

Contents

<目次>

1. 利用概要	Usage Summary	1-1
2. 関数の提供	Providing Functions	2-1
3. 動作環境	Operating environment	3-1
4. 環境構築から実行までの流れ	Flow from environment construction to execution	4-1
4.1. 環境構築・実行	Environment construction・execution	4-1
4.2. コンパイルオプション	Compile option	4-2
5. SDTP関数処理フロー	SDTP Function processing flow	5-1
6. SDTP関数説明	SDTP Function Description	6-1
6.1. SDTP関数詳細	SDTP Function detail	6-1
6.2. 既存関数からの変更方法	How to change from an existing function	6-2
7. 環境定義ファイル	Environment definition file	7-1
7.1. ファイル形式	File format	7-1
7.2. ファイル項目	File entry	7-1
7.3. ファイルイメージ	File image	7-2
7.4. 既存ファイルからの変更	Change from the existing file	7-3
8. 環境変数	Environment variable	8-1

Appendix

付録1. 用語説明	Glossary
付録2. msys環境作成手順	msys Environment creation procedure
付録3. インストール手順	Installation procedure
付録4. エラーコード一覧	Expected error codes

→ The new SDTP function can be run on Solaris, Linux, Windows. The usage image of the SDTP function is shown in Figure 1-1.

Also explanation of the terms used in the SDTP function is shown in Appendix 1 Glossary.
Usage summary

1. 利用概要

- ・ 現在、SDTP関数はデータ蓄積用関数群とSIRIUS用関数群の2種類があり、取得先を切り替えたい場合には、関数を使い分けなければならず、汎用性が低かった。
- ・ 新しいSDTP関数は、1つの関数群で取得先を意識することなく、データ蓄積とSIRIUSへの接続が可能となっている。
- ・ 実際には、SIRIUS用SDTP関数をベースに、現2種類のSDTP関数の引数を統合し、引数に受信時刻で取得するか生成時刻で取得するかを指定することで、データ蓄積に接続するか、SIRIUSに接続するか判定することになる。

なお、新しいSDTP関数は、Solaris、Linux、Windows上で稼働させることが可能である。

SDTP関数の利用イメージを図 1-1に示す。

また、SDTP関数で使用する用語の説明を「付録1 用語説明」に示す。

- ・ Currently, there are two kinds of SDTP functions, a data storage function group and a SIRIUS function group, and when you want to change the acquisition destination, you have to select the function properly, and the versatility is low.
- ・ With the new SDTP function, data can be accumulated and connected to SIRIUS without being conscious of one function acquisition destination.
- ・ In fact, based on the SDTP function for SIRIUS, we integrate the arguments of SDTP functions and designate whether to acquire at the reception time or at the generation time as the argument to connect to the data storage. It is determined whether to connect to SIRIUS

