Open Printing Status Monitoring Application Program Interface Specification

Draft Version 2004-05-22

Open Printing Status Monitoring Application Interface Specification

Table of Contents

1. Notation and Terminology

1.1 Notation Conventions

This section describes the use of courier, italic and bold fonts in this document.

Font	Description	Examples
`Courier`	Definition of functions.	`void fsgsmDestroy(FSGSMCtx *pFSGSMCtx);`
	Name of functions.	`select()`
	Source code examples.	`#include <fsgsm.h>` `FSGSMCtx *pFSGSMCtx = fsgsmNew("sample", 0, 1, NULL);`
Italic	Arguments of the function used in the description.	*nBufBytes*
	File, directory or other resource names.	*/var/log/fsgsm/printer*
Bold	Emphasis	

1.2 Conformance Terminology

In this document, capitalized terms, such as: MUST, MUST NOT, REQUIRED, SHOULD, SHOULD NOT, MAY, and OPTIONAL, are intended to be interpreted as described in [RFC2119].

1.3 Other Terminology

The following table defines special terms used in this document.

Term	Description
Status Monitoring module	A shared library or an individual program which provides a set of the Status Monitoring API functions.
Status Monitoring library	A Status Monitoring module as a shared library being linked by the Stub.
Status Monitoring program	A Status Monitoring module as an individual program which communicates with the Stub via an inter-process communication interface.
Stub	A library which connects the Monitor and the Status Monitoring module, and provides the Status Monitoring API entries to the Monitor.
Monitor	A program which calls the Status Monitoring API entries provided by the Stub.
Status Monitoring Object	An object which keeps a data set created by the Stub for controlling the Status Monitoring module.
Status Monitoring process	A process of the Status Monitoring program.

2. Introduction

2.1 Summary

This document defines the API for obtaining the printer status data from the printer in the standard data format, and the API for writing the printer command data to the printer as well. This document also defines the standard data format to describe the printer status data.

2.2 Architecture Overview

The following figure shows the modules related to the Status Monitoring API.

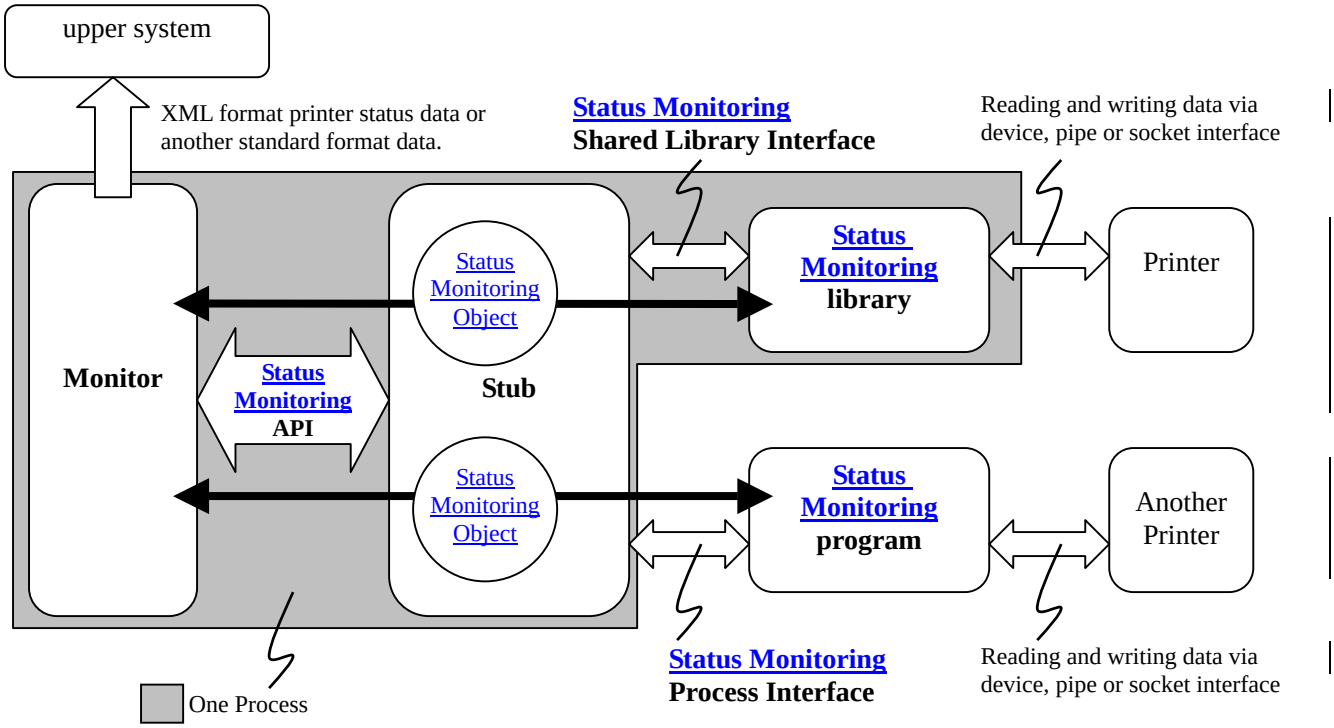

Fig.1 Architecture Overview

The Stub is a library which connects the Monitor and the Status Monitoring module, and provides the Status Monitoring API entries to the Monitor. To obtain the printer status from the printer, the Monitor creates the Status Monitoring Object with the Status Monitoring API. The Status Monitoring Object is an object which keeps a data set for controlling the Status Monitoring module. The Monitor can create multiple Status Monitoring Objects, however one Status Monitoring Object can deal with only one Status Monitoring module. The Status Monitoring library is a Status Monitoring module provided as a shared library. The Stub loads and links the Status Monitoring library when creating the Status Monitoring Object, and communicates with it via the Status Monitoring Shared Library Interface. Therefore, the Monitor, the Stub and the Status Monitoring library run as the same process. Another type of Status Monitoring module is the Status Monitoring Program that is an individual program loaded and executed by the Stub when creating the Status Monitoring Object. The Status Monitoring program communicates with the Stub via the Status Monitoring Process Interface.

The Monitor obtains the printer status data from the Status Monitoring module at an appropriate interval time. Raw printer status data just obtained from the printer via a device, pipe or socket interface depends on each printer model or protocol. The Status Monitoring module converts the raw printer status data into the device independent XML format status data, and the Monitor can obtain it via the Status Monitoring API. Furthermore, the Monitor can send the XML format printer status data to an upper system, such as an application program or a spooler program, and the upper system can parse and use the data. In other cases, the Monitor can parse the XML format printer status data by itself and convert it into another standard format data, such as IPP, and send it to the upper system, so that the upper system can deal with the device independent printer status data.

The following section in this document describes the Status Monitoring API, Status Monitoring Shared Library Interface, Status Monitoring Process Interface and the Printer Status Data Format in detail.

3. Status Monitoring API

3.1 fsgsmNew

Name

fsgsmNew – Create a new Status Monitoring Object. <This function MUST be supported.>

Synopsis

```
FSGSMCtx *fsgsmNew(char *pName, int fdRead, int fdWrite, char *pURI);
```

Arguments

pName – Name of the Status Monitoring library or Status Monitoring program.
fdRead – File descriptor for reading the printer status data from the Status Monitoring module.
fdWrite – File descriptor for writing the printer command data to the Status Monitoring module.
pURI – Device URI.

Description

- This function creates the Status Monitoring Object specified by *pName* and return the printer to it.
  pName で指定される Status Monitoring object を生成してそのポインタを返す。
- First, this function MUST assume *pName* as the name of the Status Monitoring library. The appropriate prefix (e.g. lib*) and postfix (e.g. .so) for the library name for each operating system are automatically appended in this function, and this function tries to load and link the Status Monitoring library from the appropriate library path under each operating system.
  本関数では、先ず pName に library type の Status Monitoring module が指定されたとみなして処理する。動作環境におけるライブラリ名の先頭の定型文字列（lib など）および末尾の定型文字列（.so）は、本関数の内部処理において自動的に付加し、動作環境での適切な library path からの Status Monitoring library module の load と link を試みる。
- If this function fails to load or link to the Status Monitoring library, this function MUST assume *pName* as the name of the Status Monitoring program, and this function tries to load and execute the Status Monitoring program from the appropriate executable path under each operating system.
  library type の Status Monitoring module の load または link に失敗した場合は、次に pName に process type の Status Monitoring module が指定されたとみなして処理する。process type の場合は pName に与えられた文字列をそのまま Status Monitoring module のファイル名としてみなし、動作環境での適切な実行 path からの Status Monitoring module の実行を試みる。
- This function MAY be blocked until all the initialization procedures of the Status Monitoring module are completed, or return a NULL pointer immediately without the initialization of it after receiving the signal.
  一連の処理が完了するまで本関数はブロックするが、シグナルの受信によって処理未完了のまま本関数から戻る場合がある。その場合、本関数の戻り値は NULL となる。
- This function MAY be blocked depending on the connection process between the Status Monitoring module and the printer. The Monitor SHOULD set the alarm signal or other time out mechanism before calling this function.
  Status Monitoring module がプリンタとのコネクションを確立する処理によっては、本関数は長時間ブロックする可能性がある。呼び元が一定時間内の応答を規定したい場合は、本関数を呼ぶ前にアラームを設定する等する。
- The writing functionality of the Status Monitoring module is optional. The Monitor can know whether the Status Monitoring module supports the writing functionality or not by the `fsgsmGetCap()` function. When the writing functionality is not supported, the Monitor MUST write the printer command data directly to the printer via an appropriate file descriptor.
  Status Monitoring の write function はオプション。Status Monitoring 側が write function をサポートするか否かは fsgsmGetCap 関数によって呼び元が判別する。write function がサポートされない場合は呼び元が適切な file descriptor を用いて直接 printer command data を write する必要がある。
- *pURI* is OPTIONAL. NULL for *pURI* means that the device URI is unknown and the way of how to deal with the device URI depends on the implementation of each Status Monitoring module.
  pURI で指定するデバイスの URI はオプション。URI が不明の場合等は NULL の指定を許す。NULL が指定された場合の Status Monitoring 側の処理は実装依存とする。

Return Value

Pointer to the Status Monitoring Object or NULL when error.

