



Linux版  
開発キットソフトウェア仕様書

Version 0.9.3

2020/5/11

株式会社WDS

## 改訂履歴

作成年月日	変更箇所、理由など	作成者	ソフトウェアのバージョン
2019/12/21	初版発行。	平林	Version 0.001
2020/01/10	fclVersion を追加	平林	
2020/01/28	fclInit,fclRegister,fclThreshold に追記	平林	
2020/02/12	fclRegisterJpegHandler fclStoreFeature fclLoadFeature fclFeatureCount を追加	平林	
2020/02/13	fclRegisterFaceDataHandlerを追加	平林	
2020/02/16	fclInit でコールバックの登録を止め、 fclRegisterMatchHandler で登録するように変更	平林	
2020/02/18	全関数fclLocal *に変更	平林	
2020/02/18	fclLocalCompair,fclLocal,fclLocalGetFaceDataを追加	平林	
2020/04/27	主要処理のシーケンス図を追加	WDS	
2020/05/11	表示ロゴ修正	WDS	

## 目次

1. 概要 .....	4
2. システム構成図 .....	4
3. 提供ソフトウェア .....	4
4. API .....	5
4.1. fclLocalInit .....	5
4.2. fclLocalRegisterMatchHandler .....	6
4.3. fclLocalDeinit .....	7
4.4. fclLocalSetThreshold .....	8
4.5. fclLocalRegister .....	9
4.6. fclLocalRegisterFromFile .....	10
4.7. fclLocalUnregister .....	11
4.8. fclLocalVersion .....	12
4.9. fclLocalRegisterJpegHandler .....	13
4.10. fclLocalStoreFeature .....	14
4.11. fclLocalLoadFeature .....	15
4.12. fclLocalFeatureCount .....	16
4.13. fclLocalRegisterFaceDataHandler .....	17
4.14. fclLocalCompair .....	18
4.15. fclLocalGetFaceData .....	19
5. ユースケース .....	20
6. FaceData の定義 .....	21
7. お問い合わせ先 .....	22

## 1. 概要

本書は、Eeyeカメラ開発キットを用いてシステム開発する際のソフトウェア仕様書(Linux版)となります。開発キット以外の環境にて開発を実施する場合は、適宜読み替えて開発を進めてください。

## 2. システム構成図

Eeyeカメラ開発キットのシステム構成図は以下となります。

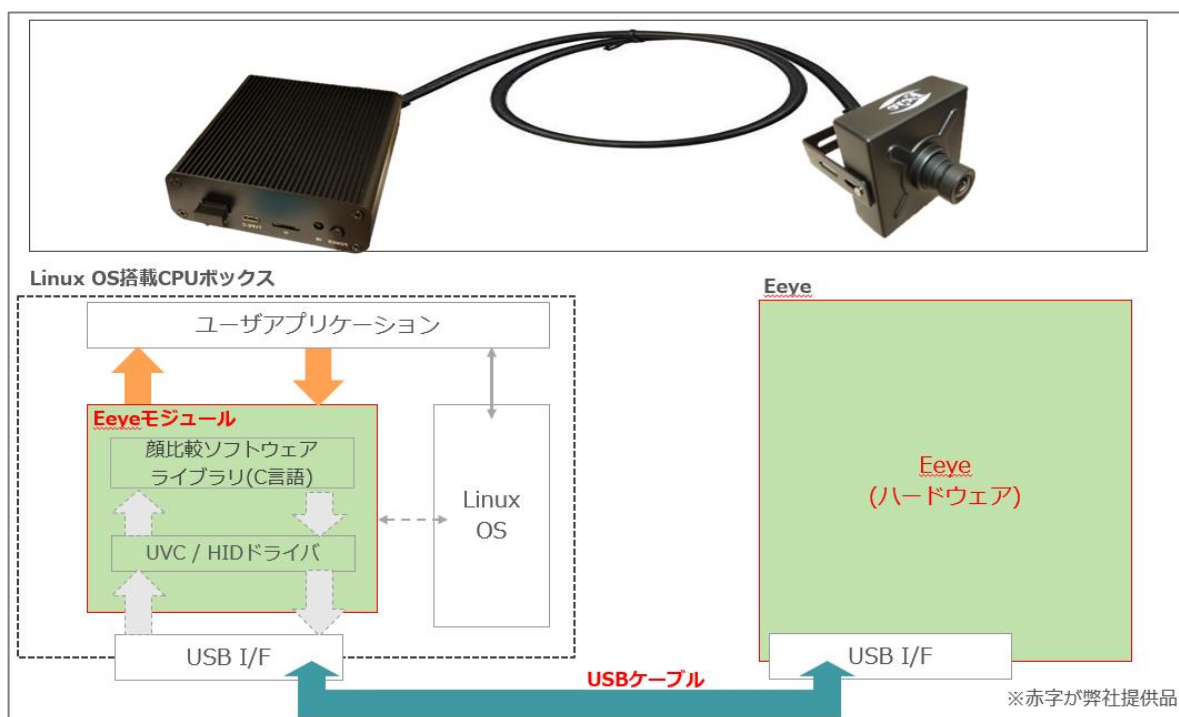


図1: Eeye開発キットシステム構成図

## 3. 提供ソフトウェア

本開発キットにて提供するソフトウェアは、Ubuntu上にて開発したC/C++言語のソースコードを提供いたします。以降の章にて、提供ソースコードのAPIと処理シーケンスについて記載します。