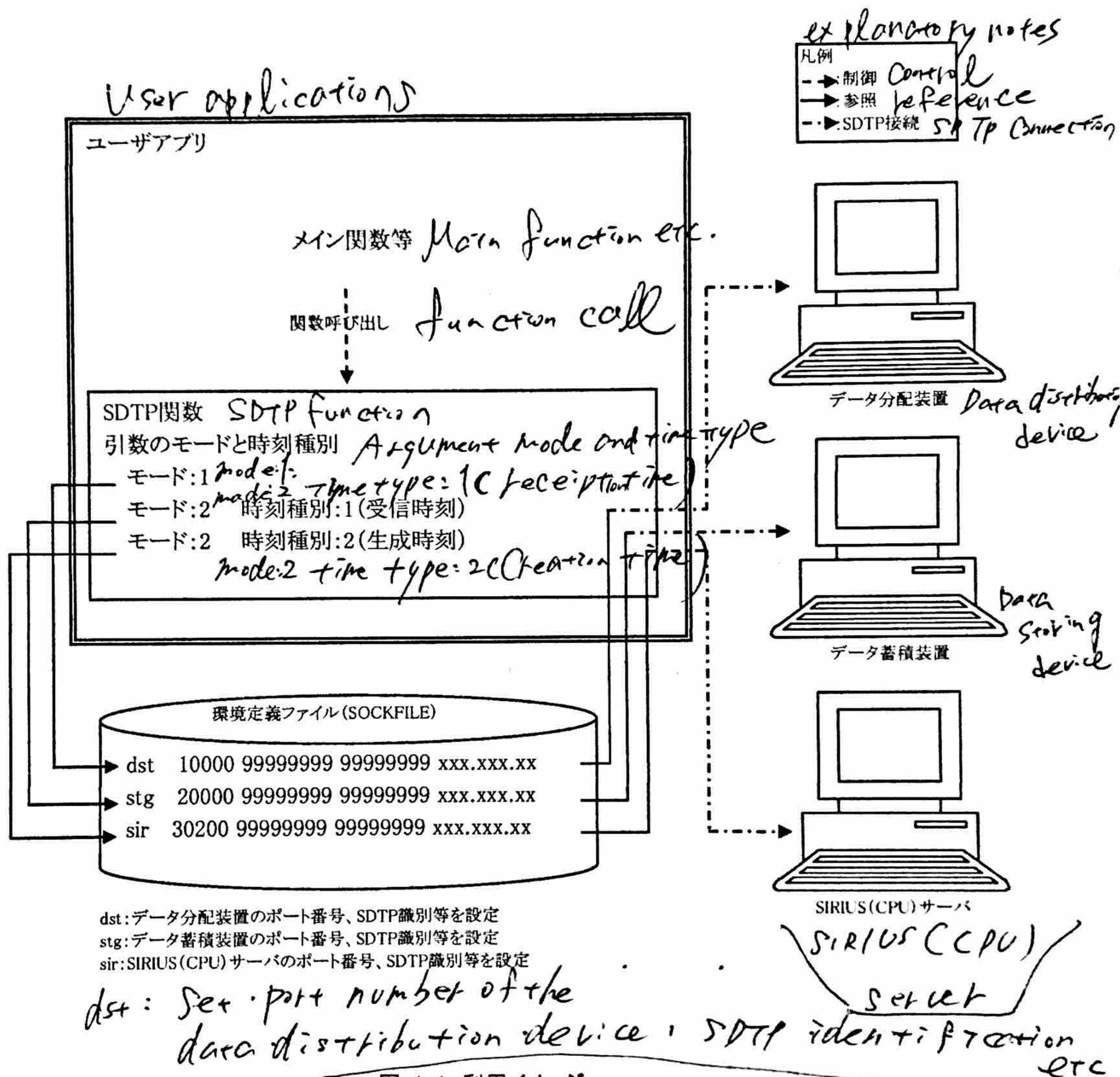


図 1-1 利用イメージ

Figure 1-1 the usage image

dst: Set port number of the data distribution device, SDTP identification etc.

stg: Set port number of the data storage device, SDTP identification etc.