3.2 fsgsmDestroy

Name

fsgsmDestroy – Destroy the Status Monitoring Object. <This function MUST be supported.>

Synopsis

```
void fsgsmDestroy(FSGSMCtx *pFSGSMCtx);
```

Arguments

pFSGSMCtx – Pointer to the Status Monitoring Object.

Description

- When the Status Monitoring Object specified by *pFSGSMCtx* is the Status Monitoring program, this function kills it and destroys the Status Monitoring Object.
- When the Status Monitoring Object specified by *pFSGSMCtx* is the Status Monitoring library, this function unlinks it and destroys the Status Monitoring Object.
  pFSGSMCtx によって指定した Status Monitoring Object によって指定される Status Monitoring module が process type の場合はそれを kill し、library type の場合はそれを unlink した後、Status Monitoring object を解放する。
- This function MAY be blocked depending on the disconnection process between the Status Monitoring module and the printer. The Monitor SHOULD set the alarm signal or other time out mechanism before calling this function.
  Status Monitoring module がプリンタとのコネクションを解放する処理によっては、本関数は長時間ブロックする可能性がある。呼び元が一定時間内の応答を規定したい場合は、本関数を呼ぶ前にアラームを設定する等する。

Return Value

None

3.3 fsgsmGetCap

Name

fsgsmGetCap – Get the Status Monitoring Object capabilities. <This function MUST be supported.>

Synopsis

```
int fsgsmGetCap(FSGSMCtx *pFSGSMCtx, FSGSMCap cap);
```

Arguments

pFSGSMCtx – Pointer to the Status Monitoring Object.
cap – Enum of the Status Monitoring Object capabilities. This function MUST support the following enums.

cap	Value	Description
FSGSM_CAP_WRITE	1	Supports the `fsgsmStartWrite()`, `fsgsmEndWrite()` and the `fsgsmWrite()` functions.
FSGSM_CAP_JOB	2	Supports the `fsgsmStartJob()`, `fsgsmEndJob()` and the `fsgsmCancelJob()` functions.
FSGSM_CAP_CTRL	3	Supports the `fsgsmCtrl()` function.

Description

- This function returns the value which indicates whether the condition specified by *cap* is true or false.
  pFSGSMCtx によって指定した Status Monitoring Object によって指定される Status Monitoring module において、cap で指定した条件が正しいか否かを返す。
- The Status Monitoring program MUST support the `fsgsmStartWrite()`, `fsgsmEndWrite()` and the `fsgsmWrite()` functions. This function MUST return FSGSM_TRUE when the Status Monitoring Object specified by *pFSGSMCtx* is the one for the Status Monitoring program.
  process type の Status Monitoring module の場合、fsgsmStartWrite, fsgsmEndWrite, fsgsmWrite 関数のサポートは必須とし、cap に FSGSM_CAP_WRITE が指定されて本関数が呼ばれた場合は、必ず FSGSM_TRUE を返さなければならないものとする。
- This function MUST NOT be blocked.
  本関数はブロックしない。

Return Value

FSGSM_TRUE	The condition specified by *cap* is true. cap で指定した条件が正
FSGSM_FALSE	The condition specified by *cap* is false. cap で指定した条件が誤
FSGSM_ERROR	Any error. エラー

The Monitor MUST destroy the Status Monitoring Object when this function returns FSGSM_ERROR.
FSGSM_ERROR 発生時は、呼び元は fsgsmDestroy 関数によって Status Monitoring Object を破棄しなければならない。

3.4 fsgsmStartJob

Name

fsgsmStartJob – Start the printing job sequence. <This function is OPTIONAL.>

Synopsis

```
int fsgsmStartJob(FSGSMCtx *pFSGSMCtx, int idJob);
```

Arguments

pFSGSMCtx – Pointer to the Status Monitoring Object.
idJob – Job ID.

Description

- The Monitor declares the start of the printing job to the Status Monitoring module specified by *pFSGSMCtx*.
  pFSGSMCtx によって指定した Status Monitoring Object によって指定される Status Monitoring module に対してジョブの開始を通知する。
- This function MUST be called before starting printing job by this function and after the end of printing job by calling the `fsgsmEndJob()` function, or return FSGSM_ERROR.
  本関数が呼ばれた後、fsgsmEngJob 関数によってジョブを終了する前に再び fsgsmStartJob 関数が呼ばれた場合はエラーとなり、戻り値として FSGSM_ERROR を返す。
- This function MUST NOT be blocked.
  本関数はブロックしない。

Return Value

FSGSM_OK	Success. 正常終了
FSGSM_EPROGRESS	The printing job specified *idJob* is in progress and cannot start a new job with the same job ID. 指定したジョブが処理中
FSGSM_ERROR	Other errors. その他のエラー

The Monitor MUST destroy the Status Monitoring Object when this function returns FSGSM_ERROR.
FSGSM_ERROR 発生時は、呼び元は fsgsmDestroy 関数によって Status Monitoring Object を破棄しなければならない。

3.5 fsgsmEndJob

Name

fsgsmEndJob – Stop the printing job sequence. <This function is OPTIONAL. If the `fsgsmStartJob()` function is supported, this function MUST be supported.>

Synopsis

```
int fsgsmEndJob(FSGSMCtx *pFSGSMCtx);
```

Arguments

pFSGSMCtx – Pointer to the Status Monitoring Object.

Description

- The Monitor checks the end of the printing job to the Status Monitoring module specified by *pFSGSMCtx* by this function.
  pFSGSMCtx によって指定した Status Monitoring Object によって指定される Status Monitoring module に対してジョブの終了を通知する。
- The Monitor MUST call this function after finishing or canceling a printing job.
  本関数は、fsgsmStartJob 関数が呼ばれた後の一連のジョブの処理完了後、必ず呼ばれなければならない。
- This function returns FSGSM_ERROR if the `fsgsmStartJob()` function is not called by the Monitor or a printing job is not executed prior to calling this function.
  fsgsmStartJob 関数が呼ばれていない状態で本関数が呼ばれた場合はエラーとなり、戻り値として FSGSM_ERROR を返す。
- This function MUST NOT be blocked.
  本関数はブロックしない。

Return Value

FSGSM_OK	Success. 正常終了
FSGSM_EPROGRESS	The printing job specified *idJob* is in progress. ジョブが処理中で終了できない
FSGSM_ERROR	Other errors. その他のエラー

The Monitor MUST destroy the Status Monitoring Object when this function returns FSGSM_ERROR.
FSGSM_ERROR 発生時は、呼び元は fsgsmDestroy 関数によって Status Monitoring Object を破棄しなければならない。
Whether this function returns FSGSM_EPROGRESS or not depends on the specification of each Status Monitoring module.
When this function returns FSGSM_EPROGRESS, the Monitor MUST call this function again to try to check the end of the

printing job after the specified interval time. The specified interval time depends on the implementation. The Status Monitoring module MUST provide all the same functionalities, including the reading printer status data by the `fsgsmRead()` function, that the Status Monitoring module provides before FSGSM_EPROGRESS occurs.
FSGSM_EPROGRESS が発生するか否かは、Status Monitoring module の仕様に依存する。尚、本エラー発生時は、呼び元は一定時間経過後、再度本関数を呼んでジョブの終了を試みなければならない。また Status Monitoring module が本エラーを発生する場合は、その発生後も fsgsmRead 関数によるプリンタステータスデータの取得等の全ての処理を、本エラー発生前と変わりなく継続して提供しなければならない。

3.6 fsgsmCancelJob

Name

fsgsmCancelJob – Cancel the printing job sequence. <This function is OPTIONAL. If the `fsgsmStartJob()` function is supported, this function MUST be supported.>

Synopsis

```
int fsgsmCancelJob(FSGSMCtx *pFSGSMCtx, int idJob);
```

Arguments

pFSGSMCtx – Pointer to the Status Monitoring Object.
idJob – Job ID.

Description

- The Monitor declares the cancellation of the printing job specified by *idJob* to the Status Monitoring module specified by *pFSGSMCtx*.
  pFSGSMCtx によって指定した Status Monitoring Object によって指定される Status Monitoring module に対して、idJob で指定したジョブの中断を通知する。
- This function MUST NOT be blocked.
  本関数はブロックしない。

Return Value

FSGSM_OK	Success. 正常終了
FSGSM_ENOJOB	The printing job specified by the *idJob* is no longer in progress. Status Monitoring module にジョブが存在しない
FSGSM_EPROGRESS	The printing job specified by the *idJob* is in progress and cannot cancel the job. ジョブが中断できない
FSGSM_ERROR	Other errors. その他のエラー

The Monitor MUST destroy the Status Monitoring Object when this function returns FSGSM_ERROR.
FSGSM_ERROR 発生時は、呼び元は fsgsmDestroy 関数によって Status Monitoring Object を破棄しなければならない。
Whether this function returns FSGSM_ENOJOB or not depends on the specification of each Status Monitoring module. If the Status Monitoring module has the capability of receiving the multiple printing jobs, the Status Monitoring module SHOULD retains each job ID for each printing job, and cancel only the printing job specified by *idJob* when the Monitor calls this function, or return FSGSM_ENOJOB when the job is no longer in progress or *idJob* is a nonexistent job ID. If the Status Monitoring module does not have the capability of receiving the multiple printing jobs, the Status Monitoring module retains only one job ID for one printing job at the same time, and cancel the printing job specified by *idJob* when the Monitor calls this function, or return FSGSM_ENOJOB when the job is no longer in progress or *idJob* is a nonexistent job ID. The way of how to deal with FSGSM_ENOJOB depends on the Monitor.
FSGSM_ENOJOB が発生するか否かは、Status Monitoring module の仕様に依存する。複数のジョブをスプール可能な Status Monitoring module の場合、Status Monitoring module はスプールおよび印刷中のジョブの数分だけ Job ID を保持し、それらのうちの1つが指定された場合は対応するジョブを中断し、それら以外の Job ID が指定された場合は、本エラーを返さなければならない。また、単一のジョブのみ処理が可能な Status Monitoring module は単一の Job ID を保持し、その Job ID が指定された場合は処理中のジョブを中断し、それ以外の Job ID が指定された場合は本エラーを返さなければならない。尚、本エラー発生時の呼び元の処理は、実装依存とする。
This function returns FSGSM_EPROGRESS when the printing job specified by *idJob* is in progress and the Status Monitoring module cannot cancel the job. The way of how to deal with FSGSM_EPROGRESS depends on the Monitor.
FSGSM_EPROGRESS は、Status Monitoring module 側に idJob に対応するジョブが存在するにも関わらず、ジョブが中断できなかった場合に発生する。本エラー発生時の呼び元の処理は、実装依存とする。

3.7 fsgsmGetReadFD

Name

fsgsmGetReadFD – Get the file descriptor for reading from the Status Monitoring Object. <This function MUST be supported.>

Synopsis

```
int fsgsmGetReadFD(FSGSMCtx *pFSGSMCtx);
```

Arguments

pFSGSMCtx – Pointer to the Status Monitoring Object.

