



Linux版
開発キットソフトウェア仕様書
(ネットワークバージョン)

Version 0.9.2

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	
2020/05/11	表示ロゴ変更	WDS	

目 次

1. 概要	4
2. システム構成図	5
3. 提供ソフトウェア	5
3.1. ライブラリファイル	5
4. API	6
4.1. fc NetInit	6
4.2. fc NetRegisterMatchHandler	7
4.3. fc NetDeinit	8
4.4. fc NetSetThreshold	9
4.5. fc NetRegister	10
4.6. fc NetRegisterFromFile	11
4.7. fc NetUnregister	12
4.8. fc NetVersion	13
4.9. fc NetRegisterJpegHandler	14
4.10. fc NetStoreFeature	15
4.11. fc NetLoadFeature	16
4.12. fc NetFeatureCount	17
4.13. fc NetRegisterFaceDataHandler	18
4.14. fc NetCompare	19
4.15. fc NetGetFaceData	20
5. FaceData の定義	21
6. 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。

ネットワーク構成図

Eeyeカメラ開発キットのネットワーク構成は以下となります。

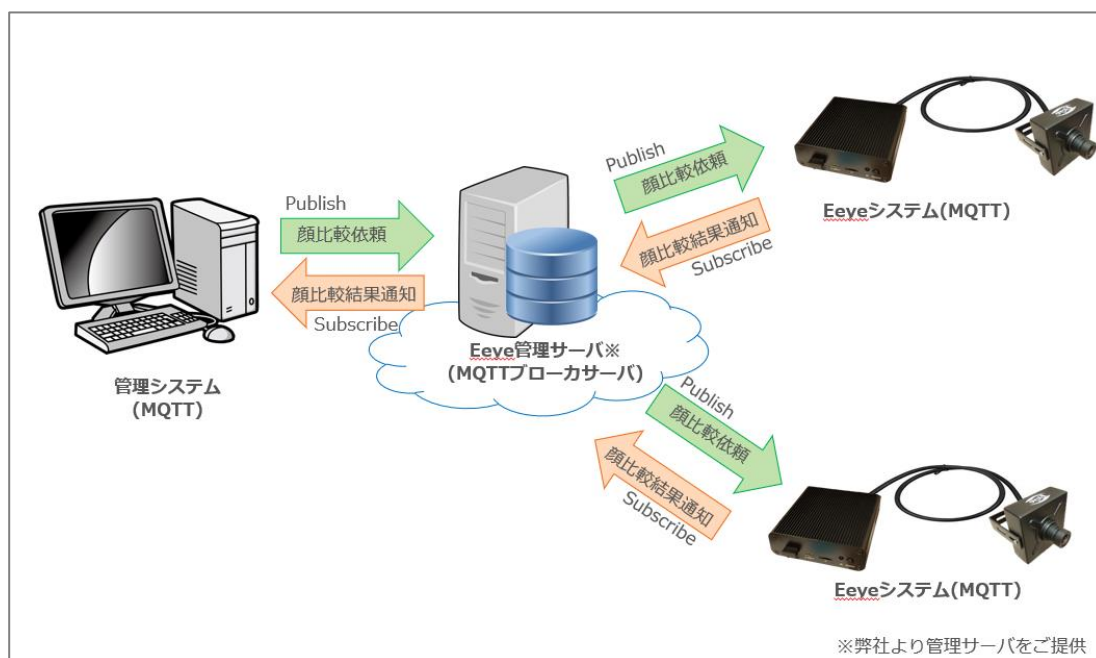


図1: ネットワーク構成図

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。

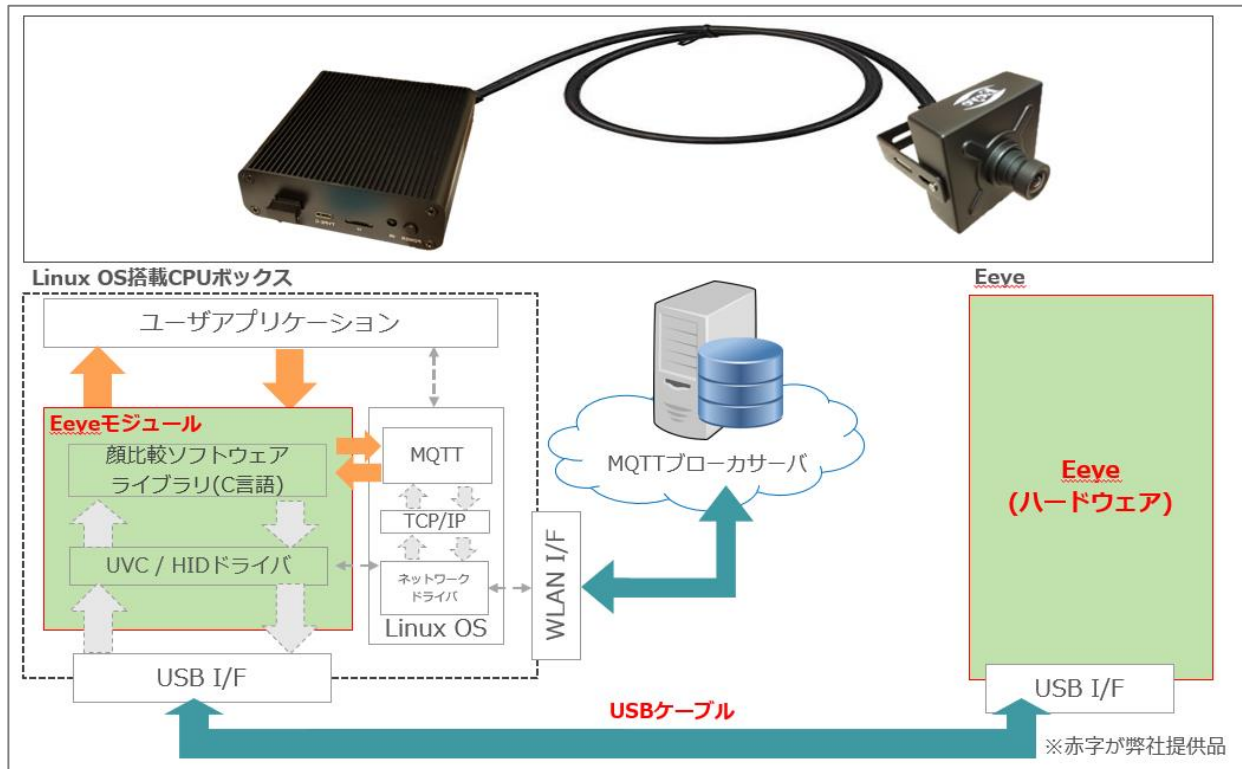


図2: Eeye開発キットシステム構成図

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言語ライブラリとして提供いたします。以降の章にて、提供ライブラリのAPIと処理シーケンスについて記載します。

3.1. ライブラリファイル

提供ファイル名は **libfcl.so** になります。

4. API

5. fclNetInit

5.1.1. インターフェース

```
bool fclNetInit(const char* address, const char* client_id);
```