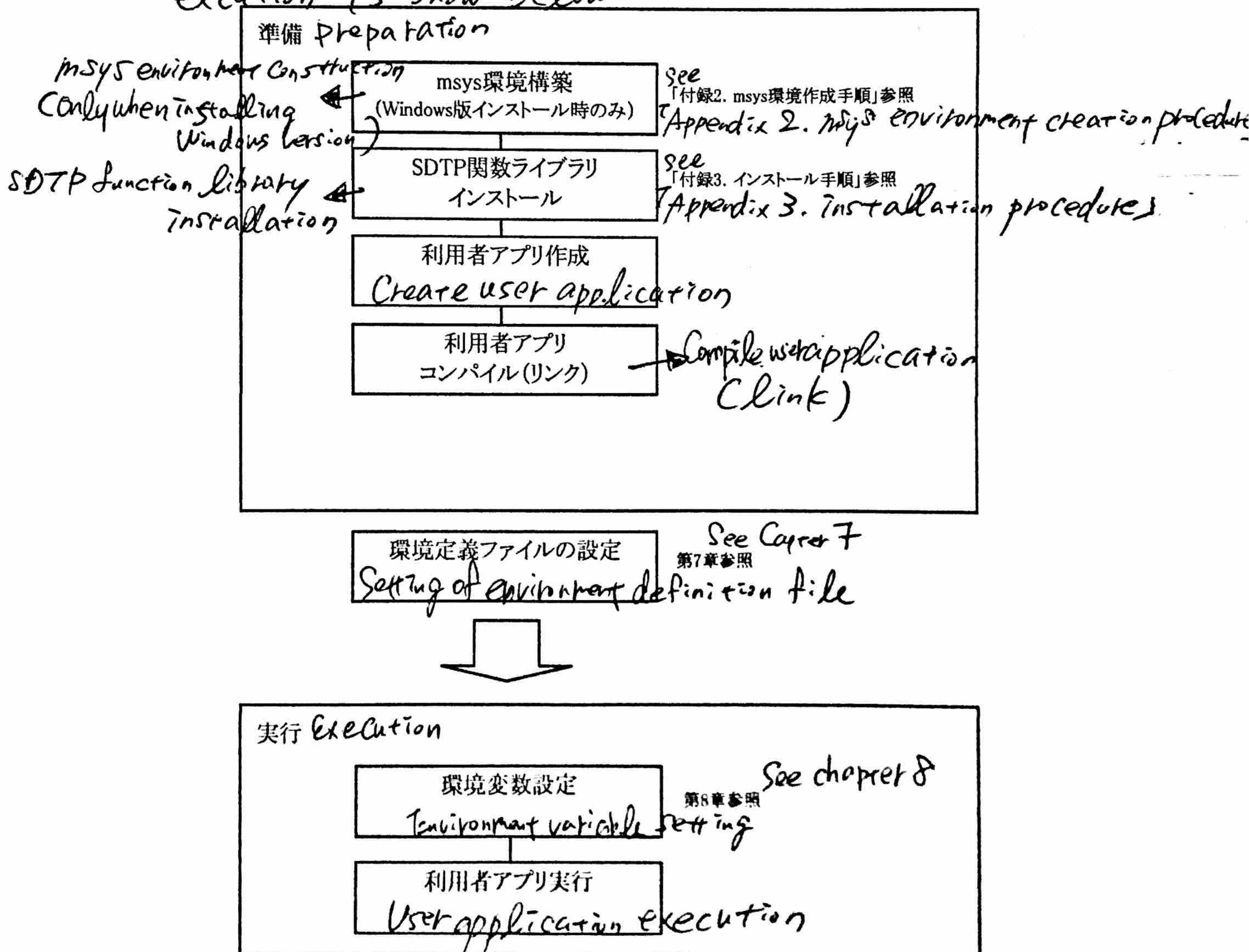
sir: Set port number of SIRIUS server, SDTP identification.