## 4. API

### 4.1. fclLocalInit

---

#### 4.1.1. インターフェース

```
bool fclLocalInit(void);
```

#### 4.1.2. 機能

ライブラリの初期化を行います。

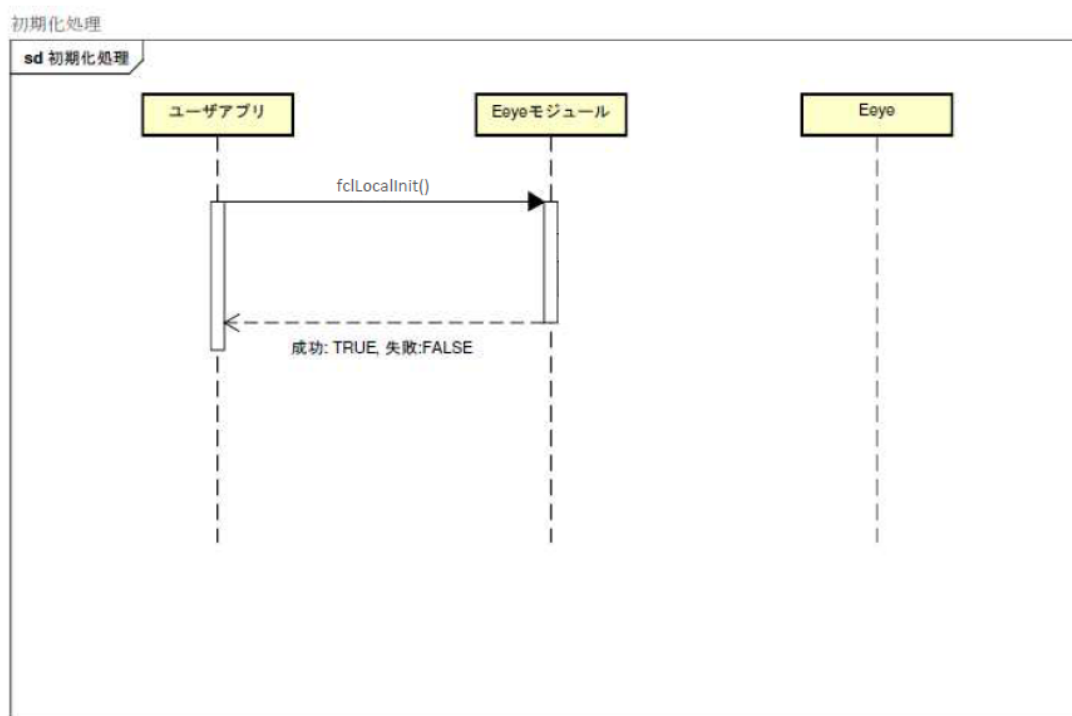
#### 4.1.3. パラメータ

なし

#### 4.1.4. 返り値

呼び出しが成功すると TRUE、失敗すると FALSE が返されます。

#### 4.1.5. 処理シーケンス



## 4.2. fclLocalRegisterMatchHandler

### 4.2.1. インターフェース

```
void fclLocalRegisterMatchHandler(void (*fclMatchHandler)(uint32_t id,float probability,bool real);
```

### 4.2.2. 機能

コールバック関数の登録を行います。

登録したJPEGファイルの顔と一致する顔を認識したときコールバック関数がコールされます。

コールバック関数が呼ばれるときid とprobability、real がパラメータとして渡されます。idは

JPEGデータの登録時 fclRegister またはfclRegisterFromFile から返された値となります。

probabilityは0から1までの値で1が最も一致していることを示します。

経験上0.4以上となる場合、顔が一致したと認められます。

またreal とはカメラがtrueと認識した顔が生顔(写真でない)を表します。

すでにコールバック関数が登録されている場合は、新しいものと置き換わります。

コールバックを取り消す場合は、パラメータにNULLを指定します。

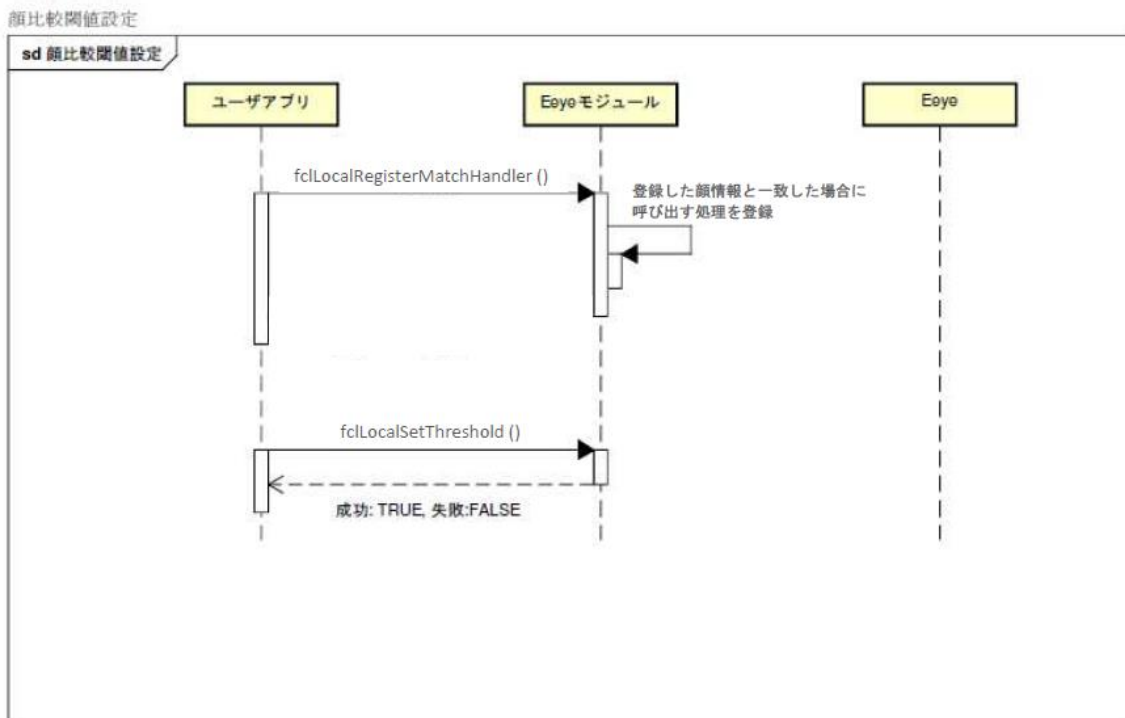
### 4.2.3. パラメータ

```
void (*fclMatchHandler)(uint32_t id,float probability,bool real) 登録するコールバック関数
```