Description

- This function returns the file descriptor for reading the printer status data from the Status Monitoring Object specified by *pFSGSMCtx*.
  pFSGSMCtx によって指定した Status Monitoring Object によって指定される Status Monitoring module の read 用 file descriptor を返す。
- The file descriptor obtained from this function is valid until the Monitor destroys the Status Monitoring Object.
  取得した file descriptor は fsgsmDestroy 関数によって Status Monitoring Object を破棄するまでの間有効。
- The Monitor can use the file descriptor obtained from this function for the UNIX `select()` function to determine its own processing.
  呼び元は本関数によって取得した file descriptor を用いて select 関数による read 処理への振り分けを行うことができる。
- This function MUST NOT be blocked.
  本関数はブロックしない。

Return Value

File descriptor.

FSGSM_ERROR	Any error. エラー

The Monitor MUST destroy the Status Monitoring Object when this function returns FSGSM_ERROR.
FSGSM_ERROR 発生時は、呼び元は fsgsmDestroy 関数によって Status Monitoring Object を破棄しなければならない。

3.8 fsgsmStartRead

Name

fsgsmStartRead – Start the reading sequence. <This function MUST be supported.>

Synopsis

```
int fsgsmStartRead(FSGSMCtx *pFSGSMCtx, FSGSMReadMode idReadMode, char *pLang);
```

Arguments

pFSGSMCtx – Pointer to the Status Monitoring Object.
idReadMode – Enum of the Status Monitoring Object reading mode.
pLang – Pointer to the language string.

Description

- The Monitor declares the start of reading the printer status data to the Status Monitoring module specified by *pFSGSMCtx*.
  pFSGSMCtx によって指定した Status Monitoring Object によって指定される Status Monitoring module からの、プリンタステータスデータの read の開始を Status Monitoring module に通知する。
- *idReadMode* is the enum of the reading mode. This function SHOULD support the following enums.

idReadMode	Value	Description
FSGSM_READ_PRT_MIB_ALL	255	Read all the printer status data.
FSGSM_READ_PRT_MIB_SUMMARY	1	Read the summarized printer status data.

- The Monitor MUST set the “language” , ”region” and “character code” string, (e.g. “ja_JP.UTF-8”) defined by the locale specification to *pLang*. The Status Monitoring module MUST generate the printer status data according to the specified language, region and character code. The Status Monitoring module MUST support UTF-8 at least. The way of how to deal with the other character codes, (e.g. ja_JP.eucJP) depends on each Status Monitoring module implementation.
  pLang には、”ja_JP.UTF-8”のようなロケール指定文字列と同様の言語＋地域＋文字コード指定文字列を指定する。Status Monitoring module は、pLang によって指定された言語および地域情報に従ってプリンタステータスデータを生成する。また、Status Monitoring module は、少なくとも UTF-8 の文字コードに従ったステータスデータを生成しなければならない。pLang に、”ja_JP.eucJP”のような UTF-8 とは異なる文字コード指定を含んだ文字列が与えられた場合、Status Monitoring module が指定された文字コードでステータスデータを生成するか否かは実装依存とする。
- The Status Monitoring module MUST generate the printer status data in English when the Monitor set a NULL value to *pLang* or when the Status Monitoring module cannot generate the printer status data according to the language and region specified by pLang.
  pLang に NULL が指定された場合や、pLang によって指定された言語および地域に従ったプリンタステータスデータの生成が不可能な場合は、Status Monitoring module は少なくとも英語によるプリンタステータスデータを生成しなければならな

い。

- The Status Monitoring module MUST establish and retain the printer status data to be read by the Monitor via the `fsgsmRead()` function.
  Status Monitoring module は本関数が呼ばれたタイミングで、呼び元に返すプリンタステータスデータを確定する。
- When the Status Monitoring module is a Status Monitoring program, the Status Monitoring program MUST write 0(one byte) into the Data Reading Pipe after the Status Monitoring program establishes and retains the printer status data, and this function MUST first read the above one byte from the Data Reading Pipe specified by the file descriptor which can be obtained by the `fsgsmGetReadFD()` function.
  process type の Status Monitoring の場合、Status Monitoring module はプリンタステータス送出の準備が出来た時点で、それを表す 1 バイトの値0を呼び元の read 用の file descriptor に繋がるパイプに書き込まなければならない。呼び元にリンクする fsgsmStartRead 関数の内部処理では、fsgsmGetReadFD 関数によって取得できる file descriptor から上記 1 バイトの値0を読み捨てなければならない。
- This function MUST return FSGSM_ERROR when the Monitor calls the following functions before the Monitor calls the `fsgsmEndRead()` function.
  `fsgsmStartRead()`, `fsgsmStartWrite()`, `fsgsmEndWrite()`
  本関数は必ず fsgsmEndRead 関数と対で呼ばれなければならない。本関数が呼ばれてから fsgsmEndRead 関数が呼ばれるまでの間に次の関数が呼ばれた場合はエラーとなり、戻り値として FSGSM_ERROR を返す。
  fsgsmStartRead, fsgsmStartWrite, fsgsmEndWrite
- This function MAY be blocked until the Status Monitoring module establishes and retains the printer status data, or return a FSGSM_INTR error caused by receiving the signals.
  プリンタステータスデータが確定するまで本関数はブロックするが、シグナルの受信によって処理の途中で本関数から戻る場合には、戻り値として FSGSM_EINTR を返す。
- This function MAY be blocked depending on the connection process between the Status Monitoring module and the printer. The Monitor SHOULD set the alarm signal or other time out mechanism before calling this function.
  Status Monitoring module がプリンタとのコネクションを確立する処理によっては、本関数は長時間ブロックする可能性がある。呼び元が一定時間内の応答を規定したい場合は、本関数を呼ぶ前にアラームを設定する等する。

Return Value

FSGSM_OK	Success. 正常終了
FSGSM_EINTR	Receive a signal. シグナル受信
FSGSM_ERROR	Other errors. その他のエラー

The Monitor MUST destroy the Status Monitoring Object when this function returns FSGSM_ERROR.
FSGSM_ERROR 発生時は、呼び元は fsgsmDestroy 関数によって Status Monitoring Object を破棄しなければならない。

3.9 fsgsmRead

Name

fsgsmRead – Read the printer status data from the Status Monitoring Object. <This function MUST be supported.>

Synopsis

```
int fsgsmRead(FSGSMCtx *pFSGSMCtx, void *pBuf, int nBufBytes);
```

Arguments

pFSGSMCtx – Pointer to the Status Monitoring Object.
pBuf – Pointer to the buffer for storing the printer status data.
nBufBytes – Number of bytes for reading the printer status data.

Description

- This function obtains the printer status data from the Status Monitoring Object specified by *pFSGSMCtx* and store the data into the buffer specified by *pBuf*. The data format of the printer status data is defined in the “Printer Status Data Format” section.
- pFSGSMCtx によって指定した Status Monitoring Object によって指定される Status Monitoring module から、プリンタステータスデータを取得して、pBuf で指定したバッファにデータを格納する。プリンタステータスデータのデータフォーマットについては、Printer Status Data Format のセクションを参照。
- When the length of the printer status data is longer than the length of the buffer specified by *nBufBytes*, this function read the *nBufBytes* bytes data from the top of the printer status data first and store it at the top of the buffer, and next time this function is called, this function reads the remaining printer status data from the next byte of the previous read data and stores it in the next address of the end of the previous written data into the buffer.
- 指定したバッファのバイト数を越えるプリンタステータスデータがある場合には、格納できるバイト数だけ pBuf の先頭から格納し、次に再度本関数が呼ばれた際には、その次のバイト位置のプリンタステータスデータから、格納できるバイト数だけ

pBuf の先頭から格納する。

- When the remaining printer status data is shorter than the length specified by *nBufBytes*, this function returns the specified shorter byte number instead of the *nBufBytes*, or returns zero when no printer status data remains.
  nBufBytes で指定した値よりもプリンタステータスデータのバイト数が小さいような場合は、nBufBytes で指定した値よりも小さい値を返す場合がある。その後更に本関数を呼ぶと、戻り値として 0 を返す。
- When receiving the signals, this function MAY read and store the printer status data which is shorter than the length specified by *nBufBytes*, and return the shorter length of the data that this function actually read and stored.
  プリンタステータスデータの read が完了するまで本関数はブロックするが、シグナルの受信によってプリンタステータスデータの read の途中で本関数から戻る場合には、nBufBytes で指定した値よりも小さい値を返す場合がある。
- This function returns FSGSM_EINTR when no printer status data is read regardless of the printer status data that should be read retains.
  1 バイトも read せずにシグナルを受信した場合は、戻り値として FSGSM_EINTR を返す。
- This function MAY be blocked depending on the connection process between the Status Monitoring module and the printer. The Monitor SHOULD set the alarm signal or other time out mechanism before calling this function.
  Status Monitoring module がプリンタとのコネクションを確立する処理によっては、本関数は長時間ブロックする可能性がある。呼び元が一定時間内の応答を規定したい場合は、本関数を呼ぶ前にアラームを設定する等する。
- The Monitor MUST call this function between the `fsgsmStartRead()` and `fsgsmEndRead()` functions. This function MUST return FSGSM_ERROR when this function is not called between the `fsgsmStartRead()` and `fsgsmEndRead()` functions.
  本関数は fsgsmStartRead 関数と fsgsmEndRead 関数の間で呼ばれなければならない。その範囲外で呼ばれた場合は、戻り値として FSGSM_ERROR を返す。

Return Value

Number of bytes of the printer status data read, or 0 when no data remains.