4. 環境構築から実行までの流れ *Flow from environment construction to execution*

4.1. 環境構築・実行 *Environment Construction・Execution*

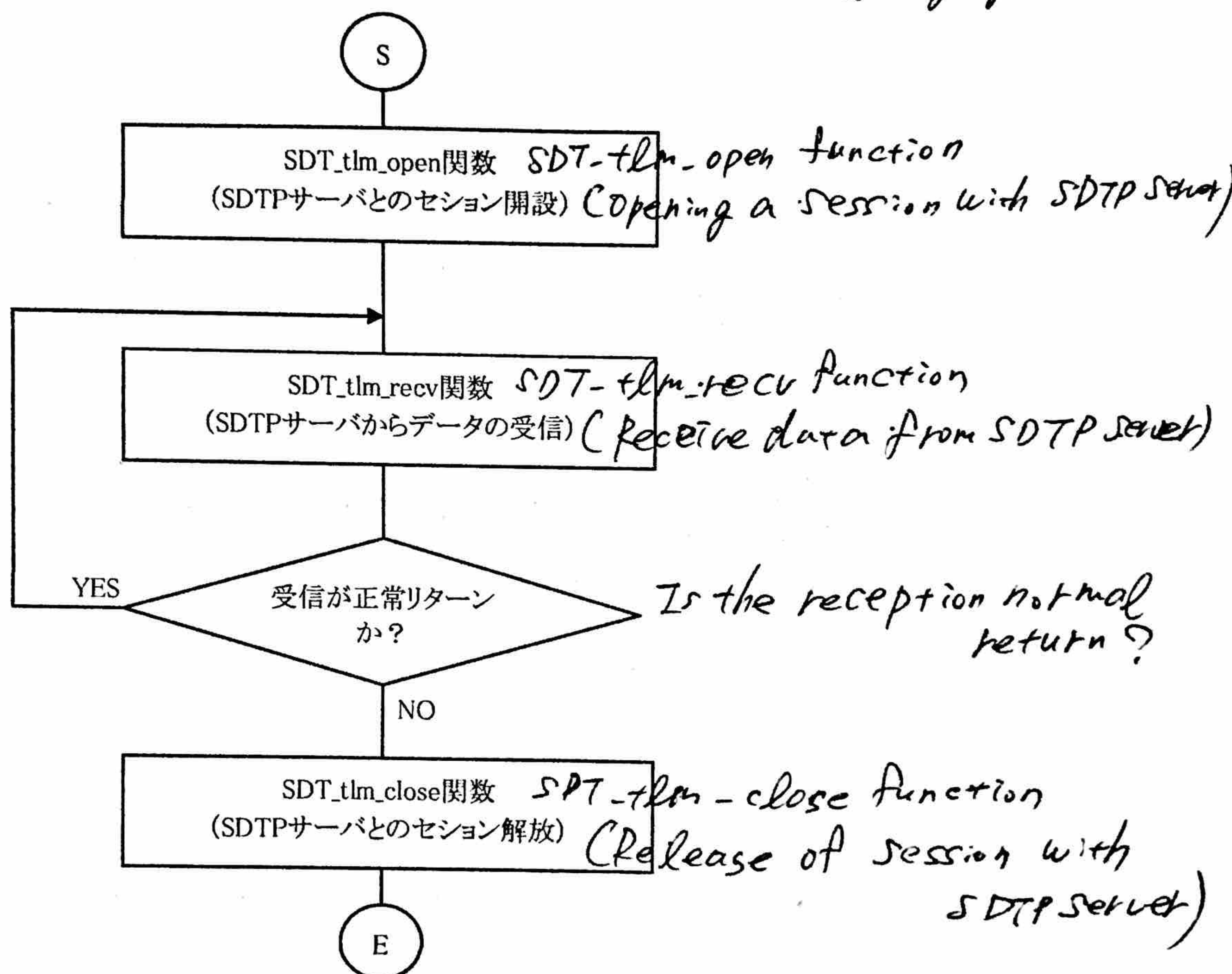
以下に環境構築から利用者アプリ実行までの流れを示す。

The flow from environment construction to user application-execution is show below



5. SDTP関数処理フロー *SDTP Function processing flow*

SDTP関数の使用例を以下のフローに示す。 *An example of using the SDTP function is shown in the following flow.*



※詳細は、ソースファイルに同梱のサンプルプログラム参照

For details, refer to the sample program included in the source file

6. SDTP関数説明 *SDTP function description*

6.1. SDTP関数詳細 *SDTP function detail*

SDTP関数の詳細を以降に示す。

The details of the SDTP function are shown below

- (1) 現SDT_tlm_openとSDT_tlm_open_siriusの引数の統合
- (2) 引数に受信時刻でデータ取得をするのか、生成時刻でデータ取得をするのかを指定できる項目を「時刻種別」として追加
- (3) 入力引数に対するconst指定とプラットフォーム非依存型の対応
- (4) エラーコードの統一と見直し



- (1) *Argument integration of current SDT_tlm_open and*
- (2) *「^{time-kind}Time type」 is added as an item that can specify SDT_tlm_open_sirius whether to acquire reception time data as argument or data acquisition at generation time.*
- (3) *Correspondence of const designation and platform-independent type to input arguments*
- (4) *Unification and review of error codes.*