### 4.2.4. 戻り値

なし

### 4.2.5. 処理シーケンス



## 4.3. fclLocalDeinit

### 4.3.1. インターフェース

```
void fclLocalDeinit(void);
```

### 4.3.2. 機能

ライブラリの機能を停止しリソースを解放します。

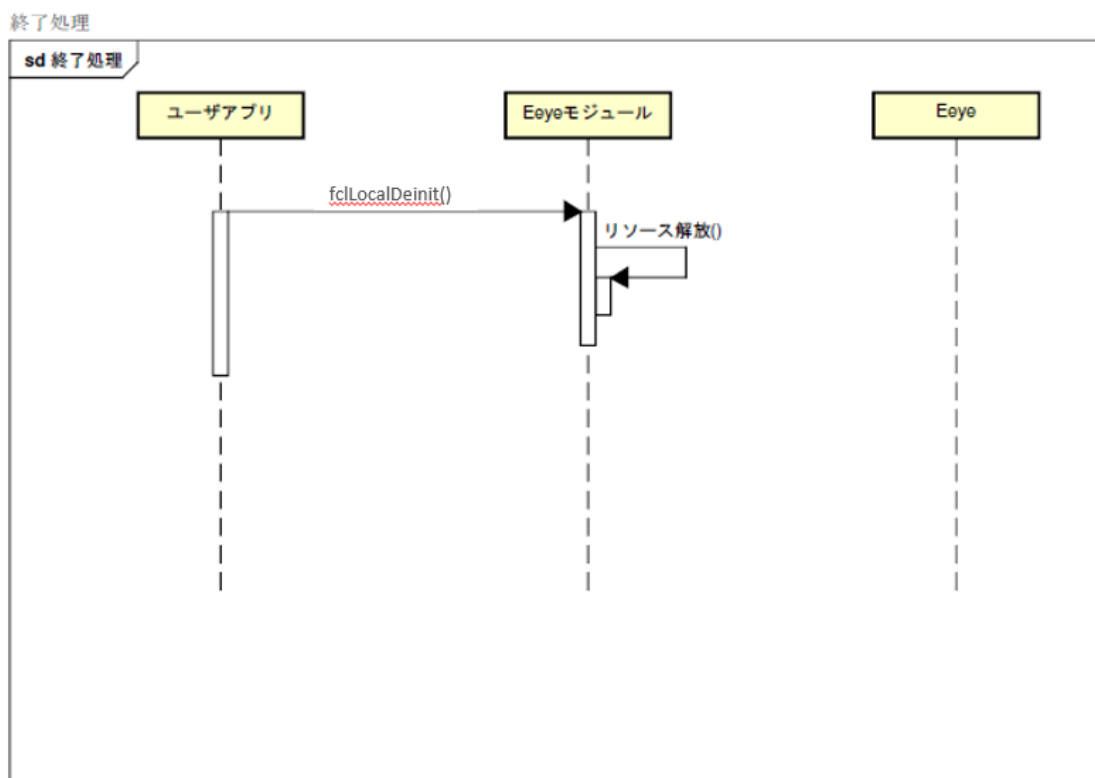
### 4.3.3. パラメータ

なし

### 4.3.4. 戻り値

なし

### 4.3.5. 処理シーケンス



## 4.4. fclLocalSetThreshold

---

### 4.4.1. インターフェース

```
bool fclLocalSetThreshold(float threshold);
```

### 4.4.2. 機能

登録した JPEG とカメラの検出した顔がどの程度似ていたらコールバック関数をコールするかの閾値を設定します。経験上 0.4 以上となる場合、顔が一致したと認められます。本ライブラリのデフォルトは 0.4となっています。

### 4.4.3. パラメータ

Float threshold      比較の閾値。比較結果がこの値以上であればコールバックされる。

### 4.4.4. 返り値

呼び出しが成功したときはTURE、失敗したときは FALSE が返されます。

### 4.4.5. 処理シーケンス

4.2.5. の処理シーケンス参照。

## 4.5. fclLocalRegister

### 4.5.1. インターフェース

```
uint32_t fclLocalRegister(void* jpeg,int jpeg_length);
```

### 4.5.2. 機能

JPEG データの登録を行います。登録が成功したら id を返します。登録が失敗した場合は0を返します。

### 4.5.3. パラメータ

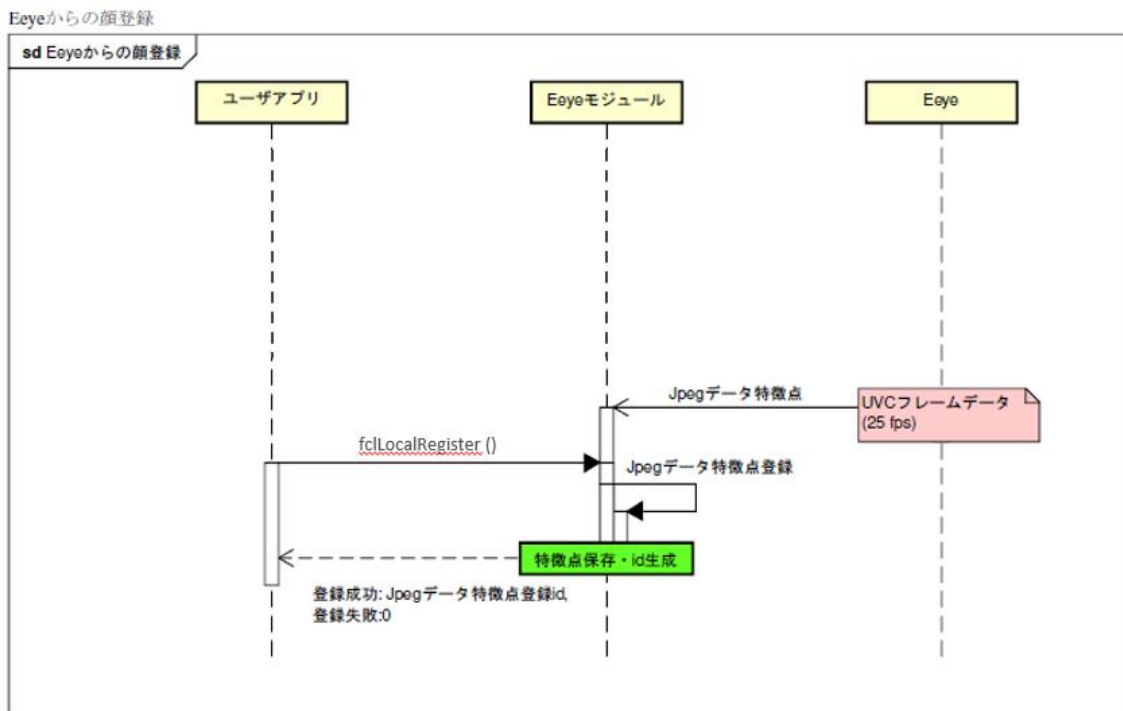
void* jpeg	JPEGデータを格納したバッファアドレス
Float threshold	JPEGデータ長

