
  すべてのタプルが表示されます。
- 開始時刻と終了時刻に同じ値を指定した場合
  すべてのタプルが表示されます。
- 終了時刻よりもあとの時刻を開始時刻に指定した場合
  オプションエラーになります。

`-info {file|tuple}`

表示する情報の種類を指定します。

- `file`：ファイルの情報を表示します。
- `tuple`：タプルの情報を表示します。

指定を省略した場合は，タプルの情報が表示されます。

-csv

表示内容を，CSV ファイル形式で表示します。

このオプションを指定した場合，タイムスタンプなどの時刻情報は，日付と時刻で 1 つのデータとして扱われます。

-data

タプルのデータオブジェクトの内容を表示します。

-info オプションでfile を指定した場合，このオプションの指定は無効になります。

-csv オプションを指定した場合，このオプションで表示されるタプルのデータオブジェクトは 1 つのデータとして扱われます。

-full

タプルログの表示結果に，ナノ秒，通番，タプル種別を付加して表示します。

また，putEnd()メソッドで投入された，入力が完了したタプルのタプル情報を表示します。

-info オプションでfile を指定した場合，このオプションの指定は無効になります。

-check

タプルログファイルに，タプルの欠落があるかを確認します。

-help

コマンドの使用方法を表示します。

このオプションが指定された場合，タプルの情報の表示，およびオプションの不正チェックをしないでコマンドを終了します。

注意事項

- タプルログファイル生成元のクエリグループが開始しているときにこのコマンドを実行した場合，エラーが発生し，メッセージKFSP94307-E が出力されます。
- デフォルトと異なる文字コードの環境で取得したタプルログファイルを指定した場合，タプルの文字列情報が正しく復元されないことがあります。

戻り値

値	説明
0	コマンドが正常に終了しました。
1	コマンドでエラーが発生しました。

出力形式

-info file オプション指定時

- -csv オプションを指定しない場合

```
File Name        : aa...aa
Version          : bb-bb-bb
```

```
Query Group Name : cc...cc
Stream Name      : dd...dd
Time Stamp Mode  : ee...ee
Stream Type      : ff...ff
Base Time        : gggg/gg/gg gg:gg:gg ggg
Update Time      : hhhh/hh/hh hh:hh:hh hhh

...（ファイルごとに繰り返し）
```

- -csv オプションを指定する場合

```
File Name,Version,Query Group Name,Stream Name,Time Stamp Mode,Stream Type,Base Time,U
pdate Time
aa...aa,bb-bb-bb,cc...cc,dd...dd,ee...ee,ff...ff,gggg/gg/gg gg:gg:gg ggg,hhhh/hh/hh hh
:hh:hh hhh
...（ファイルごとに繰り返し）
```

出力形式中の，各変数の意味を次に示します。

変数	意味
aa...aa	タプルログファイルのファイルパス（絶対パス）
bb-bb-bb	タプルログを取得した HSDP のバージョン情報
cc...cc	タプルログを取得したクエリグループ名（1～64 文字の文字列）
dd...dd	タプルログを取得したストリーム名（1～100 文字の文字列）
ee...ee	タプルログを取得したクエリグループのタイムスタンプモード 次のどちらかが表示されます。 • DataSource：データソースモード • Server：サーバモード
ff...ff	タプルログを取得したストリームの種別 • Input：入力ストリーム • Output：出力ストリーム
gggg/gg/gg gg:gg:gg ggg	タプルログ取得基準時刻（年/月/日 時:分:秒 ミリ秒） タプルログ取得基準時刻とは，クエリグループを初期化する契機でタプルログファイルに出力される時刻情報です。この時刻が同じタプルログファイルが，クエリの再実行に必要な一連のタプルログとなります。
hhhh/hh/hh hh:hh:hh hhh	タプルログファイルの更新開始時刻（年/月/日 時:分:秒 ミリ秒）

-info tuple オプション指定時

- -csv オプション，および-data オプションを指定しない場合

```
File Name : ii...ii
Record No Record Time             Time Stamp              Nano Sec Sub No  Tuple Type
Status
jj...jj   kkkk/kk/kk kk:kk:kk kkk llll/ll/ll ll:ll:ll lll mm...mm  nn...nn oo...oo
pp...pp
...(タプルごとに繰り返し)
...(ファイルごとに繰り返し)
```

- -csv オプションを指定しないで，-data オプションを指定する場合

```
File Name : ii...ii
Record No Record Time             Time Stamp              Nano Sec Sub No  Tuple Type
Status  Data
jj...jj   kkkk/kk/kk kk:kk:kk kkk llll/ll/ll ll:ll:ll lll mm...mm  nn...nn oo...oo   p
p...pp qq...qq
...(タプルごとに繰り返し)
...(ファイルごとに繰り返し)
```

- -csv オプションを指定し，-data オプションを指定しない場合

```
File Name,Record No,Record Time,Time Stamp,Nano Sec,Sub No,Tuple Type,Status
ii...ii,jj...jj,kkkk/kk/kk kk:kk:kk kkk, llll/ll/ll ll:ll:ll lll, mm...mm,nn...nn,oo..
.oo,pp...pp
...(タプルごとに繰り返し)
...(ファイルごとに繰り返し)
```

- -csv オプション，および-data オプションを指定する場合

```
File Name,Record No,Record Time,Time Stamp,Nano Sec,Sub No,Tuple Type,Status,Data
ii...ii,jj...jj,kkkk/kk/kk kk:kk:kk kkk,llll/ll/ll lll, mm...mm,nn...nn,oo...oo,pp...p
p,qq...qq
...(タプルごとに繰り返し)
...(ファイルごとに繰り返し)
```

出力形式中の，各変数の意味を次に示します。

変数	意味
ii...ii	タプルログファイルのファイルパス（絶対パス）
jj...jj	タプルログに割り当てられる通番（19 桁の 10 進数）
kkkk/kk/kk kk:kk:kk kkk	タプルログを取得した時刻（年/月/日 時:分:秒 ミリ秒）
llll/ll/ll ll:ll:ll lll	タプルのタイムスタンプ（年/月/日 時:分:秒 ミリ秒） タイムスタンプがないタプルの場合は，「--/--/-- --:--:-- ---」が表示されます。
mm...mm	タプルのタイムスタンプのナノ秒（19 桁の 10 進数） -full オプションを指定した場合に表示されます。 タイムスタンプがないタプルの場合は，「----------」が表示されます。
nn...nn	同一のタイムスタンプのタプルに付加される通番（10 桁の 10 進数） -full オプションを指定した場合に表示されます。 タイムスタンプがないタプルの場合は，「----------」が表示されます。
oo...oo	タプルの種別 -full オプションを指定した場合に表示されます。 表示されるタプルの種別を次に示します。 • PlusTuple 入力ストリームに投入されたタプル • HeartBeatTuple 入力ストリームの時刻を，指定した時刻に設定するタプル

変数	意味
oo...oo	• MinusTuple ストリーム生存期間を過ぎたタプル • EndTuple 入力データの送信完了通知 • ControlTuple 入出力データを制御するためのタプル これらのタプル種別以外のタプルの場合は，「------」が表示されます。
pp...pp	タプルの状態 タプルの状態が表示されます。 表示されるタプルの状態を次に示します。 • Put 正常です。 • PutEnd タプルの投入が完了しています。 状態がPutEnd のタプルの情報は，-full オプションを指定した場合に表示されます。 • TimeBack 時刻の逆転によって破棄されています。 • Suspend タプルの投入が中断されます。 • Resume タプルの投入が再開されます。 これらのタプルの状態以外の場合は，「------」が表示されます。
qq...qq	データオブジェクトの内容（[カラム１｜カラム２｜…]） -data オプションを指定した場合にだけ表示されます。 データオブジェクトを持たないタプル（タプルの種別がハートビートタプルの場合，およびタプルの状態がResume のタプルの場合を含む）の場合は，「[]」が表示されます。 タプルの状態がSuspend の場合，引数に指定したboolean 型は次のとおり数字に変換されて表示されます。 • True：1 • False：0 制御タプルの場合は，制御タプルに付与されたオブジェクトの情報が「|」で結合されて表示されます。

-check オプション指定時

```
File Name : rr...rr
First No Last No
ss...ss  tt...tt
Lost No
uu...uu-uu...uu
...(欠落ごとに繰り返し)

...(ファイルごとに繰り返し)
```

出力形式中の，各変数の意味を次に示します。

変数	意味
rr...rr	タプルログファイルのファイルパス（絶対パス）
ss...ss	タプルログファイルの先頭の通番※（19 桁の 10 進数）
tt...tt	タプルログファイルの末尾の通番※（19 桁の 10 進数）
uu...uu-uu...uu	タプルが欠落している範囲（欠落範囲の先頭の通番※-欠落範囲の末尾の通番※）

注※
　通番とは，タプルログに割り当てられる番号のことです。

11 定義ファイルのパラメーターへの変換とパラメーターファイルの形式

次の定義ファイルは，パラメーターファイルを用意することでパラメーター変換ができます。

- クエリ定義ファイル
- クエリグループ用プロパティファイル
- 外部定義関数定義ファイル
- アダプター構成定義ファイル

この章では，定義ファイルのパラメーター値の設定について説明します。また，パラメーターファイルの記述形式を示します。

11.1　定義ファイル

記述形式

定義ファイルにパラメーターを導入したテンプレートファイルは，Velocity Template Language（VTL）に基づいて記述する必要があります。

[指定項目]

テンプレートファイルの規則は，以下のとおりです。

- 行

  自然行は改行コードで終わる 1 行の文字列です。

  論理行とは改行コードの代わりにバックスラッシュ文字（¥）を使用し，改行コードをエスケープすることで隣接している複数の自然行までを含めて 1 行とするものです。

- コメント

  ハッシュマークを 2 つ（##）使用することでコメントとして扱われます。

  「# *」と「* #」で自然行を囲むことで，複数の行をコメントにできます。ほかの方法でコメントを追加する場合は，構成ファイルの名前を変更し，ファイルをマージしないでください。

- 変数

  変数は，構成ファイルに挿入する必要がある場所を識別する識別子です。パラメーターファイルで定義される変数を構成ファイルで使用する場合，${*パラメーターファイルで定義したパラメーター名*}と記述します。

- 変数の命名規則

  変数名称に使用できる文字は，半角の英字（a～z，A～Z），数字（0～9），ハイフン（-），およびアンダーライン（_）です。変数名称の最初の文字は英字である必要があります。

  > **メモ**
  >
  > hsdp_で始まるパラメーターは，HSDP によって予約されているため使用できません。

- 四則演算

  変数の値に実行する必要がある四則演算は，「#set Δ (${variable part name} Δ *演算記号* Δ ${variable part name})」と記述します。使用できる演算記号は，加算記号（+），減算記号（-），乗算記号（*），および除算記号（/）です。

  [例]

  ```
  #set Δ (${aaa} Δ + Δ ${bbb})
  ```

  (凡例)

  Δ：スペース

- 代入

「#set Δ ($variable part name Δ = Δ ”character string”)」や「#set Δ ($variable part name Δ = Δ ”numeric value”)」のように定義すると，変数に文字列または数値が挿入されます。

> **メモ**
>
> 変数に別の変数は代入できません。例えば，「#set Δ ${aaa} Δ = Δ ${bbb}」の場合，変数に値が挿入されると，定義の一部が欠落することがあります。

(凡例)

Δ：スペース

- 条件分岐

条件式を使用すると，「#if(*条件式1*) Δ*処理1* Δ#elseif(*条件式2*) Δ*処理2* Δ#else Δ*処理3* Δ#end 」のように定義することで，変数にさまざまな値を挿入できます。

条件式は数値と文字列を比較できます。また，「#elseif」は 0 回以上繰り返すこともできます。

(凡例)

Δ：スペース

[例]

- 等号：#if($foo == $bar)
- より大きい：#if($foo > 24)
- より小さい#if($foo < 24)
- 以上：#if($foo >= 24)
- 以下：#if($foo <= 24)
- 数値の等号：#if($foo == 24)
- 文字列の等号：#if($foo == ”bar”)
- 論理否定：#if(!$foo)

- 実例

要素内で数値を使用する：

扱えるのは，10 進数として解釈できる数値です※。

次の表に，変換後に削除される不要な数値を示します。

S. No.	変換前	変換後	
1	00320	320	冒頭の0 が削除されます。
2	003.20	3.2	冒頭の0 と，小数点以下の末尾の0 が削除されます。

小数を使用する場合，変換後，値は文字列として保存されます。

式：java.lang.Double のpublic static String to String (Double *d*)の仕様に従います。

注※

整数として指定できるのは，java.lang.Long で扱うことができる最小値から最大値の範囲です。小数として指定できるのは，java.lang.Double で扱うことができる最小値から最大値の範囲です。

ファイル名

定義ファイル名は，各定義ファイルの規則に従う必要があります。

> **メモ**
>
> ファイルをパラメーターに変換した後で定義ファイル名を変更しないでください。

ファイル格納場所

定義ファイルのディレクトリは，各定義ファイルの規則に従う必要があります。

> **メモ**
>
> ファイルをパラメーターに変換した後でディレクトリを変更しないでください。

説明

定義ファイルの中で，変換したい部分を「${*パラメーター名*}」に置き換える必要があります。パラメーターファイルでは，key-value 形式（*パラメーター名=値*）を使用して，パラメーター名（定義ファイルに記述されている）とパラメーター名の値を指定する必要があります。

> **メモ**
>
> パラメーターがほかの定義ファイルで使用されていると，演算コマンドの実行時にエラーが発生します。

指定できるパラメーター

なし

例

次のクエリグループ用プロパティファイルの例は，サンプルに含まれているパラメーターのほかに，新しいパラメーターをユーザーが定義ファイルに記述する方法を示しています。

```
querygroup.cqlFilePath=${cqlFilePath}
stream.input.S1.dispatch.type=${S1_type}
stream.input.S1.dispatch.rule=${S1_rule}
stream.input.S2.dispatch.type=${S2_type}
stream.input.S2.dispatch.rule=${S2_rule}
    :
(snip)
```

11.2　HSDP パラメーターファイル

記述形式

パラメーターファイルでキー値形式（*パラメーター名=値*）を使用して，定義ファイルに含める必要があるパラメーターの名前と，置き換える必要がある値を指定します。

パラメーター名には，hsdp_で始まる文字列は使用しないでください。

> **メモ**
>
> 製品に同梱されているサンプル定義ファイルとサンプルパラメーターファイルには，hsdp_で始まる幾つかのパラメーターが含まれています。新たなパラメーターを追加するときは，hsdp_で始まるパラメーター名は使用しないでください。このようなパラメーターを追加した場合，正常な動作は保証されません。

ファイル名

任意の名前.param

> **メモ**
>
> 任意のファイル名を使用して複数のパラメーターファイルを作成できます。

ファイル格納場所

運用ディレクトリ/conf/

説明

パラメーターファイルは，パラメーター値を設定する必要がある部分に埋め込むべきパラメーターの名前と値を定義します。

指定できるパラメーター

なし

例

サンプルに指定されて製品に同梱されているパラメーターに加えて，新しいパラメーターを使用する必要がある場合は，パラメーターファイルに定義を追加する必要があります。

次に示すパラメーターファイルの例は，「11.1　定義ファイル」の例で使用されているクエリグループ用プロパティファイルに対応しており，パラメーターファイルにパラメーターの定義を記述する方法を示しています。

```
cqlFilePath=/home/user1/wk1/query/q001
S1_type=hashing
S1_rule=column11,column12,column13
S2_type=hashing
S2_rule=column21,column22
    :
(snip)
```

12 SDP サーバおよび関連コンポーネント用定義ファイル

この章では，SDP サーバおよび関連コンポーネント用定義ファイルについて説明します。

12.1　SDP サーバおよび関連コンポーネント用定義ファイルの説明の記述形式

この章では，SDP サーバおよび関連コンポーネント用定義ファイルの説明を次の形式で記述します。ただし，ファイルによっては説明しない項目もあります。

記述形式

ファイルの記述形式を示します。

ファイル名

ファイル名を示します。

ファイルの格納先

ファイルの格納先を示します。

説明

ファイルの用途について説明します。

指定できるパラメーター

ファイルの中で指定できるパラメーターについて説明します※。

注※

システムコンフィグプロパティファイル（system_config.properties）などの一部のファイルについては，パラメーターの詳細を別の項で説明します。

注意事項

ファイル作成時の注意事項について説明します。

12.2　SDP サーバおよび関連コンポーネント用定義ファイル作成上の注意事項

SDP サーバおよび関連コンポーネント用定義ファイルを作成する際の注意事項を次に示します。

- #で始まる行は，コメント行として扱われます。
- ファイルパスを記述する際のファイルセパレーターは「/」と記述してください。

  記述例を次に示します。

  ```
  SDP_CLASS_PATH=/opt/hitachi/hsdp/system/xeads/javaclient/lib/hntrlib2-eads-j.jar
  ```

- システム起動後は，次のファイルの内容を変更しないでください。
  - SDP サーバ用 JavaVM オプションファイル（jvm_options.cfg）
  - acceleration CQL エンジン用 JavaVM オプションファイル（jvm_cengine_options.cfg）
  - システムコンフィグプロパティファイル（system_config.properties）
  - SDP サーバのログファイル出力用プロパティファイル（logger.properties）
  - 共通コンポーネント（コマンド・ロガー）のログファイル出力用プロパティファイル（hsdplogger.properties）
  - cascading 用プロパティファイル
- クエリグループ用プロパティファイル，およびストリーム用プロパティファイルを変更する場合は，「4.3　プロパティファイルの設定値を変更する」を参照してください。
- SDP サーバおよび関連コンポーネント用定義ファイルは，java.util.Properties クラスの仕様に従って記述してください。

12.3 SDP サーバおよび関連コンポーネント用定義ファイルの一覧

SDP サーバおよび関連コンポーネント用定義ファイルの一覧を次の表に示します。

表 12-1 SDP サーバおよび関連コンポーネント用定義ファイルの一覧

項番	分類	定義ファイル	概要	格納先ディレクトリ
1	JavaVM オプションの設定※	SDP サーバ用 JavaVM オプションファイル（jvm_options.cfg）	運用ディレクトリごとに必ず作成します。 このファイルには，SDP サーバを実行する際の JavaVM オプションを設定します。 なお，インプロセス連携アダプターは，このファイルで設定したオプションで動作します。	*運用ディレクトリ*/conf/
2		acceleration CQL engine 用 JavaVM オプションファイル（jvm_cengine_options.cfg）	acceleration CQL engine を適用する場合，運用ディレクトリごとに作成します。 このファイルには，acceleration CQL engine を実行する際の JavaVM オプションを設定します。 ファイルの記述形式はjvm_options_cfg と同じです。	*運用ディレクトリ*/conf/
3		SDP マネージャー用 JavaVM オプションファイル（manager_jvm.cfg）	hsdpmanager -start コマンドで SDP マネージャーを起動する場合，および SDP マネージャーを再起動する場合に読み込む JavaVM オプションを指定します。このファイルの各パラメーターで指定する JavaVM オプションは，SDP マネージャーを起動する際のjava コマンドの引数として指定されます。	*インストールディレクトリ*/system/manager/conf/
4		SDP ブローカー用 JavaVM オプションファイル（broker_jvm.cfg）	hsdpmanager -start コマンドで SDP ブローカーを起動する場合，および SDP ブローカーを再起動する場合に読み込む JavaVM オプションを指定します。このファイルの各パラメーターで指定する JavaVM オプションは，SDP ブローカーを起動する際のjava コマンドの引数として指定されます。	*インストールディレクトリ*/system/manager/conf/
5		SDP コーディネーター用 JavaVM オプションファイル（coordionator_jvm.cfg）	hsdpmanager -start コマンドで SDP コーディネーターを起動する場合，および SDP コーディネーターを再起動する場合に読み込む JavaVM オプションを指定します。このファイルの各パラメーターで指定する JavaVM オプションは，SDP コーディネーターを起動する際のjava コマンドの引数として指定されます。	*インストールディレクトリ*/system/manager/conf/

項番	分類	定義ファイル	概要	格納先ディレクトリ
6	動作環境プロパティの設定	システムコンフィグプロパティファイル（system_config.properties）	運用ディレクトリごとに必ず作成します。 SDP サーバで使用するポート番号やチューニングパラメーター をこのファイルに設定します。	*運用ディレクトリ*/conf/
7		クエリグループ用プロパティファイル	クエリグループごとに必ず作成します。 このファイルには，クエリ定義ファイルのパスやクエリグループを実行するときのチューニングパラメーターを設定します。	*運用ディレクトリ*/conf/
8		ストリーム用プロパティファイル	クエリグループ内のストリーム単位でチューニングパラメーターを設定したい場合だけ，ストリームごとに作成します。 このファイルを作成しない場合には，クエリグループ用プロパティファイルで設定した内容で動作します。	*運用ディレクトリ*/conf/
9		インプロセス連携アダプター用プロパティファイル（user_app.*アダプターグループ名またはアダプター名*.properties）	標準提供アダプターはアダプターグループごと，カスタムアダプターはアダプターごとに作成します。 このファイルには，インプロセス連携アダプターのクラス名や jar ファイルのパスを設定します。	*運用ディレクトリ*/conf/
10		SDP サーバのログファイル出力用プロパティファイル（logger.properties）	運用ディレクトリごとに作成します。作成しない場合には，デフォルト値で動作します。 このファイルには，メッセージログやトレースログの面数やサイズを設定します。	*運用ディレクトリ*/conf/
11		共通コンポーネント（コマンド・ロガー）のログファイル出力用プロパティファイル（hsdplogger.properties）	運用ディレクトリごとに作成します。作成しない場合には，デフォルト値で動作します。 このファイルには，メッセージログの面数やサイズを設定します。	*運用ディレクトリ*/conf/
12		SDP マネージャー定義ファイル	SDP マネージャーが次のコンポーネントを管理するためのプロパティを定義します。 • SDP マネージャー • SDP ブローカー • SDP コーディネーター • SDP サーバ	/opt/hitachi/hsdp/conf/
13		外部アダプター定義ファイル	外部アダプターに関連するプロパティを指定します。	*任意のディレクトリ*
14		自動生成アダプターテンプレートファイル	製品内部で自動起動される内部アダプターに関連する定義を指定します。	*運用ディレクトリ*/inadaptor/conf/xml

項番	分類	定義ファイル	概要	格納先ディレクトリ
15	ロガーに関連するプロパティの設定	SDP マネージャーコマンドのログ定義ファイル	SDP マネージャー，SDP ブローカー，および SDP コーディネーターのhsdpmanagerコマンドのロガーに関するプロパティを設定する場合に作成します。 このファイルには，ログファイルの面数やサイズを設定します。	/opt/hitachi/hsdp/conf/
16	ログ収集の設定	ログ収集定義ファイル	ログファイルを収集したい場合に，このファイルを作成します。 hsdplogcollect コマンドで収集させるログのファイルパスをこのファイルに指定します。	*任意のディレクトリ*
17	そのほかの設定	エクスポート定義ファイル	hsdpexport コマンドによってエクスポートするファイルを指定します。	*運用ディレクトリ*/conf/

注

定義ファイル内で変数を使用したい場合は，「11.　定義ファイルのパラメーターへの変換とパラメーターファイルの形式」を参照してください。

注※

JavaVM オプションについては，「12.19　JavaVM オプションの一覧」を参照してください。

SDP サーバおよび関連コンポーネントから出力されるログファイルと，定義ファイルの対応を次の表に示します。

表 12-2　ログファイルの一覧

項番	ログファイル	設定ファイル
1	SDP サーバのログファイル	SDP サーバのログファイル出力用プロパティファイル（logger.properties）
2	共通コンポーネント（コマンド・ロガー）のログファイル	共通コンポーネント（コマンド・ロガー）のログファイル出力用プロパティファイル（hsdplogger.properties）
3	SDP マネージャー，SDP ブローカー，SDP コーディネーターのログファイル	SDP マネージャー定義ファイル
4	外部アダプターのログファイル	外部アダプター定義ファイル
5	SDP マネージャーのコマンドログファイル	SDP マネージャーコマンドのログ定義ファイル
6	運用ディレクトリのセットアップログファイル	-
7	SDP マネージャーのセットアップログファイル	-
8	エクスポートログファイル	-

（凡例）

-：該当なし

12.4 SDP サーバ用 JavaVM オプションファイル（jvm_options.cfg）

記述形式

パラメーターは次の形式で記述します。

パラメーター名=JavaVMオプション

- スペースを指定できるのはコメントだけです。ただし，JavaVM オプションとしてスペースを含むパスを指定する場合には，パスをダブルクォーテーション（”）で囲んで記述してください。記述例を次に示します。

  （例）

  ```
  SDP_JVM_LOG=-XX:HitachiJavaLog:”/var/abc/SDPServerVM”
  ```

- 1 つのパラメーター名に指定できる JavaVM オプションは 1 つです。

ファイル名

jvm_options.cfg

ファイルの格納先

このファイルは，必ず次のディレクトリに格納してください。

運用ディレクトリ/conf/

説明

hsdpstart コマンドで SDP サーバを起動するとき，または Streaming Data Platform の各種コマンドを実行するときの JavaVM オプションを指定します。インプロセス連携アダプターは，SDP サーバと同じ JavaVM 上で動作するため，このファイルのSDP_USER_OPT パラメーターで指定したオプションで動作します。このファイルは，運用ディレクトリごとに必ず作成します。

なお，このファイルの各パラメーターで指定する JavaVM オプションは，次に示すようにjava コマンドのコマンドラインに指定されます。

SDP_CLASS_PATH パラメーターの場合

JavaVM オプションは，コロン区切りで，java コマンドの-classpath オプションに指定されます。

ほかのパラメーターの場合

JavaVM オプションは，半角スペース区切りで，java コマンドの引数として指定されます。

指定できるパラメーター

指定できるパラメーターとデフォルト値を次の表に示します。なお，JavaVM オプションについては，「12.19　JavaVM オプションの一覧」を参照してください。

表 12-3　指定できるパラメーターとデフォルト値（jvm_options.cfg）

項番	パラメーター名	内容	デフォルト値（省略時の値）
1	SDP_CLASS_PATH	インプロセス連携アダプターが使用するjar ファイルを指定します。指定するファイルは，システム構成によって異なります。 複数のjar ファイルを指定する場合，jar ファイルごとに指定します。指定例を次に示します。 (例) SDP_CLASS_PATH=. SDP_CLASS_PATH=/var/sdp/AP なお，jar ファイルを相対パスで指定する場合は，運用ディレクトリからの相対パスを記述してください。	なし
2	SDP_CLASSLIB_TRACE	クラスライブラリのスタックトレースを出力するかどうかを指定します。	-XX:-HitachiJavaClassLibTrace
3	SDP_CLASSLIB_TRACE_LINESIZE	クラスライブラリのスタックトレースの 1 行の文字数を指定します。	-XX:HitachiJavaClassLibTraceLineSize=1024
4	SDP_GC	ガーベージコレクションの発生時に拡張verbosegc 情報を出力するかどうかを指定します。	-XX:-HitachiVerboseGC
5	SDP_GC_PRINT_CAUSE	ガーベージコレクションの要因内容を出力するかどうかを指定します。	-XX:+HitachiVerboseGCPrintCause
6	SDP_INITIAL_MEM_SIZE	Java ヒープの初期サイズを指定します。	-Xms7.8m
7	SDP_JVM_LOG	ログファイル名のプレフィックスを指定します。	-XX:HitachiJavaLog:javalog
8	SDP_JVM_LOG_FILE_SIZE	ログファイルの 1 ファイル当たりの最大ファイルサイズを指定します。	-XX:HitachiJavaLogFileSize=256k
9	SDP_LOCALS_IN_STACK_TRACE	スレッドダンプ出力時のスタックトレースに，ローカル変数情報を出力するかどうかを指定します。	-XX:-HitachiLocalsInStackTrace
10	SDP_LOCALS_SIMPLE_FORMAT	ローカル変数情報出力を簡易フォーマットにするかどうかを指定します。	-XX:-HitachiLocalsSimpleFormat
11	SDP_MAX_MEM_SIZE	Java ヒープの最大サイズを指定します。	-Xmx83m
12	SDP_MAX_METASPACE_SIZE	Metaspace 領域の最大サイズを指定します。	-XX:MaxMetaspaceSize=2^64-1
13	SDP_NEW_RATIO	DefNew 領域に対するTenured 領域の割合を指定します。	-XX:NewRatio=2

項番	パラメーター名	内容	デフォルト値（省略時の値）
14	SDP_OOM_STACK_TRACE	OutOfMemoryError 発生時にスタックトレースを出力するかどうかを指定します。	-XX:-HitachiOutOfMemoryStackTrace
15	SDP_OUTPUT_MILLI_TIME	ミリ秒までの時間を出力するかどうかを指定します。	-XX:-HitachiOutputMilliTime
16	SDP_METASPACE_SIZE	Metaspace 領域の初期サイズを指定します。	-XX:MetaspaceSize=16m
17	SDP_SYS_OPT	Streaming Data Platform のコマンドの実行に必要な JavaVM オプションを指定します。 システム環境に依存して，コマンドでも JavaVM オプションを指定する必要がある場合に，このパラメーターを指定します。 このパラメーターの指定は，Streaming Data Platform のコマンドを実行する場合だけ有効です。	なし
18	SDP_THRD_DUMP	標準出力にスレッドダンプを出力するかどうかを指定します。	-XX:+HitachiThreadDumpToStdout
19	SDP_TRUE_TYPE_IN_LOCALS	ローカル変数情報出力時に，ローカル変数オブジェクトの実際の型名を文字列として出力するかどうかを指定します。	-XX:-HitachiTrueTypeInLocals
20	SDP_USE_G1GC	ガーベージコレクションとして G1GC を使用するかどうかを指定します。	-XX:-UseG1GC
21	SDP_USER_OPT	次のどちらかの場合に，このパラメーターで JavaVM オプションを指定します。 • 運用ユーザーが JavaVM オプションを追加したい場合 • インプロセス連携アダプターを実行するときの JavaVM オプションを指定したい場合 複数の JavaVM オプションを指定する場合には，オプションごとに指定します。指定例を次に示します。 （例） SDP_USER_OPT=-D*xxx=www* SDP_USER_OPT=-D*yyy=zzz* なお，同一のオプションを複数回指定した場合は，あとで指定したオプション（ファイルの末尾に近い方のオプション）が有効になります。	なし

注意事項

- リモートオブジェクトの参照に対して定期的にガーベージコレクションが発生することがあります。ガーベージコレクションの発生間隔は，RMI 連携時に指定する Java のsun.rmi.dgc.client.gcInterval プロパティ，およびsun.rmi.dgc.server.gcInterval プロパティにミリ秒単位で指定できます。デフォルトは60,000 ミリ秒（60 秒）です。ガーベージコレクションの発生間隔を変更する場合は，SDP_USER_OPT パラメーターにsun.rmi.dgc.client.gcInterval プロパティ，および

sun.rmi.dgc.server.gcInterval プロパティを指定し，任意の間隔を指定してください。指定できる値の範囲は，1～9,223,372,036,854,775,806 です。指定例を次に示します。

（例）

```
SDP_USER_OPT=-Dsun.rmi.dgc.client.gcInterval=1000000000
SDP_USER_OPT=-Dsun.rmi.dgc.server.gcInterval=1000000000
```

12.5　SDP マネージャー用 JavaVM オプションファイル（manager_jvm.cfg）

記述形式

パラメーターは次の形式で記述します。

パラメーター名=*JavaVMオプション*

- スペースを指定できるのはコメントだけです。ただし，JavaVM オプションとしてスペースを含むパスを指定する場合には，パスをダブルクォーテーション（”）で囲んで記述してください。記述例を次に示します。

  （例）

  ```
  MANAGER_JVM_LOG=-XX:HitachiJavaLog:”/var/abc/ManagerVM”
  ```

- 1 つのパラメーター名に指定できる JavaVM オプションは 1 つです。

ファイル名

manager_jvm.cfg

ファイルの格納先

このファイルは，必ず次のディレクトリに格納してください。

インストールディレクトリ/system/manager/conf/

説明

hsdpmanager -start コマンドで SDP マネージャーを起動する場合，および SDP マネージャーを再起動する場合に読み込む JavaVM オプションを指定します。このファイルの各パラメーターで指定する JavaVM オプションは，SDP マネージャーを起動する際の java コマンドの引数として指定されます。

指定できるパラメーター

指定できるパラメーターとデフォルト値を次の表に示します。なお，JavaVM オプションについては，「12.19　JavaVM オプションの一覧」を参照してください。

表 12-4　指定できるパラメーターとデフォルト値（jvm_options.cfg）

項番	パラメーター名	内容	デフォルト値（省略時の値）
1	MANAGER_INITIAL_MEM_SIZE	Java ヒープの初期サイズを指定します。	-Xms7.8m
2	MANAGER_MAX_MEM_SIZE	Java ヒープの最大サイズを指定します。	-Xms83m

項番	パラメーター名	内容	デフォルト値（省略時の値）
3	MANAGER_NEW_RATIO	DefNew 領域に対するTenured 領域の割合を指定します。	-XX:NewRatio=2
4	MANAGER_METASPACE_SIZE	Metaspace 領域の初期サイズを指定します。	-XX:MetaspaceSize=16m
5	MANAGER_MAX_METASPACE_SIZE	Metaspace 領域の最大サイズを指定します。	-XX:MaxMetaspaceSize=2^64-1
6	MANAGER_THRD_DUMP	標準出力にスレッドダンプを出力するかどうかを指定します。	-XX:+HitachiThreadDumpToStdout
7	MANAGER_JVM_LOG	ログファイル名のプレフィックスを指定します。	-XX:HitachiJavaLog:javalog
8	MANAGER_JVM_LOG_FILE_SIZE	ログファイルの 1 ファイル当たりの最大ファイルサイズを指定します。	-XX:HitachiJavaLogFileSize=256k
9	MANAGER_GC	ガーベージコレクションの発生時に拡張verbosegc 情報を出力するかどうかを指定します。	-XX:-HitachiVerboseGC
10	MANAGER_USE_G1GC	ガーベージコレクションとしてG1GC を使用するかどうかを指定します。	-XX:-UseG1GC
11	MANAGER_GC_PRINT_CAUSE	ガーベージコレクションの要因内容を出力するかどうかを指定します。	-XX:+HitachiVerboseGCPrintCause
12	MANAGER_OUTPUT_MILLI_TIME	ミリ秒までの時間を出力するかどうかを指定します。	-XX:-HitachiOutputMilliTime
13	MANAGER_OOM_STACK_TRACE	OutOfMemoryError 発生時のスタックトレースを出力するかどうかを指定します。	-XX:-HitachiOutOfMemoryStackTrace
14	MANAGER_CLASSLIB_TRACE	クラスライブラリのスタックトレースを出力するかどうかを指定します。	-XX:-HitachiJavaClassLibTrace
15	MANAGER_CLASSLIB_TRACE_LINESIZE	クラスライブラリのスタックトレースの 1 行の文字数を指定します。	-XX:HitachiJavaClassLibTraceLineSize=1024
16	MANAGER_LOCALS_SIMPLE_FORMAT	ローカル変数情報出力を簡易フォーマットにするかどうかを指定します。	-XX:-HitachiLocalsSimpleFormat
17	MANAGER_TRUE_TYPE_IN_LOCALS	ローカル変数情報出力時に，ローカル変数オブジェクトの実際の型名を文字列として出力するかどうかを指定します。	-XX:-HitachiTrueTypeInLocals
18	MANAGER_LOCALS_IN_STACK_TRACE	スレッドダンプ出力時のスタックトレースに，ローカル変数情報を出力するかどうかを指定します。	-XX:-HitachiLocalsInStackTrace

12.6　SDP ブローカー用 JavaVM オプションファイル（broker_jvm.cfg）

記述形式

パラメーターは次の形式で記述します。

パラメーター名=*JavaVMオプション*

- スペースを指定できるのはコメントだけです。ただし，JavaVM オプションとしてスペースを含むパスを指定する場合には，パスをダブルクォーテーション（”）で囲んで記述してください。記述例を次に示します。

  （例）

  ```
  BROKER_JVM_LOG=-XX:HitachiJavaLog:”/var/abc/BrokerVM”
  ```

- 1 つのパラメーター名に指定できる JavaVM オプションは 1 つです。

ファイル名

broker_jvm.cfg

ファイルの格納先

このファイルは，必ず次のディレクトリに格納してください。

インストールディレクトリ/system/manager/conf/

説明

hsdpmanager -start コマンドで SDP ブローカーを起動する場合，および SDP ブローカーを再起動する場合に読み込む JavaVM オプションを指定します。このファイルの各パラメーターで指定する JavaVM オプションは，SDP ブローカーを起動する際の java コマンドの引数として指定されます。

なお，このファイルの各パラメーターで指定する JavaVM オプションは，次に示すように java コマンドのコマンドラインに指定されます。

SDP_CLASS_PATH パラメーターの場合

JavaVM オプションは，コロン区切りで，java コマンドの -classpath オプションに指定されます。

ほかのパラメーターの場合

JavaVM オプションは，半角スペース区切りで，java コマンドの引数として指定されます。

指定できるパラメーター

指定できるパラメーターとデフォルト値を次の表に示します。なお，JavaVM オプションについては，「12.19　JavaVM オプションの一覧」を参照してください。

表 12-5　指定できるパラメーターとデフォルト値（jvm_options.cfg）

項番	パラメーター名	内容	デフォルト値（省略時の値）
1	BROKER_INITIAL_MEM_SIZE	Java ヒープの初期サイズを指定します。	-Xms7.8m
2	BROKER_MAX_MEM_SIZE	Java ヒープの最大サイズを指定します。	-Xms83m
3	BROKER_NEW_RATIO	DefNew 領域に対するTenured 領域の割合を指定します。	-XX:NewRatio=2
4	BROKER_METASPACE_SIZE	Metaspace 領域の初期サイズを指定します。	-XX:MetaspaceSize=16m
5	BROKER_MAX_METASPACE_SIZE	Metaspace 領域の最大サイズを指定します。	-XX:MaxMetaspaceSize=2^64-1
6	BROKER_THRD_DUMP	標準出力にスレッドダンプを出力するかどうかを指定します。	-XX:+HitachiThreadDumpToStdout
7	BROKER_JVM_LOG	ログファイル名のプレフィックスを指定します。	-XX:HitachiJavalog:javalog
8	BROKER_JVM_LOG_FILE_SIZE	ログファイルの 1 ファイル当たりの最大ファイルサイズを指定します。	-XX:HitachiJavaLogFileSize=256k
9	BROKER_GC	ガーベージコレクションの発生時に拡張verbosegc 情報を出力するかどうかを指定します。	-XX:-HitachiVerboseGC
10	BROKER_USE_G1GC	ガーベージコレクションとしてG1GC を使用するかどうか指定します。	-XX:-UseG1GC
11	BROKER_GC_PRINT_CAUSE	ガーベージコレクションの要因内容を出力するかどうかを指定します。	-XX:+HitachiVerboseGCPrintCause
12	BROKER_OUTPUT_MILLI_TIME	ミリ秒までの時間を出力するかどうかを指定します。	-XX:-HitachiOutputMilliTime
13	BROKER_OOM_STACK_TRACE	OutOfMemoryError 発生時のスタックトレースを出力するかどうかを指定します。	-XX:-HitachiOutOfMemoryStackTrace
14	BROKER_CLASSLIB_TRACE	クラスライブラリのスタックトレースを出力するかどうかを指定します。	-XX:-HitachiJavaClassLibTrace
15	BROKER_CLASSLIB_TRACE_LINESIZE	クラスライブラリのスタックトレースの 1 行の文字数を指定します。	-XX:HitachiJavaClassLibTraceLineSize=1024
16	BROKER_LOCALS_SIMPLE_FORMAT	ローカル変数情報出力を簡易フォーマットにするかどうかを指定します。	-XX:-HitachiLocalsSimpleFormat
17	BROKER_TRUE_TYPE_IN_LOCALS	ローカル変数情報出力時に，ローカル変数オブジェクトの実際の型名を文字列として出力するかどうかを指定します。	-XX:-HitachiTrueTypeInLocals

項番	パラメーター名	内容	デフォルト値（省略時の値）
18	BROKER_LOCALS_IN_STACK_TRACE	スレッドダンプ出力時のスタックトレースに，ローカル変数情報を出力するかどうかを指定します。	-XX:-HitachiLocalsInStackTrace

12.7 SDP コーディネーター用 JavaVM オプションファイル (coordionator_jvm.cfg)

記述形式

パラメーターは次の形式で記述します。

パラメーター名=JavaVMオプション

- スペースを指定できるのはコメントだけです。ただし，JavaVM オプションとしてスペースを含むパスを指定する場合には，パスをダブルクォーテーション（”）で囲んで記述してください。記述例を次に示します。

  （例）

  ```
  COORDINATOR_JVM_LOGGER_DIR=”/var/abc/coordinator/log”
  ```

- 1 つのパラメーター名に指定できる JavaVM オプションは 1 つです。

ファイル名

coordinator_jvm.cfg

ファイルの格納先

このファイルは，必ず次のディレクトリに格納してください。

インストールディレクトリ/system/manager/conf/

説明

hsdpmanager -start コマンドで SDP コーディネーターを起動する場合，および SDP コーディネーターを再起動する場合に読み込む JavaVM オプションを指定します。このファイルの各パラメーターで指定する JavaVM オプションは，SDP コーディネーターを起動する際java コマンドの引数として指定されます。

SDP コーディネーター用 JavaVM オプションファイルがない場合，または読み取りができない場合，すべてのパラメーターにデフォルト値が指定されます。

指定できるパラメーター

指定できるパラメーターとデフォルト値を次の表に示します。なお，JavaVM オプションについては，「12.19 JavaVM オプションの一覧」を参照してください。

表 12-6　指定できるパラメーターとデフォルト値（coordionator_jvm.cfg）

項番	パラメーター名	内容	デフォルト値（省略時の値）
1	COORDINATOR_MAX_MEM_SIZE_VALUE	Java ヒープの最大サイズ（単位：メガバイト）を指定します。ここで指定した値が，JavaVM のヒープサイズオプション（-Xmx および-Xms）に適用されます。パラメーターが未定義，または設定した値が整数値（0～9）でない場合，デフォルト値で動作します。	1024
2	COORDINATOR_MAX_METASPACE_SIZE_VALUE	Metaspace 領域の最大値（単位：メガバイト）を指定します。 ここで指定した値が，JavaVM のメモリサイズオプション（-XX:MetaspaceSize および-XX:MaxMetaspaceSize）に適用されます。 パラメーターが未定義，または設定した値が整数値（0～9）でない場合，デフォルト値で動作します。	1024
3	COORDINATOR_JVM_LOGGER_DIR	日立 JavaVM ログファイルが出力されるディレクトリを指定します。日立 JavaVM ログファイルはjavalog から始まるファイル名で出力されます。	/var/lib/hitachi/hsdp/coordinator/xeads/log
4	COORDINATOR_JVM_LOG_FILE_SIZE	日立 JavaVM ログファイルの 1 ファイルの最大ファイルサイズ（単位：メガバイト）を指定します。パラメーターが未定義，または設定した値が整数値（0～9）でない場合は，デフォルト値で動作します。 • ファイルサイズ（単位：メガバイト）の指定可能範囲 1～2097152	4

注意事項

- コーディネーターグループを形成する場合，COORDINATOR_MAX_MEM_SIZE パラメーターと COORDINATOR_MAX_METASPACE_SIZE パラメーターは，すべてのコーディネーター間で同じ値になるように設定してください。同じ値でない場合，hsdpmanager -start コマンドの実行時にコーディネーターグループが形成できないため，コーディネーターが起動しません。

12.8　システムコンフィグプロパティファイル（system_config.properties）

システムコンフィグプロパティファイル（system_config.properties）と，指定するパラメーターの詳細について説明します。

12.8.1　システムコンフィグプロパティファイルの詳細

記述形式

パラメーターは次の形式で記述します。

```
パラメーター名=値
```

- 値の後ろにスペースやコメントなどの文字列を追加できません。
  追加した場合は値の一部として解釈されます。
- パラメーター名だけを指定して，値を指定していない行は，無視されてデフォルト値が指定されたとして動作します。

ファイル名

system_config.properties

ファイルの格納先

このファイルは，必ず次のディレクトリに格納してください。

```
運用ディレクトリ/conf/
```

説明

SDP サーバで使用するポート番号を設定します。このファイルは，運用ディレクトリごとに必ず作成します。

指定できるパラメーター

指定できるパラメーターとデフォルト値を次の表に示します。なお，パラメーターの詳細については，「12.8.2　システムコンフィグプロパティファイルのパラメーターの詳細」で説明します。

表 12-7　指定できるパラメーターとデフォルト値（system_config.properties）

項番	パラメーター名	内容	デフォルト値	指定できる値の範囲
1	engine.externalStreamFuncVerifyMode	ストリーム間演算の戻り値の型を検証するかどうかを指定します。	true	• true • false
2	engine.initialEngineQueueSize	オペレーター間をつなぐエンジンキューで使用する，要素数の初期値を指定します。	1024	1～2147483647 の整数
3	engine.initialQueueSize	ストリームキューに登録する要素数の初期値を指定します。	1024	1～10485760 の整数
4	engine.maxEngineQueueSize	オペレーター間をつなぐエンジンキューで使用する，要素数の上限値を指定します。	4096	1～2147483647 の整数 ただし，次の条件を満たす値を指定します。 engine.initialEngineQueueSize≦ engine.maxEngineQueueSize
5	engine.maxQueueSize	ストリームキューに登録する要素数の上限値を指定します。	4096	1～2147483647 の整数
6	engine.useTupleLog	タプルログを取得するかどうかを指定します。 • true：タプルログを取得します。 • false：タプルログを取得しません。	false	• true • false
7	engine.watchQueueSize.threshold	ストリームキューに登録する要素数を監視するしきい値（単位：%）を指定します。	なし	1～99 の整数
8	logger.console.abnormal.enabled	ログファイルの初期化で障害が発生した場合に，SDP サーバの起動を続行させるかどうかを指定します。 • true：ログ出力機能を使用しないで SDP サーバの起動を続行します。 • false：SDP サーバの起動を中断します。	true	• true • false
9	mon.process_exp_time	タイムアウトと判定する処理時間（単位：ミリ秒）を指定します。	30000	1～2147483647 の整数

項番	パラメーター名	内容	デフォルト値	指定できる値の範囲
10	query.decimalMaxPrecision	クエリの算術演算の結果がDECIMAL型となる場合の，算術演算の最大精度を指定します。	38	1～38の整数
11	query.decimalRoundingMode	クエリの演算結果がDECIMAL型となる場合の，query.decimalMaxPrecisionパラメーターで指定した精度を超えたときの丸めの動作を指定します。	HALF_UP	• HALF_UP • DOWN
12	querygroup.sleepOnOverStore	SDPサーバが出力ストリームキューに空きがあるかどうかをチェックして空きがなかった場合に，クエリグループの実行をスリープさせる時間（単位：ミリ秒）を指定します。	100	1～2147483647の整数
13	querygroup.sleepOnOverStoreRetryCount	SDPサーバがクエリ実行後のタプルを出力ストリームキューへ投入する前に，出力ストリームキューに空きがあるかどうかをチェックする回数を指定します。	-1	-1～2147483647の整数
14	rmi.serverPort	SDPサーバのポート番号を指定します。	20400	1～65535の整数
15	stream.freeInputQueueSizeThreshold	入力ストリームキューに登録する要素数の上限値に対する空きサイズのしきい値（単位：%）を指定します。	なし	1～99の整数
16	stream.freeInputQueueSizeThresholdOutputMessage	SDPサーバのメッセージログに，警告メッセージを出力するかどうかを指定します。 • true：警告メッセージを出力します。 • false：警告メッセージを出力しません。	false	• true • false
17	stream.maxKeepTupleCount	タイムスタンプ調整機能で保留できるタプル数の上限値を指定します。	125828	1～1048576の整数
18	stream.timestampAccuracy	タイムスタンプ調整機能の時刻単位と時刻調整範囲を指定します。	なし	「12.8.2　システムコンフィグプロパティファイルのパラメーターの詳細」の

項番	パラメーター名	内容	デフォルト値	指定できる値の範囲
18	stream.timestampAccuracy	タイムスタンプ調整機能の時刻単位と時刻調整範囲を指定します。	なし	「stream.timestampAccuracy= { {sec | msec | usec} ,*時刻調整範囲* | unuse} 」を参照してください。
19	stream.timestampMode	タプルに時刻を付与するためのタイムスタンプモードを指定します。 • Server：サーバモードを使用します。 • DataSource：データソースモードを使用します。	Server	• Server • DataSource
20	stream.timestampPosition	タプル内の時刻データ列名を指定します。 または，stream.timestampMode が DataSource かつ，入力ストリームがカスケーディングアダプターからのデータを受け付ける場合には，__systemtime__を指定します。大文字，小文字の区別はされません。	なし	「12.8.2　システムコンフィグプロパティファイルのパラメーターの詳細」の「stream.timestampPosition=*時刻データ列名*」を参照してください。
21	stream.tupleLogMode	タイムスタンプモードがサーバモードの場合に hsdptplput コマンドを実行するかどうかを指定します。 true：hsdptplput コマンドを実行します。 false：hsdptplput コマンドを実行しません。	false	• true • false
22	tpl.backupFileCount	タプルログファイルのバックアップを残す最大世代数を指定します。 0 を指定した場合，バックアップを取得しません。	1	0～10 の整数
23	tpl.bufferCount	タプルログのバッファーの面数を指定します。	5	3～512 の整数
24	tpl.bufferSize	タプルログのバッファーの最大サイズ（単位：キロバイト）を指定します。	1024	1～2048000 の整数
25	tpl.fileCount	タプルログファイルの最大ファイル数を指定します。	3	3～512 の整数

項番	パラメーター名	内容	デフォルト値	指定できる値の範囲
26	tpl.fileSize	タプルログファイルの最大サイズ（単位：メガバイト）を指定します。	100	1～2048 の整数
27	tpl.outputLevel	タプルログの出力レベルを指定します。 1. ストリームキューへ格納するタプルについて，タプルログを出力します。 2. 時刻の逆転によって破棄するタプルについて，タプルログを出力します。 3. レベル 1 とレベル 2 のタプルについて，タプルログを出力します。	3	1～3 の整数
28	tpl.outputTrigger	タプルログファイルへの出力契機を指定します。 • 未指定：入力ストリームに投入されたタプルだけを出力します。 • buffer：入力ストリームおよび出力ストリームに投入されたタプルのタプルログを出力します。 • none：タプルログを出力しません。	未指定	• 未指定 • buffer • none
29	tpl.useOverwrite	タプルログのバッファーの枯渇時にバッファーを上書きするかどうかを指定します。 • true：タプルログのバッファーを上書きします。 • false：タプルログのバッファーを上書きしません。	true	• true • false

12.8.2　システムコンフィグプロパティファイルのパラメーターの詳細

ここでは，「12.8.1　システムコンフィグプロパティファイルの詳細」の「指定できるパラメーター」で示したシステムコンフィグプロパティファイル（system_config.properties）のパラメーターの詳細について説明します。

engine.externalStreamFuncVerifyMode=*ストリーム間演算の戻り値の型の検証*

ストリーム間演算の戻り値の型を検証するかどうかをtrue またはfalse で指定します。大文字，小文字の区別はされません。デフォルトはtrue です。

true

検証します。

false

検証しません。

false を指定することで，ストリーム間演算関数を頻繁に呼び出すシステムで性能が改善する場合があります。ただし，検証をしない場合は，トラブルシュートが困難になるので注意してください。

このパラメーターは，システムコンフィグプロパティファイル，およびクエリグループ用プロパティファイルで指定できます。指定が重複した場合，または指定を省略した場合，次の優先順位で指定値が有効になります（1.> 2.)。

1. クエリグループ用プロパティファイル
2. システムコンフィグプロパティファイル

engine.initialEngineQueueSize=*オペレーター間をつなぐエンジンキューで使用する要素数の初期値*

オペレーター間をつなぐエンジンキューで使用する，要素数の初期値を1～2147483647 の整数で指定します。デフォルトは1024 です。

> **メモ**
>
> ここで指定した要素数の初期値は，個々のエンジンキューに対する初期値になります。

engine.initialQueueSize=*ストリームキューに登録する要素数の初期値*

ストリームキューに登録する要素数の初期値を1～10485760 の整数で指定します。デフォルトは1024 です。

ここで指定した要素数を超える要素数をストリームキューに登録しようとした場合，engine.maxQueueSize パラメーターで設定する上限値までストリームキューが拡張されます。

> **メモ**
>
> ここで指定した要素数の初期値は，個々のストリームキューに対する初期値になります。

engine.maxEngineQueueSize=*オペレーター間をつなぐエンジンキューで使用する要素数の上限値*

オペレーター間をつなぐエンジンキューで使用する要素数の上限値を1～2147483647 の整数で指定します。デフォルトは4096 です。

オペレーター間をつなぐエンジンキューで使用する要素数の上限値は，engine.initialEngineQueueSize パラメーターで指定した要素数の初期値以上となるように指定してください。

> **メモ**
>
> ここで指定した要素数の上限値は，個々のエンジンキューに対する上限値になります。

engine.maxQueueSize=*ストリームキューに登録する要素数の上限値*

ストリームキューに登録する要素数の上限値を1～2147483647 の整数で指定します。デフォルトは4096 です。

ストリームキューに登録する要素数の上限値は，engine.initialQueueSize パラメーターで指定した要素数の初期値以上となるように指定してください。

> **メモ**
>
> ここで指定した要素数の上限値は，個々のストリームキューに対する上限値になります。

engine.useTupleLog= {true | false}

タプルログを取得するかしないかをtrue またはfalse で指定します。デフォルトはfalse です。

true
: タプルログを取得します。

false
: タプルログを取得しません。

engine.watchQueueSize.threshold=*ストリームキューに登録する要素数を監視するしきい値*

ストリームキューに登録する要素数を監視するしきい値（単位：%）を1～99 の整数で指定します。このパラメーターを省略した場合，要素数は監視されません。

次の計算式で算出される値が，このパラメーターの指定値を超えると，警告メッセージが出力されます。

要素数÷*engine.maxQueueSizeパラメーターの指定値*×100

警告メッセージは，いったん出力されると，要素数がしきい値を下回ったあとに再びしきい値を上回るまで出力されません。

logger.console.abnormal.enabled= {true | false}

ログファイルの初期化で障害が発生した場合に，SDP サーバの起動を続行させるかどうかをtrue またはfalse で指定します。大文字，小文字の区別はされません。デフォルトはfalse です。

true
　メッセージをコンソールに出力して SDP サーバを起動します。

false
　SDP サーバを停止します。

logger.console.enabled= {true | false}

SDP サーバが出力するメッセージをコンソールに出力するかどうかをtrue またはfalse で指定します。大文字，小文字の区別はされません。デフォルトはfalse です。

true
　メッセージをログファイルとコンソールの両方に出力します。

false
　メッセージをコンソールに出力しません。

mon.process_exp_time=*タイムアウト処理時間*

タイムアウトと判定する処理時間（単位：ミリ秒）を1～2147483647 の整数で指定します。デフォルトは30000 です。

デフォルト値でタイムアウト検知のメッセージが頻発する場合は，十分な余裕を持った値を指定してください。

query.decimalMaxPrecision=*クエリの算術演算の最大精度*

クエリの算術演算の結果がDECIMAL 型（NUMERIC 型を含めます）となる場合の，算術演算の最大精度を1～38 の整数で指定します。デフォルトは38 です。

演算結果がDECIMAL 型となる算術演算とは，次の算術演算です。

- 演算項にDECIMAL 型を含む四則演算（2 項演算）
- 引数がDECIMAL 型となる集合関数AVG/SUM
- DECIMAL 型以外のデータ型のDECIMAL 型へのキャスト

算術演算の結果が，このパラメーターで指定した精度を超えない場合は，算術演算の結果の精度は加工されません。超える場合は，このパラメーターで指定した精度に丸められます。丸めの動作は，query.decimalRoundingMode パラメーターで指定します。

query.decimalRoundingMode= {HALF_UP | DOWN}

クエリの演算結果がDECIMAL 型（NUMERIC 型を含めます）となる場合の，query.decimalMaxPrecision パラメーターで指定した精度を超えたときの丸めの動作を指定します。デフォルトはHALF_UP です。

HALF_UP

破棄される小数部の最上位の桁を四捨五入して丸めます。

DOWN

破棄される小数部の最上位の桁より 1 つ上の桁の値が減るように丸めます。

querygroup.sleepOnOverStore=*クエリグループの実行をスリープさせる時間*

SDP サーバが出力ストリームキューに空きがあるかどうかをチェックして空きがなかった場合に，クエリグループの実行をスリープさせる時間（単位：ミリ秒）を1～2147483647 の整数で指定します。デフォルトは100 です。

このパラメーターの指定は，querygroup.sleepOnOverStoreRetryCount パラメーターで1 以上を指定した場合に有効です。

このパラメーターによってスリープしたクエリグループに対しては，次のことが実行できます。

- 入力アダプターによる入力ストリームキューへのタプルの投入
- 出力アダプターによる出力ストリームキューからのタプルの取得

このパラメーターは，システムコンフィグプロパティファイル，およびクエリグループ用プロパティファイルで指定できます。指定が重複した場合，または指定を省略した場合，次の優先順位で指定値が有効になります（1.> 2.）。

1. クエリグループ用プロパティファイル
2. システムコンフィグプロパティファイル

querygroup.sleepOnOverStoreRetryCount=*出力ストリームキューの空きをチェックする回数*

SDP サーバがクエリ実行後のタプルを出力ストリームキューへ投入する前に，出力ストリームキューに空きがあるかどうかをチェックする回数を-1～2147483647 の整数で指定します。デフォルトは-1 です。

1～2147483647 を指定した場合，SDP サーバは，指定された回数だけ次の処理を実行します。

1. 出力ストリームキューに空きがあるかどうかをチェックします。

2. 出力ストリームキューに空きがなければ，出力ストリームキューのあるクエリグループの実行をスリープさせます。スリープさせる時間は，querygroup.sleepOnOverStore パラメーターで指定します。
   スリープ後，このパラメーターで指定した回数分の処理を実行していない場合は，1.の処理に戻ります。
3. 出力ストリームキューに空きがあれば，出力ストリームキューへクエリ実行後のタプルを投入します。

0 を指定した場合，SDP サーバは，出力ストリームキューに空きがあるかをチェックしないで，クエリ実行後のタプルを出力ストリームキューへ投入します。

-1 を指定した場合，SDP サーバは，出力ストリームキューに空きがあるかをチェックします。空きがない場合，次の処理を出力ストリームキューに空きができるまで実行します。

1. 出力ストリームキューに空きがあるかどうかをチェックします。
2. 出力ストリームキューに空きがなければ，出力ストリームキューのあるクエリグループの実行をスリープさせます。スリープさせる時間は，querygroup.sleepOnOverStore パラメーターで指定します。スリープ後，空きがない場合は，1.の処理に戻ります。
3. 出力ストリームキューに空きがあれば，出力ストリームキューへクエリ実行後のタプルを投入します。

この指定値とquerygroup.sleepOnOverStore パラメーターの指定値の組み合わせによって，次の属性の値が指定されます。

- アダプター構成定義ファイルのtcpcon:input タグの@queueFullRetryInterval 属性
- アダプター構成定義ファイルのカスケーディングクライアントコネクター定義のcascading タグの @queueFullRetryInterval 属性

この指定値とquerygroup.sleepOnOverStore パラメーターの指定値との組み合わせと，それによって指定される@queueFullRetryInterval 属性の値を次の表に示します。

querygroup.sleepOnOverStoreRetryCount パラメーターの値	querygroup.sleepOnOverStore パラメーターの値	@queueFullRetryInterval 属性の値
0	• 1～2147483647 • 未指定	-1
• -1 • 1～2147483647 • 未指定	1～2147483647	querygroup.sleepOnOverStore パラメーターの値
	未指定	100（デフォルト値）

このパラメーターは，システムコンフィグプロパティファイル，およびクエリグループ用プロパティファイルで指定できます。指定が重複した場合，または指定を省略した場合，次の優先順位で指定値が有効になります（1.> 2.）。

1. クエリグループ用プロパティファイル
2. システムコンフィグプロパティファイル

rmi.serverPort=*SDPサーバのポート番号*

SDP サーバのポート番号を1～65535 の整数で指定します。デフォルトは20400 です。

デフォルトのポート番号（20400）が，ほかのシステムのポート番号と重複する場合は，別のポート番号を指定してください。ただし，1～1023（Well Known ポート番号）は使用しないことをお勧めします。

stream.freeInputQueueSizeThreshold=*入力ストリームキューの最大サイズに対する空きサイズのしきい値*

入力ストリームキューに登録する要素数の上限値（engine.maxQueueSize パラメーターの指定値）に対する空きサイズのしきい値（単位：%）を1～99 の整数で指定します。

次に示す条件を満たした場合に，put(StreamTuple *tuple*)メソッドまたはput(ArrayList<StreamTuple> *tuple_list*)メソッドからSDPClientFreeInputQueueSizeThresholdOverException の例外がスローされます。このとき，入力ストリームキューへのタプルの投入は，成功しています。

このパラメーターの指定値 ≧（（*入力ストリームキューの空きサイズ*÷*入力ストリームキューの最大サイズ*）×100）

このパラメーターを省略した場合は，しきい値のチェックによる例外は発生しません。

> **メモ**
>
> engine.watchQueueSize.threshold パラメーターは，入力および出力ストリームキューの最大サイズに対する使用サイズのしきい値を設定するものであり，stream.freeInputQueueSizeThreshold とは異なるパラメーターなので注意してください。詳細については，engine.watchQueueSize.threshold パラメーターを参照してください。

このパラメーターは，システムコンフィグプロパティファイル，クエリグループ用プロパティファイル，およびストリーム用プロパティファイルで指定できます。指定が重複した場合，または指定を省略した場合，次の優先順位で指定値が有効になります（1.> 2.> 3.）。

1. ストリーム用プロパティファイル
2. クエリグループ用プロパティファイル
3. システムコンフィグプロパティファイル

stream.freeInputQueueSizeThresholdOutputMessage= {true | false}

SDP サーバのメッセージログに，警告メッセージ（メッセージKFSP42032-W）を出力するかどうかをtrue またはfalse で指定します。大文字，小文字の区別はされません。デフォルトはfalse です。

true

警告メッセージを出力します。

false

警告メッセージを出力しません。

このパラメーターの指定は，stream.freeInputQueueSizeThreshold パラメーターが指定された場合だけ有効です。

なお，警告メッセージが出力されるのは，このパラメーターでtrue を指定し，次の条件を満たした場合です。

stream.freeInputQueueSizeThresholdパラメーターの指定値 ≧（（*入力ストリームキューの空きサイズ÷入力ストリームキューの最大サイズ*）×100）

このパラメーターは，システムコンフィグプロパティファイル，クエリグループ用プロパティファイル，およびストリーム用プロパティファイルで指定できます。指定が重複した場合，または指定を省略した場合，次の優先順位で指定値が有効になります（1.＞ 2.＞ 3.）。

1. ストリーム用プロパティファイル
2. クエリグループ用プロパティファイル
3. システムコンフィグプロパティファイル

stream.maxKeepTupleCount=*タイムスタンプ調整機能で保留できるタプル数の上限値*

タイムスタンプ調整機能で保留できるタプル数の上限値を1～1048576 の整数で指定します。デフォルトは125828 です。

このパラメーターで指定したタプル数の上限値は，個々の入力ストリームのタイムスタンプ調整機能に対する上限値となります。

タプル数がこのパラメーターで指定した上限値を超えると，クエリグループが閉塞されます。

このパラメーターは，システムコンフィグプロパティファイル，クエリグループ用プロパティファイル，およびストリーム用プロパティファイルで指定できます。指定が重複した場合，または指定を省略した場合，次の優先順位で指定値が有効になります（1.＞ 2.＞ 3.）。

1. ストリーム用プロパティファイル
2. クエリグループ用プロパティファイル
3. システムコンフィグプロパティファイル

stream.timestampAccuracy= { {sec | msec | usec} ,*時刻調整範囲* | unuse}

タイムスタンプ調整機能の時刻単位と時刻調整範囲を指定します。大文字，小文字の区別はされません。

stream.timestampMode パラメーターでDataSource を指定した場合，このパラメーターを必ず指定してください。stream.timestampMode パラメーターでServer を指定した場合，このパラメーターの指定は無視されますが，パラメーターの形式チェックは実施されます。

{sec | msec | usec} ,*時刻調整範囲*

時刻単位と時刻調整範囲を指定します。時刻単位（sec，msec，またはusec）と時刻調整範囲を区切る半角コンマ（,）の前後には，スペースやタブなどを記述しないでください。記述した場合はエラーとなります。時刻単位にsec，msec，またはusec のどれかを指定し，時刻調整範囲に0 を指定した場合は基準時刻だけが時刻調整範囲となります。

それぞれの値の意味を次に示します。

sec

時刻単位として秒を使用することを指定します。

msec

時刻単位としてミリ秒を使用することを指定します。

usec

時刻単位としてマイクロ秒を使用することを指定します。

時刻調整範囲

タイムスタンプ調整での時刻の調整範囲を整数で指定します。時刻単位に指定した単位によって指定範囲が異なります。時刻単位ごとの指定範囲を次の表に示します。

表 12-8　時刻単位と指定範囲の一覧

時刻単位	指定範囲
sec（秒）	0～59 の整数
msec（ミリ秒）	0～999 の整数
usec（マイクロ秒）	0～999 の整数

unuse

時刻調整をしません。

このパラメーターは，システムコンフィグプロパティファイル，クエリグループ用プロパティファイル，およびストリーム用プロパティファイルで指定できます。指定が重複した場合，または指定を省略した場合，次の優先順位で指定値が有効になります（1.> 2.> 3.）。

1. ストリーム用プロパティファイル
2. クエリグループ用プロパティファイル
3. システムコンフィグプロパティファイル

stream.timestampMode= {Server | DataSource}

タプルに時刻を付与するためのタイムスタンプモードをServer またはDataSource で指定します。大文字，小文字の区別はされません。デフォルトはServer です。

Server

サーバモードを使用します。

DataSource

データソースモードを使用します。

DataSource を指定した場合は，stream.timestampAccuracy パラメーター，および stream.timestampPosition パラメーターを必ず指定してください。

このパラメーターは，システムコンフィグプロパティファイル，およびクエリグループ用プロパティファイルで指定できます。指定が重複した場合，または指定を省略した場合，次の優先順位で指定値が有効になります（1.> 2.)。

1. クエリグループ用プロパティファイル
2. システムコンフィグプロパティファイル

stream.timestampPosition=*時刻データ列名*

タプル内の時刻データ列名または__systemtime__を指定します。大文字，小文字の区別はされません。

stream.timestampMode パラメーターでDataSource を指定した場合，このパラメーターを必ず指定してください。stream.timestampMode パラメーターでServer を指定した場合，このパラメーターの指定は無視されますが，パラメーターの形式チェックは実施されます。

時刻データとして指定できるデータ型は，TIMESTAMP 型だけです。

時刻データの範囲は，GMT（グリニッジ標準時）の 1970/01/01 00:00:00.000000000 から 2261/12/31 23:59:59.999999999 までです。それ以外の時刻を指定した場合は，ストリームデータの送信時に例外が発生します。日本標準時は GMT より 9 時間あとであることに注意してください。

このパラメーターは，システムコンフィグプロパティファイル，クエリグループ用プロパティファイル，およびストリーム用プロパティファイルで指定できます。指定が重複した場合，または指定を省略した場合，次の優先順位で指定値が有効になります（1.> 2.> 3.)。

1. ストリーム用プロパティファイル
2. クエリグループ用プロパティファイル
3. システムコンフィグプロパティファイル

stream.tupleLogMode= {true | false}

タイムスタンプモードがサーバモードの場合にhsdptplput コマンドを実行するかどうかをtrue またはfalse で指定します。大文字，小文字の区別はされません。デフォルトはfalse です。

true

hsdptplput コマンドを実行します。

false

hsdptplput コマンドを実行しません。

stream.timestampMode パラメーターでServer を，このパラメーターでtrue を指定した場合，データ送受信用 API のStreamInput インタフェースのput メソッドでストリームデータを送信する際，SDPClientException の例外が発生します。

このパラメーターは，システムコンフィグプロパティファイル，およびクエリグループ用プロパティファイルで指定できます。指定が重複した場合，または指定を省略した場合，次の優先順位で指定値が有効になります（1.> 2.)。

1. クエリグループ用プロパティファイル
2. システムコンフィグプロパティファイル

tpl.backupFileCount=タプルログファイルのバックアップを残す最大世代数

タプルログファイルのバックアップを残す最大世代数を0～10 の整数で指定します。0 を指定した場合，バックアップを取得しません。デフォルトは1 です。

このパラメーターは，システムコンフィグプロパティファイル，クエリグループ用プロパティファイル，およびストリーム用プロパティファイルで指定できます。指定が重複した場合，または指定を省略した場合，次の優先順位で指定値が有効になります（1.> 2.> 3.)。

1. ストリーム用プロパティファイル
2. クエリグループ用プロパティファイル
3. システムコンフィグプロパティファイル

tpl.bufferCount=タプルログのバッファーの面数

タプルログのバッファーの面数を3～512 の整数で指定します。デフォルトは5 です。

このパラメーターは，システムコンフィグプロパティファイル，クエリグループ用プロパティファイル，およびストリーム用プロパティファイルで指定できます。指定が重複した場合，または指定を省略した場合，次の優先順位で指定値が有効になります（1.> 2.> 3.)。

1. ストリーム用プロパティファイル
2. クエリグループ用プロパティファイル
3. システムコンフィグプロパティファイル

tpl.bufferSize=タプルログのバッファーの最大サイズ

タプルログのバッファーの最大サイズ（単位：キロバイト）を1～2048000 の整数で指定します。デフォルトは1024 です。

このパラメーターは，システムコンフィグプロパティファイル，クエリグループ用プロパティファイル，およびストリーム用プロパティファイルで指定できます。指定が重複した場合，または指定を省略した場合，次の優先順位で指定値が有効になります（1.> 2.> 3.)。

1. ストリーム用プロパティファイル
2. クエリグループ用プロパティファイル
3. システムコンフィグプロパティファイル

tpl.fileCount=タプルログファイルの最大ファイル数

タプルログファイルの最大ファイル数を3～512 の整数で指定します。デフォルトは3 です。

このパラメーターは，システムコンフィグプロパティファイル，クエリグループ用プロパティファイル，およびストリーム用プロパティファイルで指定できます。指定が重複した場合，または指定を省略した場合，次の優先順位で指定値が有効になります（1.＞ 2.＞ 3.)。

1. ストリーム用プロパティファイル
2. クエリグループ用プロパティファイル
3. システムコンフィグプロパティファイル

tpl.fileSize=タプルログファイルの最大サイズ

タプルログファイルの最大サイズ（単位：メガバイト）を1～2048 の整数で指定します。デフォルトは100 です。

このパラメーターは，システムコンフィグプロパティファイル，クエリグループ用プロパティファイル，およびストリーム用プロパティファイルで指定できます。指定が重複した場合，または指定を省略した場合，次の優先順位で指定値が有効になります（1.＞ 2.＞ 3.)。

1. ストリーム用プロパティファイル
2. クエリグループ用プロパティファイル
3. システムコンフィグプロパティファイル

tpl.outputLevel=タプルログの出力レベル

タプルログの出力レベルを1～3 の整数で指定します。デフォルトは3 です。

1
　ストリームキューへ格納するタプルについて，タプルログを出力します。

2
　時刻の逆転によって破棄するタプルについて，タプルログを出力します。

3
　レベル 1 とレベル 2 のタプルについて，タプルログを出力します。

このパラメーターは，システムコンフィグプロパティファイル，クエリグループ用プロパティファイル，およびストリーム用プロパティファイルで指定できます。指定が重複した場合，または指定を省略した場合，次の優先順位で指定値が有効になります（1.＞ 2.＞ 3.)。

1. ストリーム用プロパティファイル
2. クエリグループ用プロパティファイル
3. システムコンフィグプロパティファイル

tpl.outputTrigger= {未指定 | buffer | none}

タプルログファイルへの出力契機を次のどれかで指定します。大文字，小文字の区別はされません。デフォルトでは，値は指定されていません。

未指定
　入力ストリームに投入されたタプルだけを出力します。

buffer
　入力ストリームおよび出力ストリームに投入されたタプルのタプルログを出力します。

none
　タプルログを出力しません。

このパラメーターは，システムコンフィグプロパティファイル，クエリグループ用プロパティファイル，およびストリーム用プロパティファイルで指定できます。指定が重複した場合，または指定を省略した場合，次の優先順位で指定値が有効になります（1.> 2.> 3.)。

1. ストリーム用プロパティファイル
2. クエリグループ用プロパティファイル
3. システムコンフィグプロパティファイル

tpl.useOverwrite= {true | false}

タプルログのバッファーの枯渇時にバッファーを上書きするかどうかをtrue またはfalse で指定します。大文字，小文字の区別はされません。デフォルトはtrue です。

true
　タプルログのバッファーを上書きします。

false
　タプルログのバッファーを上書きしません。

このパラメーターは，システムコンフィグプロパティファイル，クエリグループ用プロパティファイル，およびストリーム用プロパティファイルで指定できます。指定が重複した場合，または指定を省略した場合，次の優先順位で指定値が有効になります（1.> 2.> 3.)。

1. ストリーム用プロパティファイル
2. クエリグループ用プロパティファイル
3. システムコンフィグプロパティファイル

12.9　クエリグループ用プロパティファイル

クエリグループ用プロパティファイルと，指定するパラメーターの詳細について説明します。

12.9.1　クエリグループ用プロパティファイルの詳細

記述形式

パラメーターは次の形式で記述します。

```
パラメーター名=値
```

- 値の後ろにスペースやコメントなどの文字列を追加できません。
  追加した場合は値の一部として解釈されます。
- パラメーター名だけを指定して，値を指定していない行は，無視されてデフォルト値が指定されたとして動作します。

ファイル名

クエリグループ名.qg

クエリグループ名は，英数字（0～9，a～z，A～Z）とアンダーライン（_）で，1～64 文字で指定してください。なお，クエリグループ名の先頭に使用できる文字は半角の英字だけです。