Module Abbreviation name	Module name	Applicable language	Remarks
SDTPクライアント opening process モジュール略称名	SDT_tlm_open モジュール名	C 適用言語	備考 1/2
【機能概要】 Functional overview SDTPクライアントからテレメトリデータを取得するためのセッションを開設する処理を行う。 Perform a process of opening a session for acquiring telemetry data from the SDTP client. 【記述形式】 Description format <pre> #include "SDT_all.h" /* SDTP情報関連定義ヘッダファイル */ SDTP information related definition header file #include "SDT_tlm.h" /* SDTP情報関連定義ヘッダファイル */ int SDT_tlm_open(const int mode, const int type, const int sat_no, const int ant_band[2], const int cpn_type, const int cpn_cnt, const Cpn_Data cpn_data[], const int blk_no, const int time_kind, const char *start_time, const char *end_time, const char *passno); const int mode; /* モード */ mode const int type; /* タイプ */ type const int sat_no; /* 衛星番号 */ satellite number const int ant_band[2]; /* 受信アンテナ番号 受信バンド帯 */ receive antenna number, receive band const int cpn_type; /* CPNサービス種 */ CPN service type const int cpn_cnt; /* CPNサービス個数 */ number of CPN service const Cpn_Data cpn_data[]; /* CPNサービス情報 cpn_cnt分 */ CPN service information, const int blk_no; /* ブロック化係数 */ blocking factor const int time_kind; /* 時刻種別 */ time type const char *start_time; /* 開始時刻 */ start time const char *end_time; /* 終了時刻 */ end time const char *passno; /* パス番号 */ pass number </pre> 【パラメタ説明】 Parameter Description mode : (IN) モード(1:リアル、2:レートバッファ) type : (IN) タイプ(1:パス番号指定、2:時刻指定。モードがリアルの場合は無効) sat_no : (IN) 衛星番号 ant_band : (IN) ant_bnd[0]:受信アンテナ番号、ant_bnd[1]:受信バンド帯(指定数分、最大8) 指定数が8以外の場合は、指定数の終端判定用に指定数(n)の次の受信バンド帯の要素(ant_bnd[n][1])にゼロを入れること。 SIRIUSから旧フレーム形式衛星のデータを取得する場合は、アンテナ番号に群番号を指定する。 cpn_type : (IN) CPNサービス種別 (1:パケット個別指定 2:VCDU個別指定 8:トランスファフレーム指定) (非CCSDS 1:同期フレーム指定 2:非同期 3:無効) cpn_cnt : (IN) CPNサービス個数 cpn_data : (IN) CPNサービス情報構造体の先頭ポインタ(以下の構造体のCPNサービス個数分定義) <pre> typedef struct{ uint8_t vc_mask; /* バーチャルチャネルIDのマスク値 */ uint8_t vc_ch; /* バーチャルチャネルIDの値 */ uint8_t pc_mask[2]; /* パケットID のマスク値。ネットワークバイトオーダーで指定 */ uint8_t pc_id[2]; /* パケットID (AP-ID)の値。ネットワークバイトオーダーで指定 */ }Cpn_Data; </pre> (バーチャルチャネルID/パケットID指定の設定データ内容については「DIOSAインタフェース仕様 宇宙データ転送プロトコル(SDTP) OSO 501-Xを参照。 http://e-soda.isas.jaxa.jp/sog/document/document.html に掲載。) blk_no : (IN) ブロック化係数(1読み込み当りのデータ件数、ブロック数が大きい程転送速度が早い、1ブロック65536バイトは超えない。モードがリアルの場合は無効) time_kind : (IN) 時刻種別(1:受信時刻、2:生成時刻) (モードがリアルの場合は無効。タイプがパス番号指定、時刻指定に関わらず指定する。)			

6-2

[Parameter Description]

mode : (IN) mode (1: real; 2: Rate buffer)
type : (IN) Type (1: pass number specification; 2: time specification)
Invalid when the mode is real.

Sat_no : (IN) Satellite number

ant-band : (IN) ant-band[0]: Receiving antenna number,
ant-band[1]: Receiving band (Specified quantity, Up to 8)

If the specified number is other than 8,
put zero in the element (ant-band[n][1]) of the receiving
band next to the specified number (n) for
judgment of the end of the specified number.

When acquiring old frame type satellite data
from SIRIUS, specify the group number as
the antenna number.

cpn-type : (IN) CPN service type

1: Individual specification of packet

2: Individual specification of VCDU

8: Transfer frame specification

(Non CCSDS - 1: Sync frame specification

2: asynchronous 3: Disabled
(Inoperative?)

cpn-cnt : (IN) CPN service number

cpn-data : (IN) CPN head pointer of the service information
structure (defined as the number of CPN services
of the following structure)

typedef struct {

uint8_t vc_mask; /* Mask value of virtual
channel ID */

uint8_t vc_ch; /* Value of virtual channel ID */

uint8_t pc_mask[2]; /* Mask value of packet ID
Specify by network byte
order */

uint8_t pc_id[2]; /* Value of packet ID (AP-ID)
Specify by network byte order */

} Cpn-Data;

(for setting contents of virtual channel ID / Packet ID specification data, refer to "D/OSA interface specification, Space data transfer Protocol (SDTP) OSO-501-X.

posted on <http://c-sda.isas.jaxa.jp/s29/document/document.html>)

blk: no

:(IN) Blocking factor

(The larger the number of data per read and the larger the number of blocks, per read. is, the faster the transfer rate but not more than 65536 bytes per block.

Invalid when the mode is real.)

time_kind

:(IN) Time type (1: Reception time, 2: Creation time)
(time kind)

(Invalid when the mode is real.