### 4.5.4. 返り値

成功したときは、jpeg データに対応した id が返されます。エラーの場合は0が返されます。呼び出しが失敗するケースは以下の場合があります。

- ・ カメラと接続されていない
- ・ 既に登録されている
- ・ ファイルサイズが大きい 概ね 200KB以上
- ・ 画像サイズが大きい 概ね 1000x1000 pix
- ・ JPEG ファイルに顔が含まれていない

### 4.5.5. 処理シーケンス



## 4.6. fclLocalRegisterFromFile

### 4.6.1. インターフェース

```
uint32_t fclLocalRegisterFromFile(const char* jpegpath);
```

### 4.6.2. 機能

JPEG ファイルから JPEG データの登録を行います。登録が成功したらid を返します。登録が失敗した場合は0を返します。

現バージョンでは登録できるデータ件数は1件のみです。

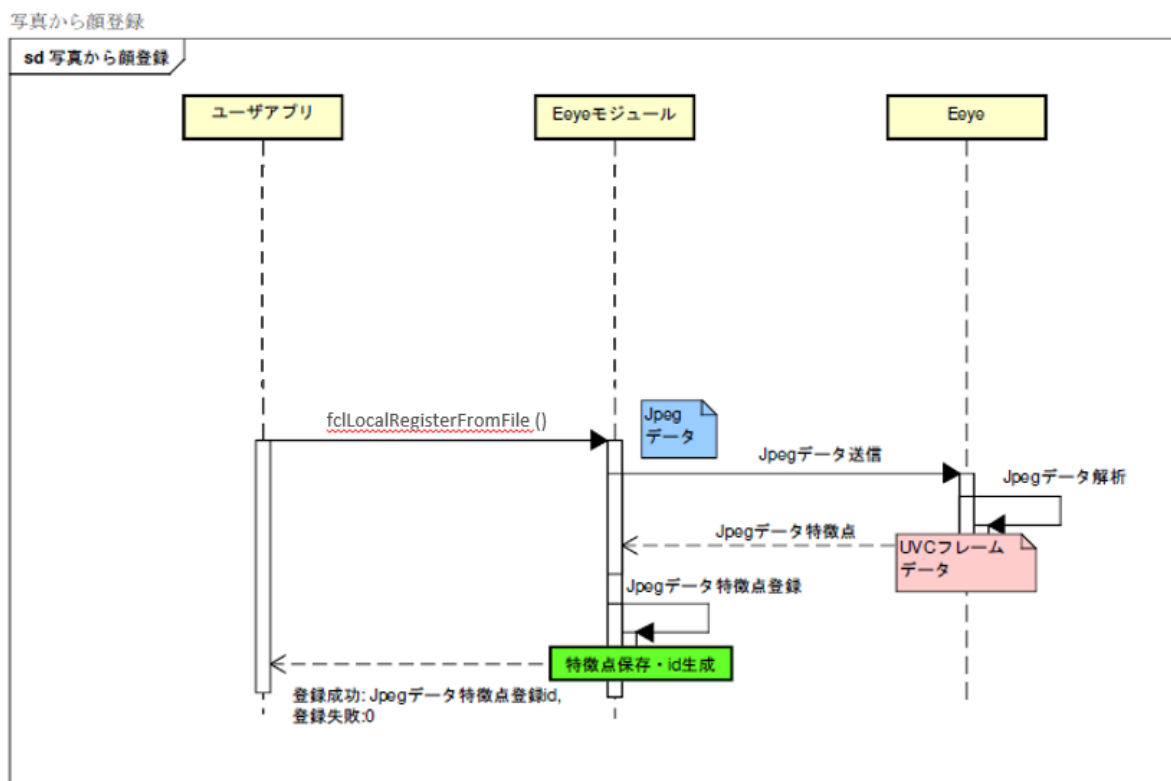
### 4.6.3. パラメータ

const char\* jpegpath     JPEG ファイルの path

### 4.6.4. 戻り値

成功したときは、jpeg データに対応した id が返されます。エラーの場合は0が返されます。

### 4.6.5. 処理シーケンス



## 4.7. fclLocalUnregister

### 4.7.1. インターフェース

```
bool fclLocalUnregister(uint32_t id);
```

### 4.7.2. 機能

登録されているJPEG データの登録を削除します。

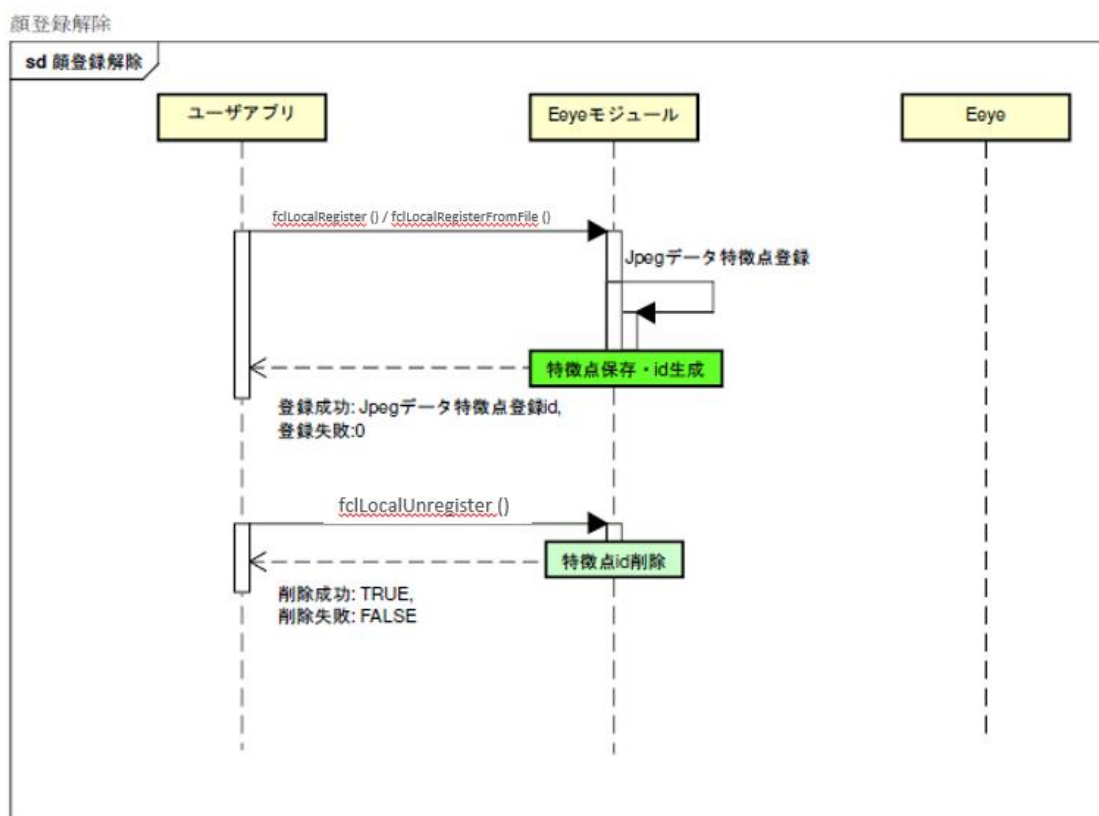