ファイルの格納先

このファイルは，必ず次のディレクトリに格納してください。

```
運用ディレクトリ/conf/
```

説明

クエリ定義ファイルのパスやクエリグループを実行するときのチューニングパラメーターを設定します。このファイルは，クエリグループごとに必ず作成します。

指定できるパラメーター

指定できるパラメーターと省略時の仮定値を次の表に示します。なお，パラメーターの詳細については，「12.9.2　クエリグループ用プロパティファイルのパラメーターの詳細」を参照してください。

表 12-9　指定できるパラメーターと省略時の仮定値（クエリグループ用プロパティファイル）

項番	パラメーター名	内容	省略時の仮定値	指定できる値の範囲
1	engine.externalStreamFuncVerifyMode	ストリーム間演算の戻り値の型を検証するかどうかを指定します。	true	• true • false
2	querygroup.cqlFilePath	クエリグループを定義するクエリ定義ファイルのパスを指定します。	なし	絶対パスまたは運用ディレクトリからの相対パス
3	querygroup.extFuncDefFilePath	外部定義関数定義ファイルのパスを指定します。	conf/xml/ExternalFunctionDefinition.xml	絶対パスまたは運用ディレクトリからの相対パス
4	querygroup.encoding	クエリグループごとにストリームデータの文字列を処理するためのエンコーディングを指定します。	UTF-8	• US-ASCII • UTF-8 • UTF-16 • UTF-16BE • UTF-16LE
5	stream.input.*ストリーム名*.dispatch.type	クエリグループをスケールアップ／スケールアウト構成で使う場合に，入力ストリームの分散方式を指定します。 入力ストリーム，*ストリーム名*に接続するアダプターは，ここで指定した分散方式で分散してデータを送信します。 省略した場合，クエリグループをスケールアップ／スケールアウト構成で使用できません。 • *ストリーム名* クエリ定義ファイルに定義した入力ストリーム名を指定します。 • 分散方式 次の分散方式から指定します。 ・roundrobin ラウンドロビンで，入力ストリームにデータを入力します。 ・hashing データのハッシュ値に基づき，入力ストリームを選択してデータを入力します。指定した入力ストリームに対し，stream.input.*ストリーム名*.dispatch.rule の設定が必要です。 ・all すべての入力ストリームに同じデータを入力します。 ・custom	なし	• roundrobin • hashing • all • custom

項番	パラメーター名	内容	省略時の仮定値	指定できる値の範囲
5	stream.input.*ストリーム名*.dispatch.type	データ送信側で決めたルールに従い，入力ストリームにデータを入力します。接続するアダプターが外部入力アダプターの場合だけ有効です。	なし	• roundrobin • hashing • all • custom
6	stream.input.*ストリーム名*.dispatch.rule	HASHING 分散に使う入力ストリームのカラム名一覧を，コンマでつないで指定します。 本プロパティは，stream.input.*ストリーム名*.dispatch.type で指定した値が，hashing の入力ストリームの場合に指定します。 *ストリーム名*には，クエリ定義ファイルに定義した入力ストリームを指定します。	なし	分散に使うデータ列名の一覧
7	stream.timestampPosition	タプルの時刻データ列名を指定します。 stream.timestampMode がDataSource かつ，入力ストリームがカスケーディングアダプターからのデータを受け付ける場合には，__systemtime__を指定します。	なし	時刻データ列名または __systemtime__
8	stream.output.*ストリーム名*.link	出力ストリーム，*ストリーム名*が接続する入力ストリームの情報を指定します。 複数入力ストリームに接続する場合には，コンマで区切り，複数の入力ストリームの情報を指定します。 本プロパティを指定すると，クエリグループ開始時に指定した出力ストリームと入力ストリームを接続する内部カスケーディングアダプターが起動します。 • *ストリーム名* クエリ定義ファイルに定義した出力ストリーム名を指定します。 • *BrokerAddr* 接続先入力ストリームのアドレス問い合わせ先 SDP ブローカーの IP アドレスまたはホスト名とポート番号を，「*IPアドレスまたはホスト名:ポート番号*」の形式で指定します。SDP ブローカーのポート番号については，「10.5 hsdpsetup」を参照してください。 • *ServerCluster* 接続先入力ストリームのクエリグループが属するサーバクラスタ名を指定します。 • *QueryGroup* 接続先入力ストリームが属するクエリグループ名を指定します。 • *InputStream* 接続先入力ストリーム名を指定します。	• *BrokerAddr* localhost • *ServerCluster* My_server_cluster	[/*BrokerAddr*/][*ServerCluster*/]*QueryGroup*/*InputStream*[,…]

項番	パラメーター名	内容	省略時の仮定値	指定できる値の範囲
9	querygroup.sleepOnOverStore	SDP サーバが出力ストリームキューに空きがあるかどうかをチェックして空きがなかった場合に，クエリグループの実行をスリープさせる時間（単位：ミリ秒）を指定します。	system_config.properties で指定した値※	1～2147483647の整数
10	querygroup.sleepOnOverStoreRetryCount	SDP サーバがクエリ実行後のタプルを出力ストリームキューへ投入する前に，出力ストリームキューに空きがあるかどうかをチェックする回数を指定します。	-1	-1～2147483647の整数
11	stream.filterCondition	タイムスタンプ調整機能でタプルのフィルタリングをする場合に，フィルタリングの条件演算式を指定します。	なし	「12.9.2　クエリグループ用プロパティファイルのパラメーターの詳細」の「stream.filterCondition=*条件演算式*」を参照してください。
12	stream.filterMode	タイムスタンプ調整機能でタプルのフィルタリングをするかどうかを指定します。 • unuse：タプルのフィルタリングをしません。 • condition：タプルのフィルタリングをします。	unuse	• unuse • condition
13	stream.freeInputQueueSizeThreshold	入力ストリームキューに登録する要素数の上限値に対する空きサイズのしきい値（単位：%）を指定します。	system_config.properties で指定した値※	1～99の整数
14	stream.freeInputQueueSizeThresholdOutputMessage	SDP サーバのメッセージログに，警告メッセージを出力するかどうかを指定します。 • true：警告メッセージを出力します。 • false：警告メッセージを出力しません。	system_config.properties で指定した値※	• true • false
15	stream.maxKeepTupleCount	タイムスタンプ調整機能で保留できるタプル数の上限値を指定します。	system_config.properties で指定した値※	1～1048576の整数
16	stream.propertyFiles	ストリームごとにプロパティを設定する場合に，ストリーム用プロパティファイルのファイル名を指定します。	なし	－
17	stream.timestampAccuracy	タイムスタンプ調整機能の時刻単位と時刻調整範囲を指定します。	system_config.properties で指定した値※	「12.9.2　クエリグループ用プロパティファイルのパラメーターの詳細」の「stream.timestampAccuracy={ {sec | msec | usec} ,*時刻*

項番	パラメーター名	内容	省略時の仮定値	指定できる値の範囲
17	stream.timestampAccuracy	タイムスタンプ調整機能の時刻単位と時刻調整範囲を指定します。	system_config.properties で指定した値※	調整範囲 | unuse} 」を参照してください。
18	stream.timestampMode	タプルに時刻を付与するためのタイムスタンプモードを指定します。 • Server：サーバモードを使用します。 • DataSource：データソースモードを使用します。	system_config.properties で指定した値※	• Server • DataSource
19	stream.timestampPosition	タプル内の時刻データ列名を指定します。	system_config.properties で指定した値※	−
20	stream.tupleLogMode	タイムスタンプモードがサーバモードの場合に hsdptplput コマンドを実行するかどうかを指定します。 true：hsdptplput コマンドを実行します。 false：hsdptplput コマンドを実行しません。	false	• true • false
21	tpl.backupFileCount	タプルログファイルのバックアップを残す最大世代数を指定します。 0 を指定した場合，バックアップを取得しません。	1	0～10 の整数
22	tpl.bufferCount	タプルログのバッファーの面数を指定します。	5	3～512 の整数
23	tpl.bufferSize	タプルログのバッファーの最大サイズ（単位：キロバイト）を指定します。	1024	1～2048000 の整数
24	tpl.fileCount	タプルログファイルの最大ファイル数を指定します。	3	3～512 の整数
25	tpl.fileSize	タプルログファイルの最大サイズ（単位：メガバイト）を指定します。	100	1～2048 の整数
26	tpl.outputLevel	タプルログの出力レベルを指定します。 1. ストリームキューへ格納するタプルについて，タプルログを出力します。 2. 時刻の逆転によって破棄するタプルについて，タプルログを出力します。 3. レベル 1 とレベル 2 のタプルについて，タプルログを出力します。	3	1～3 の整数
27	tpl.outputTrigger	タプルログファイルへの出力契機を指定します。 • 未指定：入力ストリームに投入されたタプルだけを出力します。 • buffer：入力ストリームおよび出力ストリームに投入されたタプルのタプルログを出力します。 • none：タプルログを出力しません。	未指定	• 未指定 • buffer • none

項番	パラメーター名	内容	省略時の仮定値	指定できる値の範囲
28	tpl.useOverwrite	タプルログのバッファーの枯渇時にバッファーを上書きするかどうかを指定します。 • true：タプルログのバッファーを上書きします。 • false：タプルログのバッファーを上書きしません。	true	• true • false

注※

システムコンフィグプロパティファイル（system_config.properties）で値が指定されていない場合は，システムコンフィグプロパティファイルのデフォルト値が適用されます。

注意事項

特になし。

12.9.2　クエリグループ用プロパティファイルのパラメーターの詳細

ここでは，「12.9.1　クエリグループ用プロパティファイルの詳細」の「指定できるパラメーター」で示したクエリグループ用プロパティファイルのパラメーターの詳細について説明します。

engine.externalStreamFuncVerifyMode=*ストリーム間演算の戻り値の型の検証*

ストリーム間演算の戻り値の型を検証するかどうかをtrue またはfalse で指定します。大文字，小文字の区別はされません。デフォルトはtrue です。

true

検証します。

false

検証しません。

false を指定することで，ストリーム間演算関数を頻繁に呼び出すシステムで性能が改善する場合があります。

> **メモ**
>
> 検証しない場合，トラブルシュートが困難になります。

このパラメーターは，システムコンフィグプロパティファイル，およびクエリグループ用プロパティファイルで指定できます。指定が重複した場合，または指定を省略した場合，次の優先順位で指定値が有効になります（1.> 2.）。

1. クエリグループ用プロパティファイル
2. システムコンフィグプロパティファイル

querygroup.cqlFilePath=*クエリ定義ファイルのパス*

クエリグループを定義するクエリ定義ファイルのパスを絶対パスまたは運用ディレクトリからの相対パスで指定します。このパラメーターの指定がない場合，クエリグループを登録できません。

querygroup.encoding=*エンコーディング*

クエリグループごとにストリームデータの文字列を処理するためのエンコーディングを指定します。ここで指定したエンコーディング値は，次の機能と関連しています。

- 外部アダプターの API でこのエンコーディング値を取得できます。
- アダプター構成定義ファイルのtcpcon:input タグのcharCode 属性で，このエンコーディング値が使用されます。
- カスケーディングアダプターのcascading タグのencoding 属性でこのエンコーディング値が使用されます。
- hsdpcqldebug コマンドで使用する入力データファイルおよび出力結果ファイルの文字コードに，このエンコーディング値が使用されます。

なお，同一サーバクラスタ内の，同一クエリグループでは，同じ値を指定する必要があります。

querygroup.sleepOnOverStore=*クエリグループの実行をスリープさせる時間*

SDP サーバが出力ストリームキューに空きがあるかどうかをチェックして空きがなかった場合に，クエリグループの実行をスリープさせる時間（単位：ミリ秒）を 1～2147483647 の整数で指定します。

このパラメーターの指定は，querygroup.sleepOnOverStoreRetryCount パラメーターで1 以上を指定した場合に有効です。

このパラメーターによってスリープしたクエリグループに対しては，次のことが実行できます。

- 入力アダプターによる入力ストリームキューへのタプルの投入
- 出力アダプターによる出力ストリームキューからのタプルの取得

このパラメーターは，システムコンフィグプロパティファイル，およびクエリグループ用プロパティファイルで指定できます。指定が重複した場合，または指定を省略した場合，次の優先順位で指定値が有効になります（1.> 2.)。

1. クエリグループ用プロパティファイル
2. システムコンフィグプロパティファイル

querygroup.sleepOnOverStoreRetryCount=*出力ストリームキューの空きをチェックする回数*

SDP サーバがクエリ実行後のタプルを出力ストリームキューへ投入する前に，出力ストリームキューに空きがあるかどうかをチェックする回数を-1～2147483647 の整数で指定します。デフォルトは-1 です。

1～2147483647 を指定した場合，SDP サーバは，指定された回数だけ次の処理を実行します。

1. 出力ストリームキューに空きがあるかどうかをチェックします。
2. 出力ストリームキューに空きがなければ，出力ストリームキューのあるクエリグループの実行をスリープさせます。スリープさせる時間は，querygroup.sleepOnOverStore パラメーターで指定します。

   スリープ後，このパラメーターで指定した回数分の処理を実行していない場合は，1.の処理に戻ります。
3. 出力ストリームキューに空きがあれば，出力ストリームキューへクエリ実行後のタプルを投入します。

0 を指定した場合，SDP サーバは，出力ストリームキューに空きがあるかをチェックしないで，クエリ実行後のタプルを出力ストリームキューへ投入します。

-1 を指定した場合，SDP サーバは，出力ストリームキューに空きがあるかをチェックします。空きがない場合，次の処理を出力ストリームキューに空きができるまで実行します。

1. 出力ストリームキューに空きがあるかどうかをチェックします。
2. 出力ストリームキューに空きがなければ，出力ストリームキューのあるクエリグループの実行をスリープさせます。スリープさせる時間は，querygroup.sleepOnOverStore パラメーターで指定します。スリープ後，空きがない場合は，1.の処理に戻ります。
3. 出力ストリームキューに空きがあれば，出力ストリームキューへクエリ実行後のタプルを投入します。

この指定値とquerygroup.sleepOnOverStore パラメーターの指定値の組み合わせによって，次の属性の値が指定されます。

- アダプター構成定義ファイルのtcpcon:input タグの@queueFullRetryInterval 属性
- アダプター構成定義ファイルのカスケーディングクライアントコネクター定義のcascading タグの@queueFullRetryInterval 属性

この指定値とquerygroup.sleepOnOverStore パラメーターの指定値との組み合わせと，それによって指定される@queueFullRetryInterval 属性の値を次の表に示します。

querygroup.sleepOnOverStoreRetryCount パラメーターの値	querygroup.sleepOnOverStore パラメーターの値	@queueFullRetryInterval 属性の値
0	• 1～2147483647 • 未指定	-1
• -1 • 1～2147483647 • 未指定	1～2147483647	querygroup.sleepOnOverStore パラメーターの値
	未指定	100（デフォルト値）

このパラメーターは，システムコンフィグプロパティファイル，およびクエリグループ用プロパティファイルで指定できます。指定が重複した場合，または指定を省略した場合，次の優先順位で指定値が有効になります（1.> 2.)。

1. クエリグループ用プロパティファイル
2. システムコンフィグプロパティファイル

stream.filterCondition=*条件演算式*

タイムスタンプ調整機能でタプルのフィルタリングをする場合に，フィルタリングの条件演算式を指定します。タプルのフィルタリングでは，条件演算式の列名に指定した列の内容と，定数に指定した内容に対して条件演算をします。条件演算の結果，一致する内容を持つタプルはタイムスタンプ調整機能で保留し，一致しないタプルは破棄します。

条件演算式の記述方法については，「12.9.3　条件演算式の記述規則」を参照してください。

stream.filterMode パラメーターでcondition を指定した場合，このパラメーターを必ず指定してください。stream.filterMode パラメーターでunuse を指定した場合，このパラメーターの指定は無視されますが，パラメーターの形式チェックは実施されます。

このパラメーターは，クエリグループ用プロパティファイル，およびストリーム用プロパティファイルで指定できます。指定が重複した場合，または指定を省略した場合，次の優先順位で指定値が有効になります（1.> 2.)。

1. ストリーム用プロパティファイル
2. クエリグループ用プロパティファイル

stream.filterMode= {unuse | condition}

タイムスタンプ調整機能でタプルのフィルタリングをするかどうかをunuse またはcondition で指定します。大文字，小文字の区別はされません。デフォルトはunuse です。

unuse

タプルのフィルタリングをしません。

condition

タプルのフィルタリングをします。

condition を指定した場合は，stream.filterCondition パラメーターを必ず指定してください。

stream.timestampAccuracy パラメーターでunuse を指定した場合，このパラメーターの指定は無視されますが，パラメーターの形式チェックは実施されます。

このパラメーターは，クエリグループ用プロパティファイル，およびストリーム用プロパティファイルで指定できます。指定が重複した場合，または指定を省略した場合，次の優先順位で指定値が有効になります（1.> 2.)。

1. ストリーム用プロパティファイル
2. クエリグループ用プロパティファイル

stream.freeInputQueueSizeThreshold=*入力ストリームキューの最大サイズに対する空きサイズのしきい値*

入力ストリームキューに登録する要素数の上限値（システムコンフィグプロパティファイルのengine.maxQueueSize パラメーターの指定値）に対する空きサイズのしきい値（単位：%）を1～99 の整数で指定します。

次に示す条件を満たした場合に，put(StreamTuple *tuple*)メソッドまたはput(ArrayList<StreamTuple> *tuple_list*)メソッドからSDPClientFreeInputQueueSizeThresholdOverException の例外がスローされます。このとき，入力ストリームキューへのタプルの投入は，成功しています。

このパラメーターの指定値 ≧（（*入力ストリームキューの空きサイズ*÷*入力ストリームキューの最大サイズ*）×100）

このパラメーターを省略した場合は，しきい値のチェックによる例外は発生しません。

このパラメーターは，システムコンフィグプロパティファイル，クエリグループ用プロパティファイル，およびストリーム用プロパティファイルで指定できます。指定が重複した場合，または指定を省略した場合，次の優先順位で指定値が有効になります（1.> 2.> 3.）。

1. ストリーム用プロパティファイル
2. クエリグループ用プロパティファイル
3. システムコンフィグプロパティファイル

stream.freeInputQueueSizeThresholdOutputMessage= {true | false}

SDP サーバのメッセージログに，警告メッセージ（メッセージKFSP42032-W）を出力するかどうかをtrue またはfalse で指定します。大文字，小文字の区別はされません。

true
　警告メッセージを出力します。

false
　警告メッセージを出力しません。

このパラメーターの指定は，stream.freeInputQueueSizeThreshold パラメーターが指定された場合だけ有効です。

メモ

なお，警告メッセージが出力されるのは，このパラメーターでtrue を指定し，次の条件を満たした場合です。

stream.freeInputQueueSizeThresholdパラメーターの指定値 ≧（（*入力ストリームキューの空きサイズ*÷*入力ストリームキューの最大サイズ*）×100）

このパラメーターは，システムコンフィグプロパティファイル，クエリグループ用プロパティファイル，およびストリーム用プロパティファイルで指定できます。指定が重複した場合，または指定を省略した場合，次の優先順位で指定値が有効になります（1.＞ 2.＞ 3.）。

1. ストリーム用プロパティファイル
2. クエリグループ用プロパティファイル
3. システムコンフィグプロパティファイル

stream.maxKeepTupleCount=*タイムスタンプ調整機能で保留できるタプル数の上限値*

タイムスタンプ調整機能で保留できるタプル数の上限値を1～1048576 の整数で指定します。

このパラメーターで指定したタプル数の上限値は，個々の入力ストリームのタイムスタンプ調整機能に対する上限値となります。

タプル数がこのパラメーターで指定した上限値を超えると，クエリグループが閉塞されます。

このパラメーターは，システムコンフィグプロパティファイル，クエリグループ用プロパティファイル，およびストリーム用プロパティファイルで指定できます。指定が重複した場合，または指定を省略した場合，次の優先順位で指定値が有効になります（1.＞ 2.＞ 3.）。

1. ストリーム用プロパティファイル
2. クエリグループ用プロパティファイル
3. システムコンフィグプロパティファイル

stream.propertyFiles=*ストリーム用プロパティファイル名*

ストリームごとにプロパティを設定する場合に，ストリーム用プロパティファイルのファイル名を指定します。

このパラメーターで指定するファイル名には，パス名を含めないでください。パス名を含めて指定した場合，パス名を含めて 1 つのファイル名として認識されます。

複数のファイル名を指定する場合は，半角コンマ（,）で区切って指定します。次の場合はエラーになります。

- 半角コンマの前後にファイル名が指定されていない場合
- 区切り文字の前後にスペース，タブなどを記述した場合
- 同じストリームのストリーム用プロパティファイルが複数個指定された場合

なお，クエリグループに存在しないストリームのストリーム用プロパティファイルが指定された場合，指定されたファイルは解析対象となりますが，ファイルの定義内容は無視されます。

stream.timestampAccuracy= { {sec | msec | usec} ,*時刻調整範囲* | unuse}

タイムスタンプ調整機能の時刻単位と時刻調整範囲を指定します。大文字，小文字の区別はされません。

stream.timestampMode パラメーターでDataSource を指定した場合，このパラメーターを必ず指定してください。stream.timestampMode パラメーターでServer を指定した場合，このパラメーターの指定は無視されますが，パラメーターの形式チェックは実施されます。

{sec | msec | usec} ,*時刻調整範囲*

時刻単位と時刻調整範囲を指定します。時刻単位（sec，msec，またはusec）と時刻調整範囲を区切る半角コンマ（,）の前後には，スペースやタブなどを記述しないでください。記述した場合はエラーとなります。時刻単位にsec，msec，またはusec のどれかを指定し，時刻調整範囲に0 を指定した場合は基準時刻だけが時刻調整範囲となります。

それぞれの値の意味を次に示します。

sec

時刻単位として秒を使用することを指定します。

msec

時刻単位としてミリ秒を使用することを指定します。

usec

時刻単位としてマイクロ秒を使用することを指定します。

時刻調整範囲

タイムスタンプ調整での時刻の調整範囲を整数で指定します。時刻単位に指定した単位によって指定範囲が異なります。時刻単位ごとの指定範囲を次の表に示します。

表 12-10　時刻単位と指定範囲の一覧

時刻単位	指定範囲
sec（秒）	0～59 の整数
msec（ミリ秒）	0～999 の整数
usec（マイクロ秒）	0～999 の整数

unuse

時刻調整をしません。

このパラメーターは，システムコンフィグプロパティファイル，クエリグループ用プロパティファイル，およびストリーム用プロパティファイルで指定できます。指定が重複した場合，または指定を省略した場合，次の優先順位で指定値が有効になります（1.> 2.> 3.）。

1. ストリーム用プロパティファイル
2. クエリグループ用プロパティファイル
3. システムコンフィグプロパティファイル

stream.timestampMode= {Server | DataSource}

タプルに時刻を付与するためのタイムスタンプモードをServer またはDataSource で指定します。大文字，小文字の区別はされません。

Server
: サーバモードを使用します。

DataSource
: データソースモードを使用します。

  DataSource を指定した場合は，stream.timestampAccuracy パラメーター，および stream.timestampPosition パラメーターを必ず指定してください。

このパラメーターは，システムコンフィグプロパティファイル，およびクエリグループ用プロパティファイルで指定できます。指定が重複した場合，または指定を省略した場合，次の優先順位で指定値が有効になります（1.> 2.)。

1. クエリグループ用プロパティファイル
2. システムコンフィグプロパティファイル

stream.timestampPosition=*時刻データ列名*

タプル内の時刻データ列名を指定します。大文字，小文字の区別はされません。

stream.timestampMode パラメーターでDataSource を指定した場合，このパラメーターを必ず指定してください。stream.timestampMode パラメーターでServer を指定した場合，このパラメーターの指定は無視されますが，パラメーターの形式チェックは実施されます。

時刻データとして指定できるデータ型は，TIMESTAMP 型だけです。

時刻データの範囲は，GMT（グリニッジ標準時）の1970/01/01 00:00:00.000000000 から2261/12/31 23:59:59.999999999 までです。それ以外の時刻を指定した場合は，ストリームデータ送信時に例外が発生します。日本標準時は GMT より9 時間あとであることに注意してください。

このパラメーターは，システムコンフィグプロパティファイル，クエリグループ用プロパティファイル，およびストリーム用プロパティファイルで指定できます。指定が重複した場合，または指定を省略した場合，次の優先順位で指定値が有効になります（1.> 2.> 3.)。

1. ストリーム用プロパティファイル
2. クエリグループ用プロパティファイル
3. システムコンフィグプロパティファイル

stream.tupleLogMode= {true | false}

タイムスタンプモードがサーバモードの場合にhsdptplput コマンドを実行するかどうかをtrue またはfalse で指定します。大文字，小文字の区別はされません。デフォルトはfalse です。

true

hsdptplput コマンドを実行します。

false

hsdptplput コマンドを実行しません。

stream.timestampMode パラメーターでServer を，このパラメーターでtrue を指定した場合，データ送受信用 API のStreamInput インタフェースのput メソッドでストリームデータを送信する際，SDPClientException の例外が発生します。

このパラメーターは，システムコンフィグプロパティファイル，およびクエリグループ用プロパティファイルで指定できます。指定が重複した場合，または指定を省略した場合，次の優先順位で指定値が有効になります（1.> 2.)。

1. クエリグループ用プロパティファイル
2. システムコンフィグプロパティファイル

tpl.backupFileCount=タプルログファイルのバックアップを残す最大世代数

タプルログファイルのバックアップを残す最大世代数を0～10 の整数で指定します。0 を指定した場合，バックアップを取得しません。デフォルトは1 です。

このパラメーターは，システムコンフィグプロパティファイル，クエリグループ用プロパティファイル，およびストリーム用プロパティファイルで指定できます。指定が重複した場合，または指定を省略した場合，次の優先順位で指定値が有効になります（1.> 2.> 3.)。

1. ストリーム用プロパティファイル
2. クエリグループ用プロパティファイル
3. システムコンフィグプロパティファイル

tpl.bufferCount=タプルログのバッファーの面数

タプルログのバッファーの面数を3～512 の整数で指定します。デフォルトは5 です。

このパラメーターは，システムコンフィグプロパティファイル，クエリグループ用プロパティファイル，およびストリーム用プロパティファイルで指定できます。指定が重複した場合，または指定を省略した場合，次の優先順位で指定値が有効になります（1.> 2.> 3.)。

1. ストリーム用プロパティファイル
2. クエリグループ用プロパティファイル

3. システムコンフィグプロパティファイル

tpl.bufferSize=タプルログのバッファーの最大サイズ

タプルログのバッファーの最大サイズ（単位：キロバイト）を1～2048000 の整数で指定します。デフォルトは1024 です。

このパラメーターは，システムコンフィグプロパティファイル，クエリグループ用プロパティファイル，およびストリーム用プロパティファイルで指定できます。指定が重複した場合，または指定を省略した場合，次の優先順位で指定値が有効になります（1.＞ 2.＞ 3.）。

1. ストリーム用プロパティファイル
2. クエリグループ用プロパティファイル
3. システムコンフィグプロパティファイル

tpl.fileCount=タプルログファイルの最大ファイル数

タプルログファイルの最大ファイル数を3～512 の整数で指定します。デフォルトは3 です。

このパラメーターは，システムコンフィグプロパティファイル，クエリグループ用プロパティファイル，およびストリーム用プロパティファイルで指定できます。指定が重複した場合，または指定を省略した場合，次の優先順位で指定値が有効になります（1.＞ 2.＞ 3.）。

1. ストリーム用プロパティファイル
2. クエリグループ用プロパティファイル
3. システムコンフィグプロパティファイル

tpl.fileSize=タプルログファイルの最大サイズ

タプルログファイルの最大サイズ（単位：メガバイト）を1～2048 の整数で指定します。デフォルトは100 です。

このパラメーターは，システムコンフィグプロパティファイル，クエリグループ用プロパティファイル，およびストリーム用プロパティファイルで指定できます。指定が重複した場合，または指定を省略した場合，次の優先順位で指定値が有効になります（1.＞ 2.＞ 3.）。

1. ストリーム用プロパティファイル
2. クエリグループ用プロパティファイル
3. システムコンフィグプロパティファイル

tpl.outputLevel=タプルログの出力レベル

タプルログの出力レベルを1～3 の整数で指定します。デフォルトは3 です。

1

ストリームキューへ格納するタプルについて，タプルログを出力します。

2

時刻の逆転によって破棄するタプルについて，タプルログを出力します。

3

レベル 1 とレベル 2 のタプルについて，タプルログを出力します。

このパラメーターは，システムコンフィグプロパティファイル，クエリグループ用プロパティファイル，およびストリーム用プロパティファイルで指定できます。指定が重複した場合，または指定を省略した場合，次の優先順位で指定値が有効になります（1.> 2.> 3.)。

1. ストリーム用プロパティファイル
2. クエリグループ用プロパティファイル
3. システムコンフィグプロパティファイル

tpl.outputTrigger= {未指定 | buffer | none}

タプルログファイルへの出力契機を次のどれかで指定します。大文字，小文字の区別はされません。デフォルトでは，値は指定されていません。

未指定

入力ストリームに投入されたタプルだけを出力します。

buffer

入力ストリームおよび出力ストリームに投入されたタプルのタプルログを出力します。

none

タプルログを出力しません。

このパラメーターは，システムコンフィグプロパティファイル，クエリグループ用プロパティファイル，およびストリーム用プロパティファイルで指定できます。指定が重複した場合，または指定を省略した場合，次の優先順位で指定値が有効になります（1.> 2.> 3.)。

1. ストリーム用プロパティファイル
2. クエリグループ用プロパティファイル
3. システムコンフィグプロパティファイル

tpl.useOverwrite= {true | false}

タプルログのバッファーの枯渇時にバッファーを上書きするかどうかをtrue またはfalse で指定します。大文字，小文字の区別はされません。デフォルトはtrue です。

true

タプルログのバッファーを上書きします。

false

タプルログのバッファーを上書きしません。

このパラメーターは，システムコンフィグプロパティファイル，クエリグループ用プロパティファイル，およびストリーム用プロパティファイルで指定できます。指定が重複した場合，または指定を省略した場合，次の優先順位で指定値が有効になります（1.> 2.> 3.)。

1. ストリーム用プロパティファイル
2. クエリグループ用プロパティファイル
3. システムコンフィグプロパティファイル

12.9.3 条件演算式の記述規則

ここでは，stream.filterCondition パラメーターで指定する条件演算式の記述規則について説明します。

条件演算式の形式

条件演算式は，次の形式で記述します。

```
'('列名 比較演算子 定数')'[ {AND | OR} '('列名 比較演算子 定数')'
                          {AND | OR} '('列名 比較演算子 定数')'…]
```

複数の比較演算を組み合わせる場合は，演算式をAND またはOR でつないで指定します。指定できる演算式の数は 1～10 個です。

条件演算式に指定できる要素について説明します。

列名

比較演算対象の列名を指定します。列名はストリームの定義（register stream 句）のスキーマ指定文字列で指定した列名を指定します。

比較演算子

条件の判定をするための演算子を指定します。指定できる比較演算子の種類およびその意味を次の表に示します。

表 12-11 指定できる比較演算子

比較演算子	比較演算子の使用例	使用例の意味
<=	A <= B	A はB 以下
>=	A >= B	A はB 以上
<	A < B	A はB より小さい
>	A > B	A はB より大きい

比較演算子	比較演算子の使用例	使用例の意味
=	A = B	A はB と等しい
!=	A != B	A はB と等しくない

定数

比較演算を実施する対象の値を整定数または文字列定数で指定します。

- 整定数

  比較演算を数値で実施する場合に指定します。数値として指定できる範囲は，-9223372036854775808～9223372036854775807 の整数になります。

- 文字列定数

  比較演算を文字列で実施する場合に指定します。文字列を指定する場合はアポストロフィ（'）で囲んで記述してください。文字列として指定できるのは半角文字，および全角文字です。指定できる文字数は，100 文字以内です。

条件演算式の記述例

条件演算式の記述例を次に示します。

```
stream.filterCondition=(xxx='abc')AND(zzz>18)AND(zzz<60)
```

この例では，列名 *xxx* がabc で，列名 *zzz* が 18 より大きく 60 より小さいタプルだけを保留します。

条件演算式の注意事項

- 演算式内にAND とOR を混在させることはできません。
- AND とOR，および列名で，大文字，小文字の区別はされません。
- 列名に指定したデータ型が文字データ（CHAR，VARCHAR）の場合，比較演算子に「=」と「!=」以外は指定できません。
- 整定数を指定する場合，列名に指定できるデータ型は CQL で使用できる次のデータ型になります。

  INT

  SMALLINT

  TINYINT

  BIGINT

  DEC

  NUMERIC

  REAL

  FLOAT

  DOUBLE

- 整定数を使用する場合，列名に指定したデータ型と整定数に指定した数値の範囲はチェックされません。
- 文字列定数では，文字列をアポストロフィ（'）で囲んで記述しますが，アポストロフィ（'）を文字列として使用したい場合は，1 個のアポストロフィを表すために 2 個のアポストロフィを記述してください。例えば，abc'abc と指定したい場合には，'abc''abc'と記述してください。
- 文字列定数に指定した文字列より大きい文字列が，該当の列に設定されていた場合は，先頭から指定した文字数までを演算の対象とします。例えば，文字列定数に指定した文字列が 'abc'（文字数が 3）で，該当の列に設定されていた文字列が 'abcde'の場合，先頭からの 'abc'までが演算の対象となることから，文字列定数 'abc'と合致します。
- 文字列定数では，空文字（文字列の先頭と最終を表すアポストロフィ（'）の間に何も指定しない）の指定はできません。条件式を囲む括弧，比較演算子，AND，およびOR の前後には，スペースまたはタブを記述できますが，スペースまたはタブは無視されます。ただし，最後の演算式の閉じ括弧の後ろには，スペースおよびタブは記述できません。
- 文字列定数を指定する場合，列名に指定できるデータ型は CQL で使用できる次のデータ型になります。

  CHAR

  VARCHAR

CQL で使用できるデータ型については，マニュアル『Hitachi Streaming Data Platform アプリケーション開発ガイド』を参照してください。

12.10　ストリーム用プロパティファイル

ここでは，ストリーム用プロパティファイルについて，ファイルの詳細と，ファイルに指定するパラメーターの詳細に分けて説明します。

12.10.1　ストリーム用プロパティファイルの詳細

記述形式

パラメーターは次の形式で記述します。

```
パラメーター名=値
```

- 値の後ろにスペースやコメントなどの文字列を追加できません。
  追加した場合は値の一部として解釈されます。
- パラメーター名だけを指定して，値を指定していない行は，無視されます。

ファイル名

ファイル名は任意です。このファイル名は，クエリグループ用プロパティファイルの`stream.propertyFiles`パラメーターで指定します。

ファイルの格納先

このファイルは，必ず次のディレクトリに格納してください。

```
運用ディレクトリ/conf/
```

説明

ストリーム単位でチューニングパラメーターを設定します。このファイルは，クエリグループ内のストリーム単位でチューニングパラメーターを設定したい場合だけ，ストリームごとに作成します。

指定できるパラメーター

指定できるパラメーターと省略時の仮定値を次の表に示します。なお，パラメーターの詳細については，「12.10.2　ストリーム用プロパティファイルのパラメーターの詳細」を参照してください。

表 12-12　指定できるパラメーターと省略時の仮定値（ストリーム用プロパティファイル）

項番	パラメーター名	内容	省略時の仮定値	指定できる値の範囲
1	stream.filterCondition	タイムスタンプ調整機能でタプルのフィルタリングをする場合に，フィルタリングの条件演算式を指定します。	クエリグループ用プロパティファイルで指定した値※	－
2	stream.filterMode	タイムスタンプ調整機能でタプルのフィルタリングをするかどうかを指定します。 • unuse：タプルのフィルタリングをしません。 • condition：タプルのフィルタリングをします。	クエリグループ用プロパティファイルで指定した値※	• unuse • condition
3	stream.freeInputQueueSizeThreshold	入力ストリームキューに登録する要素数の上限値に対する空きサイズのしきい値（単位：%）を指定します。	クエリグループ用プロパティファイルで指定した値※	1～99 の整数
4	stream.freeInputQueueSizeThresholdOutputMessage	SDP サーバのメッセージログに，警告メッセージを出力するかどうかを指定します。 • true：警告メッセージを出力します。 • false：警告メッセージを出力しません。	クエリグループ用プロパティファイルで指定した値※	• true • false
5	stream.maxKeepTupleCount	タイムスタンプ調整機能で保留できるタプル数の上限値を指定します。	クエリグループ用プロパティファイルで指定した値※	1～1048576 の整数
6	stream.streamName	ストリームの名称を指定します。	なし	－
7	stream.timestampAccuracy	タイムスタンプ調整機能の時刻単位と時刻調整範囲を指定します。	クエリグループ用プロパティファイルで指定した値※	「12.10.2　ストリーム用プロパティファイルのパラメーターの詳細」の「stream.timestampAccuracy= { {sec | msec | usec} ,*時刻調整範囲* | unuse} 」を参照してください。
8	stream.timestampPosition	タプル内の時刻データ列名または__systemtime__を指定します。	クエリグループ用プロパティファイルで指定した値※	「12.10.2　ストリーム用プロパティファイルのパラメーターの詳細」の「stream.timestampPosition=*時刻データ列名*」を参照してください。
9	tpl.backupFileCount	タプルログファイルのバックアップを残す最大世代数を指定します。 0 を指定した場合，バックアップを取得しません。	クエリグループ用プロパティ	0～10 の整数

項番	パラメーター名	内容	省略時の仮定値	指定できる値の範囲
9	tpl.backupFileCount	タプルログファイルのバックアップを残す最大世代数を指定します。 0 を指定した場合，バックアップを取得しません。	ファイルで指定した値※	0～10 の整数
10	tpl.bufferCount	タプルログのバッファーの面数を指定します。	クエリグループ用プロパティファイルで指定した値※	3～512 の整数
11	tpl.bufferSize	タプルログのバッファーの最大サイズ（単位：キロバイト）を指定します。	クエリグループ用プロパティファイルで指定した値※	1～2048000 の整数
12	tpl.fileCount	タプルログファイルの最大ファイル数を指定します。	クエリグループ用プロパティファイルで指定した値※	3～512 の整数
13	tpl.fileSize	タプルログファイルの最大サイズ（単位：メガバイト）を指定します。	クエリグループ用プロパティファイルで指定した値※	1～2048 の整数
14	tpl.outputLevel	タプルログの出力レベルを指定します。 1. ストリームキューへ格納するタプルについて，タプルログを出力します。 2. 時刻の逆転によって破棄するタプルについて，タプルログを出力します。 3. レベル 1 とレベル 2 のタプルについて，タプルログを出力します。	クエリグループ用プロパティファイルで指定した値※	1～3 の整数
15	tpl.outputTrigger	タプルログファイルへの出力契機を指定します。 • 未指定：対象のストリームが入力ストリームの場合，対象の入力ストリームに投入されたタプルのタプルログを出力します。対象のストリームが出力ストリームの場合，タプルログを出力しません。 • buffer：対象のストリームに投入されたタプルのタプルログを出力します。 • none：タプルログを出力しません。	クエリグループ用プロパティファイルで指定した値※	• 未指定 • buffer • none
16	tpl.useOverwrite	タプルログのバッファーの枯渇時にバッファーを上書きするかどうかを指定します。 • true：タプルログのバッファーを上書きします。 • false：タプルログのバッファーを上書きしません。	クエリグループ用プロパティファイルで指定した値※	• true • false

（凡例）

－：該当しません。

注※

クエリグループ用プロパティファイルで値が指定されていない場合は，クエリグループ用プロパティファイルのデフォルト値が適用されます。

12.10.2　ストリーム用プロパティファイルのパラメーターの詳細

ここでは，「12.10.1　ストリーム用プロパティファイルの詳細」の「指定できるパラメーター」で示したストリーム用プロパティファイルのパラメーターの詳細について説明します。

stream.filterCondition=*条件演算式*

タイムスタンプ調整機能でタプルのフィルタリングをする場合に，フィルタリングの条件演算式を指定します。タプルのフィルタリングでは，条件演算式の列名に指定した列の内容と，定数に指定した内容に対して条件演算をします。条件演算の結果，一致する内容を持つタプルはタイムスタンプ調整機能で保留し，一致しないタプルは破棄します。

条件演算式の記述方法については，「12.9.3　条件演算式の記述規則」を参照してください。

stream.filterMode パラメーターでcondition を指定した場合，このパラメーターを必ず指定してください。stream.filterMode パラメーターでunuse を指定した場合，このパラメーターの指定は無視されますが，パラメーターの形式チェックは実施されます。

このパラメーターは，クエリグループ用プロパティファイル，およびストリーム用プロパティファイルで指定できます。指定が重複した場合，または指定を省略した場合，次の優先順位で指定値が有効になります（1.＞ 2.）。

1. ストリーム用プロパティファイル
2. クエリグループ用プロパティファイル

stream.filterMode= {unuse | condition}

タイムスタンプ調整機能でタプルのフィルタリングをするかどうかをunuse またはcondition で指定します。大文字，小文字の区別はされません。

unuse

タプルのフィルタリングをしません。

condition

タプルのフィルタリングをします。

condition を指定した場合は，stream.filterCondition パラメーターを必ず指定してください。

stream.timestampAccuracy パラメーターでunuse を指定した場合，このパラメーターの指定は無視されますが，パラメーターの形式チェックは実施されます。

このパラメーターは，クエリグループ用プロパティファイル，およびストリーム用プロパティファイルで指定できます。指定が重複した場合，または指定を省略した場合，次の優先順位で指定値が有効になります（1.> 2.)。

1. ストリーム用プロパティファイル
2. クエリグループ用プロパティファイル

stream.freeInputQueueSizeThreshold=*入力ストリームキューの最大サイズに対する空きサイズのしきい値*

入力ストリームキューに登録する要素数の上限値（システムコンフィグプロパティファイルの engine.maxQueueSize パラメーターの指定値）に対する空きサイズのしきい値（単位：%）を1～99 の整数で指定します。

次に示す条件を満たした場合に，put(StreamTuple *tuple*)メソッドまたはput(ArrayList<StreamTuple> *tuple_list*)メソッドからSDPClientFreeInputQueueSizeThresholdOverException の例外がスローされます。このとき，入力ストリームキューへのタプルの投入は，成功しています。

このパラメーターの指定値 ≧（（*入力ストリームキューの空きサイズ*÷*入力ストリームキューの最大サイズ*）×100）

このパラメーターを省略した場合は，しきい値のチェックによる例外は発生しません。

このパラメーターは，システムコンフィグプロパティファイル，クエリグループ用プロパティファイル，およびストリーム用プロパティファイルで指定できます。指定が重複した場合，または指定を省略した場合，次の優先順位で指定値が有効になります（1.> 2.> 3.)。

1. ストリーム用プロパティファイル
2. クエリグループ用プロパティファイル
3. システムコンフィグプロパティファイル

stream.freeInputQueueSizeThresholdOutputMessage= {true | false}

SDP サーバのメッセージログに，警告メッセージ（メッセージKFSP42032-W）を出力するかどうかをtrue またはfalse で指定します。大文字，小文字の区別はされません。

true

警告メッセージを出力します。

false

警告メッセージを出力しません。

このパラメーターの指定は，stream.freeInputQueueSizeThreshold パラメーターが指定された場合だけ有効です。

メモ

警告メッセージは，このパラメーターでtrue を指定し，次の条件を満たした場合に表示されます。

stream.freeInputQueueSizeThresholdパラメーターの指定値 ≧（（*入力ストリームキューの空きサイズ÷入力ストリームキューの最大サイズ*）×100）

このパラメーターは，システムコンフィグプロパティファイル，クエリグループ用プロパティファイル，およびストリーム用プロパティファイルで指定できます。指定が重複した場合，または指定を省略した場合，次の優先順位で指定値が有効になります（1.＞ 2.＞ 3.）。

1. ストリーム用プロパティファイル
2. クエリグループ用プロパティファイル
3. システムコンフィグプロパティファイル

stream.maxKeepTupleCount=*タイムスタンプ調整機能で保留できるタプル数の上限値*

タイムスタンプ調整機能で保留できるタプル数の上限値を1～1048576 の整数で指定します。

このパラメーターで指定したタプル数の上限値は，個々の入力ストリームのタイムスタンプ調整機能に対する上限値となります。

タプル数がこのパラメーターで指定した上限値を超えると，クエリグループが閉塞されます。

このパラメーターは，システムコンフィグプロパティファイル，クエリグループ用プロパティファイル，およびストリーム用プロパティファイルで指定できます。指定が重複した場合，または指定を省略した場合，次の優先順位で指定値が有効になります（1.＞ 2.＞ 3.）。

1. ストリーム用プロパティファイル
2. クエリグループ用プロパティファイル
3. システムコンフィグプロパティファイル

stream.streamName=*ストリームの名称*

このプロパティファイルの定義内容を適用するストリームの名称を指定します。

指定したストリームの名称は，すべて大文字として扱われます。このパラメーターの指定がない場合は，エラーになります。

stream.timestampAccuracy= { {sec | msec | usec} ,*時刻調整範囲* | unuse}

タイムスタンプ調整機能の時刻単位と時刻調整範囲を指定します。大文字，小文字の区別はされません。

{sec | msec | usec} ,*時刻調整範囲*

時刻単位と時刻調整範囲を指定します。時刻単位（sec，msec，またはusec）と時刻調整範囲を区切る半角コンマ（,）の前後には，スペースやタブなどを記述しないでください。記述した場合はエラーとなります。時刻単位にsec，msec，またはusec のどれかを指定し，時刻調整範囲に0 を指定した場合は基準時刻だけが時刻調整範囲となります。

それぞれの値の意味を次に示します。

sec

時刻単位として秒を使用することを指定します。

msec

時刻単位としてミリ秒を使用することを指定します。

usec

時刻単位としてマイクロ秒を使用することを指定します。

時刻調整範囲

タイムスタンプ調整での時刻の調整範囲を整数で指定します。時刻単位に指定した単位によって指定範囲が異なります。時刻単位ごとの指定範囲を次の表に示します。

表 12-13　時刻単位と指定範囲の一覧

時刻単位	指定範囲
sec（秒）	0～59 の整数
msec（ミリ秒）	0～999 の整数
usec（マイクロ秒）	0～999 の整数

unuse

時刻調整をしません。

このパラメーターは，システムコンフィグプロパティファイル，クエリグループ用プロパティファイル，およびストリーム用プロパティファイルで指定できます。指定が重複した場合，または指定を省略した場合，次の優先順位で指定値が有効になります（1.＞ 2.＞ 3.）。

1. ストリーム用プロパティファイル
2. クエリグループ用プロパティファイル
3. システムコンフィグプロパティファイル

stream.timestampPosition=*時刻データ列名*

タプル内の時刻データ列名を指定します。大文字，小文字の区別はされません。

または，stream.timestampMode がDataSource かつ，入力ストリームがカスケーディングアダプターからのデータを受け付ける場合には，__systemtime__を指定します。

時刻データとして指定できるデータ型は，TIMESTAMP 型だけです。

時刻データの範囲は，GMT（グリニッジ標準時）の1970/01/01 00:00:00.000000000 から2261/12/31 23:59:59.999999999 までです。それ以外の時刻を指定した場合は，ストリームデータ送信時に例外が発生します。日本標準時は GMT より 9 時間あとであることに注意してください。

このパラメーターは，システムコンフィグプロパティファイル，クエリグループ用プロパティファイル，およびストリーム用プロパティファイルで指定できます。指定が重複した場合，または指定を省略した場合，次の優先順位で指定値が有効になります（1.> 2.> 3.)。

1. ストリーム用プロパティファイル
2. クエリグループ用プロパティファイル
3. システムコンフィグプロパティファイル

tpl.backupFileCount=タプルログファイルのバックアップを残す最大世代数

タプルログファイルのバックアップを残す最大世代数を0～10 の整数で指定します。0 を指定した場合，バックアップを取得しません。デフォルトは1 です。

このパラメーターは，システムコンフィグプロパティファイル，クエリグループ用プロパティファイル，およびストリーム用プロパティファイルで指定できます。指定が重複した場合，または指定を省略した場合，次の優先順位で指定値が有効になります（1.> 2.> 3.)。

1. ストリーム用プロパティファイル
2. クエリグループ用プロパティファイル
3. システムコンフィグプロパティファイル

tpl.bufferCount=タプルログのバッファーの面数

タプルログのバッファーの面数を3～512 の整数で指定します。デフォルトは5 です。

このパラメーターは，システムコンフィグプロパティファイル，クエリグループ用プロパティファイル，およびストリーム用プロパティファイルで指定できます。指定が重複した場合，または指定を省略した場合，次の優先順位で指定値が有効になります（1.> 2.> 3.)。

1. ストリーム用プロパティファイル
2. クエリグループ用プロパティファイル
3. システムコンフィグプロパティファイル

tpl.bufferSize=タプルログのバッファーの最大サイズ

タプルログのバッファーの最大サイズ（単位：キロバイト）を1～2048000 の整数で指定します。デフォルトは1024 です。

このパラメーターは，システムコンフィグプロパティファイル，クエリグループ用プロパティファイル，およびストリーム用プロパティファイルで指定できます。指定が重複した場合，または指定を省略した場合，次の優先順位で指定値が有効になります（1.＞ 2.＞ 3.)。

1. ストリーム用プロパティファイル
2. クエリグループ用プロパティファイル
3. システムコンフィグプロパティファイル

tpl.fileCount=タプルログファイルの最大ファイル数

タプルログファイルの最大ファイル数を3～512 の整数で指定します。デフォルトは3 です。

このパラメーターは，システムコンフィグプロパティファイル，クエリグループ用プロパティファイル，およびストリーム用プロパティファイルで指定できます。指定が重複した場合，または指定を省略した場合，次の優先順位で指定値が有効になります（1.＞ 2.＞ 3.)。

1. ストリーム用プロパティファイル
2. クエリグループ用プロパティファイル
3. システムコンフィグプロパティファイル

tpl.fileSize=タプルログファイルの最大サイズ

タプルログファイルの最大サイズ（単位：メガバイト）を1～2048 の整数で指定します。デフォルトは100 です。

このパラメーターは，システムコンフィグプロパティファイル，クエリグループ用プロパティファイル，およびストリーム用プロパティファイルで指定できます。指定が重複した場合，または指定を省略した場合，次の優先順位で指定値が有効になります（1.＞ 2.＞ 3.)。

1. ストリーム用プロパティファイル
2. クエリグループ用プロパティファイル
3. システムコンフィグプロパティファイル

tpl.outputLevel=タプルログの出力レベル

タプルログの出力レベルを1～3 の整数で指定します。デフォルトは3 です。

1

ストリームキューへ格納するタプルについて，タプルログを出力します。

2

時刻の逆転によって破棄するタプルについて，タプルログを出力します。

3

レベル 1 とレベル 2 のタプルについて，タプルログを出力します。

このパラメーターは，システムコンフィグプロパティファイル，クエリグループ用プロパティファイル，およびストリーム用プロパティファイルで指定できます。指定が重複した場合，または指定を省略した場合，次の優先順位で指定値が有効になります（1.> 2.> 3.)。

1. ストリーム用プロパティファイル
2. クエリグループ用プロパティファイル
3. システムコンフィグプロパティファイル

tpl.outputTrigger= {未指定 | buffer | none}

タプルログファイルへの出力契機を次のどれかで指定します。大文字，小文字の区別はされません。デフォルトでは，値は指定されていません。

未指定

対象のストリームが入力ストリームの場合，対象の入力ストリームに投入されたタプルのタプルログを出力します。対象のストリームが出力ストリームの場合，タプルログを出力しません。

buffer

対象のストリームに投入されたタプルのタプルログを出力します。

none

タプルログを出力しません。

このパラメーターは，システムコンフィグプロパティファイル，クエリグループ用プロパティファイル，およびストリーム用プロパティファイルで指定できます。指定が重複した場合，または指定を省略した場合，次の優先順位で指定値が有効になります（1.> 2.> 3.)。

1. ストリーム用プロパティファイル
2. クエリグループ用プロパティファイル
3. システムコンフィグプロパティファイル

tpl.useOverwrite= {true | false}

タプルログのバッファーの枯渇時にバッファーを上書きするかどうかをtrue またはfalse で指定します。大文字，小文字の区別はされません。デフォルトはtrue です。

true

タプルログのバッファーを上書きします。

`false`

タプルログのバッファーを上書きしません。

このパラメーターは，システムコンフィグプロパティファイル，クエリグループ用プロパティファイル，およびストリーム用プロパティファイルで指定できます。指定が重複した場合，または指定を省略した場合，次の優先順位で指定値が有効になります（1.> 2.> 3.)。

1. ストリーム用プロパティファイル
2. クエリグループ用プロパティファイル
3. システムコンフィグプロパティファイル

12.11　インプロセス連携用プロパティファイル

記述形式

パラメーターは次の形式で記述します。

パラメーター名=値

- 値の後ろにスペースやコメントなどの文字列を追加できません。
  追加した場合は値の一部として解釈されます。
- パラメーター名だけを指定して，値を指定していない行は，無視されます。

ファイル名

user_app.*アダプターグループ名またはアダプター名*.properties

標準提供アダプターの場合はアダプターグループ名を，カスタムアダプターの場合はアダプター名を，英数字（0～9，a～z，A～Z）とアンダーライン（_）で指定します。指定できる文字数は，1～32 文字です。

メモ

インプロセス連携をするアダプターグループ名またはアダプター名の先頭に使用できる文字は半角の英字だけです。

ファイルの格納先

このファイルは，必ず次のディレクトリに格納してください。

運用ディレクトリ/conf/

説明

インプロセス連携をするアダプターのクラス名やjar ファイルのパスを設定します。標準提供アダプターはアダプターグループごと，カスタムアダプターはアダプターごとに作成します。

指定できるパラメーター

指定できるパラメーターとデフォルト値を次の表に示します。

表 12-14　指定できるパラメーターとデフォルト値（インプロセス連携アダプター用プロパティファイル）

項番	パラメーター名	内容	デフォルト値
1	user_app.classname	インプロセスで連携するアダプターグループまたはアダプターのメインクラス名を指定します。※1 このクラスは，システムが提供するinterface クラスを実装したクラスとなります。 インプロセスで連携する場合，このパラメーターの指定がないと，エラーとなります。	なし
2	user_app.classpath_dir	user_app.classname パラメーターで指定されたクラスに対するクラスファイルの格納ディレクトリのパス，またはjar ファイルのパスを指定します。※1 ※2 パスは，絶対パスまたは運用ディレクトリからの相対パスで指定します。 このパラメーターに指定できるパスは 1 つだけです。 インプロセスで連携する場合，このパラメーターの指定がないと，エラーとなります。	なし

注※1

標準提供アダプターの場合は，指定値は固定です。次の値を指定してください。

user_app.classname=jp.co.Hitachi.soft.sdp.adaptor.AdaptorManager

user_app.classpath_dir=*インストールディレクトリ*/system/sdp/lib/sdp.jar

注※2

SDP サーバ用 JavaVM オプションファイルのSDP_CLASS_PATH パラメーターで指定したパスにjar ファイルが存在する場合，SDP_CLASS_PATH パラメーターで指定したパスにあるjar ファイルが優先されます。

ユーザーが作成したインプロセス連携アダプターから，このパラメーターに指定したクラスパス以外のパスを参照する場合は，SDP サーバ用 JavaVM オプションファイルのSDP_CLASS_PATH パラメーターにパスを指定してください。なお，SDP サーバ用 JavaVM オプションファイルのSDP_CLASS_PATH パラメーターに指定したパスは，SDP サーバの起動時だけ読み込まれます。そのため，SDP サーバの起動後にパスの値やjar ファイルの内容を変更しても反映されません。

12.12 SDP サーバのログファイル出力用プロパティファイル（logger.properties）

記述形式

パラメーターは次の形式で記述します。

パラメーター名=値

- 値の後ろにスペースやコメントなどの文字列を追加できません。
  追加した場合は値の一部として解釈されます。
- パラメーター名だけを指定して，値を指定していない行は，無視されます。

ファイル名

logger.properties

ファイルの格納先

このファイルは，必ず次のディレクトリに格納してください。

運用ディレクトリ/conf/

説明

このファイルには，SDP サーバのログファイルに関連するパラメーターを指定します。メッセージログやトレースログの最大面数，および最大サイズを設定します。このファイルは，運用ディレクトリごとに作成します。作成しない場合には，デフォルト値で動作します。

指定できるパラメーター

指定できるパラメーターとデフォルト値を次の表に示します。

表 12-15 指定できるパラメーターとデフォルト値（logger.properties）

項番	パラメーター名	内容	デフォルト値	指定できる値の範囲
1	logger.serverMessageFileCount	メッセージログファイルの最大面数を指定します。	3	2～16 の整数
2	logger.serverMessageMaxFileSize	メッセージログファイルの最大サイズ（単位：バイト）を指定します。	1048576	4096～2147483647 の整数
3	logger.serverTraceFileCount	トレースログファイルの最大面数を指定します。	3	2～16 の整数

項番	パラメーター名	内容	デフォルト値	指定できる値の範囲
4	logger.serverTraceMaxFileSize	トレースログファイルの最大サイズ（単位：バイト）を指定します。	1048576	4096～2147483647 の整数
5	logger.logfileDir	次に示すログファイルの格納先を指定します。 • SDP サーバ用ログファイル • 内部アダプター用ログファイル • 内部カスタムアダプター用ログファイル ディレクトリを指定する場合は絶対パスで指定してください。	*運用ディレクトリ*/logs	1～255 文字のディレクトリの絶対パス 指定できる文字は次のとおりです。 • 半角英数字 • アンダーライン（_） • スラッシュ（/）

注意事項

- SDP サーバ起動時にこのファイルが存在しない場合は，すべてのパラメーターがデフォルト値で起動します。
- パラメーターに不正な値（空文字を含む）を指定した場合，またはパラメーターを省略した場合は，デフォルト値で動作します。
- パラメーターを変更した場合は，SDP サーバを停止したあと，ログ出力ディレクトリ内のすべてのファイルとサブディレクトリを別のディレクトリに移動するか，削除する必要があります。
- logger.logfileDir パラメーターにパスを指定する場合は，ローカルファイルシステムのパスを指定してください。ネットワークドライブをマウントしたパスを指定したときのログ出力の動作は保証されません。
- logger.logfileDir パラメーターに指定したパスが存在しない場合は，自動的にディレクトリが作成されます。
- logger.logfileDir パラメーターに指定したパスへのログの出力に失敗した場合は，ログファイルにログは出力されません。
- logger.logfileDir パラメーターにログファイルの格納先を指定できるのは，Java エンジンの場合だけです。acceleration CQL エンジンの場合はログファイルの格納先ディレクトリを指定できません。
- 複数の運用ディレクトリ下にあるログファイルの格納先ディレクトリを変更する場合，それぞれ同一のディレクトリに変更しないでください。複数のプロセスから，同じファイルへ書き込みを実行することになり，ログファイルが壊れるおそれがあります。

12.13 共通コンポーネント（コマンド・ロガー）のログファイル出力用プロパティファイル（hsdplogger.properties）

記述形式

次の形式で各パラメーターを指定します。

パラメーター名=値

ファイル名

hsdplogger.properties

ファイル格納場所

このファイルは，必ず次のディレクトリに格納してください。

運用ディレクトリ/conf/

説明

このファイルには，共通コンポーネントのログファイルに関連するパラメーターを指定します。

指定できるパラメーター

指定できるパラメーターとそのデフォルト値を次の表に示します。

表 12-16 指定できるパラメーターと対応するデフォルト値（hsdplogger.properties）

パラメーター名	説明	デフォルト値	許容される値の範囲
commandMessageFileCount	コマンドメッセージログファイルの最大数を指定します。	4	2～16 の整数
commandMessageMaxFileSize	コマンドメッセージログファイルの最大サイズ（単位：バイト）を指定します。	16777216	4096～16777216 の整数
commandMessageLogLevel	コマンドメッセージログファイルのログレベルを指定します。 2 を指定すると，より詳細なメッセージが出力されます。	1	1 | 2
commandTraceFileCount	コマンドトレースログファイルの最大数を指定します。	4	2～16 の整数
commandTraceMaxFileSize	コマンドトレースログファイルの最大サイズ（単位：バイト）を指定します。	16777216	4096～16777216 の整数
serverMessageFileCount	サーバメッセージログファイルの最大数を指定します。	4	2～16 の整数

パラメーター名	説明	デフォルト値	許容される値の範囲
serverMessageMaxFileSize	サーバメッセージログファイルの最大サイズ（単位：バイト）を指定します。	16777216	4096～16777216 の整数
serverMessageLogLevel	サーバメッセージログファイルのログレベルを指定します。 2 を指定すると，より詳細なメッセージが出力されます。	1	1 | 2
serverTraceFileCount	サーバトレースログファイルの最大数を指定します。	4	2～16 の整数
serverTraceMaxFileSize	サーバトレースログファイルの最大サイズ（単位：バイト）サイズを指定します。	16777216	4096～16777216 の整数
hsdpLogfileDir	共通コンポーネント用ログファイルの格納先を指定します。 ディレクトリを指定する場合は絶対パスで指定してください。	*運用ディレクトリ*/logs	1～255 文字のディレクトリの絶対パス 指定できる文字は次のとおりです。 • 半角英数字 • アンダーライン（_） • スラッシュ（/）

注意事項

- このファイルが存在しない場合は，デフォルト値で動作します。
- パラメーターに不正な値（空文字を含む）を指定した場合，またはパラメーターを省略した場合は，デフォルト値で動作します。
- パラメーターを変更した場合は，SDP サーバを停止したあと，ログ出力ディレクトリ内のすべてのファイルとサブディレクトリを別のディレクトリに移動するか，削除する必要があります。
- hsdpLogfileDir パラメーターにパスを指定する場合は，ローカルファイルシステムのパスを指定してください。ネットワークドライブをマウントしたパスを指定したときのログ出力の動作は保証されません。
- hsdpLogfileDir パラメーターに指定したパスが存在しない場合は，自動的にディレクトリが作成されます。
- hsdpLogfileDir パラメーターに指定したパスへのログの出力に失敗した場合は，ログファイルにログは出力されません。
- 複数の運用ディレクトリ下にあるログファイルの格納先ディレクトリを変更する場合，それぞれ同一のディレクトリに変更しないでください。複数のプロセスから，同じファイルへ書き込みを実行することになり，ログファイルが壊れるおそれがあります。

12.14　SDP マネージャー定義ファイル

記述形式

表 12-17　SDP マネージャー定義ファイルのプロパティ

項番	プロパティ名	説明	値の範囲	デフォルト値
1	hsdp_cpu_no_list	次のコンポーネントのプロセスに割り当てられている CPU 番号を指定します。 • SDP マネージャー • SDP ブローカー • SDP コーディネーター • SDP サーバ CPU 番号は，{*n* | *m1-m2*} [,…]の形式で指定する必要があります。ここでは，*n* は CPU 番号，*m1-m2* は*m1-m2* の CPU 番号の範囲を表します。複数の*n* または*m1-m2* を指定するには，コンマ（,）で区切って指定します。 • コンマ（,）と次の CPU 番号の間にスペースを挿入しないでください。 • 範囲形式*m1-m2* で指定する CPU 番号は，昇順にする必要があります。 例えば，次の指定は正しくありません。 6-4 CPU リスト定義の例を次に示します。 • 0,2,3：CPU #0，#2，#3 が割り当てられます。 • 4-7：CPU #4～#7 が割り当てられます。 • 0,2,4-6：CPU #0，#2，#4～#6 が割り当てられます。 指定した値が無効な場合，hsdpmanager -start コマンドは失敗します。 指定を省略した場合，プロセスに割り当てる CPU 番号は限定されません。	0 から（*ホストのCPUの数*-1）	CPU 番号は限定されません。
2	hsdp_restart	プロセスが異常終了したときに，そのプロセスを自動的に再起動する機能を有効にするかどうかを指定します。false を指定すると，再起動機能は無効になり，ログ通知（監視中の対象プロセスが失敗した場合）だけが有効になります。 なお，メッセージログ出力機能の初期化に失敗した場合は，このプロパティの指定値によって動作が異なります。 • true：メッセージログの出力先が syslog に切り替わります。出力先のパスは OS の設定値を確認してください。 • false：メッセージログは出力されません。 指定した値が無効な場合，hsdpmanager -start コマンドは失敗します。	true|false true：有効にします。 false：無効にします。	false

項番	プロパティ名	説明	値の範囲	デフォルト値
3	hsdp_restart_watch_time	プロセスの再起動失敗回数監視中の最大経過時間を分単位で指定します。指定した時間中にプロセスが連続して3回失敗すると，再起動はキャンセルされます。その場合，問題を修正し，hsdpmanager -start コマンドを実行します（SDP サーバの場合，hsdpstart -restart コマンドを実行します）。このプロパティは，監視対象の各プロセスに適用されます。 0 を指定した場合，再起動はキャンセルされません（無制限に再起動がリトライされます）。指定した時間が3 x hsdp_coordinator_boot_timeout より短い場合，再起動を停止できないことがあり，その結果無制限のリトライが発生します。 指定した値が無効な場合，デフォルト値が使用されます。	0～2147483647 の整数です。	10
4	hsdp_manager	SDP マネージャーを使用するかどうかを指定します。 false を指定した場合，SDP マネージャーのプロセスは起動しません。このため，監視対象のプロセスダウン時にログ通知機能および再起動機能は動作しません。指定した値が無効な場合，デフォルト値が使用されます。	true|false true：使用します。 false：使用しません。	true
5	hsdp_broker※	SDP ブローカーを使用するかどうかを指定します。 false を指定すると，外部アダプターまたはカスケーディングアダプターを受け付けたり，hsdpstatusshow コマンドを使用してほかのホストに関する情報を表示したりできません。これは，SDP ブローカープロセスが起動しないためです。 指定した値が無効な場合，デフォルト値が使用されます。	true|false true：使用します。 false：使用しません。	true
6	hsdp_coordinator※	SDP コーディネーターを使用するかどうかを指定します。 false を指定した場合，SDP コーディネーターのプロセスは起動しません。このため，外部アダプターまたはカスケーディングアダプターが SDP ブローカーに接続した際に，他ホストのストリーム情報を検索できません。また，SDP コーディネーターと連携する SDP ブローカーで，自ホストのストリーム情報を検索できません。 指定した値が無効な場合，デフォルト値が使用されます。	true|false true：使用します。 false：使用しません。	true
7	hsdp_retry_times	次のプロセス間の通信が失敗した場合の最大リトライ回数を指定します。 • SDP マネージャー • SDP ブローカー • SDP サーバ 指定した値が無効な場合，デフォルト値が使用されます。 このプロパティに基づくリトライ中に，送信側のプロセスはプロセス，クエリグループ，または内部アダプターに対する操作を実行できません。	-1～2147483647 （-1 を指定すると，無制限に通信がリトライされます。0 を指定すると，通信はリトライされません）	30

項番	プロパティ名	説明	値の範囲	デフォルト値
7	hsdp_retry_times	デフォルト値でタイムアウトが頻繁に発生する場合は，このプロパティの値を調整します。	-1～2147483647 (-1 を指定すると，無制限に通信がリトライされます。0 を指定すると，通信はリトライされません)	30
8	hsdp_retry_interval	次のプロセス間の通信が失敗した場合のリトライ間隔（秒単位）を指定します。 • SDP マネージャー • SDP ブローカー • SDP サーバ 指定した値が無効な場合，デフォルト値が使用されます。 このプロパティに基づくリトライ中に，送信側のプロセスはプロセス，クエリグループ，または内部アダプターに対する操作を実行できません。 デフォルト値でタイムアウトが頻繁に発生する場合は，このプロパティの値を調整します。	1～2147483647	3
9	hsdp_coordinator_boot_timeout	リンクされたコーディネーターがすべて起動するまでのタイムアウト時間を秒単位で指定します。 hsdp_coordinator_boot_timeout に指定された値が，hsdp_restart_watch_time に指定された値の 3 倍より長い場合，再起動がキャンセルされないことがあります(無制限に再起動がリトライされることがあります)。 指定した値が無効な場合，デフォルト値が使用されます。	1～86400	60
10	hsdp_manager_log_filecount	SDP マネージャーのメッセージログファイルの最大数を指定します。 指定した値が無効な場合，デフォルト値が使用されます。	2～16 の整数です。	4
11	hsdp_manager_log_filesize	SDP マネージャーのメッセージログファイルの最大サイズ（単位：バイト）を指定します。 指定した値が無効な場合，デフォルト値が使用されます。	4096～16777216 の整数です。	16777216
12	hsdp_manager_log_level	SDP マネージャーのメッセージログファイルのログレベルを指定します。 2 を指定すると，より詳細なメッセージが出力されます。 指定した値が無効な場合，デフォルト値が使用されます。	1 | 2	1
13	hsdp_manager_trace_filecount	SDP マネージャーのトレースログファイルの最大数を指定します。 指定した値が無効な場合，デフォルト値が使用されます。	2～16 の整数です。	4
14	hsdp_manager_trace_filesize	SDP マネージャーのトレースログファイルの最大サイズ（単位：バイト）を指定します。 指定した値が無効な場合，デフォルト値が使用されます。	4096～16777216 の整数です。	16777216
15	hsdp_manager_trace_level	SDP マネージャーのトレースログファイルのログレベルを指定します。	1 | 2	1

項番	プロパティ名	説明	値の範囲	デフォルト値
15	hsdp_manager_trace_level	2 を指定すると，より詳細なメッセージが出力されます。 指定した値が無効な場合，デフォルト値が使用されます。	1 | 2	1
16	hsdp_broker_log_filecount	SDP ブローカーのメッセージログファイルの最大数を指定します。 指定した値が無効な場合，デフォルト値が使用されます。	2～16 の整数です。	4
17	hsdp_broker_log_filesize	SDP ブローカーのメッセージログファイルの最大サイズ（単位：バイト）を指定します。 指定した値が無効な場合，デフォルト値が使用されます。	4096～16777216 の整数です。	16777216
18	hsdp_broker_log_level	SDP ブローカーのメッセージログファイルのログレベルを指定します。 2 を指定すると，より詳細なメッセージが出力されます。 指定した値が無効な場合，デフォルト値が使用されます。	1 | 2	1
19	hsdp_broker_trace_filecount	SDP ブローカーのトレースログファイルの最大数を指定します。 指定した値が無効な場合，デフォルト値が使用されます。	2～16 の整数です。	4
20	hsdp_broker_trace_filesize	SDP ブローカーのトレースログファイルの最大サイズ（単位：バイト）を指定します。 指定した値が無効な場合，デフォルト値が使用されます。	4096～16777216 の整数です。	16777216
21	hsdp_broker_trace_level	SDP ブローカーのトレースログファイルのログレベルを指定します。 2 を指定すると，より詳細なメッセージが出力されます。 指定した値が無効な場合，デフォルト値が使用されます。	1 | 2	1
22	hsdp_coordinator_log_filecount	SDP コーディネーターのメッセージログファイルの最大数を指定します。 指定した値が無効な場合，デフォルト値が使用されます。	2～16 の整数です。	4
23	hsdp_coordinator_log_filesize	SDP コーディネーターのメッセージログファイルの最大サイズ（単位：バイト）を指定します。 指定した値が無効な場合，デフォルト値が使用されます。	4096～16777216 の整数です。	16777216
24	hsdp_coordinator_log_level	SDP コーディネーターのメッセージログファイルのログレベルを指定します。 2 を指定すると，より詳細なメッセージが出力されます。 指定した値が無効な場合，デフォルト値が使用されます。	1 | 2	1
25	hsdp_coordinator_trace_filecount	SDP コーディネーターのトレースログファイルの最大数を指定します。 指定した値が無効な場合，デフォルト値が使用されます。	2～16 の整数です。	4
26	hsdp_coordinator_trace_filesize	SDP コーディネーターのトレースログファイルの最大サイズ（単位：バイト）を指定します。 指定した値が無効な場合，デフォルト値が使用されます。	4096～16777216 の整数です。	16777216

項番	プロパティ名	説明	値の範囲	デフォルト値
27	hsdp_coordinator_trace_level	SDP コーディネーターのトレースログファイルのログレベルを指定します。 2 を指定すると，より詳細なメッセージが出力されます。 指定した値が無効な場合，デフォルト値が使用されます。	1 | 2	1
28	hsdp_logfile_dir	次に示すログファイルの格納先を指定します。 • SDP マネージャー用ログファイル • SDP ブローカー用ログファイル • SDP コーディネーター用ログファイル 格納先ディレクトリを変更する場合は，絶対パスで指定してください。	1～255 文字のディレクトリの絶対パス 指定できる文字は次のとおりです。 • 半角英数字 • アンダーライン（_） • スラッシュ（/）	/var/log/hitachi/hsdp
注※ 2 つのプロパティには，同じ値（true またはfalse のどちらか）を設定してください。異なる値を設定した場合，SDP ブローカーと SDP コーディネーターは正常に動作しないおそれがあります。				

ファイル名

hsdp_setup_manager.cfg

ファイル格納場所

/opt/hitachi/hsdp/conf/

説明

次のコンポーネントのプロパティは，SDP マネージャー定義ファイルで定義されます。

- SDP マネージャー
- SDP ブローカー
- SDP コーディネーター
- SDP サーバ

SDP マネージャー定義ファイルが使用される場合，SDP マネージャーは次のように動作します。

- hsdpmanager コマンドが実行されます。
- SDP マネージャーが起動します。
- SDP マネージャーは，SDP マネージャー定義ファイルを読み込みます。
- SDP マネージャーは，SDP マネージャー定義ファイルのプロパティの定義に従って動作します。

SDP マネージャー定義ファイルの更新後，各プロセスでプロパティを有効にする契機は，変更されたプロパティによって異なります。

SDP マネージャー，SDP ブローカー，および SDP コーディネーターのプロパティは，次のどちらかの契機によってプロセスで有効になります。

- hsdpmanager コマンドの-start オプションが初めて実行されたとき。
- hsdpmanager コマンドの-start オプションが，hsdpmanager コマンドの-stop オプションによって停止されたあとに実行されたとき。