Type is specified regardless of

pass number specification or time specification)

Module Abbreviation name	Module name	Application Language	Remarks
モジュール略称名	モジュール名	適用言語	備考
SDTPテレメトリ開設処理	SDT_tlm_open	C	2/2
start_time : (IN)タイプが時刻指定の場合、開始時刻(YYYYMMDDhhmmss) (When the type is time specification, the start time specification) (モードがリアルの場合は無効) (Invalid when the mode is real) end_time : (IN)タイプが時刻指定の場合、終了時刻(YYYYMMDDhhmmss) (When the type is time specification, the end time specification) (モードがリアルの場合は無効) (Invalid when the mode is real) Passno : (IN)タイプがパス番号指定の場合、パス番号(YYMMDD9999) (When the type specifies the pass number, the pass number) (モードがリアルの場合は無効) (Invalid when the mode is real)			

Return value (Invalid when the mode is real)

【復帰値】

Return value

復帰値	意味 meaning	処置 the action
0	正常終了 Successful termination	
-20001	入力パラメタエラー Input parameter error	入力パラメタを見直す。Review input parameters
-20003	開設否定応答受信 Receive response not to be established	入力パラメタ、環境定義ファイルを見直す。Review input parameters and environment definition file
-20004	タイムアウト time out	クローズする。to close
-20005	SOCKET生成エラー socket generation error	環境定義ファイルを見直す。Review environment definition file
-20007	connectエラー Connect error	環境定義ファイルを見直す。Review environment definition file
-20009	サーバ側CLOSE Server side close	クローズする。To close
-20010	SDTP初期化エラー SDTP Initialization error	入力パラメタを見直す。Review input parameters
-20011	環境変数取得エラー Environment variable acquisition error	環境変数設定を見直す。Review environment variable setting
-20012	ファイルアクセスエラー	環境定義ファイルを見直す。Review environmental definition file
-20013	環境定義ファイル内容異常	環境定義ファイルを見直す。Review environmental definition file
-20014	環境変数設定エラー	クローズする。to close
上記以外 other than above	付録4のエラーコードを負にした値(詳細は、付録4を参照) Value with negative error code in Appendix 4	クローズする。to close

Notes

【注意事項】

関数の引数mode値およびtime_kind値から接続先およびSDTPの動作が決定し、環境定義ファイルの接続先識別の対応情報を使用して接続を行う。対応関係を以下に示す。

SDT_tlm_openの引数 Argument of SDT_tlm_open		SDTP関数の動作 SDTP function behavior	使用する環境 定義ファイル の接続先識別
mode	time_kind		
1(リアル)(real)	—	データ分配装置に接続し、リアルタイムに受信を行う。Connect to the data distribution device and receive in real time	ds
2(レートバッファ) Rate buffer	1(受信時刻) reception time	データ蓄積装置に接続し、レートバッファで蓄積データを受信する。Connect to the data storage device and receive the accumulated data in the rate buffer	stg
2(レートバッファ) Rate buffer	2(生成時刻) generation time	SIRIUSに接続し、レートバッファで蓄積データを受信する。Connect to SIRIUS and receive accumulated data in the rate buffer	sir

The connection destination and the operation of SDTP are determined from the function argument mode value and time_kind value, and connection is made using correspondence information of the connection destination identification of the environment definition file. The correspondence is shown below

SDTP telemetry opening process

Module Abbreviation name	Module name	Application language	Remarks
モジュール略称名	モジュール名	適用言語	備考
SDTPテレメトリ受信処理	SDT_tlm_recv	C	

【機能概要】 Functional overview

SDTPサーバからテレメトリデータを取得する。Acquire telemetry data from SDTP server

【記述形式】 Description format

```
#include "SDT_all.h" /* SDTP情報関連定義ヘッダファイル */
#include "SDT_tlm.h" /* SDTP情報関連定義ヘッダファイル */
```

SDTP information related definition header file

```
int SDT_tlm_recv(int *pdu_type, uint8_t *recv_data, int *recv_len);
```

```
int *pdu_type; /* 受信したPDUのPDU種別 */ PDU type of the received PDU
uint8_t *recv_data; /* 受信データ部格納領域 */ Stage area of receive data part.
int *recv_len; /* 受信データ長 */ Receive data length
```

PDU type of the received PDU

【パラメタ説明】 Description of parameters

pdu_type : (OUT) 受信したPDUのPDU種別

recv_data : (OUT) 受信データ部格納領域

recv_len : (OUT) 受信データ長

Receive data length

Refer to DIOSA interface specification space data transfer protocol (SDTP) OSO 501-Xを参照。

<http://c-soda.isas.jaxa.jp/sog/document/document.html> に掲載。Posted on

OSO-501-X

【復帰値】 Return value

Stage area of receive data part.