FSGSM_EINTR	Receive a signal with no data read. 0 バイト read でのシグナル受信
FSGSM_ERROR	Other errors. その他のエラー

3.10 fsgsmEndRead

Name

fsgsmEndRead – Stop the reading sequence. <This function MUST be supported.>

Synopsis

```
int fsgsmEndRead(FSGSMCtx *pFSGSMCtx);
```

Arguments

pFSGSMCtx – Pointer to the Status Monitoring Object.

Description

- The Monitor declares the end of reading the printer status data to the Status Monitoring module specified by *pFSGSMCtx*.
  pFSGSMCtx によって指定した Status Monitoring Object によって指定される Status Monitoring module からの、プリンタステータスデータの read の完了を Status Monitoring module に通知する。
- The Status Monitoring module can update the printer status data after this function is called by the Monitor.
  Status Monitoring module は本関数が呼ばれた後、呼び元に返すプリンタステータスデータの更新を再開して良い。
- The Monitor MUST call this function after calling the `fsgsmStartRead()` function.
  本関数は、fsgsmStartRead 関数が呼ばれた後の一連の read 処理完了後、必ず呼ばれなければならない。
- This function returns FSGSM_ERROR if the `fsgsmStartRead()` function is not called by the Monitor prior to calling this function.
  fsgsmStartRead 関数が呼ばれていない状態で本関数が呼ばれた場合はエラーとなり、戻り値として FSGSM_ERROR を返す。
- This function MUST NOT be blocked.
  本関数はブロックしない。

Return Value

FSGSM_OK	Success. 正常終了
FSGSM_ERROR	Any error. エラー

The Monitor MUST destroy the Status Monitoring Object when this function returns FSGSM_ERROR.
FSGSM_ERROR 発生時は、呼び元は fsgsmDestroy 関数によって Status Monitoring Object を破棄しなければならない。

3.11 fsgsmGetWriteFD

Name

fsgsmGetWriteFD – Get the file descriptor for writing to the Status Monitoring Object. <This function MUST be supported.>

Synopsis

```
int fsgsmGetWriteFD(FSGSMCtx *pFSGSMCtx);
```

Arguments

pFSGSMCtx – Pointer to the Status Monitoring Object.

Description

- This function returns the file descriptor for writing the printer command data to the Status Monitoring Object specified by *pFSGSMCtx*.
  pFSGSMCtx によって指定した Status Monitoring Object によって指定される Status Monitoring module の write 用 file descriptor を返す。
- The file descriptor obtained from this function is valid until the Monitor destroys the Status Monitoring Object.
  取得した file descriptor は fsgsmDestroy 関数によって Status Monitoring Object を破棄するまでの間有効。
- The Monitor can use the file descriptor obtained from this function for the UNIX `select()` function to determine its own processing.
  呼び元は本関数によって取得した file descriptor を用いて select 関数による write 処理への振り分けを行うことができる。
- This function MUST NOT be blocked.
  本関数はブロックしない。

Return Value

File descriptor.

FSGSM_ERROR	Any error. エラー

The Monitor MUST destroy the Status Monitoring Object when this function returns FSGSM_ERROR.
FSGSM_ERROR 発生時は、呼び元は fsgsmDestroy 関数によって Status Monitoring Object を破棄しなければならない。

3.12 fsgsmStartWrite

Name

fsgsmStartWrite – Start the writing sequence. <This function is OPTIONAL. If the `fsgsmWrite()` function is supported, this function MUST be supported.>

Synopsis

```
int fsgsmStartWrite(FSGSMCtx *pFSGSMCtx);
```

Arguments

pFSGSMCtx – Pointer to the Status Monitoring Object.

Description

- The Monitor declares the start of writing the printer command data to the Status Monitoring module specified by *pFSGSMCtx*.
  pFSGSMCtx によって指定した Status Monitoring Object によって指定される Status Monitoring module への、データの write の開始を Status Monitoring module に通知する。
- This function MUST return FSGSM_ERROR when the Monitor calls the following functions before the Monitor calls the `fsgsmEndWrite()` function.
  `fsgsmStartRead()`, `fsgsmStartWrite()`, `fsgsmEndRead()`
  本関数は必ず fsgsmEndWrite 関数と対で呼ばれなければならない。本関数が呼ばれてから fsgsmEndWrite 関数が呼ばれるまでの間に次の関数が呼ばれた場合はエラーとなり、戻り値として FSGSM_ERROR を返す。
  fsgsmStartRead, fsgsmStartWrite, fsgsmEndRead
- This function MAY be blocked depending on the connection process between the Status Monitoring module and the printer. The Monitor SHOULD set the alarm signal or other time out mechanism before calling this function. This function returns FSGSM_EINTR when receiving the signals.
- Status Monitoring module がプリンタとのコネクションを確立する処理によっては、本関数は長時間ブロックする可能性がある。呼び元が一定時間内の応答を規定したい場合は、本関数を呼ぶ前にアラームを設定する等する。シグナルの受信によって処理の途中で本関数から戻る場合には、戻り値として FSGSM_EINTR を返す。

Return Value

FSGSM_OK	Success. 正常終了
FSGSM_EINTR	Receive a signal. シグナル受信

FSGSM_ERROR Other errors.
その他のエラー

The Monitor MUST destroy the Status Monitoring Object when this function returns FSGSM_ERROR.
FSGSM_ERROR 発生時は、呼び元は fsgsmDestroy 関数によって Status Monitoring Object を破棄しなければならない。

3.13 fsgsmWrite

Name

fsgsmWrite – Write data to the Status Monitoring Object. <This function is OPTIONAL.>

Synopsis

```
int fsgsmWrite(FSGSMCtx *pFSGSMCtx, void *pBuf, int nBufBytes);
```

Arguments

pFSGSMCtx – Pointer to the Status Monitoring Object.
pBuf – Pointer to the buffer for writing data.
nBufBytes – Number of bytes of the writing data.

Description

- This function reads the printer command data from the buffer specified by *pBuf* and write it into the Status Monitoring Object specified by *pFSGSMCtx*. The length of the reading and writing printer command data is specified by *nBufBytes*.
  pFSGSMCtx によって指定した Status Monitoring Object によって指定される Status Monitoring module へ、pBuf で指定したバッファ中のデータを nBufBytes で指定したバイト数だけ書き込む。
- When receiving the signals, this function MAY read and write the printer command data which is shorter than the length specified by *nBufBytes*, and return the shorter length of the data that this function actually read and wrote. The Monitor MUST call this function again to write the rest of the printer command data into the Status Monitoring Object with the appropriate address in the buffer and the appropriate length of the rest of the printer command data.
  データの write が完了するまで本関数はブロックするが、シグナルの受信によってデータの write の途中で本関数から戻る場合には、nBufBytes で指定した値よりも小さい値を返す場合がある。その際、残りのデータを書き込むためには、呼び元がバッファの適切なバイト位置から、本関数によって再度データを書き込む必要がある。
- This function returns FSGSM_EINTR when no printer command data is written regardless of the printer command data that should be written retains.
  1 バイトも write せずにシグナルを受信した場合は、戻り値として FSGSM_EINTR を返す。
- This function MAY be blocked depending on the connection process between the Status Monitoring module and the printer. The Monitor SHOULD set the alarm signal or other time out mechanism before calling this function.
  Status Monitoring module がプリンタとのコネクションを確立する処理によっては、本関数は長時間ブロックする可能性がある。呼び元が一定時間内の応答を規定したい場合は、本関数を呼ぶ前にアラームを設定する等する。
- The Monitor MUST call this function between the `fsgsmStartWrite()` and `fsgsmEndWrite()` functions. This function MUST return FSGSM_ERROR when this function is not called between the `fsgsmStartWrite()` and `fsgsmEndWrite()` functions.
  本関数は fsgsmStartWrite 関数と fsgsmEndWrite 関数の間で呼ばれなければならない。その範囲外で呼ばれた場合は、戻り値として FSGSM_ERROR を返す。

Return Value

Number of bytes of the writing data written or 0 when no data written.
FSGSM_EINTR Receive a signal with no data written.
0 バイト write でのシグナル受信
FSGSM_ERROR Other errors.
その他のエラー

3.14 fsgsmEndWrite

Name

fsgsmEndWrite – Stop the writing sequence. <This function is OPTIONAL. If the `fsgsmWrite()` function is supported, this function MUST be supported.>

Synopsis

```
int fsgsmEndWrite(FSGSMCtx *pFSGSMCtx);
```

Arguments

pFSGSMCtx – Pointer to the Status Monitoring Object.