### 4.7.3. パラメータ

uint32\_t id          fclRegister/fclRegisterFromFile の呼び出しで返された id

### 4.7.4. 戻り値

成功したときは TRUE、失敗したときは FALSE を返します。

### 4.7.5. 処理シーケンス



## 4.8. fclLocalVersion

---

### 4.8.1. インターフェース

char\* fclLocalVersion(void);

### 4.8.2. 機能

ライブラリがビルドされた日付を返します。

### 4.8.3. パラメータ

なし

### 4.8.4. 返回值

ライブラリがビルドされた日付を返します。

## 4.9. fclLocalRegisterJpegHandler

### 4.9.1. インターフェース

```
void fclLocalRegisterJpegHandler(void (*_fclJpegHandler)(uint32_t frame_no,const void* jpegbuf,int len));
```

### 4.9.2. 機能

カメラから映像フレームを受信したときに呼び出すコールバック関数を登録します。  
すでにコールバック関数が登録されている場合は、新しいものと置き換わります。  
コールバックを取り消す場合は、\_fclJpegHandlerパラメータにNULLを指定します。

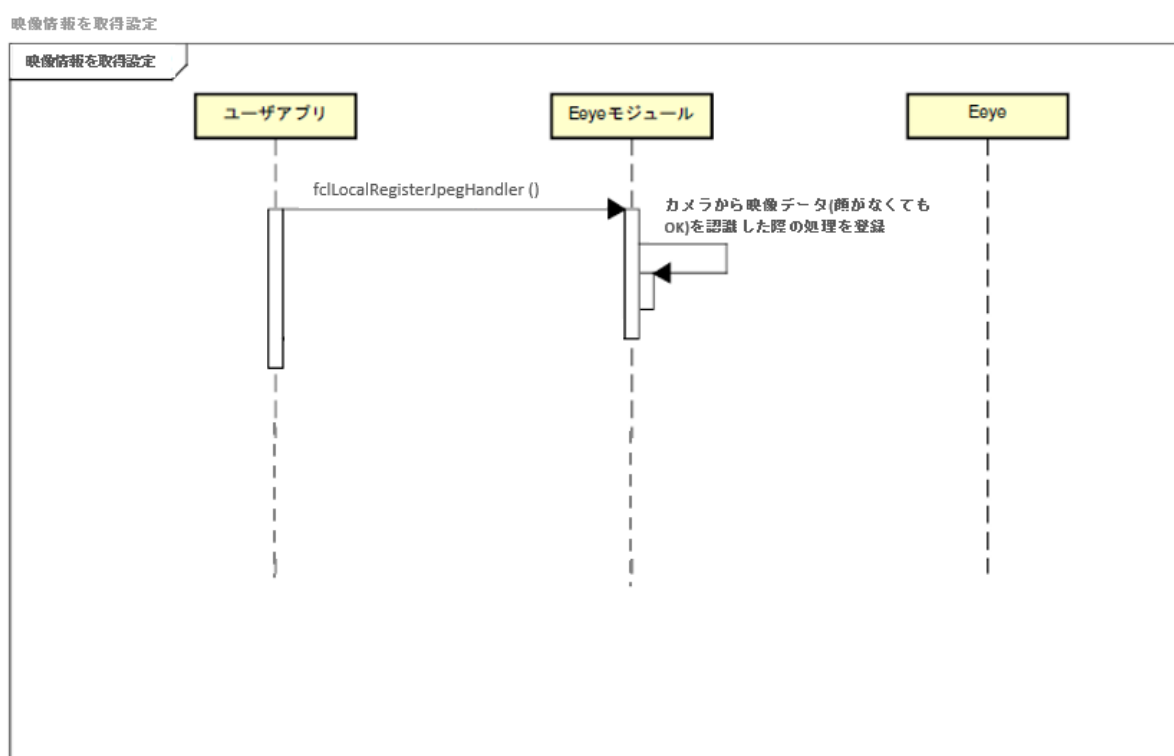
### 4.9.3. パラメータ

`void (*_fclJpegHandler)(uint32_t frame_no,const void*,int len)`      登録するコールバック関数