5.1.2. 機能

ライブラリの初期化を行います。

5.1.3. パラメータ

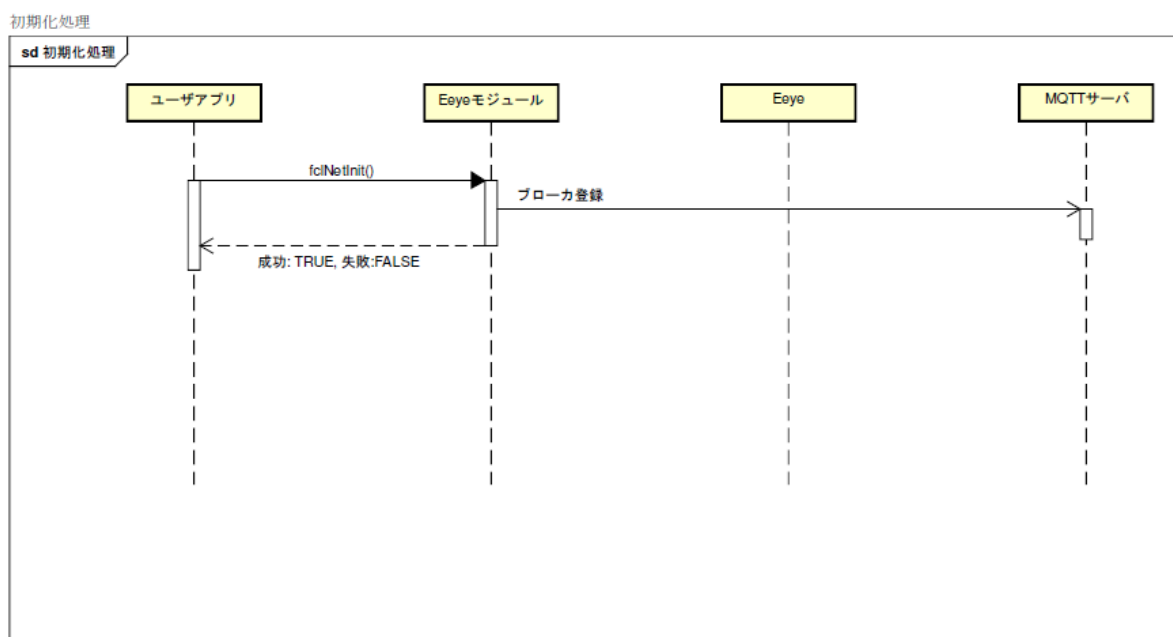
const char* address MQTT ブローカアドレスFDQN または IP アドレス‘:’に続けてポートを指定します。ポートを省略した場合は、1883が使用されます。

const char* client_id カメラがこのライブラリを識別する ID (ASCII8 文字以内)

5.1.4. 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。

5.1.5. 処理シーケンス



5.2. fclNetRegisterMatchHandler

5.2.1. インターフェース

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

5.2.2. 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。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。