Description

- The Monitor declares the end of writing the printer command data to the Status Monitoring module specified by *pFSGSMCtx*.
  pFSGSMCtx によって指定した Status Monitoring Object によって指定される Status Monitoring module への、スデータの write の完了を Status Monitoring module に通知する。
- The Monitor MUST call this function after calling the `fsgsmStartWrite()` function.
  本関数は、fsgsmStartWrite 関数が呼ばれた後の一連の write 処理完了後、必ず呼ばれなければならない。
- This function returns FSGSM_ERROR if the `fsgsmStartWrite()` function is not called by the Monitor prior to calling this function.
  fsgsmStartWrite 関数が呼ばれていない状態で本関数が呼ばれた場合はエラーとなり、戻り値として FSGSM_ERROR を返す。
- This function MUST NOT be blocked.
  本関数はブロックしない。

Return Value

FSGSM_OK	Success. 正常終了
FSGSM_ERROR	Any error. エラー

The Monitor MUST destroy the Status Monitoring Object when this function returns FSGSM_ERROR.
FSGSM_ERROR 発生時は、呼び元は fsgsmDestroy 関数によって Status Monitoring Object を破棄しなければならない。

3.15 fsgsmCtrl

Name

fsgsmCtrl – Request the extended operation. <This function is OPTIONAL.>

Synopsis

```
int fsgsmCtrl(FSGSMCtx *pFSGSMCtx, int idRequest, void *pData, int nDataBytes);
```

Arguments

pFSGSMCtx – Pointer to the Status Monitoring Object.
idRequest – Request ID.
pData – Pointer to the data for each request.
nDataBytes – Number of bytes of the data.

Description

- The Monitor requests the Status Monitoring Object specified by *pFSGSMCtx* to do the extended operation specified by *idRequest* and *nDataBytes*.
  pFSGSMCtx によって指定した Status Monitoring Object によって指定される Status Monitoring module に対して、拡張制御を要求する。
- idRequest is defined as below.

idRequest	Category
0 ～ 65535	Reserved.
65536 or above	Defined by each developer.

Return Value

Number of bytes of the data read or 0 when no data remains.

FSGSM_EINTR	Receive the signal. and the operation has been canceled. シグナル受信により処理が中断された
FSGSM_ERROR	Other errors. その他のエラー

The Monitor MUST destroy the Status Monitoring Object when this function returns FSGSM_ERROR.
FSGSM_ERROR 発生時は、呼び元は fsgsmDestroy 関数によって Status Monitoring Object を破棄しなければならない。
The data stored in the buffer specified by *pData* is not guaranteed after this function returns FSGSM_EINTR. The Monitor MUST set again the appropriate data into the buffer before calling this function.
本関数を呼ぶ前に*pData に格納したデータの内容は、FSGSM_EINTR 発生時は保証されない。正しい処理を行うためには再度*pData の内容を設定した後、本関数を呼ばなければならない。

4. Status Monitoring Shared Library Interface

4.1 Corresponding functions

The following table defines the Status Monitoring Shared Library Interface which has the corresponding functions to the Status Monitoring API. The Status Monitoring library MUST provide identical functions as the Status Monitoring Shared Library Interface.

Status Monitoring API function	Status Monitoring Shared Library Interface function
fsgsmNew()	fsgsmLibNew()
fsgsmDestroy()	fsgsmLibDestroy()
fsgsmGetCap()	fsgsmLibGetCap()
fsgsmStartJob()	fsgsmLibStartJob()
fsgsmEndJob()	fsgsmLibEndJob()
fsgsmCancelJob()	fsgsmLibCancelJob()
fsgsmStartRead()	fsgsmLibStartRead()
fsgsmGetReadFD()	fsgsmLibGetReadFD()
fsgsmRead()	fsgsmLibRead()
fsgsmEndRead()	fsgsmLibEndRead()
fsgsmStartWrite()	fsgsmLibStartWrite()
fsgsmGetWriteFD()	fsgsmLibGetWriteFD()
fsgsmWrite()	fsgsmLibWrite()
fsgsmEndWrite()	fsgsmLibEndWrite()
fsgsmCtrl()	fsgsmLibCtrl()

- The functions that the Status Monitoring API must provide also MUST be provided by the Status Monitoring library.
- The optional functions of the Status Monitoring API that the Status Monitoring library provides MUST be consistent with the return value of the `fsgsmLibGetCap()` function of the Status Monitoring Shared Library Interface.

4.2 fsgsmLibNew

Name

fsgsmLibNew – Create a new extended object that depends on each Status Monitoring library.

Synopsis

```
void *fsgsmLibNew(int fdRead, int fdWrite, char *pURI);
```

Arguments

fdRead – File descriptor for reading the printer status data.
fdWrite – File descriptor for writing the printer command data.
pURI – Device URI.

Description

- This function provides the functionality of the `fsgsmNew()` function except loading and linking the Status Monitoring library.
- This function MUST create an extended object that keeps a data set to control itself, and return a void pointer to it. Other Status Monitoring Shared Library Interface functions receive the pointer to the extended object as a first argument.

Return Value

Void pointer to the extended object or NULL when error.

4.3 Other functions

Each Status Monitoring Shared Library Interface function except the `fsgsmLibNew()` function MUST be along the following specs.

Arguments

First argument – A void pointer to the extended object created by the `fsgsmLibNew()` function.
Other arguments – Same arguments as the equivalent function of the Status Monitoring API provides.

Functionality

Provides the same functionality as the equivalent function of the Status Monitoring API provides.

Return Value

Returns the same value as the equivalent function of the Status Monitoring API returns.

5. Status Monitoring Process Interface

5.1 Pipe and Command Line Options

Status Monitoring process and Stub exchange the printer command data and printer status data via the following four pipes.
process type の Status Monitoring module と呼び元は、次の4つのパイプによりデータおよびコマンドを送受信する。

Data Writing Pipe	The pipe that the Stub sends the printer command data to the Status Monitoring process. 呼び元から Status Monitoring process へデータを送信するためのパイプ。
Data Reading Pipe	The pipe that the Status Monitoring process sends the printer status data to the Stub. Status Monitoring process から呼び元へプリンタステータスデータを送信するためのパイプ。
Command Writing Pipe	The pipe that the Stub send the request commands to the Status Monitoring process. 呼び元から Status Monitoring process へコマンドを送信するためのパイプ。
Command Reading Pipe	The pipe that the Status Monitoring process send the acknowledge commands to the Stub. Status Monitoring process から呼び元へコマンド応答を送信するためのパイプ。

- These pipes MUST be created by the Stub before the Stub. forks and executes the Status Monitoring process.
  これらのパイプは、Status Monitoring process の起動に先だって呼び元によって生成されなければならない。
- The Stub can obtain the file descriptor for the Data Reading Pipe by the `fsgsmGetReadFD()` function, and the file descriptor for the Data Writing Pipe by the `fsgsmGetWriteFD()` function.
  呼び元は Data Reading Pipe の file descriptor を fsgsmGetReadFD 関数で、Data Writing Pipe の file descriptor を fsgsmGetWriteFD 関数で取得する。
- The Command Writing Pipe and Command Reading Pipe are used inside the Stub and cannot be referred by the Monitor.
  Command Writing Pipe および Command Reading Pipe は呼び元からは直接は参照されず、Status Monitoring API の内部処理において利用される。
- The Status Monitoring process can refer each file descriptor by its arguments. The procedures in the `fsgsmNew()` function in the Stub MUST fork and execute the Status Monitoring process with the following four arguments.
  Status Monitoring process は、次のように process 起動時の引数によって、各パイプの file descriptor を受け取る。
  呼び元の内部処理では、次の4つの引数を全て指定して、Status Monitoring process を起動しなければならない。

Command Line Option	Description	MUST or OPTIONAL
--data-write-fd	Data Writing Pipe file descriptor.	MUST
--data-read-fd	Data Reading Pipe file descriptor.	MUST
--cmd-write-fd	Command Writing Pipe file descriptor.	MUST
--cmd-read-fd	Command Reading Pipe file descriptor.	MUST
--output-fd	File descriptor to write the printer command data to the printer.	MUST
--input-fd	File descriptor to read the printer status data from the printer.	MUST

- Additionally, the procedures in the `fsgsmNew()` function in the Stub MAY pass the following argument to the Status Monitoring process.
  さらに、呼び元は次の引数によって Status Monitoring process に対してプリンタの URI を渡す場合がある。

Command Line Option	Description	MUST or OPTIONAL
--printer-uri	Device URI given to the `fsgsmNew()` function. If the *pURI* for the `fsgsmNew()` function is NULL, this argument MUST NOT be passed to the Status Monitoring process.	OPTIONAL

- Spaces and one '=' character can be used for the delimiter between each command line option and file descriptor number.
  各引数と file descriptor number の間の区切り文字には、複数の空白文字および1つの'='を許す。空白文字のみも可とする。
  example) --data-write-fd = 5 --data-read-fd = 6 --cmd-write-fd = 7 --cmd-read-fd = 8 --output-fd = 4 input-fd = 4

- The procedures in the `fsgsmNew()` function in the Stub MUST NOT pass the standard error file descriptor (2) for all the above options.
  呼び元は、上記6つ全ての file descriptor に 2 (標準エラー出力) を指定してはいけない。Status Monitoring process は標準エラー出力に対してエラーメッセージを出力することができる。

5.2 Command Format

The command packet format for the Command Writing Pipe and Command Reading Pipe is below.
Command Writing Pipe および Command Reading Pipe 経由で送受信されるコマンドのフォーマットは次の通り。

Byte Offset	Description
0 – 3	Command ID
4 – 7	Data Length (=n bytes)
8 – 8 + (n –1)	Data

- Command ID and Data Length MUST be four bytes big-endian integer.
  Command ID および Data のバイト数は Big Endian の 4 バイト整数で指定する。
- Data MUST be big-endian and aligned on the one byte boundary.
  Data には Big Endian でデータを格納する。alignment は 1byte 境界とする。
- Command ID, Data and the request and acknowledge sequences for each command are defined for each Status Monitoring API function. The Command IDs for each function are defined in the following table.
  Command ID、Data および送受信のシーケンスは、Status Monitoring process の初期化処理および API の各々の関数毎に定義される。各々のコマンド ID と初期化処理、各 API の対応は次の通り。