### 4.9.4. 戻り値

なし

### 4.9.5. 処理シーケンス



## 4.10. fclLocalStoreFeature

### 4.10.1. インターフェース

```
int fclLocalStoreFeature(FEATURE* features,int feature_count);
```

### 4.10.2. 機能

fclLocalRegister, fclLocalRegisterFromFile で登録したJPEG データを解析した特徴点を全てバッファへ書き出します。

features の要素は以下の構造となります。

```
typedef struct _FEATURE {
    uint32_t    id;
    uint8_t     feature[2048];
} FEATURE;
```

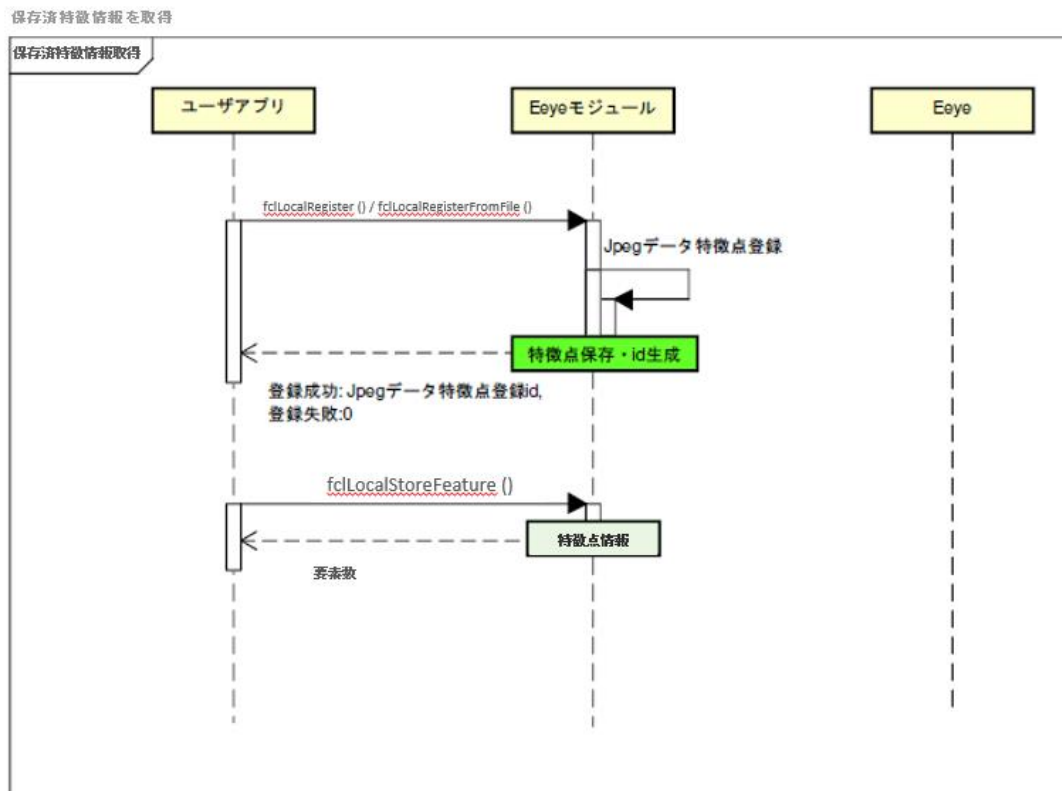
### 4.10.3. パラメータ

FEATURE* features	特徴点を受け取るバッファ
int feature_count	要素数で表したバッファの大きさ

### 4.10.4. 返り値

書き出された要素数

### 4.10.5. 処理シーケンス



## 4.11. fclLocalLoadFeature

---

### 4.11.1. インターフェース

```
int fclLocalLoadFeature(FEATURE* features,int feature_count);
```

### 4.11.2. 機能

fclLocalStoreFeature で出力した特徴点データを読み込みます。

features の要素は以下の構造となります。

```
typedef struct _FEATURE {  
    uint32_t    id;  
    uint8_t     feature[2048];  
} FEATURE;
```

### 4.11.3. パラメータ

FEATURE* features	特徴点バッファ
int feature_count	要素数

### 4.11.4. 返回值

読みこまれた要素数

## 4.12. fclLocalFeatureCount

---

### 4.12.1. インターフェース

int fclLocalFeatureCount(void);

### 4.12.2. 機能

fclLocalRegister, fclLocalRegisterFile で登録した件数を返します。

### 4.12.3. パラメータ

なし

### 4.12.4. 戻り値

fclLocalRegister, fclLocalRegisterFile で登録した件数を返します。

## 4.13. fclLocalRegisterFaceDataHandler

### 4.13.1. インターフェース

```
void fclLocalRegisterFaceDataHandler(void (*faceDataHandler)(uint32_t frame_no, FaceData* buf, int data_count, uint8_t* jpegbuf, int jpeglen));
```

### 4.13.2. 機能

顔を認識したときにその特徴データを取得するコールバックを登録します。  
登録を解除する場合は、パラメータにNULLを渡します。

### 4.13.3. パラメータ

```
void (*faceDataHandler)(int frame_no, FaceData* buf, int data_count, uint8_t* jpegbuf, int jpeglen)      登録するコールバック関数
```