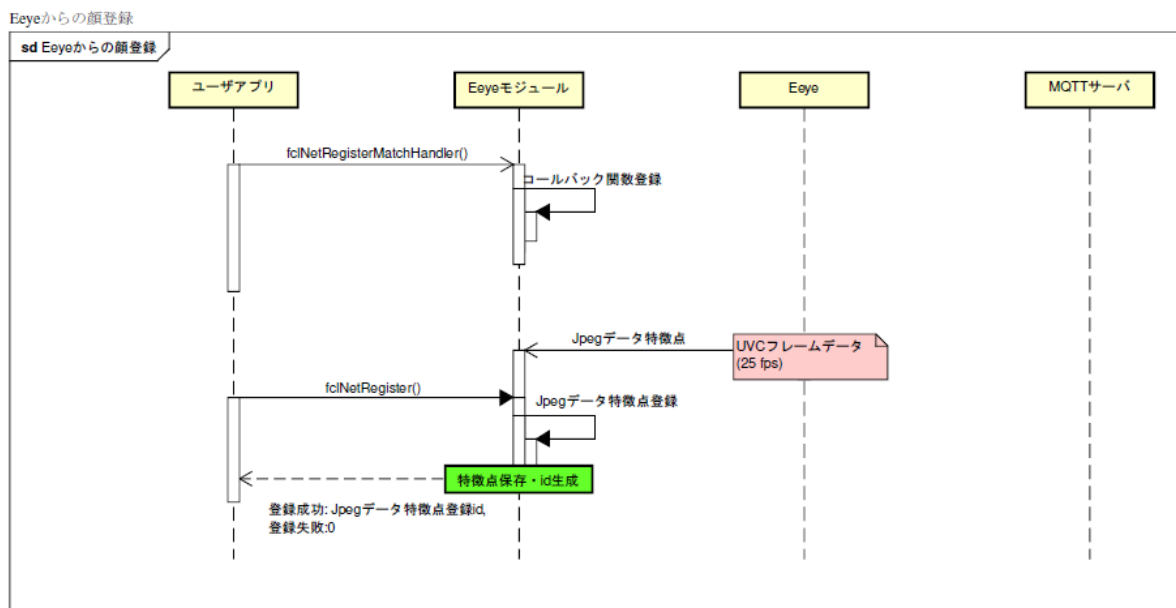
またreal はtrue がカメラが認識した顔が生顔(写真でない)を表します。

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。

5.2.3. パラメータ

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

5.2.4. 処理シーケンス



5.3. fclNetDeinit

5.3.1. インターフェース

```
void fclNetDeinit(void);
```

5.3.2. 機能

ライブラリの機能を停止しリソースを解放します。

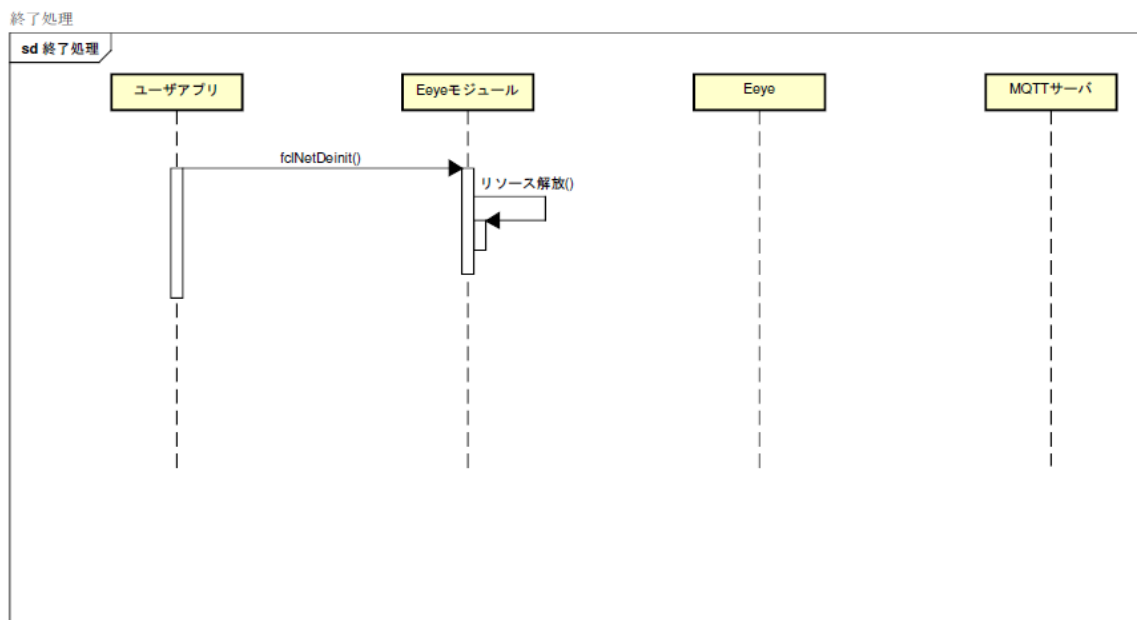
5.3.3. パラメータ

なし

5.3.4. 戻り値

なし

5.3.5. 処理シーケンス



5.4. fc1NetSetThreshold

5.4.1. インターフェース

```
bool fc1NetSetThreshold(const char* camid,float threshold);
```