Command ID	Value	Description
FSGSM_CMD_NEW	0x00000001	Call the `fsgsmNew()` function.
FSGSM_CMD_DESTROY	0x00000002	Call the `fsgsmDestroy()` function.
FSGSM_CMD_GETCAP	0x00000003	Call the `fsgsmGetCap()` function.
FSGSM_CMD_STARTJOB	0x00000011	Call the `fsgsmStartJob()` function.
FSGSM_CMD_ENDJOB	0x00000012	Call the `fsgsmEndJob()` function.
FSGSM_CMD_CANCELJOB	0x00000013	Call the `fsgsmCancelJob()` function.
FSGSM_CMD_STARTREAD	0x00000021	Call the `fsgsmStartRead()` function.
FSGSM_CMD_ENDREAD	0x00000022	Call the `fsgsmRead()` function.
FSGSM_CMD_READ	0x00000023	Call the `fsgsmEndRead()` function.
FSGSM_CMD_STARTWRITE	0x00000031	Call the `fsgsmStartWrite()` function.
FSGSM_CMD_ENDWRITE	0x00000032	Call the `fsgsmWrite()` function.
FSGSM_CMD_WRITE	0x00000033	Call the `fsgsmEndWrite()` function.
FSGSM_CMD_CTRL	0x00000034	Call the `fsgsmCtrl()` function.

- The Command IDs for the acknowledge sequence are defined in the following table.
  また、上記コマンドへの応答に用いる Command ID として次のものを定義する。

Command ID	Value	Description
FSGSM_CMD_OK	0x80000000	Success
FSGSM_CMD_ERROR	0x80000001	Invalid command

- When the Status Monitoring process detect an unknown or illegal command in the “Stub ==> Status Monitoring process” request sequence for each command, the Status Monitoring process MUST send the following command to the Stub in the next “Stub <== Status Monitoring process” acknowledge sequence, and quit all remaining sequences for the command.
  後述の各々のコマンドの各々の Stub ==> Status Monitoring process のシーケンスにおいて、Status Monitoring process 側が未定義のコマンド等の不正コマンドを検出した場合は、それに続く Stub <== Status Monitoring process のシーケンスにおいて、Status Monitoring process 側は次のコマンドを Stub へ送出し、それ以降のシーケンスを中断しなければならない。
  Stub <== Status Monitoring process
  Command ID = FSGSM_CMD_ERROR
  Data Length = 0

5.3 FSGSM_CMD_NEW

Description

- Initialize and set up the properties of the Status Monitoring process just after `fsgsmNew()` function forks and executes it.
  fsgsmNew 関数の実行に伴う Status Monitoring process の起動直後の確認処理を行う。

Sequence

1) The Status Monitoring process receive the following command as soon as possible after being forked and executed.
   Status Monitoring process は起動後、出来るだけ早い時点で次のコマンドを Stub から受信する。
   Stub ==> Status Monitoring process
   Command ID = FSGSM_CMD_NEW
   Data Length = 4
   Data = Status Monitoring version.

   In the current version, the Stub MUST set 0x00010000 to the Status Monitoring version.
   Status Monitoring version には現時点では常に 0x00010000 を指定する。
2) After receiving the command in sequence “1”, the Status Monitoring process compares the version received in sequence “1” and the version supported by the Status Monitoring process. If the version in sequence “1” is less or equal to the version supported by the Status Monitoring process, the Status Monitoring process proceed to sequence “3”. If the version in sequence “1” is greater than the version supported by the Status Monitoring process, the Status Monitoring process MUST send the following command to the Stub, and idle until being killed by the Stub by the SIGTERM signal.
   Status Monitoring process は、1)のコマンドを受信後、その Data 中の version と Status Monitoring process がサポートする version を比較する。Data 中の version≦Status Monitoring process である場合は、3)以降の処理を継続する。
   Data 中の version＞Status Monitoring process である場合は、次のコマンドを Stub に送信し、Stub から SIGTERM によって中断されるまで永久スリープしなければならない。
   Stub <== Status Monitoring process
   Command ID = FSGSM_CMD_OK
   Data Length = 4
   Data = FSGSM_ERROR (Type:int, Length:4bytes)

3) The Status Monitoring process initializes itself. If some error occurs, the Status Monitoring process MUST send the following command to the Stub, and idle until being killed by the Stub by the SIGTERM signal.
   Status Monitoring process は、自身の初期化処理を継続し、その途中何らかのエラーが発生した場合は、次のコマンドを Stub に送信後、Stub から SIGTERM によって中断されるまで永久スリープしなければならない。
   Stub <== Status Monitoring process
   Command ID = FSGSM_CMD_OK
   Data Length = 4
   Data = FSGSM_ERROR (Type:int, Length:4bytes)

4) After the initialization of the Status Monitoring process, the Status Monitoring process sends the following command to the Stub.
   Status Monitoring process は、その初期化処理が完了した後、次のコマンドを Stub に送信する。
   Stub <== Status Monitoring process
   Command ID = FSGSM_CMD_OK
   Data Length = 4
   Data = FSGSM_OK (Type:int, Length:4bytes)

5) After receiving the command in sequence “4”, the Stub creates a new Status Monitoring Object, and returns the pointer to the Monitor.
   Stub は 4)のコマンドを受信後、適切な Status Monitoring Object を生成して呼び元にそのポインタを返す。

- The Stub MUST wait at least 30 seconds for the command from the Status Monitoring process. If the Stub cannot receive any commands from the Status Monitoring process within 30 seconds, the Stub MUST kill the Status Monitoring process by sending the SIGTERM to it, and clean up the Stub itself and return an error code to the Monitor.
  Stub は、Status Monitoring process を起動後少なくとも 30 秒間、Status Monitoring process からのコマンドの受信を待たなければならない。30 秒間待ってもコマンドが受信できない場合は、Status Monitoring process を SIGTERM で kill し、後処理を行わなければならない。
- When the Stub receives the FSGSM_ERROR in “Data”, the Stub MUST send the SIGTERM signal to the Status Monitoring program to kill it.
  Stub は、Status Monitoring process から Data=FSGSM_ERROR を受信した場合は、Status Monitoring process に対して SIGTERM を送信し、Status Monitoring process の処理を中断しなければならない。

5.4 FSGSM_CMD_DESTROY

Description

- Execute the procedures for the `fsgsmDestroy()` function.
  fsgsmDestoy 関数の実行を Status Monitoring process に通知する。

Sequence

1) When the Monitor calls the `fsgsmDestroy()` function, the Stub sends the following command to the Status Monitoring process. fsgsmDestroy 関数の実行に伴い、Stub は次のコマンドを Status Monitoring process へ送信する。
   Stub ==> Status Monitoring process
   Command ID = FSGSM_CMD_DESTROY
   Data Length = 0

2) After receiving the command of sequence “1”, the Status Monitoring process executes the termination procedures for itself, and after that, sends the following command to the Stub.
   Status Monitoring process は 1)のコマンドを受信後、終了処理を行ってから次のコマンドを Stub へ送信する。
   Stub <== Status Monitoring process
   Command ID = FSGSM_CMD_OK
   Data Length = 0

   And the Status Monitoring process quits itself.
   その後、Status Monitoring process は process を終了する。

3) After receiving the command in sequence “2”, the Stub clean up its properties, and return to the Monitor.
   Stub は 2)のコマンドを受信後、Status Monitoring process 終了の後処理を行う。

5.5 FSGSM_CMD_GETCAP

Description

- Execute the procedures for the `fsgsmGetCap()` function.
  fsgsmGetCap 関数の実行を Status Monitoring process に通知する。

Sequence

1) When the Monitor calls the `fsgsmGetCap()` function, the Stub sends the following command to the Status Monitoring process.
   fsgsmGetCap 関数の実行に伴い、Stub は次のコマンドを Status Monitoring process へ送信する。
   Stub ==> Status Monitoring process
   Command ID = FSGSM_CMD_GETCAP
   Data Length = 4
   Data = *cap* (Type:FSGSMCtxap, Length:4bytes) : Argument *cap* of the `fsgsmGetCap()` function.

2) After receiving the command of sequence “1”, the Status Monitoring process sends the following command to the Stub.
   Status Monitoring process は 1)のコマンドを受信後、次のコマンドを Stub へ送信する。
   Stub <== Status Monitoring process
   Command ID = FSGSM_CMD_OK
   Data Length = 4
   Data = Return Value (Type:int, Length:4bytes) : Return value of the `fsgsmGetCap()` function.

   If the Status Monitoring process receives an illegal command in sequence “1”, the Status Monitoring process sends the following command to the Stub.
   1)で不正な cap 値を Status Monitoring process が受信した場合は、次のコマンドを Stub へ送信する。
   Stub <== Status Monitoring process
   Command ID = FSGSM_CMD_OK
   Data Length = 4
   Data = FSGSM_ERROR (Type:int, Length:4bytes) : Return value of the `fsgsmGetCap()` function.

5.6 FSGSM_CMD_STARTJOB

Description

- Execute the procedures for the `fsgsmStartJob()` function.
  fsgsmStartJob 関数の実行を Status Monitoring process に通知する。

Sequence

1) When the Monitor calls the `fsgsmStartJob()` function, the Stub sends the following command to the Status Monitoring process.
   fsgsmStartJob 関数の実行に伴い、Stub は次のコマンドを Status Monitoring process へ送信する。
   Stub ==> Status Monitoring process
   Command ID = FSGSM_CMD_STARTJOB
   Data Length = 4
   Data = *idJob* (Type:int, Length:4bytes) : Argument *idJob* of the `fsgsmStartJob()` function.