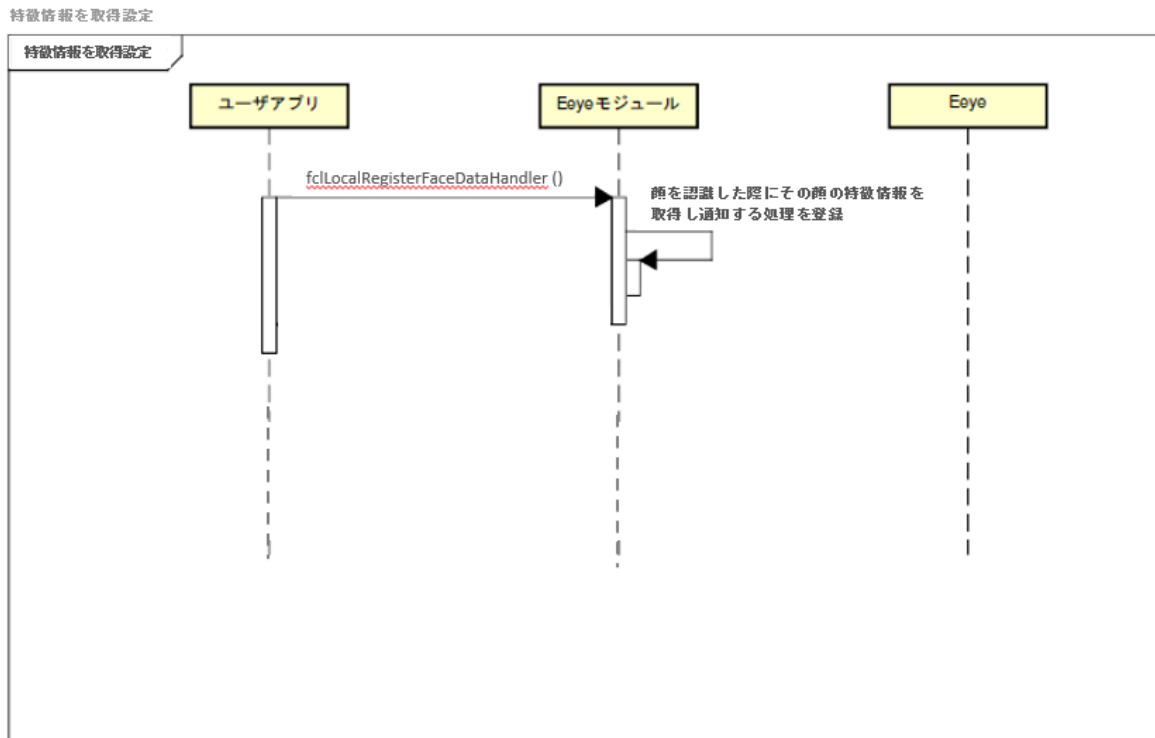
コールバックされる時のパラメータは以下のようになります。

int	frame_no	JPEG フレーム番号
FaceData*	buf	特徴データ
int	data_count	特徴データの件数
uint8_t*	jpegbuf	特徴データに付随するJPEGデータ
int	jpeglen	JPEGデータのバイト長

### 4.13.4. 戻り値

なし

### 4.13.5. 処理シーケンス



## 4.14. fclLocalCompair

---

### 4.14.1. インターフェース

```
float fclLocalCompair(const uint8_t* feature_1,const uint8_t* feature_2);
```

### 4.14.2. 機能

2つのfeature (FaceData中にあるFeature要素)を比較し結果を返します。

### 4.14.3. パラメータ

const uint8_t* feature_1	FaceData中のFeature要素
const uint8_t* feature_2	FaceData中のFeature要素