5.4.2. 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。

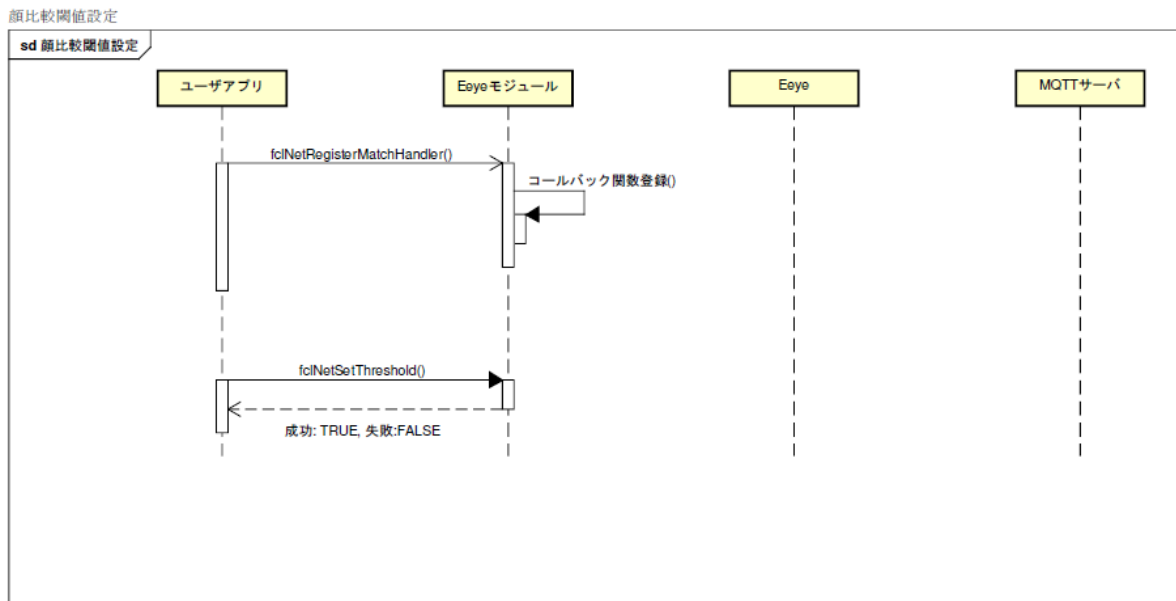
5.4.3. パラメータ

const char* camid	カメラID (ASCII8 文字以内)
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。

5.4.4. 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。

5.4.5. 処理シーケンス



5.5. fclNetRegister

5.5.1. インターフェース

```
uint32_t fclNetRegister(const char* camid, void* jpeg, int jpeg_length);
```

5.5.2. 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。

5.5.3. パラメータ

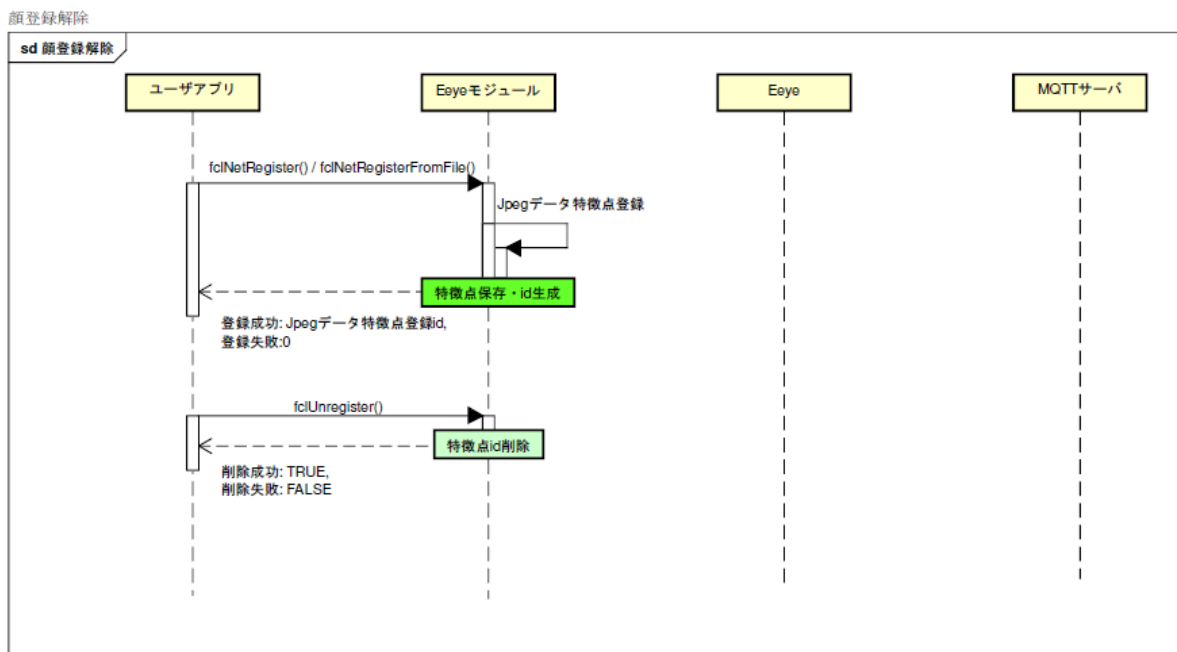
const char* camid	カメラID (ASCII8 文字以内)
void* jpeg	JPEG データを格納したバッファアドレス
int jpeg_length	JPEG データ長