2) After receiving the command of sequence "1", the Status Monitoring process sends the following command to the Stub.
   Status Monitoring process は 1)のコマンドを受信後、次のコマンドを Stub へ送信する。
   Stub <== Status Monitoring process
   Command ID = FSGSM_CMD_OK
   Data Length = 4
   Data = Return Value (Type:int, Length:4bytes) : Return value of the `fsgsmStartJob()` function.

5.7 FSGSM_CMD_ENDJOB

Description

- Execute the procedures for the `fsgsmEndJob()` function.
  fsgsmEndJob 関数の実行を Status Monitoring process に通知する。

Sequence

1) When the Monitor calls the `fsgsmEndJob()` function, the Stub sends the following command to the Status Monitoring process.
   fsgsmEndJob 関数の実行に伴い、Stub は次のコマンドを Status Monitoring process へ送信する。
   Stub ==> Status Monitoring process
   Command ID = FSGSM_CMD_ENDJOB
   Data Length = 0

2) After receiving the command of sequence "1", the Status Monitoring process sends the following command to the Stub.
   Status Monitoring process は 1)のコマンドを受信後、次のコマンドを Stub へ送信する。
   Stub <== Status Monitoring process
   Command ID = FSGSM_CMD_OK
   Data Length = 4
   Data = Return Value (Type:int, Length:4bytes) : Return value of the `fsgsmEndJob()` function.

5.8 FSGSM_CMD_CANCELJOB

Description

- Execute the procedures for the `fsgsmCancelJob()` function.
  fsgsmCancelJob 関数の実行を Status Monitoring process に通知する。

Sequence

1) When the Monitor calls the `fsgsmCancelJob()` function, the Stub sends the following command to the Status Monitoring process.
   fsgsmCancelJob 関数の実行に伴い、Stub は次のコマンドを Status Monitoring process へ送信する。
   Stub ==> Status Monitoring process
   Command ID = FSGSM_CMD_CANCELJOB
   Data Length = 4
   Data = *idJob* (Type:int, Length:4bytes) : Argument *idJob* of the `fsgsmCancelJob()` function.

2) After receiving the command of sequence "1", the Status Monitoring process sends the following command to the Stub.
   Status Monitoring process は 1)のコマンドを受信後、ジョブの終了処理を行ってから次のコマンドを Stub へ送信する。
   Stub <== Status Monitoring process

Command ID = FSGSM_CMD_OK
Data Length = 4
Data = Return Value (Type:int, Length:4bytes) : Return value of the `fsgsmCancelJob()` function.

5.9 FSGSM_CMD_STARTREAD

Description

- Execute the procedures for the `fsgsmStartRead()` function.
  fsgsmStartRead 関数の実行を Status Monitoring process に通知する。

Sequence

1) When the Monitor calls the `fsgsmStartRead()` function, the Stub sends the following command to the Status Monitoring process.
   fsgsmStartRead 関数の実行に伴い、Stub は次のコマンドを Status Monitoring process へ送信する。
   Stub ==> Status Monitoring process
   Command ID = FSGSM_CMD_STARTREAD
   Data Length = 4 + 4 + n ()
   Data = *idReadMode* (Type:int, Length:4bytes) : Argument *idReadMode* of the `fsgsmStartRead()` function.
   n (Type:int, Length:4bytes) : Number of bytes of the argument *pLang* string.
   String (Type:char, Length:n bytes) : *pLang* string without a zero termination code.

2) After receiving the command of sequence “1”, the Status Monitoring process prepares for sending the printer status data to the Stub, and sends the following command to the Stub.
   Status Monitoring process は 1)のコマンドを受信後、プリンタステータスデータの Stub への送出準備処理を完了してから、次のコマンドを Stub へ送信する。
   Stub <== Status Monitoring process
   Command ID = FSGSM_CMD_OK
   Data Length = 4
   Data = Return Value (Type:int, Length:4bytes) : Return value of the `fsgsmStartRead()` function.

The Status Monitoring process establishes and retains the printer status data in the buffer and initializes the pointer for reading the data from the sequence “2” buffer. After that, the Status Monitoring process MUST NOT update the printer status data until receiving the FSGSM_CMD_ENDREAD command.
送出するプリンタステータスデータは 2)の時点で決定し、その read 開始位置を初期化する。以降 Status Monitoring process は、FSGSM_CMD_ENDREAD コマンドを受信するまで、プリンタステータスデータを更新してはならない。

5.10 FSGSM_CMD_ENDREAD

Description

- Execute the procedures for the `fsgsmEndRead()` function.
  fsgsmEndRead 関数の実行を Status Monitoring process に通知する。

Sequence

1) When the Monitor calls the `fsgsmEndRead()` function, the Stub sends the following command to the Status Monitoring process.
   fsgsmEndRead 関数の実行に伴い、Stub は次のコマンドを Status Monitoring process へ送信する。
   Stub ==> Status Monitoring process
   Command ID = FSGSM_CMD_ENDREAD
   Data Length = 0

2) After receiving the command of sequence “1”, the Status Monitoring process sends the following command to the Stub.
   Status Monitoring process は 1)のコマンドを受信後、自身のプリンタステータスデータ更新の再開処理を行ってから、次のコマンドを Stub へ送信する。
   Stub <== Status Monitoring process
   Command ID = FSGSM_CMD_OK
   Data Length = 4
   Data = Return Value (Type:int, Length:4bytes) : Return value of the `fsgsmEndRead()` function.

After sequence “2”, the Status Monitoring process is permitted to update the printer status data in the buffer.

5.11 FSGSM_CMD_READ

Description

- Execute the procedures for the `fsgsmRead()` function.
  fsgsmRead 関数の実行を Status Monitoring process に通知し、プリンタステータスデータを Stub に送出する。

Sequence

1) When the Monitor calls the `fsgsmRead()` function, the Stub sends the following command to the Status Monitoring process.
   fsgsmRead 関数の実行に伴い、Stub は次のコマンドを Status Monitoring process へ送信する。
   Stub ==> Status Monitoring process
   Command ID = FSGSM_CMD_READ
   Data Length = 4
   Data = *nBufBytes* (Type:int, Length:4bytes) : Argument *nBufBytes* of the `fsgsmRead()` function.

2) After receiving the command of sequence “1”, if the printer status data has already been established and retained in the buffer, the Status Monitoring process sends the following command to the Stub.
   Status Monitoring process は 1)のコマンドを受信後、プリンタステータスデータの送出準備が完了していた場合は、次のコマンドを Stub へ送信する。
   Stub <== Status Monitoring process
   Command ID = FSGSM_CMD_OK
   Data Length = 4
   Data = Number of bytes for sending the printer status data. (Type:int, Length:4bytes)

   The Status Monitoring process compares the number of bytes of the printer status data that the Status Monitoring process retains in the buffer and the number of bytes in “Data” received at sequence “1”, and sets the smaller one into the “Data” for sequence “2”.
   送出する Byte 数は、1)で受信した nBufBytes と Status Monitoring process が保持しているプリンタステータスデータのバイト数のうち、小さい方の値とする。

3) The Status Monitoring process reads the printer status data from the reading point of the buffer to the number of bytes in “Data” calculated at sequence “2”, and sends it to the Stub via the Data Reading Pipe.
   Status Monitoring process は、Data Reading Pipe 経由でプリンタステータスデータのうち read 開始位置から 2)で受信したバイト数分だけ Stub へ送信する。

4) After receiving the command of sequence “2”, the Stub receives the “Data” bytes printer status data from the Status Monitoring process via the Data Reading Pipe.
   Stub は 2)のコマンドを受信後 Data Reading Pipe 経由で 2)で受信したバイト数分のプリンタステータスデータを受信する。

5.12 FSGSM_CMD_STARTWRITE

Description

- Execute the procedures for the `fsgsmStartWrite()` function.
  fsgsmStartWrite 関数の実行を Status Monitoring process に通知する。

Sequence

1) When the Monitor calls the `fsgsmStartWrite()` function, the Stub sends the following command to the Status Monitoring process.
   fsgsmStartWrite 関数の実行に伴い、Stub は次のコマンドを Status Monitoring process へ送信する。
   Stub ==> Status Monitoring process
   Command ID = FSGSM_CMD_STARTWRITE
   Data Length = 0

2) After receiving the command of sequence “1”, the Status Monitoring process prepares for receiving the printer command data from the Stub, and sends the following command to the Stub.
   Status Monitoring process は 1)のコマンドを受信後、Stub からのデータ受信準備処理を完了してから、次のコマンドを Stub へ送信する。
   Stub <== Status Monitoring process
   Command ID = FSGSM_CMD_OK
   Data Length = 4

Data = Return Value (Type:int, Length:4bytes) : Return value of the `fsgsmStartWrite()` function.

5.13 FSGSM_CMD_ENDWRITE

Description

- Execute the procedures for the `fsgsmEndWrite()` function.
  fsgsmEndWrite 関数の実行を Status Monitoring process に通知する。

Sequence

1) When the Monitor calls the `fsgsmEndWrite()` function, the Stub sends the following command to the Status Monitoring process.
   fsgsmEndWrite 関数の実行に伴い、Stub は次のコマンドを Status Monitoring process へ送信する。
   Stub ==> Status Monitoring process
   Command ID = FSGSM_CMD_ENDREAD
   Data Length = 0

2) After receiving the command of sequence “1”, the Status Monitoring process sends the following command to the Stub.
   Status Monitoring process は 1)のコマンドを受信後、次のコマンドを Stub へ送信する。
   Stub <== Status Monitoring process
   Command ID = FSGSM_CMD_OK
   Data Length = 4
   Data = Return Value (Type:int, Length:4bytes) : Return value of the `fsgsmEndWrite()` function.