メモ

変更されたプロパティは，SDP マネージャー，SDP ブローカー，および SDP コーディネーターの各プロセスが異常終了したあと，（hsdpmanager コマンドまたは SDP マネージャーの再起動機能を使用した）再起動中には有効になりません。

SDP サーバの場合：

プロパティは，SDP サーバの起動時に各プロセスで有効になります。

メモ

- 変更されたプロパティは，SDP サーバプロセスが異常終了したあと，（hsdpmanager コマンドまたは SDP マネージャーの再起動機能を使用した）再起動中に有効になります。
- SDP マネージャーがこのファイルを使用しないで起動された場合，SDP マネージャーはプロパティのデフォルト値に従って動作します。

指定できるパラメーター

なし

例

なし

注意事項

- パラメーターに無効な値もしくは範囲外の値を指定した場合，またはパラメーターが定義されていない場合は，デフォルト値が使用されます。
- このファイルが存在しない場合は，デフォルト値が使用されます。
- パラメーターを変更した場合は，SDP サーバを停止したあと，ログ出力ディレクトリ内のすべてのファイルとサブディレクトリを別のディレクトリに移動するか，削除する必要があります。
- hsdp_logfile_dir パラメーターにパスを指定する場合は，ローカルファイルシステムのパスを指定してください。ネットワークドライブをマウントしたパスを指定したときのログ出力の動作は保証しません。

- hsdp_logfile_dir パラメーターに指定したパスが存在しない場合は，自動的にディレクトリが作成されます。
- hsdp_logfile_dir パラメーターに指定したパスへのログの出力に失敗した場合は，ログファイルにログは出力されません。

サンプル

SDP マネージャー定義ファイルの定義サンプルは，次のファイルに格納されています。

/opt/hitachi/hsdp/template/setup/hsdp_setup_manager.cfg

12.15　外部アダプター定義ファイル

記述形式

java.util.Properties クラスの仕様に従って記述してください。

次のフォーマットに従い，1 行ごとに各パラメーターを指定してください。

キー=値

- キーと値の区切り文字として，等号（=），ピリオド（.），またはスペースを使用できます。
- コメントを記載する場合は，パラメーターの行とは別の行に，先頭にハッシュマーク（#）を付けてコメントを記載してください。
- 値の後ろには，スペースやコメントなどの文字列を追加できません。追加した場合，値の一部として見なされます。
- キーにスペースは使用できません。
- 区切り文字の前後にあるスペース，およびキーの前にあるスペースは，すべて無視されます。
- パラメーター名だけを指定して値を指定していない行は，デフォルト値が設定されます。デフォルト値が存在しないパラメーターの場合はエラーになります。

ファイル名

任意のファイル名

ファイル格納先

任意のディレクトリ

説明

外部アダプターに関連するプロパティをこのファイルに指定します。ユーザー任意のパラメーターも指定できます。ただし，プレフィックスに「hsdp.」が付いたパラメーターは外部アダプターの予約語として登録されているため，指定しないでください。

ファイル内のプロパティ値を変更する場合は，外部アダプターを停止してから変更してください。

指定できるパラメーター

パラメーターとして指定できるキーと値の組み合わせを次の表に示します。

表 12-18　指定できるキー値

項番	キー	値	デフォルト値
1	hsdp.adaptor.name	外部アダプターの名称を 1～64 文字の半角英数字およびアンダーライン（_）で指定します。先頭に使用できる文字は半角の英字だけです。 この名称は外部アダプターのログファイルが出力されるディレクトリ名になります。複数の外部アダプターを使用する場合は，ホスト内でほかのアダプターが使用していない名称にしてください。	exadp
2	hsdp.broker.address	SDP ブローカーのホスト名または IP アドレスとポート番号を次の形式で指定します。 *ホスト名またはIPアドレス:ポート番号*	localhost:20425
3	hsdp.broker.retry.count	SDP ブローカー接続失敗時のリトライ回数を指定します。 指定可能な範囲：-1～INT_MAX -1 を指定した場合は無制限にリトライし続けます。	30
4	hsdp.broker.retry.interval	SDP ブローカー接続失敗時のリトライ間隔を指定します（単位：秒）。 指定可能な範囲：1～INT_MAX	3
5	hsdp.target.name.*n*	データを送受信する宛先を次の形式で指定します。 指定形式： [*/BrokerAddr/*][*ServerCluster/*]*QueryGroup/Stream*[,[*/BrokerAddr/*][*ServerCluster/*]*QueryGroup/Stream*][,...] • *BrokerAddr*：接続先 SDP ブローカーの IP アドレスまたはホスト名 • *ServerCluster*：接続先サーバクラスタ名 1～64 文字の半角英数字，およびアンダーライン（_）が使用できます。先頭に使用できる文字は半角の英字だけです。 • *QueryGroup*：接続先クエリグループ名 1～64 文字の半角英数字，およびアンダーライン（_）が使用できます。先頭に使用できる文字は半角の英字だけです。 • *Stream*：接続先ストリーム名 1～100 文字の半角英数字，およびアンダーライン（_）が使用できます。先頭に使用できる文字は半角の英字だけです。 *BrokerAddr* を省略した場合，hsdp.broker.address への指定値が適用されます。 *ServerCluster* を省略した場合，デフォルト値"My_server_cluster"が適用されます。 *n* （範囲：1～INT_MAX）に異なる値を指定することで，複数の宛先を指定できます。 複数の宛先を指定したキーをHSDPAdaptorManager クラスの openStreamInput()メソッドに指定した場合，put()メソッドで送信を要求したデータは指定された複数の宛先に送信されます。 このパラメーターに複数の宛先を指定する場合は，宛先間で次の項目を一致させてください。 • スキーマ（ストリーム名を含む） • 並列数 • 分散方式	None

項番	キー	値	デフォルト値
5	hsdp.target.name.*n*	• タイムスタンプモード また，宛先の CQL に定義するストリームのスキーマ，およびクエリグループ用プロパティファイルに定義する分散方式は，大文字と小文字を区別して同一にしてください。 複数の宛先を 1 つのキーに指定する場合，同じ宛先は指定できません。 外部出力アダプターを使用する場合，1 つのキーに複数の宛先を指定できません。指定した場合，起動時にエラーになります。 定義例（*Broker* および *ServerCluster* を省略）： hsdp.target.name.1=QG1/S1 hsdp.target.name.2=QG2/S2 定義例（*Broker* を指定，*ServerCluster* を省略）： hsdp.target.name.1=/192.168.10.20:20425/QG1/S1 定義例（1 つのキーに複数の宛先を指定）： hsdp.target.name.1=QG1/S1,QG2/S1	None
6	hsdp.send.unit	外部入力アダプターが宛先のストリームに一度に送信する最大データ数を指定します。単位時間当たりの送信データ数が多い場合，一度に送信するデータ数を増やすことで，処理性能が向上することがあります。 このパラメーターは，外部入力アダプターの場合だけ有効です。 指定可能な範囲：1～1024	1024
7	hsdp.send.timeout	SDP サーバへのデータ送信間隔を指定します（単位：ミリ秒）。このパラメーターは外部入力アダプターの場合だけ有効です。 指定可能な範囲：0～INT_MAX	5
8	hsdp.send.queuesize	外部アダプター内で送信データを保持するキューのサイズを指定します。分散方式がROUNDROBIN の場合は，HSDPStreamInput インタフェースの openStreamInput()メソッドで指定した宛先ごとに 1 つのキューが作成されます。分散方式がROUNDROBIN 以外の場合は，コネクションの接続先ごとにキューが作成されます。 キューの空きがなくデータを登録できない場合，古いデータが削除され，新しいデータが登録されます。実際の送信データ数に合わせて設定してください。 指定可能な範囲：1～INT_MAX	10000
9	hsdp.send.queuefull.action	外部アダプター内で送信データを保持するキューが満杯の場合の動作を指定します。このパラメーターは外部入力アダプターの場合だけ有効です。 • discard HSDPStreamInput インタフェースのput メソッドおよびputControl メソッドで送信したデータは送信キューに登録されません。その際，HSDPAdaptorQueueFullException の例外がスローされます。 • overwrite 送信キューに登録されている最も古いデータが削除され，HSDPStreamInput インタフェースのput メソッドおよびputControl メソッドで送信したデータが送信キューに登録されます。登録完了後，HSDPAdaptorQueueSizeLackException の例外がスローされます。	discard

項番	キー	値	デフォルト値
9	hsdp.send.queuefull.action	discard またはoverwrite のどちらかを指定した場合でも，スケールアップ／スケールアウト構成などの送信先が複数となる構成のときは，送信キューが 1 つでも満杯になるとHSDPAdaptorQueueFullException または HSDPAdaptorQueueSizeLackException の例外がスローされます。	discard
10	hsdp.receive.queuesize	外部アダプター内で受信データを保持するキューのサイズを指定します。HSDPStreamOutput インタフェースのregister()メソッドで指定したコールバックオブジェクトごとに，1 つのキューが作成されます。 指定可能な範囲：1～INT_MAX	10000
11	hsdp.connection.retry.count	コネクション切断時の再接続リトライ回数を指定します。 指定可能な範囲：-1～INT_MAX -1 を指定した場合は無限にリトライします。	-1
12	hsdp.connection.retry.interval	コネクション切断時の再接続リトライ間隔を指定します（単位：秒）。 指定可能な範囲：1～INT_MAX	1
13	hsdp.logger.message.filecount	メッセージログファイルの最大面数を指定します。 指定可能な範囲：2～16	4
14	hsdp.logger.message.maxfilesize	メッセージログファイルの最大サイズを指定します（単位：バイト）。 指定可能な範囲：4096～16777216	16777216
15	hsdp.logger.trace.filecount	トレースログファイルの最大面数を指定します。 指定可能な範囲：2～16	4
16	hsdp.logger.trace.maxfilesize	トレースログファイルの最大サイズを指定します（単位：バイト）。 指定可能な範囲：4096～16777216	16777216
17	hsdp.logger.loglevel	メッセージおよびトレースのログレベルを指定します。 2 を指定した場合，詳細メッセージを出力します。 指定可能な値：1 または2	1
18	hsdp.tcp.connect.timeout	TCP コネクション確立のタイムアウト時間を指定します（単位：秒）。 指定可能な範囲：0～INT_MAX 0 を指定した場合は，タイムアウトしなくなります。	20
19	hsdp.tcp.sendbuffersize	OS のソケット送信バッファーのサイズを指定します（単位：バイト）。指定した値は，SO_SNDBUF socket option に指定されます。 指定可能な範囲：1～INT_MAX	OS 設定値
20	hsdp.tcp.receivebuffersize	OS のソケット受信バッファーのサイズを指定します（単位：バイト）。指定した値は，SO_RCVBUF socket option に指定されます。 指定可能な範囲：1～INT_MAX	OS 設定値
21	hsdp.message.suppress.count	SDP サーバへのコネクション接続リトライ中に表示されるメッセージの抑止回数を指定します。一度メッセージを出力すると，それ以降にリトライが失敗した場合に表示されるメッセージ出力は，指定回数抑止されます。 例えば，冗長構成で一部の接続先と接続できない場合に出力されるメッセージ量を調整できます。	60

項番	キー	値	デフォルト値
21	hsdp.message.suppress.count	指定可能な範囲：-1～INT_MAX -1 を指定した場合は，リトライ中にメッセージを出力しません。	60
22	hsdp.message.output.interval	次のメッセージの出力間隔を指定します（単位：秒）。 • KFHD13022-W • KFHD13029-W • KFHD13031-W 送信キューごとに，メッセージ出力から指定時間以上経過するまでメッセージの出力は抑止されます。 例えば，冗長構成で一部の接続先と接続できない場合に出力されるメッセージ量を調整できます。 指定可能な範囲：-1，1～INT_MAX -1 を指定した場合，メッセージは出力されません。	60
23	hsdp.logfile.dir	外部アダプターのログファイルを格納するディレクトリを指定します。 指定可能な値：1～255 文字のディレクトリの絶対パス 指定できる文字は次のとおりです。 • 半角英数字 • アンダーライン（_） • スラッシュ（/）	/var/log/hitachi/hsdp/exadaptor/*外部アダプター名*※

注※

外部アダプター定義ファイルのhsdp.adaptor.name プロパティで指定した名前

注意事項

- パラメーターに無効な値もしくは範囲外の値を指定した場合，またはパラメーターが定義されていない場合は，デフォルト値が使用されます。
- このファイルが存在しない場合は，デフォルト値が使用されます。
- パラメーターを変更した場合は，SDP サーバを停止したあと，ログ出力ディレクトリ内のすべてのファイルとサブディレクトリを別のディレクトリに移動するか，削除する必要があります。
- hsdp.logfile.dir パラメーターにパスを指定する場合は，ローカルファイルシステムのパスを指定してください。ネットワークドライブをマウントしたパスを指定したときのログ出力の動作は保証されません。
- hsdp.logfile.dir パラメーターに指定したパスが存在しない場合は，自動的にディレクトリが作成されます。
- hsdp.logfile.dir パラメーターに指定したパスへのログの出力に失敗した場合は，ログファイルにログは出力されません。
- 外部アダプターのログファイルの格納先ディレクトリを変更する場合，複数の外部アダプターの格納先ディレクトリを同一のディレクトリに変更しないでください。複数の外部アダプターのプロセスから，同じファイルへ書き込みを実行することになり，ログファイルが壊れるおそれがあります。

12.16　SDP マネージャーコマンドのログ定義ファイル

記述形式

パラメーターは次の形式で記述します。

```
パラメーター名=値
```

- 値の後ろにスペースやコメントなどの文字列を追加できません。
  追加した場合は値の一部として解釈されます。
- パラメーター名だけを指定して，値を指定していない行は，無視されます。

ファイル名

hsdpmanagerlogger.properties

ファイルの格納先

このファイルは，必ず次のディレクトリに格納してください。

```
/opt/hitachi/hsdp/conf/
```

説明

このファイルには，次のコンポーネントに関連する SDP マネージャーコマンド（hsdpmanager）のロガーに関するプロパティを指定します。

- SDP マネージャー
- SDP ブローカー
- SDP コーディネーター

指定できるパラメーター

指定できるパラメーターとデフォルト値を次の表に示します。

表 12-19　指定できるパラメーターとデフォルト値（hsdpmanagerlogger.properties）

項番	パラメーター名	内容	デフォルト値	指定できる値の範囲
1	commandMessageFileCount	コマンドメッセージログファイルの最大数を指定します。	4	2～16 の整数
2	commandMessageMaxFileSize	コマンドメッセージログファイルの最大サイズ（単位：バイト）を指定します。	16777216	4096～16777216 の整数

項番	パラメーター名	内容	デフォルト値	指定できる値の範囲
3	commandMessageLogLevel	コマンドメッセージログファイルのログレベルを指定します。 2 を指定すると，より詳細なメッセージが出力されます。	1	1 | 2
4	commandTraceFileCount	コマンドトレースログファイルの最大数を指定します。	4	2～16 の整数
5	commandTraceMaxFileSize	コマンドトレースログファイルの最大サイズ（単位：バイト）を指定します。	16777216	4096～16777216 の整数
6	commandTraceLogLevel	コマンドトレースログファイルのログレベルを指定します。 2 を指定すると，より詳細なメッセージが出力されます。	1	1 | 2
7	hsdpLogfileDir	次に示すログファイルの格納先を指定します。 • コマンドメッセージログファイル • コマンドトレースログファイル 格納先ディレクトリを変更する場合は，絶対パスで指定してください。	/var/log/hitachi/hsdp	1～255 文字のディレクトリの絶対パス 指定できる文字は次のとおりです。 • 半角英数字 • アンダーライン（_） • スラッシュ（/）

注意事項

- このファイルが存在しない場合は，デフォルト値で動作します。
- パラメーターに不正な値（空文字を含む）を指定した場合，またはパラメーターを省略した場合は，デフォルト値で動作します。
- パラメーターを変更した場合は，SDP マネージャー，SDP ブローカー，および SDP コーディネーターを停止したあと，ログ出力ディレクトリ内のすべてのファイルとサブディレクトリを別のディレクトリに移動するか，削除する必要があります。
- hsdpLogfileDir パラメーターにパスを指定する場合は，ローカルファイルシステムのパスを指定してください。ネットワークドライブをマウントしたパスを指定したときのログ出力の動作は保証されません。
- hsdpLogfileDir パラメーターに指定したパスが存在しない場合は，自動的にディレクトリが作成されます。
- hsdpLogfileDir パラメーターに指定したパスへのログの出力に失敗した場合は，ログファイルにログは出力されません。

12.17　ログ収集定義ファイル

記述形式

絶対パスまたは運用ディレクトリからの相対パスで，ファイルパスを 1 行に 1 つずつ指定してください。

OS でサポートされているワイルドカード表記を指定できます。

ファイル名

任意の名前

ファイル格納先

任意のディレクトリ

説明

hsdplogcollect コマンドで収集するファイルをこのファイルで指定します。

指定できるパラメーター

ファイルパスとワイルドカード（*）が指定できます。

注意事項

- ディレクトリのパスは指定できません。ディレクトリのパスを指定した場合，無視して処理が続行されます。
- ファイルパスにシンボリックリンクを指定した場合は，シンボリックリンク自身が収集されます。
- ログ収集定義ファイルを作成した場合，ログ収集定義ファイル内のファイルパスにログ収集定義ファイル自身も指定してください。
- 巨大なサイズのファイルを指定しないでください。コマンドの実行時間が長くなるだけでなく，コマンドが出力するファイルのサイズが大きくなるため，ファイルをメールなどで送付する場合に時間が掛かるようになります。

例

```
/home/hsdp/conf/abc.properties
/var/hitachi/hsdp/logs/*.log
```

12.18　エクスポート定義ファイル

記述形式

絶対パスまたは運用ディレクトリからの相対パスで，ファイルパスを 1 行ずつ指定してください。

次のファイルパスは，エクスポート定義ファイルにデフォルトで指定されています。

項番	ファイルパス
1	*運用ディレクトリ*/conf/*
2	*運用ディレクトリ*/conf/xml/*
3	*運用ディレクトリ*/query/*
4	*運用ディレクトリ*/lib/*

エクスポートするファイルを追加する場合は，このファイルにファイルパスを追加してください。または，このファイルをコピーしたエクスポート定義ファイルに，ファイルパスを追加してください。この場合，hsdpexport コマンドの-file オプションのオプション引数に，コピーしたエクスポート定義ファイルのパスを指定してください。

ファイル名

hsdpexport.cfg

ファイル格納先

運用ディレクトリ/conf/

説明

hsdpexport コマンドによってエクスポートするファイルを，このファイルに指定します。

次のディレクトリ以下のファイルは指定してもエクスポートされません。

- *運用ディレクトリ*/logs
- *運用ディレクトリ*/export
- *運用ディレクトリ*/spool
- *運用ディレクトリ*/trc

例

特になし。

注意事項

- ディレクトリのパスは指定できません。ディレクトリのパスを指定した場合，無視して処理が続行されます。
- ファイルパスにシンボリックリンクを指定した場合は，シンボリックリンク自身が収集されます。
- 巨大なサイズのファイルを指定しないでください。コマンドの実行時間が長くなるだけでなく，コマンドが出力するファイルのサイズが大きくなるため，ファイルをほかのホストにファイル転送する場合に，時間が掛かるようになります。

12.19　JavaVM オプションの一覧

ここでは，日立 JavaVM のオプションについて説明します。日立 JavaVM のオプションは，次のファイルで指定できます。

- SDP サーバ用 JavaVM オプションファイル（jvm_options.cfg）
- SDP マネージャー用 JavaVM オプションファイル（manager_jvm.cfg）
- SDP ブローカー用 JavaVM オプションファイル（broker_jvm.cfg）
- SDP コーディネーター用 JavaVM オプションファイル（coordionator_jvm.cfg）
- acceleration CQL エンジン用 JavaVM オプションファイル（jvm_cengine_options.cfg）

  このファイルへの指定方法，およびこのシステムでのデフォルト値については，「12.4　SDP サーバ用 JavaVM オプションファイル（jvm_options.cfg）」を参照してください。

JavaVM オプションの一覧を次の表に示します。なお，ここで説明する JavaVM オプションのデフォルト値は，SDP サーバ用 JavaVM オプションファイル（jvm_options.cfg），SDP マネージャー用 JavaVM オプションファイル（manager_jvm.cfg），SDP ブローカー用 JavaVM オプションファイル（broker_jvm.cfg），SDP コーディネーター用 JavaVM オプションファイル（coordionator_jvm.cfg），または acceleration CQL エンジン用 JavaVM オプションファイル（jvm_cengine_options.cfg）で JavaVM オプションを指定していない場合のデフォルト値です。

表 12-20　JavaVM オプションの一覧

項番	分類	オプション名	説明
1	JavaVM メモリ空間のサイズや割合指定オプション	-Xms<*サイズ*>※1	Java ヒープの初期サイズを指定します。
		-Xmx<*サイズ*>※1	Java ヒープの最大サイズを指定します。
		-XX:NewRatio=<*値*>	DefNew 領域に対するTenured 領域の割合を指定します。<*値*>が2 の場合は，DefNew 領域とTenured 領域の比が，1：2 になります。
		-XX:MetaspaceSize=<*サイズ*>※1	Metaspace 領域の初期サイズを指定します。
		-XX:MaxMetaspaceSize=<*サイズ*>※1	Metaspace 領域の最大サイズを指定します。
2	拡張スレッドダンプ機能オプション	-XX:[+|-]HitachiThreadDumpToStdout	標準出力にスレッドダンプを出力するかどうかを指定します。 • -XX:+HitachiThreadDumpToStdout： 拡張スレッドダンプを標準出力，および日立 JavaVM ログファイルに出力します。 • -XX:-HitachiThreadDumpToStdout： 拡張スレッドダンプを標準出力に出力しません。日立 JavaVM ログファイルだけに出力します。

項番	分類	オプション名	説明
2	拡張スレッドダンプ機能オプション	-XX:[+|-]HitachiThreadDumpToStdout	デフォルト値は-XX:+HitachiThreadDumpToStdout です。
3	日立 JavaVM ログファイルオプション	-XX:HitachiJavaLog:<*文字列*>	ログファイル名のプレフィックスを指定します。ログファイル名は「<*文字列*>*xx*.log」(*xx* は01～99 の通し番号）で生成されます。 例えば，*文字列*にSamp と指定するとログファイル名は「Samp01.log」となります。 *文字列*にはパスも指定できます。*文字列*にディレクトリを指定した場合，そのディレクトリにデフォルトの名称でログファイルを作成します。 デフォルト値はjavalog です。
		-XX:HitachiJavaLogFileSize=<*整数値*>	1 ファイルの最大サイズ（単位：**キロバイト**）を1024～INT_MAX の整数で指定します。最大サイズを超えた場合，そのファイルは出力されません。 デフォルト値は256 です。 範囲外の値が指定された場合は1024 となります。
		-XX:HitachiJavaLogNumberOfFile=<*整数値*>	ログファイルの単純増加を防ぐため，作成する最大ファイル数を1～99 の整数で指定します。最大ファイル数を超えた場合は，再度，最初に作成したファイルへの出力を開始します。 デフォルト値は4 です。 100 以上の値が指定された場合は99 となります。0 以下の値が指定された場合は1 となります。
4	詳細時間出力オプション	-XX:[+|-]HitachiOutputMilliTime	**ミリ秒**までの時間を出力するかどうかを指定します。 • -XX:+HitachiOutputMilliTime： 日立 JavaVM ログファイルに出力する日時に，**ミリ秒**まで出力します。 • -XX:-HitachiOutputMilliTime： 日立 JavaVM ログファイルに出力する日時に，**秒**まで出力します。 デフォルト値は-XX:-HitachiOutputMilliTime です。
5	メモリ管理方式オプション	-XX:-UseG1GC	ガーベージコレクションとしてG1GC を使用するかどうか指定します。
6	拡張verbosegc 機能オプション	-XX:[+|-]HitachiVerboseGC ※2	ガーベージコレクションが発生した場合，拡張verbosegc 情報を出力するかどうかを指定します。 • -XX:+HitachiVerboseGC： ガーベージコレクションが発生した場合，拡張verbosegc 情報を日立 JavaVM ログファイルに出力します。ガーベージコレクションの内部領域である Eden，Survivor，Tenured，Perm 種別の情報を拡張verbosegc 情報として出力します。

項番	分類	オプション名	説明
6	拡張verbosegc 機能オプション	-XX:[+|-]HitachiVerboseGC※2	• -XX:-HitachiVerboseGC： ガーベージコレクションが発生した場合，拡張verbosegc 情報を日立 JavaVM ログファイルに出力しません。 デフォルト値は-XX:-HitachiVerboseGC です。
		-XX:[+|-]HitachiVerboseGCPrintCause※3	ガーベージコレクションの要因内容を出力するかどうかを指定します。 • -XX:+HitachiVerboseGCPrintCause： ガーベージコレクションの要因内容を，拡張verbosegc 情報の行末に出力します。 • -XX:-HitachiVerboseGCPrintCause： 拡張verbosegc 情報を通常形式で出力します。 デフォルト値は-XX:+HitachiVerboseGCPrintCause です。
7	OutOfMemoryError 発生時の拡張機能オプション	-XX:[+|-]HitachiOutOfMemoryStackTrace※2	OutOfMemoryError 発生時のスタックトレースを出力するかどうかを指定します。 • -XX:+HitachiOutOfMemoryStackTrace： OutOfMemoryError 発生時に，例外情報とスタックトレースを日立 JavaVM ログファイルに出力します。スタックトレースは 1 スタックごとにバッファーに格納し，コード変換したあとに出力します。スタックトレースの出力は，OutOfMemoryError がスローされるたびに行われるため，OutOfMemoryError をキャッチして再スローした場合には複数回出力されます。なお，スレッド作成時にOutOfMemoryError となった場合は，スタックトレースは出力されません。 • -XX:-HitachiOutOfMemoryStackTrace： OutOfMemoryError 発生時に，スタックトレースを日立 JavaVM ログファイルに出力しません。 デフォルト値は-XX:-HitachiOutOfMemoryStackTrace です。
8	クラスライブラリトレース機能オプション	-XX:[+|-]HitachiJavaClassLibTrace※2	クラスライブラリのスタックトレースを出力するかどうかを指定します。 • -XX:+HitachiJavaClassLibTrace： クラスライブラリのスタックトレースを出力します。 • -XX:-HitachiJavaClassLibTrace： クラスライブラリのスタックトレースを出力しません。 デフォルト値は-XX:-HitachiJavaClassLibTrace です。
		-XX:HitachiJavaClassLibTraceLineSize=*<整数値>*	クラスライブラリのスタックトレースの 1 行の文字数（単位：バイト）を1024～INT_MAX の整数で指定します。1 行の文字数が指定した文字数を超えた場合は，at 以降

項番	分類	オプション名	説明
8	クラスライブラリトレース機能オプション	-XX:HitachiJavaClassLibTraceLineSize=<*整数値*>	の文字列の前半部分を削除して，指定された文字数分出力します。 デフォルト値は1024 です。 範囲外の値が指定された場合は1024 となります。
9	ローカル変数情報出力機能オプション	-XX:[+|-]HitachiLocalsInStackTrace	スレッドダンプ出力時のスタックトレースに，ローカル変数情報を出力するかどうかを指定します。 • -XX:+HitachiLocalsInStackTrace： スレッドダンプ出力時のスタックトレースに，ローカル変数情報を出力します。 • -XX:-HitachiLocalsInStackTrace： スレッドダンプ出力時のスタックトレースに，ローカル変数情報を出力しません。 デフォルト値は-XX:-HitachiLocalsInStackTrace です。
		-XX:[+|-]HitachiLocalsSimpleFormat	ローカル変数情報出力を簡易フォーマットにするかどうかを指定します。 • -XX:+HitachiLocalsSimpleFormat： ローカル変数情報出力を簡易フォーマットで出力します。 • -XX:-HitachiLocalsSimpleFormat： ローカル変数情報出力を通常フォーマットで出力します。 デフォルト値は-XX:-HitachiLocalsSimpleFormat です。
		-XX:[+|-]HitachiTrueTypeInLocals	ローカル変数情報出力時に，ローカル変数オブジェクトの実際の型名を文字列として出力するかどうかを指定します。 • -XX:+HitachiTrueTypeInLocals： ローカル変数情報に，実際のオブジェクト型名を出力します。 • -XX:-HitachiTrueTypeInLocals： ローカル変数情報に，実際のオブジェクト型名を出力しません。 デフォルト値は-XX:-HitachiTrueTypeInLocals です。

注※1

<*サイズ*>の単位はバイトです。

k またはK を付記するとキロバイトで指定できます。また，m またはM を付記するとメガバイトで指定できます。

(例)

-Xms6291456：6,291,456 バイト

-Xms6144k：6,144 キロバイト

-Xms6m：6 メガバイト

注※2

これらのオプションを指定した場合，日立 JavaVM ログファイルが出力されます。

注※3

-XX:+HitachiVerboseGC オプションが指定されている場合は，このオプションも指定されます。

13 出力されるログファイル

この章では，SDP サーバおよび関連コンポーネントから出力されるログファイルについて説明します。

13.1 SDP サーバのログファイル

記述形式

SDP サーバのログファイルには，メッセージログファイルのヘッダーおよびメッセージ情報が保存されています。ヘッダー情報には，先頭行，タイトル行，およびラベル行が含まれています。メッセージ情報には，メッセージ行が含まれています。各行は，以下のように出力されます。

表 13-1 タイトル行の形式

項目	説明
OS information	ログ出力機能がアクティブな OS の情報を示します。
Time zone	OS のタイムゾーンを示します。
Start time	最初のファイルの場合は，ログ出力機能が使用された時間を示します。 2 ファイル目以降の場合は，ログファイルが切り替わった時間を示します。

ラベル行の形式

yyyy/mm/dd hh:*mm*:*ss.sss pid tid メッセージID メッセージ*(LANG=en) CRLF

表 13-2 メッセージ行の形式

項目	説明
number	メッセージレコードの 4 桁のシリアル番号を示します。
date	メッセージの収集日を示します。(*yyyy/mm/dd*)
time	メッセージの収集時間を示します。(*hh*:*mm*:*ss.sss*)
application-name	HSDP のバージョン番号が含まれる，一意のアプリケーション名を示します。
pid	プロセス ID を示します。
tid	スレッドの識別子を示します。
message-ID	KFSP*nnnnn-x*
message-type	次のどれかの値を示します。 EC：例外の発生 ER：エラーメッセージ スペース：EC やER 以外 なお，内部カスタムアダプター用メッセージログの場合，スペースになります。
message-text	メッセージ ID に対応するメッセージを示します。
linefeed code	Carriage Return and Line Feed (CRLF) を示します。

ファイル名

ログファイルグループ A は CQL エンジン（Java エンジン）を使用した場合のログ，ログファイルグループ B は acceleration CQL エンジンを使用した場合のログを示します。

- SDP サーバ用ログファイル
  ログファイルグループ A
  SDPServerMessage*N*※1.log
  SDPServerTrace*N*※1.log
  ログファイルグループ B
  SDPServerCMessage*N*※1.log
  SDPServerCTrace*N*※1.log
- 内部アダプター用ログファイル
  ログファイルグループ A
  ADP_*XXX*※2-AdaptorMessage*N*※1.log
  ADP_*XXX*※2-AdaptorTrace*N*※1.log
  ログファイルグループ B
  ADP_*XXX*※2-AdaptorCMessage*N*※1.log
  ADP_*XXX*※2-AdaptorCTrace*N*※1.log
- 内部カスタムアダプター用ログファイル
  ログファイルグループ A
  CADP_*YYY*※3-CustomAdaptorMessage*N*※1.log
  CADP_*YYY*※3-CustomAdaptorTrace*N*※1.log

注※1

メッセージログファイルのシリアル番号

注※2

アダプターグループ名

注※3

SDPClientLogger インタフェースのinitialize()メソッドで指定した任意の名前

メモ

例えば，メッセージログファイルが 16 ファイルある場合は，シリアル番号の範囲は1～16 になります。

ファイルの格納先

SDP サーバのログファイル出力用プロパティファイル（logger.properties）のlogger.logfileDir パラメーターで指定した格納先。

logger.logfileDir パラメーターについては，「12.12　SDP サーバのログファイル出力用プロパティファイル（logger.properties）」を参照してください。

説明

メッセージログファイルは，Streaming Data Platform から出力されたメッセージを含んでいます。メッセージログファイルの指定項目について，次の表に示します。ファイル名で説明しているとおり，ログファイルグループ A とログファイルグループ B があります。

表 13-3　メッセージログファイルの指定項目

項目	説明	
	ログファイルグループ A	ログファイルグループ B
ファイルサイズ	ログファイル出力用プロパティファイル（logger.properties）でファイルサイズを指定します。ファイルサイズが最大値に近づくと，次に出力されるメッセージのサイズによっては，指定したサイズを超えるおそれがあります。	ログファイル出力用プロパティファイル（logger.properties）でファイルサイズを指定します。
新しいログファイルへの切り替え時期	出力されるファイルサイズが，規定の最大サイズを超えると，現在のログファイルが切り替えられます。ログファイル数は，ログファイル出力用プロパティファイル（logger.properties）で指定できます。	現在のファイルと出力されるメッセージの合計サイズが，規定の最大サイズを超えると，現在のログファイルが切り替えられます。出力されるファイル数は，ログファイル出力用プロパティファイル（logger.properties）で指定できます。
切り替えたファイルのサイズ	ログファイルが切り替えられると，新しいログファイルのサイズは 0（すべてのログが削除される）になります。	
ログファイルが切り替えられた場合のログの保存	ログファイルが切り替えられると，古いログはすべて削除され，新しいログがファイルの上部から記述されます。	
プロセスの再起動時に選択されるファイル	プロセスが再起動されると，データは最新のタイムスタンプのファイルに出力されます。	
管理ファイル	管理ファイルは作成されません。	

SDP サーバのログファイル出力用プロパティファイルの詳細については，「12.12　SDP サーバのログファイル出力用プロパティファイル（logger.properties）」を参照してください。

指定できるパラメーター

特になし。

例

```
**** Linux 2.6.32-358.el6.x86_64 TZ=GMT 2014/07/15 18:46:05.120
     yyyy/mm/dd hh:mm:ss.sss pid tid メッセージID メッセージ(LANG=en)
0000 2014/07/15 18:46:07.723 SDPServer 0108 01642BD6 0179F36B KFSP81002-I The server has sta
rted.
```

13.2 共通コンポーネント（コマンド・ロガー）のログファイル

記述形式

ログファイルには，メッセージログファイルのヘッダーおよびメッセージ情報が保存されています。ヘッダー情報には，先頭行，タイトル行，およびラベル行が含まれています。メッセージ情報には，メッセージ行が含まれています。各行は，以下のように出力されます。

表 13-4 先頭行の行形式

項目	説明
application-name	HSDP アプリケーションの名前を，HSDP のバージョン番号が含まれる，一意のアプリケーション名で示します。
pid	プロセス ID を示します。
file size	ログ定義ファイルのパラメーターに定義されたファイルサイズを示します。
number of message log files	ログ定義ファイルのパラメーターに定義されたログファイルの数を示します。
log output feature mode	16 進数の数字を示します。
linefeed code	Carriage Return and Line Feed（CRLF）を示します。

表 13-5 タイトル行の形式

項目	説明
OS information	ログ出力機能がアクティブな OS の情報を示します。
Time zone	OS のタイムゾーンを示します。
Start time	最初のファイルの場合は，ログ出力機能が使用された時間を示します。 2 ファイル目以降の場合は，ログファイルが切り替わった時間を示します。

ラベル行の形式

yyyy/mm/dd hh:*mm*:*ss.sss pid tid メッセージID メッセージ*(LANG=en) CRLF

表 13-6 メッセージ行の形式

項目	説明
number	メッセージレコードの 4 桁のシリアル番号を示します。
date	メッセージの収集日を示します。(*yyyy/mm/dd*)
time	メッセージの収集時間を示します。(*hh:mm:ss.sss*)
application-name	HSDP アプリケーションの名前を，HSDP のバージョン番号が含まれる，一意のアプリケーション名で示します。
pid	プロセス ID を示します。

項目	説明
tid	スレッドの識別子を示します。
message-ID	KFHD*nnnnn-x*
message-text	メッセージ ID に対応するメッセージを示します。
linefeed code	Carriage Return and Line Feed（CRLF）を示します。

ファイル名

- hsdpcommandmessage*N*※.log
- hsdpservermessage*N*※.log
- hsdpcommandtrace*N*※.log
- hsdpservertrace*N*※.log

注※

メッセージログファイルのシリアル番号です。

> **メモ**
>
> 16 ファイルある場合，シリアル番号は1～16 になります。

ファイル格納場所

共通コンポーネント（コマンド・ロガー）のログファイル出力用プロパティファイル（hsdplogger.properties）のhsdpLogfileDir パラメーターで指定した格納先。

hsdpLogfileDir パラメーターについては，「12.13　共通コンポーネント（コマンド・ロガー）のログファイル出力用プロパティファイル（hsdplogger.properties）」を参照してください。

説明

共通コンポーネント（コマンド・ロガー）によって出力されるログファイルの詳細を説明します。

共通コンポーネント（コマンド・ロガー）のログファイルは，次の表の項目に基づいて出力されます。

項目	説明
ファイルサイズ	共通コンポーネント（コマンド・ロガー）のログファイル出力用プロパティファイルのパラメーターに定義したファイルサイズで出力されます。
新しいログファイルへの切り替え時期	出力されるファイルサイズが，規定の最大サイズを超えると，現在のログファイルは，別のログファイルに切り替わります。ログファイル数は，共通コンポーネント（コマンド・ロガー）のログファイル出力用プロパティファイルのパラメーターで指定できます。

項目	説明
切り替えたファイルのサイズ	ログファイルが，別のログファイルに切り替えられると，新しいログファイルのサイズは，規定のサイズ（固定サイズ）になります。
ログファイルが切り替えられた場合のログの保存	ログファイルが切り替えられると，ログはファイルの最初から上書きされます。ファイルの終端には，EOF マーカー※1 が出力されます。
プロセスの再起動時に選択されるファイル	プロセスが再起動されると，データは最新のタイムスタンプのファイルに出力されます。
管理ファイル	mmap ディレクトリ※2 に作成される管理ファイルを示します。

注※1

ファイルの終端は次のとおりです。

```
-----------------------< End of Data >-----------------------
```

注※2

デフォルトではログ出力ディレクトリに次のディレクトリが作成されます。

運用ディレクトリ/logs/mmap/

メモ

- ログ出力機能の管理ファイルは，mmap ディレクトリに作成されます。
- 管理ファイル作成後に共通コンポーネント（コマンド・ロガー）のログファイル出力用プロパティファイルのパラメーターを変更したい場合は，ログファイルと管理ファイルを削除するか，ログ出力先ディレクトリ下のログファイルとmmap ディレクトリを別のディレクトリに移動してください。

例

```
**** hsdpstart 01642BD6  0016777216 0004 00000000
**** Linux 2.6.32-358.el6.x86_64               TZ=GMT        2013/12/21 18:46:05.120
     yyyy/mm/dd hh:mm:ss.sss                 pid      tid    メッセージID    メッセージ(LAN
G=en)
0000 2014/01/15 18:46:07.723 hsdpstart 2.1.0 01642BD6 0179F36B  KFHD92010-I Command processi
ng will now start.(command = hsdpstart).
```

13.3 SDP マネージャー，SDP ブローカー，SDP コーディネーターのログファイル

記述形式

ログファイルには，メッセージログファイルのヘッダーおよびメッセージ情報が保存されています。ヘッダー情報には，先頭行，タイトル行，およびラベル行が含まれています。メッセージ情報には，メッセージ行が含まれています。各行は，以下のように出力されます。

表 13-7　先頭行の行形式

項目	説明
application-name	HSDP アプリケーションの名前を，HSDP のバージョン番号が含まれる，一意のアプリケーション名で示します。
pid	プロセス ID を示します。
file size	ログ定義ファイルのパラメーターに定義されたファイルサイズを示します。
number of message log files	ログ定義ファイルのパラメーターに定義されたログファイルの数を示します。
log output feature mode	16 進数の数字を示します。
linefeed code	Carriage Return and Line Feed（CRLF）を示します。

表 13-8　タイトル行の形式

項目	説明
OS information	ログ出力機能がアクティブな OS の情報を示します。
Time zone	OS のタイムゾーンを示します。
Start time	最初のファイルの場合は，ログ出力機能が使用された時間を示します。 2 ファイル目以降の場合は，ログファイルが切り替わった時間を示します。

ラベル行の形式

yyyy/mm/dd　hh:*mm*:*ss.sss　pid　tid　メッセージID　メッセージ*(LANG=en) CRLF

表 13-9　メッセージ行の形式

項目	説明
number	メッセージレコードの 4 桁のシリアル番号を示します。
date	メッセージの収集日を示します。(*yyyy/mm/dd*)
time	メッセージの収集時間を示します。(*hh:mm:ss.sss*)
application-name	HSDP アプリケーションの名前を，HSDP のバージョン番号が含まれる，一意のアプリケーション名で示します。

項目	説明
pid	プロセス ID を示します。
tid	スレッドの識別子を示します。
message-ID	KFHD*nnnnn-x*
message-text	メッセージ ID に対応するメッセージを示します。
linefeed code	Carriage Return and Line Feed（CRLF）を示します。

ファイル名

- SDP ブローカーのログファイルの場合：
  BrokerMessage*N*※.log
  BrokerTrace*N*※.log
- SDP コーディネーターのログファイルの場合：
  CoordinatorMessage*N*※.log
  CoordinatorTrace*N*※.log
- SDP マネージャーのログファイルの場合：
  ManagerMessage*N*※.log
  ManagerTrace*N*※.log

注※

メッセージログファイルのシリアル番号です。

> **メモ**
>
> 16 ファイルある場合，シリアル番号は1～16 になります。

ファイル格納場所

SDP マネージャー定義ファイル（hsdp_setup_manager.cfg）のhsdp_logfile_dir パラメーターで指定した格納先。

hsdp_logfile_dir パラメーターについては，「12.14　SDP マネージャー定義ファイル」を参照してください。

説明

次のコンポーネントによって出力されるログファイルの詳細を説明します。

- SDP ブローカー
- SDP コーディネーター

- SDP マネージャー

これらのコンポーネントのログファイルは，次の表の項目に基づいて出力されます。

項目	説明
ファイルサイズ	SDP マネージャー定義ファイルのパラメーターに定義したファイルサイズで出力されます。
新しいログファイルへの切り替え時期	出力されるファイルサイズが，規定の最大サイズを超えると，現在のログファイルが切り替えられます。ログファイル数は，SDP マネージャー定義ファイルのパラメーターで指定できます。
切り替えたファイルのサイズ	ログファイルが切り替えられると，新しいログファイルのサイズは0（すべてのログが削除される）になります。
ログファイルが切り替えられた場合のログの保存	ログファイルが切り替えられると，古いログはすべて削除され，新しいログがファイルの上部から記述されます。
プロセスの再起動時に選択されるファイル	プロセスが再起動されると，データは最新のタイムスタンプのファイルに出力されます。
管理ファイル	管理ファイルは作成されません。

例

```
**** hsdpstart 01642BD6  0016777216 0004 00000000
**** Linux 2.6.32-358.el6.x86_64                TZ=GMT        2013/12/21 18:46:05.120
     yyyy/mm/dd hh:mm:ss.sss                  pid      tid    メッセージID    メッセージ(LAN
G=en)
0000 2014/01/15 18:46:07.723 hsdpstart 2.1.0 01642BD6 0179F36B  KFHD92010-I Command processi
ng will now start.(command = hsdpstart).
```

13.4 外部アダプターのログファイル

記述形式

ログファイルには，メッセージログファイルのヘッダーおよびメッセージ情報が保存されています。ヘッダー情報には，先頭行，タイトル行，およびラベル行が含まれています。メッセージ情報には，メッセージ行が含まれています。各行は，以下のように出力されます。

表 13-10 先頭行の行形式

項目	説明
application-name	HSDP アプリケーションの名前を，HSDP のバージョン番号が含まれる，一意のアプリケーション名で示します。
pid	プロセス ID を示します。
file size	ログ定義ファイルのパラメーターに定義されたファイルサイズを示します。
number of message log files	ログ定義ファイルのパラメーターに定義されたログファイルの数を示します。
log output feature mode	16 進数の数字を示します。
linefeed code	Carriage Return and Line Feed（CRLF）を示します。

表 13-11 タイトル行の形式

項目	説明
OS information	ログ出力機能がアクティブな OS の情報を示します。
Time zone	OS のタイムゾーンを示します。
Start time	最初のファイルの場合は，ログ出力機能が使用された時間を示します。 2 ファイル目以降の場合は，ログファイルが切り替わった時間を示します。

ラベル行の形式

yyyy/mm/dd hh:*mm*:*ss.sss pid tid メッセージID メッセージ*(LANG=en) CRLF

表 13-12 メッセージ行の形式

項目	説明
number	メッセージレコードの 4 桁のシリアル番号を示します。
date	メッセージの収集日を示します。(*yyyy/mm/dd*)
time	メッセージの収集時間を示します。(*hh:mm:ss.sss*)
application-name	HSDP アプリケーションの名前を，HSDP のバージョン番号が含まれる，一意のアプリケーション名で示します。
pid	プロセス ID を示します。

項目	説明
tid	スレッドの識別子を示します。
message-ID	KFHD*nnnnn-x*
message-text	メッセージ ID に対応するメッセージを示します。
linefeed code	Carriage Return and Line Feed（CRLF）を示します。

ファイル名

- ExAdaptorMessage*N*※.log
- ExAdaptorTrace*N*※.log

注※

メッセージログファイルのシリアル番号です。

> **メモ**
>
> 16 ファイルある場合，シリアル番号は1～16 になります。

ファイル格納場所

外部アダプター定義ファイルのhsdp.logfile.dir パラメーターで指定した格納先。

hsdp.logfile.dir パラメーターについては，「12.15　外部アダプター定義ファイル」を参照してください。

説明

外部アダプターによって出力されるログファイルの詳細を説明します。

外部アダプターのログファイルは，次の表の項目に基づいて出力されます。

項目	説明
ファイルサイズ	外部アダプター定義ファイルのパラメーターに定義したファイルサイズで出力されます。
新しいログファイルへの切り替え時期	出力されるファイルサイズが，規定の最大サイズを超えると，現在のログファイルが切り替えられます。ログファイル数は，外部アダプター定義ファイルのパラメーターで指定できます。
切り替えたファイルのサイズ	ログファイルが切り替えられると，新しいログファイルのサイズは0（すべてのログが削除される）になります。
ログファイルが切り替えられた場合のログの保存	ログファイルが切り替えられると，古いログはすべて削除され，新しいログがファイルの上部から記述されます。

項目	説明
プロセスの再起動時に選択されるファイル	プロセスが再起動されると，データは最新のタイムスタンプのファイルに出力されます。
管理ファイル	管理ファイルは作成されません。

例

```
**** hsdpstart 01642BD6  0016777216 0004 00000000
**** Linux 2.6.32-358.el6.x86_64                TZ=GMT        2013/12/21 18:46:05.120
     yyyy/mm/dd hh:mm:ss.sss                  pid      tid    メッセージID    メッセージ(LAN
G=en)
0000 2014/01/15 18:46:07.723 hsdpstart 2.1.0 01642BD6 0179F36B  KFHD92010-I Command processi
ng will now start.(command = hsdpstart).
```

13.5 SDP マネージャーコマンドのログファイル

記述形式

ログファイルには，メッセージログファイルのヘッダーおよびメッセージ情報が保存されています。ヘッダー情報には，先頭行，タイトル行，およびラベル行が含まれています。メッセージ情報には，メッセージ行が含まれています。各行は，以下のように出力されます。

表 13-13　先頭行の行形式

項目	説明
application-name	HSDP アプリケーションの名前を，HSDP のバージョン番号が含まれる，一意のアプリケーション名で示します。
pid	プロセス ID を示します。
file size	ログ定義ファイルのパラメーターに定義されたファイルサイズを示します。
number of message log files	ログ定義ファイルのパラメーターに定義されたログファイルの数を示します。
log output feature mode	16 進数の数字を示します。
linefeed code	Carriage Return and Line Feed（CRLF）を示します。

表 13-14　タイトル行の形式

項目	説明
OS information	ログ出力機能がアクティブな OS の情報を示します。
Time zone	OS のタイムゾーンを示します。
Start time	最初のファイルの場合は，ログ出力機能が使用された時間を示します。 2 ファイル目以降の場合は，ログファイルが切り替わった時間を示します。

ラベル行の形式

yyyy/mm/dd　hh:*mm*:*ss.sss　pid　tid　メッセージID　メッセージ*(LANG=en) CRLF

表 13-15　メッセージ行の形式

項目	説明
number	メッセージレコードの 4 桁のシリアル番号を示します。
date	メッセージの収集日を示します。(*yyyy/mm/dd*)
time	メッセージの収集時間を示します。(*hh:mm:ss.sss*)
application-name	HSDP アプリケーションの名前を，HSDP のバージョン番号が含まれる，一意のアプリケーション名で示します。
pid	プロセス ID を示します。

項目	説明
tid	スレッドの識別子を示します。
message-ID	KFHD*nnnnn-x*
message-text	メッセージ ID に対応するメッセージを示します。
linefeed code	Carriage Return and Line Feed（CRLF）を示します。

ファイル名

hsdpmanagercommandmessage*N*※.log

hsdpmanagercommandtrace*N*※.log

注※

メッセージログファイルのシリアル番号です。

> **メモ**
>
> 16 ファイルある場合，シリアル番号は1～16 になります。

ファイル格納場所

SDP マネージャーコマンドのログ定義ファイル（hsdpmanagerlogger.properties）のhsdpLogfileDir パラメーターで指定した格納先。

hsdpLogfileDir パラメーターについては，「12.16　SDP マネージャーコマンドのログ定義ファイル」を参照してください。

説明

ここでは，次のコンポーネントに関連するコマンド（hsdpmanager）によって出力されるログファイルについて説明します。

- SDP マネージャー
- SDP ブローカー
- SDP コーディネーター

これらのコンポーネントのログファイルは，次の表の項目に基づいて出力されます。

表 13-16　SDP マネージャーコマンドのログファイル

項目	説明
ファイルサイズ	SDP マネージャーコマンドのログ定義ファイルのパラメーターに定義されたファイルサイズで出力されます。

項目	説明
新しいログファイルへの切り替え時期	出力されるファイルサイズが，規定の最大サイズを超えると，現在のログファイルは，別のログファイルに切り替わります。ログファイル数は，SDP マネージャーコマンドのログ定義ファイルのパラメーターで指定できます。
切り替えたファイルのサイズ	ログファイルが，別のログファイルに切り替えられると，新しいログファイルのサイズは，規定のサイズ（固定サイズ）になります。
ログファイルが切り替えられた場合のログの保存	ログファイルが切り替えられると，ログはファイルの最初から上書きされます。ファイルの終端には，EOF マーカー※1 が出力されます。
プロセスの再起動時に選択されるファイル	プロセスが再起動されると，データは最新のタイムスタンプのファイルに出力されます。
管理ファイル	mmap ディレクトリ※2 に作成される管理ファイルを示します。

注※1

ファイルの終端は次のとおりです。

```
-----------------------< End of Data >-----------------------
```

注※2

デフォルトではログ出力ディレクトリに次のディレクトリが作成されます。

```
var/log/hitachi/hsdp/mmap/
```

メモ

- ログ出力機能の管理ファイルは，mmap ディレクトリに作成されます。
- 管理ファイル作成後に SDP マネージャーコマンドのログ定義ファイルのパラメーターを変更したい場合は，ログファイルと管理ファイルを削除するか，ログ出力先ディレクトリ下のログファイルとmmap ディレクトリを別のディレクトリに移動してください。

例

```
**** hsdpmanager 01642BD6  0016777216 0004 00000000
**** Linux 2.6.32-358.el6.x86_64                TZ=GMT        2013/12/21 18:46:05.120
     yyyy/mm/dd hh:mm:ss.sss                  pid      tid    メッセージID    メッセージ(LAN
G=en)
0000 2014/01/15 18:46:07.723 hsdpstart 2.1.0 01642BD6 0179F36B KFHD92010-I Command processin
g will now start.(command = hsdpmanager).
```

13.6 運用ディレクトリのセットアップログファイル

記述形式

```
***** Setup at タイムスタンプ *****
キー=値
```

- *タイムスタンプ*

  設定が更新された時刻を示します。タイムスタンプは次のように出力されます。

  yyyy/*mm*/*dd*hh:*mm*:*ss*

  (*yyyy*：年，*mm*：月，*dd*：日，*hh*：時，*mm*：分， *ss*：秒)

- *キー=値*

  次のように，キー値形式で各行に出力される設定を示します。

キー	*値*
user_id	hsdpsetup コマンドを実行したユーザーの名前を出力します。
working_directory	運用ディレクトリのパスを出力します。この値は，-dir オプションの引数に指定された値に対応します。
server_cluster_name	運用ディレクトリで動作する SDP サーバが属するサーバクラスタの名前を出力します。この値は，-sc オプションの引数に指定された値に対応します。-sc オプションの引数を指定していない場合は，My_server_cluster が出力されます。

ファイル名

```
hsdpsetup.log
```

ファイル格納場所

```
運用ディレクトリ/logs/
```

説明

運用ディレクトリの設定がこのファイルに出力されます。hsdpsetup コマンドを使用して運用ディレクトリを設定するたびに，新しい設定がファイルの末尾に追加されます。ファイルのサイズは任意です。

指定できるパラメーター

なし

例

```
***** Setup at 2015/12/18 15:39:03 *****
user_id=hsdp
```

```
working_directory=/home/hsdp/wd
server_cluster_name=My_server_cluster
```

13.7 SDP マネージャーセットアップログファイル (hsdpsetupmanager.log)

記述形式

```
***** Manager 更新契機 Definition at タイムスタンプ *****
キー=値
```

- *更新契機*

  セットアップ内容が更新された契機を表します。次の契機があります。

  Manual：hsdpsetup コマンドの実行による更新

- *タイムスタンプ*

  セットアップ内容が更新された時刻を表します。タイムスタンプのフォーマットは次のとおりです。

  yyyy/mm/dd hh:mm:ss（*yyyy*：年，*mm*：月，*dd*：日，*hh*：時， *mm*：分，*ss*：秒）

- *キー=値*

  セットアップ内容を表します。1 行ごとにキー値形式で出力します。

キー	*値*
hsdp_host	外部アダプターまたは内部カスケーディングアダプターがストリームに接続する際に使用するホスト名，または IP アドレスを出力します。この値は，-host オプションのオプション引数で指定した値に対応します。-host オプションが省略されている場合は，localhost を出力します。
hsdp_port	HSDP が使用するポート群の先頭ポート番号を出力します。-port オプションのオプション引数で指定した値に対応します。-port オプションが省略されている場合は，「20425～20432」を出力します。
hsdp_coordinator_group	コーディネーターグループに所属する SDP コーディネーターのホストのホスト名または IP アドレスを出力します。-chosts オプションのオプション引数で指定した値に対応します。-chosts オプションが省略されている場合は，「localhost」を出力します。
hsdp_coordinator_multiplicity	コーディネーターグループで保持するデータの多重度を出力します。-cmulti オプションのオプション引数で指定した値に対応します。-cmulti オプションが省略されている場合は，「1」を出力します。
hsdp_coordinator_multicast_address	コーディネーターグループに所属する SDP コーディネーターが使用するマルチキャストアドレスを出力します。-cmaddr オプションのオプション引数で指定した値に対応します。-cmaddr オプションが省略されている場合は，「239.255.2.1」を出力します。

ファイル名

hsdpsetupmanager.log

ファイル格納先

/var/log/hitachi/hsdp/

説明

SDP マネージャー，SDP ブローカー，および SDP コーディネーターのセットアップ内容がこのファイルに出力されます。hsdpsetup コマンドで SDP マネージャー，SDP ブローカー，および SDP コーディネーターをセットアップするたび，セットアップ内容がファイルの末尾に追加されます。ファイルサイズに制限はありません。SDP コーディネーターを動的に追加した場合は，追加によって更新されたセットアップ内容が，SDP マネージャーによってファイルの末尾に自動で追加されます。

指定できるパラメーター

特になし。

例

```
***** Manager Manual Definition at 2014/12/18 15:49:29 *****
hsdp_host=127.0.0.1
hsdp_port=20425-20432
hsdp_coordinator_group=127.0.0.1
hsdp_coordinator_multiplicity=1
hsdp_coordinator_multicast_address=239.255.2.1
```

13.8　エクスポートログファイル（hsdpexport.log）

記述形式

メッセージ

パス

メッセージ：エクスポート対象のファイルが存在しない場合に，コンソールに出力したメッセージを表示します。

パス：tar 形式で圧縮されたディレクトリ，およびファイルパスの一覧を表示します。1 行ごとに 1 ファイルまたは 1 ディレクトリが出力されます。ファイルやディレクトリのパスはtar ファイルに格納した形式で出力されます。

ファイル名

hsdpexport.log

ファイル格納先

運用ディレクトリ/logs/

説明

hsdpexport コマンドによって出力されます。次の内容が格納されます。

- エクスポート対象のファイルが存在しない場合にコンソールに出力したメッセージ
- tar 形式で圧縮されたディレクトリおよびファイルパスの一覧

hsdpexport コマンドの実行時にエクスポートログファイルがすでに存在していた場合，そのファイルを上書きします。

指定できるパラメーター

特になし。

例

```
KFHD99104-W An invalid file path was specified. The file path was removed,
    and data export started. (specified file path =
    ./conf/QueryGroupTest.qg)KFHD99104-W An invalid file path was specified. The file path w
as removed, and data export
    started. (specified file path = ./query/QueryGroupTest.cql) ./conf/./conf/xml/./conf/xml
/adp_input.xml./conf/xml/adp_output.xml /var/log/hitachi/hsdp/hsdpsetupmanager.log
```

14 アダプター構成定義ファイル

この章では，標準提供アダプターを使用する場合に作成する，アダプター構成定義ファイルについて説明します。

カスタムアダプターを使用する場合は，アダプター構成定義ファイルを作成する必要はありません。

14.1　アダプター構成定義ファイルの説明の記述形式

この章では，アダプター構成定義ファイルの説明を次の形式で記述します。ただし，ファイルによっては説明しない項目もあります。また，各ファイルの固有情報を記載している場合があります。

記述形式

定義の記述形式を示します。

ファイル名

ファイル名を示します。

ファイルの格納先

ファイルの格納先を示します。

定義の詳細

定義するタグの詳細について説明します。

記述例

各定義の記述例を示します。

14.2　アダプター構成定義ファイル作成上の注意事項

アダプター構成定義ファイルを作成する際の注意事項を次に示します。

- アダプター構成定義ファイルは**XML1.0** 形式で記述します。記述形式の仕様については，W3C による XML の仕様書（『Extensible Markup Language (XML) 1.0』）を参照してください。
- アダプター構成定義ファイルの中で特殊文字（記号）を使用する場合は，サニタイジング（特殊文字の無効化）が必要です。次の表に示す対応に従って，特殊文字を置換して記述してください。

サニタイジングが必要な特殊文字	置換後の文字
<	<
>	>
&	&
”	"
’	'または'

14.3　アダプター構成定義ファイル

アダプター構成定義ファイルでは，アダプターグループやコネクターなど，標準提供アダプターの構成について定義します。

ここでは，アダプター構成定義ファイルの概要と，アダプター構成定義ファイルの名前空間 URI について説明します。定義ファイルの各定義の詳細は，「14.4　アダプター構成定義ファイルの共通定義」以降で説明します。また，定義ファイルの記述例については，「14.11　アダプター構成定義ファイルの記述例」で説明します。

14.3.1　アダプター構成定義ファイルの概要

記述形式

記述形式は，「14.4　アダプター構成定義ファイルの共通定義」以降で，定義ごとに説明します。

ファイル名

任意の名前

ファイルの格納先

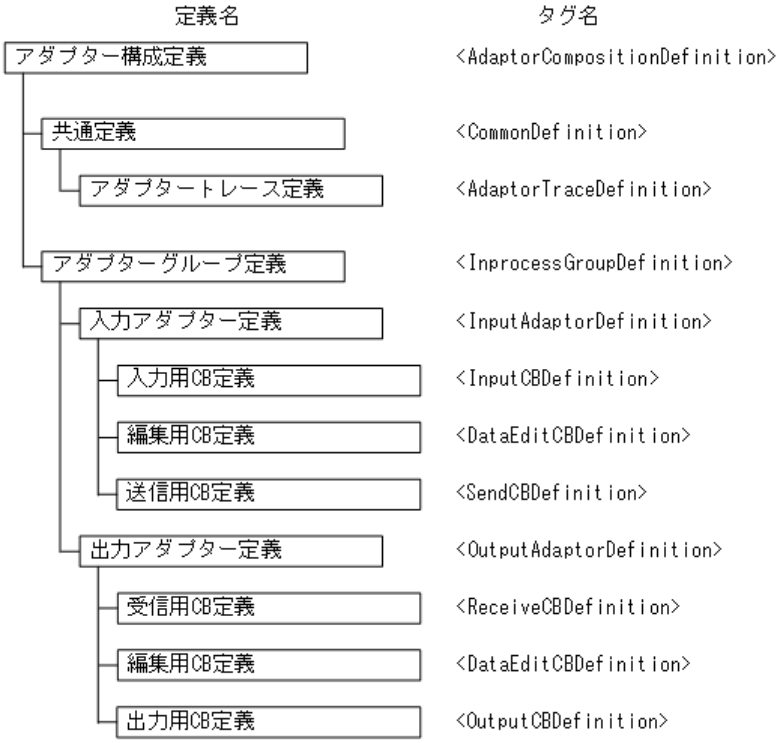
このファイルは，必ず次のディレクトリに格納してください。

運用ディレクトリ/conf/xml/

定義の詳細

アダプター構成定義ファイルの定義は，階層構造になっています。アダプター構成定義ファイルの構造を次の図に示します。

図 14-1　アダプター構成定義ファイルの構造



階層構造になっている定義は，親子関係にあります。例えば，入力アダプター定義は，アダプターグループ定義の**子要素**として，アダプターグループ定義のタグ内に定義します。また，アダプターグループ定義は，入力アダプター定義の**親要素**となります。

アダプター構成定義ファイルの定義の一覧を次の表に示します。それぞれの定義の詳細は，表中の参照先で説明します。

表 14-1　アダプター構成定義ファイルの定義の一覧

項番	定義		説明	参照先
1	共通定義 (CommonDefinition)	アダプタートレース定義 (AdaptorTraceDefinition タグ)	すべての標準提供アダプターで共通の情報を定義します。	14.4　アダプター構成定義ファイルの共通定義
2	アダプターグループ定義	インプロセスグループ定義 (InprocessGroupDefinition タグ)	インプロセス連携で起動する入力アダプター，または出力アダプターを定義します。	14.5　アダプター構成定義ファイルのアダプターグループ定義
3	アダプター定義	入力アダプター定義 (InputAdaptorDefinition タグ)	入力アダプターを構成するコールバックを定義します。 この定義は，入力アダプターごとに定義します。	14.5.2　アダプター構成定義ファイルのアダプター定義

項番	定義			説明	参照先
4	アダプター定義		出力アダプター定義 (OutputAdaptorDefinition タグ)	出力アダプターを構成するコールバックを定義します。 この定義は，出力アダプターごとに定義します。	14.5.2　アダプター構成定義ファイルのアダプター定義
5	CB 定義	入力用 CB 定義 (InputCBDefinition タグ)	ファイル入力コネクター定義 (FileInputConnectorDefinition タグ)	ファイルを入力する場合に使用する，ファイル入力コネクターの処理について定義します。	14.6　アダプター構成定義ファイルの CB 定義 14.7　アダプター構成定義ファイルの入出力用 CB 定義
6			HTTP パケット入力コネクター定義 (HttpPacketInputConnectorDefinition タグ)	HTTP パケットを入力する場合に使用する，HTTP パケット入力コネクターの処理について定義します。	
7			TCP データ入力コネクター定義 (TcpDataInputConnectorDefinition タグ)	TCP データを入力する場合に使用する，TCP データ入力コネクターの処理について定義します。	
8		出力用 CB 定義 (OutputCBDefinition タグ)	ファイル出力コネクター定義 (FileOutputConnectorDefinition タグ)	ファイルに出力する場合に使用する，ファイル出力コネクターの処理について定義します。	
9			ダッシュボード出力コネクター定義 (DashboardOutputConnectorDefinition タグ)	ダッシュボードへ出力する場合に使用する，ダッシュボード出力コネクターの処理について定義します。	
10		編集用 CB 定義 (DataEditCBDefinition タグ)	フォーマット変換定義 (FormatDefinition タグ)	ファイルの入力，または出力を使用する場合に実施する，フォーマット変換の処理について定義します。	14.6　アダプター構成定義ファイルの CB 定義 14.8　アダプター構成定義ファイルのデータ編集用 CB 定義
11			マッピング定義 (MappingDefinition タグ)	マッピングの処理について定義します。	
12			フィルター定義 (FilterDefinition タグ)	レコードのフィルタリングの処理について定義します。	
13			レコード抽出定義 (RecordExtractionDefinition タグ)	レコードの抽出の処理について定義します。	
14		送信用 CB 定義 (SendCBDefinition タグ)	入力ストリーム定義 (streamInfo タグ)	入力アダプターが接続する入力ストリームについて定義します。	14.6　アダプター構成定義ファイルの CB 定義

項番	定義			説明	参照先
15	CB 定義	受信用 CB 定義（ReceiveCBDefinition タグ）	出力ストリーム定義（streamInfo タグ）	出力アダプターが接続する出力ストリームについて定義します。	14.9　アダプター構成定義ファイルの送受信用 CB 定義

CB 定義の定義の順序

CB 定義は，入力アダプター定義の場合と，出力アダプター定義の場合とで，定義の順序が異なります。それぞれの場合の定義の順序を次に示します。

入力アダプター定義の場合の定義の順序

1. 入力用 CB 定義（ファイル入力コネクター定義など）
2. 編集用 CB 定義（フォーマット変換定義など）
3. 送信用 CB 定義（入力ストリーム定義）

出力アダプター定義の場合の定義の順序

1. 受信用 CB 定義（出力ストリーム定義）
2. 編集用 CB 定義（フォーマット変換定義など）
3. 出力用 CB 定義（ファイル出力コネクター定義など）

また，編集用 CB 定義については，次に示すように，使用する機能に応じて定義します。

- 標準提供アダプターで入出力するデータの種類によって，編集用 CB 定義で定義する内容が異なります。例えば，フォーマット変換定義は，ファイルの入力，またはファイルへの出力の場合しか定義しません。
- フィルター定義とレコード抽出定義は，任意で実行する処理です。必要に応じて定義します。

CB 定義で定義するコールバックの処理については，「16.　標準提供アダプターでのデータ処理，および定義ファイルの設定」を参照してください。

14.3.2　アダプター構成定義ファイルの名前空間 URI

アダプター構成定義ファイルの名前空間 URI について説明します。

アダプター構成定義ファイルでは，**名前空間 URI** は，次の形式で表現されます。

`http://www.hitachi.co.jp/soft/xml/sdp/adaptor/`*`xxx`*

形式の「*xxx*」の部分は，アダプター構成定義ファイルの定義によって異なります。名前空間は，指定する定義に応じて宣言してください。定義と「*xxx*」の部分に記述する名前空間の対応を次の表に示します。

表 14-2　定義と名前空間の対応

項番	定義	名前空間
1	アダプター構成定義	definition
2	共通定義	definition/common
3	アダプタートレース定義	
4	インプロセスグループ定義	definition/adaptor
5	入力アダプター定義	
6	出力アダプター定義	
7	入力用 CB 定義	definition/callback
8	出力用 CB 定義	
9	編集用 CB 定義	
10	送信用 CB 定義	
11	受信用 CB 定義	
12	ファイル入力コネクター定義	definition/callback/FileInputConnectorDefinition
13	HTTP パケット入力コネクター定義	definition/callback/HttpPacketInputConnectorDefinition
14	TCP データ入力コネクター定義	definition/callback/TcpDataInputConnectorDefinition
15	ファイル出力コネクター定義	definition/callback/FileOutputConnectorDefinition
16	ダッシュボード出力コネクター定義	definition/callback/DashboardOutputConnectorDefinition
17	カスケーディングクライアントコネクター定義	definition/callback/CascadingClientConnectorDefinition
18	フォーマット変換定義	definition/callback/FormatDefinition
19	マッピング定義	definition/callback/MappingDefinition
20	フィルター定義	definition/callback/FilterDefinition
21	レコード抽出定義	definition/callback/RecordExtractionDefinition
22	送信コネクター定義	definition/callback/SendConnectorDefinition
23	受信コネクター定義	definition/callback/ReceiveConnectorDefinition

14.3.3　CB 定義クラス名

CB 定義のクラス名ごとの形式について説明します。

CB 定義のクラス名は次の形式で表現されます。

```
jp.co.Hitachi.soft.sdp.adaptor.callback.xxx
```

この形式では，*xxx* は CB 定義の定義ごとに変化します。指定した定義に合った，クラス名を宣言する必要があります。定義と *xxx* に指定するクラス名の一覧を次の表に示します。

定義		名前空間
入力用 CB 定義	ファイル入力コネクター定義	io.FileInputCBImpl
	HTTP パケット入力コネクター定義	io.packetinput.HttpPacketInputCBImpl
	TCP データ入力コネクター定義	io.tcpinput.TcpDataInputCBImpl
出力用 CB 定義	ダッシュボード出力コネクター定義	io.dashboard.DashboardOutputCBImpl
	ファイル出力コネクター定義	io.FileOutputCBImpl
	カスケーディングクライアントコネクター定義	io.cascading.CascadingOutputCBImpl
編集用 CB 定義	フォーマット変換定義	入力アダプター定義 dataedit.formattranslate.InputFormatTranslatorCBImpl 出力アダプター定義 dataedit.formattranslate.OutputFormatTranslatorCBImpl
	マッピング定義	入力アダプター定義 dataedit.mapping.InputMappingCBImpl 出力アダプター定義 dataedit.mapping.OutputMappingCBImpl
	フィルター定義	dataedit.filter.FilterCBImpl
	レコード抽出定義	dataedit.recordextract.RecordExtractionCBImpl
送信用 CB 定義	送信コネクター定義	sendreceive.SendConnectorCBImpl
受信用 CB 定義	受信コネクター定義	CQL エンジンから CQL エンジン用内部出力アダプターに出力する場合 sendreceive.ReceiveConnectorCBImpl acceleration CQL エンジンから CQL エンジン用内部出力アダプターに出力する場合 sendreceive.ExternalReceiveConnectorCBImpl

14.4　アダプター構成定義ファイルの共通定義

ここでは，アダプター構成定義ファイルの共通定義について説明します。共通定義の一覧を次の表に示します。

表 14-3　共通定義の一覧

項番	共通定義	親要素	参照先
1	共通定義（CommonDefinition タグ）	アダプター構成定義	14.4.1　共通定義
2	アダプタートレース定義（AdaptorTraceDefinition タグ）	共通定義	なし

14.4.1　共通定義

説明

この定義は 1 個だけ記述できます。この定義は省略できません。

記述形式

```
<CommonDefinition>
  <adaptorTraceDefinition/>
</CommonDefinition>
```

定義の詳細

CommonDefinition タグ（全体情報の定義）

　共通定義の全体情報を定義します。

14.5　アダプター構成定義ファイルのアダプターグループ定義

ここでは，アダプター構成定義ファイルのアダプターグループ定義（InprocessGroupDefinition タグ）について説明します。アダプターグループ定義の一覧を次の表に示します。

表 14-4　アダプターグループ定義の一覧

項番	アダプターグループ定義	親要素	参照先
1	インプロセスグループ定義（InprocessGroupDefinition タグ）	アダプター構成定義	14.5.1　インプロセスグループ定義

14.5.1　インプロセスグループ定義

説明

インプロセスグループ定義（InprocessGroupDefinition タグ）では，インプロセス連携で起動する入力アダプター，または出力アダプターを定義します。この定義は 1 個だけ記述できます。

記述形式

```
<InprocessGroupDefinition name="アダプターグループ名"
 dashboardPortNo="RMIサーバのポート番号">
  アダプター定義
</InprocessGroupDefinition>
```

定義の詳細

InprocessGroupDefinition タグ（全体情報の定義）

インプロセスグループ定義の全体情報を定義します。

name="*アダプターグループ名*"

アダプターグループを識別するための名称を1～32 文字の半角英数字，またはアンダーライン（_）で指定します。先頭の文字に指定できるのは，半角英字だけです。この属性は省略できません。

ここで指定するアダプターグループ名は，インプロセス連携用プロパティファイルのファイル名に指定するアダプターグループ名と一致させてください。

dashboardPortNo="*RMIサーバのポート番号*"

ダッシュボード出力コネクターが使用する RMI サーバのポート番号を1～65535 の整数で指定します。省略した場合，20421 が仮定されます。

アダプター定義

アダプター定義については，「14.5.2　アダプター構成定義ファイルのアダプター定義」を参照してください。

14.5.2　アダプター構成定義ファイルのアダプター定義

ここでは，アダプター構成定義ファイルのアダプター定義について説明します。

アダプター定義は，「14.5　アダプター構成定義ファイルのアダプターグループ定義」で説明したアダプターグループ定義（InprocessGroupDefinition タグ）の子要素として定義します。

アダプター定義の一覧を次の表に示します。

表 14-5　アダプター定義の一覧

項番	アダプター定義	親要素	参照先
1	入力アダプター定義 （InputAdaptorDefinition タグ）	アダプターグループ定義	14.5.3　入力アダプター定義
2	出力アダプター定義 （OutputAdaptorDefinition タグ）		14.5.4　出力アダプター定義

14.5.3　入力アダプター定義

説明

入力アダプター定義は，「14.5.2　アダプター構成定義ファイルのアダプター定義」で説明したアダプターグループ定義（InprocessGroupDefinition タグ）の子要素として定義します。

入力アダプター定義は，64 個まで記述できます。この定義は省略できます。

記述形式

```
<InputAdaptorDefinition name="アダプター名"
  interval="アダプターの実行間隔"
  charCode=" {SJIS | MS932 | EUC-JP | UTF-8 | UTF-8-BOM | UTF-16BE | UTF-16BE-BOM | UTF-16LE | UTF-1
6LE-BOM} "
  lineFeed=" {CR_LF | LF} " language=" {C | Java} ">
  CB定義
</InputAdaptorDefinition>
```