### 4.14.4. 返り値

比較した結果を返します。

0から1までの値で1が最も一致していることを示します。

経験上0.4以上となる場合、顔が一致したと認められます。

## 4.15. fclLocalGetFaceData

### 4.15.1. インターフェース

```
int fclLocalGetFaceData(FaceData* facedata,int max_facedata,const void* jpegbuf,int len);
```

### 4.15.2. 機能

JPEGデータからFaceDataを取り出します。

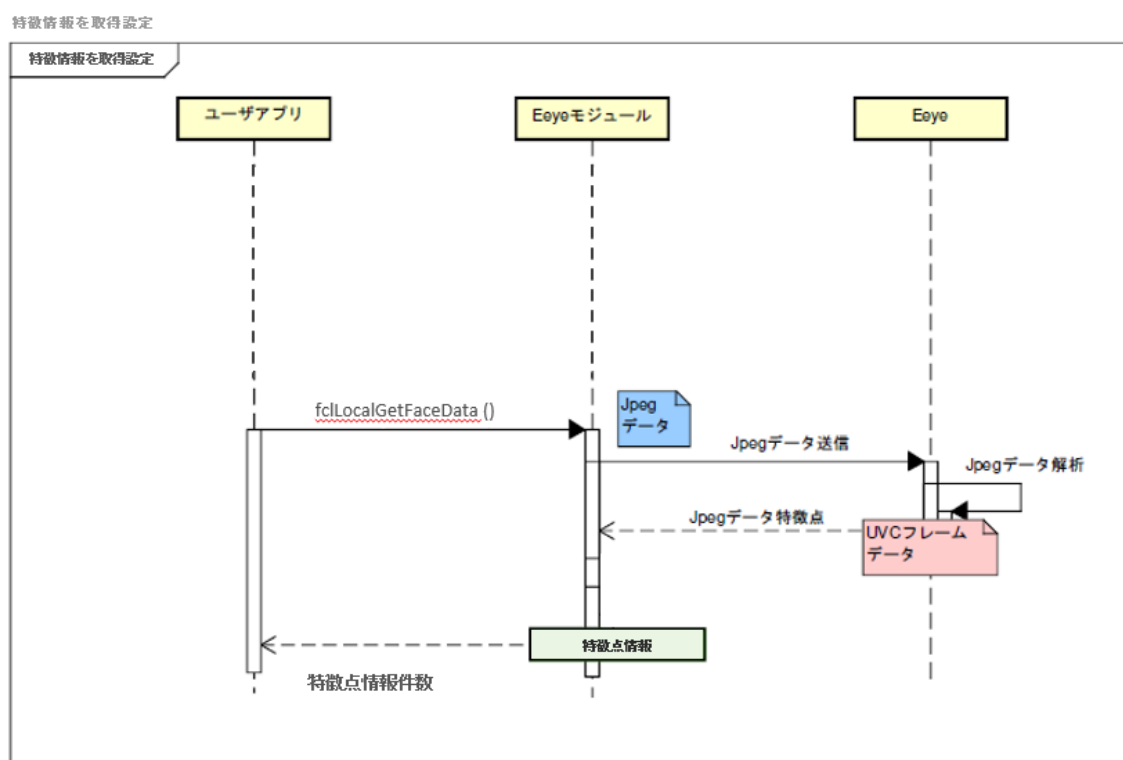
### 4.15.3. パラメータ

FaceData* facedata	FaceDataを受け取るバッファ
int max_facedata	上記バッファの大きさ(件数)
const void* jpegbuf	JPEGデータのバッファ
int len	JPEGデータの大きさ(バイト数)

### 4.15.4. 戻り値

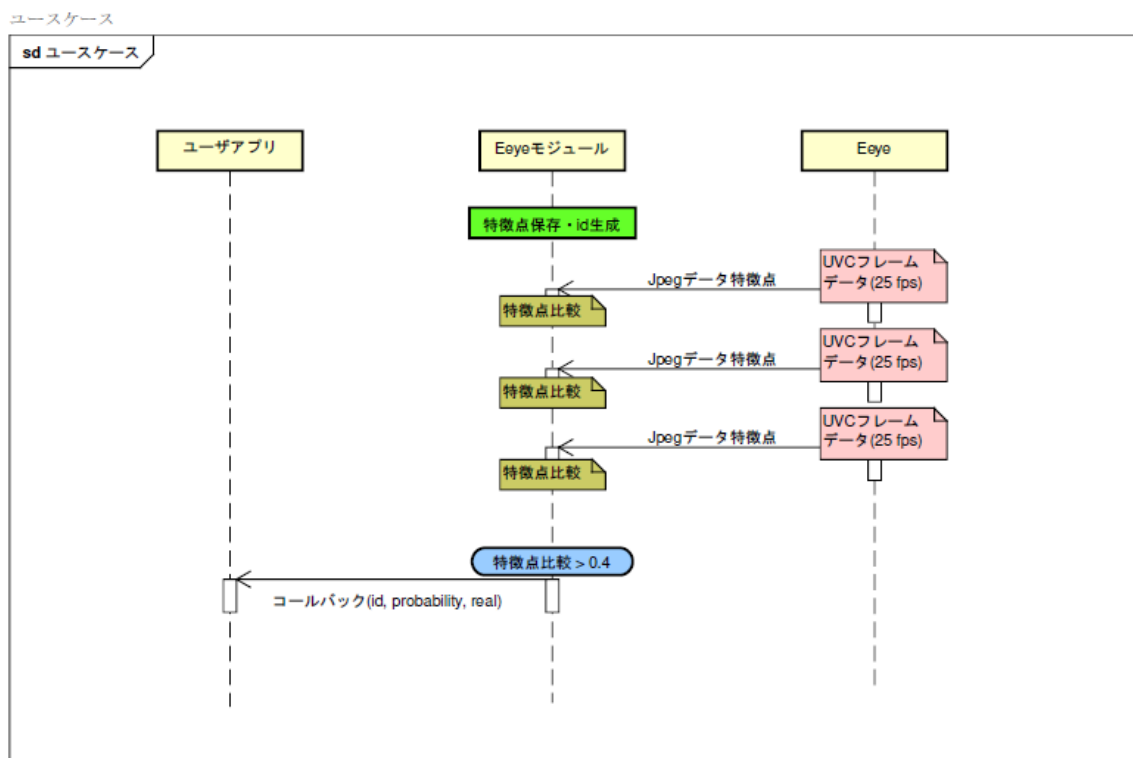
FaceDataの件数を返します。エラーまたは顔のないJPEGの場合は0が返ります。

### 4.15.5. 処理シーケンス



## 5. ユースケース

以下に本システムの参考ユースケースについて記載します。



## 6. FaceData の定義

---

FaceData の定義は以下の通りです。

```
typedef struct _FaceData{
    int32_t  track_id;          :顔を追跡するid。
    char     snap_type[16];    :顔のスナップショットタイプ

    int32_t  rect_left;        :顔枠左端
    int32_t  rect_top;         :顔枠上端
    int32_t  rect_right;       :顔枠右端
    int32_t  rect_bottom;      :顔枠下端

    float    blur;             :顔(映像)のボヤケ状態
    float    pose_roll;        :顔角度(傾斜)
    float    pose_yaw;         :顔角度(左右)
    float    pose_pitch;       :顔角度(上下)

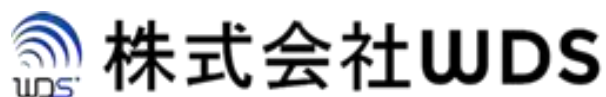
    uint8_t  feature[2048];    :顔特徴情報
    float    score;            :顔特徴情報抽出時の状態(顔の向き等)
    float    liveness;         :写真かリアルかの判定(1:リアル、0:写真)

    int32_t  attr_glasses;     :眼鏡の有無(1: 有、0:無)
    int32_t  attr_mouth_open;  :口の開閉(1: 開、0:閉)
    int32_t  attr_gender;      :性別(1: 男、0:女)
    int32_t  attr_age;         :見た目年齢
    int32_t  attr_beauty;      :顔のバランス(黄金比)
    int32_t  attr_eye_close;   :目の開閉(1: 開、0:閉)
    int32_t  attr_smile;       :口角の状態(1:笑っている、0:笑っていない)
} FaceData;
```

## 7. お問い合わせ先

お問合せ、ご質問につきましては以下へご連絡ください。

お問合せ先メールアドレス：[info@wd-s.com](mailto:info@wd-s.com)



株式会社WDS(ダブリューディーエス)

〒170-0005

東京都豊島区南大塚3-50-1 ウィンド大塚ビル412