5.14 FSGSM_CMD_WRITE

Description

- Execute the procedures for the `fsgsmWrite()` function.
  fsgsmWrite 関数の実行を Status Monitoring process に通知し、Stub からデータを受信する。

Sequence

1) When the Monitor calls the `fsgsmWrite()` function, the Stub sends the following command to the Status Monitoring process.
   fsgsmWrite 関数の実行に伴い、Stub は次のコマンドを Status Monitoring process へ送信する。
   Stub ==> Status Monitoring process
   Command ID = FSGSM_CMD_WRITE
   Data Length = 0

2) After receiving the command of sequence “1”, if the Status Monitoring process is ready to receive the printer command data, the Status Monitoring process sends the following command to the Stub.
   Status Monitoring process は 1)のコマンドを受信後、データの受信準備が完了していた場合は、次のコマンドを Stub へ送信する。
   Stub <== Status Monitoring process
   Command ID = FSGSM_CMD_OK
   Data Length = 4
   Data = Maximum number of bytes of the data that the Status Monitoring process can receive. (Type:int, Length:4bytes)

3) After receiving the command of sequence “2”, if the number of bytes in “Data” is greater than zero, the Stub compares the number of bytes of the printer command data that the Stub retains in the buffer and the number of bytes in “Data” received at sequence “2”, and sets the smaller one into the “Data” and sends the following command to the Status Monitoring process.
   Stub ==> Status Monitoring process
   Command ID = FSGSM_CMD_OK
   Data Length = 4
   Data = Number of bytes of the data that the Stub sends. (Type:int, Length:4bytes)

4) After sending the command of sequence “3”, if the number of bytes in “Data” of sequence “3” is greater than zero, the Stub sends the printer command data to the Status Monitoring process to the number of bytes in “Data” via the Data Writing Pipe.

5) If the number of bytes in “Data” of sequence “3” is greater than zero, the Status Monitoring process receives the printer command data from the Stub to the number of bytes in “Data” via the Data Writing Pipe.

5.15 FSGSM_CMD_CTRL

Description

- Execute the procedures for the `fsgsmCtrl()` function.
  fsgsmCtrl 関数の実行を Status Monitoring process に通知し、Stub からデータを受信する。

Sequence

1) When the Monitor calls the `fsgsmCtrl()` function, the Stub sends the following command to the Status Monitoring process.
   fsgsmCtrl 関数の実行に伴い、Stub は次のコマンドを Status Monitoring process へ送信する。
   Stub ==> Status Monitoring process
   Command ID = FSGSM_CMD_CTRL
   Data Length = 8
   Data = *idRequest* (Type:int, Length:4bytes) : Argument *idRequest* of the `fsgsmCtrl()` function.
   nDataBytes (Type:int, Length:4bytes) : Argument *nDataBytes* of the `fsgsmCtrl()` function.

2) After receiving the command of sequence “1”, if the Status Monitoring process is ready to receive the data, the Status Monitoring process sends the following command to the Stub.
   Status Monitoring process は 1)のコマンドを受信後、データの受信準備が完了していた場合は、次のコマンドを Stub へ送信する。
   Stub <== Status Monitoring process
   Command ID = FSGSM_CMD_OK
   Data Length = 0

3) The Stub sends the *nDataBytes* bytes data in the buffer specified by the argument *pData* of the `fsgsmCtrl()` function to the Status Monitoring process via the Command Writing Pipe.
   Stub は、Command Writing Pipe 経由でバッファ pData の nDataBytes のデータを Status Monitoring process へ送出する。

4) After sending the command of sequence “2”, the Status Monitoring process receives the *nDataBytes* bytes data from the Stub via the Command Writing Pipe.
   Status Monitoring process は 2)のコマンドを送信後 Command Writing Pipe 経由で nDataBytes のデータを受信する。

5) After receiving the data of sequence “4”) and executing the procedure specified by *idRequest*, the Status Monitoring process sends the following command to the Stub.
   Status Monitoring process は 4)のデータを受信して idRequest に対応した処理をした後、次のコマンドを Stub へ送信する。
   Stub <== Status Monitoring process
   Command ID = FSGSM_CMD_OK
   Data Length = 8
   Data = Return Value (Type:int, Length:4bytes) : Return value of the `fsgsmCtrl()` function.
   nDataBytes (Type:int, Length:4bytes) : Number of bytes of the return data stored in the *pData* buffer.

   The value of *nDataBytes* in “Data” of the command of sequence “5”) might change from the value of *nDataBytes* in “Data” of the command of sequence “1”.
   ここで得られる nDataBytes の値は 1)で与えられる nDataBytes の値と異なっている可能性がある。

6) The Status Monitoring process sends the *nDataBytes* bytes data established at sequence “5” via the Command Reading Pipe.
   Status Monitoring process は、Command Reading Pipe 経由で nDataBytes のデータを Stub へ送信する。尚、nDataBytes は 5)の値を参照する。

7) After receiving the command of sequence “5”, the Stub receives the *nDataBytes* bytes data established at sequence “5” via the Command Reading Pipe, and store it into the buffer specified by *pData*.
   Stub は 5)のコマンドを受信後 Command Reading Pipe 経由で nDataBytes のデータを受信してバッファ pData に格納する。尚、nDataBytes は 5)の値を参照する。

5.16 Signal Handling

The following table defines how the Status Monitoring process SHOULD or MUST deal with each signal.
Process Type の Status Monitoring process は、次のシグナルについて各々適切な処理を行う必要がある。

Signal	Processing
SIGTERM	After receiving this signal, the Status Monitoring process SHOULD terminate itself after terminating the printer status data reading procedure and printer command writing procedure. If the Status Monitoring process receives this signal after receiving the FSGSM_CMD_STARTJOB command and before dealing with the FSGSM_CMD_ENJOB command, the termination procedure is equivalent to the procedure of dealing with the FSGSM_CMD_CANCELJOB, FSGSM_CMD_ENDJOB and FSGSM_CMD_DESTROY commands in turn. If the Status Monitoring process receives this signal before receiving the FSGSM_CMD_STARTJOB command or after dealing with the FSGSM_CMD_ENDJOB command, the termination procedure is equivalent to the procedure of dealing with the FSGSM_CMD_DESTROY command. The Status Monitoring process MUST NOT send any commands to the Stub after receiving this signal. プリンタステータスデータの read 処理、データの write 処理等の全ての処理を終了し、自分自身を終了する。その場合の終了処理は、FSGSM_CMD_STARTJOB コマンドの受信後でかつ FSGSM_CMD_ENDJOB コマンドの受信前であれば、FSGSM_CMD_CANCELJOB、FSGSM_CMD_ENDJOB、FSGSM_CMD_DESTROY の3つのコマンドを1つずつ順番に受信した場合に実行される終了処理と等価の処理でなければならない。また、FSGSM_CMD_STARTJOB コマンドの受信前であれば、FSGSM_CMD_DESTROY コマンドを受信した場合に実行される終了処理と等価の処理でなければならない。尚、シグナル受信後は、Stub 側に一切のコマンドを送信してはならない。
SIGPIPE	Same as above.
SIGHUP	When receiving this signal during any request or acknowledge sequences of any commands, the Status Monitoring process MUST cancel the command sequence and wait for a new command, such as FSGSM_CMD_WRITE or FSGSM_CMD_STARTJOB, etc, sent from the Stub. 各々のコマンドの送受信シーケンスにおいて、シーケンスの途中の段階で本シグナルを受信した場合は、そのコマンドのシーケンスをリセットして、Stub 側からの FSGSM_CMD_STARTJOB や FSGSM_CMD_WRITE 等の新たなコマンドを待つ状態に復帰しなければならない。

6. Printer Status Data Format

6.1 Printer Status Data XML Schema

The data format of the printer status data obtained from the `fsgsmRead()` function is defined in a set of XML schema files, including the Printer MIB v2 schema. These XML schema files are being considered in the PWG WBMM group, and currently located at the following FTP address.

ftp://ftp.pwg.org/pub/pwg/wbmm/schemas/

6.2 Printer Status Data Filtering

Obtaining the entire XML based printer status data periodically from the Status Monitoring module may require a high data processing capability as well as a high data transfer rate. To reduce the amount of the printer status data, the Status Monitoring API provides a filtering mechanism to obtain only the changed or variable printer status data.

When the Monitor calls the `fsgsmStartRead()` function, the printer status data is established and retained, and after that, the Monitor can obtain the printer status data from the `fsgsmRead()` function. The printer status data obtained from the `fsgsmRead()` function MUST be along the data specified by *idReadMode* given by the `fsgsmStartRead()` function prior to calling the `fsgsmRead()` function.

idReadMode	Description
FSGSM_READ_PRT_MIB_ALL	Read all the printer status data. The printer status data SHOULD include all the printer status data that the Status Monitoring module supports. Initially, the Monitor obtains the entire printer status data from the Status Monitoring module.
FSGSM_READ_PRT_MIB_SUMMARY	Read the summarized printer status data. The printer status data SHOULD include only the changed or variable data. The Monitor or upper system which receives the printer status data from the Monitor can detect the changed printer status data by comparing the printer status data obtained in the FSGSM_READ_PRT_MIB_ALL mode with the data obtained in this mode.

7. Constants

7.1 Return Values

The following table defines the return value used in the Status Monitoring API function definitions.

Define	Value
FSGSM_OK	0
FSGSM_TRUE	1
FSGSM_FALSE	0
FSGSM_ERROR	-1
FSGSM_EINTR	-2
FSGSM_EPROGRESS	-3
FSGSM_ENOJOB	-4

8. Contributors and Author

8.1 Contributors

TORATANI Yasumasa	Canon Inc.
Osamu MIHARA	FUJI XEROX Printing Systems
KANJO Hidenori	BBR INC.
YOSHIDA Mikio	BBR INC.
Shinpei KITAYAMA	EPSON KOWA
YAMAGISHI Toshihiro	Turbolinux
Hisao NAKAMURA	E&D
Koji OTANI	AXE

8.2 Author

TORATANI Yasumasa

Canon Inc.
30-2, Shimomaruko 3-Chome, Ohta-ku, Tokyo 146-8501, Japan
Email: toratani.yasumasa@canon.co.jp