定義の詳細

InputAdaptorDefinition タグ（全体情報の定義）

入力アダプター定義の全体情報を定義します。

name="*アダプター名*"

入力アダプターを識別するための名称を 1～100 文字の半角英数字，またはアンダーライン（_）で指定します。先頭の文字に指定できるのは，半角英字だけです。この属性は省略できません。アダプター名は，アダプターグループ定義内で一意となるように指定してください。

interval="*アダプターの実行間隔*"

入力アダプターを実行する間隔（単位：ミリ秒）を0～60000の整数で指定します。

入力アダプター内に定義されている最後のコールバックが終了してから最初のコールバックを起動するまでの間，指定した間隔で入力アダプターの処理が停止します。0を指定した場合は停止しません。省略した場合，0が仮定されます。

charCode="{SJIS | MS932 | EUC-JP | UTF-8 | UTF-8-BOM | UTF-16BE | UTF-16BE-BOM | UTF-16LE | UTF-16LE-BOM}"

入力元で扱われている文字コードを指定します。省略した場合，MS932が仮定されます。

文字コードとcharCode属性に指定する値の対応を次の表に示します。

文字コード	charCode属性の指定値
シフトJIS	SJIS
MS932	MS932
EUC	EUC-JP
UTF-8（BOMなし）	UTF-8
UTF-8（BOMあり）	UTF-8-BOM
UTF-16（ビッグエンディアンBOMなし）	UTF-16BE
UTF-16（ビッグエンディアンBOMあり）	UTF-16BE-BOM
UTF-16（リトルエンディアンBOMなし）	UTF-16LE
UTF-16（リトルエンディアンBOMあり）	UTF-16LE-BOM

lineFeed="{CR_LF | LF}"

入力元の改行コードを指定します。省略した場合，CR_LFが仮定されます。

指定できる値を次に示します。

CR_LF：CR（Carriage Return）とLF（Line Feed）の組み合わせを改行として扱います。

LF：LF（Line Feed）を改行として扱います。

language="{C | Java}"

入力アダプターが接続するストリームのタイプを指定します。省略した場合，Javaが仮定されます。

CB定義

指定できるCB定義を次に示します。

入力用CB定義

編集用CB定義

送信用CB定義

CB定義については，「14.6　アダプター構成定義ファイルのCB定義」を参照してください。

14.5.4　出力アダプター定義

説明

出力アダプター定義は，「14.5　アダプター構成定義ファイルのアダプターグループ定義」で説明したアダプターグループ定義（InprocessGroupDefinition タグ）の子要素として定義します。

出力アダプター定義は，64 個まで記述できます。この定義は省略できます。

記述形式

```
<OutputAdaptorDefinition name="アダプター名"
  interval="アダプターの実行間隔"
  charCode=" {SJIS | MS932 | EUC-JP | UTF-8 | UTF-8-BOM | UTF-16BE | UTF-16BE-BOM | UTF-16LE | UTF-16LE-BOM} "
  lineFeed=" {CR_LF | LF} " language=" {C | Java} ">
  CB定義
</OutputAdaptorDefinition>
```

定義の詳細

OutputAdaptorDefinition タグ（全体情報の定義）

出力アダプター定義の全体情報を定義します。

name="*アダプター名*"

出力アダプターを識別するための名称を1～100 文字の半角英数字，またはアンダーライン（_）で指定します。先頭の文字に指定できるのは，半角英字だけです。この属性は省略できません。アダプター名は，アダプターグループ定義内で一意となるように指定してください。

interval="*アダプターの実行間隔*"

出力アダプターを実行する間隔（単位：ミリ秒）を0～60000 の整数で指定します。

出力アダプター内に定義されている最後のコールバックが終了してから最初のコールバックを起動するまでの間，指定した間隔で出力アダプターの処理が停止します。0 を指定した場合は停止しません。省略した場合，0 が仮定されます。

charCode=" {SJIS | MS932 | EUC-JP | UTF-8 | UTF-8-BOM | UTF-16BE | UTF-16BE-BOM | UTF-16LE | UTF-16LE-BOM} "

出力先で扱われている文字コードを指定します。省略した場合，MS932 が仮定されます。文字コードとcharCode 属性に指定する値の対応については，「14.5.3　入力アダプター定義」のcharCode 属性の説明を参照してください。

lineFeed=" {CR_LF | LF} "

出力先の改行コードを指定します。省略した場合，CR_LF が仮定されます。

指定できる値を次に示します。

CR_LF：CR（Carriage Return）と LF（Line Feed）の組み合わせを改行として扱います。

LF：LF（Line Feed）を改行として扱います。

language=”{C | Java}”

出力アダプターが接続するストリームのタイプを指定します。省略した場合，Java が仮定されます。

CB 定義

指定できる CB 定義を次に示します。

受信用 CB 定義

編集用 CB 定義

出力用 CB 定義

CB 定義については，「14.6　アダプター構成定義ファイルの CB 定義」を参照してください。

14.6　アダプター構成定義ファイルの CB 定義

ここでは，アダプター構成定義ファイルの CB 定義について説明します。

CB 定義では，標準提供アダプターのデータの入出力，データの編集，およびタプルの送受信の各処理を実施する，コールバックについて定義します。

CB 定義は，次に示す定義の子要素として定義します。

- 「14.5.3　入力アダプター定義」で説明した入力アダプター定義（InputAdaptorDefinition タグ）
- 「14.5.4　出力アダプター定義」で説明した出力アダプター定義（OutputAdaptorDefinition タグ）

CB 定義の一覧を次の表に示します。

表 14-6　CB 定義の一覧

項番	CB 定義	親要素	参照先
1	入力用 CB 定義（InputCBDefinition タグ）	入力アダプター定義	14.6.1　入力用 CB 定義
2	出力用 CB 定義（OutputCBDefinition タグ）	出力アダプター定義	14.6.2　出力用 CB 定義
3	編集用 CB 定義（DataEditCBDefinition タグ）	入力アダプター定義，または出力アダプター定義	14.6.3　編集用 CB 定義
4	送信用 CB 定義（SendCBDefinition タグ）	入力アダプター定義	14.6.4　送信用 CB 定義
5	受信用 CB 定義（ReceiveCBDefinition タグ）	出力アダプター定義	14.6.5　受信用 CB 定義

なお，CB 定義は，入力アダプター定義の場合と，出力アダプター定義の場合とで，定義の順序が異なります。CB 定義の定義の順序については，「14.3.1　アダプター構成定義ファイルの概要」の「CB 定義の定義の順序」を参照してください。

14.6.1　入力用 CB 定義

説明

入力用 CB 定義（InputCBDefinition タグ）は，「14.5.3　入力アダプター定義」で説明した入力アダプター定義（InputAdaptorDefinition タグ）の子要素として定義します。

記述形式

```
<InputCBDefinition class="クラス名"
  name="コールバック名"
  interval="コールバックの実行間隔">
  入力コネクター定義
</InputCBDefinition>
```

定義の詳細

InputCBDefinition タグ（全体情報の定義）

入力用 CB 定義の全体情報を定義します。

class="*クラス名*"

入力用 CB 定義の機能が実装されているクラス名を指定します。この値は固定値です。

クラス名は，*入力コネクター定義*で指定する入力コネクター定義の種類ごとに異なります。指定するクラス名を次の表に示します。

入力コネクター定義の種類	class 属性の値
ファイル入力コネクター定義	jp.co.Hitachi.soft.sdp.adaptor.callback.io.FileInputCBImpl
HTTP パケット入力コネクター定義	jp.co.Hitachi.soft.sdp.adaptor.callback.io.packetinput.HttpPacketInputCBImpl
TCP データ入力コネクター定義	jp.co.Hitachi.soft.sdp.adaptor.callback.io.tcpinput.TcpDataInputCBImpl

name="*コールバック名*"

機能を識別するためのコールバック名を 1～100 文字の半角英数字，およびアンダーライン（_）で指定します。先頭の文字に指定できるのは，半角英字だけです。この属性は省略できません。コールバック名にはほかの CB 定義と重複した値を指定できますが，メッセージログで入力アダプターの処理の識別情報として出力するため，できるだけ重複しないように指定してください。

interval="*コールバックの実行間隔*"

コールバックの実行間隔（単位：ミリ秒）を0～60000 の整数で指定します。コールバックを実行したあと，指定した間隔で入力アダプターの処理が停止します。0 の場合は停止しません。省略した場合，0 が設定されます。

入力コネクター定義

指定できる CB 定義を次に示します。

ファイル入力コネクター定義

HTTP パケット入力コネクター定義

TCP データ入力コネクター定義

CB 定義については，「14.7　アダプター構成定義ファイルの入出力用 CB 定義」を参照してください。

14.6.2　出力用 CB 定義

説明

出力用 CB 定義（OutputCBDefinition タグ）は，「14.5.4　出力アダプター定義」で説明した出力アダプター定義（OutputAdaptorDefinition タグ）の子要素として定義します。

記述形式

```
<OutputCBDefinition class="クラス名"
  name="コールバック名"
  interval="コールバックの実行間隔">
  出力コネクター定義
</OutputCBDefinition>
```

定義の詳細

OutputCBDefinition タグ（全体情報の定義）

出力用 CB 定義の全体情報を定義します。

class="*クラス名*"

出力用 CB 定義の機能が実装されているクラス名を指定します。この値は固定値です。

クラス名は，*出力コネクター定義*で指定する出力コネクター定義の種類ごとに異なります。指定するクラス名を次の表に示します。

出力コネクター定義の種類	class 属性の値
ファイル出力コネクター定義	jp.co.Hitachi.soft.sdp.adaptor.callback.io.FileOutputCBImpl
ダッシュボード出力コネクター定義	jp.co.Hitachi.soft.sdp.adaptor.callback.io.dashboard.DashboardOutputCBImpl
カスケーディングクライアントコネクター定義	jp.co.Hitachi.soft.sdp.adaptor.callback.io.cascading.CascadingOutputCBImpl

name="*コールバック名*"

機能を識別するためのコールバック名を 1～100 文字の半角英数字，およびアンダーライン（_）で指定します。先頭の文字に指定できるのは，半角英字だけです。この属性は省略できません。コールバック名にはほかの CB 定義と重複した値を指定できますが，メッセージログで出力アダプターの処理の識別情報として出力するため，できるだけ重複しないように指定してください。

interval="*コールバックの実行間隔*"

コールバックの実行間隔（単位：ミリ秒）を0～60000 の整数で指定します。コールバックを実行したあと，指定した間隔で出力アダプターの処理が停止します。0 の場合は停止しません。省略した場合，0 が設定されます。

出力コネクター定義

指定できる CB 定義を次に示します。

ファイル出力コネクター定義

ダッシュボード出力コネクター定義

カスケーディングクライアントコネクター定義

CB 定義については，「14.7　アダプター構成定義ファイルの入出力用 CB 定義」を参照してください。

14.6.3　編集用 CB 定義

説明

編集用 CB 定義（DataEditCBDefinition タグ）は，次に示す定義の子要素として定義します。

- 「14.5.3　入力アダプター定義」で説明した入力アダプター定義（InputAdaptorDefinition タグ）
- 「14.5.4　出力アダプター定義」で説明した出力アダプター定義（OutputAdaptorDefinition タグ）

記述形式

```
<DataEditCBDefinition class="クラス名"
  name="コールバック名"
  interval="コールバックの実行間隔">
  データ編集用CB定義
</DataEditCBDefinition>
```

定義の詳細

DataEditCBDefinition タグ（全体情報の定義）

編集用 CB 定義の全体情報を定義します。

class="*クラス名*"

編集用 CB 定義の機能が実装されているクラス名を指定します。この値は固定値です。

クラス名は，*データ編集用 CB 定義*で指定するデータ編集用 CB 定義の種類ごとに異なります。また，入力アダプター定義で指定するか，出力アダプター定義で指定するかによって，指定するクラス名が異なる場合があります。指定するクラス名を次の表に示します。

データ編集用 CB 定義の種類	class 属性の値
フォーマット変換定義	入力アダプター定義で指定するか，出力アダプター定義で指定するかによって，値が異なります。 入力アダプター定義で指定する値 jp.co.Hitachi.soft.sdp.adaptor.callback.dataedit.formattranslate.InputFormatTranslatorCBImpl 出力アダプター定義で指定する値 jp.co.Hitachi.soft.sdp.adaptor.callback.dataedit.formattranslate.OutputFormatTranslatorCBImpl
マッピング定義	入力アダプター定義で指定するか，出力アダプター定義で指定するかによって，値が異なります。 入力アダプター定義で指定する値 jp.co.Hitachi.soft.sdp.adaptor.callback.dataedit.mapping.InputMappingCBImpl 出力アダプター定義で指定する値 jp.co.Hitachi.soft.sdp.adaptor.callback.dataedit.mapping.OutputMappingCBImpl

データ編集用 CB 定義の種類	class 属性の値
フィルター定義	入力アダプター定義で指定する場合，出力アダプター定義で指定する場合，どちらの場合でも次の値を指定します。 `jp.co.Hitachi.soft.sdp.adaptor.callback.dataedit.filter.FilterCBImpl`
レコード抽出定義	レコード抽出定義は，出力アダプター定義では指定できません。 入力アダプター定義では，次の値を指定します。 `jp.co.Hitachi.soft.sdp.adaptor.callback.dataedit.recordextract.RecordExtractionCBImpl`

name="*コールバック名*"

機能を識別するためのコールバック名を 1～100 文字の半角英数字，およびアンダーライン（_）で指定します。先頭の文字に指定できるのは，半角英字だけです。この属性は省略できません。コールバック名にはほかの CB 定義と重複した値を指定できますが，メッセージログで，入力アダプターまたは出力アダプターの処理の識別情報として出力するため，できるだけ重複しないように指定してください。

interval="*コールバックの実行間隔*"

コールバックの実行間隔（単位：ミリ秒）を0～60000 の整数で指定します。コールバックを実行したあと，指定した間隔で，入力アダプターまたは出力アダプターの処理が停止します。0 の場合は停止しません。省略した場合，0 が設定されます。

データ編集用CB定義

指定できる CB 定義を次に示します。

フォーマット変換定義

マッピング定義

フィルター定義

レコード抽出定義（出力アダプター定義では指定できません）

CB 定義については，「14.8　アダプター構成定義ファイルのデータ編集用 CB 定義」を参照してください。

14.6.4　送信用 CB 定義

説明

送信用 CB 定義（SendCBDefinition タグ）は，「14.5.3　入力アダプター定義」で説明した入力アダプター定義（InputAdaptorDefinition タグ）の子要素として定義します。

記述形式

```
<SendCBDefinition class="クラス名"
  name="コールバック名"
  interval="コールバックの実行間隔">
```

```
  入力ストリーム定義
</SendCBDefinition>
```

定義の詳細

SendCBDefinition タグ（全体情報の定義）

送信用 CB 定義の全体情報を定義します。

class="*クラス名*"

送信用 CB 定義の機能が実装されているクラス名を指定します。この値は固定値です。指定するクラス名を次に示します。

jp.co.Hitachi.soft.sdp.adaptor.callback.sendreceive.SendCBImpl

name="*コールバック名*"

機能を識別するためのコールバック名を 1～100 文字の半角英数字，およびアンダーライン（_）で指定します。先頭の文字に指定できるのは，半角英字だけです。この属性は省略できません。コールバック名にはほかの CB 定義と重複した値を指定できますが，メッセージログで入力アダプターの処理の識別情報として出力するため，できるだけ重複しないように指定してください。

interval="*コールバックの実行間隔*"

コールバックの実行間隔（単位：ミリ秒）を0～60000 の整数で指定します。コールバックを実行したあと，指定した間隔で入力アダプターの処理が停止します。0 の場合は停止しません。省略した場合，0 が設定されます。

入力ストリーム定義

入力ストリーム定義については，「14.9.1　入力ストリーム定義」を参照してください。

14.6.5　受信用 CB 定義

説明

受信用 CB 定義（ReceiveCBDefinition タグ）は，「14.5.4　出力アダプター定義」で説明した出力アダプター定義（OutputAdaptorDefinition タグ）の子要素として定義します。

記述形式

```
<ReceiveCBDefinition class="クラス名"
  name="コールバック名"
  interval="コールバックの実行間隔">
  出力ストリーム定義
</ReceiveCBDefinition>
```

定義の詳細

ReceiveCBDefinition タグ（全体情報の定義）

受信用 CB 定義の全体情報を定義します。

class="*クラス名*"

受信用 CB 定義の機能が実装されているクラス名を指定します。この値は固定値です。指定するクラス名を次に示します。

jp.co.Hitachi.soft.sdp.adaptor.callback.sendreceive.ReceiveCBImpl

name="*コールバック名*"

機能を識別するためのコールバック名を 1～100 文字の半角英数字，およびアンダーライン（_）で指定します。先頭の文字に指定できるのは，半角英字だけです。この属性は省略できません。コールバック名にはほかの CB 定義と重複した値を指定できますが，メッセージログで出力アダプターの処理の識別情報として出力するため，できるだけ重複しないように指定してください。

interval="*コールバックの実行間隔*"

コールバックの実行間隔（単位：ミリ秒）を0～60000 の整数で指定します。コールバックを実行したあと，指定した間隔で出力アダプターの処理が停止します。0 の場合は停止しません。省略した場合，0 が設定されます。

出力ストリーム定義

出力ストリーム定義については，「14.9　アダプター構成定義ファイルの送受信用 CB 定義」を参照してください。

14.7　アダプター構成定義ファイルの入出力用 CB 定義

ここでは，アダプター構成定義ファイルの入出力用 CB 定義について説明します。

入力コネクター定義は，「14.6.1　入力用 CB 定義」で説明した入力用 CB 定義（InputCBDefinition タグ）の子要素として定義します。また，出力コネクター定義は，「14.6.2　出力用 CB 定義」で説明した出力用 CB 定義（OutputCBDefinition タグ）の子要素として定義します。

入出力用 CB 定義の一覧を次の表に示します。

表 14-7　入出力用 CB 定義の一覧

項番	入出力用 CB 定義		親要素	参照先
1	入力コネクター定義	ファイル入力コネクター定義（FileInputConnectorDefinition タグ）	入力用 CB 定義	14.7.1　ファイル入力コネクター定義
2		HTTP パケット入力コネクター定義（HttpPacketInputConnectorDefinition タグ）		14.7.2　HTTP パケット入力コネクター定義
3		TCP データ入力コネクター定義（TcpDataInputConnectorDefinition タグ）		14.7.5　TCP データ入力コネクター定義
4	出力コネクター定義	ファイル出力コネクター定義（FileOutputConnectorDefinition タグ）	出力用 CB 定義	14.7.3　ファイル出力コネクター定義
5		ダッシュボード出力コネクター定義（DashboardOutputConnectorDef タグ）		14.7.4　ダッシュボード出力コネクター定義

14.7.1　ファイル入力コネクター定義

説明

ファイル入力コネクター定義（FileInputConnectorDefinition タグ）は，入力用 CB 定義（InputCBDefinition タグ）の子要素として定義します。

なお，ファイルの入力の処理については，「16.3　ファイルの入力」を参照してください。

記述形式

```
<FileInputConnectorDefinition>
  <input readType=" {BATCH | REAL_TIME} "
   compositionType=" {WRAP_AROUND | ANTI_WRAP_AROUND} "
   interval="監視時間間隔"
```

```
    retryCount="監視リトライ回数"
    readOrder=" {DEFINED | MODIFIED} "
    readUnit="読み込み単位"/>
  <file path="入力パス名"
    name="入力ファイル名"
    messageLog=" {ON | OFF} "/>
</FileInputConnectorDefinition>
```

定義の詳細

FileInputConnectorDefinition タグ（全体情報の定義）

ファイル入力コネクター定義の全体情報を定義します。この定義は必ず 1 個だけ記述します。

input タグ（読み込み定義）

入力ファイルの読み込み処理，および入力ファイルの監視処理の情報を定義します。この定義は必ず 1 個だけ記述します。

readType=" {BATCH | REAL_TIME} "

入力ファイルの読み込み方式を指定します。省略した場合，REAL_TIME が仮定されます。

指定できる値を次に示します。

BATCH：バッチ処理モードで，入力ファイルを読み込みます。

REAL_TIME：リアルタイム処理モードで，入力ファイルを読み込みます。

compositionType=" {WRAP_AROUND | ANTI_WRAP_AROUND} "

入力ファイルのファイル構成を指定します。省略した場合，ANTI_WRAP_AROUND が仮定されます。

指定できる値を次に示します。

WRAP_AROUND

ラップアラウンド構成の入力ファイルを読み込みます。

なお，readType 属性にBATCH を指定する場合は，WRAP_AROUND を指定できません。また，readType 属性にREAL_TIME，およびreadOrder 属性にDEFINED を指定する場合も，WRAP_AROUND を指定できません。

ANTI_WRAP_AROUND

非ラップアラウンド構成の入力ファイルを読み込みます。

なお，readOrder 属性にMODIFIED を指定する場合は，ANTI_WRAP_AROUND を指定できません。

interval="*監視時間間隔*"

readType 属性でREAL_TIME を指定した場合に，入力ファイル格納ディレクトリを監視して，入力対象のファイルが新規作成されているかどうかを確認する時間間隔（単位：ミリ秒）を0～1000000 の整数で指定します。0 を指定した場合，監視処理を連続で実行します。省略した場合，1000 が仮定されます。

retryCount="*監視リトライ回数*"

入力ファイル格納ディレクトリの監視処理のリトライ回数を0～1000 の整数で指定します。省略した場合，100 が仮定されます。

リトライ回数がこの属性で指定した値を超えると，入力ファイル格納ディレクトリに入力対象のファイルが存在しないことを示す警告メッセージKFSP46203-W が 1 回出力され，監視が継続されます。

監視処理のリトライは，その時点の入力対象ファイルごとに実行され，入力対象ファイルの読み込みが行われた時点でリトライ回数が0 にリセットされます。このため，リトライ回数のしきい値判定は入力ファイルごとに実行され，警告メッセージは入力ファイルごとに 1 回出力されます。

readOrder=”{DEFINED | MODIFIED} ”

入力ファイルを読み込む順番を指定します。省略した場合，DEFINED が仮定されます。

- DEFINED

  file タグのname 属性で指定した順番で，入力ファイルを読み込みます。

- MODIFIED

  入力ファイルの更新時刻の順番で，入力ファイルを読み込みます。

readUnit=”*読み込み単位*”

ファイル入力コネクターが一度に読み込むレコード数（単位：行数）を1～10000 の整数で指定します。省略した場合，1 が仮定されます。

file タグ（入力ファイルの定義）

入力ファイルの情報を定義します。この定義は必ず 1 個だけ記述します。

path=”*入力パス名*”

入力ファイル格納ディレクトリのパス名を 1～160 文字の半角文字で指定します。この属性は省略できません。

この属性に指定できる文字を次に示します。

半角英数字，「!」，「#」，「$」，「%」，「&」，「’ 」，「(」，「)」，「=」，「~」，「`」，「{」，「}」，「+」，「-」，「^」，「.」，「_」，「@」，「¥」，「/」，「:」，半角スペース

name=”*入力ファイル名*”

入力対象のファイル名をファイル名，または通番で指定します。この属性は省略できません。

ファイル名で指定する場合

ファイル名は，path 属性で指定した入力ファイル格納ディレクトリからの相対パスで指定します。複数指定する場合は，コンマ（,）で区切って指定してください。ただし，サブディレクトリは指定できません。入力ファイル名 1 つにつき，半角文字で 1～60 文字以内となるように指定してください。

なお，予約デバイス名（NUL など）をファイル名に指定しないでください。予約デバイス名を指定した場合，動作は保証されません。

通番で指定する場合

path 属性で指定した入力ファイル格納ディレクトリに格納されているファイルを通番で指定します。指定方法については，「入力ファイル名の通番での指定」を参照してください。

入力ファイル名は，1～1,024 文字の半角文字で指定します。「¥」，「/」，および「:」は指定できません。

この属性に指定できる文字を次に示します。

半角英数字，「!」，「#」，「$」，「%」，「&」，「’ 」，「(」，「)」，「=」，「~」，「`」，「{」，「}」，「+」，「-」，「^」，「.」，「_」，「@」，半角スペース

messageLog=”{ON | OFF}”

入力ファイルの読み込み開始時に，開始時刻と入力ファイル名をメッセージログに出力するかどうかを指定します。省略した場合，OFF が仮定されます。

指定できる値を次に示します。

ON：メッセージログに出力します。

OFF：メッセージログに出力しません。

記述例

```
<?xml version=”1.0” encoding=”UTF-8”?>
<root:AdaptorCompositionDefinition
 xmlns:ficon=”http://www.hitachi.co.jp/soft/xml/sdp/adaptor/definition/callback/FileInputCon
nectorDefinition”>
<!-- 途中略 -->
<!-- 入力用CB定義 -->
<cb:InputCBDefinition
 class=”jp.co.Hitachi.soft.sdp.adaptor.callback.io.FileInputCBImpl” name=”inputer1”>
  <!-- ファイル入力コネクター定義 -->
  <ficon:FileInputConnectorDefinition>
    <!-- 読み込み定義 -->
    <ficon:input readType=”REAL_TIME” interval=”600000”
     retryCount=”100” readUnit=”1”/>
    <!-- 入力ファイルの定義 -->
    <ficon:file path=” /tmp/” name=”a[1-100].txt”/>
  </ficon:FileInputConnectorDefinition>
</cb:InputCBDefinition>
```

入力ファイル名の通番での指定

file タグのname 属性で，入力ファイル名を通番で指定する場合の記述形式について説明します。

通番の記述形式

接頭文字列通番指定1中間文字列通番指定2接尾文字列

指定値について説明します。

接頭文字列,*中間文字列*,*接尾文字列*

半角文字列で指定します。

通番指定 1，*通番指定 2*

「[」と「]」で囲まれた「下限値-上限値」，または「下限値-」で，通番を指定します。下限値，および上限値は，8 桁以下の半角数字で指定します。

通番の記述規則

- 通番指定ができるファイルは 1 つだけです。
- 接頭文字列，中間文字列，および接尾文字列には，「[」または「]」を指定できません。

- 通番指定の下限値は省略できません。上限値を省略した場合は，上限値として最大値（99999999）が仮定されます。
- 通番は，「*下限値≦上限値*」，かつ「*上限値の桁数≧下限値の桁数*」となるように指定してください。
- 通番指定のファイルを入力する際の優先順位は，次のとおりです。

  1.*通番指定 1* の値が小さいもの

  2.*通番指定 2* の値が小さいもの

  例えば，「a1b2.txt，a2b1.txt」では「a1b2.txt」が先に入力されて，「a2b2.txt，a2b1.txt」では「a2b1.txt」が先に入力されます。
- 中間文字列には，数字でない文字を 1 つ以上含めてください。
- ファイル入力コネクターは，下限値から上限値までのファイル名を順番に入力します。入力ファイルの通番が不連続な場合も，通番の順序に従って入力します。例えば，通番が「1，2，4」となっていて 3 が抜けている場合，1 と 2 に続いて 4 を入力します。

通番の指定例

通番の指定例を次の表に示します。

指定例	処理結果	正常時の入力対象のファイルの並び順
a[1-100].txt	○	a1.txt，a2.txt，…（省略）…，a9.txt，a10.txt，a11.txt，a12.txt，…（省略）…，a99.txt，a100.txt
a[01-100].txt	○	a01.txt，a02.txt，…（省略）…，a09.txt，a10.txt，a11.txt，a12.txt，…（省略）…，a99.txt，a100.txt
a[001-100].txt	○	a001.txt，a002.txt，…（省略）…，a009.txt，a010.txt，a011.txt，a012.txt，…（省略）…，a099.txt，a100.txt
a[01-10]_[1-100].txt	○	a01_1.txt，a01_2.txt，…（省略）…，a01_9.txt，a01_10.txt，a01_11.txt，…（省略）…，a01_99.txt，a01_100.txt，a02_1.txt，a02_2.txt，…（省略）…，a02_9.txt，a02_10.txt，a02_11.txt，…（省略）…，a02_99.txt，a02_100.txt，…（省略）…，a10_1.txt，a10_2.txt，…（省略）…，a10_9.txt，a10_10.txt，a10_11.txt，…（省略）…，a10_99.txt，a10_100.txt
a[5-3].txt	×※1	－
a[001-9].txt	×※2	－
a[1-005].txt	○	a1.txt，a2.txt，a3.txt，a4.txt，a5.txt
a.txt,b[1-3].txt	×※3	－
a[1-3].txt，b[4-6].txt	×※3	－

（凡例）

○：正常に処理されます。

×：エラーとなります。

－：該当しません。

注※1

下限値が上限値よりも大きいため，エラーとなります。

注※2

下限値の桁数が上限値よりも大きいため，エラーとなります。

注※3

通番指定がある場合，複数ファイルは指定できないため，エラーとなります。

14.7.2　HTTP パケット入力コネクター定義

説明

HTTP パケット入力コネクター定義（HttpPacketInputConnectorDefinition タグ）は，入力用 CB 定義（InputCBDefinition タグ）の子要素として定義します。

なお，HTTP パケットの入力の処理については，「16.4.1　HTTP パケットの入力の概要」を参照してください。

記述形式

```
<HttpPacketInputConnectorDefinition>
  <input buffersize="入出力バッファーサイズ"
   assemblingtime="分割パケット組み立て時間">
    <packetdata
     globalheader="グローバルヘッダー領域長"
     packetheader="パケットヘッダー領域長"
     packetoffset="パケットデータサイズ領域オフセット"
     packetlength="パケットデータサイズ領域長"
     timeoffset="タイムスタンプ領域オフセット"/>
    <command path="コマンドパス名"
     parameter="コマンドパラメーター"/>
  </input>
  <output unit="最大出力単位">
    <record name="レコード名" type=" {REQUEST | RESPONSE} ">
      <field name="フィールド名"/>
    </record>
  </output>
</HttpPacketInputConnectorDefinition>
```

定義の詳細

HttpPacketInputConnectorDefinition タグ（全体情報の定義）

HTTP パケット入力コネクター定義の全体情報を定義します。この定義は必ず 1 個だけ記述します。

input タグ（読み込み定義）

HTTP パケットの入出力情報を定義します。この定義は必ず 1 個だけ記述します。

buffersize="*入出力バッファーサイズ*"

入力バッファーに格納できるHTTPパケット（単位：パケット），および出力バッファーに格納できる共通形式レコード（単位：レコード）の最大数を1～12288の整数で指定します。省略した場合，4096が仮定されます。

assemblingtime="*分割パケット組み立て時間*"

分割パケット組み立て時間（単位：ミリ秒）を1～5000の整数で指定します。省略した場合，2000が仮定されます。

分割パケットとは，最大セグメント長（MSS：Maximum Segment Size）に従ってTCP階層で分割されたパケットデータのことです。HTTPパケット入力コネクターが取得したデータが分割パケットの場合は，TCPプロトコルヘッダーの情報を基にパケットを組み立てます。分割パケット組み立て時間とは，新たな分割パケットが到着してから，到着済みのパケットと連結して組み立てるまでに掛かる時間のことで，分割パケットが連結されるごとにリセットしてカウントされます。なお，IP階層で分割されるパケットデータの組み立てはしません。

分割パケットの組み立てで，ここで指定した時間を超えた場合は，組み立て中の分割パケットは破棄されます。また，組み立てに使用した分割パケットの最新の時刻情報は，タイムスタンプとして使用されます。

packetdataタグ（HTTPパケット定義）

取得するHTTPパケットのフォーマットを定義します。この定義は必ず1個だけ記述します。

globalheader="*グローバルヘッダー領域長*"

グローバルヘッダー領域の長さ（単位：バイト）を1～128の整数で指定します。省略した場合，24が仮定されます。

packetheader="*パケットヘッダー領域長*"

パケットヘッダー領域の長さ（単位：バイト）を9～128の整数で指定します。省略した場合，16が仮定されます。

packetoffset="*パケットデータサイズ領域オフセット*"

パケットヘッダーの先頭からパケットデータサイズ領域までのオフセット（単位：バイト）を0～127の整数で指定します。省略した場合，8が仮定されます。

packetlength="*パケットデータサイズ領域長*"

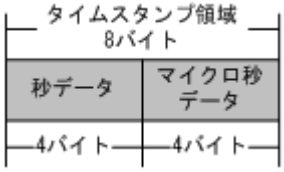
パケットデータサイズ領域の長さ（単位：バイト）を1～4の整数で指定します。省略した場合，4が仮定されます。

timeoffset="*タイムスタンプ領域オフセット*"

パケットヘッダーの先頭からタイムスタンプ領域までのオフセット（単位：バイト）を0～120の整数で指定します。省略した場合，0が仮定されます。

タイムスタンプ領域とは，パケットアナライザーがHTTPパケットをキャプチャーしたときのタイムスタンプを格納している，パケットヘッダーの領域です。タイムスタンプ領域のフォーマットを次の図に示します。

図 14-2　タイムスタンプ領域のフォーマット



command タグ（コマンド定義）

使用するパケットアナライザーの起動コマンドの情報を定義します。この定義は必ず 1 個だけ記述します。

Streaming Data Platform では，パケットアナライザーとして tcpdump を使用できます。

tcpdump を使用する場合に，この定義で指定する内容については，「tcpdump を使用する場合のcommand タグの指定内容」を参照してください。

path=”*コマンドパス名*”

パケットアナライザーの起動コマンドの絶対パス名を 1～100 文字の半角文字（ASCII コードの32～126）で指定します。

parameter=”*コマンドパラメーター*”

パケットアナライザーの起動コマンドに渡すパラメーターを 1～100 文字の半角文字（ASCII コードの32～126）で指定します。

output タグ（出力定義）

HTTP パケット入力コネクターが HTTP パケットを共通形式レコードに変換する際の，共通形式レコードの出力情報を定義します。この定義は必ず 1 個だけ記述します。

unit=”*最大出力単位*”

共通形式レコードの最大出力単位（単位：レコード）を1～1000 の整数で指定します。省略した場合，100 が仮定されます。

record タグ（レコード定義）

共通形式レコードのレコード情報を定義します。この定義は 1～2 個記述します。

name=”*レコード名*”

共通形式レコードのレコード名を 1～100 文字の半角英数字，およびアンダーライン（_）で指定します。先頭の文字に指定できるのは，半角英字だけです。この属性は省略できません。レコード名は，record タグ内で一意となるように指定してください。

type=”{REQUEST | RESPONSE}”

共通形式レコードの種類を指定します。この属性は省略できません。

指定できる値を次に示します。

"REQUEST"：リクエストレコードです。HTTP プロトコル通信で，クライアントからホストへのリクエスト送信時に発生したデータを格納する共通形式レコードです。

"RESPONSE"：レスポンスレコードです。HTTP プロトコル通信で，ホストからクライアントへのレスポンス送信時に発生したデータを格納する共通形式レコードです。

field タグ（フィールド定義）

共通形式レコードを構成するフィールドの情報を定義します。この定義は 1～16 個記述します。

name="*フィールド名*"

フィールド名を指定します。

フィールド名には，HTTP パケットから抽出したいデータの識別子を指定します。ここで指定した識別子に対応したデータが，各フィールドの値に格納されます。

フィールド名は，レコード定義内で一意となるように指定します。

フィールド名に指定できる識別子は，record タグの type 属性で指定する共通形式のレコードの種類によって異なります。フィールド名に指定できる識別子については，「フィールド名に指定できる識別子」を参照してください。

抽出されたデータは，Java のデータ型に変換されてフィールド値に格納されます。フィールド値に格納されるデータの Java のデータ型については，「14.7.2　HTTP パケット入力コネクター定義」の「(1)　フィールド値に格納されるデータの Java のデータ型」を参照してください。

記述例

```
<?xml version="1.0" encoding="UTF-8"?>
<root:AdaptorCompositionDefinition
 xmlns:hpicon="http://www.hitachi.co.jp/soft/xml/sdp/adaptor/definition/callback/HttpPacketI
nputConnectorDefinition">
<!-- 途中略 -->

<!-- 入力用CB定義 -->
<cb:InputCBDefinition
 class="jp.co.Hitachi.soft.sdp.adaptor.callback.io.packetinput.HttpPacketInputCBImpl" name="
inputer2">
  <!-- HTTPパケット入力コネクター定義 -->
  <hpicon:HttpPacketInputConnectorDefinition>
    <!-- 読み込み定義 -->
    <hpicon:input buffersize="096" assemblingtime="2000">
      <!-- HTTPパケット定義 -->
      <hpicon:packetdata globalheader="24" packetheader="16" packetoffset="8"
       packetlength="4" timeoffset="0"/>
      <!-- コマンド定義 -->
      <hpicon:command path="/usr/sbin/tcpdump"
       parameter=" -i eth0 -s 65535 -w - -n tcp port 80 or port 8080"/>
    </hpicon:input>
    <!-- 出力定義 -->
    <hpicon:output unit="100">
      <!-- レコード定義 -->
      <hpicon:record name="RECORD1" type="REQUEST">
        <!-- フィールド定義 -->
        <hpicon:field name="SEND_IP"/>
        <hpicon:field name="RECEIVE_IP"/>
        <hpicon:field name="SEND_PORT"/>
        <hpicon:field name="RECEIVE_PORT"/>
        <hpicon:field name="MESSAGE_TYPE"/>
        <hpicon:field name="TARGET_URI"/>
      </hpicon:record>
    </hpicon:output>
```

```
  </hpicon:HttpPacketInputConnectorDefinition>
</cb:InputCBDefinition>
```

tcpdump を使用する場合のcommand タグの指定内容

パケットアナライザーとして tcpdump を使用する場合に，command タグで指定する内容について説明します。なお，ここでは，tcpdump のバージョン 3.9.4 を使用する場合の例を示します。tcpdump の起動コマンドの詳細については，tcpdump のドキュメントを参照してください。

HTTP パケット入力コネクターでは，次の書式で記述された tcpdump の起動コマンドをサポートします。

```
/usr/sbin/tcpdump△-i△LANインタフェース名△-s△内部バッファーサイズ△-w△-△-n△tcp△port△ポート番号△and△host△IPアドレス
```

書式の「△」は半角スペースを表します。command タグのparameter 属性では，半角スペースを省略しないで記述してください。

書式に示した値について説明します。

tcpdump
: tcpdump の起動コマンドです。command タグのpath 属性で，tcpdump のパスを絶対パスで指定してください。

-i△*LANインタフェース名*
: 解析対象のコンピュータに接続されている LAN インタフェース名を指定するオプションです。このオプションと値は，command タグのparameter 属性で指定します。

-s△*内部バッファーサイズ*
: キャプチャーしたパケットデータを格納する内部バッファーのサイズ（単位：バイト）を指定します。HTTP では TCP のパケットサイズ以上を指定する必要があり，通常は，65,535 バイトを指定します。このオプションと値は，command タグのparameter 属性で指定します。

-w△-
: -w オプションは，キャプチャーしたパケットの出力先として，ファイル，または標準出力を指定するオプションです。HTTP パケット入力コネクターを使用する場合には，パケットを標準出力するため，「-w△-」と指定します。このオプションと値は，command タグのparameter 属性で指定します。

-n△tcp△port△*ポート番号*△and△host△*IPアドレス*
: パケットキャプチャーライブラリー（libpcap）のフィルター書式でキャプチャーフィルターを指定するオプションです。HTTP パケット入力コネクターを使用する場合には，HTTP プロトコルで使用するポート番号と，解析対象のコンピュータの IP アドレスを指定します。このオプションと値は，command タグのparameter 属性で指定します。

フィールド名に指定できる識別子

field タグのname 属性のフィールド名として指定できる識別子を次の表に示します。

表 14-8　field タグのname 属性のフィールド名として指定できる識別子

項番	識別子	データ	内容	プロトコル	レコードの種類ごとの指定可否	
					リクエスト	レスポンス
1	TIME	時刻※1	パケットデータが到着した時刻	－	○	○
2	PACKET_LENGTH	パケットサイズ※2	パケットデータの長さ（単位：バイト）	－	○	○
3	SEND_MAC	送信元 MAC アドレス	パケット送信元の MAC アドレス	Ethernet	○	○
4	SEND_IP	送信元 IP アドレス	パケット送信元の IP アドレス	IP	○	○
5	RECEIVE_IP	送信先 IP アドレス	パケット送信先の IP アドレス	IP	○	○
6	SEND_PORT	送信元ポート番号	パケット送信元のポート番号	TCP	○	○
7	RECEIVE_PORT	送信先ポート番号	パケット送信先のポート番号	TCP	○	○
8	MESSAGE_TYPE	メッセージ種別	Request，Response の種別	HTTP	○	○
9	METHOD_NAME	メソッド情報	GET，POST などのメソッド情報	HTTP	○	×
10	TARGET_URI	URI 情報※3	アクセス先の URI 情報	HTTP	○	×
11	REFERER	Referer※3	リンク元の URI 情報	HTTP	○	×
12	COOKIE	Cookie※3 ※4	クッキー情報※5	HTTP	○	○
13	STATUS_CODE	ステータスコード	要求の処理結果	HTTP	×	○
14	CONNECTION	Connection	接続の永続性情報	HTTP	○	○
15	CONTENT_LENGTH	Content-Length	コンテンツの長さ（単位：バイト）	HTTP	○	○
16	CONTENT_TYPE	Content-Type	コンテンツの種類	HTTP	○	○
17	MESSAGE_BODY	メッセージ・ボディ※3	実データ	HTTP	○※6	×

（凡例）

○：識別子を指定できます。

×：識別子を指定できません。

－：該当しません。

注※1

時刻は，パケットヘッダーのタイムスタンプから取得します。

注※2

パケットサイズは，HTTP メッセージの開始行，ヘッダーサイズ，および HTTP ヘッダー"Content-Length"が参照する値の合計値です。"Content-Length"がない場合，"Content-Length"が参照する値は0とします。

注※3

パーセントエンコードされている文字列は，UTF-8 でデコードします。

注※4

リクエストレコードとレスポンスレコードでCookie データの取得方法が異なります。

リクエストレコードの場合は，HTTP ヘッダー"Cookie"が参照するデータを取得します。

レスポンスレコードの場合は，HTTP ヘッダー"Set-Cookie"が参照するデータを取得します。HTTP ヘッダー"Cookie2"と"Set-Cookie2"が参照するデータは取得しません。

注※5

クッキー情報には複数のクッキーが含まれている場合があります。それぞれのクッキーをフィールドとして扱いたい場合は，マッピング定義のmap タグのfunction 属性でregexsubstring を指定して，正規表現の文字列を取得してください。

注※6

メッセージ・ボディは，メソッド情報がPOST，Content-Type のメディアタイプがtext で（サブタイプの値は問わない），かつContent-Length が存在する場合だけ取得できます。

(1) フィールド値に格納されるデータの Java のデータ型

field タグのname 属性で指定した識別子に従って抽出された各プロトコルデータは，次の表に示す Java のデータ型に変換されます。Java のデータ型への変換の際に，プロトコルデータの値が値の範囲の上限を超えている場合は，上限値までがフィールド値に格納されます。指定した識別子に対応するデータがプロトコルデータにない場合，フィールド値には，String 型のときは空文字，Integer 型のときは-1 が格納されます。

表 14-9　フィールド値に格納されるデータの Java のデータ型

項番	データ	プロトコル	Java のデータ型	値の範囲
1	時刻※	−	Timestamp	1970/01/01 00:00:00.000000～2261/12/31 23:59:59.999999
2	パケットサイズ	−	Integer	0～2,147,483,647
3	送信元 MAC アドレス	Ethernet	String	17 文字（00:00:00:00:00:00～FF:FF:FF:FF:FF:FF）
4	送信元ポート番号	TCP	Integer	0～65,535

項番	データ	プロトコル	Java のデータ型	値の範囲
5	送信先ポート番号	TCP	Integer	0～65,535
6	送信元 IP アドレス	IP	String	• IPv4 の場合 7～15 文字（0.0.0.0～255.255.255.255） • IPv6 の場合 40 文字（0000:0000:0000:0000:0000:0000:0000:0000～FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF）
7	送信先 IP アドレス	IP	String	
8	データ種別	HTTP	String	7～8 文字（Request またはResponse）
9	メソッド情報	HTTP	String	1～127 文字（GET，CONNECT など）
10	URI 情報	HTTP	String	1～255 文字
11	Referer	HTTP	String	
12	Cookie	HTTP	String	1～4,096 文字
13	ステータスコード	HTTP	String	3 文字（200，404 など）
14	Connection	HTTP	String	1～127 文字（close またはKeep-Alive）
15	Content-Length	HTTP	Integer	0～2,147,483,647
16	Content-Type	HTTP	String	3～255 文字
17	メッセージ・ボディ	HTTP	String	0～2,048 文字

（凡例）

－：該当しません。

注※

フィールドに時刻データを指定する場合は，クエリ定義ファイルの CQL データ型（TIMESTAMP）の有効桁数も合わせて 6 桁以上を指定してください。

14.7.3　ファイル出力コネクター定義

説明

ファイル出力コネクター定義（FileOutputConnectorDefinition タグ）は，出力用 CB 定義（OutputCBDefinition タグ）の子要素として定義します。なお，ファイルの出力の処理については，「16.8 ファイルへの出力」を参照してください。

記述形式

```
<FileOutputConnectorDefinition>
  <output compositionType=" {WRAP_AROUND | ANTI_WRAP_AROUND} "
   maxNumber="出力ファイル最大数"
   maxSize="出力ファイル最大サイズ""/>
  <file path="出力パス名"
   prefix="プレフィックス名"
   addDate=" {ON | OFF} "
   extension="拡張子"/>
</FileOutputConnectorDefinition>
```

定義の詳細

FileOutputConnectorDefinition タグ（全体情報の定義）

ファイル出力コネクター定義の全体情報を定義します。この定義は必ず 1 個だけ記述します。

output タグ（書き込み定義）

出力ファイルの書き込み処理の情報を定義します。この定義は必ず 1 個だけ記述します。

compositionType=" {WRAP_AROUND | ANTI_WRAP_AROUND} "

出力ファイル構成を指定します。ラップアラウンドと非ラップアラウンドがあります。どちらの場合も，出力先ファイル名と同名のファイルが存在する場合には上書きします。省略した場合，WRAP_AROUND が仮定されます。

指定できる値を次に示します。

WRAP_AROUND：ラップアラウンド構成でファイルを出力します。

maxNumber 属性で指定したファイル数に達するまでレコードを出力し，指定したファイル数を超えると，最初に作成した出力ファイルに上書きして出力します。ディスク容量の圧迫によるシステムの動作不正を防ぐため，通常の運用ではWRAP_AROUND を指定してください。

また，WRAP_AROUND を指定する場合は，file タグのaddDate 属性にOFF を指定してください。

ANTI_WRAP_AROUND：非ラップアラウンド構成でファイルを出力します。

maxNumber 属性で指定したファイル数に達するまでレコードを出力し，指定したファイル数を超えると，標準提供アダプターはレコードの出力を停止します。

maxNumber="*出力ファイル最大数*"

出力先のファイルの最大数を1～100000 の整数で指定します。省略した場合，256 が仮定されます。

maxSize="*出力ファイル最大サイズ*""

1 ファイルに書き込むレコードの行数を1～10000 の整数で指定します。この属性は省略できません。

file タグ（出力ファイルの定義）

出力ファイルの情報を定義します。この定義は必ず 1 個だけ記述します。

出力ファイルは，ここで定義した内容に従って生成されます。出力ファイル名の生成規則については，「14.7.3　ファイル出力コネクター定義」の「(1)　出力ファイル名の生成規則」を参照してください。

path="*出力パス名*"

ファイルを出力するディレクトリのパス名を 1～160 文字の半角文字で指定します。ここで指定したディレクトリが存在しない場合は，そのディレクトリを生成します。この属性は省略できません。

この属性に指定できる文字を次に示します。

半角英数字，「!」，「#」，「$」，「%」，「&」，「’」，「(」，「)」，「=」，「~」，「`」，「{」，「}」，「+」，「-」，「^」，「.」，「_」，「@」，「¥」，「/」，「:」，半角スペース

prefix="*プレフィックス名*"

ファイル名のプレフィックス（接頭辞）を 1～60 文字の半角文字で指定します。path 属性で指定した出力パス名からの相対パスで指定します。この属性は省略できません。

「¥」，「/」，および「:」は指定できません。

この属性に指定できる文字を次に示します。

半角英数字，「!」，「#」，「$」，「%」，「&」，「’」，「(」，「)」，「=」，「~」，「`」，「{」，「}」，「+」，「-」，「^」，「.」，「_」，「@」，半角スペース

なお，予約デバイス名（NUL など）をファイル名に指定しないでください。予約デバイス名を指定した場合，動作は保証されません。予約デバイス名を指定したファイルをオープンしようとした場合，メッセージKFSP46211-E が出力されてエラーとなります。

addDate="{ON | OFF}"

出力ファイル名に日時（*hhmmss_MMDD_YYYY* 形式）を付与するかどうかを指定します。省略した場合，OFF が仮定されます。

指定できる値を次に示します。

ON：日時を付与します。

OFF：日時を付与しません。

extension="*拡張子*"

出力ファイル名に付与する*拡張子*を指定します。省略した場合，*拡張子*なしのファイルが出力されます。

この属性に指定できる文字を次に示します。

半角英数字，「!」，「#」，「$」，「%」，「&」，「’」，「(」，「)」，「=」，「~」，「`」，「{」，「}」，「+」，「-」，「^」，「.」，「_」，「@」，半角スペース

記述例

```
<?xml version="1.0" encoding="UTF-8"?>
<root:AdaptorCompositionDefinition
 xmlns:focon="http://www.hitachi.co.jp/soft/xml/sdp/adaptor/definition/callback/FileOutputConnectorDefinition">
<!-- 途中略 -->

<!-- 出力用CB定義 -->
<cb:OutputCBDefinition class="jp.co.Hitachi.soft.sdp.adaptor.callback.io.FileOutputCBImpl" name="outputer1">
  <!-- ファイル出力コネクター定義 -->
  <focon:FileOutputConnectorDefinition>
```

```
    <!-- 書き込み定義 -->
    <focon:output compositionType="WRAP_AROUND"
    maxNumber="100" maxSize="10"/>
    <!-- 出力ファイルの定義 -->
    <focon:file path="/home/user1/data/" prefix="out"
    addDate="OFF" extension="csv"/>
  </focon:FileOutputConnectorDefinition>
</cb:OutputCBDefinition>
```

(1) 出力ファイル名の生成規則

出力ファイルは，file タグで指定した内容に従って生成されます。

出力されるファイル名の形式を次に示します。

```
'['プレフィックス名']' 【'['日時_']'】'['通番']'【.拡張子】
```

（凡例）

- [ ]：この記号で囲まれた項目は，この記号を省略しないでそのまま記述することを示します。
- 【】：この記号で囲まれた項目は省略できることを示します。

ファイル名の値について説明します。

プレフィックス名

prefix 属性で指定した値が使用されます。

日時

日時は，*hhmmss_MMDD_YYYY* 形式です。

- *hh*：時（00～23）
- *mm*：分（00～59）
- *ss*：秒（00～59）
- *MM*：月（01～12）
- *DD*：日（01～31）
- *YYYY*：年（西暦４桁表記）

出力ファイル名に日時を付与するかどうかは，addDate 属性で指定します。addDate 属性にON を指定した場合は日時が付与され，OFF を指定した場合は日時が付与されません。

通番

通番は，出力ファイルが生成された順番になります。通番は，1 からoutput タグのmaxNumber 属性で指定した値までです。標準提供アダプターを再起動した場合も同様に，通番は1 からになります。

出力ファイル名に日時を付与する場合，年月日が変更された場合も，継続して通番が付与されます。

拡張子

extension 属性に指定した値が使用されます。

(例)

- path 属性の指定値：/home/user1/data/
- prefix 属性の指定値：a
- addDate 属性の指定値：ON
- extension 属性の指定値：csv

この例の場合のファイルの生成番号と生成されるファイル名を次の表に示します。

生成番号	生成されるファイル名
1	/home/user1/data/a235015_0706_2009_1.csv
2	/home/user1/data/a235959_0706_2009_2.csv
3	/home/user1/data/a000130_0707_2009_3.csv

なお，output タグのcompositionType 属性でWRAP_AROUND を指定し，かつaddDate 属性でON を指定している場合，日時が付与されるため，通番が同じでも別ファイルとなります。

14.7.4　ダッシュボード出力コネクター定義

説明

ダッシュボード出力コネクター定義（DashboardOutputConnectorDefinition タグ）は，出力用 CB 定義（OutputCBDefinition タグ）の子要素として定義します。

なお，ダッシュボードへの出力の処理については，「16.9　ダッシュボードへの出力」を参照してください。

記述形式

```
<DashboardOutputConnectorDefinition
 MaxNum="最大レコード保持数"
 Record="レコード名"
 ReadRecordRemoveFlag=" {ON | OFF} ">
  <RecordHoldTime
   DateReference=" {CURRENT_DATE | LAST_UPDATE} "
   RecordTime="レコード保持期間"
   DateFieldPosition="時刻フィールド番号"/>
  <DataProcessingDefinition
   Name=" {HistoryRecorder | NoDataProcessing} ">
    <HistoryRecorder/>
    <NoDataProcessing/>
```

```
  </DataProcessingDefinition>
</DashboardOutputConnectorDefinition>
```

定義の詳細

DashboardOutputConnectorDefinition タグ（全体情報の定義）

ダッシュボード出力コネクター定義の全体情報を定義します。この定義は必ず 1 個だけ記述します。

MaxNum=”*最大レコード保持数*”

レコード保持領域内でのレコードの最大保持数を1～10000 の整数で指定します。省略した場合，10000 が仮定されます。

レコード数がこの属性で指定した値を超えると，古いレコードから順番に削除します。

Record=”*レコード名*”

ダッシュボード出力の対象となるレコードを指定します。この属性は省略できません。

ReadRecordRemoveFlag=”{ON | OFF} ”

ダッシュボードが取得したレコードをダッシュボード出力コネクターから削除するかどうかを指定します。省略した場合，OFF が仮定されます。

指定できる値を次に示します。

ON：レコードを削除します。

OFF：レコードを削除しません。

RecordHoldTime タグ（レコード保持期間の定義）

保持期間によってダッシュボード出力コネクターのレコードを削除する場合に，レコードを保持する期間を定義します。この定義を省略した場合，保持期間によるレコードの削除は行われません。

なお，レコードが削除されるのは，次の条件を満たす場合です。

```
（DateReference属性の基準時刻）-（RecordTime属性の保持期間）
＞（DateFieldPosition属性のフィールドの時刻）
```

例えば，基準時刻が10:00:10，保持期間が5 秒，フィールドの時刻が10:00:04 だった場合，そのレコードは削除されます。

DateReference=”{CURRENT_DATE | LAST_UPDATE} ”

レコードを削除するための基準時刻を指定します。この属性は省略できません。

指定できる値を次に示します。

CURRENT_DATE：システムの現在の時刻を基準時刻とします。

LAST_UPDATE：最後にタプルを受信した時刻を基準時刻とします。

RecordTime=”*レコード保持期間*”

レコードの保持期間（単位：秒）を0～86400 の整数で指定します。この属性は省略できません。

DateFieldPosition=”*時刻フィールド番号*”

レコード内で時刻が設定されているフィールドの番号を1～3000 の整数で指定します。この属性は省略できません。

DataProcessingDefinition タグ（データの加工処理の定義）

ダッシュボード出力コネクターが出力するダッシュボード表示用データの加工処理について定義します。この定義を省略した場合，データの加工処理は行われません。

Name=”{HistoryRecorder | NoDataProcessing} ”

ダッシュボード表示用データの加工処理をするかどうかを指定します。この属性は省略できません。

指定できる値を次に示します。

HistoryRecorder：加工処理をします。グラフを表示したい場合には，HistoryRecorder を指定します。加工処理をする場合，最大レコード保持数，およびレコード保持期間の定義に従うすべてのレコードをダッシュボード表示用データ内に保持します。HistoryRecorder を指定した場合，DataProcessingDefinition タグの下に空のHistoryRecorder タグを記述します。

NoDataProcessing：加工処理をしません。加工処理が不要な場合には，NoDataProcessing を指定します。加工処理をしない場合，ダッシュボード出力コネクターからレコードを取得するたびに，それまでのダッシュボード表示用データがすべて削除されます。NoDataProcessing を指定した場合，DataProcessingDefinition タグの下に空のNoDataProcessing タグを記述します。

HistoryRecorder タグ（履歴保持定義）

DataProcessingDefinition タグのName 属性でHistoryRecorder を指定した場合に，DataProcessingDefinition タグの下に空のHistoryRecorder タグを記述します。

Name 属性でNoDataProcessing を指定した場合には，このタグを記述する必要はありません。

NoDataProcessing タグ（無加工出力定義）

DataProcessingDefinition タグのName 属性でNoDataProcessing を指定した場合に，DataProcessingDefinition タグの下に空のNoDataProcessing タグを記述します。

Name 属性でHistoryRecorder を指定した場合には，このタグを記述する必要はありません。

記述例

```
<?xml version="1.0" encoding="UTF-8"?>
<root:AdaptorCompositionDefinition
xmlns:docon="http://www.hitachi.co.jp/soft/xml/sdp/adaptor/definition/callback/DashboardOutputConnectorDefinition">
<!-- 途中略 -->

<!-- 出力用CB定義 -->
<cb:OutputCBDefinition class="jp.co.Hitachi.soft.sdp.adaptor.callback.io.dashboard.DashboardOutputCBImpl" name="outputer2">
  <!-- ダッシュボード出力コネクター定義 -->
  <docon:DashboardOutputConnectorDefinition MaxNum="1000" Record="RECORD2"
   ReadRecordRemoveFlag="OFF">
    <!-- レコード保持期間の定義 -->
    <docon:RecordHoldTime DateReference="LAST_UPDATE"
     RecordTime="300" DateFieldPosition="1"/>
    <!-- データの加工処理の定義 -->
    <docon:DataProcessingDefinition Name="NoDataProcessing">
      <docon:NoDataProcessing/>
    </docon:DataProcessingDefinition>
```

```
  </docon:DashboardOutputConnectorDefinition>
</cb:OutputCBDefinition>
```

14.7.5　TCP データ入力コネクター定義

説明

TCP データ入力コネクター定義（TCPDataInputConnectorDefinition タグ）は，出力用 CB 定義（OutputCBDefinition タグ）の子要素として定義します。

なお，TCP データ入力の処理については，「16.2　TCP データの入力」を参照してください。

```
<tcpcon:TCPDataInputConnectorDefinition sleeptime="スリープ時間">
<tcpcon:input bufferSize="バッファーのサイズ" receiveBufferSize="バッファーのサイズ" charCod
e="文字コード"
maxThreads="最大スレッド数">
<tcpcon:binary offset="シークするヘッダーデータのサイズ">
<tcpcon:data name="データ名" type="データ型" size="データのサイズ" offset="シークするデータ
のサイズ">
:
</tcpcon:binary>
</tcpcon:input>
<tcpcon:output bufferSize="バッファーのサイズ" unit="送信されるレコードの単位">
<tcpcon:record name="レコード名">
<tcpcon:fields>
<tcpcon:field name="フィールド名"/>
:
</tcpcon:fields>
</tcpcon:record>
</tcpcon:output>
</tcpcon:TCPDataInputConnectorDefinition>
```

定義の詳細

表 14-10　要素と属性の一覧

要素または属性※	説明	値	定義数	C アダプターに指定可能
TCPDataInputConnectorDefinition	TCP データ入力コネクターを定義します。	-	1	はい
input	コネクターの入力情報を定義します。	-	1	はい
binary	TCP データとヘッダーデータで繰り返されるデータの単位の形式を定義します。	-	1	はい

要素または属性※	説明	値	定義数	Cアダプターに指定可能
data	繰り返されるデータの単位を定義します。このデータはレコードのフィールドとして形成されます。	-	1～3000	はい
@name	データの名前を定義します。	1～100 文字の文字列（英数字とアンダーライン）です。最初の文字は英字である必要があります。input 要素内で一意の値を設定する必要があります。	1	はい
@type	データのデータ型を定義します。	次の型を指定できます。 BYTE | SHORT | INT | LONG | FLOAT | DOUBLE | BIGDECIMAL| CHAR | VARCHAR | DATE | TIME| TIMESTAMP	1	はい
@size	データのサイズ（単位：バイト）を定義します。	1～214748364 の整数［バイト］ 指定したサイズが指定した型のサイズより大きい場合，指定した型のサイズ（単位：バイト）だけが先頭から解析され，残りは無視されます。データ型とサイズは次のとおりです。 • INT：4 バイト • DATE：8 バイト • TIME：8 バイト • TIMESTAMP：8 バイトまたは12 バイト • BYTE：1 バイト • SHORT：2 バイト • LONG：8 バイト • FLOAT：4 バイト • DOUBLE：8 バイト • BIGDECIMAL：該当なし • CHAR：該当なし • VARCHAR：該当なし 可変長の最大サイズを指定します。 TIMESTAMP 型を指定した場合： • 指定したサイズが12 バイト未満の場合，最初の8 バイトがミリ秒として解析されます。 • 指定したサイズが12 バイト以上の場合，最初の8 バイトはミリ秒として解析され，次の4 バイトはナノ秒として解析されます。 指定したサイズが各データ型のサイズより小さい場合，アダプターはBYTE，FLOAT，およびDOUBLE 型を除いてゼロ拡張をします。	1	はい

要素または属性※	説明	値	定義数	Cアダプターに指定可能
@size	データのサイズ（単位：バイト）を定義します。	DATE型，TIME型，BIGDECIMAL型のTCPデータ上でのデータ構造： • DATE型：java.sql.DateクラスのgetTime()メソッドで取得できるlong値（1970年1月1日00:00:00 GMTからのミリ秒） • TIME型：java.sql.TimeクラスのgetTime()メソッドで取得できるlong値（1970年1月1日00:00:00 GMTからのミリ秒） • BIGDECIMAL型：データ長（2バイト）＋BigDecimal型の文字列表現データ（*n*バイト）で表現します。先頭2バイトはBigDecimal型の文字列表現のデータ長です。	1	はい
@offset	シークするデータのサイズ（単位：バイト）	0～214748364の整数 デフォルト値：0［バイト］	0～1	はい
@offset	シークするヘッダーデータのサイズ（単位：バイト）を定義します。	0～214748364の整数 デフォルト値：0［バイト］	0～1	はい
@bufferSize	TCPデータを一時的にためておくバッファーのサイズを指定します。	1～214748364の整数 デフォルト値：10000［バイト］ data要素の合計のサイズより小さい値を指定した場合，data要素の合計サイズがこの属性値として設定されます。	0～1	はい
@port	SDPブローカーと連携してアダプターを自動起動する場合は，この属性を指定しないでください。 hsdpstartinproコマンドでアダプターを手動起動する場合は，TCP接続を待機するために使用されるポート番号を指定します。	1～65535の整数	0～1	はい
@charCode	TCPデータを文字列にコピーする文字コードを指定します。	ASCII | UTF-8 | UTF-16 | UTF-16BE | UTF-16LE システムコンフィグプロパティファイルのquerygroup.encodingパラメーターにUS-ASCIIを指定した場合は，ASCIIを指定してください。	1	はい
@receiveBufferSize	OSのソケット受信バッファーのサイズを指定します。	1～214748364の整数	0～1	はい
@maxThreads	データ受信のための最大スレッド数を定義します。	1～128の整数 デフォルト値：16［スレッド］	0～1	はい

要素または属性※	説明	値	定義数	Cアダプターに指定可能
@maxThreads	この定義は，C用TCPデータ入力コネクターにだけ有効です。 接続が受け付けられると，システムは接続に対応するスレッドを（指定された数まで）作成します。 接続数が最大スレッド数を超える場合は，最小接続スレッドに割り当てられている接続が処理されます。	1～128の整数 デフォルト値：16 [スレッド]	0～1	はい
@idleTimeout	無通信監視タイマーを秒単位で指定します。無期限のタイムアウトを設定するには，0を指定します。 データが送信されない間，キープアライブパケットが送信元ホストから一定間隔で送信されます。 キープアライブパケットまたはデータが指定されたタイムアウト値内に送信されない場合は，エラーメッセージが出力され，接続が閉じられます。	0～172800または10～172800の整数 デフォルト値：60［秒］	0～1	はい
@queueFullRetryInterval	TCPアダプター内のキューが満杯の場合に，データを再送信する動作および間隔（ミリ秒）を指定します。	-1～2147483647の整数 デフォルト値：100［秒］ 0～2147483647を指定した場合： 指定した時間（ミリ秒）スリープしてから，データ送信をリトライし続けます。 -1を指定した場合： データ送信のリトライはしないで，該当データを破棄します。	0～1	はい
output	コネクターの出力情報を定義します。	-	1	はい
@unit	次のコールバックに一度に送信されるレコードの単位を定義します。	1～214748364の整数 デフォルト値：10000［レコード］	0～1	はい
@bufferSize	レコードを一時的にためておくバッファーのサイズを指定します。	1～214748364の整数 デフォルト値：20000［レコード］	0～1	いいえ
record	コネクターが次のコールバックに送信するレコードの情報を定義します。	-	1	はい
@name	コネクターが次のコールバックに送信するレコードの名前を定義します。	1～100文字の文字列（英数字とアンダーライン）です。最初の文字は英字である必要があります。	1	はい
fields	レコードに含まれるフィールドの情報を定義します。	-	1	はい

要素または属性※	説明	値	定義数	Cアダプターに指定可能
field	各フィールドの情報を定義します。要素の数は，data要素の数と同じである必要があります。	-	1～3000	はい
@name	フィールドの名前を定義します。 name属性の名前がこの値と同じdata要素のデータが，フィールドとして形成されます。	1～100文字の文字列（英数字とアンダーライン）です。最初の文字は英字である必要があります。fields要素内で一意の値を設定する必要があります。	1	はい

注※

最初の文字がアットマーク（@）の場合は，属性です。

（凡例）

-：該当なし

記述例

```
<tcpcon:TCPDataInputConnectorDefinition>
<tcpcon:input port="30000" charCode="UTF-8">
<tcpcon:binary offset="0">
<tcpcon:data name="MONITOR_ID" type="STRING" size="32" offset="0">
<tcpcon:data name="IP_ADDRESS" type="STRING" size="15" offset="0">
<tcpcon:data name="PORT_NUMBER" type="STRING" size="5" offset="0">
<tcpcon:data name="TIMESTAMP" type="TIMESTAMP" size="8" offset="0">
<tcpcon:data name="PACKETSIZE" type="INT" size="2" offset="0">
</tcpcon:binary>
</tcpcon:input>
<tcpcon:output unit="10000">
<tcpcon:record name="PACKET_DATA">
<tcpcon:fields>
<tcpcon:field name="MONITOR_ID"/>
<tcpcon:field name="IP_ADDRESS"/>
<tcpcon:field name="PORT_NUMBER"/>
<tcpcon:field name="TIMESTAMP"/>
<tcpcon:field name="PACKETSIZE"/>
</tcpcon:fields>
</tcpcon:record>
</tcpcon:output>
</tcpcon:TCPDataInputConnectorDefinition>
```

14.7.6　カスケーディングクライアントコネクター定義

説明

カスケーディングクライアントコネクター定義（CascadingClientConnectorDefinition タグ）は，出力用 CB 定義（OutputCBDefinition タグ）の子要素として定義します。

なお，カスケーディングアダプターの処理については，「16.7　クエリグループまたは外部出力アダプターへの出力」を参照してください。

定義の詳細

表 14-11　要素と属性の一覧

要素または属性※1	説明	値	定義数	C アダプターに指定可能
CascadingClientConnectorDefinition	カスケーディングクライアントコネクターを定義します。	-	1	はい
@maxThreads	この定義は，C 用カスケーディングクライアントコネクターにだけ有効です。 データ送信のための最大スレッド数を定義します。 cascading 要素に対応するスレッドを指定した数まで作成するためです。 cascading 要素の数が最大スレッド数を超える場合は，cascading 要素を処理している最小番号のスレッドから順にcascading 要素が割り当てられます。	1～214748364716 の整数値を指定します。 デフォルト値：16 [スレッド]	0～1	はい
cascading	出力ストリームの情報を定義します。	-	1～16※2	はい
@name	cascading 要素の名前を定義します。	値を1～100 の英数字（アンダーラインを含む）で指定します。最初の文字は英字である必要があります。 CascadingClientConnectorDefinition 要素内で一意の名前を指定します。	1	はい
@source	カスケーディングする入力元のストリームのレコード名を指定します。	値を1～100 の英数字（アンダーラインを含む）で指定します。最初の文字は英字である必要があります。	1	はい

要素または属性※1	説明	値	定義数	Cアダプターに指定可能
@source	名前は，前段のコールバックで定義され，CascadingClientConnectorDefinition 要素内で一意である必要があります。	値を1～100 の英数字（アンダーラインを含む）で指定します。最初の文字は英字である必要があります。	1	はい
@encoding	@source で指定したレコードに，STRING 型データが含まれる場合は，そのレコードの文字コードを指定します。	US-ASCII |UTF-8 | UTF-16 | UTF-16BE | UTF-16LE • デフォルト値：US-ASCII	0～1	いいえ
@queueFullRetryInterval	カスケーディングアダプター内のキューが満杯の場合に，データを再送信する動作および間隔（ミリ秒）を指定します。	-1～2147483647 の整数値を指定します。 デフォルト値：100［秒］ 0～2147483647 を指定した場合： 指定した時間（ミリ秒）スリープしてから，データ送信をリトライし続けます。 -1 を指定した場合： キューの最も古いデータを削除し，該当のデータを送信します。データ送信のリトライはしません。	0～1	いいえ
targetStream	接続先ストリームの情報を定義します。 自動生成アダプターテンプレートファイルのcascading_out.xml を編集する場合，このタグの指定を省略してください。	-	0～1	はい
@name	対象ストリーム名を指定します。	値を1～100 の英数字（アンダーラインを含む）で指定します。最初の文字は英字である必要があります。	1	はい
@querygroup	対象クエリグループ名を指定します。	値を1～64 の英数字（アンダーラインを含む）で指定します。最初の文字は英字である必要があります。	1	はい
@serverCluster	データ送信のためのサーバクラスタ名を指定します。	値を1～64 の英数字（アンダーラインを含む）で指定します。最初の文字は英字である必要があります。 デフォルト値：My_server_cluster	0～1	いいえ
@brokerAddr	データ送信のためのブローカーのアドレスを指定します。 省略した場合，localhost が問い合わせを受信します。	次の形式で値を指定します。 *ホスト名またはIPアドレス:ポート番号*	0～1	いいえ

要素または属性※1	説明	値	定義数	Cアダプターに指定可能
dispatch	SDP ブローカー連携によってアダプターを自動起動する場合は，このタグとcascadingFile タグを指定しないでください。 対象ストリームに 2 か所以上の宛先が含まれる場合に，データの負荷分散方式を指定します。 指定を省略した場合，アダプターはラウンドロビンによってデータを送信します。	-	0～1	はい
@type	このストリームの負荷分散方式を指定します。	次のどれかの値を指定できます。 • HASHING：カスケーディングアダプターは，hashingKey で定義されるタプル内の列のハッシュ値を使用して，タプルの送信先を決定します。 • ROUNDROBIN：カスケーディングアダプターは，ラウンドロビンによってタプルを送信します。 • STATIC：カスケーディングアダプターは，condition タグで定義される静的ルールを使用してタプルの送信先を決定します。 デフォルト値：ROUNDROBIN	1	はい
hashingKey	ハッシュ化対象の列を定義します。 負荷分散方式がHASHING の場合は，この要素を指定する必要があります。	-	0～255	はい
@columnName	ハッシュ値を算出するのに使用する，レコードの列名を指定します。 この属性値は，hashingKey 要素間で重複できません。	1～100 の英数字（アンダーラインを含む）で指定します。最初の文字は英字である必要があります。	1	はい
cascadingProperties	カスケーディングアダプターの通信情報を定義します。 このプロパティは，cascadingFile タグが定義されていない場合にだけ指定できます。	-	0～1	いいえ
retryConnect	targetStream タグで指定されたストリームとの接続が切断された場合に，再接続方法を指定します。	-	0～1	いいえ

要素または属性※1	説明	値	定義数	Cアダプターに指定可能
retryConnect	この属性は，targetStream が指定されている場合にだけ指定できます。 retryConnect タグを省略した場合，リトライが毎秒際限なく試行されます。リトライの試行に連続して失敗すると，エラーメッセージが出力され，その後 60 回メッセージの出力が抑制されます。	-	0～1	いいえ
@count	データ送信時のリトライ回数を指定します。 -1 を指定すると，無制限に接続がリトライされます。 0 を指定すると，接続はリトライされません。	-1～2147483647 の整数値を指定します。	1	いいえ
@interval	リトライと次のリトライ間のタイムアウト（秒単位）を指定します。	1～2147483647 の整数値を指定します。	1	いいえ
@messageSuppressCount	リトライが連続して失敗する場合に，何回メッセージを抑制するかを指定します。 メッセージの出力後，接続の試行が連続して失敗すると，メッセージは指定した値の回数抑制されます。 -1 を指定した場合，リトライの試行時にメッセージは出力されません。	-1～2147483647 の整数値を指定します。 デフォルト値は60 です。	0～1	いいえ
retryWait	接続先の外部アダプターが切断されたあとの，再接続するまでの待ち時間を指定します。 この属性は，targetStream が除かれている場合にだけ指定できます。 指定を省略した場合，再接続の待ち時間は 30 秒になります。	-	0～1	いいえ
@timeout	再接続の最大待ち時間（単位：秒）を指定します。指定した時間内に接続が確立されない場合は，それ以上接続を待ちません。0 を指定すると，再接続を待ちません。-1 を指定すると，再接続をいつまでも待ち続けます。	-1～2147483647 の整数値を指定します。デフォルト値は30 です。	1	いいえ
buffer	内部バッファー情報を指定します。	-	0～1	いいえ