Receive communication release request PDU

復帰値	意味	処置
0	正常終了(受信データ有り)	Successful completion (with received data)
1	正常終了(受信データ無し)	Successful completion (no received data)
2	通信解放依頼PDU受信	クローズする。to close
-20002	通信シーケンスエラー	クローズする。to close
-20004	タイムアウト time out	クローズする。to close
-20006	データ抜け発生 Data missing occurs	クローズする。to close
	(シーケンスカウンタエラー) (Sequence counter error)	処理続行可能 Processing can continue
-20009	サーバ側CLOSE Server side close	クローズする。to close
上記以外 other than above	付録4のエラーコードを負にした値(詳細は、付録4を参照)	クローズする。to close

Value with negative error code in Appendix 4 (For detail, refer to Appendix 4)

【注意事項】 Note

なし None

Communication sequence error

When changing from the old `SDT_tlm_open` to the new `SDT_tlm_open`, change the `ant_id` and `band` set in the existing processing to the two-dimensional array of `ant_band` and `blk_no`. Also add `time_kind` between `blk_no` and `start_time`.

How to change from an existing function 6.2. 既存関数からの変更方法

Change from old `SDT_tlm_open` to new `SDT_tlm_open`

(1) `IESDT_tlm_open`から新`SDT_tlm_open`への変更

`IESDT_tlm_open`から新`SDT_tlm_open`に変更する場合、既存処理で設定していた`ant_id`と`band`を`ant_band`の2次元配列に設定するように変更し、2次元配列を新`SDT_tlm_open`に指定する。また、`blk_no`と`start_time`の間に`time_kind`を追加する。

`SDT_tlm_open(mode, type, sat_no, ant_id, band, cpn_type, cpn_cnt, cpn_data, blk_no, start_time, end_time, pathno)`

`SDT_tlm_open(mode, type, sat_no, ant_band, cpn_type, cpn_cnt, cpn_data[], blk_no, time_kind, start_time, end_time, passno)`

Change from `SDT_tlm_open-sirius` to new `SDT_tlm_open`

(2) `SDT_tlm_open-sirius`から新`SDT_tlm_open`への変更

`SDT_tlm_open-sirius`から新`SDT_tlm_open`に変更する場合、`blk_no`と`start_time`の間に`time_kind`を追加する。

When changing from `SDT_tlm_open-sirius` to new `SDT_tlm_open`, add `time_kind` between `blk_no` and `start_time`

`SDT_tlm_open-sirius(mode, type, sat_no, ant_band, cpn_type, cpn_cnt, cpn_data, blk_no, start_time, end_time, pathno)`

`SDT_tlm_open(mode, type, sat_no, ant_band, cpn_type, cpn_cnt, cpn_data[], blk_no, time_kind, start_time, end_time, passno)`

When using the SDTP function, the following environment definition file is necessary, so the user sees it under an arbitrary directory.

Although the file name of the environment definition file is arbitrary (it is not possible to have blank in the file name), SOCKFILE is recommended.

7. 環境定義ファイル

・SDTP関数を使用する際には、以下の環境定義ファイルが必要であるため、利用者が任意のディレクトリ配下に設定しておく。環境定義ファイルのファイル名は任意(ファイル名に空白があるものは不可)であるが、SOCKFILEを推奨する。

7.1. ファイル形式

This file is an ASCII text file composed of information lines and comment lines.

本ファイルは、情報行とコメント行で構成されるASCIIテキストファイルである。

情報行は、改行コードを除いてASCII文字から構成され、情報行内の各項目は、1文字以上の「スペース(0x20)」で区切られる。The information line consists of ASCII characters except for the line feed code, and each item in the information line is separated by one or more spaces (0x20).
コメント行は、先頭位置に「#(0x23)」を設定した行とし、#より後の文字は、改行コードを除けばどの文字コードでも良い。A comment line is a line in which '#' (0x23) is set at the head by one or more positions and any character after # can be any character code except for the line feed code.
改行コードは、OSの文字コードに依存するため、本SDTP関数を動作させる文字コードに合致した改行コードを設定する。(例: Shift-JIS: CR+LF、EUC: LF、UTF-8: LF等)

なお、空行は不可とする。Since the line feed code depends on the character code of the OS, a line feed code matching the character code for operating this SDTP function is set (examples: shift-JIS, etc...)