要素または属性※1	説明	値	定義数	Cアダプターに指定可能
@sendBufferSize	内部入力ストリームキューのサイズを指定します。	1～2147483647 の整数値を指定します。 デフォルト値は200000 です。	0～1	いいえ
tcpParameters	通信プロトコルがTCP の場合，TCP パラメーターを指定します。	-	0～1	いいえ
connectTimeout	バックエンドサーバと接続する際のタイムアウト値を定義します。 targetStream タグを定義していない場合，このタグは無視されます。	-	0～1	いいえ
@timeout	タイムアウト値を秒単位で指定します。無期限のタイムアウトを設定するには，0 を指定します。	0～3600 の整数値を指定します。 デフォルト値は20 です。	1	いいえ
condition	タプルを送信するための静的ルールを定義します。 負荷分散方式がSTATIC の場合は，この要素を指定する必要があります。	-	0～32	はい
@name	condition 要素の名前を定義します。定義した名前は，条件と，cascading 用プロパティファイルに記述されている宛先を結び付けるのに使用します。	値を1～100 の英数字（アンダーラインを含む）で指定します。最初の文字は英字である必要があります。 cascading の要素内で一意の名前を指定します。	1	はい
pattern	各列の値の許容されるパターンを定義します。	-	1～32	はい
@columnName	パターンを適用する，レコード中の列の名前を指定します。 大文字と小文字は区別されます。 受信用コネクター定義内でのレコード中の列の名前指定で，${hsdp_adp_outputSchema}パラメーターを使用している場合は，列の名前はすべて大文字となっているため，注意してください。 指定する列のデータ型は次のどれかです。STRING / BYTE / SHORT / INT / LONG。 名前は，condition 要素内で重複できません。	1～100 の英数字（アンダーラインを含む）で指定します。最初の文字は英字である必要があります。	1	はい

要素または属性※1	説明	値	定義数	C アダプターに指定可能
@value	列の値の許容可能なパターンを指定します。アダプターは，このパターンに一致するタプルだけ送信します。※3	列のデータ型に応じて，下記の「パターンの形式」に説明されるように文字列を指定します。	1	はい
cascadingFile	SDP ブローカー連携によってアダプターを自動起動する場合は，この属性を指定しないでください。 cascading 用プロパティファイルの情報を定義します。	-	0～1	はい
@path	cascading 用プロパティファイルの絶対パスまたは運用ディレクトリからの相対パスを定義します。	1～160 の半角文字の文字列を指定します。	1	はい

注※1

最初の文字がアットマーク（@）の場合は，属性です。

注※2

cascading 要素の下にあるtargetStream タグを指定しない場合は，cascading 要素を 1 回だけ指定できます。

注※3

condition タグに複数のパターンが指定されている場合，アダプターは，すべてのパターンが一致した場合にだけタプルを送信します。タプルが 2 つ以上の条件に一致する場合，条件に一致する 1 つの送信先がタプルの送信先となります。

（凡例）

-：該当なし

パターンの形式

列のデータ型	value 属性の形式	例	備考
CHAR VARCHAR	Java アダプターについては，Java の正規表現の文字列を指定します。 CQL データ型がCHARACTER で，指定した値の長さが，設定されているCHARACTER の長さより短い場合，残りの文字列に対してスペースを入力してください。 パターンマッチングのエンコーディングは，UTF-8 が適用されます。	probe_[a-zA-Z0-9]+	-

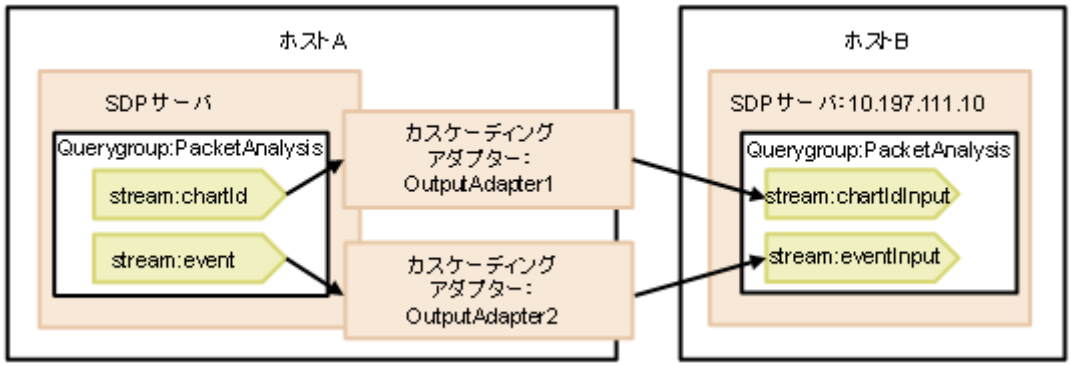
列のデータ型	value 属性の形式	例	備考
BYTE SHORT INT LONG	数字の範囲を指定します。 [*範囲*][*範囲*] ..[*範囲*] *範囲*は，範囲内の下限値と上限値をコンマで区切った 2 個の数字，または下限値と上限値が等しい場合，1 個の数字で表します。 *範囲*::=*数字*|*数字*,*数字* *数字*に指定できる値は，次のとおりです。 • BYTE：-128～127 • SHORT：-32768～32767 • INT：-2147483648～2147483647 • LONG：-9223372036854775808～9223372036854775807	[-20,-10] [0] [10,20] (-15，0，10，11 は上記の例で指定する範囲に一致します。-1，1，21 は一致しません)	1～32 組の範囲を指定します。
(凡例) -：該当なし			

(1) アダプター構成定義ファイルの記述例

以下は，カスケーディング接続の例と，HASHING，ROUNDROBIN，およびSTATIC を使用してアダプターがバックエンドサーバに送信する情報の例を示しています。

カスケーディング接続の例

図 14-3 カスケーディング接続の例



アダプター構成定義ファイルの記述例

- カスケーディング接続の例に対するアダプター構成定義ファイルの例を示します。

```
<?xml version="1.0" encoding="UTF-8"?>
 <!-- All Rights Reserved.Copyright (C) 2014, Hitachi, Ltd. -->
 <root:AdaptorCompositionDefinition
  xmlns:root="http://www.hitachi.co.jp/soft/xml/sdp/adaptor/definition"
  xmlns:cmn="http://www.hitachi.co.jp/soft/xml/sdp/adaptor/definition/common"
  xmlns:adp="http://www.hitachi.co.jp/soft/xml/sdp/adaptor/definition/adaptor"
  xmlns:cb="http://www.hitachi.co.jp/soft/xml/sdp/adaptor/definition/callback"
  xmlns:rcon="http://www.hitachi.co.jp/soft/xml/sdp/adaptor/definition/callback/ReceiveCo
```

```
nnectorDefinition"
  xmlns:caclcon="http://www.hitachi.co.jp/soft/xml/sdp/adaptor/definition/callback/Cascad
ingClientConnectorDefinition"
  >
  <!-- 共通定義 -->
  <cmn:CommonDefinition>
  <!-- アダプタートレース定義 -->
  <cmn:AdaptorTraceDefinition trace="OFF"/>
  </cmn:CommonDefinition>
  <!-- インプロセスグループ定義 -->
  <adp:InprocessGroupDefinition name="AnalysisOutputAdaptor">
  <!-- 出力アダプター定義 -->
  <!-- カスケーディング出力アダプター定義1-->
  <adp:OutputAdaptorDefinition name="OutputAdaptor1" interval="0" charCode="UTF-8" lineFe
ed="LF">
  <!-- 受信コネクターCB定義 -->
  <cb:ReceiveCBDefinition class="jp.co.Hitachi.soft.sdp.adaptor.callback.sendreceive.Rece
iveConnectorCBImpl" name="Receiver1">
  <rcon:ReceiveConnectorDefinition>
  <rcon:streamOutputs>
  <rcon:streamOutput>
  <rcon:stream name="chartId" querygroup="PacketAnalysis" />
  <rcon:record name="OUTPUT_DATA">
  <rcon:column name="ID" type="STRING" />
  <rcon:column name="TIMESTAMP" type="TIMESTAMP" />
  <rcon:column name="METRIC_ID" type="STRING" />
  <rcon:column name="BAND_WIDTH" type="DOUBLE" />
  <rcon:column name="MB_THRD" type="DOUBLE" />
  <rcon:column name="THRD_UP" type="DOUBLE" />
  <rcon:column name="THRD_DOWN" type="DOUBLE" />
  <rcon:column name="GRADIENT" type="DOUBLE" />
  <rcon:column name="INTERCEPT" type="DOUBLE" />
  <rcon:column name="BASELINE" type="DOUBLE" />
  <rcon:column name="BASELINE_HIGH" type="DOUBLE" />
  <rcon:column name="BASELINE_LOW" type="DOUBLE" />
  </rcon:record>
  </rcon:streamOutput>
  <rcon:streamOutput>
  <rcon:stream name="event" querygroup="PacketAnalysis" />
  <rcon:record name="OUTPUT_EVENT">
  <rcon:column name="ID" type="STRING" />
  <rcon:column name="TIMESTAMP" type="TIMESTAMP" />
  <rcon:column name="METRIC_ID" type="STRING" />
  <rcon:column name="METRIC_NAME" type="STRING" />
  <rcon:column name="TYPE" type="STRING" />
  <rcon:column name="EVENT" type="INT" />
  <rcon:column name="INFO_1" type="STRING" />
  <rcon:column name="INFO_2" type="STRING" />
  <rcon:column name="LEVEL" type="STRING" />
  </rcon:record>
  </rcon:streamOutput>
  </rcon:streamOutputs>
  </rcon:ReceiveConnectorDefinition>
  </cb:ReceiveCBDefinition>
  <!-- カスケーディング出力CB定義 -->
  <cb:OutputCBDefinition class="jp.co.Hitachi.soft.sdp.adaptor.callback.io.cascading.Casc
adingOutputCBImpl" name="Outputer1">
  <caclcon:CascadingClientConnectorDefinition>
```

```
  <!-- カスケーディングストリーム1-->
  <caclcon:cascading name="cascade01" source="OUTPUT_DATA">
  <caclcon:targetStream name="chartIdInput" querygroup="PacketAnalysis"/>
  </caclcon:cascading>
  <!-- カスケーディングストリーム2-->
  <caclcon:cascading name="cascade02" source="OUTPUT_EVENT">
  <caclcon:targetStream name="eventInput" querygroup="PacketAnalysis"/>
  </caclcon:cascading>
  <!--cascading用プロパティファイル -->
  <caclcon:cascadingFile path="../../appfile/static/cascading_properties.xml "/>
  </caclcon:CascadingClientConnectorDefinition>
  </cb:OutputCBDefinition>
  </adp:OutputAdaptorDefinition>
  </adp:InprocessGroupDefinition>
 </root:AdaptorCompositionDefinition>
```

- HASHING を使用してアダプターが複数のバックエンドサーバに送信する場合の出力アダプター定義の例

```
 <!-- カスケーディング出力アダプター定義2-->
 <adp:OutputAdaptorDefinition name="OutputAdaptor1" interval="0" charCode="UTF-8" lineFee
d="LF">
  <!-- 受信コネクターCB定義-->
  <cb:ReceiveCBDefinition class="jp.co.Hitachi.soft.sdp.adaptor.callback.sendreceive.Rece
iveConnectorCBImpl" name="Receiver1">
  <rcon:ReceiveConnectorDefinition>
  <rcon:streamOutputs>
  <rcon:streamOutput>
  <rcon:stream name="packetData" querygroup="PacketAnalysis" />
  <rcon:record name="DATA">
  <rcon:column name="ID" type="STRING" />
  <rcon:column name="TIMESTAMP" type="TIMESTAMP" />
  <rcon:column name="SPORT" type="STRING" />
  <rcon:column name="DPORT" type="STRING" />
  <rcon:column name="PACKET_SIZE" type="DOUBLE" />
  </rcon:record>
  </rcon:streamOutput>
  </rcon:streamOutputs>
  </rcon:ReceiveConnectorDefinition>
  </cb:ReceiveCBDefinition>
  <!-- カスケーディング出力CB定義 -->
  <cb:OutputCBDefinition class="jp.co.Hitachi.soft.sdp.adaptor.callback.io.cascading.Casc
adingOutputCBImpl" name="Outputer1">
  <caclcon:CascadingClientConnectorDefinition>
  <!-- カスケーディングストリーム -->
  <caclcon:cascading name="cascade03" source="DATA">
  <caclcon:targetStream name="dataInput" querygroup="MiddleAnalysis"/>
  <!-- HASHINGによるデータの送信 -->
  <caclcon:dispatch type="HASHING">
  <caclcon:hashingKey columnName="ID"/>
  <caclcon:hashingKey columnName="SPORT"/>
  <caclcon:hashingKey columnName="DPORT"/>
  </caclcon:dispatch>
  </caclcon:cascading>
  <!-- cascading用プロパティファイル -->
  <caclcon:cascadingFile path="../../appfile/static/cascading_properties.xml "/>
  </caclcon:CascadingClientConnectorDefinition>
```

```
  </cb:OutputCBDefinition>
 </adp:OutputAdaptorDefinition>
```

- ROUNDROBIN を使用してアダプターが複数のバックエンドサーバに送信する場合の出力アダプター定義の例

```
 <!-- カスケーディング出力アダプター定義1-->
 <adp:OutputAdaptorDefinition name="OutputAdaptor1" interval="0" charCode="UTF-8" lineFeed="LF">
  <!-- 受信コネクターCB定義 -->
  (HASHINGの例と同様)
  <!-- カスケーディング出力CB定義 -->
  <cb:OutputCBDefinition class="jp.co.Hitachi.soft.sdp.adaptor.callback.io.cascading.CascadingOutputCBImpl" name="Outputer1">
  <caclcon:CascadingClientConnectorDefinition>
  <!-- カスケーディングストリーム -->
  <caclcon:cascading name="cascade04" source="DATA">
  <caclcon:targetStream name="dataInput" querygroup="MiddleAnalysis"/>
  <!-- ROUNDROBINによるデータの送信 -->
  <caclcon:dispatch type="ROUNDROBIN"/>
  </caclcon:cascading>
  <!-- cascading用プロパティファイル -->
  <caclcon:cascadingFile path="../../appfile/static/cascading_properties.xml "/>
  </caclcon:CascadingClientConnectorDefinition>
  </cb:OutputCBDefinition>
 </adp:OutputAdaptorDefinition>
```

- STATIC ルールを使用してアダプターが複数のバックエンドサーバに送信する情報の例

```
 <!-- カスケーディング出力アダプター定義4-->
 <adp:OutputAdaptorDefinition name="OutputAdaptor1" interval="0" charCode="UTF-8" lineFeed="LF">
  <!-- 受信コネクターCB定義 -->
  (HASHINGの例と同様)
  <!-- カスケーディング出力CB定義 -->
  <cb:OutputCBDefinition class="jp.co.Hitachi.soft.sdp.adaptor.callback.io.cascading.CascadingOutputCBImpl" name="Outputer1">
  <caclcon:CascadingClientConnectorDefinition>
  <!-- カスケーディングストリーム -->
  <caclcon:cascading name="cascade05" source="DATA">
  <caclcon:targetStream name="dataInput" querygroup="MiddleAnalysis"/>
  <!-- STATICルールによるデータの送信 -->
  <caclcon:dispatch type="STATIC">
  <caclcon:condition name="route1">
  <caclcon:pattern columnName="ID" value="portA[0-9]+"/>
  </caclcon:condition>
  <caclcon:condition name="route2">
  <caclcon:pattern columnName="ID" value="portB[0-9]+"/>
  </caclcon:condition>
  <caclcon:condition name="route3">
  <caclcon:pattern columnName="ID" value="portC[0-9]+"/>
  </caclcon:condition>
  </caclcon:dispatch>
  </caclcon:cascading>
  <!-- cascading用プロパティファイル -->
  <caclcon:cascadingFile path="../../appfile/static/cascading_properties.xml "/>
```

```
  </caclcon:CascadingClientConnectorDefinition>
  </cb:OutputCBDefinition>
</adp:OutputAdaptorDefinition>
```

(2) cascading 用プロパティファイル

説明

SDP ブローカー連携によってカスケーディングアダプターを自動起動する場合，cascading 用プロパティファイルは不要です。SDP ブローカー連携をしないでカスケーディングアダプターを`hsdpstartinpro` コマンドによって手動起動する場合，cascading 用プロパティファイルを利用します。

cascading 用プロパティファイルでは，カスケーディングアダプターの通信方法について指定します。このファイルは，SDP の起動時に読み込まれます。SDP の起動後に変更された内容は適用されません。

ファイル名

任意の名前を指定できます。このファイルのパスをアダプター構成定義ファイルに指定してください。詳細については，「14.7.6　カスケーディングクライアントコネクター定義」を参照してください。

ファイル格納場所

任意のディレクトリに格納できます。このファイルのパスをアダプター構成定義ファイルに指定してください。詳細については，「14.7.6　カスケーディングクライアントコネクター定義」を参照してください。

形式

要素または属性※	説明	値	定義数
cascadingProperties	カスケーディングアダプターの通信情報を定義します。	-	1
retry	データ送信時のリトライ情報を定義します。この要素を指定しないと，アダプターはリトライを実行しようとします。	-	0～1
@times	データ送信時のリトライ回数を指定します。	1～100 の値です。	1
@timeout	リトライと次のリトライ間のタイムアウト（秒単位）を指定します。	1～100 の値です。	1
output	出力先として，アダプター構成定義ファイルで定義されるストリームを指定します。	-	1～16
@name	アダプター構成定義ファイル中に指定した<CascadingClientConnectorDefinition>要素の cascading@name と同じ名前を指定します。	1～100 文字の英数字（アンダーラインを含む）で指定します。最初の文字は英字である必要があります。	1
@protocol	通信プロトコルを指定します。	TCP を指定します。 カスケーディングアダプターは TCP で通信します。	1

要素または属性※	説明	値	定義数
retry	この表の２行目のretry 要素と同様です。 この表の２行目のretry 要素が指定されると，ここで設定する値が有効になります。	-	0～1
@times	この表の３行目のtimes 属性と同様です。	この表の３行目のtimes 属性と同様です。	1
@timeout	この表の４行目のtimeout 属性と同様です。	この表の４行目のtimeout 属性と同様です。	1
tcpParameters	TCP のパラメーターを指定します。	-	0～1
connectTimeout	バックエンドサーバに接続する際のタイムアウト値を定義します。	-	0～1
@timeout	タイムアウト値を秒単位で指定します。無期限のタイムアウトを設定するには，0 を指定します。	0～3600 の整数です。 デフォルト値は5 です。	1
destination	接続先の情報を定義します。 SDP ブローカー連携をしないで，接続先の TCP データ入力アダプターをhsdpstartinpro コマンドによって手動で起動する場合だけ，この属性を指定します。	-	1～32
@condition	アダプター構成定義ファイルに指定したcascading 要素内の<condition>タグのname 属性を 1 つ指定します。 ストリームの負荷分散方式がSTATIC の場合は，この属性を指定する必要があり，output 要素内で重複させることはできません。	1～100 の英数字（アンダーラインを含む）で指定します。最初の文字は英字である必要があります。	0～1
@host	接続先ホスト名を指定します。	1～63 文字の文字列です。IP アドレスを指定できます。	1
@port	接続先の TCP データ入力アダプターの TCP データ入力コネクター定義に定義されている TCP 待ち受けポート番号を指定します。	1024～65535 の整数です。	1

注※
最初の文字がアットマーク（@）の場合は，属性です。

（凡例）
-：該当なし

例 1

```
<?xml version="1.0" encoding="UTF-8"?>
<ccd:cascadingProperties xmlns:ccd="http://www.hitachi.co.jp/soft/xml/sdp/adaptor/definition
/cascading/CascadingProperties">
<ccd:retry times="5" timeout="1"/>
<ccd:output name="cascade01" protocol="TCP">
<ccd:destination host="10.197.111.10" port="6666"/>
```

```
</ccd:output>
<ccd:output name="cascade02" protocol="TCP">
<ccd:destination host="10.197.111.10" port="6666"/>
</ccd:output>
</ccd:cascadingProperties>
```

例 2

```
<?xml version="1.0" encoding="UTF-8"?>
<ccd:cascadingProperties xmlns:ccd="http://www.hitachi.co.jp/soft/xml/sdp/adaptor/definition
/cascading/CascadingProperties">
<ccd:retry times="5" timeout="1"/>
<!-- TCPでデータを送信-->
<ccd:output name="cascade03" protocol="TCP">
<!-- HASHINGによるデータの配信 -->
<ccd:destination host="192.168.0.20" port="31000"/>
<ccd:destination host="192.168.0.21" port="31000"/>
<ccd:destination host="192.168.0.22" port="31000"/>
</ccd:output>
<!-- TCPでデータを送信-->
<ccd:output name="cascade04" protocol="TCP">
<!-- ROUNDROBINによるデータの配信 -->
<ccd:destination host="192.168.0.100" port="31000"/>
<ccd:destination host="192.168.0.100" port="31001"/>
<ccd:destination host="192.168.0.100" port="31002"/>
</ccd:output>
</ccd:cascadingProperties>
```

メモ

デフォルトのファイルの保存場所は変更しないでください。保存場所を変更する場合は，ログ収集定義にファイルパスを追加してください。ログ収集定義の詳細については，「12.17　ログ収集定義ファイル」を参照してください。

(3)　ストリーム用プロパティファイル

カスケーディングストリームに関連するプロパティを次の表に示します。

説明

表 14-12　指定できるパラメーター

項番	パラメーター名	説明	デフォルト値	許容される値の範囲
1	stream.timestampPosition	stream.timestampMode がDataSource に設定され，入力ストリームがカスケーディングアダプターからのデータを受け付ける場合は，__systemtime__ を指定します。	なし	タイムスタンプデータの列名または __systemtime__

システムコンフィグプロパティファイル（system_config.properties）のパラメーターの詳細

- stream.timestampPosition=*時刻データ列名*

  「12.8　システムコンフィグプロパティファイル（system_config.properties）」と同じです。

14.8　アダプター構成定義ファイルのデータ編集用 CB 定義

ここでは，アダプター構成定義ファイルのデータ編集用 CB 定義について説明します。

データ編集用 CB 定義は，「14.6.3　編集用 CB 定義」で説明した編集用 CB 定義（DataEditCBDefinition タグ）の子要素として定義します。

データ編集用 CB 定義の一覧を次の表に示します。

表 14-13　データ編集用 CB 定義の一覧

項番	データ編集用 CB 定義	親要素	参照先
1	フォーマット変換定義 （FormatDefinition タグ）	編集用 CB 定義	14.8.1　フォーマット変換定義
2	マッピング定義 （MappingDefinition タグ）		14.8.2　マッピング定義
3	フィルター定義 （FilterDefinition タグ）		14.8.3　フィルター定義
4	レコード抽出定義 （RecordExtractionDefinition タグ）		14.8.4　レコード抽出定義

14.8.1　フォーマット変換定義

説明

フォーマット変換定義（FormatDefinition タグ）は，「14.6.3　編集用 CB 定義」で説明した編集用 CB 定義（DataEditCBDefinition タグ）の子要素として定義します。なお，フォーマット変換の処理については，「16.3.2　ファイルの入力のデータ処理の流れ」の「フォーマット変換」，または「16.8.2　ファイルへの出力のデータ処理の流れ」の「(4)　フォーマット変換」を参照してください。

記述形式

```
<FormatDefinition ioType=" {INPUT | OUTPUT} ">
  <common>
    <unmatchedFormat> {IGNORE | WARNING | ERROR}
    </unmatchedFormat>
    <format timestampformat="形式番号" year="開始年" month="開始月"/>
  </common>
  <records>
    <record name="レコード名" exp="レコード構成"
      timestampformat="形式番号" year="開始年" month="開始月">
      <field name="フィールド名"
        type=" {INT | SHORT | BYTE | LONG | BIG_DECIMAL | FLOAT | DOUBLE | STRING | DATE | TIME | TIME
STAMP} "
        pattern="パターン"/>
```

```
    </record>
  </records>
</FormatDefinition>
```

定義の詳細

FormatDefinition タグ（全体情報の定義）

フォーマット変換定義の全体情報を定義します。この定義は必ず 1 個だけ記述します。

ioType=”{INPUT | OUTPUT}”

この定義で指定する標準提供アダプターの種類を指定します。この属性は省略できません。

指定できる値を次に示します。

INPUT：入力アダプターでフォーマット変換を定義する場合に指定します。

OUTPUT：出力アダプターでフォーマット変換を定義する場合に指定します。

common タグ（共通定義）

フォーマット変換定義の共通情報を定義します。この定義は必ず 1 個だけ記述します。

unmatchedFormat タグ（変換パターンに一致しないレコードの定義）

このタグでは，フォーマット変換のパターンに一致しないレコードを検知したときの対処を指定します。

この定義は 1 個だけ記述できます。また，この定義は省略できます。省略した場合，ERROR が仮定されます。

指定できる値を次に示します。

- IGNORE

  検知した内容を無視して，標準提供アダプターの処理を続行します。

- WARNING

  警告メッセージを出力して，標準提供アダプターの処理を続行します。

- ERROR

  エラーメッセージを出力して，標準提供アダプターを停止します。

format タグ（フォーマット定義）

データ型種別で扱うTIMESTAMP 型の文字列表現について定義します。この定義は 1 個だけ記述できます。また，TIMESTAMP 型の文字列表現を使用しない場合，この定義は省略できます。

timestampformat=”*形式番号*”

TIMESTAMP 型の形式番号を1～4 の整数で指定します。

省略した場合，1 が仮定されます。

なお，format タグのtimestampformat 属性を指定し，かつrecords タグのtimestampformat 属性を指定した場合には，records タグでの指定内容が有効になります。

形式番号として指定できる値の意味を次の表に示します。

形式番号	文字列表現の形式	形式
1（デフォルト値）	*年-月-日 時:分:秒.ミリ秒（1～9桁）*	*yyyy-MM-dd HH:mm:ss.fffffffff*

形式番号	文字列表現の形式	形式
2※1	*月名※2 日 時:分:秒*	*MMM dd HH:mm:ss*
3	*年/月/日 時:分:秒.ミリ秒（3桁）*	*yyyy/MM/dd HH:mm:ss.SSS*
4	*日/月名※2/年:時:分:秒*	*dd/MMM/yyyy:HH:mm:ss*

注※1

形式番号2は，形式に年がありません。このため，入力アダプターで形式番号2を指定する場合には，year属性およびmonth属性で開始年月を指定してください。出力アダプターで形式番号2を指定する場合には，year属性およびmonth属性は指定できません。

なお，year属性およびmonth属性を指定しない場合，標準提供アダプターの起動時点のシステムの年月がTIMESTAMP型の開始年月となります。

注※2

英字3桁の英語の月名です。Jan（1月），Feb（2月），Mar（3月），Apr（4月），May（5月），Jun（6月），Jul（7月），Aug（8月），Sep（9月），Oct（10月），Nov（11月），Dec（12月）です。

year=”*開始年*”

timestampformat属性で2を指定した場合，この属性でTIMESTAMP型の開始年を指定します。年は，1970～2261の整数で指定します。この属性は，出力アダプターの場合には指定できません。

TIMESTAMP型の開始年月の指定については，「TIMESTAMP型の開始年月の指定」を参照してください。

month=”*開始月*”

timestampformat属性で2を指定した場合，この属性でTIMESTAMP型の開始月を指定します。年は，1～12の整数で指定します。この属性は，出力アダプターの場合には指定できません。

TIMESTAMP型の開始年月の指定については，「TIMESTAMP型の開始年月の指定」を参照してください。

recordsタグ（レコード群定義）

レコード群の情報を定義します。この定義は必ず1個だけ記述します。この定義は省略できません。

recordタグ（レコード定義）

各レコードの情報を定義します。この定義は10個まで記述できます。この定義は省略できません。

name=”*レコード名*”

レコードの情報を識別するための名称を1～100文字の半角英数字，およびアンダーライン（_）で指定します。先頭の文字に指定できるのは，半角英字だけです。

レコード名は，recordsタグ内で一意となるように指定してください。この属性は省略できません。

exp=”*レコード構成*”

レコードを構成するフィールドを1～1,000,000文字で指定します。

レコード構成内に，このレコード固有の区切り文字を指定できます。また，フィールドごとに異なる区切り文字を指定することもできます。この属性は省略できません。

レコード構成の記述形式については，「レコード構成の記述形式」を参照してください。

timestampformat="*形式番号*"

TIMESTAMP 型の形式番号を1～4 の整数で指定します。省略した場合，format タグのtimestampformat 属性の指定値が仮定されます。

なお，format タグのtimestampformat 属性を指定し，かつrecords タグのtimestampformat 属性を指定した場合には，records タグでの指定内容が有効になります。

形式番号として指定できる値の意味については，format タグのtimestampformat 属性の説明を参照してください。

year="*開始年*"

timestampformat 属性で2 を指定した場合，この属性でTIMESTAMP 型の開始年を指定します。年は，1970～2261 の整数で指定します。この属性は，出力アダプターの場合には指定できません。

TIMESTAMP 型の開始年月の指定については，「TIMESTAMP 型の開始年月の指定」を参照してください。

month="*開始月*"

timestampformat 属性で2 を指定した場合，この属性でTIMESTAMP 型の開始月を指定します。年は，1～12 の整数で指定します。この属性は，出力アダプターの場合には指定できません。

TIMESTAMP 型の開始年月の指定については，「TIMESTAMP 型の開始年月の指定」を参照してください。

field タグ（フィールド定義）

フィールド情報を定義します。この定義は 3,000 個まで記述できます。この定義は省略できません。

name="*フィールド名*"

フィールドの情報を識別するための名称を 1～100 文字の半角英数字，およびアンダーライン（_）で指定します。先頭の文字に指定できるのは，半角英字だけです。

フィールド名は，records タグ内で一意となるように指定してください。この属性は省略できません。

type="｛INT｜SHORT｜BYTE｜LONG｜BIG_DECIMAL｜FLOAT｜DOUBLE｜STRING｜DATE｜TIME｜TIMESTAMP｝"

フィールドの Java のデータ型に対応するデータ型を指定します。この属性は省略できません。

この属性に指定する値と，対応する Java のデータ型および CQL のデータ型については，「type 属性の指定値と，対応する Java のデータ型および CQL のデータ型」を参照してください。

pattern="*パターン*"

type 属性にSTRING を指定した場合，フィールド値のパターンを正規表現で指定します。正規表現の解析には，java.util.regex.Pattern クラスを使用します。このため，正規表現は，java.util.regex.Pattern クラスがサポートする正規表現の範囲で記述してください。「$」は使用できません。

なお，type 属性にSTRING 以外を指定している場合，または出力アダプターで定義している場合には，この属性は指定できません。指定した場合は，エラーとなります。

パターンには定数が指定できます。パターンに定数が指定された場合，フィールド値を定数として扱います。

この属性を省略した場合のデータ型種別の表記規則については，「データ型種別の表記規則」を参照してください。

記述例

```
<?xml version="1.0" encoding="UTF-8"?>
<root:AdaptorCompositionDefinition
xmlns:form="http://www.hitachi.co.jp/soft/xml/sdp/adaptor/definition/callback/FormatDefiniti
on">
<!-- 途中略 -->

<!-- 編集用CB定義 -->
<cb:DataEditCBDefinition class="jp.co.Hitachi.soft.sdp.adaptor.callback.dataedit.formattrans
late.InputFormatTranslatorCBImpl" name="editor1">
  <!-- フォーマット変換定義 -->
  <form:FormatDefinition ioType="INPUT">
    <form:common/>
      <!-- レコード群定義 -->
      <form:records>
        <form:record name="R1" exp="($_F1),($_F2),($_F3),($_F4),($_F5)">
        <!-- フィールド定義 -->
          <form:field name="F1" type="INT"/>
          <form:field name="F2" type="STRING" pattern="[^,]*"/>
          <form:field name="F3" type="STRING" pattern="[A-Z]+.[A-Z]+"/>
          <form:field name="F4" type="INT"/>
          <form:field name="F5" type="INT"/>
        </form:record>
      </form:records>
  </form:FormatDefinition>
</cb:DataEditCBDefinition>
```

TIMESTAMP 型の開始年月の指定

入力アダプターでformat タグのtimestampformat 属性に形式番号2 を指定し，format タグのyear 属性およびmonth 属性を指定した場合には，次のようにTIMESTAMP 型のフィールドの開始年月が決定されます。

- 入力アダプター起動直後の 1 つ目の入力レコード

  format タグのyear 属性およびmonth 属性で指定した値が，TIMESTAMP 型のフィールドの開始年月となります。
- 2 つ目以降の入力レコード

  前回の入力レコードのTIMESTAMP 型のフィールドの年月を基準年月として，開始年月を決定します。

なお，2 つ目以降の入力レコードの開始年月は，入力レコードのフィールドの月の値を基準年月と比較して，次のように決定されます。

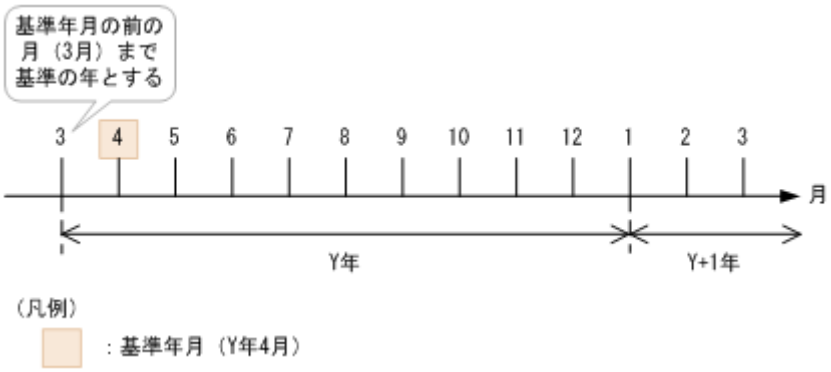
- 「入力レコードのフィールドの月の値 >= 基準年月の前の月」の場合

  基準年月の年がフィールドの年の値となります。ただし，基準年月の前の月が12月で，かつフィールドの月の値が12月の場合は，基準年月の前の年がフィールドの年の値となります。
- 「入力レコードのフィールドの月の値 < 基準年月の先月」の場合

基準年月の翌年がフィールドの年の値となります。ただし，基準年月の前の月が12月の場合は，基準年月の年がフィールドの年の値となります。

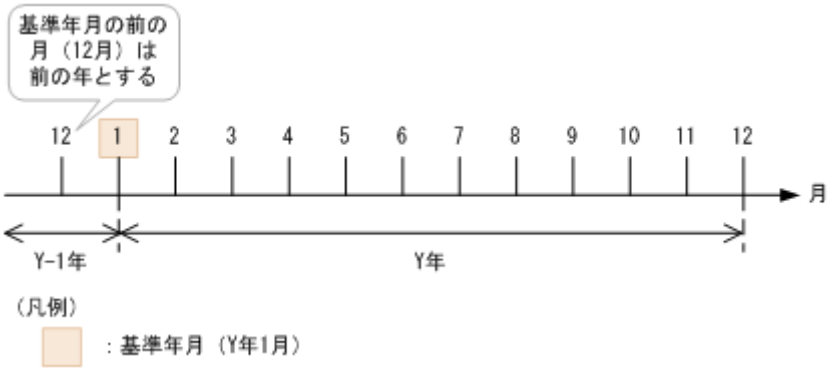
基準年月を *Y* 年4 月とした場合の例を次の図に示します。入力レコードのフィールドの月の値が3～12 の場合，フィールドの年の値は *Y* となります。月の値が1 または2 の場合，フィールドの年の値は *Y*+1 となります。

図 14-4　基準年月を Y 年 4 月とした場合



基準年月を *Y* 年1 月とした場合の例を次の図に示します。入力レコードのフィールドの月の値が12 の場合，フィールドの年の値は *Y*−1 となります。月の値が1～11 の場合，フィールドの年の値は *Y* となります。

図 14-5　基準年月を Y 年 1 月とした場合



レコード構成の記述形式

record タグのexp 属性で指定するレコード構成の記述形式について説明します。

記述形式を次に示します。

```
{〔区切り文字指定〕（$フィールド名指定）〔区切り文字指定〕
 |〔区切り文字指定〕（$フィールド名指定）〔区切り文字指定〕…}
```

区切り文字指定

このレコードを構成する個々のフィールドを区切る文字列を指定します。FormatDefinition タグのioType 属性がINPUT の場合，文字列として正規表現が使用できます。正規表現で特別な意味を持つ文字（「(」，「)」，「[」，「]」，「{」，「}」，「-」，「|」，「/」，「¥」，「.」，「*」，「?」，「+」，「^」，「$」）を本来の文字として指定するには，バックスラッシュ（¥）でエスケープする必要があります。

FormatDefinition タグのioType 属性がOUTPUT の場合，文字列として正規表現を使用できません。そのため，正規表現で特別な意味を持つ文字に対してエスケープする必要はありません。

フィールド名指定

このレコードを構成する個々のフィールド名を指定します。

レコード構成の記述規則を次に示します。

- 区切り文字指定に対するダブルクォーテーション（”）は不要です。
- フィールド名の指定の先頭には「($_」を，末尾には「)」を指定してください。
- フィールド名の指定では，このレコードを構成するすべてのフィールド名を指定してください。
- field タグでは，このレコードを構成するフィールド名を左から順番に指定してください。
- フィールド名の指定では，このレコードを構成するフィールド名を重複して指定できません。また，このレコードを構成するフィールド名以外のフィールド名は指定できません。
- 特殊文字を使用する場合は，文字を置換してください。文字の置換については，「14.2　アダプター構成定義ファイル作成上の注意事項」の特殊文字（記号）の対応表を参照してください。

レコード構成の記述例を次に示します。

記述例 1

- フィールドF1～F5 という 5 つのフィールドがある。
- すべてのフィールドは，コンマ「,」で区切られている。

このようなレコードのレコード構成は次のように記述します。

```
”($_F1),($_F2),($_F3),($_F4),($_F5)”
```

記述例 2

- フィールドF1～F3 という 3 つのフィールドがある。
- レコードの先頭が「<」で始まる。
- フィールドF1 とフィールドF2 が「> MSG」の区切り文字で区切られている。
- フィールドF2 とフィールドF3 が半角スペースの区切り文字で区切られている。

このようなレコードのレコード構成は次のように記述します。

```
”&lt;($_F1)&gt; MSG($_F2) ($_F3)”
```

「<」と「>」は特殊文字のため，それぞれ「<」および「>」に置き換えます。

type 属性の指定値と，対応する Java のデータ型および CQL のデータ型

field タグのtype 属性の指定値と，対応する Java のデータ型および CQL のデータ型を次の表に示します。

なお，各データ型の表記規則については，「表 14-15　field タグのpattern 属性を省略した場合のデータ型種別の表記規則」の説明を参照してください。また，CQL のデータ型については，マニュアル『Hitachi Streaming Data Platform アプリケーション開発ガイド』を参照してください。

表 14-14　type 属性の指定値と，対応する Java のデータ型および CQL のデータ型

項番	データ型種別（type 属性の指定値）	分類	データ形式	Java のデータ型	CQL のデータ型
1	INT	数値データ	整数型４バイト	プリミティブ型int	INT〔EGER〕
2	SHORT		整数型２バイト	プリミティブ型short	SMALLINT
3	BYTE		整数型１バイト	プリミティブ型byte	TINYINT
4	LONG		整数型８バイト	プリミティブ型long	BIGINT
5	BIG_DECIMAL		固定小数点数	java.math.BigDecimal クラス	DEC〔IMAL〕※1 NUMERIC※1
6					DECIMAL(*m*)※2 NUMERIC(*m*)※2
7	FLOAT		実数型４バイト	プリミティブ型float	REAL
8	DOUBLE		実数型８バイト	プリミティブ型double	FLOAT DOUBLE
9	STRING	文字データ	文字列	java.lang.String クラス	CHAR〔ACTER〕※3
10					CHAR〔ACTER〕(*n*)※4 VARCHAR(*p*)※5
11	DATE	日付/時刻データ	日付（*年月日*）	java.sql.Date クラス	DATE
12	TIME		時間（*時分秒*）	java.sql.Time クラス	TIME
13	TIMESTAMP		日時（*年月日＋時分秒＋ナノ秒*）	java.sql.Timestamp クラス	TIMESTAMP※6
14					TIMESTAMP〔(*q*)〕※7

注※1

桁数として 15 を仮定します。桁数が 15 を超える場合，タプル送信時にエラーとなります。

注※2

m は正整数で，$1 \leqq m \leqq 38$ です。桁数が m を超える場合，タプル送信時にエラーとなります。

注※3

文字数として1 を仮定します。文字数が1 を超える場合，タプル送信時にエラーとなります。

注※4

n は正整数で，$1 \leqq n \leqq 255$ です。文字数が n を超える場合，タプル送信時にエラーとなります。

注※5

p は正整数で，$1 \leqq p \leqq 32767$ です。文字数が p を超える場合，タプル送信時にエラーとなります。

注※6

年月日＋時分秒＋ミリ秒（3桁）を仮定します。ミリ秒以上の精度を指定した場合，タプル送信エラーとなります。

注※7

q は整数で，0≦q≦9 です。q は小数秒以下の桁数を示します。桁数に q 以上の精度を指定した場合，タプル送信エラーとなります。

データ型種別の表記規則

field タグのpattern 属性を省略した場合のデータ型種別の表記規則を次の表に示します。

表 14-15　field タグのpattern 属性を省略した場合のデータ型種別の表記規則

項番	データ型種別（type 属性の指定値）	各データ型のパターン（正規表現）	説明	変更の可否※
1	INT	"〔-〕{0,1}〔0-9〕+"	先頭がマイナス（-）の符号が 0 回または 1 回出現し，さらに0～9 の数字が 1 回以上繰り返される文字列のパターンです。	×
2	SHORT			×
3	BYTE			×
4	LONG			×
5	BIG_DECIMAL	"〔-〕{0,1}〔0-9〕+¥.〔0-9〕+"	次の文字列のパターンです。 • 先頭がマイナス（-）の符号が 0 回または 1 回出現します。 • 0～9 の数字が 1 回以上繰り返されます。 • ピリオド（.）を挟んで，0～9 の数字が 1 回以上繰り返されます。	×
6	FLOAT	"〔-〕{0,1}〔0-9〕+¥.〔0-9〕+"	次の文字列のパターンです。 • 先頭がマイナス（-）の符号が 0 回または 1 回出現します。 • 0～9 の数字が 1 回以上繰り返されます。 • ピリオド（.）を挟んで，0～9 の数字が 1 回以上繰り返されます。	×
7	DOUBLE			×
8	STRING	"〔^, ;〕*"	スペース，コンマ（,），およびセミコロン（;）を除く任意の文字が繰り返される文字列のパターンです。 データ型種別がSTRING の場合，パターンの変更（指定）ができます。 パターン変更（指定）例 区切り文字にピリオド（.）だけを使用する場合 "〔^ .〕+"	○

項番	データ型種別（type 属性の指定値）	各データ型のパターン（正規表現）	説明	変更の可否※
9	DATE	"〔0-9〕{1,4}-〔0-9〕{1,2}-〔0-9〕{1,2}"	*yyyy-mm-dd* 形式の文字列のパターンです。ただし，指定できる文字は次の範囲とし，桁数だけチェックします。 • *yyyy* ：0～9999 の数字 • *mm* ：0～99 の数字 • *dd* ：0～99 の数字 *mm*，*dd* がそれぞれ *MM*，*DD* の指定範囲外の値の場合，エラーメッセージ KFSP46322-E が出力されます。	×
10	TIME	"〔0-9〕{1,2}:〔0-9〕{1,2}:〔0-9〕{1,2}"	*hh*:*mm*:*ss* 形式の文字列のパターンです。ただし，指定できる文字は次の範囲とし，桁数だけチェックします。 • *hh* ：0～99 の数字 • *mm* ：0～99 の数字 • *ss* ：0～99 の数字 *hh*，*mm*，*ss* がそれぞれ *HH*，*MM*，*ss* の指定範囲外の値の場合，java.sql.Time の仕様に従って正規の時刻に換算します。 （例） 16:22:66→16:23:06	×
11	TIMESTAMP	条件によって異なります。詳細については，「表 14-16　データ型種別が TIMESTAMP の場合の表記規則」を参照してください。	条件によって異なります。詳細については，「表 14-16　データ型種別が TIMESTAMP の場合の表記規則」を参照してください。	×

（凡例）

○：変更できます。

×：変更できません。

注※

pattern 属性を指定することで，各データ型種別の文字列のパターンを変更できるかどうかを表します。

表 14-16　データ型種別が TIMESTAMP の場合の表記規則

条件	各データ型のパターン（正規表現）	説明
timestampformat 属性の値が1 の場合	"[0-9]{1,4}-[0-9]{1,2}-[0-9]{1,2}[0-9]{1,2}:[0-9]{1,2}:[0-9]{1,2}¥.[0-9]{1,9}"	*yyyy-mm-dd hh:mm:ss.fffffffff* 形式の文字列のパターンです。ただし，指定できる文字は次の範囲とし，桁数だけチェックします。※ • *yyyy* ：0～9999 の数字 • *mm* ：0～99 の数字 • *dd* ：0～99 の数字 • *hh* ：0～99 の数字 • *mm* ：0～99 の数字 • *ss* ：0～99 の数字 • *fffffffff* ：0～999999999 の数字
timestampformat 属性の値が2 の場合	"[A-Za-z]{3}[ ]+[0-9]{1,2}[ ]+[0-9]{1,2}:[0-9]{1,2}:[0-9]{1,2}"	*MMM dd HH:mm:ss* 形式の文字列のパターンです。ただし，指定できる文字は次の範囲とし，桁数だけチェックします。※ • *MMM* ：A～Z，a～z の英字（3 桁の英語の月名） • *dd* ：0～99 の数字 • *HH* ：0～99 の数字 • *mm* ：0～99 の数字 • *ss* ：0～99 の数字
timestampformat 属性の値が3 の場合	"[0-9]{1,4}/[0-9]{1,2}/[0-9]{1,2}[ ]+[0-9]{1,2}:[0-9]{1,2}:[0-9]{1,2}¥.[0-9]{3}"	*yyyy/MM/dd HH:mm:ss.SSS* 形式の文字列のパターンです。ただし，指定できる文字は次の範囲とし，桁数だけチェックします。※ • *yyyy* ：0～9999 の数字 • *MM* ：0～99 の数字 • *dd* ：0～99 の数字 • *HH* ：0～99 の数字

条件	各データ型のパターン（正規表現）	説明
timestampformat 属性の値が3 の場合	"[0-9]{1,4}/[0-9]{1,2}/[0-9]{1,2}[ ]+[0-9]{1,2}:[0-9]{1,2}:[0-9]{1,2}¥.[0-9]{3}"	• *mm* ：0～99 の数字 • *ss* ：0～99 の数字 • *SSS* ：000～999 の数字
timestampformat 属性の値が4 の場合	"[0-9]{1,2}/[A-Za-z]{3}/[0-9]{1,4}:[0-9]{1,2}:[0-9]{1,2}:[0-9]{1,2}"	*dd*/*MMM*/*yyyy*:*HH*:*mm*:*ss* 形式の文字列のパターンです。ただし，指定できる文字は次の範囲とし，桁数だけチェックします。※ • *dd* ：0～99 の数字 • *MMM* ：～Z，a～z の英字（3 桁の英語の月名） • *yyyy* ：0～9999 の数A 字 • *HH* ：0～99 の数字 • *mm* ：0～99 の数字 • *ss* ：0～99 の数字

注※

mm（月），*dd*，*hh*，*mm*（分），*ss*，*sss* がそれぞれの指定範囲外の値の場合，java.sql.Timestamp の仕様に従って正規の年，月，日，および時刻に換算します。

（例）

2001-13-10 16:22:66.101→2002-01-10 16:23:06.101

14.8.2　マッピング定義

説明

マッピング定義（MappingDefinition タグ）は，「14.6.3　編集用 CB 定義」で説明した編集用 CB 定義（DataEditCBDefinition タグ）の子要素として定義します。

なお，マッピングの処理については，「16.3.2　ファイルの入力のデータ処理の流れ」の「(1)　マッピング」，または「16.8.2　ファイルへの出力のデータ処理の流れ」の「(3)　マッピング」を参照してください。

記述形式

```
<MappingDefinition ioType=" {INPUT | OUTPUT} ">
  <source>
    <streams>
      <stream name="ストリーム名"
       querygroup="クエリグループ名">
        <column name="列名"
         type=" {INT | SHORT | BYTE | LONG | BIG_DECIMAL | FLOAT | DOUBLE | STRING  | DATE | TIME | TI
MESTAMP} "/>
      </stream>
    </streams>
  </source>
  <target>
    <streams>
      <stream name="ストリーム名"
       querygroup="クエリグループ名">
        <column name="列名"
         type=" {INT | SHORT | BYTE | LONG | BIG_DECIMAL | FLOAT | DOUBLE | STRING  | DATE | TIME | TI
MESTAMP} "/>
      </stream>
    </streams>
    <records>
      <record name="レコード名">
        <field name="フィールド名" type=" {INT | SHORT | BYTE | LONG | BIG_DECIMAL | FLOAT | DOUBL
E | STRING  | DATE | TIME | TIMESTAMP} "/>
    </records>
  </target>
  <intermediate>
    <mappings source="変換元名"
     querygroup="クエリグループ名"
     target="変換先名">
      <map source="変換元パス式"
       function="組み込み関数名" argument1="第1引数"
       argument2="第2引数" target="変換先パス式"/>
    </mappings>
  </intermediate>
</MappingDefinition>
```

定義の詳細

MappingDefinition タグ（全体情報の定義）

マッピング変換定義の全体情報を定義します。この定義は必ず 1 個だけ記述します。

ioType="{INPUT|OUTPUT}"

この定義で指定する標準提供アダプターの種類を指定します。この属性は省略できません。

指定できる値を次に示します。

INPUT：入力アダプターでマッピングを定義する場合に指定します。

OUTPUT：出力アダプターでマッピングを定義する場合に指定します。

source タグ（マッピング変換元定義）

マッピングの変換元のストリームの情報を定義します。

このタグに指定する内容は，次の表に示すように，MappingDefinition タグのioType 属性の指定値とマッピングの対象によって異なります。

ioType 属性の指定値	マッピングの対象	source タグに指定する内容
INPUT	レコードとストリーム間のマッピング	空要素タグを指定します。
	レコード間のマッピング	
OUTPUT	レコードとストリーム間のマッピング	streams タグを指定します。
	レコード間のマッピング	空要素タグを指定します。

target タグ（マッピング変換先定義）

マッピングの変換先のストリーム，またはレコードの情報を定義します。

このタグに指定する内容は，次の表に示すように，MappingDefinition タグのioType 属性の指定値とマッピングの対象によって異なります。

ioType 属性の指定値	マッピングの対象	target タグに指定する内容
INPUT	レコードとストリーム間のマッピング	streams タグを指定します。
	レコード間のマッピング	records タグを指定します。
OUTPUT	レコードとストリーム間のマッピング	空要素タグを指定します。
	レコード間のマッピング	records タグを指定します。

streams タグ（ストリーム群定義）

レコードとストリームとの間のマッピングの場合に，source タグまたはtarget タグ下で，変換元または変換先のストリーム群の情報を定義します。

この定義を記述する場合は，必ず 1 個だけ記述します。

stream タグ（ストリーム定義）

変換元または変換先のストリームの情報を定義します。

この定義は 1,024 個まで記述できます。この定義は省略できません。

name="*ストリーム名*"

ストリーム名を 1～100 文字の半角英数字，およびアンダーライン（_）で指定します。名前の先頭は半角英字だけ指定できます。また，ストリーム名とクエリグループ名の組は，streams タグ内で一意となるように指定してください。この属性は省略できません。

querygroup="*クエリグループ名*"

クエリグループ名を 1～64 文字の半角英数字，およびアンダーライン（_）で指定します。名前の先頭は半角英字だけ指定できます。また，ストリーム名とクエリグループ名の組は，streams タグ内で一意となるように指定してください。この属性は省略できません。

column タグ（列定義）

変換元または変換先のストリームの列の情報を定義します。

この定義は 3,000 個まで記述できます。この定義は省略できません。

name="＜列名＞"

列名を 1～100 文字の半角英数字，およびアンダーライン（_）で指定します。名前の先頭は半角英字だけ指定できます。また，列名は，streams タグ内の列定義の中で一意となるように指定してください。この属性は省略できません。

type="{INT | SHORT | BYTE | LONG | BIG_DECIMAL | FLOAT | DOUBLE | STRING | DATE | TIME | TIMESTAMP}"

CQL のデータ型に対応するデータ型種別を指定します。

この属性に指定する値と，対応する CQL のデータ型については，「14.8.1　フォーマット変換定義」の「type 属性の指定値と，対応する Java のデータ型および CQL のデータ型」を参照してください。

records タグ（レコード群定義）

レコード間のマッピングの場合に，target タグ下で，変換先のレコード群の情報を定義します。

この定義を記述する場合は，必ず 1 個だけ記述します。

record タグ（レコード定義）

変換先のレコードの情報を定義します。この定義は 1,024 個まで記述できます。この定義は省略できません。

name="*レコード名*"

レコード名を 1～100 文字の半角英数字，およびアンダーライン（_）で指定します。先頭の文字に指定できるのは，半角英字だけです。この属性は 1 個だけ記述できます。この属性は省略できません。レコード名は，該当の標準提供アダプター内で一意となるように指定してください。

field タグ（フィールド定義）

変換先のレコードのフィールド情報を定義します。この定義は 3,000 個まで記述できます。この定義は省略できません。

name="*フィールド名*"

フィールド名を 1～100 文字の半角英数字，およびアンダーライン（_）で指定します。先頭の文字に指定できるのは，半角英字だけです。この属性は 1 個だけ記述できます。この属性は省略できません。フィールド名は，該当のレコード内で一意となるように指定してください。

type="{INT | SHORT | BYTE | LONG | BIG_DECIMAL | FLOAT | DOUBLE | STRING | DATE | TIME | TIMESTAMP}"

フィールドの Java のデータ型に対応するデータ型を指定します。この属性は省略できません。

この属性に指定する値と，対応する CQL のデータ型については，「14.8.1　フォーマット変換定義」の「表 14-14　type 属性の指定値と，対応する Java のデータ型および CQL のデータ型」を参照してください。

intermediate タグ（マッピング中間定義）

マッピング変換元からマッピング変換先へのマッピング情報を定義します。

この定義は必ず 1 個だけ記述します。

mappings タグ（マッピング群定義）

マッピング情報を定義します。

この定義は 1,024 個まで記述できます。この定義は省略できません。

source="*変換元名*"

変換元のストリーム名，またはレコード名を 1～100 文字の半角英数字，およびアンダーライン（_）で指定します。この属性は省略できません。

この属性に指定する内容は，次の表に示すように，MappingDefinition タグの ioType 属性の指定値とマッピングの対象によって異なります。

ioType 属性の指定値	マッピングの対象	source 属性に指定する内容
INPUT	レコードとストリーム間のマッピング	変換元のレコード名を指定します。
	レコード間のマッピング	
OUTPUT	レコードとストリーム間のマッピング	stream タグの name 属性に指定したストリーム名を指定します。
	レコード間のマッピング	変換元のレコード名を指定します。

querygroup="*クエリグループ名*"

レコードとストリームとの間のマッピングの場合に，変換元または変換先のストリームが属するクエリグループ名を 1～64 文字の半角英数字，およびアンダーライン（_）で指定します。名前の先頭は半角英字だけ指定できます。また，クエリグループ名と変換元名の組，またはクエリグループ名と変換先名の組は，mappings タグ内で一意となるように指定してください。

この属性に指定する内容は，次の表に示すように，MappingDefinition タグの ioType 属性の指定値とマッピングの対象によって異なります。

ioType 属性の指定値	マッピングの対象	querygroup 属性に指定する内容
INPUT	レコードとストリーム間のマッピング	target タグ下の streams タグ内のクエリグループ名を指定します。
	レコード間のマッピング	指定しません。
OUTPUT	レコードとストリーム間のマッピング	source タグ下の streams タグ内のクエリグループ名を指定します。
	レコード間のマッピング	指定しません。

target="*変換先名*"

変換先のストリーム名，またはレコード名を 1～100 文字で指定します。変換先名は mappings タグ内で一意となるように指定してください。この属性は省略できません。

この属性に指定する内容は，次の表に示すように，MappingDefinition タグの ioType 属性の指定値とマッピングの対象によって異なります。

ioType 属性の指定値	マッピングの対象	target 属性に指定する内容
INPUT	レコードとストリーム間のマッピング	stream タグのname 属性に指定したストリーム名を指定します。
	レコード間のマッピング	record タグのname 属性に指定したレコード名を指定します。
OUTPUT	レコードとストリーム間のマッピング	変換先のレコード名を指定します。
	レコード間のマッピング	record タグのname 属性に指定したレコード名を指定します。

map タグ（マッピング情報定義）

マッピング情報を定義します。このタグの指定順は，変換先となるストリーム，またはレコードの構成の順番と一致させる必要があります。この定義は，3,000 個まで記述できます。この定義は省略できません。

source="*変換元パス式*"

変換元パス式として，変換元のフィールド名を 1～1,024 文字で指定します。この属性は省略できません。

この属性に指定する内容は，次の表に示すように，MappingDefinition タグのioType 属性の指定値とマッピングの対象によって異なります。

ioType 属性の指定値	マッピングの対象	source 属性に指定する内容
INPUT	レコードとストリーム間のマッピング	mappings タグのsource 属性に指定したレコード名に属するフィールド名を指定します。
	レコード間のマッピング	
OUTPUT	レコードとストリーム間のマッピング	mappings タグのsource 属性に指定したストリーム名に属するフィールド名を指定します。
	レコード間のマッピング	mappings タグのsource 属性に指定したレコード名に属するフィールド名を指定します。

function="*組み込み関数名*"

レコード間のマッピングを実施する場合は，map タグのfunction 属性で組み込み関数を使用して，マッピングの変換元の共通形式レコードから文字列や時刻を取得し，取得した文字列や時刻をマッピングの変換先の共通形式レコードに反映できます。

この属性では，組み込み関数を使用して文字列や時刻を取得する場合に，組み込み関数名を指定します。なお，レコードとストリーム間のマッピング（入力ストリーム用の共通形式レコードへのマッピング，または出力ストリーム用の共通形式レコードからのマッピング）では，この属性は指定できません。

この属性に指定できる文字列を次に示します。

regexsubstring：入力文字列から，正規表現の部分文字列を取得する場合に指定します。戻り値は，String 型の文字列です。regexsubstring を指定する場合，argument1 属性，およびargument2 属性で引数を指定してください。

getTupleTime：この値は，出力アダプターの場合だけ指定できます。出力アダプターで出力ストリームから出力されたタプルの時刻を取得する場合に指定します。戻り値は，TIMESTAMP 型の時刻（単位：ミリ秒）です。getTupleTime を指定する場合，argument1 属性，およびargument2 属性を指定する必要はありません。

なお，MappingDefinition タグのioType 属性の指定値がINPUT の場合には，getTupleTime は指定できません。

subTime：2 つのフィールドの時刻の差分を算出する場合に指定します。2 つの時刻のうち，大きい値から小さい値を引いて，時刻の差分を算出します。戻り値は，LONG 型の時刻（単位：ミリ秒）です。subTime を指定する場合，argument1 属性，およびargument2 属性で引数を指定してください。

argument1="*第1引数*"

function 属性で指定した組み込み関数の第 1 引数を指定します。

function 属性でregexsubstring を指定した場合

入力文字列が格納されているString 型のフィールドのフィールド名を 1～100 文字の半角英数字，およびアンダーライン（_）で指定します。

function 属性でsubTime を指定した場合

時刻が格納されているTIMESTAMP 型のフィールドのフィールド名を 1～100 文字の半角英数字，およびアンダーライン（_）で指定します。

argument2="*第2引数*"

function 属性で指定した組み込み関数の第 2 引数を指定します。

function 属性でregexsubstring を指定した場合

部分文字列の抽出条件を 2～1,024 文字の次に示す形式で指定します。

[*正規表現*] "("[*正規表現*]")"[*正規表現*]

正規表現の解析には，java.util.regex.Pattern クラスを使用します。正規表現は，java.util.regex.Pattern クラスがサポートする正規表現の範囲で記述してください。

第 1 引数で指定したフィールドの入力文字列が，第 2 引数で指定した正規表現の文字列に一致する場合には，「(」と「)」で囲まれた部分に相当する文字列（String 型）が戻り値となります。一致しない場合には，空文字が戻り値となります。

function 属性でsubTime を指定した場合

時刻が格納されているTIMESTAMP 型のフィールドのフィールド名を 1～100 文字の半角英数字，およびアンダーライン（_）で指定します。

target="*変換先パス式*"

変換先パス式として，変換先のフィールド名を 1～100 文字の半角英数字，およびアンダーライン（_）で指定します。フィールド名はmappings タグ内で一意となるように指定してください。この属性は省略できません。

次に示すとおり，この属性の指定内容は，MappingDefinition タグのioType 属性に指定した内容によって異なります。

ioType 属性にINPUT を指定した場合

変換先のストリーム名またはレコード名に属するフィールド名を指定します。

ioType 属性にOUTPUT を指定した場合

変換先のレコード名に属するフィールド名を指定します。

記述例

- 入力アダプターでレコードとストリーム間のマッピングの場合

  入力アダプターで，レコードとストリーム間のマッピングをする場合の記述例を次に示します。

```
<?xml version="1.0" encoding="UTF-8"?>
<root:AdaptorCompositionDefinition
<!-- 途中略 -->
 xmlns:map="http://www.hitachi.co.jp/soft/xml/sdp/adaptor/definition/callback/MappingDefi
nition">
<!-- 途中略 -->

<!-- 編集用CB定義 -->
<cb:DataEditCBDefinition class="jp.co.Hitachi.soft.sdp.adaptor.callback.dataedit.mapping.
InputMappingCBImpl" name="editor1">
<!-- マッピング定義 -->
  <map:MappingDefinition ioType="INPUT">
    <map:source/>
    <map:target>
      <map:streams>
        <map:stream name="S1" querygroup="QG1">
          <map:column name="id" type="INT"/>
          <map:column name="time" type="STRING"/>
          <map:column name="name" type="STRING"/>
          <map:column name="val" type="INT"/>
          </map:stream>
      </map:streams>
    </map:target>
    <map:intermediate>
      <map:mappings source="R1" querygroup="QG1" target="S1">
        <map:map source="F1" target="id"/>
        <map:map source="F2" target="time"/>
        <map:map source="F3" target="name"/>
        <map:map source="F5" target="val"/>
      </map:mappings>
    </map:intermediate>
  </map:MappingDefinition>
</cb:DataEditCBDefinition>
```

- 入力アダプターでレコード間のマッピングの場合

  入力アダプターで，レコード間のマッピングをする場合の記述例を次に示します。この記述例では，function 属性で組み込み関数を使用して，正規表現文字列と時刻の差分を取得しています。

```
<?xml version="1.0" encoding="UTF-8"?>
<root:AdaptorCompositionDefinition
<!-- 途中略 -->
 xmlns:map="http://www.hitachi.co.jp/soft/xml/sdp/adaptor/definition/callback/MappingDefi
nition">
<!-- 途中略 -->

<!-- 編集用CB定義 -->
<cb:DataEditCBDefinition class="jp.co.Hitachi.soft.sdp.adaptor.callback.dataedit.mapping.
```

```
InputMappingCBImpl" name="editor1">
<!-- マッピング定義 -->
  <map:MappingDefinition ioType="INPUT">
    <map:source/>
    <map:target>
      <map:records>
        <map:record name="RECORD1">
          <map:field name="id" type="INT"/>
          <map:field name="F21" type="STRING"/>
          <map:field name="F22" type="LONG"/>
        </map:record>
      </map:records>
    </map:target>
    <map:intermediate>
      <map:mappings source="RECORD0" target="RECORD1">
        <map:map source="F1" target="id"/>
        <map:map function="regexsubstring" argument1="F11"
         argument2=".*Data=([0-9]+).*" target="F21"/>
        <map:map function="subTime" argument1="F12" argument2=F13" target="F22"/>
      </map:mappings>
    </map:intermediate>
  </map:MappingDefinition>
</cb:DataEditCBDefinition>
```

- 出力アダプターでレコードとストリーム間のマッピングの場合

  出力アダプターで，レコードとストリーム間のマッピングをする場合の記述例を次に示します。

```
<?xml version="1.0" encoding="UTF-8"?>
<root:AdaptorCompositionDefinition
<!-- 途中略 -->
 xmlns:map="http://www.hitachi.co.jp/soft/xml/sdp/adaptor/definition/callback/MappingDefi
nition">
<!-- 途中略 -->

<!-- 編集用CB定義 -->
<cb:DataEditCBDefinition class="jp.co.Hitachi.soft.sdp.adaptor.callback.dataedit.mapping.
InputMappingCBImpl" name="editor1">
<!-- マッピング定義 -->
  <map:MappingDefinition ioType="INPUT">
    <map:source>
      <map:streams>
        <map:stream name="S1" querygroup="QG1">
          <map:column name="id" type="INT"/>
          <map:column name="time" type="TIMESTAMP"/>
          <map:column name="name" type="STRING"/>
        </map:stream>
      </map:streams>
    </map:source>
    <map:target/>
    <map:intermediate>
      <map:mappings source="S1" querygroup="QG1" target="RECORD1">
        <map:map source="id" target="F21"/>
        <map:map source="time" target="F22"/>
        <map:map source="name" target="F23"/>
      </map:mappings>
    </map:intermediate>
```

```
  </map:MappingDefinition>
</cb:DataEditCBDefinition>
```

- 出力アダプターでレコード間のマッピングの場合

  出力アダプターで，レコード間のマッピングをする場合の記述例を次に示します。この記述例では，function 属性で組み込み関数を使用して，正規表現文字列，タプルの時刻，および時刻の差分を取得しています。

```
<?xml version="1.0" encoding="UTF-8"?>
<root:AdaptorCompositionDefinition
<!-- 途中略 -->
 xmlns:map="http://www.hitachi.co.jp/soft/xml/sdp/adaptor/definition/callback/MappingDefi
nition">
<!-- 途中略 -->

<!-- 編集用CB定義 -->
<cb:DataEditCBDefinition class="jp.co.Hitachi.soft.sdp.adaptor.callback.dataedit.mapping.
InputMappingCBImpl" name="editor1">
<!-- マッピング定義 -->
  <map:MappingDefinition ioType="INPUT">
    <map:source/>
    <map:target>
      <map:records>
        <map:record name="RECORD1">
          <map:field name="F21" type="STRING"/>
          <map:field name="F22" type="TIMESTAMP"/>
          <map:field name="F23" type="LONG"/>
        </map:record>
      </map:records>
    </map:target>
    <map:intermediate>
      <map:mappings source="RECORD0" target="RECORD1">
        <map:map function="regexsubstring" argument1="F11"
         argument2=".*Data=([0-9]+).*" target="F21"/>
        <map:map function="getTupleTime" target="F22"/>
        <map:map function="subTime" argument1="F12" argument2=F13" target="F23"/>
      </map:mappings>
    </map:intermediate>
  </map:MappingDefinition>
</cb:DataEditCBDefinition>
```

14.8.3　フィルター定義

説明

フィルター定義（FilterDefinition タグ）は，「14.6.3　編集用 CB 定義」で説明した編集用 CB 定義（DataEditCBDefinition タグ）の子要素として定義します。

なお，レコードのフィルタリングの処理については，「16.5　レコードのフィルタリング」を参照してください。

記述形式

```
<FilterDefinition>
  <record source="フィルター対象のレコード名"
   conditionName="レコード条件名" condition=" {AND | OR} ">
    <field source="フィールド名"
     condition=" {eq | ge | gt | le | lt | ne} " value="条件値"/>
  </record>
</FilterDefinition>
```

定義の詳細

FilterDefinition タグ（全体情報の定義）

フィルター定義の全体情報を定義します。この定義は必ず 1 個だけ記述します。

record タグ（レコード条件の定義）

レコード条件を定義します。この定義は 10 個まで記述できます。この定義は省略できません。

レコード条件とは，フィルター対象のレコードを取捨選択するための条件の一つです。レコード条件には，フィルター対象のレコード名を指定します。

source="*フィルター対象のレコード名*"

レコードのフィルタリングの前に処理したコールバックから渡されたレコード形式の中で，フィルター対象とするレコード名を 1～100 文字の半角英数字，およびアンダーライン（_）で指定します。先頭の文字に指定できるのは，半角英字だけです。

この属性は 1 個だけ記述できます。

conditionName="*レコード条件名*"

レコード条件名を 1～100 文字の半角英数字，およびアンダーライン（_）で指定します。先頭の文字に指定できるのは，半角英字だけです。

この定義は必ず 1 個だけ記述します。また，FilterDefinition タグ内で一意となるように指定してください。

condition=" {AND | OR} "

レコード条件の論理条件を指定します。省略した場合，AND が仮定されます。

指定できる値を次に示します。

AND：複数のフィールド条件にすべて一致したフィールドを持つレコードを出力結果として出力します。

OR：複数のフィールド条件に一つでも一致したフィールドを持つレコードを出力結果として出力します。

field タグ（フィールド条件の定義）

フィールド条件を定義します。この定義は 10 個まで記述できます。この定義は省略できません。

フィールド条件とは，フィルター対象のレコードを取捨選択するための条件の一つです。フィールド条件には，フィルター対象のレコードのフィールド名，比較演算記号，および条件値を指定します。

source="*フィールド名*"

record タグのsource で指定したレコード形式のフィールド名を 1～100 文字の半角英数字，およびアンダーライン（_）で指定します。先頭の文字に指定できるのは，半角英字だけです。

この属性は省略できません。

condition="{eq | ge | gt | le | lt | ne}"

フィールド条件の比較演算子を表す値を指定します。

この属性に指定できる値は，field タグのvalue 属性に文字データを指定するか，数値データを指定するかによって異なります。指定できる値を次の表に示します。省略した場合，eq が仮定されます。

この属性の指定値	比較演算子	value 属性の指定値	
		文字データ	数値データ
eq	=	○	○
ge	>=	×	○
gt	>	×	○
le	<=	×	○
lt	<	×	○
ne	!=	○	○

（凡例）

○：指定できます。

×：指定できません。

比較演算子の意味を次の表に示します。

比較演算子	比較演算子の使用例	使用例の意味
=	A = B	A はB と等しい
>=	A >= B	A はB 以上
>	A > B	A はB より大きい
<=	A <= B	A はB 以下
<	A < B	A はB より小さい
!=	A != B	A はB と等しくない

value="*条件値*"

フィールド条件の条件値を文字データ，または数値データで指定します。使用できる文字，または数値を次に示します。この属性は省略できません。

文字データの場合

指定できる文字数は 1～128 文字です。

文字列として正規表現が使用できます。正規表現の解析には，java.util.regex.Pattern クラスを使用します。このため，正規表現は，java.util.regex.Pattern クラスがサポートする正規表現の範囲で記述してください。

正規表現で特別な意味を持つ文字を本来の文字として使用するには，バックスラッシュ（¥）でエスケープする必要があります。

正規表現で特別な意味を持つ文字を次に示します。

「(」，「)」，「[」，「]」，「{」，「}」，「-」，「|」，「/」，「¥」，「.」，「*」，「?」，「+」，「^」，「$」

数値データの場合

-9223372036854775808～9223372036854775807 の整数を指定できます。

記述例

```
<?xml version="1.0" encoding="UTF-8"?>
<root:AdaptorCompositionDefinition
<!-- 途中略 -->
 xmlns:filter="http://www.hitachi.co.jp/soft/xml/sdp/adaptor/definition/callback/FilterDefin
ition">
<!-- 途中略 -->

<!-- 編集用CB定義 -->
<cb:DataEditCBDefinition class="jp.co.Hitachi.soft.sdp.adaptor.callback.dataedit.filter.Filt
erCBImpl" name="filter1">
  <!-- フィルター定義 -->
  <filter:FilterDefinition>
    <!-- レコード条件の定義 -->
    <filter:record source="R1" conditionName="filterName1" condition="AND">
      <!-- フィールド条件の定義 -->
      <filter:field source="F11" condition="ge"  value="1"/>
      <filter:field source="F11" condition="le"  value="100"/>
      <filter:field source="F21" condition="eq"  value=".*TARO.*"/>
    </filter:record>
    <!-- レコード条件の定義 -->
    <filter:record source="R2" conditionName="filterName2" operator="OR">
      <!-- フィールド条件の定義 -->
      <filter:field source="F21" value="1"/>
      <filter:field source="F22" condition="ne"  value=".*HANAKO.*"/>
    </filter:record>
  </filter:FilterDefinition>
</cb:DataEditCBDefinition>
```

14.8.4　レコード抽出定義

説明

レコード抽出定義（RecordExtractionDefinition タグ）は，「14.6.3　編集用 CB 定義」で説明した編集用 CB 定義（DataEditCBDefinition タグ）の子要素として定義します。

なお，レコードの抽出の処理については，「16.6　レコードの抽出」を参照してください。

記述形式

```
<RecordExtractionDefinition>
  <targetrecords>
    <targetrecord name="抽出対象レコード名">
      <record source="レコード名"
       timeposition="時刻フィールド名" condition=" {AND | OR} ">
        <field source="フィールド名"
         condition=" {eq | ge | gt | le | lt | ne} " value="条件値"/>
      </record>
    </targetrecord>
  </targetrecords>
  <extractions size="レコード最大保持数" timeout=" {ON | OFF} "
   samerecord=" {overwrite | delete} " >
    <extraction name="抽出条件名" timelimit="タイムアウト時間">
      <targets>
        <target sourceL ="抽出対象レコード名"
         sourceR="抽出対象レコード名" condition="AND">
          <fieldcondition sourceL="フィールド名"
           condition="eq" sourceR="フィールド名"/>
        </target>
      </targets>
      <extractrecord name="抽出レコード名">
        <select source="抽出対象レコード名"/>
      </extractrecord>
      <timeoutrecord name="タイムアウトレコード名" />
    </extraction>
  </extractions>
</RecordExtractionDefinition>
```

定義の詳細

RecordExtractionDefinition タグ（全体情報の定義）

レコード抽出定義の全体情報を定義します。この定義は必ず 1 個だけ記述します。

targetrecords タグ（抽出対象条件群の定義）

抽出対象条件群の情報を定義します。抽出対象条件とは，抽出対象のレコードを取捨選択するための条件の一つです。抽出対象条件には，レコード条件とフィールド条件を定義します。レコード条件（record タグ）では抽出対象のレコード名を，フィールド条件（field タグ）では抽出対象のレコードのフィールド値を指定します。

targetrecord タグ（抽出対象条件の定義）

抽出対象条件を定義します。この定義は 10 個まで記述できます。この定義は省略できません。

name="*抽出対象レコード名*"

抽出したレコードに付けるレコード名を 1～100 文字の半角英数字，およびアンダーライン（_）で指定します。先頭の文字に指定できるのは，半角英字だけです。この定義は 1 個だけ記述できます。

この属性で指定したレコード名は，select タグのsource 属性，target タグのsourceL 属性とsourceR 属性で指定します。

record タグ（レコード条件定義）

レコード条件を定義します。この定義は 1 個だけ記述できます。

source="*レコード名*"

レコードの抽出の前に処理したコールバックから渡されたレコード形式の中で，抽出対象とするレコード名を 1～100 文字の半角英数字，およびアンダーライン（_）で指定します。この属性は省略できません。

timeposition="*時刻フィールド名*"

TIMESTAMP 型の時刻情報のフィールドを 1～100 文字の半角英数字，およびアンダーライン（_）で指定します。この属性は省略できません。

condition="{AND | OR}"

レコード条件の論理条件を定義します。省略した場合，AND が仮定されます。

指定できる値を次に示します。

AND：複数のフィールド条件にすべて一致したフィールドを持つレコードを出力結果として出力します。

OR：複数のフィールド条件に一つでも一致したフィールドを持つレコードを出力結果として出力します。

field タグ（フィールド条件定義）

フィールド条件を定義します。この定義は 10 個まで記述できます。この定義は省略できません。

フィールド条件は，抽出対象とするレコードのフィールドごとに指定します。

source="*フィールド名*"

record タグのsource で指定したレコード形式のフィールド名を 1～100 文字の半角英数字，およびアンダーライン（_）で指定します。この属性は省略できません。

condition="{eq | ge | gt | le | lt | ne}"

フィールド条件の比較演算子を表す値を指定します。

この属性に指定できる値は，field タグのvalue 属性に文字データを指定するか，数値データを指定するかによって異なります。指定できる値を次に示します。省略した場合，eq が仮定されます。

文字データの場合

eq，またはne が指定できます。

数値データの場合

eq，ge，gt，le，lt，またはne が指定できます。

指定できる値の意味については，「14.8.3　フィルター定義」のfield タグのcondition 属性の説明を参照してください。

value="*条件値*"

フィールド条件の条件値を文字データ，または数値データで指定します。使用できる文字，または数値を次に示します。この属性は省略できません。

文字データの場合

指定できる文字数は 1～128 文字です。

文字列として正規表現が使用できます。正規表現の解析には，java.util.regex.Pattern クラスを使用します。このため，正規表現は，java.util.regex.Pattern クラスがサポートする正規表現の範囲で記述してください。

正規表現で特別な意味を持つ文字を本来の文字として使用するには，バックスラッシュ（¥）でエスケープする必要があります。

正規表現で特別な意味を持つ文字を次に示します。

「(」，「)」，「[」，「]」，「{」，「}」，「-」，「|」，「/」，「¥」，「.」，「*」，「?」，「+」，「^」，「$」

数値データの場合

-9223372036854775808～9223372036854775807 の整数を指定できます。

extractions タグ（抽出条件群の定義）

抽出条件群の情報を定義します。抽出条件とは，抽出対象のレコードを取捨選択するための条件の一つです。この定義は省略できません。

size="*レコード最大保持数*"

レコードバッファーに保持するレコード数の上限値を1～1000000 の整数で指定します。この属性を省略した場合，10000 が仮定されます。

timeout="{ON | OFF} "

extraction タグのtimelimit 属性で指定した時間が経過し，レコードバッファーに保持しているレコードがタイムアウトした場合に，レコード（タイムアウトレコード）を出力するかどうかを指定します。

この属性を省略した場合，OFF が仮定されます。

指定できる値を次に示します。

ON：タイムアウトした場合に，タイムアウトレコードを出力します。

OFF：タイムアウトした場合に，タイムアウトレコードを出力しません。

なお，タイムアウトレコードについては，timeoutrecord タグで，レコード名を定義します。

samerecord="{overwrite | delete} "

同一の抽出対象のレコードが入力された場合の対処を指定します。省略した場合，overwrite が仮定されます。

指定できる値を次に示します。

overwrite：同一レコードと見なされた既存のレコードが，レコードバッファーのどの位置にあるかによって処理が異なります。最も後ろのレコードの場合，新しいレコードで，レコードバッファーに保持していた既存のレコードを上書きします。最も後ろのレコードでない場合，上書きしたレコードより後ろのレコードを破棄して，新しいレコードを保持します。

delete：入力されたレコードを破棄します。

extraction タグ（抽出条件の定義）

抽出条件を定義します。この定義は10 個まで記述できます。この定義は省略できません。

name="*抽出条件名*"

抽出条件名を 1～100 文字の半角英数字，およびアンダーライン（_）で指定します。先頭の文字に指定できるのは，半角英字だけです。

この定義は 1 個だけ記述できます。

timelimit="*タイムアウト時間*"

レコードバッファーに保持したレコードをタイムアウトと見なすまでの，レコードの生存時間（単位：ミリ秒）を1～3600000 の整数（1 ミリ秒～1 時間）で指定します。

この属性を省略した場合，10000 が仮定されます。

targets タグ（レコード間条件群の定義）

レコード間条件群を定義します。

レコード間条件とは，抽出条件を成立させるための条件の一つです。target タグではレコードの入力順序を指定し，fieldcondition タグではレコード間でフィールド値を比較するためのフィールド名を指定します。

target タグ（レコード間条件の定義）

レコード間条件として，レコードの入力順序を指定します。この定義は9 個まで記述できます。この定義は省略できません。

この定義では，sourceL 属性，およびsourceR 属性でレコード名を指定して，レコードの入力順序を定義します。例えば，次のように指定した場合には，レコードの入力順序は，R1→R2→R3 となり，この入力順序どおりに入力されたレコードがレコード条件に一致するレコードとなります。

```
       :
<rex:target sourceL="R1" sourceR="R2">
       :
<rex:target sourceL="R2" sourceR="R3">
       :
```

sourceL="*抽出対象レコード名*"

レコード間条件の左辺のレコード名を 1～100 文字の半角英数字，およびアンダーライン（_）で指定します。先頭の文字に指定できるのは，半角英字だけです。

sourceR="*抽出対象レコード名*"

レコード間条件の右辺のレコード名を 1～100 文字の半角英数字，およびアンダーライン（_）で指定します。先頭の文字に指定できるのは，半角英字だけです。

condition="AND"

レコード間条件の論理条件を指定します。省略した場合，AND が仮定されます。

指定できる値を次に示します。

AND：複数のレコード間条件にすべて一致したレコードを出力結果として出力します。

fieldcondition タグ（フィールドの定義）

レコード間条件として，フィールド名を指定します。この定義は10 個まで記述できます。この定義は省略できません。

この定義では，sourceL 属性，およびsourceR 属性で，target タグで指定したレコードのフィールド名を指定します。例えば，次のように指定した場合には，レコードR1 のフィールドF11 と，レコードR2 のフィールドF21 の値を比較することになります。レコードR1 のフィールドF11 と，レコードR2 のフィールドF21 の値が一致する場合は，レコード間条件に一致すると判定されます。

```
       :
<rex:target sourceL="R1" sourceR="R2">
<rex:fieldcondition sourceL="F11" condition="eq" sourceR="F21"/>
       :
```

sourceL="*フィールド名*"

target タグのsourceL 属性で指定したレコードのフィールド名を 1～100 文字の半角英数字，およびアンダーライン（_）で指定します。先頭の文字に指定できるのは，半角英字だけです。

condition="eq"

レコード条件の比較演算子を表す値を指定します。指定できる値は，文字データと数値データのどちらを指定する場合も，比較演算子「=」を表す，eq です。省略した場合，eq が仮定されます。

sourceR="*フィールド名*"

target タグのsourceR 属性で指定したレコードのフィールド名を 1～100 文字の半角英数字，およびアンダーライン（_）で指定します。先頭の文字に指定できるのは，半角英字だけです。

extractrecord タグ（抽出レコードの定義）

抽出レコードを定義します。

抽出レコードとは，抽出条件に一致したレコードを結合して，新たに生成するレコードのことです。

name="*抽出レコード名*"

抽出レコードのレコード名を 1～100 文字の半角英数字，およびアンダーライン（_）で指定します。先頭の文字に指定できるのは，半角英字だけです。また，抽出レコード名は標準提供アダプター内で一意となるように指定してください。

select タグ（抽出対象レコードの定義）

抽出レコード生成のために結合するレコードを定義します。この定義は 10 個まで記述できます。この定義は省略できません。

source="*抽出対象レコード名*"

targetrecord タグのname 属性で指定したレコード名を 1～100 文字の半角英数字，およびアンダーライン（_）で指定します。この属性は省略できません。

timeoutrecord タグ（タイムアウトレコードの定義）

タイムアウトの発生時に出力するレコードを定義します。この定義は，extractions タグのtimeout 属性にON を指定した場合は，必ず定義してください。timeout 属性にOFF を指定した場合は，この定義は無視されます。

name="*タイムアウトレコード名*"

出力するレコード名を 1～100 文字の半角英数字，およびアンダーライン（_）で指定します。この属性は省略できません。また，タイムアウトレコード名は標準提供アダプター内で一意となるように指定してください。

記述例

```
<?xml version="1.0" encoding="UTF-8"?>
<root:AdaptorCompositionDefinition
 xmlns:rex ="http://www.hitachi.co.jp/soft/xml/sdp/adaptor/definition/callback/RecordExtract
ionDefinition">
<!-- 途中略 -->

<!-- 編集用CB定義 -->
<cb:DataEditCBDefinition class="jp.co.Hitachi.soft.sdp.adaptor.callback.dataedit.recordextra
ct.RecordExtractionCBImpl" name="extractor1">
  <!-- レコード抽出定義 -->
  <rex:RecordExtractionDefinition>
    <!-- 抽出対象条件群の定義 -->
    <rex:targetrecords>
      <rex:targetrecord name="element1">
        <rex:record source="R1" timeposition="F11" condition="AND">
          <rex:field source="F12" condition="eq" value="TARO"/>
          <rex:field source="F13" condition="gt" value="100"/>
        </rex:record>
      </rex:targetrecord>
      <rex:targetrecord name="element2">
        <rex:record source="R2" timeposition="F21" condition="OR">
          <rex:field source="F22" condition="eq" value="JIRO"/>
          <rex:field source="F23" condition="le" value="50"/>
        </rex:record>
      </rex:targetrecord>
    </rex:targetrecords>
    <!-- 抽出条件群の定義 -->
    <rex:extractions size="100000" timeout="ON" samerecord="overwrite">
      <rex:extraction name="T1" timelimit="10000">
        <rex:targets>
          <rex:target sourceL="element1" sourceR="element2" condition="AND">
            <rex:fieldcondition sourceL="F13" condition="eq" sourceR="F23"/>
            <rex:fieldcondition sourceL="F14" condition="eq" sourceR="F24"/>
          </rex:target>
        </rex:targets>
        <!-- 抽出レコードの定義 -->
        <rex:extractrecord name="ER1">
          <rex:select source="element1"/>
          <rex:select source="element2"/>
        </rex:extractrecord>
        <!-- タイムアウトレコードの定義 -->
        <rex:timeoutrecord name="TIMEOUT"/>
      </rex:extraction>
    </rex:extractions>
  </rex:RecordExtractionDefinition>
</cb:DataEditCBDefinition>
```

この記述例での定義内容を次に示します。

- 抽出対象のレコード形式

  R1，R2

- R1，R2 のフィールドの構成

R1 の場合

F11（TIMESTAMP 型），F12（STRING 型），F13（INT 型），F14（INT 型）

R2 の場合

F21（TIMESTAMP 型），F22（STRING 型），F23（INT 型），F24（INT 型）

- 抽出対象条件

  R1 の場合

  F12 がTARO であり，かつF13 が100 より大きい。

  R2 の場合

  F22 がJIRO，またはF23 が50 以下である。

- 抽出条件

  レコードは，R1，R2 の順番で入力される。

  R1 のF13 と，R2 のF23 の値が等しい。

  R1 のF14 と，R2 のF24 の値が等しい。

- 抽出レコード

  R1 とR2 を結合したレコードを生成する。

- タイムアウト

  レコードがタイムアウトするまでの時間は10 秒である。また，タイムアウトした場合，TIMEOUT というレコード名でレコードを出力する。

14.9　アダプター構成定義ファイルの送受信用 CB 定義

ここでは，アダプター構成定義ファイルの送受信用 CB 定義について説明します。

入力ストリーム定義（streamInfo タグ）は，「14.6.4　送信用 CB 定義」で説明した送信用 CB 定義（SendCBDefinition タグ）の子要素として定義します。また，出力ストリーム定義（streamInfo タグ）は，「14.6.5　受信用 CB 定義」で説明した受信用 CB 定義（ReceiveCBDefinition タグ）の子要素として定義します。

送受信用 CB 定義の一覧を次の表に示します。

表 14-17　送受信用 CB 定義の一覧

項番	編集用 CB 定義	親要素	参照先
1	入力ストリーム定義（streamInfo タグ）	送信用 CB 定義	14.9.1　入力ストリーム定義
2	出力ストリーム定義（streamInfo タグ）	受信用 CB 定義	14.9.2　出力ストリーム定義

14.9.1　入力ストリーム定義

説明

送信用 CB 定義では送信先となる入力ストリームに関する情報を指定します。

記述形式

```
<streamInfo name="入力ストリーム名"
 querygroup="クエリグループ名"/>
```

定義の詳細

この定義は 1,024 個まで記述できます。この定義は省略できません。

streamInfo タグ（全体情報の定義）

入力ストリーム定義の全体情報を定義します。

name="*入力ストリーム名*"

入力ストリーム名を 1～100 文字の半角英数字，およびアンダーライン（_）で指定します。名前の先頭は半角英字だけ指定できます。

入力ストリーム名は送信用 CB 定義内で一意となるように指定してください。この属性は省略できません。

なお，クエリ定義ファイルのregister stream 句で指定したストリーム名が，アダプター構成定義ファイルで指定する入力ストリーム名となります。register stream 句については，マニュアル『Hitachi Streaming Data Platform アプリケーション開発ガイド』を参照してください。

querygroup="*クエリグループ名*"

クエリグループ名を 1～64 文字の半角英数字，およびアンダーライン（_）で指定します。名前の先頭は半角英字だけ指定できます。この属性は省略できません。

14.9.2 出力ストリーム定義

説明

受信用 CB 定義では送信元となる出力ストリームに関する情報を指定します。

記述形式

```
<streamInfo name="出力ストリーム名"
 querygroup="クエリグループ名"/>
```

定義の詳細

この定義は 1,024 個まで記述できます。また，この定義は省略できません。

streamInfo タグ（全体情報の定義）

入力ストリーム定義の全体情報を定義します。

name="*出力ストリーム名*"

出力ストリーム名を 1～100 文字の半角英数字，およびアンダーライン（_）で指定します。名前の先頭は半角英字だけ指定できます。

出力ストリーム名は受信用 CB 定義内で一意となるように指定してください。この属性は省略できません。

クエリ定義ファイルのregister query 句で指定したクエリ名が，アダプター構成定義ファイルで指定する出力ストリーム名となります。

register query 句については，マニュアル『Hitachi Streaming Data Platform アプリケーション開発ガイド』を参照してください。

querygroup="*クエリグループ名*"

クエリグループ名を 1～64 文字の半角英数字，およびアンダーライン（_）で指定します。名前の先頭は半角英字だけ指定できます。この属性は省略できません。

14.9.3　送信コネクター定義

形式

```
<SendConnectorDefinition>
 <streamInputs>
 <streamInput>
 <record name="レコード名" />
 <stream name="ストリーム名" querygroup="クエリグループ名" />
 </streamInput>
 :
 </streamInputs>
</SendConnectorDefinition>
```

説明

表 14-18　要素と属性の一覧

要素または属性※	説明	値	定義数
SendConnectorDefinition	送信コネクターを定義します。	-	1
streamInputs	入力ストリーム群に関する情報を定義します。	-	1
streamInput	入力ストリームに関する情報を定義します。	-	1～1024
record	送信先のレコードに関する情報を定義します。	-	1
@name	レコードの名前を指定します。	1～100 の半角英数字（アンダーライン（_）を含む）で指定します。最初の文字は半角の英字である必要があります。このレコード名は，対応する標準提供アダプター内で一意である必要があります。	1
stream	送信先の入力ストリームに関する情報を定義します。	指定したストリーム名とクエリグループ名のペアは，対応する標準提供アダプター内で一意である必要があります。	1
@name	入力ストリームの名前を指定します。	1～64 の半角英数字（アンダーライン（_）を含む）で指定します。最初の文字は半角の英字である必要があります。	1
@querygroup	クエリグループの名前を指定します。	1～64 の半角英数字（アンダーライン（_）を含む）で指定します。最初の文字は半角の英字である必要があります。	1

注※

要素または属性※	説明	値	定義数
最初の文字がアットマーク（@）の場合は，属性です。			
（凡例） -：該当なし			

例

```
<cb:SendCBDefinition class="jp.co.Hitachi.soft.sdp.adaptor.callback.sendreceive.SendConnecto
rCBImpl" name="tcpDataSender">
 <scon:SendConnectorDefinition>
 <scon:streamInputs>
 <scon:streamInput>
 <scon:record name="PACKET_DATA" />
 <scon:stream name="packetData" querygroup="PacketAnalysis" />
 </scon:streamInput>
 </scon:streamInputs>
 </scon:SendConnectorDefinition>
</cb:SendCBDefinition>
```

14.9.4　受信コネクター定義

形式

```
 <ReceiveConnectorDefinition>
  <streamOutputs>
  <streamOutput>
  <stream name="ストリーム名" querygroup="クエリグループ名" />
  <record name="レコード名">
  <column name="列名" type="列のデータ型" />
  :
  </record>
  </streamOutput>
  :
  </streamOutputs>
 </ReceiveConnectorDefinition>
```

説明

表 14-19　要素と属性の一覧

要素または属性※			説明	値	定義数
ReceiveConnectorDefinition			受信コネクターを定義します。	-	1
	streamOutputs		出力ストリーム群に関する情報を定義します。	-	1
		streamOutput	出力ストリームに関する情報を定義します。	-	1～1024

要素または属性※	説明	値	定義数
stream	受信元の出力ストリームに関する情報を定義します。	指定したストリーム名とクエリグループ名のペアは，streams タグ内で一意である必要があります。	1
@name	出力ストリームの名前を指定します。	1～100 の半角英数字（アンダーライン（_）を含む）で指定します。最初の文字は半角の英字である必要があります。	1
@querygroup	クエリグループの名前を指定します。	1～64 の半角英数字（アンダーライン（_）を含む）で指定します。最初の文字は半角の英字である必要があります。	1
record	レコードに関する情報を定義します。	-	1
column	レコード内の各列に関する情報を定義します。なお，製品に同梱されているアダプター構成定義ファイルのサンプルファイルで${hsdp_adp_outputSchema}パラメーターを利用している場合は，アダプター起動時に，対応する出力ストリームのスキーマが自動的に読み取られ，スキーマに応じたcolumn 要素が${hsdp_adp_outputSchema}パラメーターの個所に自動的に挿入されます。このとき，挿入されたcolumn 要素のname 属性に指定される列名は，常に大文字に変換されます。	-	1～3000
@name	列の名前を指定します。	1～100 の半角英数字（アンダーライン（_）を含む）で指定します。最初の文字は半角の英字である必要があります。指定した列名は，record 要素内で一意である必要があります。	1
@type	列のデータ型を指定します。	次の型を指定できます。 INT | SHORT | BYTE | LONG | BIG_DECIMAL | FLOAT| DOUBLE| STRING| DATE | TIME | TIMESTAMP	1
@name	レコードの名前を指定します。	1～100 の半角英数字（アンダーライン（_）を含む）で指定します。最初の文字は半角の英字である必要があります。このレコード名は，対応する標準提供アダプター内で一意である必要があります。	1

注※
最初の文字がアットマーク（@）の場合は，属性です。

（凡例）

要素または属性※	説明	値	定義数
-：該当なし			

例

```
 <cb:ReceiveCBDefinition class="jp.co.Hitachi.soft.sdp.adaptor.callback.sendreceive.ReceiveC
onnectorCBImpl" name="eventReceiver">
  <rcon:ReceiveConnectorDefinition>
  <rcon:streamOutputs>
  <rcon:streamOutput>
  <rcon:stream name="eventSmtp" querygroup="PacketAnalysis" />
  <rcon:record name="EVENT_DATA">
  <rcon:column name="id" type="STRING" />
  <rcon:column name="ts" type="TIMESTAMP" />
  <rcon:column name="metricId" type="STRING" />
  <rcon:column name="metricName" type="STRING" />
  <rcon:column name="type" type="STRING" />
  <rcon:column name="event" type="INT" />
  <rcon:column name="info_1" type="STRING" />
  <rcon:column name="info_2" type="STRING" />
  <rcon:column name="level" type="STRING" />
  </rcon:record>
  </rcon:streamOutput>
  </rcon:streamOutputs>
  </rcon:ReceiveConnectorDefinition>
 </cb:ReceiveCBDefinition>
```

14.10　自動生成アダプターテンプレートファイル

記述形式

TCP データ入力アダプターの定義については，「14.7.5　TCP データ入力コネクター定義」を参照してください。

カスケーディングアダプターの定義については，「14.7.6　カスケーディングクライアントコネクター定義」を参照してください。

ファイル名

- tcpinput.xml

  入力ストリームにデータを入力する TCP データ入力アダプターの定義

- cascading_out.xml

  外部出力アダプターにデータを出力するカスケーディングアダプターの定義

- cascading_link.xml

  クエリグループ用プロパティファイルで指定したストリームにデータを出力するカスケーディングアダプターの定義

ファイルの格納先

運用ディレクトリ/inadaptor/conf/xml

説明

製品内部で自動起動される内部アダプターに関連する定義を，このファイルで指定します。

注意事項

通常はこのファイルを編集する必要はありません。アダプターに関連するチューニング要素を変更したい場合にだけ編集してください。

例

特になし。

14.11　アダプター構成定義ファイルの記述例

ここでは，アダプター構成定義ファイルの記述例を示します。

14.11.1　記述例 1

ここでは，次のサンプルファイルに記述されているアダプター構成定義ファイルの記述例を示したあとに，記述例の内容を説明します。

アダプター構成定義ファイルの記述例

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- All Rights Reserved. Copyright (C) 2010, Hitachi, Ltd.  -->
<root:AdaptorCompositionDefinition
 xmlns:root="http://www.hitachi.co.jp/soft/xml/sdp/adaptor/definition"
 xmlns:cmn="http://www.hitachi.co.jp/soft/xml/sdp/adaptor/definition/common"
 xmlns:adp="http://www.hitachi.co.jp/soft/xml/sdp/adaptor/definition/adaptor"
 xmlns:cb="http://www.hitachi.co.jp/soft/xml/sdp/adaptor/definition/callback"
 xmlns:ficon="http://www.hitachi.co.jp/soft/xml/sdp/adaptor/definition/callback/FileInputCon
nectorDefinition"
 xmlns:focon="http://www.hitachi.co.jp/soft/xml/sdp/adaptor/definition/callback/FileOutputCo
nnectorDefinition"
 xmlns:form="http://www.hitachi.co.jp/soft/xml/sdp/adaptor/definition/callback/FormatDefinit
ion"
 xmlns:map="http://www.hitachi.co.jp/soft/xml/sdp/adaptor/definition/callback/MappingDefinit
ion">

 <!-- 共通定義 -->
 <cmn:CommonDefinition>
  <!-- アダプタートレース定義 -->
  <cmn:AdaptorTraceDefinition trace="OFF"/>
 </cmn:CommonDefinition>

 <!-- ↓↓↓↓↓↓↓↓ インプロセスグループ定義 ↓↓↓↓↓↓↓↓ -->
 <!-- インプロセスグループ定義1 -->
 <adp:InprocessGroupDefinition name="InprocessAPTest">
  <!-- ↓↓↓↓↓↓ 入力アダプター定義 複数定義可能 ↓↓↓↓↓↓ -->
  <!-- 入力アダプター定義1 -->
  <adp:InputAdaptorDefinition name="InputAdaptor1"
   interval="0" charCode="MS932" lineFeed="CR_LF">

   <!-- 入力用CB定義 -->
   <cb:InputCBDefinition class="jp.co.Hitachi.soft.sdp.adaptor.callback.io.FileInputCBImpl"
name="inputer">
    <!-- 入力コネクター定義 -->
    <ficon:FileInputConnectorDefinition>
     <ficon:input readType="REAL_TIME" interval="1000" retryCount="100"
     readUnit="1"/>
     <ficon:file path="/var/tmp/input/Inprocess/"
     name="Inprocess_Data1.csv" messageLog="OFF"/>
    </ficon:FileInputConnectorDefinition>
   </cb:InputCBDefinition>
```

```
   <!-- データ編集用CB定義 -->
   <cb:DataEditCBDefinition class="jp.co.Hitachi.soft.sdp.adaptor.callback.dataedit.formattr
anslate.InputFormatTranslatorCBImpl" name="editor1">
    <!-- フォーマット変換定義 -->
    <form:FormatDefinition ioType="INPUT">
     <form:common>
      <form:unmatchedFormat>ERROR</form:unmatchedFormat>
     </form:common>
     <form:records>
      <form:record name="RECORD0" exp="($_name),($_num)">
       <form:field name="name" type="STRING"/>
       <form:field name="num" type="LONG"/>
      </form:record>
     </form:records>
    </form:FormatDefinition>
   </cb:DataEditCBDefinition>

   <!-- データ編集用CB定義 -->
   <cb:DataEditCBDefinition class="jp.co.Hitachi.soft.sdp.adaptor.callback.dataedit.mapping.
InputMappingCBImpl" name="editor3">
    <!-- マッピング定義 -->
    <map:MappingDefinition ioType="INPUT">
     <map:source/>
     <map:target>
      <map:streams>
       <map:stream name="DATA0" querygroup="Inprocess_QueryGroupTest">
        <map:column name="name" type="STRING"/>
        <map:column name="num" type="LONG"/>
       </map:stream>
      </map:streams>
     </map:target>
     <map:intermediate>
      <map:mappings source="RECORD0" querygroup="Inprocess_QueryGroupTest"
      target="DATA0">
       <map:map source="name" target="name"/>
       <map:map source="num" target="num"/>
      </map:mappings>
     </map:intermediate>
    </map:MappingDefinition>
   </cb:DataEditCBDefinition>

   <!-- 送信用CB定義 -->
   <cb:SendCBDefinition class="jp.co.Hitachi.soft.sdp.adaptor.callback.sendreceive.SendCBImp
l" name="sender">
    <cb:streamInfo name="DATA0" querygroup="Inprocess_QueryGroupTest"/>
   </cb:SendCBDefinition>

  </adp:InputAdaptorDefinition>
  <!-- ↑↑↑↑↑↑ 入力アダプター定義 複数定義可能 ↑↑↑↑↑↑ -->

  <!-- ↓↓↓↓↓↓ 出力アダプター定義 複数定義可能 ↓↓↓↓↓↓ -->
  <adp:OutputAdaptorDefinition name="OutputAdaptor1"
   charCode="MS932" lineFeed="CR_LF">

   <!-- 受信用CB定義 -->
   <cb:ReceiveCBDefinition class="jp.co.Hitachi.soft.sdp.adaptor.callback.sendreceive.Receiv
eCBImpl" name="receiver">
```

```
    <cb:streamInfo name="FILTER1" querygroup="Inprocess_QueryGroupTest"/>
   </cb:ReceiveCBDefinition>

   <!-- データ編集用CB定義 -->
   <cb:DataEditCBDefinition class="jp.co.Hitachi.soft.sdp.adaptor.callback.dataedit.mapping.
OutputMappingCBImpl" name="editor1">
    <!-- マッピング定義 -->
    <map:MappingDefinition ioType="OUTPUT">
     <map:source>
      <map:streams>
       <map:stream name="FILTER1" querygroup="Inprocess_QueryGroupTest">
        <map:column name="name" type="STRING"/>
        <map:column name="num" type="LONG"/>
       </map:stream>
      </map:streams>
     </map:source>
     <map:target/>
     <map:intermediate>
      <map:mappings source="FILTER1" querygroup="Inprocess_QueryGroupTest"
      target="RECORD1">
       <map:map source="name" target="name"/>
       <map:map source="num" target="num"/>
      </map:mappings>
     </map:intermediate>
    </map:MappingDefinition>
   </cb:DataEditCBDefinition>

   <!-- データ編集用CB定義 -->
   <cb:DataEditCBDefinition class="jp.co.Hitachi.soft.sdp.adaptor.callback.dataedit.formattr
anslate.OutputFormatTranslatorCBImpl" name="editor3">
    <!-- フォーマット変換定義 -->
    <form:FormatDefinition ioType="OUTPUT">
     <form:common/>
     <form:records>
      <form:record name="RECORD1" exp="($_name),($_num)">
       <form:field name="name" type="STRING"/>
       <form:field name="num" type="LONG"/>
      </form:record>
     </form:records>
    </form:FormatDefinition>
   </cb:DataEditCBDefinition>

   <!-- 出力用CB定義 -->
   <cb:OutputCBDefinition class="jp.co.Hitachi.soft.sdp.adaptor.callback.io.FileOutputCBImpl
" name="outputer">
    <!-- 出力コネクター定義 -->
    <focon:FileOutputConnectorDefinition>
     <focon:output compositionType="WRAP_AROUND" maxNumber="256"
     maxSize="1000"/>
     <focon:file path="/var/tmp/output/Inprocess/" prefix="out"
     addDate="OFF" extension="csv"/>
    </focon:FileOutputConnectorDefinition>
   </cb:OutputCBDefinition>
  </adp:OutputAdaptorDefinition>
  <!-- ↑↑↑↑↑↑↑ 出力アダプター定義 複数定義可能 ↑↑↑↑↑↑↑ -->

 </adp:InprocessGroupDefinition>
 <!-- ↑↑↑↑↑↑↑↑↑ インプロセスグループ定義 ↑↑↑↑↑↑↑↑↑ -->
```

```
</root:AdaptorCompositionDefinition>
```

記述例の内容

この記述例では，標準提供アダプターと SDP サーバは，インプロセスで連携します。入力アダプター「InputAdaptor1」でファイルを入力し，ストリームデータの集計・分析結果のデータは，出力アダプター「OutputAdaptor1」でファイルへ出力します。

入力アダプター「InputAdaptor1」では，次の表に示す処理を実施します。なお，括弧内は，アダプター構成定義ファイルでの定義名です。

コールバックの種類	コールバックでの処理
入力用コールバック（入力用 CB 定義）	ファイルの入力（ファイル入力コネクター定義）
編集用コールバック（編集用 CB 定義）	フォーマット変換（フォーマット変換定義）
	レコードとストリーム間のマッピング（マッピング定義）
送信用コールバック（送信用 CB 定義）	タプル送信（入力ストリーム定義）

入力アダプター「InputAdaptor1」の各定義の内容を次に示します。

- ファイル入力コネクター定義の内容
  入力データの格納先：/var/tmp/input/Inprocess/
  ファイル名：Inprocess_Data1.csv
  読み込み処理モード：リアルタイム処理モード
  読み込み単位：一度に読み込むレコード数は 1 行
- フォーマット変換定義の内容
  ファイル入力コネクターで出力された入力形式レコードを共通形式レコードに変換します。
- マッピング定義の内容
  レコードとストリーム間のマッピングで，フォーマット変換で出力された共通形式レコードと，入力ストリームの形式に対応した共通形式レコードとを関連づけます。
- 入力ストリーム定義の内容
  マッピングで出力された共通形式レコードをタプルに変換して，クエリグループ「Inprocess_QueryGroupTest」の入力ストリーム「DATA0」に送信します。

出力アダプター「OutputAdaptor1」では，次の表に示す処理を実施します。なお，括弧内は，アダプター構成定義ファイルでの定義名です。

コールバックの種類	コールバックでの処理
受信用コールバック（受信用 CB 定義）	タプル受信（出力ストリーム定義）
編集用コールバック（編集用 CB 定義）	レコードとストリーム間のマッピング（マッピング定義）

コールバックの種類	コールバックでの処理
編集用コールバック（編集用 CB 定義）	フォーマット変換（フォーマット変換定義）
出力用コールバック（出力用 CB 定義）	ファイルへの出力（ファイル出力コネクター定義）

出力アダプター「OutputAdaptor1」の各定義での定義内容を次に示します。

- 出力ストリーム定義の内容

  クエリグループ「Inprocess_QueryGroupTest」の出力ストリーム「FILTER1」からタプルを受信して，共通形式レコードに変換します。

- マッピング定義の内容

  レコードとストリーム間のマッピングで，出力ストリームの形式に対応した共通形式レコードと，フォーマット変換で入力する共通形式レコードとを関連づけます。

- フォーマット変換定義の内容

  マッピングで出力された共通形式レコードを出力形式レコードに変換します。

- ファイル出力コネクター定義の内容

  フォーマット変換で出力された出力形式レコードを次の条件でファイルに出力します。

  ファイル構成：ラップアラウンド構成

  出力データの格納先：/var/tmp/output/Inprocess/

  ファイル名および出力順：out1.csv，out2.csv，…，out256.csv

- アダプター構成定義ファイルで指定するストリーム名

  アダプター構成定義ファイルで指定するストリーム名は，クエリ定義ファイルで指定した内容と合わせてください。

  入力ストリームの場合

  クエリ定義ファイルのregister stream 句で指定したストリーム名「DATA0」が，アダプター構成定義ファイルで指定する入力ストリーム名となります。

  出力ストリームの場合

  クエリ定義ファイルのregister query 句で指定したクエリ名「FILTER1」が，アダプター構成定義ファイルで指定する出力ストリーム名となります。

この記述例の場合のクエリ定義ファイルを次に示します。

```
register stream DATA0(name VARCHAR(10), num BIGINT);

register query FILTER1 ISTREAM(SELECT * FROM DATA0[ROWS 1] WHERE DATA0.NUM <= 24);
```

クエリの定義については，マニュアル『Hitachi Streaming Data Platform アプリケーション開発ガイド』を参照してください。

14.11.2　記述例 2

ここでは，次のサンプルファイルに記述されているアダプター構成定義ファイルの記述例を示したあとに，記述例の内容を説明します。

アダプター構成定義ファイルの記述例

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- All Rights Reserved. Copyright (C) 2010, Hitachi, Ltd.  -->
<root:AdaptorCompositionDefinition
 xmlns:root="http://www.hitachi.co.jp/soft/xml/sdp/adaptor/definition"
 xmlns:cmn="http://www.hitachi.co.jp/soft/xml/sdp/adaptor/definition/common"
 xmlns:adp="http://www.hitachi.co.jp/soft/xml/sdp/adaptor/definition/adaptor"
 xmlns:cb="http://www.hitachi.co.jp/soft/xml/sdp/adaptor/definition/callback"
 xmlns:hpicon="http://www.hitachi.co.jp/soft/xml/sdp/adaptor/definition/callback/HttpPacketI
nputConnectorDefinition"
 xmlns:map="http://www.hitachi.co.jp/soft/xml/sdp/adaptor/definition/callback/MappingDefinit
ion"
 xmlns:filter="http://www.hitachi.co.jp/soft/xml/sdp/adaptor/definition/callback/FilterDefin
ition"
 xmlns:rex="http://www.hitachi.co.jp/soft/xml/sdp/adaptor/definition/callback/RecordExtracti
onDefinition"
 xmlns:docon="http://www.hitachi.co.jp/soft/xml/sdp/adaptor/definition/callback/DashboardOut
putConnectorDefinition"
 >

 <!-- 共通定義 -->
 <cmn:CommonDefinition>
  <!-- アダプタートレース定義 -->
  <cmn:AdaptorTraceDefinition trace="OFF"/>
 </cmn:CommonDefinition>

 <!-- ↓↓↓↓↓↓↓↓ インプロセスグループ定義 ↓↓↓↓↓↓↓↓ -->
 <!-- インプロセスグループ定義1 -->
 <adp:InprocessGroupDefinition name="InprocessAPTest">
  <!-- ↓↓↓↓↓↓ 入力アダプター定義 複数定義可能 ↓↓↓↓↓↓ -->
  <!-- 入力アダプター定義1 -->
  <adp:InputAdaptorDefinition name="InputAdaptor1" interval="0"
   charCode="MS932" lineFeed="CR_LF">

   <!-- 入力用CB定義 -->
   <cb:InputCBDefinition class="jp.co.Hitachi.soft.sdp.adaptor.callback.io.packetinput.HttpP
acketInputCBImpl" name="inputer">
    <!-- パケット入力コネクター定義 -->
    <hpicon:HttpPacketInputConnectorDefinition>
     <hpicon:input buffersize="4096" assemblingtime="2000">
      <hpicon:packetdata globalheader="24" packetheader="16" packetoffset="8"
       packetlength="4" timeoffset="0"/>
      <hpicon:command path="/usr/sbin/tcpdump" parameter=" -i eth0 -s 65536 -w - -n tcp port
 80 or port 8080"/>
     </hpicon:input >
     <hpicon:output unit="100">
      <hpicon:record name="REQUEST" type="REQUEST" >
       <hpicon:field name="SEND_IP"/>
       <hpicon:field name="RECEIVE_IP"/>
       <hpicon:field name="SEND_PORT"/>
```

```
       <hpicon:field name="RECEIVE_PORT"/>
       <hpicon:field name="TARGET_URI"/>
       <hpicon:field name="TIME"/>
      </hpicon:record >
      <hpicon:record name="RESPONSE" type="RESPONSE" >
       <hpicon:field name="SEND_IP"/>
       <hpicon:field name="RECEIVE_IP"/>
       <hpicon:field name="SEND_PORT"/>
       <hpicon:field name="RECEIVE_PORT"/>
       <hpicon:field name="TIME"/>
      </hpicon:record >
     </hpicon:output>
    </hpicon:HttpPacketInputConnectorDefinition>
   </cb:InputCBDefinition>

   <!-- データ編集用CB定義 -->
   <cb:DataEditCBDefinition class="jp.co.Hitachi.soft.sdp.adaptor.callback.dataedit.filter.F
ilterCBImpl" name="editor1">
    <!-- フィルター定義 -->
    <filter:FilterDefinition>
     <!-- レコード条件 -->
     <filter:record source="REQUEST" conditionName="filterName1"
      condition="AND">
      <!-- フィールド条件 -->
      <filter:field source="TARGET_URI" condition="eq"
       value="http:¥/¥/www.*"/>
     </filter:record>
    </filter:FilterDefinition>
   </cb:DataEditCBDefinition>

   <!-- データ編集用CB定義 -->
   <cb:DataEditCBDefinition class="jp.co.Hitachi.soft.sdp.adaptor.callback.dataedit.recordex
tract.RecordExtractionCBImpl" name="editor2">

    <!-- レコード抽出定義 -->
    <rex:RecordExtractionDefinition>
     <!-- 抽出対象レコード群定義 -->
     <rex:targetrecords>
      <rex:targetrecord name="element1">
       <rex:record source="REQUEST" timeposition="TIME" condition="AND">
        <rex:field source="SEND_PORT" condition="ne" value="0"/>
       </rex:record>
      </rex:targetrecord>
      <rex:targetrecord name="element2">
       <rex:record source="RESPONSE" timeposition="TIME" condition="AND">
        <rex:field source="RECEIVE_PORT" condition="ne" value="0"/>
       </rex:record>
      </rex:targetrecord>
     </rex:targetrecords>
     <!-- 抽出条件群定義 -->
     <rex:extractions size="100000" timeout="OFF" samerecord="overwrite">
      <rex:extraction name="BIND_PACKET" timelimit="10000">
       <rex:targets>
        <rex:target sourceL="element1" sourceR="element2" condition="AND">
         <rex:fieldcondition sourceL="SEND_IP"      condition="eq"
          sourceR="RECEIVE_IP"/>
         <rex:fieldcondition sourceL="RECEIVE_IP"   condition="eq"
          sourceR="SEND_IP"/>
```

```
        <rex:fieldcondition sourceL="SEND_PORT"    condition="eq"
         sourceR="RECEIVE_PORT"/>
        <rex:fieldcondition sourceL="RECEIVE_PORT" condition="eq"
         sourceR="SEND_PORT"/>
       </rex:target>
      </rex:targets>
      <!-- 抽出レコード定義 -->
      <rex:extractrecord name="BINDRECORD">
       <rex:select source="element1"/>
       <rex:select source="element2"/>
      </rex:extractrecord>
     </rex:extraction>
    </rex:extractions>
   </rex:RecordExtractionDefinition>
  </cb:DataEditCBDefinition>

  <!-- データ編集用CB定義 -->
  <cb:DataEditCBDefinition class="jp.co.Hitachi.soft.sdp.adaptor.callback.dataedit.mapping.
InputMappingCBImpl" name="editor3">
   <!-- マッピング定義 -->
   <map:MappingDefinition ioType="INPUT">
    <map:source/>
    <map:target>
     <map:records>
      <map:record name="RESULT">
       <map:field name="SEND_IP"      type="STRING"/>
       <map:field name="RECEIVE_IP"   type="STRING"/>
       <map:field name="SEND_PORT"    type="INT"/>
       <map:field name="RECEIVE_PORT" type="INT"/>
       <map:field name="URI"          type="STRING"/>
       <map:field name="SUBTIME"      type="LONG"/>
       <map:field name="TIME"         type="TIMESTAMP"/>
      </map:record>
     </map:records>
    </map:target>
    <map:intermediate>
     <map:mappings source="BINDRECORD" target="RESULT">
      <map:map source="element1_SEND_IP"      target="SEND_IP"/>
      <map:map source="element1_RECEIVE_IP"   target="RECEIVE_IP"/>
      <map:map source="element1_SEND_PORT"    target="SEND_PORT"/>
      <map:map source="element1_RECEIVE_PORT" target="RECEIVE_PORT"/>
      <map:map source="element1_TARGET_URI"   target="URI"/>
      <map:map function="subTime" argument1="element1_TIME"
       argument2="element2_TIME" target="SUBTIME"/>
      <map:map source="element2_TIME"         target="TIME"/>
     </map:mappings>
    </map:intermediate>
   </map:MappingDefinition>
  </cb:DataEditCBDefinition>

  <!-- データ編集用CB定義 -->
  <cb:DataEditCBDefinition class="jp.co.Hitachi.soft.sdp.adaptor.callback.dataedit.mapping.
InputMappingCBImpl" name="editor4">
   <!-- マッピング定義 -->
   <map:MappingDefinition ioType="INPUT">
    <map:source/>
    <map:target>
     <map:streams>
```

```
       <map:stream name="s1" querygroup="Inprocess_QueryGroupTest">
        <map:column name="sendip"      type="STRING"/>
        <map:column name="receiveip"   type="STRING"/>
        <map:column name="sendport"    type="INT"/>
        <map:column name="receiveport" type="INT"/>
        <map:column name="uri"         type="STRING"/>
        <map:column name="subtime"     type="LONG"/>
        <map:column name="time"        type="TIMESTAMP"/>
       </map:stream>
      </map:streams>
     </map:target>
     <map:intermediate>
      <map:mappings source="RESULT" querygroup="Inprocess_QueryGroupTest" target="s1">
       <map:map source="SEND_IP"      target="sendip"/>
       <map:map source="RECEIVE_IP"   target="receiveip"/>
       <map:map source="SEND_PORT"    target="sendport"/>
       <map:map source="RECEIVE_PORT" target="receiveport"/>
       <map:map source="URI"          target="uri"/>
       <map:map source="SUBTIME"      target="subtime"/>
       <map:map source="TIME"         target="time"/>
      </map:mappings>
     </map:intermediate>
    </map:MappingDefinition>
   </cb:DataEditCBDefinition>

   <!-- 送信用CB定義 -->
   <cb:SendCBDefinition class="jp.co.Hitachi.soft.sdp.adaptor.callback.sendreceive.SendCBImp
l" name="sender">
    <cb:streamInfo name="s1" querygroup="Inprocess_QueryGroupTest"/>
   </cb:SendCBDefinition>
  </adp:InputAdaptorDefinition>
  <!-- ↑↑↑↑↑↑ 入力アダプター定義 複数定義可能 ↑↑↑↑↑↑ -->

  <!-- ↓↓↓↓↓↓ 出力アダプター定義 複数定義可能 ↓↓↓↓↓↓ -->
  <adp:OutputAdaptorDefinition name="OutputAdaptor1" charCode="MS932"
   lineFeed="CR_LF">

   <!-- 受信用CB定義 -->
   <cb:ReceiveCBDefinition class="jp.co.Hitachi.soft.sdp.adaptor.callback.sendreceive.Receiv
eCBImpl" name="receiver">
    <cb:streamInfo name="q1" querygroup="Inprocess_QueryGroupTest"/>
   </cb:ReceiveCBDefinition>

   <!-- データ編集用CB定義 -->
   <cb:DataEditCBDefinition class="jp.co.Hitachi.soft.sdp.adaptor.callback.dataedit.mapping.
OutputMappingCBImpl" name="editor1">
    <!-- マッピング定義 -->
    <map:MappingDefinition ioType="OUTPUT">
     <map:source>
      <map:streams>
       <map:stream name="q1" querygroup="Inprocess_QueryGroupTest">
        <map:column name="sendip"      type="STRING"/>
        <map:column name="receiveip"   type="STRING"/>
        <map:column name="sendport"    type="INT"/>
        <map:column name="receiveport" type="INT"/>
        <map:column name="uri"         type="STRING"/>
        <map:column name="subtime"     type="LONG"/>
        <map:column name="time"        type="TIMESTAMP"/>
```

```
      </map:stream>
     </map:streams>
    </map:source>
    <map:target/>
    <map:intermediate>
     <map:mappings source="q1" querygroup="Inprocess_QueryGroupTest"
      target="RECORD1">
      <map:map source="sendip"      target="SEND_IP"/>
      <map:map source="receiveip"   target="RECEIVE_IP"/>
      <map:map source="sendport"    target="SEND_PORT"/>
      <map:map source="receiveport" target="RECEIVE_PORT"/>
      <map:map source="uri"         target="URI"/>
      <map:map source="subtime"     target="SUBTIME"/>
      <map:map source="time"        target="TIME"/>
     </map:mappings>
    </map:intermediate>
   </map:MappingDefinition>
  </cb:DataEditCBDefinition>

  <!-- データ編集用CB定義 -->
  <cb:DataEditCBDefinition class="jp.co.Hitachi.soft.sdp.adaptor.callback.dataedit.mapping.
InputMappingCBImpl" name="editor2">
   <!-- マッピング定義 -->
   <map:MappingDefinition ioType="OUTPUT">
    <map:source/>
    <map:target>
     <map:records>
      <map:record name="RECORD2" >
       <map:field name="SEND_IP"         type="STRING"/>
       <map:field name="RECEIVE_IP"      type="STRING"/>
       <map:field name="SEND_PORT"       type="INT"/>
       <map:field name="RECEIVE_PORT"    type="INT"/>
       <map:field name="URI"             type="STRING"/>
       <map:field name="SUBTIME"         type="LONG"/>
       <map:field name="TIME"            type="TIMESTAMP"/>
       <map:field name="GET_TUPLE_TIME" type="TIMESTAMP"/>
      </map:record>
     </map:records>
    </map:target>
    <map:intermediate>
     <map:mappings source="RECORD1" target="RECORD2">
      <map:map source="SEND_IP"         target="SEND_IP"/>
      <map:map source="RECEIVE_IP"      target="RECEIVE_IP"/>
      <map:map source="SEND_PORT"       target="SEND_PORT"/>
      <map:map source="RECEIVE_PORT"    target="RECEIVE_PORT"/>
      <map:map source="URI"             target="URI"/>
      <map:map source="SUBTIME"         target="SUBTIME"/>
      <map:map source="TIME"            target="TIME"/>
      <map:map function="getTupleTime" target="GET_TUPLE_TIME"/>
     </map:mappings>
    </map:intermediate>
   </map:MappingDefinition>
  </cb:DataEditCBDefinition>

  <!-- 出力用CB定義 -->
  <cb:OutputCBDefinition class="jp.co.Hitachi.soft.sdp.adaptor.callback.io.dashboard.Dashbo
ardOutputCBImpl" name="outputer">
   <!-- ダッシュボード出力コネクター定義 -->
```

```
    <docon:DashboardOutputConnectorDefinition Record="RECORD2">
     <docon:RecordHoldTime
      DateReference="LAST_UPDATE"
                    RecordTime="300"
                    DateFieldPosition="8" />
    </docon:DashboardOutputConnectorDefinition>
   </cb:OutputCBDefinition>
  </adp:OutputAdaptorDefinition>
  <!-- ↑↑↑↑↑↑↑ 出力アダプター定義 複数定義可能 ↑↑↑↑↑↑↑ -->

 </adp:InprocessGroupDefinition>
 <!-- ↑↑↑↑↑↑↑↑↑ インプロセスグループ定義 ↑↑↑↑↑↑↑↑↑ -->

</root:AdaptorCompositionDefinition>
```

記述例の内容

この記述例では，標準提供アダプターと SDP サーバは，インプロセスで連携します。入力アダプター「InputAdaptor1」では，HTTP パケット情報の中から対応するリクエスト情報とレスポンス情報を結合し，それぞれが持っていた時刻情報の差から通信時間を算出します。出力アダプター「OutputAdaptor1」では，HTTP 通信情報を取得し，現在時刻から過去 5 分間の情報をダッシュボード出力コネクターで出力します。

入力アダプター「InputAdaptor1」では，次の表に示す処理を実施します。なお，括弧内は，アダプター構成定義ファイルでの定義名です。

コールバックの種類	コールバックでの処理
入力用コールバック（入力用 CB 定義）	HTTP パケットの入力（HTTP パケット入力コネクター定義）
編集用コールバック（編集用 CB 定義）	レコードのフィルタリング（フィルター定義）
	レコードの抽出（レコード抽出定義）
	レコード間のマッピング（マッピング定義）
	レコードとストリーム間のマッピング（マッピング定義）
送信用コールバック（送信用 CB 定義）	タプル送信（入力ストリーム定義）

入力アダプター「InputAdaptor1」の各定義の内容を次に示します。

- HTTP パケット入力コネクター定義の内容

  パケットアナライザーとして，tcpdump を使用します。コマンドパラメーターには，解析対象コンピュータ（以降，サーバとします）の次の情報を指定します。

  LAN interface name：eth0

  HTTP プロトコルに用いるポート番号：80 または8080

  HTTP パケット入力コネクターで，サーバが受信したパケット（リクエストパケット）と送信したパケット（レスポンスパケット）をレコード化します。各レコードでは，次のフィールドを保持します。

  リクエストレコード

送信元 IP アドレス，送信先 IP アドレス，送信元ポート番号，送信先ポート番号，URI 情報，およびサーバがパケットを受信した時刻

レスポンスレコード

送信元 IP アドレス，送信先 IP アドレス，送信元ポート番号，送信先ポート番号，およびサーバがパケットを送信した時刻

- フィルター定義の内容

  リクエストレコードのうち，URI が「http://www.*」のレコードだけを取得します。なお，*は任意の 0 文字以上の文字列です。

- レコード抽出定義の内容

  次の条件に一致するリクエストレコードとレスポンスレコードを結合した抽出レコードを生成します。

  リクエストレコードの送信元 IP アドレスが，レスポンスレコードの送信先 IP アドレスと等しい。

  リクエストレコードの送信先 IP アドレスが，レスポンスレコードの送信元 IP アドレスと等しい。

  リクエストレコードの送信元ポート番号が，レスポンスレコードの送信先ポート番号と等しい。

  リクエストレコードの送信先ポート番号が，レスポンスレコードの送信元ポート番号と等しい。

- マッピング定義の内容

  1 個目のマッピング（レコード間のマッピング）で，レコードの抽出で結合されたレコードから，次のフィールドを保持するレコードを生成します。

  送信元 IP アドレス，送信先 IP アドレス，送信元ポート番号，送信先ポート番号，URI 情報（パラメーターを除く），通信応答時間（サーバがパケットを受信，および送信した時刻の差），およびサーバがパケットを送信した時刻

  なお，通信応答時間は，function 属性でsubTime を指定して取得します。

  2 個目のマッピング（レコードとストリーム間のマッピング）で，1 個目のマッピングで出力された共通形式レコードと，入力ストリームの形式に対応した共通形式レコードとを関連づけます。

- 入力ストリーム定義の内容

  マッピングで出力された共通形式レコードをタプルに変換して，クエリグループ「Inprocess_QueryGroupTest」の入力ストリーム「s1」に送信します。

出力アダプター「OutputAdaptor1」では，次の表に示す処理を実施します。なお，括弧内は，アダプター構成定義ファイルでの定義名です。

コールバックの種類	コールバックでの処理
受信用コールバック（受信用 CB 定義）	タプル受信（出力ストリーム定義）
編集用コールバック（編集用 CB 定義）	レコードとストリーム間のマッピング（マッピング定義）
	レコード間のマッピング（マッピング定義）
出力用コールバック（出力用 CB 定義）	ダッシュボードへの出力（ダッシュボード出力コネクター定義）

出力アダプター「OutputAdaptor1」の各定義の内容を次に示します。

- 出力ストリーム定義の内容

クエリグループ「Inprocess_QueryGroupTest」の出力ストリーム「q1」からタプルを受信して，共通形式レコードに変換します。

- マッピング定義の内容

  1 個目のマッピング（レコードとストリーム間のマッピング）で，出力ストリームの形式に対応した共通形式レコードと，次のコールバックの形式に対応した共通形式レコードとを関連づけます。

  2 個目のマッピング（レコード間のマッピング）で，1 個目のマッピングで出力されたレコードから，次のフィールドを保持するレコードを生成します。

  送信元 IP アドレス，送信先 IP アドレス，送信元ポート番号，送信先ポート番号，URI 情報（パラメーターを除く），通信応答時間（サーバがパケットを受信，および送信した時刻の差），サーバがパケットを送信した時刻，および出力ストリームから出力されたタプルの時刻

  なお，出力ストリームから出力されたタプルの時刻は，function 属性でgetTupleTime を指定して取得します。

- ダッシュボード出力コネクター定義の内容

  マッピングで出力されたレコードを次の条件でダッシュボードに出力します。

  ダッシュボード出力コネクターが最後にタプルを受信した時刻を基準時刻とする。

  基準時刻から 5 分（300 秒）以内のレコードを保持する。

  タプルの時刻情報は，8 番目のフィールド（GET_TUPLE_TIME）にある。

  RMI サーバのポート番号は「20421」を使用する（デフォルトのため，定義では指定しない）。

  なお，ダッシュボードでデータを表示するときの接続名は「InprocessAPTest/OutputAdaptor1/outputer」となります。

- アダプター構成定義ファイルで指定するストリーム名

  アダプター構成定義ファイルで指定するストリーム名は，クエリ定義ファイルで指定した内容と合わせてください。

  入力ストリームの場合

  クエリ定義ファイルのregister stream 句で指定したストリーム名（s1）が，アダプター構成定義ファイルで指定する入力ストリーム名となります。

  出力ストリームの場合

  クエリ定義ファイルのregister query 句で指定したクエリ名（q1）が，アダプター構成定義ファイルで指定する出力ストリーム名となります。

  この記述例の場合のクエリ定義ファイルを次に示します。

```
REGISTER STREAM s1(sendip VARCHAR(15),receiveip VARCHAR(15),sendport INTEGER,receiveport
INTEGER,uri VARCHAR(255),subtime BIGINT,times TIMESTAMP(9));

REGISTER QUERY q1 RSTREAM(SELECT * FROM s1[RANGE 5 MINUTE]);
```

  クエリの定義については，マニュアル『Hitachi Streaming Data Platform アプリケーション開発ガイド』を参照してください。

14.11.3　出力アダプター定義（最新のデータの表示）

棒グラフ，円グラフ，散布図または表で，ストリームデータの集計・分析結果の最新のデータをダッシュボードに表示する場合，クエリ定義ファイルでのrstream の最新の計算結果だけを表示できるようにします。

ここでは，最新のデータを表示する場合の出力アダプター定義の記述例を示したあとに，記述例の内容を説明します。

記述例

```
<adp:OutputAdaptorDefinition name="OutputAdaptor1" charCode="MS932" lineFeed="CR_LF">

    <!-- 受信用CB定義 -->
    <cb:ReceiveCBDefinition class="jp.co.Hitachi.soft.sdp.adaptor.callback.sendreceive.Recei
veCBImpl" name="receiver">
        <cb:streamInfo name="QUERY" querygroup="Inprocess_QueryGroupTest"/>
    </cb:ReceiveCBDefinition>

    <!-- データ編集用CB定義 -->
    <cb:DataEditCBDefinition class="jp.co.Hitachi.soft.sdp.adaptor.callback.dataedit.mapping
.OutputMappingCBImpl" name="editor1">
        <!-- マッピング定義 -->
        <map:MappingDefinition ioType="OUTPUT">
            <map:source>
                <map:streams>
                    <map:stream name="QUERY" querygroup="Inprocess_QueryGroupTest">
                        <map:column name="sendip"      type="STRING"/>
                        <map:column name="subtime"     type="LONG"/>
                    </map:stream>
                </map:streams>
            </map:source>
            <map:target/>
            <map:intermediate>
                <map:mappings source="QUERY" querygroup="Inprocess_QueryGroupTest" target="R
ECORD1">
                    <map:map source="sendip"      target="SEND_IP"/>
                    <map:map source="subtime"     target="SUBTIME"/>
                </map:mappings>
            </map:intermediate>
        </map:MappingDefinition>
    </cb:DataEditCBDefinition>

    <!-- データ編集用CB定義 -->
    <cb:DataEditCBDefinition class="jp.co.Hitachi.soft.sdp.adaptor.callback.dataedit.mapping
.InputMappingCBImpl" name="editor2">
        <!-- マッピング定義 -->
        <map:MappingDefinition ioType="OUTPUT">
            <map:source/>
            <map:target>
                <map:records>
                    <map:record name="RECORD2" >
                        <map:field name="SEND_IP"        type="STRING"/>
                        <map:field name="SUBTIME"        type="LONG"/>
                        <map:field name="GET_TUPLE_TIME" type="TIMESTAMP"/>
                    </map:record>
```

```
            </map:records>
        </map:target>
        <map:intermediate>
            <map:mappings source="RECORD1" target="RECORD2">
                <map:map source="SEND_IP"        target="SEND_IP"/>
                <map:map source="SUBTIME"        target="SUBTIME"/>
                <map:map function="getTupleTime" target="GET_TUPLE_TIME"/>
            </map:mappings>
        </map:intermediate>
    </map:MappingDefinition>
</cb:DataEditCBDefinition>

<cb:OutputCBDefinition class="jp.co.Hitachi.soft.sdp.adaptor.callback.io.dashboard.DashboardOutputCBImpl" name="outputer">
    <!-- ダッシュボード出力コネクター定義 -->
    <docon:DashboardOutputConnectorDefinition Record="RECORD2">
        <docon:RecordHoldTime DateReference="LAST_UPDATE" RecordTime="0" DateFieldPosition="3" />
        <docon:DataProcessingDefinition Name="HistoryRecorder">
            <docon:HistoryRecorder/>
        </docon:DataProcessingDefinition>
    </docon:DashboardOutputConnectorDefinition>
</cb:OutputCBDefinition>

</adp:OutputAdaptorDefinition>
```

記述例の内容

この記述例では，ストリームデータの集計・分析結果のデータを出力アダプター「OutputAdaptor1」でダッシュボードへ出力します。

出力アダプター「OutputAdaptor1」では，次の表に示す処理を実施します。なお，括弧内は，アダプター構成定義ファイル内での定義名です。

コールバックの種類	コールバックでの処理
受信用コールバック（受信用 CB 定義）	タプル受信（出力ストリーム定義）
編集用コールバック（編集用 CB 定義）	レコードとストリーム間のマッピング（マッピング定義）
	レコード間のマッピング（マッピング定義）
	フォーマット変換（フォーマット変換定義）
送信用コールバック（送信用 CB 定義）	ダッシュボードへの出力（ダッシュボード出力コネクター定義）

出力アダプター「OutputAdaptor1」の各定義での定義内容を次に示します。

- 出力ストリーム定義の内容
  クエリグループ「Inprocess_QueryGroupTest」の出力ストリーム「QUERY」からタプルを受信して，共通形式レコードに変換します。
- マッピング定義の内容

レコードとストリーム間のマッピングで，出力ストリームの形式に対応した共通形式レコードと，フォーマット変換で入力する共通形式レコードとを関連づけます。

また，レコードの間のマッピングで，rstream の最新の結果がどのレコードかがわかるように，タプルの時刻を取得（map タグのfunction 属性にgetTupleTime を指定）して共通形式レコードに出力します。

- フォーマット変換定義の内容

  マッピングで出力された共通形式レコードを出力形式レコードに変換します。

- ダッシュボード出力コネクター定義の内容

  フォーマット変換で出力された出力形式レコードに対し，レコード保持期間（RecordHoldTime タグのRecordTime 属性）に0 を設定し，最新の結果以外のレコードをすべて削除するようにします。

- アダプター構成定義ファイルで指定するストリーム名

  アダプター構成定義ファイルで指定するストリーム名は，クエリ定義ファイルで指定した内容と合わせてください。

  入力ストリームの場合

  クエリ定義ファイルのregister stream 句で指定したストリーム名「STREAM」が，アダプター構成定義ファイルで指定する入力ストリーム名となります。

  出力ストリームの場合

  クエリ定義ファイルのregister query 句で指定したクエリ名「QUERY」が，アダプター構成定義ファイルで指定する出力ストリーム名となります。

  この記述例の場合のクエリ定義ファイルを次に示します。この定義ファイルでは，送信先 IP アドレス（sendip）ごとに，直近 1 分間の中での送受信時間（subtime）の最大値を求め，その結果をrstream で出力する処理を記述しています。

```
register stream STREAM(sendip VARCHAR(15),receiveip VARCHAR(15),sendport INT,receiveport
INT,uri VARCHAR(255),subtime BIGINT,times TIMESTAMP(9));

register query QUERY rstream(
  select STREAM.sendip as sendip, max(STREAM.subtime) as subtime
  from STREAM[RANGE 1 MINUTE]
  group by STREAM.sendip);
```

  クエリの定義については，マニュアル『Hitachi Streaming Data Platform アプリケーション開発ガイド』を参照してください。

14.11.4　出力アダプター定義（履歴を含むデータの表示）

折れ線グラフなどで，現時点までの履歴を含む結果をダッシュボードに表示する場合，現時点から過去n秒までのrstream の計算結果を表示できるようにします。

この場合，出力アダプター定義のダッシュボード出力コネクター定義では，レコード保持期間（RecordHoldTime タグのRecordTime 属性）にn を設定して，getTupleTime の結果を格納するフィールドに，現時点から過去n 秒までの結果を保持する設定をします。

これ以外の出力アダプターの記述例，および記述例の内容については，「14.11.3　出力アダプター定義（最新のデータの表示）」を参照してください。

15 外部定義関数定義ファイル

ここでは，外部定義関数を使用する場合に作成する，外部定義関数定義ファイルについて説明します。外部定義関数定義ファイルと，その説明に用いる形式は，アダプター構成定義ファイルの形式と同様です。

外部定義関数定義ファイルとその説明の形式については，「14.1　アダプター構成定義ファイルの説明の記述形式」および「14.2　アダプター構成定義ファイル作成上の注意事項」を参照してください。

15.1　外部定義関数定義ファイル

外部定義関数定義ファイルは，外部定義関数の関数名と戻り値を定義します。また，ファイルとユーザーが実装した外部定義関数のコード（Java のクラスまたは C 言語のライブラリ）を指定します。ユーザーが（Java で）作成したクラスとの関係も定義します。

ここでは，外部定義関数定義ファイルの概要とその名前空間 URI を説明しています。さらに，定義ファイル内の各定義の詳細についても説明しています。定義ファイルを示す例については，「15.4　外部定義関数定義ファイルの記述例」を参照してください。

15.1.1　外部定義関数定義ファイルの概要

説明

外部定義関数定義ファイルは，外部定義関数に関連する情報を指定します。

ファイル名

任意の名前

ファイル格納場所

外部定義関数定義ファイルは，常に次のディレクトリに保存してください。

運用ディレクトリ/conf/xml/

記述形式

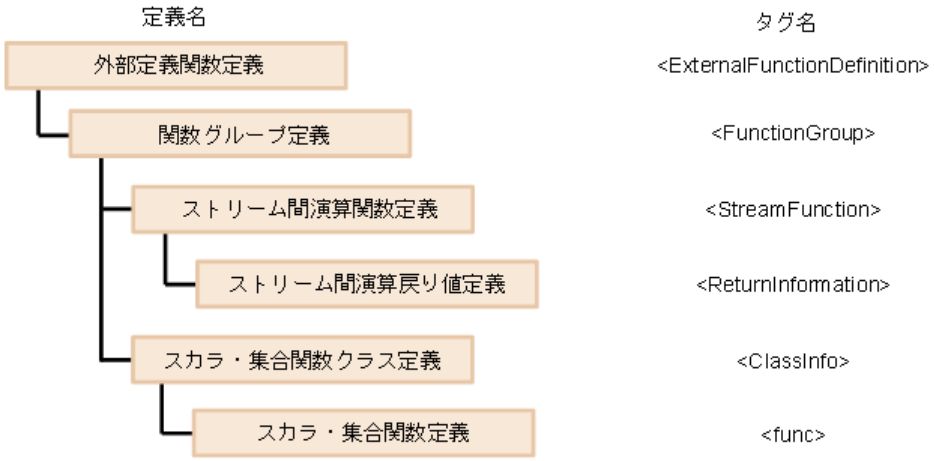
```
<?xml version="1.0" encoding="UTF-8"?>
<!-- All Rights Reserved. Copyright (C) 2014, Hitachi, Ltd. -->
<root:ExternalFunctionDefinition
 xmlns:root="http://www.hitachi.co.jp/soft/xml/sdp/function"
 xmlns:group="http://www.hitachi.co.jp/soft/xml/sdp/function/functiongroup">
  <group:FunctionGroup name="function-group-name" path="library-path" language="<development
-language>">
    <group:StreamFunction name="function-name" class="class-name">
      <group:ReturnInformation name="row-name" type="type-name"/>
                    :
    </group:StreamFunction>
  </group:FunctionGroup>
                    :
  <group:FunctionGroup name="function-group-name" path="library-path">
    <group:ClassInfo name="class-name">
      <group:func name="method-name"/>
                    :
    </group:ClassInfo>
```

```
  </group:FunctionGroup>
</root:ExternalFunctionDefinition>
```

定義の詳細

外部定義関数定義ファイルの定義は階層構造になっています。外部定義関数定義ファイルの構造を次の図に示します。

図 15-1　外部定義関数定義ファイルの構造



階層構造の定義は親子関係になっています。例えば，ストリーム間演算関数定義は，関数グループの定義タグで関数グループ定義の子要素として定義されます。また，関数グループの定義はストリーム間演算関数定義の親要素として定義されます。

次の表に，外部定義関数定義ファイル内の定義一覧を示します。

表 15-1　外部定義関数定義ファイルの定義一覧

定義	説明	参考
関数グループ定義 (FunctionGroup タグ)	外部定義関数の関数名，戻り値，およびユーザーが実装したコード（Java のクラスまたは C 言語の関数のライブラリ）を指定します。	15.2　外部定義関数定義ファイルの関数グループ定義
ストリーム間演算関数定義 (StreamFunction タグ)		15.3.1　ストリーム間演算関数定義
ストリーム間演算戻り値定義 (ReturnInformation タグ)		15.3.2　ストリーム間演算戻り値定義
スカラ・集合関数クラス定義 (ClassInfo タグ)	外部定義関数を実装したクラスを定義します。	15.3.3　スカラ・集合関数クラス定義
スカラ・集合関数定義 (func タグ)	外部定義関数を実装したメソッドを定義します。	15.3.4　スカラ・集合関数定義

15.1.2　外部定義関数定義ファイルの名前空間 URI

外部定義関数定義ファイルに，名前空間 URI の宣言を入力する必要があります。

宣言を次に示します。

```
http://www.hitachi.co.jp/soft/xml/sdp/function
```

```
http://www.hitachi.co.jp/soft/xml/sdp/function/functiongroup
```

15.2　外部定義関数定義ファイルの関数グループ定義

外部定義関数定義ファイルの関数グループ定義について説明します。

関数グループの定義では，FunctionGroup タグで関数グループを定義します。このタグでは，外部定義関数の実装を指定する関数グループを定義します。

最大 64 個の関数グループ定義を定義できます。この定義は省略できます。

記述形式

```
<FunctionGroup name="関数グループ名"
   path="クラスパス" language="{C|Java}">
   関数定義
</FunctionGroup>
```

定義の詳細

FunctionGroup タグ（全体情報の定義）

FunctionGroup タグは，関数グループ定義の全体情報を定義します。

- name="*関数グループ名*"

  関数グループを特定する名前を指定します。関数グループ名は，1～100 文字の半角英数字またはアンダーライン（_）で指定します。最初の文字には，半角英文字を指定してください。関数グループ名には，CQL の予約語を使用できません。この属性は省略できません。指定する関数名グループ名は，大文字と小文字の区別なしで，外部定義関数の定義内で一意の名前にする必要があります。

- path="*クラスパス*"

  Java の場合，外部定義関数を実装したクラスの jar ファイルのファイルパスまたはディレクトリパスを指定します。C 言語の場合，ライブラリのパスを指定します。クラスパスは 1 つ指定できます。この属性は省略できません。相対パスを指定する場合は，運用ディレクトリからの相対パスを使用してください。OS によって，パスの区切り文字，および大文字と小文字を区別するかどうかが異なります。

- language="{C|Java}"

  外部定義関数を処理するエンジンモードを定義します。この属性を省略すると，Java が仮定されます。なお，外部定義スカラ関数または外部定義集合関数の場合，この属性にエンジン種別を指定しても，必ず Java エンジンで動作します。

関数定義

関数の定義の詳細については，「15.3　外部定義関数定義ファイルの関数定義」を参照してください。

15.3　外部定義関数定義ファイルの関数定義

外部定義関数定義ファイルの関数定義について説明します。

次の表に，関数定義の一覧を示します。

表 15-2　関数定義の一覧

関数定義	親要素
ストリーム間演算関数定義 (StreamFunction タグ)	関数グループ定義
ストリーム間演算戻り値定義 (ReturnInformation タグ)	ストリーム間演算関数定義
スカラ・集合関数クラス定義 (ClassInfo タグ)	関数グループ定義
スカラ・集合関数定義 (func タグ)	スカラ・集合関数クラス定義

15.3.1　ストリーム間演算関数定義

ストリーム間演算関数定義（StreamFunction タグ）は，外部定義関数の関数名と，ユーザーが Java で作成したクラスとの関係を定義します。

ストリーム間演算関数定義で指定できる関数グループの最大定義数は，16 個です。

記述形式

```
<StreamFunction name="関数名"
   class="クラス名">
    ストリーム間演算戻り値定義
</StreamFunction>
```

定義の詳細

StreamFunction タグ（全体情報の定義）

StreamFunction タグは，ストリーム間演算関数定義の全体情報を定義します。

- name="*関数名*"

  外部定義関数を識別する名前を指定します。関数名は，1～100 文字の半角英数字またはアンダーライン（_）で指定します。最初の文字には，半角英文字を指定してください。関数名には，CQL の予約語を使用できません。この属性は省略できません。指定する関数名には，大文字と小文字の区別なしで，同じ関数グループ定義内で一意の名前を指定してください。

- class="*クラス名*"

  外部定義関数を実装したJavaのクラスのクラス名を指定します。クラスがパッケージにまとめられている場合，パッケージ名を含む形式のクラス名を指定してください。jp.co.Hitachi.soft.sdpで始まるパッケージ名は指定できません。クラス名には，大文字と小文字を区別して，同じ関数グループ定義内で一意の名前を指定してください。

  FunctionGroupタグのlanguageがJavaの場合，この属性を必ず指定してください。FunctionGroupタグのlanguageがCの場合，この属性は指定できません。

ストリーム間演算戻り値定義

ストリーム間演算戻り値定義の詳細については，「15.3.2　ストリーム間演算戻り値定義」を参照してください。

15.3.2　ストリーム間演算戻り値定義

説明

ストリーム間演算戻り値定義（ReturnInformationタグ）は，外部定義関数の処理結果である出力ストリームデータの各列の列名と型名を定義します。

1個の関数定義に対して，最大3,000個の戻り値定義を指定できます。この定義は省略できません。

定義形式

```
<ReturnInformation name="列名"
   type="型名">
</ReturnInformation >
```

定義の詳細

ReturnInformationタグ（全体情報の定義）

ストリーム間演算戻り値定義の全体情報を定義します。

- name="*列名*"

  外部定義関数の処理結果である出力ストリームデータの列を識別する名前を指定します。列名は，1～100文字の半角英数字またはアンダーライン（_）で指定します。最初の文字には，半角英文字を指定してください。列名には，CQLの予約語を使用できません。この属性は省略できません。列名には，大文字と小文字を区別しないで，同じ関数定義内で一意の列名を指定してください。

- type="*型名*"

  列名に対応する型を指定します。型名は大文字で指定してください。型名に指定できるのは，CQLのデータ型です。

15.3.3　スカラ・集合関数クラス定義

スカラ・集合関数クラス定義（ClassInfo タグ）は，外部定義関数のクラスを定義します。

スカラ・集合関数クラス定義で指定できる関数グループの数は，0～64 個です。この定義は省略できません。

記述形式

```
<ClassInfo name="クラス名">
</ClassInfo>
```

定義の詳細

ClassInfo タグ（全体情報の定義）

ClassInfo タグは，スカラ・集合関数クラス定義の全体情報を定義します。

- name="*クラス名*"