7.2. ファイル項目

本ファイルの項目一覧を表 7-1に示す。
The item list of this file is shown in Table 7-1

表 7-1 ファイル項目一覧 Table 7-1 File item list

No.	項目名 Item name	項目説明 Item description	備考 Remarks
1.	接続先識別 Connection destination identification	接続先装置の識別 dst: データ分配装置 Data distribution device stg: データ蓄積装置 Data storage device sir: SIRIUS (CPU) サーバ Server	半角英小文字とする。 Lower case letter
2.	ポート番号 Port number	接続先装置に接続するポート番号。ポート番号は以下の通りとする。 10000: データ分配 Data distribution 20000: データ蓄積 Data accumulation 30200: SIRIUS (CPU) サーバ Server	数値のみ Numerical only
3.	送信元識別子 Source identification	利用者側システム装置のSDTPインタフェース識別番号	*1
4.	送信先識別子 Destination identification	データ発生元のSDTPインタフェース識別番号	*1
5.	接続先ホスト名 Connection destination host name (IPアドレス) (IP address)	接続先装置のホスト名または、IPアドレス Host name or IP address of the connected device	*2

*1: 送信元識別子、送信先識別子は「DIOSAインタフェース仕様 宇宙データ転送プロトコル (SDTP) OSO 501-X (個別規定) <http://c-soda.isas.jaxa.jp/sdg/document/document.htm>」を参照。
*2: 接続先装置のホスト名 (IPアドレス) 識別子。SDTP OSO 501-X (個別規定) 参照。
UNIX系は、/etc/hosts、Windows系は、C:\WINDOWS\system32\drivers\etc\hostsファイルに定義しておくこと。(IPアドレスの場合は定義不要。)

Host name of the connected device (IP address)

UNIX type should be defined in /etc/hosts, Windows type in C:\WINDOWS\system32\drivers\etc\hosts file. (In case of IP address, definition is unnecessary.)

7.3. ファイルイメージ File Image

本ファイルのイメージを図 7-1に示す。The image of this file is shown in Figure 7-1

Connection destination		Source Identifier	Destination Identifier		
#接続先識別	ポート番号	送信元識別子	送信先識別子	接続先ホスト名	
dst	10000	999999999999	999999999999	XXXXXXX	
stg	20000	999999999999	999999999999	XXX.XXX.XXX.XXX	
sir	30200	999999999999	999999999999	XXXXXXX	

※未使用機器は定義不要

Unused devices need not be defined

図 7-1 ファイルイメージ

Figure 7-1 File Image

7.4. 既存ファイルからの変更

Change from the existing file
The correspondence when changing the old environment definition file to the new environment definition file is shown below.

①は①'、②は②'、③は③'に対応し、変更箇所は接続先識別のみである。

① corresponds to ①', ② corresponds to ②', ③ corresponds to ③'. And the changed part is only the connection destination identification.

● 旧環境定義ファイル(データ分配・データ蓄積アクセス用) old environment definition file (Data allocation for data storage access)

	#モード mode	ポート番号 port number	送信元識別子 source identifier	送信先識別子 destination identifier	接続先ホスト名 connection destination host name
①	1	10000	999999999999	999999999999	XXXXXXX · host name
②	2	20000	999999999999	999999999999	XXX.XXX.XXX.XXX

Old environment definition file (Data allocation, For SIRIUS access)
● 旧環境定義ファイル(データ分配・SIRIUSアクセス用)

	#モード mode	ポート番号 port number	送信元識別子 source identifier	送信先識別子 destination identifier	接続先ホスト名 connection destination host name
①	1	10000	999999999999	999999999999	XXXXXXX host name
③	2	30200	999999999999	999999999999	XXXXXXX

● 新環境定義ファイル New environment definition file

	接続先識別 connection destination identification	ポート番号 port number	送信元識別子 source identifier	送信先識別子 destination identifier	接続先ホスト名 connection destination host name
①	dst	10000	999999999999	999999999999	XXXXXXX destination
②'	stg	20000	999999999999	999999999999	XXX.XXX.XXX.XXX host name
③'	sir	30200	999999999999	999999999999	XXXXXXX

Source identifier Destination identifier