  1 文字以上の半角文字列または全角文字列でクラス名を指定してください。クラスがパッケージ化されている場合，パッケージ名を含む形式のクラス名を指定します。

15.3.4　スカラ・集合関数定義

スカラ・集合関数定義（func タグ）は，外部定義関を実装したメソッドを定義します。

スカラ・集合関数定義で指定できる関数グループの数は，1 個です。この定義は省略できません。

記述形式

```
<func name="メソッド名"/>
```

定義の詳細

func タグ（全体情報の定義）

func タグは，スカラ・集合関数定義の全体情報を定義します。

- name="*メソッド名*"

  外部定義関数を実装したメソッド名を指定します。メソッド名は，1～100 文字の半角英数字またはアンダーライン（_）で指定します。最初の文字には半角英字を指定してください。メソッド名には，CQL の予約語と同じ名前を使用できません。

  メソッド名は，大文字と小文字が区別されます。ただし，関数グループ定義内では，大文字と小文字の区別なしで一意の名前である必要があります。

メモ

クラスファイルに作成したメソッドのメソッド名がmethodの場合は，この属性にはmethodと指定します。また，クラスファイル内のメソッド名がMethodの場合は，この属性にはMethodと指定します。ただし，同じ関数グループ定義内で，この属性値にmethodとMethodが指定された場合，関数名の重複エラーが発生します。

15.4　外部定義関数定義ファイルの記述例

外部定義関数定義ファイルの記述例を示します。

外部定義関数定義ファイルの例

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- All Rights Reserved.Copyright (C) 2013, Hitachi, Ltd.-->
<root:ExternalFunctionDefinition
 xmlns:root="http://www.hitachi.co.jp/soft/xml/sdp/function"
xmlns:group="http://www.hitachi.co.jp/soft/xml/sdp/function/functiongroup">

 <!--関数グループ定義-->
 <group:FunctionGroup name="FG1" path="lib">
  <!--ストリーム間演算関数定義-->
  <group:StreamFunction name="FUNC1" class="samples.ExternalStreamFunction">
   <!--ストリーム間演算戻り値定義-->
   <group:ReturnInformation name="R1" type="INT" />
   <group:ReturnInformation name="R2" type="VARCHAR(10)" />
   <group:ReturnInformation name="R3" type="BIGINT" />
   <group:ReturnInformation name="R4" type="TIMESTAMP(9)" />
  </group:StreamFunction>
  <!--スカラ・集合関数クラス定義-->
  <group:ClassInfo name="samples.Functions">
   <!--スカラ・集合関数定義-->
   <group:func name="Mod"/>
   <group:func name="Countif"/>
  </group:ClassInfo>
 </group:FunctionGroup>

</root:ExternalFunctionDefinition>
```

説明

例では，関数グループ名FG1，関数名FUNC1 の外部定義関数を定義しています。

各定義の内容は次のとおりです。

- 関数グループ定義

  関数グループ名：FG1

  クラスパス：lib

- ストリーム間演算関数定義

  関数名：FUNC1

  クラス名：samples.ExternalStreamFunction

- ストリーム間演算戻り値定義

  外部定義関数の処理結果である出力ストリームデータの各列の列名および型名を，次の表に示すように定義しています。

表 15-3　出力ストリームデータの列名と型名の例

出力ストリームデータの列	列名	型名
1 列目	R1	INT
2 列目	R2	VARCHAR(10)
3 列目	R3	BIGINT
4 列目	R4	TIMESTAMP(9)

上記の例に対応するクエリ定義ファイルは次のとおりです。

```
REGISTER STREAM DATA0(ID VARCHAR(10), VAL BIGINT);

REGISTER QUERY Q1 SELECT ID, VAL FROM DATA0[PARTITION BY ID ROWS 1];
REGISTER QUERY Q2 ISTREAM (SELECT * FROM Q1);
REGISTER QUERY Q3 DSTREAM (SELECT * FROM Q1);
REGISTER QUERY SUM1 $*FG1.FUNC1[3](Q2,Q3);
```

- スカラ・集合関数クラス定義

  クラス名：samples.Functions

- スカラ・集合関数定義

  メソッド名：Mod

  メソッド名：Countif

  上記の例に対応するクエリ定義ファイルは次のとおりです。

```
REGISTER STREAM S1 (C1 INTEGER);

// 入力ストリームS1から，カラムC1が3で割り切れる値のタプルだけ出力するクエリ
REGISTER QUERY FILTER
ISTREAM ( SELECT * FROM S1[ROWS 10] WHERE $$FG1.MOD(S1.C1, 3) = 0 );

// 入力ストリームS1から，カラムC1が10のタプル個数を出力するクエリ
REGISTER QUERY COUNT10
ISTREAM ( SELECT $#FG1.COUNTIF(S1.C1, 10) FROM S1[ROWS 10]);
```

クエリ定義の詳細については，マニュアル『Hitachi Streaming Data Platform アプリケーション開発ガイド』を参照してください。

16 標準提供アダプターでのデータ処理，および定義ファイルの設定

この章では，「12.　SDP サーバおよび関連コンポーネント用定義ファイル」および「14.　アダプター構成定義ファイル」の定義ファイルで設定するデータの入出力やデータ編集の機能について，各処理の概要や処理の流れ，および必要な設定を説明しています。

16.1　この章の構成

この章では，標準提供アダプターでのデータ処理の概要や流れ，およびデータ処理をするために必要な定義ファイルの設定について説明します。また，標準提供アダプターまたはカスタムアダプターで，SDP サーバがタプルのタイムスタンプを調整するタイムスタンプ調整についても説明します。

この章で説明する機能の一覧，および参照先を次の表に示します。

表 16-1　機能の一覧および参照先

項番	機能		入力アダプター	出力アダプター	参照先
1	標準提供アダプターで使用できる機能	TCP データの入力	○	×	16.2　TCP データの入力
2		ファイルの入力	○	×	16.3　ファイルの入力
3		HTTP パケットの入力	○	×	16.4　HTTP パケットの入力
4		レコードのフィルタリング	○	○	16.5　レコードのフィルタリング
5		レコードの抽出	○	×	16.6　レコードの抽出
6		クエリグループまたは外部出力アダプターへの出力	×	○	16.7　クエリグループまたは外部出力アダプターへの出力
7		ファイルへの出力	×	○	16.8　ファイルへの出力
8		ダッシュボードへの出力	×	○	16.9　ダッシュボードへの出力

（凡例）

○：使用できます。

×：使用できません。

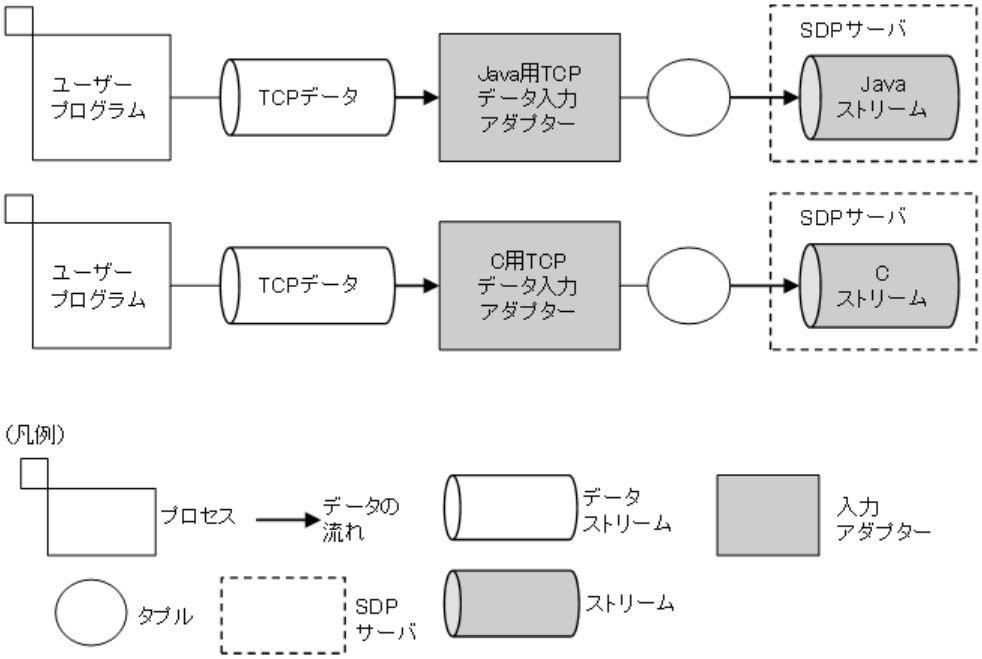
16.2　TCP データの入力

ここでは，TCP データ入力の概要，TCP データ入力アダプターの構成，TCP データ入力コネクター，および TCP データの入力の設定について説明します。

16.2.1　TCP データ入力の概要

Streaming Data Platform では，内部標準アダプターの 1 つに TCP データ入力アダプターを提供しています。TCP データ入力アダプターは，ユーザープログラム，およびカスケーディングアダプターと TCP 接続を確立し，データを受信します。TCP データ入力アダプターは，受信した TCP データをタプルに変換し，タプルを SDP サーバに送信します。

図 16-1　TCP データの受信とタプルの送信



TCP データ入力アダプター：SDP サーバの Java ストリームまたは C ストリームにタプルを送信します。

16.2.2　TCP データ入力アダプターの構成

TCP データ入力コネクターは，TCP データ入力アダプターの入力コネクターとして設定する必要があります。以下の図と表は，入力データアダプター構成でのコールバックの組み合わせを示しています。外部入力アダプターを TCP データの送信元として使用する場合，TCP データ入力アダプターを使用するホスト上で SDP ブローカーが開始されている必要があります。外部入力アダプターから接続要求を受信すると，SDP ブローカーは，通信に必要となる TCP データ入力アダプターを起動します。

TCP データ入力アダプターには，接続先のストリームによって Java 用と C 用の 2 種類のタイプがあります。また，それぞれのタイプで組み合わせられるコールバックが異なります。

図 16-2　Java 用 TCP データ入力アダプターの構成

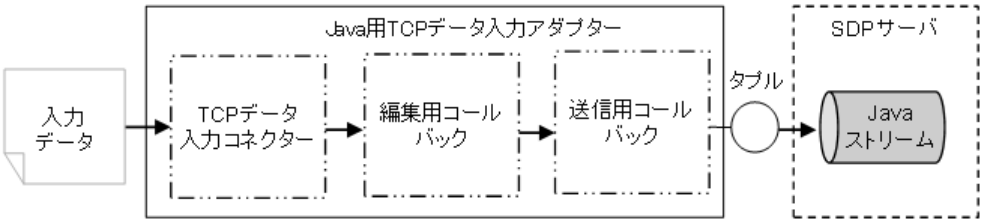


図 16-3　C 用 TCP データ入力アダプターの構成

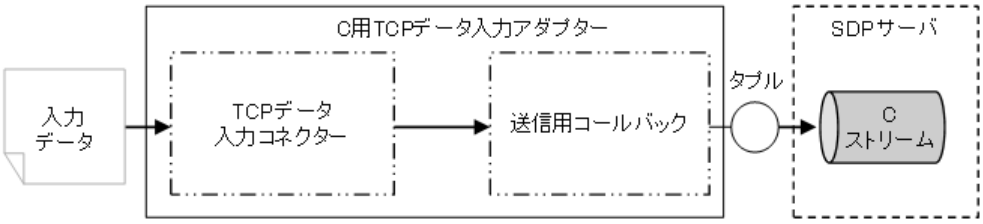


表 16-2　コールバックの組み合わせ一覧

アダプター タイプ	コールバックの組み合わせ		
	入力用コールバック	編集用コールバック	送信用コールバック
Java 用 TCP データ入力アダプター	TCP データ入力コネクター	編集用コールバックを設定可能。または省略可能。	送信用コールバック
C 用 TCP データ入力アダプター	TCP データ入力コネクター	-	送信用コールバック

16.2.3　データ送信元として動作するユーザープログラム

TCP データ入力アダプターにデータを送信するユーザープログラムは，外部アダプターライブラリ，または HSDP クライアントライブラリによって実装できます。これらのライブラリを利用しないで，任意の TCP クライアントライブラリを利用して，データ送信ユーザープログラムを実装することもできます。

外部アダプターライブラリを使用してユーザープログラムを実装する場合の設定

ユーザープログラムは，外部入力アダプターの定義ファイルで，TCP データ入力アダプターを使用するホストのストリーム情報と SDP ブローカーのアドレスの両方を指定します。これによって，外部入力アダプターで通信が確立されます。

HSDP クライアントライブラリを使用してユーザープログラムを実装する場合の設定

ユーザープログラムは，データ送信先の TCP データ入力アダプターが動作するホストの，ホスト名またはアドレスと，TCP データ入力アダプターの待ち受けポート番号を指定します。これによって，HSDP クライアントライブラリで通信が確立されます。

ライブラリを使わず任意の TCP クライアントのユーザープログラムを実装した場合に必要な設定

ユーザープログラムは，データ送信先の TCP データ入力アダプターが動作するホストの，ホスト名またはアドレスと，TCP データ入力アダプターの待ち受けポート番号を指定し，TCP クライアントと TCP データ入力アダプターとの接続を確立します。

16.2.4　TCP データ入力コネクター

このセクションでは，TCP データ入力コネクターの詳細について説明します。

TCP データ入力アダプターの接続数およびタプルの順序

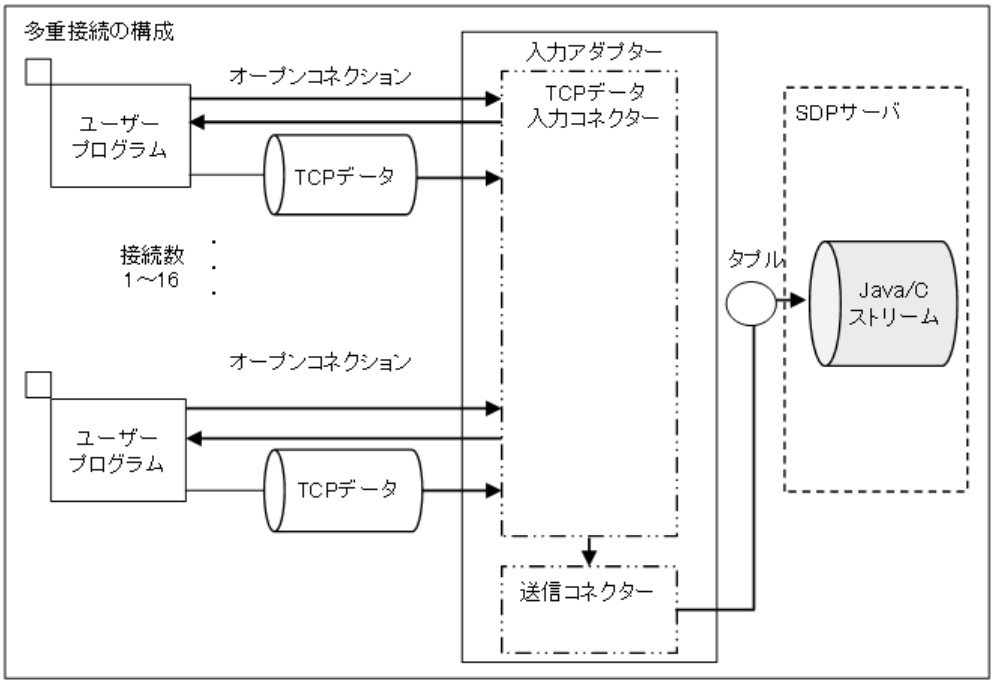
TCP データ入力アダプターの起動後，TCP データ入力コネクターは，TCP 接続を介してデータソースからデータを受信します。ユーザープログラムと，Java 用 TCP データ入力アダプターまたは C 用 TCP データ入力アダプターとの間で確立される接続数を次の表に示します。

また，複数のデータソースからデータを受信する場合，タプルに設定されたタイムスタンプの順番と，タプルの到着順が一致しないことがあります。この場合の動作についても次の表に示します。

表 16-3　接続数

アダプタータイプ	接続数	タプル時系列
Java 用 TCP データ入力アダプター	「図 16-4　接続数」に示されるとおり，（各アダプターにつき）1～16 の接続を確立できます。	タプルに設定されたタイムスタンプと順序が逆転して到着したタプルは破棄されます。そのため，SDP サーバのストリームに送信されるタプルは時系列順であることが保証されています。
C 用 TCP データ入力アダプター	「図 16-4　接続数」に示されるとおり，（各アダプターにつき）1～16 の接続を確立できます。	タプルに設定されたタイムスタンプと順序が逆転して到着したタプルを破棄しません。そのため，時系列順のデータを分析したい場合は，外部定義関数で時系列順に並べる処理を実装する必要があります。

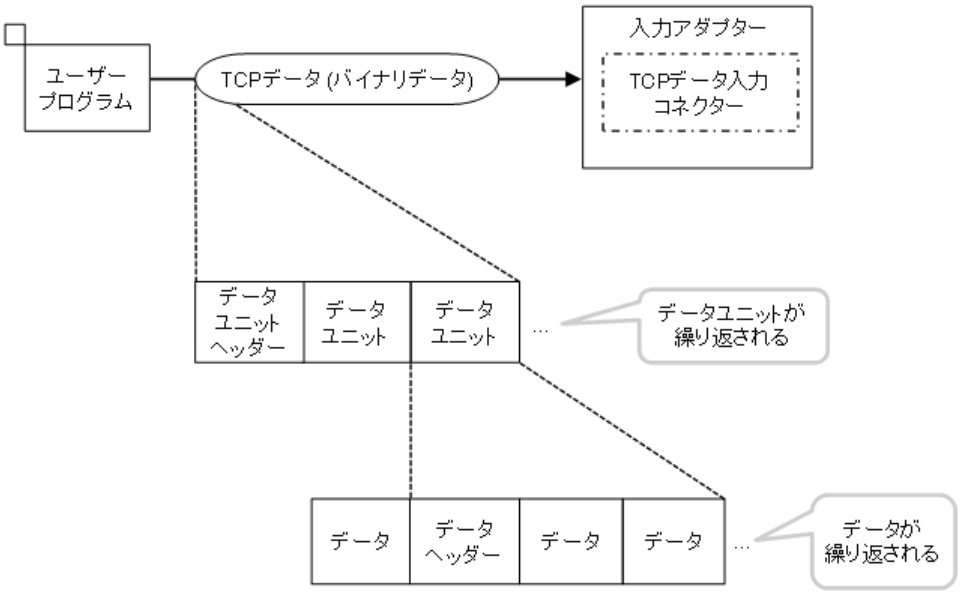
図 16-4　接続数



TCP データ入力コネクターの TCP データ形式

このコネクターでは，次のように TCP データを入力します。

図 16-5　TCP データ形式



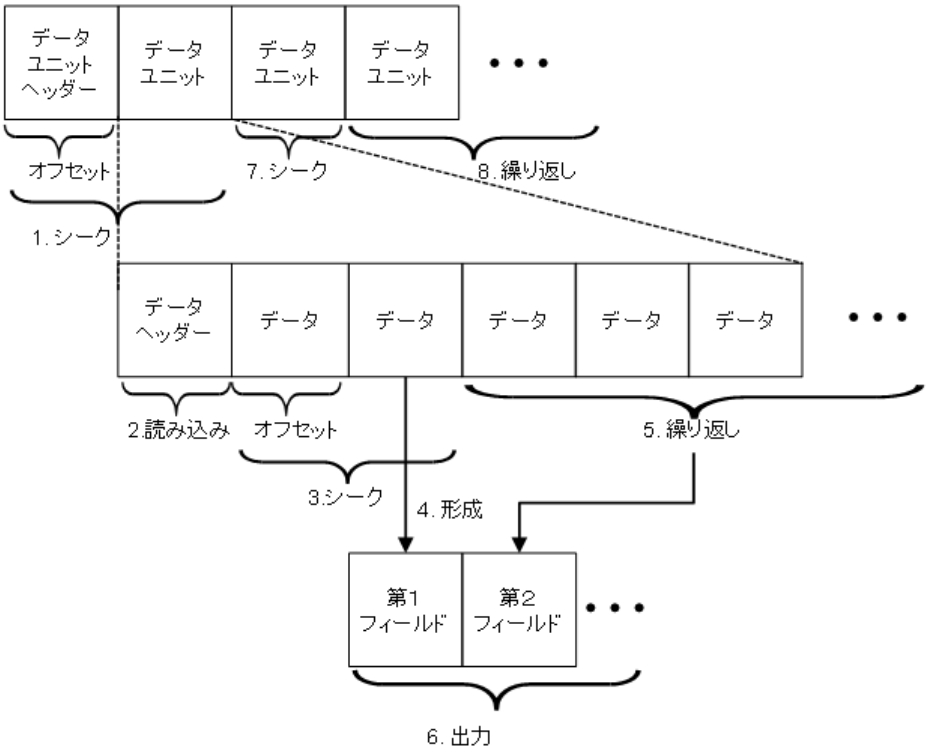
この図に示したとおり，TCP データは，データユニットヘッダーと一連の 1 つ以上のデータユニットで構成されます。

データユニットヘッダーは，ユーザーが任意の用途で必要に応じて付けられるデータ領域であり，TCP データ入力コネクターの処理には利用しません。

各データユニットは，既定のデータ項目数で構成されます。

TCP データ入力コネクターは，次のように受け取った TCP データを処理し，ストリームに入力するレコードへと形成します。

図 16-6　データユニットをレコードに形成



この図に示した各セクションは次のとおりです。

1. アダプター構成定義ファイルで定義するデータユニットヘッダーのデータサイズをオフセットとして，データユニットをシークします。

   デフォルトのオフセットサイズは 0 バイトです。
2. シークしたデータユニットのデータヘッダーを読み込みます。
3. アダプター構成定義ファイルで定義するデータサイズをオフセットとして，データをシークします。

   オフセットは，シークするデータごとに定義できます。デフォルトのオフセットサイズは 0 バイトです。
4. 読み込んだデータをレコードを構成するフィールドとして形成します。
5. セクション 3 と 4 を繰り返します。
6. 最後のデータまでシークが完了したら，形成したレコードを次のコールバックへ出力します。
7. 次のデータユニットをシークします。
8. 最後のデータユニットをシークするまで，セクション 2 から 7 を繰り返します。

ユーザーはアダプターの構成ファイルで，オフセットの各バイトサイズと形成対象のデータを定義し，レコードフィールドに形成するデータを選択できます。ヘッダー情報の詳細は次のとおりです。

項目	説明	サイズ	データ型
データの種類	データの種類を指定します。 標準データを表す0を指定します。その他の値は，将来の拡張用に予約されています。	2バイト	short
（予約済み）	将来の拡張用に予約されたドメインです。	2バイト	short

データ型が可変長の文字（VARCHAR），またはUnicodeの固定長の文字（CHAR）のデータの場合，データ送信元のユーザープログラムは，次のデータ形式でTCPデータ入力コネクターにデータを送信します。TCPデータ入力コネクターは，受け取った文字データをレコードに形成します。

図 16-7　TCPデータ入力コネクターのデータ形式

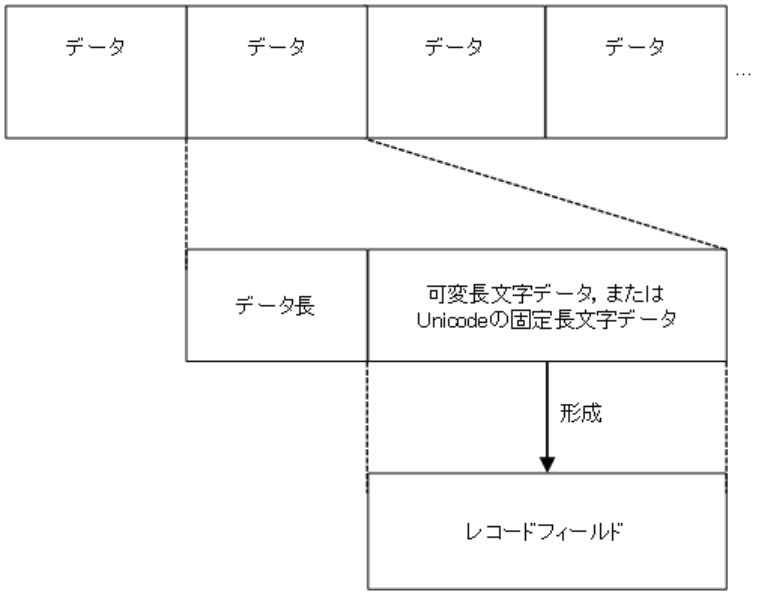


表 16-4　データ形式の説明

項目	説明	サイズ	データ型	値
データ長	可変長文字データを格納するバイト配列の長さを指定します。 この値が次の値を超える場合，TCPデータ入力コネクターは警告メッセージKFSP48922-Wを出力し，先頭から許容する最大サイズで文字列変換処理をします。 • アダプター構成定義ファイルのTCPデータ入力コネクター定義のsize属性値とcharCode属性値に指定した文字コードのエンコーディングから求められる，TCPデータ入力コネクターが受信できる最大のバイトサイズ 文字列変換の結果，この文字列の長さがアダプター構成定義ファイルのTCPデータ入力コネクター定義のsize属性値を超える場合，TCPデータ入力コネクターは警	2バイト	short	0～32767の整数

項目	説明	サイズ	データ型	値
データ長	告メッセージKFSP48916-W を出力し，可変長文字データを先頭からsize 属性値の長さに切り詰めます。 0 を指定すると，可変長文字データを省略する必要があります。この場合，TCP データ入力コネクターはレコードフィールドに null 文字を作成します。	2 バイト	short	0～32767 の整数
可変長文字データ	可変長文字データを格納するバイト配列を指定します。バイト配列の長さは，データ長に指定した値と同じでなければなりません。 TCP データ入力コネクターは，このデータの値（例えば，文字コードと制御文字）をチェックし，指定されたバイト配列を変更しないでレコードフィールドに形成します。 ただし，文字列変換の結果，この文字列の長さがアダプター構成定義ファイルの TCP データ入力コネクター定義のsize 属性値を超える場合，TCP データ入力コネクターは警告メッセージKFSP48916-W を出力し，固定長文字データを先頭からsize 属性値の長さに切り詰めます。 入力データをinput@charCode タグに指定されたエンコーディングの値でデコードします。 Java 用 TCP データ入力アダプターの場合：デコードできないバイトデータは，指定されたエンコーディング値に対応する Java の文字セットのデフォルト置換バイト配列に置き換えて処理します。 C 用 TCP データ入力アダプターの場合：指定されたバイト配列を何も置き換えないでそのまま処理します。	1～32767 バイト	varchar	すべての文字

TCP データ入力コネクターのデータのバイトオーダー

このコネクターは，ビッグエンディアン方式のバイトオーダーに従って，データをレコードフィールドに形成します。

TCP データ入力コネクターの TCP 接続の待ち受け再開

ユーザープログラムが TCP 接続を閉じると，このコネクターは TCP 接続の待ち受けを再開し，TCP データ入力アダプターは処理を継続します。ユーザープログラムが TCP データの送信中に TCP 接続を閉じると，このコネクターは受信している TCP データを削除し，TCP 接続の待ち受けを再開します。

16.2.5　TCP データの入力の設定

SDP ブローカー連携によって TCP データ入力アダプターを自動起動する場合，TCP データ入力アダプターのアダプター構成定義ファイルは必要ありません。

TCP データ入力アダプターが特定のポート番号でデータを受信する場合，アダプター構成定義ファイルの TCP データ入力コネクター定義でコネクターの情報を定義してください。TCP データ入力コネクターの定義については，「14.7.5　TCP データ入力コネクター定義」を参照してください。

16.2.6　TCP データ入力アダプターの機能の一覧

TCP データ入力アダプターの機能の一覧を示します。

表 16-5　TCP データ入力アダプターの機能の一覧

機能			Java 用 TCP データ入力アダプターにサポートされているか	C 用 TCP データ入力アダプターにサポートされているか
大分類	中分類	小分類		
固定長データ型	BYTE	-	はい	はい
	SHORT	-	はい	はい
	INT	-	はい	はい
	LONG	-	はい	はい
	FLOAT	-	はい	はい
	DOUBLE	-	はい	はい
	BIG_DECIMAL	-	はい	いいえ
	CHAR	ASCII	はい	はい
		Unicode※1	はい	いいえ
	DATE	-	はい	いいえ
	TIME	-	はい	いいえ
	TIMESTAMP	-	はい	はい
可変長データ型	VARCHAR	ASCII	はい	はい
		Unicode※1	はい	いいえ
データ形式	固定長データ	-	はい	はい
	可変長データ	-	いいえ	はい
データオフセット設定	-	-	はい	はい
ゼロ拡張	-	-	はい	はい
レコード形式	単一レコード	-	はい	はい
	複数レコード	-	いいえ	いいえ
接続管理 （TCP データ送信元とアダプター間）	シングル接続	-	はい	はい

機能			Java 用 TCP データ入力アダプターにサポートされているか	C 用 TCP データ入力アダプターにサポートされているか
大分類	中分類	小分類		
接続管理 (TCP データ送信元とアダプター間)	マルチ接続	-	はい	はい
SDP ブローカー接続	TCP 接続の統合	-	はい	いいえ
型チェック	アダプター入力ストリーム	-	はい	はい
アダプターと TCP データ送信元との接続性	Java 用 TCP カスケーディングアダプター	-	はい	はい
	C 用 TCP カスケーディングアダプター	-	はい	はい
	外部入力アダプター	-	はい	いいえ
	HSDP クライアントライブラリ	-	はい※2	はい

注※1

指定できる文字コードは，アダプター構成定義ファイルの TCP データ入力コネクター定義の charCode 属性値で指定できる値と同じです。TCP データ入力コネクター定義の詳細については，「14.7.5　TCP データ入力コネクター定義」を参照してください。

注※2

固定長データだけサポートされています。

オフセットパラメーターを使用して，ヘッダーをスキップしてください。

16.3　ファイルの入力

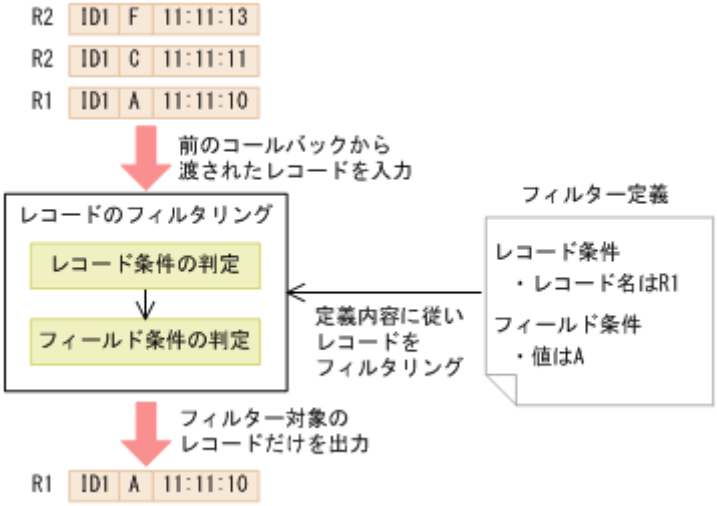
ここでは，ファイル入力の概要，データ処理の流れ，および定義ファイルでの設定内容について説明します。

16.3.1　ファイルの入力の概要

ファイルの入力では，エラーログやアクセスログなどのファイルからデータを読み込み，読み込んだデータをストリームデータ処理エンジンで処理できる形式に変換してからストリームデータ処理エンジンに送信します。ファイルの入力の処理は入力アダプターで実施します。

ファイルの入力の概要を次の図に示します。

図 16-8　ファイルの入力の概要

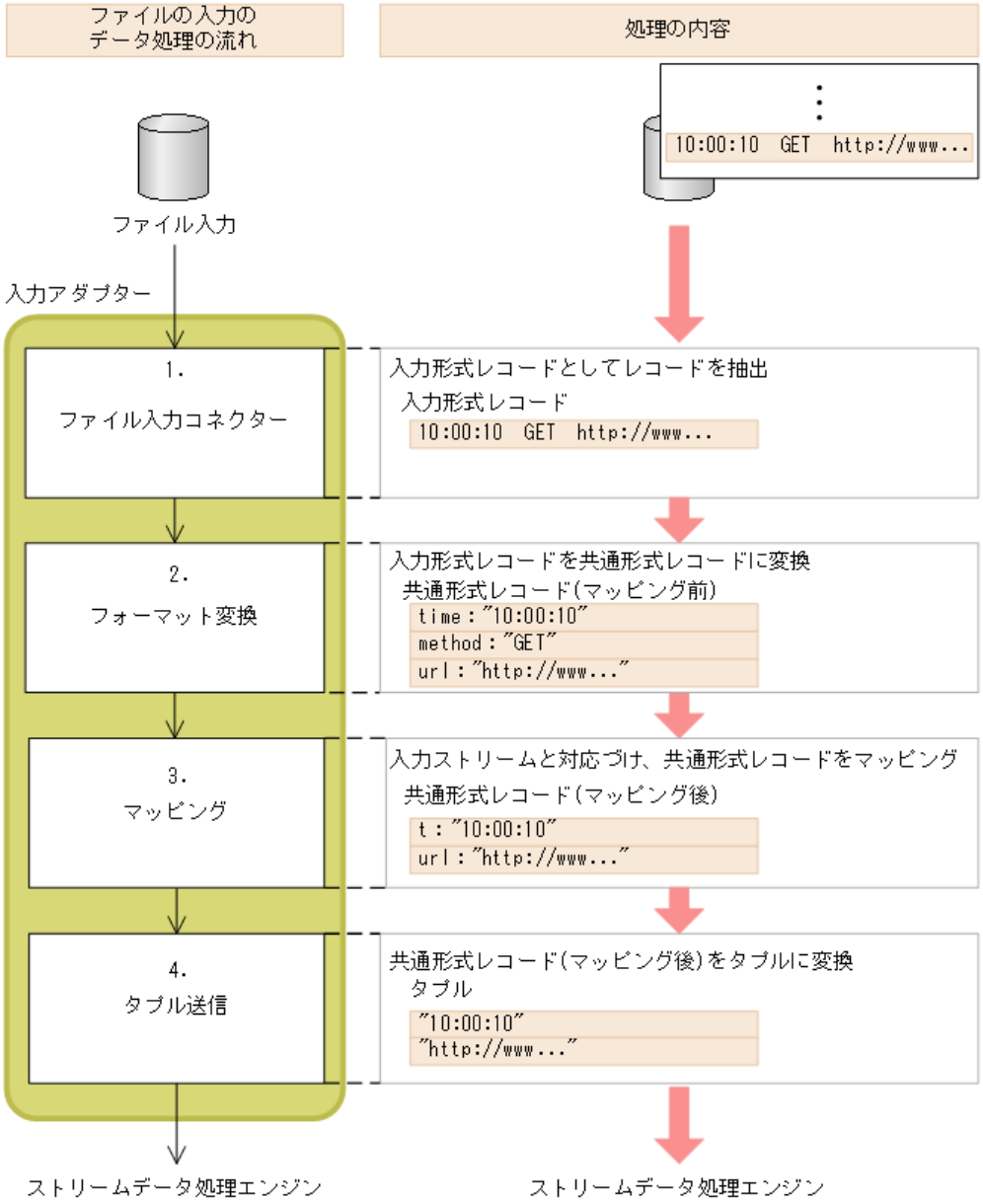


1. データ入力
   ファイル入力コネクターで，入力ファイルからデータを読み込みます。
2. データ編集
   読み込んだデータのフォーマット変換およびマッピングをして，データを編集します。
3. タプル送信
   編集したデータをタプルに変換して，ストリームデータ処理エンジンに送信します。

16.3.2　ファイルの入力のデータ処理の流れ

入力アダプターでのファイルの入力のデータ処理の流れと処理の内容を次の図に示します。

図 16-9　ファイルの入力のデータ処理の流れと処理の内容



入力アダプターで扱うデータ形式について，「入力アダプターで扱うデータ形式」で説明します。また，各処理の詳細について，「ファイル入力コネクターによるファイルの入力」以降で説明します。

入力アダプターで扱うデータ形式

入力アダプターで扱うデータ形式には，入力形式レコード，および共通形式レコードがあります。それぞれのデータ形式について説明します。

入力形式レコード

入力ファイルから取得した，行単位のデータのことです。入力アダプターでは，1 行のデータを 1 入力形式レコードとして扱います。

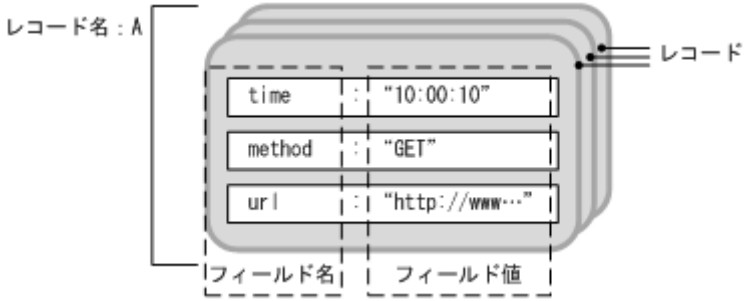
共通形式レコード

複数のフィールド名とその値であるフィールド値から成る，データの集合のことです。フィールド名とは，入力形式レコードから意味のある単位（フィールド）に切り出したデータに付けた名前で，フィールド値とはその値のことを指します。

共通形式レコードは，入力アダプターおよび出力アダプター内部で扱う共通のデータ形式です。

標準提供アダプターでは，共通形式レコードの複数のフィールド名とそれぞれに対応するフィールド値の組み合わせをレコード構成として管理し，レコード名で識別します。

図 16-10　共通形式レコードの構成



ファイル入力コネクターによるファイルの入力

ファイル入力についての定義は，アダプター構成定義ファイルのファイル入力コネクター定義で定義します。

ファイル入力コネクターでは，入力ファイル格納ディレクトリに格納されている 1 つ以上のファイルから行単位でデータを読み込んで，入力形式レコードに変換します。ファイル入力コネクターで読み込んだ 1 行のデータが，1 入力形式レコードとなります。

なお，入力アダプターでは複数の入力形式レコードを一度に読み込むことができます。ファイル入力コネクターで読み込んだ入力形式レコード数が，フォーマット変換，マッピング，タプル送信の各処理で一度に処理するレコード数となります。

ここでは，ファイル入力コネクターで読み込む入力ファイルの種類や構成などについて説明します。

ファイルの種類

ファイル入力コネクターで読み込む入力ファイルは，文字データだけで構成されたテキストファイルです。レコードは可変長です。

ファイル構成

ファイル入力コネクターで読み込む入力ファイルは，次のファイル構成のファイルです。

ファイル構成	説明	前提条件
ラップアラウンド	入力対象のファイル数が固定されている場合に，設定されたファイルの順番に情報を書き込むファイル構成です。	情報を書き込むファイルの順番が決まっていて，必ずその順番で書き込みます。 書き込みが完了したファイルに書き込む場合は，データをクリアしてから新しい情報を書き込みます。

ファイル構成	説明	前提条件
非ラップアラウンド	入力対象のファイル数が固定されていない場合に，設定されたファイルの順番に情報を書き込むファイル構成です。	情報を書き込むファイルの順番が決まっていて，必ずその順番で書き込みます。

また，レコードおよびファイルの生成順序は時系列である必要があります。

ファイル名

ファイル入力コネクターで読み込むファイルは，ファイル入力コネクター定義のfile タグのname 属性で，ファイル名または通番で指定します。name 属性については，「14.7.1　ファイル入力コネクター定義」を参照してください。

ファイルを読み込む順番

ファイル入力コネクターは，ファイル入力コネクター定義のfile タグのname 属性で指定したファイル名の順番，または入力ファイルの更新時刻の順番で入力ファイルを読み込みます。ファイルの読み込み順は，input タグのreadOrder 属性で指定します。

なお，入力ファイルの更新時刻が等しい場合，読み込みの順番は保証されません。

ファイル読み込み処理モード

ファイル入力コネクターがファイルを読み込む方式には，バッチ処理モードとリアルタイム処理モードがあります。それぞれの処理モードについて次の表に示します。なお，どちらの処理モードの場合も，読み込みが完了すると入力アダプターが自動で停止します。

処理モード	読み込み処理の内容	
	読み込み方式	読み込み完了条件
バッチ処理モード	入力アダプターの起動時に，入力ファイル格納ディレクトリに入力ファイルがあるかどうかを確認して，入力ファイルがあった場合に読み込み処理をします。	次の条件のどちらかに一致した時点で，読み込みを完了します。 • 入力アダプターの起動時に，入力ファイル格納ディレクトリにあった，すべてのファイルの読み込みが完了したとき。 • 定義に指定したすべてのファイルの読み込みが完了したとき。
リアルタイム処理モード	入力ファイル格納ディレクトリを一定間隔で監視し，入力ファイルが出現した時点で，逐次読み込み処理をします。 入力ファイル格納ディレクトリの監視間隔，および監視回数はファイル入力コネクター定義のinput タグで定義します。監視回数を超えた場合は，警告メッセージを出力したあと，処理を続行します。	次の条件のどちらかに一致した時点で，読み込みを完了します。 • 入力ファイルを通番で指定する場合：通番の最後に当たるファイルの読み込みが完了したとき。 • 入力ファイルをファイル名で指定する場合：定義に指定したすべてのファイルの読み込みが完了したとき。

ファイル入力コネクターの動作中に，入力ファイルを入力ファイル格納ディレクトリにコピーした場合または移動した場合，ファイル入力コネクターは 1 秒間待機したあと，再度ファイルの読み込み処理をします。この処理は読み込みができるまで，最大 5 回繰り返し，読み込みができない場合はメッセージKFSP46200-E が出力されます。

また，入力ファイル格納ディレクトリで入力ファイルを新規に作成した場合，または入力ファイルに追加の書き込みがあった場合，新規作成または追加書き込みしたレコードに読み込まれません。

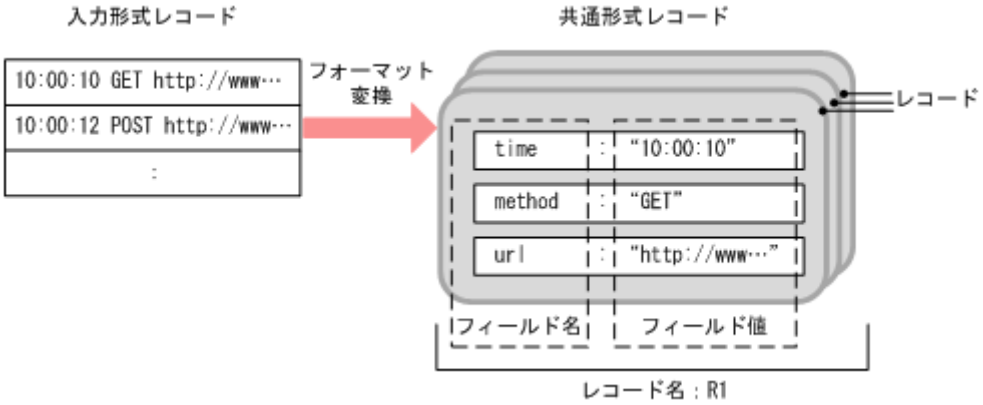
フォーマット変換

フォーマット変換についての定義は，アダプター構成定義ファイルのフォーマット変換定義で定義します。

フォーマット変換では，入力形式レコードから意味のある単位（フィールド）に切り出して，共通形式レコードに変換します。

入力形式レコードから共通形式レコードへのフォーマット変換の例を次の図に示します。この例では，入力形式レコードは，空白で区切られた 3 つのフィールドで構成され，第 1 フィールドはTIME 型，第 2，3 フィールドは文字列型にフォーマット変換されます。

図 16-11　入力形式レコードから共通形式レコードへのフォーマット変換の例



この図の場合の共通形式レコードのレコード構成と指定するフォーマット変換定義のタグを次の表に示します。

レコード構成	タグ
レコード名：R1 レコード構成：($_time)△($_method)△($_url)	record タグ（レコード定義）
フィールド名：time，型：TIME	field タグ（フィールド定義）
フィールド名：method，型：STRING	
フィールド名：url，型：STRING	

(凡例)

△：半角スペース

変換できるデータ型および共通形式レコードのレコード構成の設定については，「14.8.1　フォーマット変換定義」を参照してください。

なお，フォーマット変換では，複数のレコード構成を定義できます。複数のレコード構成が定義された場合は，どのレコード構成に該当するかを自動で選択してフォーマット変換します。

複数のレコード構成が定義された場合，レコード構成は次のように選択されます。

1. ファイル入力コネクターでは，読み込んだ入力形式レコードの構成が，フォーマット変換定義のrecordタグで指定したレコード構成に一致するかどうかを，レコード構成を定義した順にチェックします。一致した最初のレコード構成を選択します。
2. 1.で一致するレコード構成がない場合，入力形式レコードは破棄されます。この場合，入力アダプターは，フォーマット変換定義のunmatchedFormat タグで定義された，次のどれかを処理します。
   - 処理を続行します。
   - 警告メッセージを出力し，処理を続行します。
   - エラーメッセージを出力し，入力アダプターを停止します。

> **メモ**
>
> フォーマット変換のあとには，レコードのフィルタリング，レコードの抽出，およびレコード間のマッピングが順不同で実施できます。必要に応じて実施してください。
>
> レコードのフィルタリングについては，「16.5　レコードのフィルタリング」を参照してください。レコードの抽出については，「16.6　レコードの抽出」を参照してください。また，レコード間のマッピングについては，「16.3.2　ファイルの入力のデータ処理の流れ」の「(1)　マッピング」を参照してください。

(1)　マッピング

マッピングについての定義は，アダプター構成定義ファイルのマッピング定義で定義します。

マッピングには，レコードとストリーム間のマッピング，およびレコード間のマッピングがあります。それぞれのマッピングの概要を次の表に示します。

表 16-6　マッピングの概要

項番	マッピングの種類	説明
1	レコードとストリーム間のマッピング	マッピングの前のコールバックで出力される共通形式レコード（マッピングの変換元）と，入力ストリームの形式に従った共通形式レコード（マッピングの変換先）との関連づけをします。 レコードとストリーム間のマッピングは，タプル送信の前に必ず実施します。
2	レコード間のマッピング	マッピングの前のコールバックで出力される共通形式レコード（マッピングの変換元）を編集し，マッピングの変換先の共通形式レコードに変換します。 レコード間のマッピングは，フォーマット変換のあと，かつレコードとストリーム間のマッピングの前に，必要に応じて実施します。 マッピングの変換元の共通形式レコードのフィールド名を変更したい場合や，複数のフィールドから次のコールバックの処理に不要なフィールドを削除したい場合などに使用できます。

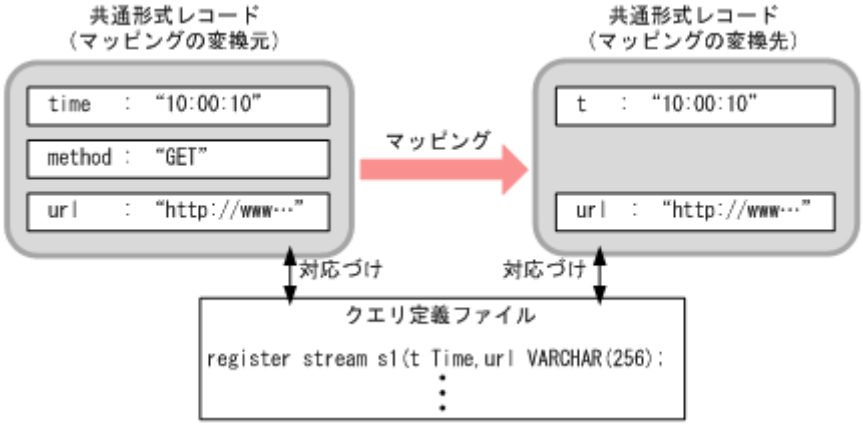
項番	マッピングの種類	説明
2	レコード間のマッピング	また，組み込み関数※を使用して，マッピングの変換元の共通形式レコードから文字列や時刻を取得し，取得した文字列や時刻をマッピングの変換先の共通形式レコードに反映することもできます。 なお，レコード間のマッピングは複数定義できます。

※

レコード間のマッピングで使用できる組み込み関数は，アダプター構成定義ファイルのmap タグのfunction 属性で指定します。function 属性で指定できる組み込み関数については，「14.8.2　マッピング定義」を参照してください。

入力アダプターでのレコードとストリーム間のマッピングの例を次の図に示します。この例では，入力ストリームs1 に必要なtime フィールドとurl フィールドを入力ストリームのスキーマにマッピングして，共通形式レコード（マッピングの変換先）に変換しています。

図 16-12　入力アダプターでのレコードとストリーム間のマッピングの例



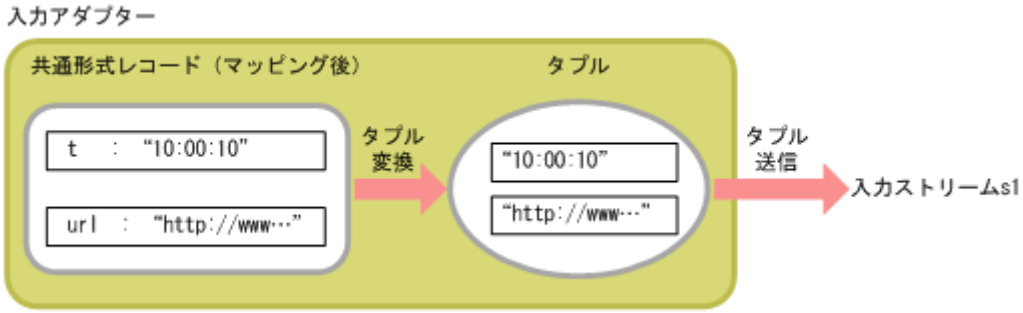
(2)　タプル送信

タプル送信についての定義は，アダプター構成定義ファイルの入力ストリーム定義に定義します。

タプル送信では，マッピング結果を基に，共通形式レコードをタプルに変換したあとで，入力ストリーム定義に従って入力ストリームにタプルを送信します。

入力アダプターから入力ストリームへのタプル送信の例を次の図に示します。この例では，入力ストリームs1 にタプルを送信します。

図 16-13　入力アダプターからのタプル送信の例



16.3.3　ファイルの入力の設定

ファイル入力について設定が必要な定義ファイルは，アダプター構成定義ファイルです。アダプター構成定義ファイルの入力アダプター定義で，次の CB 定義に設定します。

- 入力用 CB 定義

  ファイル入力コネクター定義に，入力コネクターの情報を定義します。定義の詳細については，「14.6.1 入力用 CB 定義」，および「14.7.1　ファイル入力コネクター定義」を参照してください。

- 編集用 CB 定義

  フォーマット変換定義に，入力データのデータ形式を定義します。マッピング定義には，マッピング情報を定義します。定義の詳細については，「14.6.3　編集用 CB 定義」，「14.8.1　フォーマット変換定義」，または「14.8.2　マッピング定義」を参照してください。

- 送信用 CB 定義

  入力ストリーム定義に，入力アダプターが接続する入力ストリームの情報を定義します。定義の詳細については，「14.6.4　送信用 CB 定義」，および「14.9.1　入力ストリーム定義」を参照してください。

16.4　HTTP パケットの入力

ここでは，HTTP パケットの入力の概要，データ処理の流れ，および定義ファイルでの設定内容について説明します。

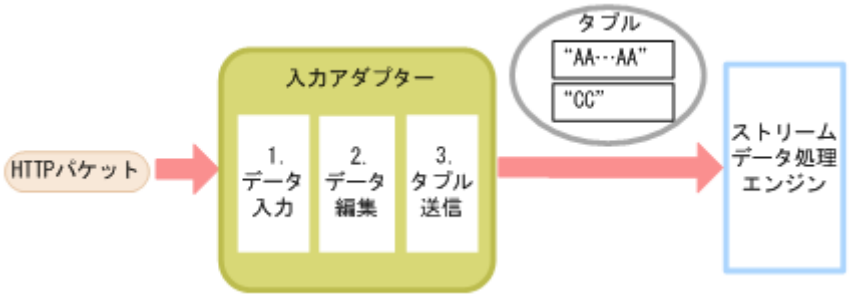
16.4.1　HTTP パケットの入力の概要

HTTP パケットの入力では，HTTP パケットデータを標準入力で取得し，取得したデータを解析して，ストリームデータ処理エンジンで処理できる形式に変換してからストリームデータ処理エンジンに送信します。

HTTP パケットの入力の処理は入力アダプターで実施します。

HTTP パケットの入力の概要を次の図に示します。

図 16-14　HTTP パケットの入力の概要



1. データ入力

   HTTP パケット入力コネクターで，パケットアナライザーから標準出力に出力されたデータを読み込みます。読み込んだ HTTP パケットデータを解析して，データを変換します。

2. データ編集

   マッピングをして，データを編集します。

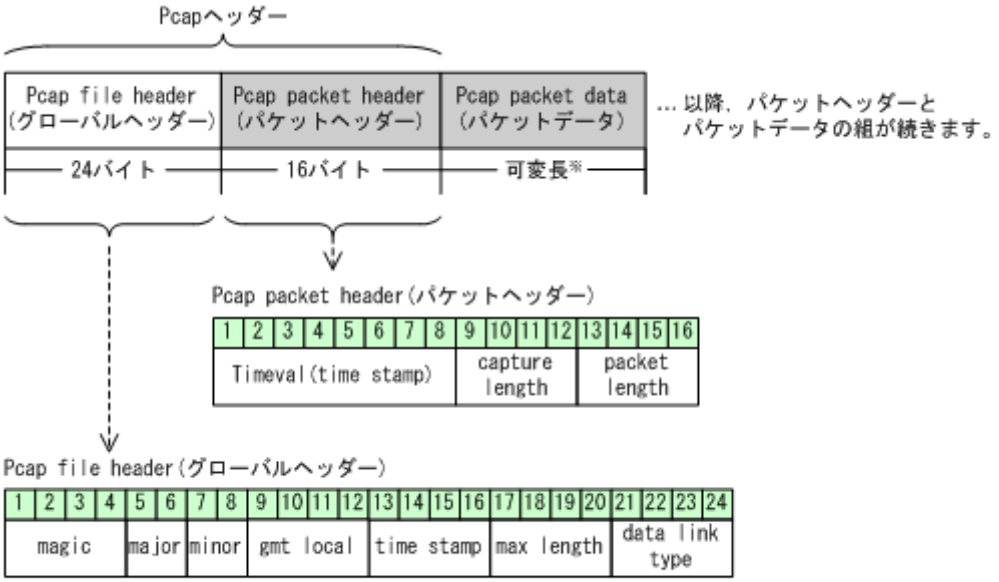
3. タプル送信

   編集したデータをタプルに変換して，SDP サーバに送信します。

なお，HTTP パケットを入力するには，Pcap 形式でパケットを出力できるパケットアナライザーが必要です。Linux の場合は`tcpdump` を使用できます。

HTTP パケットの入力で扱えるのは，次の図に示すPcap 形式のフォーマットで出力される HTTP パケットです。

図 16-15　Pcap 形式のフォーマット



注※　サイズはパケットヘッダーから取得します。

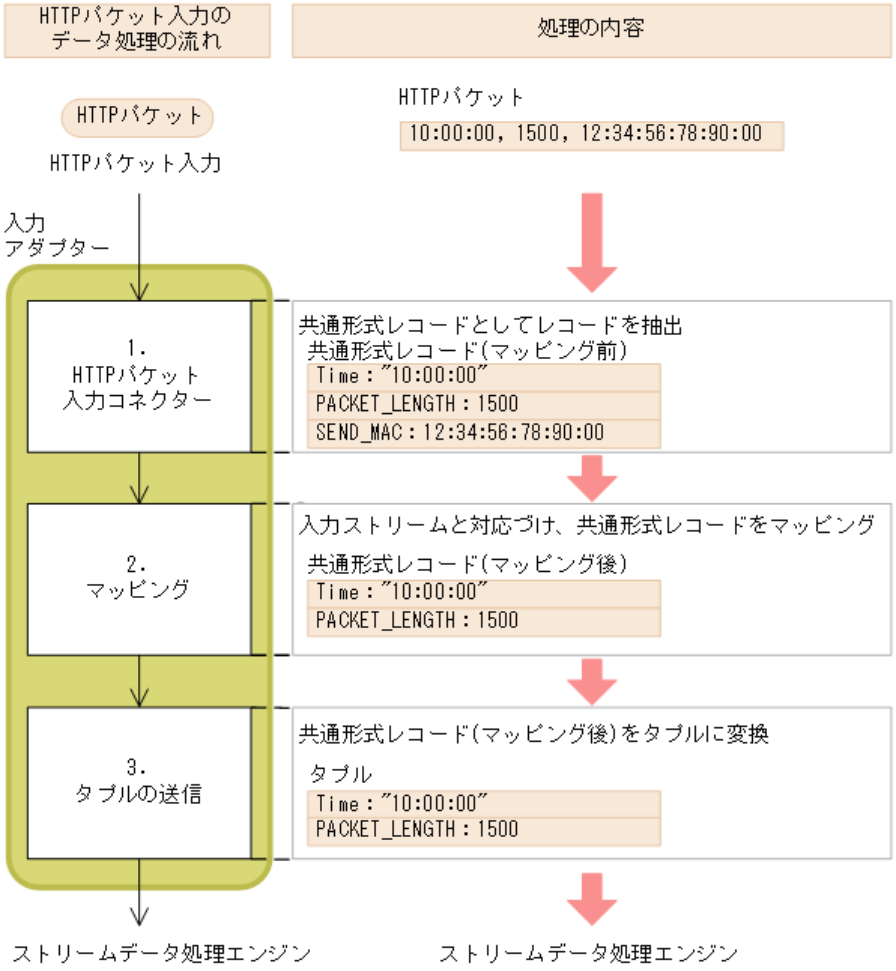
Pcap 形式のフォーマットでは，ネットワークを流れるパケット部分にPcap ヘッダーが付加されています。Pcap ヘッダーのうち，Pcap file header は，パケットアナライザー起動後，最初に作成されるグローバルヘッダーです。グローバルヘッダーが作成されたあと，パケットヘッダーであるPcap packet header とパケットデータであるPcap packet data を 1 組としたデータが作成され続けます。

Pcap 形式のフォーマット，またはそれ以外の HTTP パケットのフォーマットに関する情報は，アダプター構成定義ファイルの HTTP パケット入力コネクター定義に定義します。

16.4.2　HTTP パケットの入力のデータ処理の流れ

入力アダプターでの HTTP パケットの入力の，データ処理の流れと処理の内容を次の図に示します。

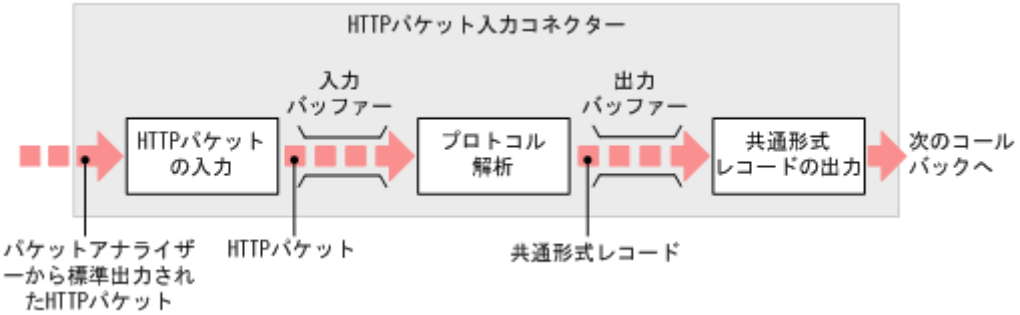
図 16-16　HTTP パケットの入力のデータ処理の流れと処理の内容



ここでは，HTTP パケット入力コネクターによる HTTP パケットの入力について説明します。マッピング，およびタプル送信については，「16.3.2　ファイルの入力のデータ処理の流れ」の「(1)　マッピング」および「(2)　タプル送信」を参照してください。入力アダプターで扱うデータ形式については，「16.3.2 ファイルの入力のデータ処理の流れ」の「入力アダプターで扱うデータ形式」を参照してください。

HTTP パケット入力コネクターでの HTTP パケット入力の処理の概要を次の図に示します。

図 16-17　HTTP パケット入力コネクターでの HTTP パケットの入力の処理の概要



1. HTTP パケットの入力

HTTP パケット入力コネクターでは，パケットアナライザーから標準出力に出力された HTTP パケットを標準入力で取得します。取得した HTTP パケットは入力バッファーに格納されます。

2. プロトコル解析

   格納されたデータのプロトコルを解析して，解析したプロトコルデータから共通形式レコードに変換します。なお，HTTP パケットの入力とプロトコル解析の処理は非同期で行われます。変換された共通形式レコードは出力バッファーに格納されます。

3. 共通形式レコードの出力

   出力バッファーに格納された共通形式レコードを次のコールバックに出力します。

16.4.3　HTTP パケットの入力の設定

HTTP パケットの入力について設定が必要な定義ファイルは，アダプター構成定義ファイルです。アダプター構成定義ファイルの入力アダプター定義で，次の CB 定義に設定します。

- 入力用 CB 定義

  HTTP パケット入力コネクター定義に，HTTP パケット入力コネクターの情報を定義します。定義の詳細については，「14.6.1　入力用 CB 定義」，および「14.7.2　HTTP パケット入力コネクター定義」を参照してください。

- 編集用 CB 定義

  マッピング定義に，マッピング情報を定義します。定義の詳細については，「14.6.3　編集用 CB 定義」，および「14.8.2　マッピング定義」を参照してください。

- 送信用 CB 定義

  入力ストリーム定義に，入力アダプターが接続する入力ストリームの情報を定義します。定義の詳細については，「14.6.4　送信用 CB 定義」，および「14.9.1　入力ストリーム定義」を参照してください。

16.5　レコードのフィルタリング

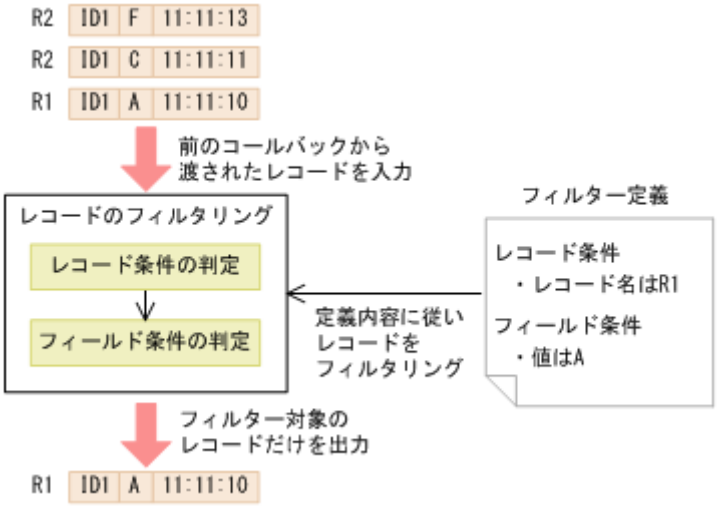
ここでは，レコードのフィルタリングの概要，データ処理の流れ，および必要な設定について説明します。

16.5.1　レコードのフィルタリングの概要

特定のレコードだけをストリームデータ処理の対象にしたい場合，編集用コールバックとして，フィルターを使用します。

レコードのフィルタリングの概要を次の図に示します。

図 16-18　レコードのフィルタリングの概要



レコードのフィルタリングでは，アダプター構成定義ファイルのフィルター定義で定義した内容に従って，入力されたレコードが条件に一致するかどうかを判定します。レコード形式，およびレコードに設定されているフィールド値から目的のレコードを取捨選択して出力します。レコードのフィルタリングの処理の詳細については，「16.5.2　レコードのフィルタリングのデータ処理の流れ」で説明します。

フィルターの入出力データの形式

フィルターの入出力データの形式は，共通形式レコードです。

フィルターを実施するタイミング

フィルターは，入力アダプター，または出力アダプターの編集用コールバックとして実施します。

入力アダプターの場合は，入力コネクターによるデータ入力またはフォーマット変換のあとにフィルターを実施します。出力アダプターの場合は，タプル受信後のマッピングのあとにフィルターを実施します。

16.5.2　レコードのフィルタリングのデータ処理の流れ

レコードのフィルタリングでは，レコード条件の判定，およびフィールド条件の判定を実施します。レコード条件の判定，およびフィールド条件の判定には，次の内容を指定します。

- レコード条件

  record タグで，フィルター対象のレコード名を指定します。

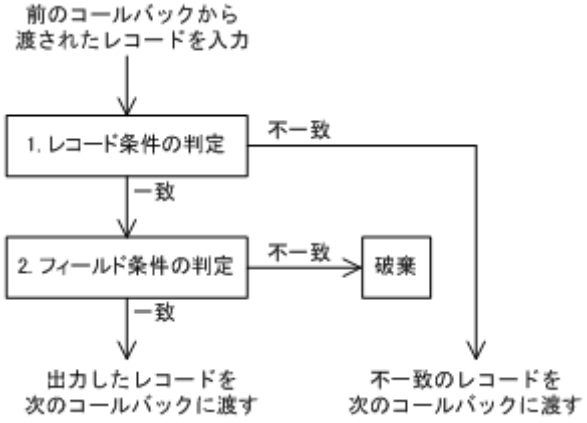
- フィールド条件

  field タグで，フィルター対象のレコードのフィールド名，比較演算記号，および条件値を指定します。

例えば，レコード名がR1，フィールド名がF1，フィールド値が100 よりも大きいという条件を指定したい場合には，レコード条件のレコード名には「source=”R1”」，フィールド条件には「source=”F1” condition=”gt” value=”100”」と指定します。

レコードのフィルタリングのデータ処理の流れを次の図に示します。

図 16-19　レコードのフィルタリングのデータ処理の流れ



1. レコード条件の判定

   入力されたレコードのレコード名が，レコード条件に指定したレコード名と一致するかどうかを判定します。

   - レコード条件に一致したレコード

     フィールド条件の判定の処理へ進みます。

   - レコード条件に一致しなかったレコード

     一致しなかったレコードは，そのまま出力されて，次のコールバックに渡されます。

2. フィールド条件の判定

   レコード条件に一致したレコードのフィールド値が，フィールド条件に指定したフィールド値と一致するかどうかを判定します。

   - フィールド条件に一致したレコード

     レコード条件，およびフィールド条件が成立したレコードが出力されます。出力されたレコードは，次のコールバックに渡されます。

- フィールド条件に一致しなかったレコード
  一致しなかったレコードは，そこで破棄されます。

16.5.3　レコードのフィルタリングの設定

レコードのフィルタリングについて設定が必要なファイルは，アダプター構成定義ファイルです。

アダプター構成定義ファイルの入力アダプター定義，または出力アダプター定義で，次の CB 定義に設定します。

- 編集用 CB 定義
  フィルター定義に，レコードをフィルタリングするためのレコード条件とフィールド条件を定義します。定義の詳細については，「14.6.3　編集用 CB 定義」，および「14.8.3　フィルター定義」を参照してください。

16.6　レコードの抽出

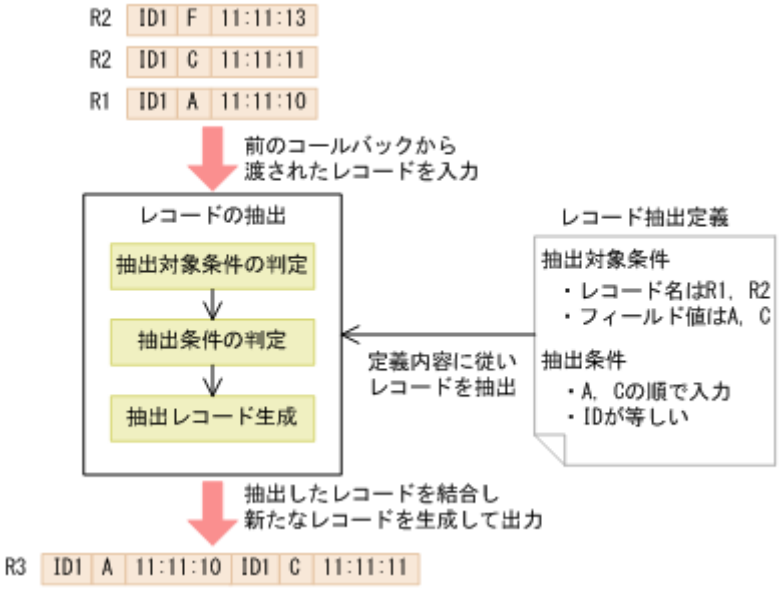
ここでは，レコードの抽出の概要，データ処理の流れ，および必要な設定について説明します。

16.6.1　レコードの抽出の概要

特定のレコードを抽出したあと，それらのレコードを意味のある一つのレコードにする場合，編集用コールバックとして，レコード抽出を使用します。

レコードの抽出の概要を次の図に示します。

図 16-20　レコードの抽出の概要



レコードの抽出では，アダプター構成定義ファイルのレコード抽出定義で定義した内容に従って，入力されたレコードが条件に一致するかどうかを判定します。レコード形式，レコードに設定されているフィールド値，レコードの入力順序などから目的のレコードを抽出したあと，それらのレコードを結合して新たなレコードを生成して出力します。レコードの抽出で，レコードを結合して新たに生成するレコードのことを抽出レコードといいます。レコードの抽出の処理の詳細については，「16.6.2　レコードの抽出のデータ処理の流れ」で説明します。

レコード抽出の入出力データの形式

レコード抽出の入出力データの形式は，共通形式レコードです。

レコード抽出を実施するタイミング

レコード抽出は，入力アダプターの編集用コールバックとして実施します。HTTP パケット入力コネクターによるデータ入力またはフォーマット変換のあとにレコード抽出を実施します。

なお，出力アダプターでは，レコード抽出は使用できません。

レコード抽出の使用例

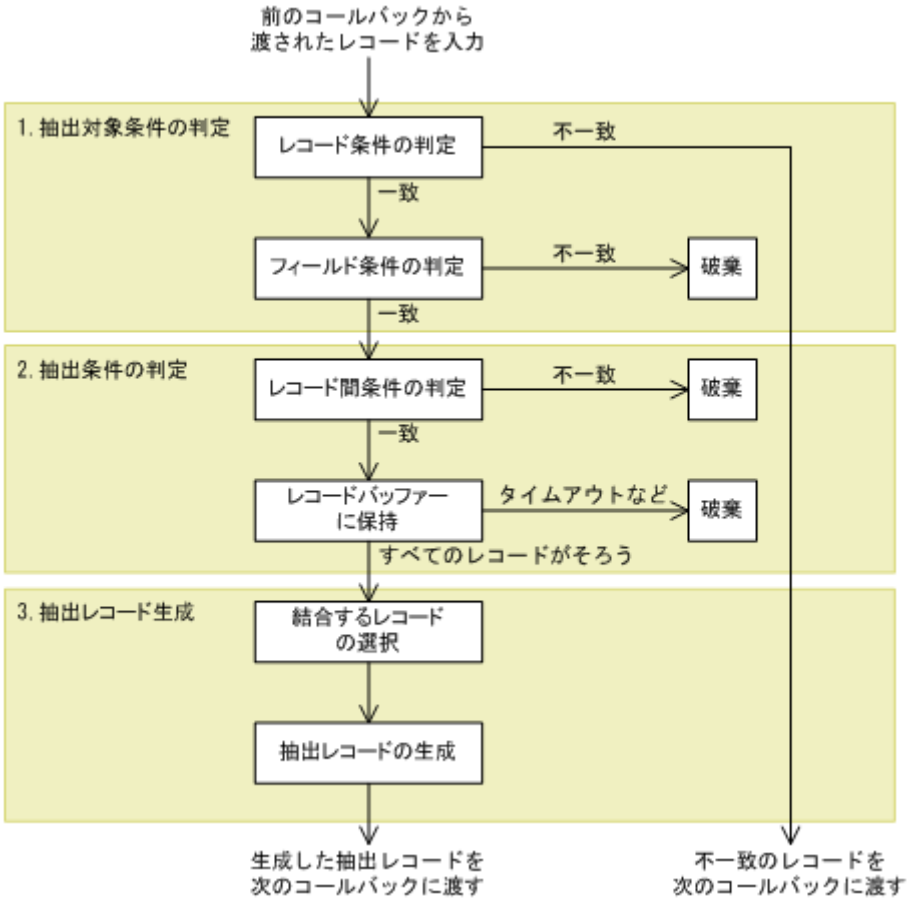
例えば，「16.4　HTTP パケットの入力」で説明した HTTP パケット入力コネクターとレコード抽出を組み合わせた場合には，HTTP パケットの要求から応答までの時間を測定できます。この場合，レコード抽出では，送信元 IP アドレスとポート番号，および送信先 IP アドレスとポート番号が等しいリクエストとレスポンスのパケットのレコードを抽出し，それらのレコードを対にして結合します。結合されたレコードで，リクエストとレスポンスのパケットの時刻を比較することで，要求から応答までの時間を測定できます。

また，結合されたレコードに対して，さらにレコードを抽出して，指定した URL のレコードだけを抽出したあと，指定した順序で URL が遷移しているレコードを結合します。これによって，HTTP パケットの URL からユーザーのサイト遷移状況を監視することもできます。

16.6.2　レコードの抽出のデータ処理の流れ

レコードの抽出では，レコードを抽出して，抽出したレコードを結合して新たなレコードを生成するまでに，抽出対象条件の判定，抽出条件の判定，および抽出レコード生成の処理を実施します。レコードの抽出のデータ処理の流れを次の図に示します。

図 16-21　レコードの抽出のデータ処理の流れ



図中の 1.～3.の処理の詳細について，以降で説明します。なお，説明に出てくるアダプター構成定義ファイルのレコード抽出定義のタグや属性の詳細については，「14.8.4　レコード抽出定義」を参照してください。

(1)　抽出対象条件の判定

入力されたレコードに対して，抽出対象条件の判定をします。

抽出対象条件とは，抽出対象のレコードを取捨選択するための条件の一つです。抽出対象条件は，レコード抽出定義のtargetrecord タグで指定します。

抽出対象条件には，レコード条件とフィールド条件を指定します。

- レコード条件
  record タグで，抽出対象のレコード名を指定します。
- フィールド条件
  field タグで，抽出対象のレコードのフィールド名，比較演算記号，および条件値を指定します。

例えば，レコード名がR1，フィールド名がF1，フィールド値がTARO という条件を指定したい場合には，レコード条件のレコード名には「source=”R1”」，フィールド条件には「source=”F1” condition=”eq” value=”TARO”」と指定します。

抽出対象条件の判定では，レコード抽出定義で定義した抽出対象条件に従って，レコード条件の判定，およびフィールド条件の判定が行われます。抽出対象条件の判定の流れを次に示します。

1. レコード条件の判定
   入力されたレコードのレコード名が，レコード条件に指定したレコード名と一致するかどうかを判定します。
   - レコード条件に一致したレコード
     フィールド条件の判定の処理へ進みます。
   - レコード条件に一致しなかったレコード
     一致しなかったレコードは，そのまま出力されて，次のコールバックに渡されます。
2. フィールド条件の判定
   レコード条件に一致したレコードのフィールド値が，フィールド条件に指定したフィールド値と一致するかどうかを判定します。
   - フィールド条件に一致したレコード
     抽出対象条件が成立して，抽出条件の判定の処理へ進みます。抽出条件の判定の処理については，「16.6.2　レコードの抽出のデータ処理の流れ」の「(2)　抽出条件の判定」を参照してください。
   - フィールド条件に一致しなかったレコード
     一致しなかったレコードは，そこで破棄されます。

(2)　抽出条件の判定

抽出対象条件が成立したレコードに対して，抽出条件の判定をします。

抽出条件とは，抽出対象のレコードを取捨選択するための条件の一つです。抽出条件は，レコード抽出定義のextraction タグで指定します。

抽出条件には，レコードバッファーに保持するレコードの上限値や，レコード間条件を指定します。レコード間条件には，レコードの入力順序と，レコード間でフィールド値を比較するためのフィールド値を指定します。レコード間条件は，targets タグで指定します。

抽出条件の判定では，レコード抽出定義で定義した抽出条件に従って，レコード間条件の判定が行われます。レコード間条件に一致したレコードは，抽出条件が成立するまでレコードバッファーに保持されます。抽出条件の判定の流れを次に示します。

1. レコード間条件の判定

   抽出対象条件が成立したレコードに対して，レコード間条件で指定した次の内容と一致するかどうかを判定します。

   - レコードの入力順序が一致するかどうか。※
   - レコード間のフィールド値が一致するかどうか。※

   注※

   レコード内の時刻情報が同一である複数のレコードが入力された場合，抽出条件はレコードの入力順序で判定します。

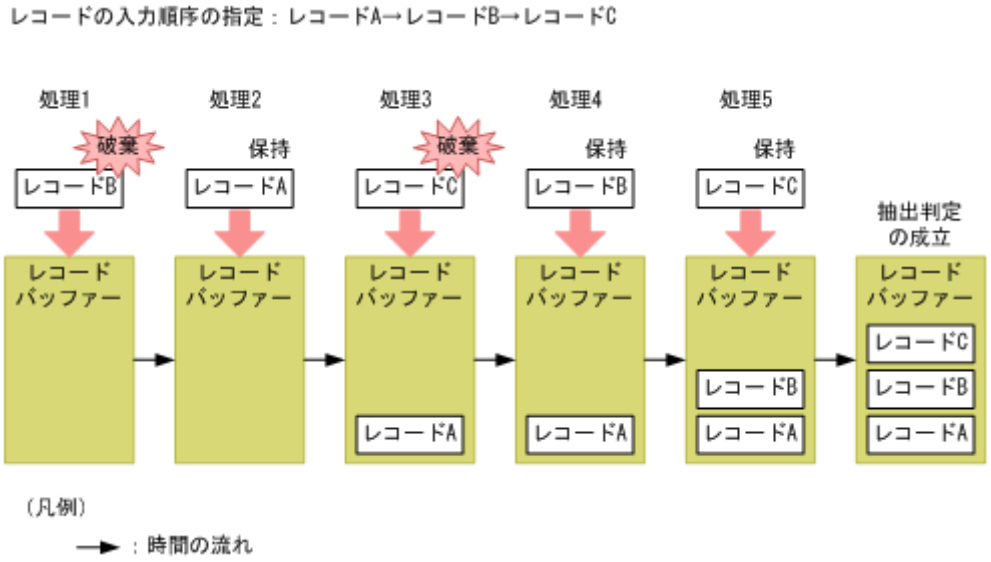
   これらの条件に一致したレコードはレコードバッファーに保持され，一致しなかったレコードはそこで破棄されます。

2. レコードバッファーに保持

   レコード間条件に一致したレコードは，レコードバッファーに保持され，レコード間条件で指定したすべてのレコードがそろって抽出条件が成立すると，抽出レコード生成の処理へ進みます。抽出レコード生成の処理については，「16.6.2　レコードの抽出のデータ処理の流れ」の「(3)　抽出レコード生成」を参照してください。

レコードの入力順序を指定している場合のレコード保持と破棄の流れの例を次の図に示します。

図 16-22　レコードの入力順序を指定している場合のレコード保持と破棄の流れの例



この例では，レコードの入力順序をレコードA→レコードB→レコードC と指定しています。このため，入力されたレコードは次のように保持，または破棄されます。

- 処理 1：レコードバッファー内にレコードがない状態で入力されたレコードB は，入力順序に一致しないため破棄されます。
- 処理 2：レコードバッファー内にレコードがない状態で入力されたレコードA は，入力順序に一致するため保持されます。
- 処理 3：レコードA のあとに入力されたレコードC は，入力順序に一致しないため破棄されます。
- 処理 4：レコードA のあとに入力されたレコードB は，入力順序に一致するため保持されます。
- 処理 5：レコードA とレコードB のあとに入力されたレコードC は，入力順序に一致するため保持されます。レコードA→レコードB→レコードC の順序でレコードが入力され，すべてのレコードがそろうため，ここで抽出条件が成立します。

抽出条件の判定では，レコード内の時刻情報が逆転した場合や，レコードバッファーに保持しているレコードがタイムアウトした場合などにもレコードを破棄します。それぞれの内容について，次に説明します。

- レコード内の時刻情報が逆転した場合

  レコード内の時刻情報の逆転が発生したレコードが入力された場合には，レコードを破棄します。
- レコードバッファーに保持しているレコードがタイムアウトした場合

  レコードの抽出では，レコードバッファーに保持しているレコードがタイムアウトする条件として，レコードの生存時間を指定できます。レコードの生存時間は，extraction タグのtimelimit 属性で指定します。

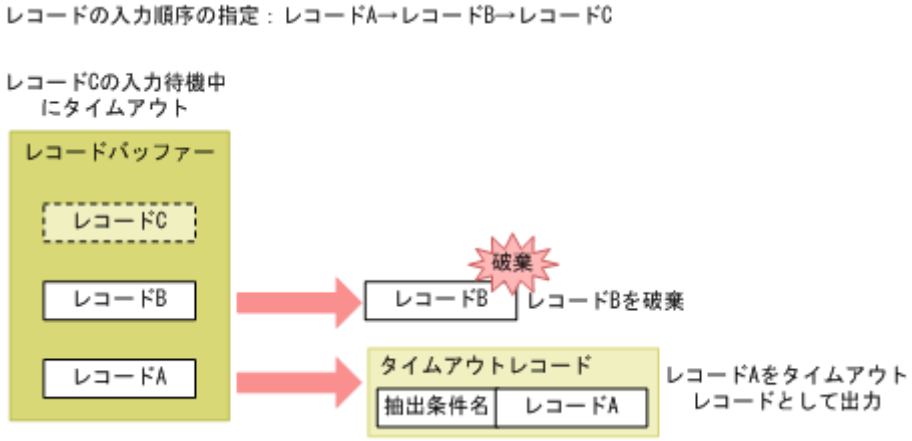
  指定した生存時間が経過しても抽出条件が成立していない場合には，タイムアウトと見なして，レコードバッファーに保持されていたレコードを破棄します。

  このとき，extractions タグのtimeout 属性でタイムアウトレコードの出力を指定していると，レコードバッファーに保持されていたレコードのうち，最初のレコードだけを出力できます。

タイムアウト時に出力されるレコードのことをタイムアウトレコードといいます。タイムアウトレコードには，extraction タグのname 属性で指定した抽出条件名が入力されたフィールドが追加されます。

タイムアウト時のレコードの処理の流れを次の図に示します。

図 16-23　タイムアウト時のレコードの処理の流れ



この例の条件は次のとおりです。

- レコードA，レコードB，レコードC の 3 つがそろうと，抽出条件をすべて満たして，抽出レコード生成の処理へ進むことができる。
- タイムアウトした場合には，タイムアウトレコードを出力する。
- レコードC の入力待機中にタイムアウトした。

このため，レコードB は破棄され，レコードA はタイムアウトレコードとして出力されます。

- レコードバッファーに保持するレコード数の上限値を超えた場合

  レコードの抽出では，レコードバッファーに保持するレコード数を管理するために，保持するレコード数の上限値を指定できます。保持するレコード数の上限値は，extractions タグのsize 属性で指定します。上限値を超えた場合には，レコードバッファーに保持しているすべてのレコードを破棄します。

- 同一レコードが入力された場合

  レコードの抽出では，同一の抽出対象条件を満たし，かつレコード間条件に指定したフィールドの値が等しいレコードは，同一レコードと見なします。同一レコードが入力された場合には，extractions タグのsamerecord 属性で指定した内容に従って，次の処理が行われます。

  samerecord 属性でoverwrite を指定した場合

  同一レコードと見なされた既存のレコードが，レコードバッファーのどの位置にあるかによって処理が異なります。

  最後のレコードの場合，入力されたレコードで，レコードバッファーに保持していた既存のレコードを上書きします。

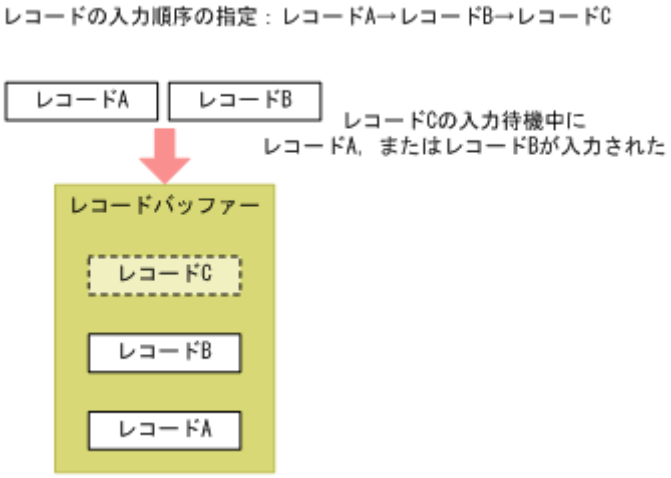
  ほかのレコードの場合，レコードバッファーに保持していた既存のレコードをすべて破棄し，入力されたレコードを保持します。

  samerecord 属性でdelete を指定した場合

  入力されたレコードを破棄します。

次の図に示す例を使用して，同一レコードが入力された場合の処理について説明します。

図 16-24　同一レコードが入力された場合の処理



この例の条件は次のとおりです。

- レコードA，レコードB，レコードC の 3 つがそろうと，抽出条件をすべて満たして，抽出レコード生成の処理へ進むことができる。
- samerecord 属性でoverwrite を指定している。
- レコードC の入力待機中に，レコードA，またはレコードB が入力された。

この条件の場合，レコードA とレコードB のどちらが入力されるかによって，次のように処理が異なります。

- レコードA の場合

  レコードバッファーに保持していた既存のレコードA とレコードB を破棄し，入力されたレコードA を保持します。
- レコードB の場合

  入力されたレコードB で，レコードバッファーに保持していた既存のレコードB を上書きします。

(3)　抽出レコード生成

抽出レコード生成では，抽出条件が成立したレコードを結合して，抽出レコードを生成します。抽出レコード生成の流れを次に示します。

1. 結合するレコードの選択

   抽出条件が成立したレコードの中から，抽出レコードとして結合するレコードを選択します。結合するレコードは，select タグのsource 属性で指定します。
2. 抽出レコードの生成

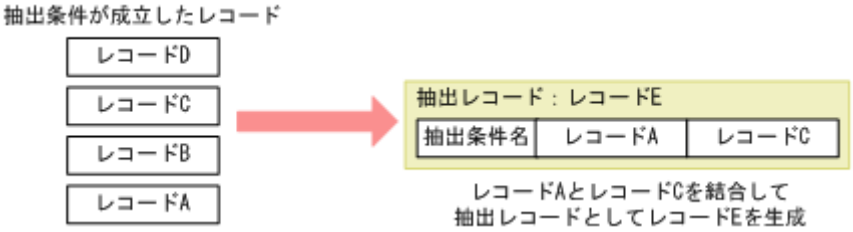
   手順 1.で選択したレコードを結合して，抽出レコードを生成します。

生成される抽出レコードには，extraction タグのname 属性で指定した抽出条件名が入力されたフィールドが追加されます。

- 抽出条件名が入力されるフィールド名：「condition」
- ほかのフィールド：「*<抽出対象レコード名>*_*<フィールド名>*」

抽出レコードの例を次の図に示します。

図 16-25　抽出レコードの例



この例では，レコードA とレコードC を結合して，抽出レコードとして，新たなレコードE が生成されます。

16.6.3　レコードの抽出の設定

レコードの抽出について設定が必要なファイルは，アダプター構成定義ファイルです。

アダプター構成定義ファイルの入力アダプター定義で，次の CB 定義に設定します。

- 編集用 CB 定義
  レコード抽出定義に，レコードを抽出するための抽出対象条件と抽出条件などを定義します。定義の詳細については，「14.6.3　編集用 CB 定義」，および「14.8.4　レコード抽出定義」を参照してください。

16.7　クエリグループまたは外部出力アダプターへの出力

ここでは，クエリグループまたは外部出力アダプターへの出力の概要，データ処理の流れ，および定義ファイルでの設定内容について説明します。

16.7.1　クエリグループまたは外部出力アダプターへの出力の概要

Streaming Data Platform では，内部標準アダプターの 1 つにカスケーディングアダプターを提供しています。カスケーディングアダプターは，クエリグループ間を接続するための TCP データ入力アダプター，または外部出力アダプター宛てに，TCP ソケットを介してデータを送信します。

カスケーディングアダプターは，SDP ブローカー連携によって自動的に生成・起動することができます。

クエリグループ間のデータ送信プロセスに，内部カスタムアダプターは使用できません。

カスケーディングアダプターがデータを受け取るストリーム（出力ストリーム）が Java ストリームの場合，Java 用カスケーディングアダプターを使用し，C ストリームの場合は，C 用カスケーディングアダプターを使用します。Java/C の指定は，アダプター構成定義ファイルの`OutputAdaptorDefinition` タグの `language` パラメータで行います。

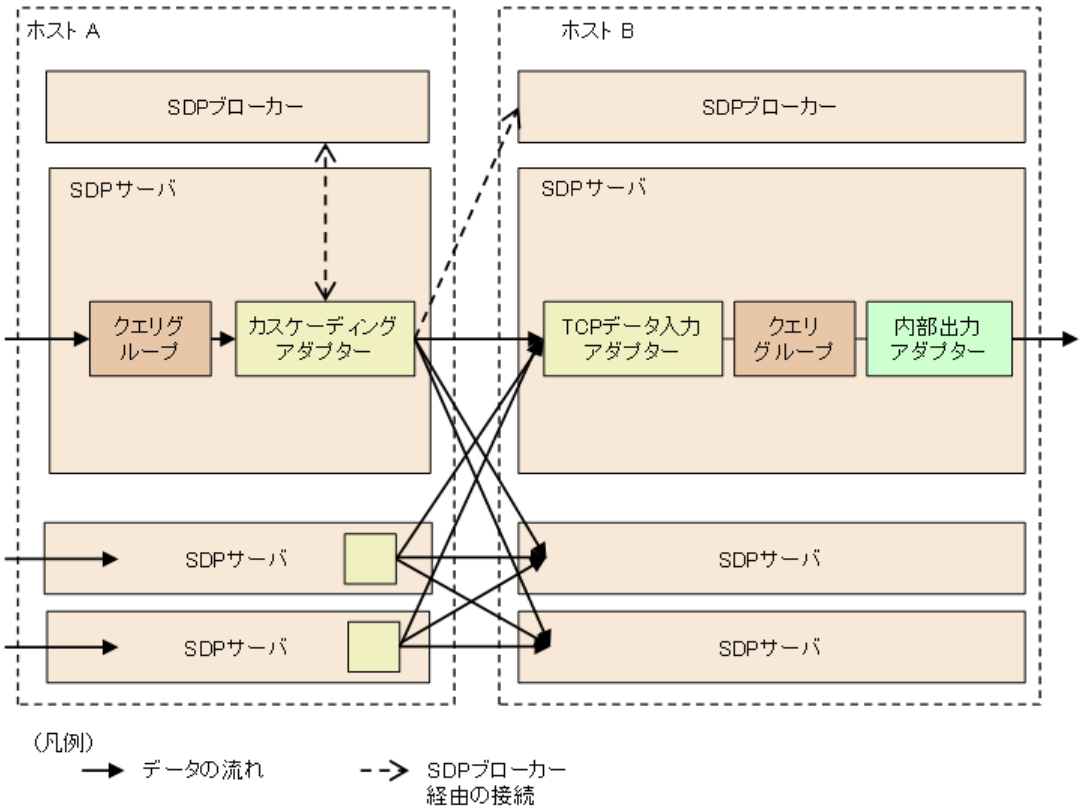
(1)　クエリグループへのデータ送信

ほかのクエリグループのストリームへデータを送信する場合，カスケーディングアダプターと TCP データ入力アダプターの接続が必要になります。

次の図は，ホスト A とホスト B 内のクエリグループを接続するカスケーディングアダプターの構成を示しています。カスケーディングアダプターは，ホスト A の出力ストリームからホスト B の入力ストリームに，TCP データ入力アダプターを介してデータを送信します。接続された入力および出力ストリームについての情報は，送信元のクエリグループ用プロパティファイルで定義されます。カスケーディングアダプターは，対応するクエリグループ（この場合，ホスト A のクエリグループ）の起動と同時に自動起動します。起動後，カスケーディングアダプターは，ホスト A の SDP ブローカーに接続するストリームのアドレスを問い合わせて取得し，その後ホスト B の SDP ブローカーを介して接続を確立します。

なお，1 つの SDP サーバに複数のクエリグループを登録し，そのクエリグループ間でデータを送信する場合も，同様にカスケーディングアダプターと TCP データ入力アダプターが必要になります。

図 16-26　クエリグループへのデータ送信

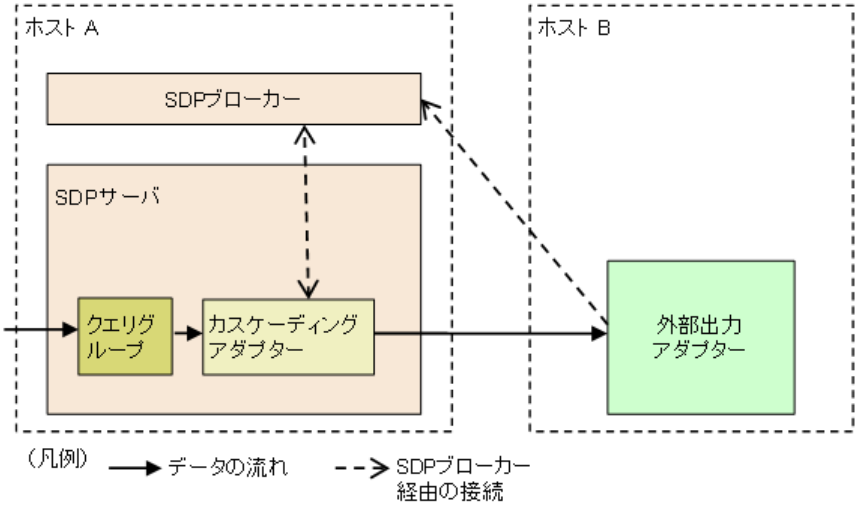


(2)　外部出力アダプターへのデータ送信

次の図は，外部出力アダプターに接続するカスケーディングアダプターの構成を示しています。外部出力アダプターが TCP 接続を要求すると，SDP ブローカー連携によってカスケーディングアダプターが自動起動します。起動後，カスケーディングアダプターは確立された接続を使用して外部出力アダプターにデータを送信します。

この構成では，アダプター構成定義ファイルの作成を省略できます。

図 16-27　外部出力アダプターへのデータ送信



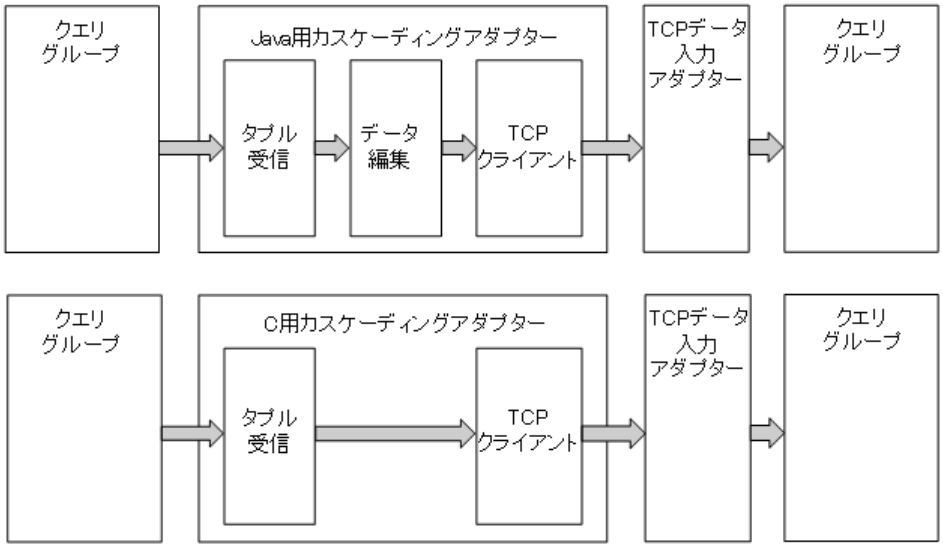
16.7.2　クエリグループまたは外部出力アダプターへのデータ処理の流れ

ここでは，カスケーディングアダプターの処理概要，および通信方法について説明します。

(1)　カスケーディングアダプターの処理概要

カスケーディングアダプターは以下のコールバックから構成されます。

図 16-28　カスケーディングアダプターの構成



コールバック

タプル受信

　このコールバックは，クエリグループのストリームサーバからタプルを受信します。

データ編集

　このコールバックは，受信したデータのフォーマットを変更します。このコールバックは省略できます。

TCP クライアント

　このコールバックは，TCP データ入力アダプターを介して，送信先ストリームにタプルを入力します。
　ソケットオプションTCP_NODELAY が有効である場合は，データが即時に送信されます。

コールバックの組み合わせ

次の表に，カスケーディングアダプターにおける，コールバックの組み合わせを示します。

表 16-7　コールバックの組み合わせ一覧

アダプタータイプ	コールバックの組み合わせ		
	受信用コールバック	編集用コールバック	出力用コールバック
Java 用カスケーディングアダプター	受信用コールバック	編集用コールバックを設定可能。または省略可能。	カスケーディングコールバック（TCP クライアント）

アダプタータイプ	コールバックの組み合わせ		
	受信用コールバック	編集用コールバック	出力用コールバック
C 用カスケーディングアダプター	受信用コールバック	-	カスケーディングコールバック（TCP クライアント）

16.7.3　カスケーディングアダプターの通信方法

カスケーディングアダプターを使用する場合は，クエリグループ間の通信に TCP ソケットを使用します。

通信方法の詳細を次の表に示します。

表 16-8　カスケーディングアダプターの通信方法の詳細

項目	説明
接続性※1	カスケーディングアダプターを HSDP サーバのストリームデータ処理エンジンで実行している TCP データ入力アダプターに接続できます。
送信されるタプル内で利用可能なデータ型	利用できるデータ型を次に示します。 • INT • SHORT • BYTE • LONG • BIG_DECIMAL • FLOAT • DOUBLE • STRING※2 • DATE • TIME • TIMESTAMP データ型の詳細については，「表 14-14　type 属性の指定値と，対応する Java のデータ型および CQL のデータ型」を参照してください。
TCP ポート	**宛先ポート** • SDP ブローカーを使用する場合 カスケーディングアダプターは，宛先ホストのブローカーのスタンバイポート番号を使用します。 • SDP ブローカーを使用しない場合 カスケーディングアダプターは，cascading 用プロパティファイルに定義されている接続先ポート番号を使用します。 **送信元ポート** カスケーディングアダプターは，各宛先にランダムに割り当てられる TCP ポートを使用します。
注※1 詳細については，「表 16-9　カスケーディングアダプターの接続の詳細」を参照してください。	

項目	説明
注※2 Java 用のカスケーディングアダプターは，データを送信する前に，STRING データをカスケーディングクライアントコネクター定義で指定された特定の文字コードのバイト配列に変換します。STRING データを指定した文字コードのバイト配列に変換できない場合，アダプターはタプルを破棄します。	

16.7.4　カスケーディングアダプターの接続の詳細

カスケーディングアダプターと TCP データ入力アダプターに，それぞれ Java アダプターまたは C アダプターを使用した場合の接続の互換性を次に示します。

表 16-9　カスケーディングアダプターの接続の詳細

項目			カスケーディングアダプター	
			Java 用	C 用
接続先	標準提供アダプター	Java 用 TCP データ入力アダプター	○	○
		C 用 TCP データ入力アダプター	○	○
凡例： ○：接続可能				

16.7.5　カスケーディングアダプターの使用方法

カスケーディングアダプターは，次の 2 つのケースで使用できます。

- TCP データ入力アダプターと接続してクエリグループ間でデータを送信する。
- 外部出力アダプターと接続してデータを送信する。

TCP データ入力アダプターとの接続

カスケーディングアダプターは，TCP データ入力アダプターと接続してほかのクエリグループにデータを送信できます。

カスケーディングアダプターと接続先 TCP データ入力アダプターの起動方法によって，データ送信に必要な宛先の指定方法が異なります。それぞれのアダプターの起動方法の組み合わせは次のとおりです。

表 16-10　カスケーディングアダプターと接続先 TCP 入力アダプターの起動方法の組み合わせ

組み合わせのパターン	カスケーディングアダプターの起動方法	接続先 TCP データ入力アダプターの起動方法
組み合わせ 1	hsdpstartinpro コマンドによる手動起動	hsdpstartinpro コマンドによる手動起動

組み合わせのパターン	カスケーディングアダプターの起動方法	接続先 TCP データ入力アダプターの起動方法
組み合わせ 2	hsdpstartinpro コマンドによる手動起動	SDP ブローカー連携による自動起動
組み合わせ 3	SDP ブローカー連携による自動起動	SDP ブローカー連携による自動起動

これらの組み合わせごとの，SDP ブローカー起動要否およびカスケーディングアダプターの宛先指定方法を次に示します。

表 16-11　カスケーディングアダプターと TCP データ入力アダプターの起動方法ごとの，SDP ブローカー起動要否および宛先指定方法

SDP ブローカーの起動要否およびカスケーディングアダプターの宛先指定方法		カスケーディングアダプターと TCP データ入力アダプターのアダプターの起動方法の組み合わせ		
		組み合わせ 1	組み合わせ 2	組み合わせ 3
SDP ブローカーの起動要否		必要	必要	不要
カスケーディングアダプターの宛先指定方法	データ送信元のクエリグループ用プロパティファイルの stream.output.*出力ストリーム名*.link	－	－	接続先 TCP データ入力アダプターが接続する宛先入力ストリームの情報を指定する。
	カスケーディングアダプターのアダプター構成定義ファイルの targetStream 要素	接続先 TCP データ入力アダプターが接続する宛先入力ストリームの情報を指定する。	接続先 TCP データ入力アダプターが接続する宛先入力ストリームの情報を指定する。	－
	送信元の cascading 用プロパティファイルの destination 要素	接続先 TCP データ入力アダプターの IP アドレス／ポート番号を指定する。	－	－

（凡例）

－：該当しません。

クエリグループ用プロパティファイルの詳細については「12.9　クエリグループ用プロパティファイル」を参照してください。

アダプター構成定義ファイルの詳細については，「14.　アダプター構成定義ファイル」を参照してください。

なお，カスケーディングアダプターの宛先がデータパラレル構成の場合，宛先入力ストリーム情報を指定する際のクエリグループ名指定方法に注意する必要があります。データパラレル構成の詳細については，マニュアル『Hitachi Streaming Data Platform 概説』を参照してください。

外部アダプターとの接続

カスケーディングアダプターは，外部出力アダプターと接続して外部出力アダプターにデータを送信できます。

外部出力アダプターが SDP ブローカーを介してカスケーディングアダプターに接続を要求すると，カスケーディングアダプターが SDP ブローカー連携によって自動起動します。

手動で起動したカスケーディングアダプターに外部アダプターは接続できません。このため，外部アダプターを利用するためには SDP ブローカーの起動が必要です。

外部出力アダプターとカスケーディングアダプターを接続するには，外部アダプター定義ファイルの `hsdp.target.name.n` プロパティに，カスケーディングアダプターが接続している出力ストリームの情報を指定します。

外部アダプター定義ファイルの詳細については，「12.15　外部アダプター定義ファイル」を参照してください。

16.7.6　カスケーディングアダプターの負荷分散方式

カスケーディングアダプターは，パラレル構成をとる複数の SDP サーバの TCP データ入力アダプターに接続できます。

複数の宛先に接続した場合，カスケーディングアダプターは，アダプター構成定義ファイルまたは接続先のクエリグループ用プロパティファイルに指定された負荷分散方式に基づいて，出力ストリームに出力されたデータの宛先を決めます。

アダプター構成定義ファイルの詳細については，「14.　アダプター構成定義ファイル」を参照してください。

クエリグループ用プロパティファイルの詳細については「12.9　クエリグループ用プロパティファイル」を参照してください。

指定できる負荷分散方式を次の表で説明します。どれか 1 つのタイプだけを使用できます。

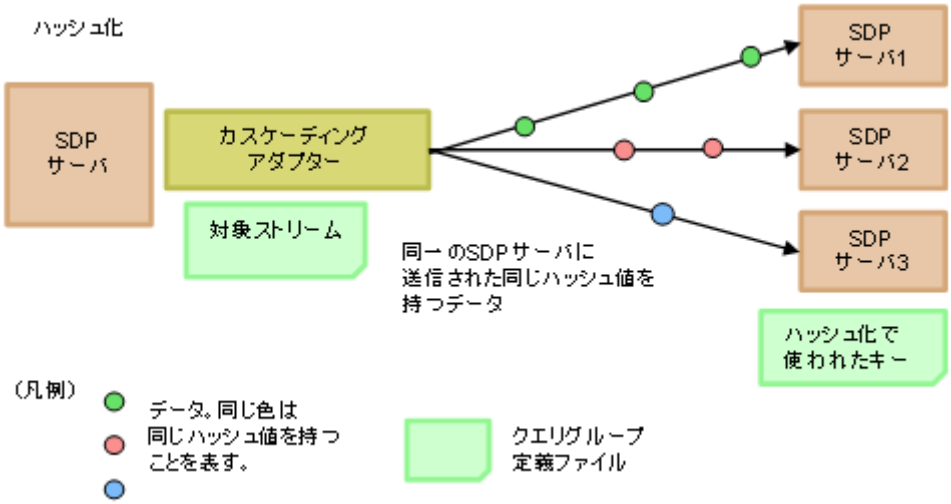
表 16-12　負荷分散方式

負荷分散方式	説明	図
ハッシュ化	各種データの宛先は，タプル内のカラムデータのハッシュ値によって決まります。	「図 16-29　ハッシュ化の概要」
ラウンドロビン	データはラウンドロビンによって SDP サーバに配信されます。	「図 16-30　ラウンドロビンの概要」
スタティックルール	各種データの宛先は，ユーザーが定義したスタティックルールによって決まります。このルールは，タプル内のカラムの値ごとに指定できます。例えば，タプルに「ID」カラムがある場合，これを指定して，ID="port1"のデータが SDP サーバ 1 に，ID="port2"のデータが SDP サーバ 2 に送信されるようにします。タプルがどのスタティックルールにも一致しない場合，アダプターはそのタプルを破棄し，ログファイルにメッセージを出力します。	「図 16-31　スタティックルールの概要」

負荷分散方式	説明	図
すべて	同等のデータが，すべての宛先 SDP サーバに送信されます。	「図 16-32 「すべて」の概要」

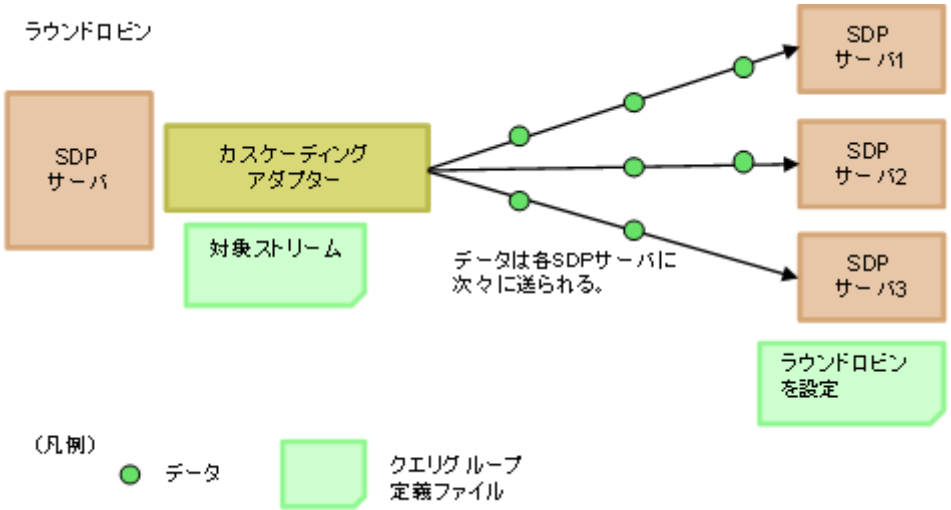
ハッシュ化

図 16-29　ハッシュ化の概要



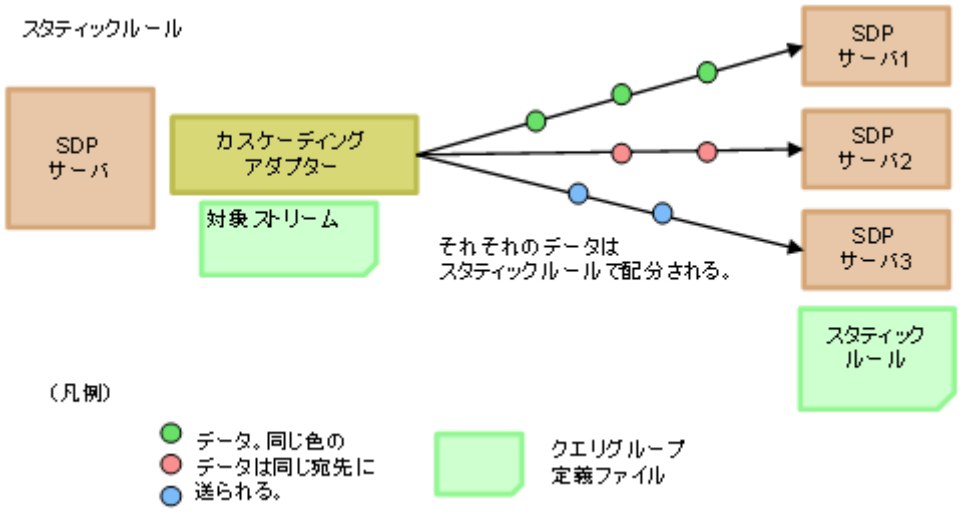
ラウンドロビン

図 16-30　ラウンドロビンの概要



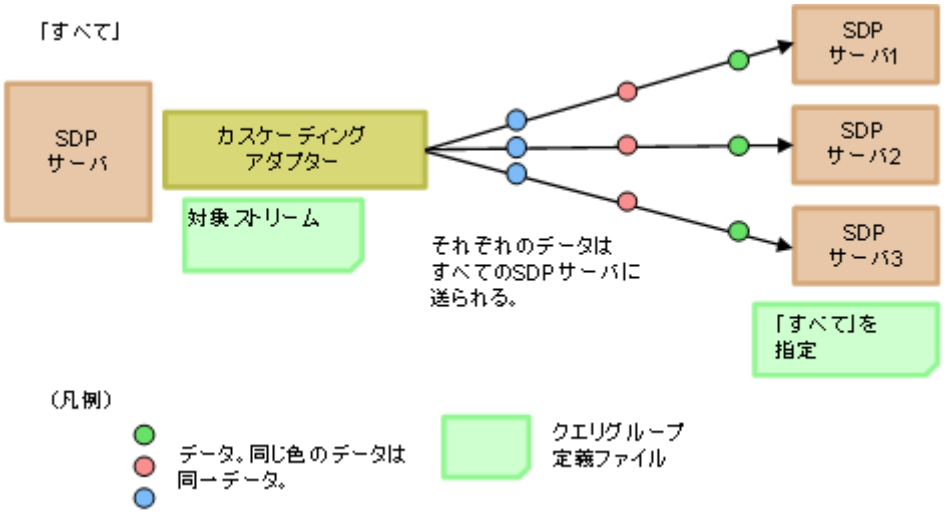
スタティックルール

図 16-31　スタティックルールの概要



「すべて」

図 16-32　「すべて」の概要



アダプターの起動方法ごとの負荷分散方式の指定方法は，次のとおりです。

表 16-13　データ送信先のクエリグループの構成およびアダプターの起動方法ごとの負荷分散方式の指定方法

アダプターの起動方法	データ送信先のクエリグループの構成	
	スケールアップ構成	スケールアウト構成
カスケーディングアダプター：hsdpstartinpro コマンドによる手動起動 送信先 TCP 入力アダプター：hsdpstartinpro コマンドによる手動起動	カスケーディングアダプターのアダプター構成定義ファイルに負荷分散方式を指定します。デフォルトはラウンドロビンです。データ送信先クエリグループのクエリグループ用プロパティファイルで指定された負荷分散方式は無視されます。	

アダプターの起動方法	データ送信先のクエリグループの構成	
	スケールアップ構成	スケールアウト構成
カスケーディングアダプター：hsdpstartinpro コマンドによる手動起動 送信先 TCP 入力アダプター：SDP ブローカー連携による自動起動	データ送信先クエリグループのクエリグループ用プロパティファイルに負荷分散方式を指定します。カスケーディングアダプターのアダプター構成定義ファイルには指定できません。	
カスケーディングアダプター：SDP ブローカー連携による自動起動 送信先 TCP 入力アダプター：SDP ブローカー連携による自動起動	データ送信先クエリグループのクエリグループ用プロパティファイルに負荷分散方式を指定します。カスケーディングアダプターのアダプター構成定義ファイルには指定できません。	

16.7.7 複数のカスケーディングアダプターから同じ入力ストリームにデータを送信する構成

カスタムデベロッパーが複数のカスケーディングアダプターを使用しており，SDP サーバの同じ入力ストリームにデータを送信する場合，データはタイムスタンプ順に並んでいる必要があります。同じ入力ストリームにデータを送信するには，次の 2 つの構成を利用できます。

表 16-14 複数のカスケーディングアダプターの構成

構成	説明
「図 16-33 例 1」	複数の出力ストリームが 1 つの入力ストリームに接続します。 この構成は，レイテンシの増加が許可される場合に有効です。 時系列データを分析する場合，入力タプルは SDP サーバ 1 でタイムスタンプ順に並んでいる必要があります。
「図 16-34 例 2」	複数の出力ストリームが複数の入力ストリームに接続します。 出力ストリームと入力ストリームの数は同じです。 この構成は，レイテンシの増加が許可されない場合に有効です。 時系列データを分析する場合，SDP サーバ 2 で UNION クエリが実行される時に，入力タプルがタイムスタンプ順に並んでいる必要があります。 ストリームの 1 つが停止すると，図中の SDP サーバ 2 は停止します。 SDP サーバ 2 が acceleration CQL エンジンの場合，この構成は使用できません。

図 16-33　例 1

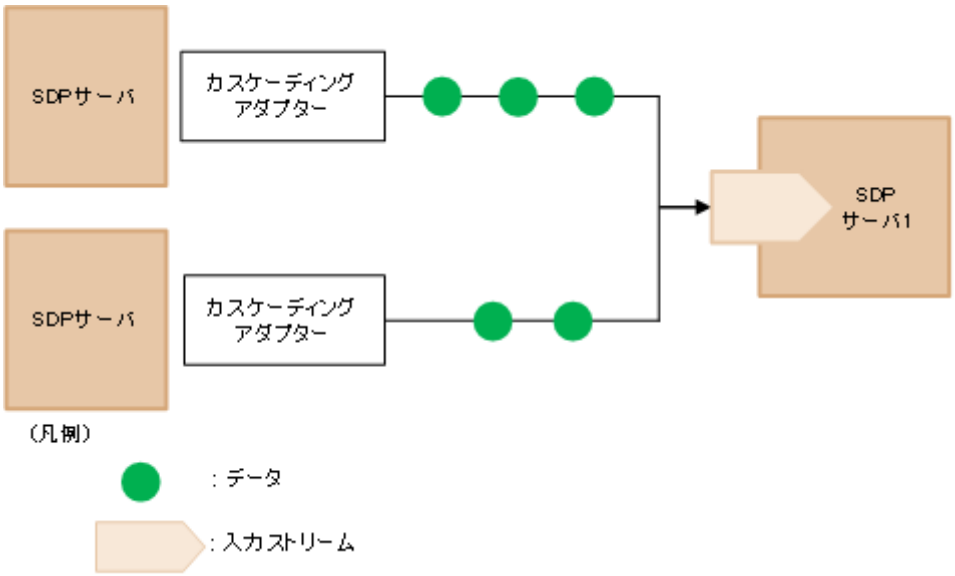
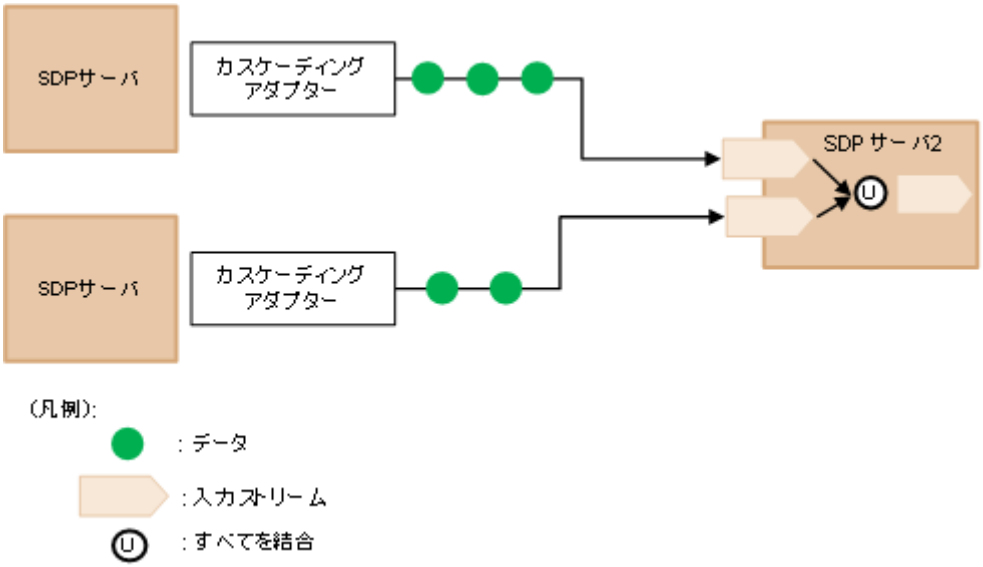


図 16-34　例 2



ファイアウォールを経由してデータを送信するには，SDP サーバをデプロイしている人が，ファイアウォールの設定で送信元および宛先ポートを解放する必要があります。ポートの使用については，「表 16-8　カスケーディングアダプターの通信方法の詳細」を参照してください。

16.7.8　カスケーディングアダプターの時刻同期設定

データソースモードの SDP サーバは，SDP サーバにデータが到着したタイミングで，データに付けられているタイムスタンプを基に，ストリーム処理での時刻を進めます。したがって，複数の SDP サーバを接続する構成の場合，前段の SDP サーバがデータに適切なタイムスタンプを付けて，後段の SDP サーバの時刻を同期する必要があります。SDP サーバ間を接続するカスケーディングアダプターは，この時刻同期をするための機能を提供します。

表 16-15　カスケーディングアダプターの時刻同期機能

機能	説明
分析の時刻同期	カスケーディングアダプターはタプルのシステム時刻フィールド（StreamTuple クラスの*システム時刻*フィールド）に分析時刻を設定し，宛先ストリームエンジンにタプルを渡します。 宛先ストリームエンジンで時刻を使用するには，宛先ストリームエンジンのタイムスタンプモードをデータソースモードに設定する必要があります。
ハートビート	接続先のデータモードの SDP サーバは，データが到着しないとストリーム処理の時刻を進めることができません。そのためカスケーディングアダプターは定期的にハートビートを送信し，時刻を同期します。

「図 16-35　システム構成」および「表 16-16　設定の詳細」に，カスケーディングアダプターの時刻同期機能の使用例を示します。

図 16-35　システム構成

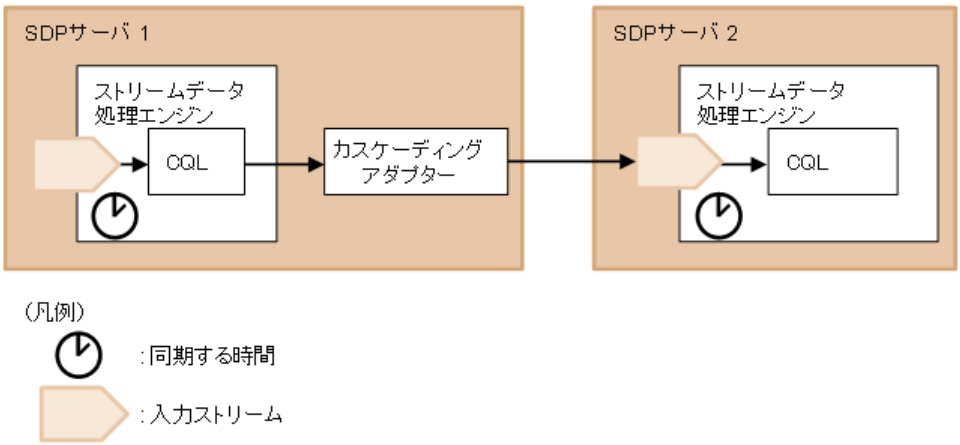


表 16-16　設定の詳細

サーバ	コンポーネント	設定	説明

SDP サーバ 1	ストリームデータ処理エンジン	次のどちらかを設定します。
		`stream.timestampMode=Server`
		`stream.timestampMode=DataSource`
			どちらかのタイムスタンプモードを指定できます。

	CQL	ストリームデータ処理エンジンの時刻制御に基づくクエリを登録します。	-
SDP サーバ 2	ストリームデータ処理エンジン	`stream.timestampMode=DataSource`	SDP サーバ 2 は，データソースモードに設定する必要があります。
		`stream.timestampPosition = __systemtime__`	カスケーディングアダプターが接続する入力ストリームの__systemtime__を設定することで，SDP サーバ 2 がタプルのシステム時刻を使用できるよう指定できます。

サーバ	コンポーネント	設定	説明
SDP サーバ 2	CQL	ストリームデータ処理エンジンの時刻制御に基づくクエリを登録します。	-
（凡例） -：該当なし			

16.7.9　その他のカスケーディングアダプターの特徴

これまでに説明した機能以外のカスケーディングアダプターの特徴は次のとおりです。

- データの送信時に接続エラーが発生すると，カスケーディングアダプターは，データの再送信を試みます。

  クエリグループに接続するカスケーディングアダプターの場合：

  アダプターが，宛先 SDP サーバの入力ストリームから切断されると，カスケーディングアダプターは再接続を試みます。リトライ回数，リトライ間隔，および再接続後に残りのデータを送信するかどうかをクエリグループ用プロパティファイルで指定できます。失敗した回数がリトライ回数を上回ると，接続先ストリーム（エラーの発生場所）へのデータ送信処理が停止します。

  外部出力アダプターに接続するカスケーディングアダプターの場合：

  カスケーディングアダプターが，外部出力アダプターから切断されると，外部出力アダプターは再接続を試みます。アダプター構成定義ファイルで各リトライの待ち時間を指定できます。接続をリトライして待ち時間が経過したあとで接続が再確立しない場合は，外部アダプターによる再接続を待機する処理が停止します。

- 保留中のタプルの数がカスケーディングアダプターの内部キューサイズを超えると，最も古い保留中のタプルがキューから削除されます。

16.8　ファイルへの出力

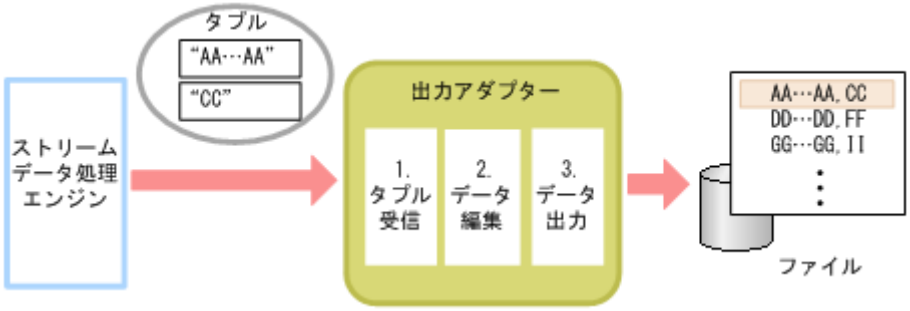
ここでは，ファイルへの出力の概要，データ処理の流れ，および定義ファイルでの設定内容について説明します。

16.8.1　ファイルへの出力の概要

ファイルへの出力では，ストリームデータ処理エンジンで処理したデータを，出力形式に合わせてデータを変換してから出力します。ファイルへの出力の処理は出力アダプターで実施します。

ファイルへの出力の概要を次の図に示します。

図 16-36　ファイルへの出力の概要



1. タプル受信

   ストリームデータ処理エンジンで処理された，ストリームデータの集計・分析結果のタプルを受信します。

2. データ編集

   受信したデータのマッピングおよびフォーマット変換をして，データを編集します。

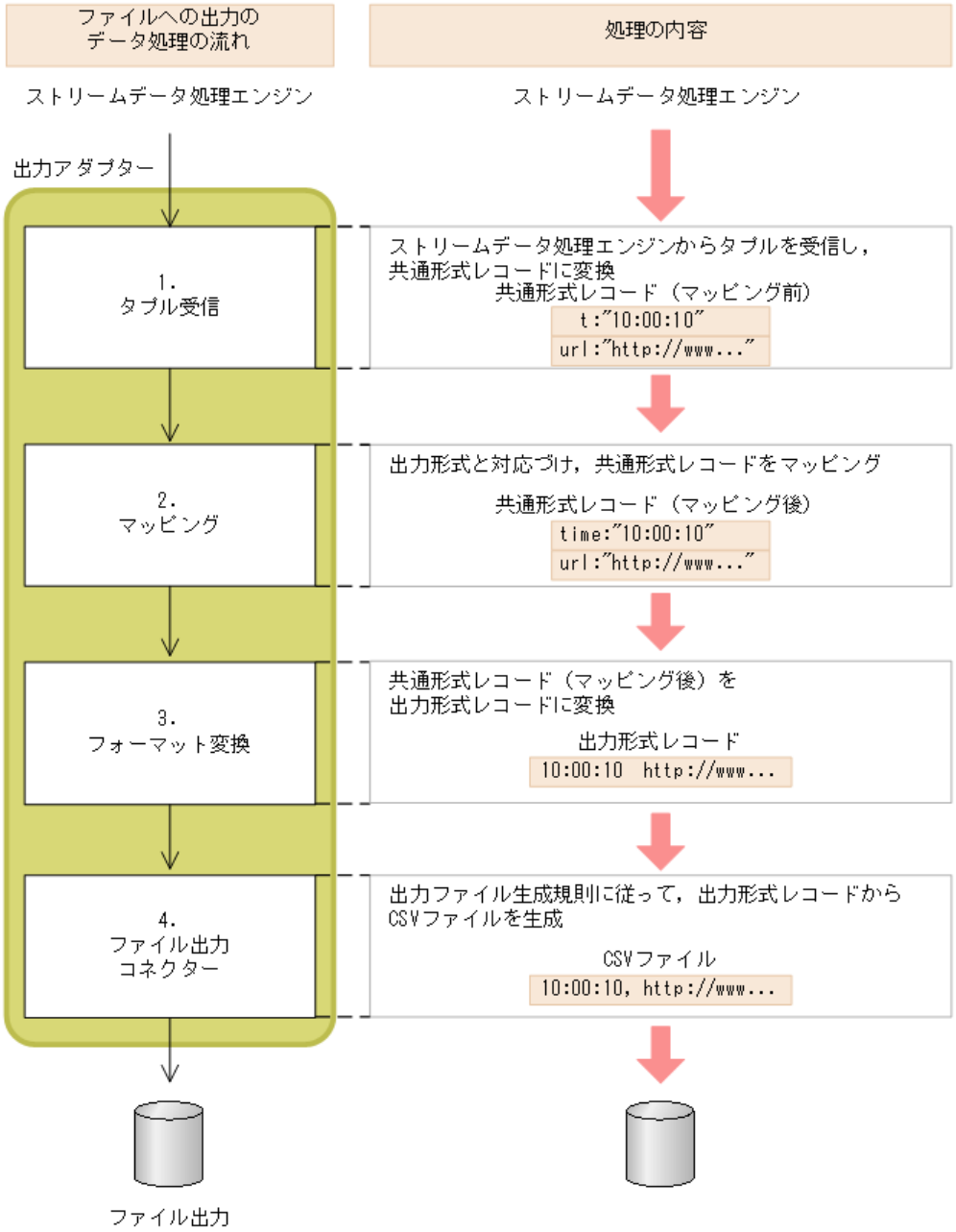
3. データ出力

   ファイル出力コネクターで，編集したデータを出力形式に合わせたデータに変換して出力します。

16.8.2　ファイルへの出力のデータ処理の流れ

出力アダプターでのファイルへの出力のデータ処理の流れと処理の内容を次の図に示します。

図 16-37　ファイルへの出力のデータ処理の流れと処理の内容



出力アダプターで扱うデータ形式について，「16.8.2　ファイルへの出力のデータ処理の流れ」の「(1)　出力アダプターで扱うデータ形式」で説明します。また，各処理の詳細について，「16.8.2　ファイルへの出力のデータ処理の流れ」の「(2)　タプル受信」以降で説明します。

(1)　出力アダプターで扱うデータ形式

出力アダプターで扱うデータ形式には，共通形式レコードおよび出力形式レコードがあります。

共通形式レコードについては，入力アダプターで扱う共通形式レコードと同じです。共通形式レコードについては，「16.3.2　ファイルの入力のデータ処理の流れ」の「入力アダプターで扱うデータ形式」の共通形式レコードの説明を参照してください。ここでは出力形式レコードについて説明します。

出力形式レコード

出力ファイルに出力する，行単位のデータのことです。出力アダプターでは，1 行のデータを 1 出力形式レコードとして扱います。

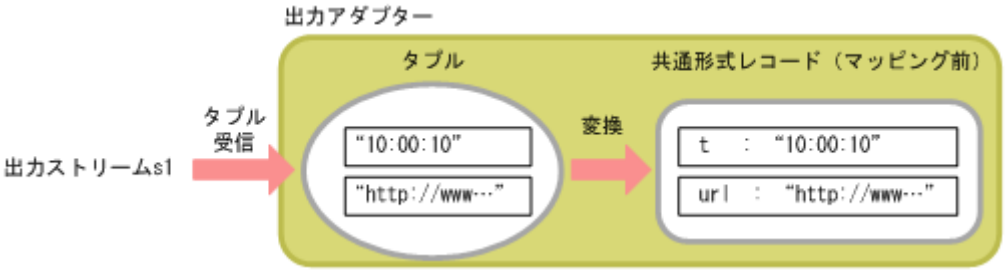
(2) タプル受信

タプル受信についての定義は，アダプター構成定義ファイルの出力ストリーム定義に定義します。

タプル受信では，出力ストリームから送信されたタプルを受信し，共通形式レコードに変換します。

出力アダプターでの，出力ストリームからのタプル受信の例を次の図に示します。この例では，出力ストリームs1 からタプルを受信します。

図 16-38 出力アダプターでのタプル受信の例



(3) マッピング

マッピングについての定義は，アダプター構成定義ファイルのマッピング定義で定義します。

マッピングには，レコードとストリーム間のマッピング，およびレコード間のマッピングがあります。それぞれのマッピングの概要を次の表に示します。

表 16-17 マッピングの概要

項番	マッピングの種類	説明
1	レコードとストリーム間のマッピング	タプル受信で変換された共通形式レコード（マッピングの変換元）を，出力形式に従った共通形式レコード（マッピングの変換先）に変換します。 レコードとストリーム間のマッピングは，タプル受信のあとに必ず実施します。
2	レコード間のマッピング	レコードとストリーム間のマッピングで変換された共通形式レコードを編集し，マッピングの変換先の共通形式レコードに変換します。 レコード間のマッピングは，レコードとストリーム間のマッピングのあと，かつフォーマット変換の前に，必要に応じて実施します。 マッピングの変換元の共通形式レコードのフィールド名を変更したい場合や，複数のフィールドから次のコールバックの処理に不要なフィールドを削除したい場合などに使用できます。

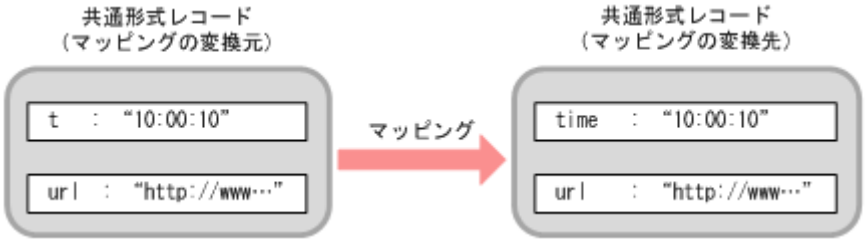
項番	マッピングの種類	説明
2	レコード間のマッピング	また，組み込み関数※を使用して，マッピングの変換元の共通形式レコードから文字列や時刻を取得し，取得した文字列や時刻をマッピングの変換先の共通形式レコードに反映することもできます。 なお，レコード間のマッピングは複数定義できます。

注※

レコード間のマッピングで使用できる組み込み関数は，アダプター構成定義ファイルのmap タグのfunction 属性で指定します。function 属性で指定できる組み込み関数については，「14.8.2　マッピング定義」を参照してください。

出力アダプターでのマッピングの例を次の図に示します。この例では，t フィールドとurl フィールドを，アダプターの出力形式に従ってtime フィールドとurl フィールドに変換し，マッピングの変換先の共通形式レコードに変換します。

図 16-39　出力アダプターでのレコードとストリーム間マッピングの例



メモ

レコードとストリーム間のマッピングのあとには，レコード間のマッピングおよびレコードのフィルタリングが順不同で実施できます。必要に応じて実施してください。

レコードのフィルタリングについては，「16.5　レコードのフィルタリング」を参照してください。

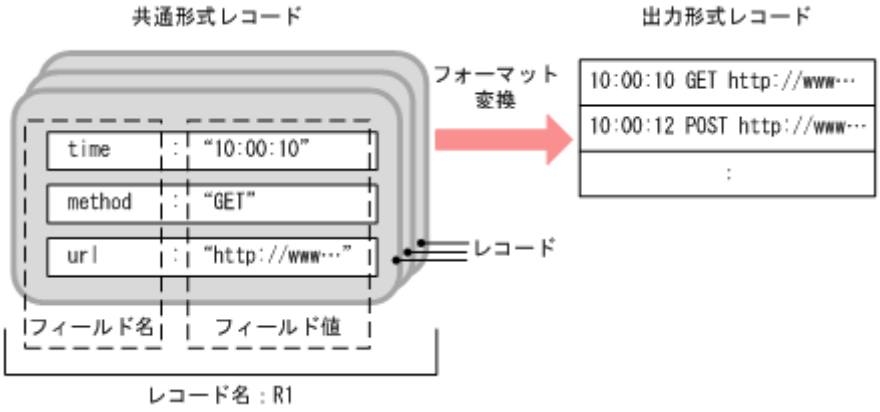
(4)　フォーマット変換

フォーマット変換についての定義は，アダプター構成定義ファイルのフォーマット変換定義で定義します。

フォーマット変換では，共通形式レコードを出力形式レコードに変換します。

共通形式レコードから出力形式レコードへのフォーマット変換の例を次の図に示します。この例では，TIME 型と文字列型のフィールド値を持つ共通形式レコードから，空白で区切られた 3 つのフィールドで構成された出力形式レコードにフォーマット変換されます。

図 16-40　共通形式レコードから出力形式レコードへのフォーマット変換の例



この図の場合の共通形式レコードのレコード構成と指定するフォーマット変換定義のタグを次の表に示します。

表 16-18　指定する定義ファイルとレコード構成

レコード構成	タグ
レコード名：R1 レコード構成：($_time)△($_method)△($_url)	record タグ（レコード定義）
フィールド名：time，型：TIME	field タグ（フィールド定義）
フィールド名：method，型：STRING	
フィールド名：url，型：STRING	

（凡例）

△：半角スペース

変換できるデータ型および共通形式レコードのレコード構成の設定については，「14.8.1　フォーマット変換定義」を参照してください。

(5)　ファイル出力コネクターによるファイルへの出力

ファイル出力についての定義は，アダプター構成定義ファイルのファイル出力コネクター定義で定義します。

ファイル出力コネクターで，入力された出力形式レコードを，定義された出力ディレクトリに出力します。

ファイル出力コネクターで出力できるファイルのファイル構成を次の表に示します。

ファイル構成	説明
ラップアラウンド	ファイル出力コネクター定義に指定した出力ファイル生成規則に従ってファイルを生成します。最大ファイル数に達すると，最初に起動したファイルを上書きして書き込みます。

ファイル構成	説明
非ラップアラウンド	ファイル出力コネクター定義に指定した出力ファイル生成規則に従ってファイルを生成します。最大ファイル数に達すると，出力アダプターでレコードの出力を停止し，レコードを破棄します。

16.8.3　ファイルへの出力の設定

ファイルへの出力について設定が必要な定義ファイルは，アダプター構成定義ファイルです。アダプター構成定義ファイルの出力アダプター定義で，次の CB 定義に設定します。

- 出力用 CB 定義

  ファイル出力コネクター定義に，出力コネクターの情報を定義します。定義の詳細については，「14.6.2 出力用 CB 定義」，および「14.7.3　ファイル出力コネクター定義」を参照してください。
- 編集用 CB 定義

  フォーマット変換定義に，出力データのデータ形式を定義します。マッピング定義には，マッピング情報を定義します。定義の詳細については，「14.6.3　編集用 CB 定義」，「14.8.1　フォーマット変換定義」，または「14.8.2　マッピング定義」を参照してください。
- 受信用 CB 定義

  出力ストリーム定義に，出力アダプターが接続する出力ストリームの情報を定義します。定義の詳細については，「14.6.5　受信用 CB 定義」，および「14.9.2　出力ストリーム定義」を参照してください。

16.9　ダッシュボードへの出力

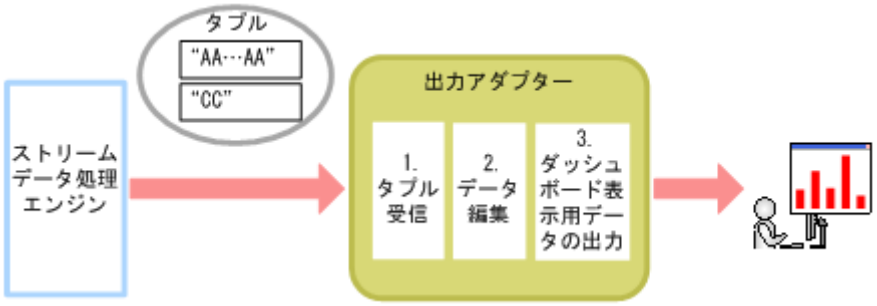
ここでは，ダッシュボードへの出力の概要，データ処理の流れ，および定義ファイルでの設定内容について説明します。

16.9.1　ダッシュボードへの出力の概要

ダッシュボードへの出力では，ストリームデータの集計・分析結果を棒グラフや折れ線グラフの形式でダッシュボードに表示するために，ストリームデータ処理エンジンで処理したデータを，ダッシュボード表示用データに変換します。

ダッシュボードへの出力の概要を次の図に示します。

図 16-41　ダッシュボードへの出力の概要



1. タプル受信

   ストリームデータ処理エンジンで処理された，ストリームデータの集計・分析結果のタプルを受信します。

2. データ編集

   受信したデータのマッピングをします。

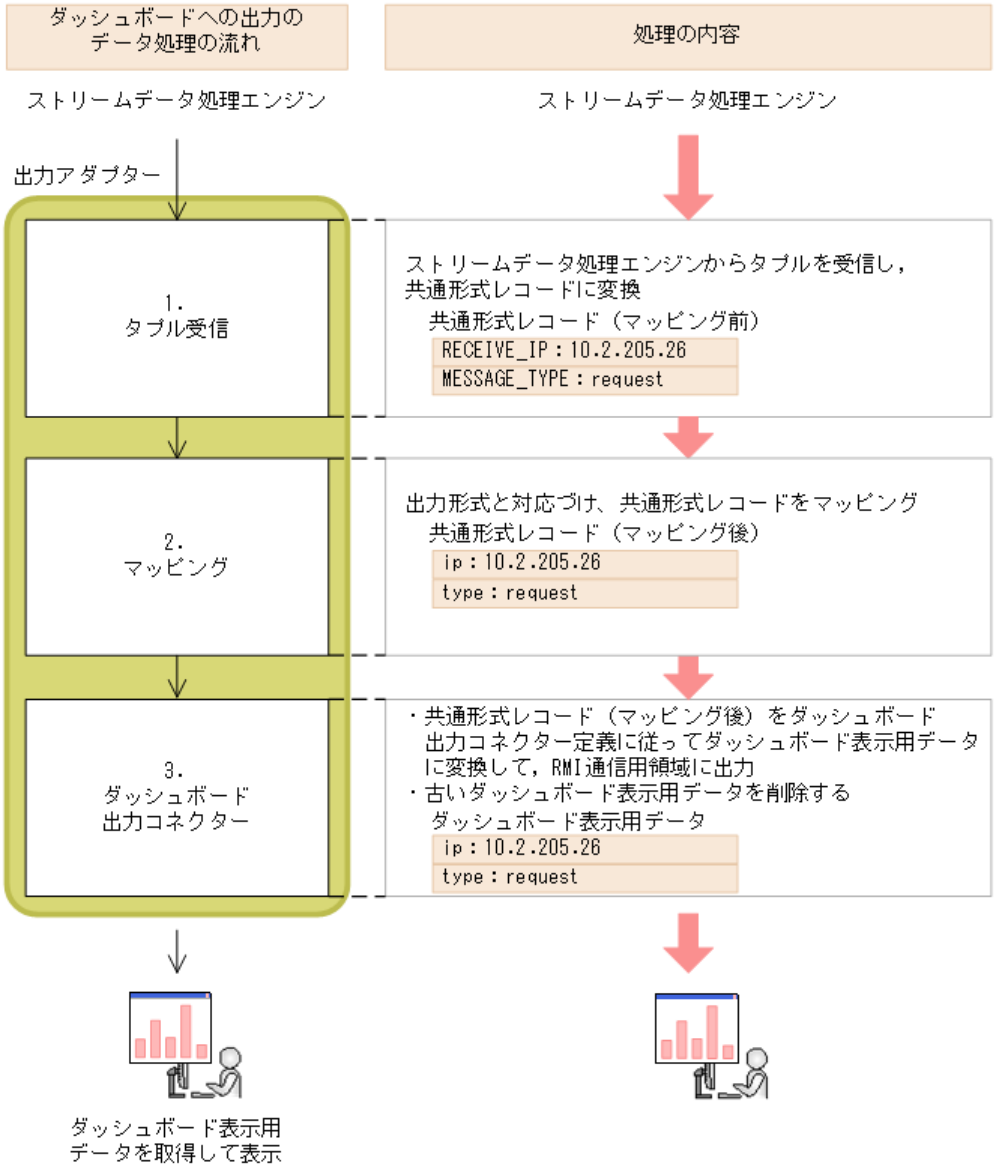
3. ダッシュボード表示用データの出力

   ダッシュボード出力コネクターで，編集したデータをダッシュボード表示用データに変換して出力します。また，古くなったデータを削除します。

16.9.2　ダッシュボードへの出力のデータ処理の流れ

ダッシュボードへの出力のデータ処理の流れと処理の内容を次の図に示します。

図 16-42　ダッシュボードへの出力のデータ処理の流れと処理の内容



タプル受信，およびマッピングについては，「16.8.2　ファイルへの出力のデータ処理の流れ」の「(2)　タプル受信」および「(3)　マッピング」を参照してください。また，出力アダプターで扱うデータ形式については，「16.8.2　ファイルへの出力のデータ処理の流れ」の「(1)　出力アダプターで扱うデータ形式」を参照してください。

(1)　ダッシュボード出力コネクターでのダッシュボード表示用データへの変換

ダッシュボード出力コネクターでダッシュボード表示用データへ変換するための定義は，アダプター構成定義ファイルのダッシュボード出力コネクター定義で定義します。

ダッシュボード出力コネクターでは，マッピング後の共通形式レコードを取得し，ダッシュボード出力コネクター定義に従ってダッシュボード表示用データに変換して，RMI 通信用領域に出力します。ダッシュボードでは，RMI 通信用領域に出力したダッシュボード表示用データを取得して表示します。

(2) ダッシュボード出力コネクターでのレコードの削除

ダッシュボード出力コネクターでレコードを削除するための定義は，アダプター構成定義ファイルのダッシュボード出力コネクター定義に定義します。

ダッシュボード出力コネクターで取得したダッシュボード表示用データがダッシュボード出力コネクターのメモリを圧迫しないように，ダッシュボード出力コネクター定義に定義したレコードの削除条件に従って，ダッシュボード出力コネクターのレコード保持領域からダッシュボード表示用データを削除します。

レコードの削除には，次の方法があります。

レコード保持期間による削除

レコードの保持期間による削除では，ダッシュボード出力コネクターに新しいレコードが追加されたとき，レコード保持領域内のレコードのうち，基準時刻から保持期間を引いた時刻よりもレコードに設定された時刻情報が古いレコードが削除されます。例えば，基準時刻が10:00:10，保持期間が5 秒，レコードに設定された時刻が10:00:04 だった場合，そのレコードは削除されます。

基準時刻から保持期間を引いた時刻とレコードに設定された時刻情報が同じ場合，レコードは削除されません。

なお，レコードの保持期間による削除は，レコード保持領域の各レコードについて，レコードに設定された時刻情報がレコードの到着順に昇順になっているものとして動作します。昇順になっていない場合，基準時刻から保持期間を引いた時刻よりもレコードに設定された時刻情報が新しいレコードのあとに到着した，基準時刻から保持期間を引いた時刻よりも古いレコードは削除されません。

なお，時刻の比較をする際の時刻の精度はミリ秒です。ミリ秒より小さい精度は切り捨てられます。

レコードの保持期間による削除は，ダッシュボード出力コネクター定義のRecordHoldTime タグの，DateReference 属性，RecordTime 属性，およびDateFieldPosition 属性に指定します。指定する属性の詳細については，「14.7.4　ダッシュボード出力コネクター定義」を参照してください。

最大レコード保持数による削除

最大レコード保持数による削除では，ダッシュボード出力コネクターに新しいレコードが追加された時，レコード保持領域内のレコード数が最大レコード保持数よりも大きい場合に，レコード数が最大レコード保持数と等しくなるまでレコードを削除します。このとき，レコードは，レコード保持領域内に追加された順番で古いレコードから削除されます。

最大レコード保持数による削除は，ダッシュボード出力コネクター定義の DashboardOutputConnectorDefinition タグのMaxNum 属性に指定します。属性の詳細については，「14.7.4　ダッシュボード出力コネクター定義」を参照してください。

取得済みレコードの削除

取得済みレコードの削除では，1 つのダッシュボードだけがダッシュボード表示用データを取得する場合に，ダッシュボードが取得済みのレコードを，ダッシュボード出力コネクターのレコード保持領域から削除します。

取得済みレコードの削除は，ダッシュボード出力コネクター定義のDashboardOutputConnectorDefinition タグのReadRecordRemoveFlag 属性に指定します。属性の詳細については，「14.7.4　ダッシュボード出力コネクター定義」を参照してください。

(3)　ダッシュボードでのダッシュボード表示用データの取得

ダッシュボードに，ダッシュボード表示用データを取得させるための定義は，アダプターグループ定義に定義します。

ダッシュボードでは，ダッシュボード出力コネクターの RMI 通信用領域にアクセスして，出力されたダッシュボード表示用データを取得します。

なお，ダッシュボードが RMI 通信用領域にアクセスするためには，RMI サーバのポート番号を指定します。RMI サーバのポート番号は，アダプターグループ定義のインプロセスグループ定義の`dashboardPortNo`属性に指定します。指定する属性の詳細については，「14.5.1　インプロセスグループ定義」を参照してください。

16.9.3　ダッシュボードへの出力の設定

ダッシュボードへの出力で設定が必要な定義ファイルは，アダプター構成定義ファイルです。アダプター構成定義ファイルの，アダプターグループ定義と，出力アダプター定義の編集用 CB 定義および出力用 CB 定義とに設定します。

- アダプターグループ定義

  インプロセスグループ定義に RMI サーバのポート番号を指定します。定義の詳細については，「14.5.1 インプロセスグループ定義」を参照してください。

- 編集用 CB 定義

  マッピング定義に，マッピング情報を定義します。定義の詳細については，「14.6.3　編集用 CB 定義」，および「14.8.2　マッピング定義」を参照してください。

- 出力用 CB 定義

  ダッシュボード出力コネクター定義に，ダッシュボード出力コネクターの情報を設定します。定義の詳細については，「14.6.2　出力用 CB 定義」，および「14.7.4　ダッシュボード出力コネクター定義」を参照してください。

株式会社 日立製作所
〒 100-8280　東京都千代田区丸の内一丁目 6 番 6 号