5.5.4. 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

5.5.5. 処理シーケンス



5.6. fclNetRegisterFromFile

5.6.1. インターフェース

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

5.6.2. 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。

5.6.3. パラメータ

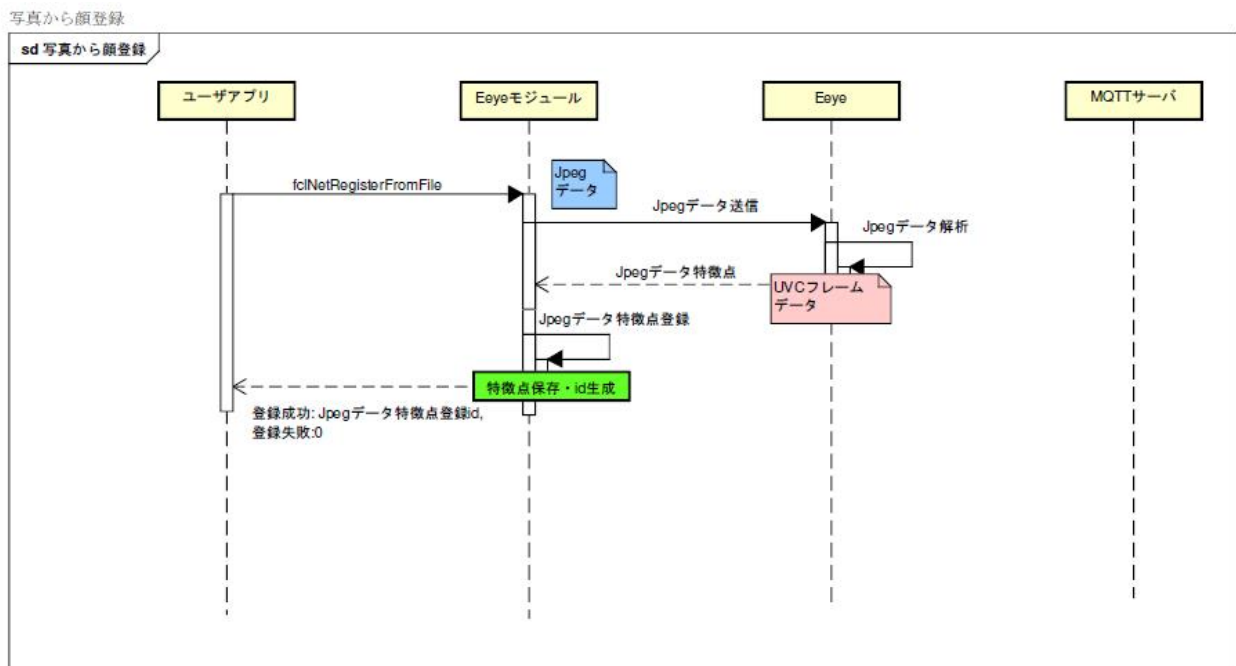
const char* camid	カメラID (ASCII8 文字以内)
const char* jpegpath	JPEG ファイルの path

5.6.4. 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。

5.6.5. 処理シーケンス

4.5.5.の処理シーケンス参照。



5.7. fclNetUnregister

5.7.1. インターフェース

```
bool fclUnregister(uint32_t id);
```

5.7.2. 機能

登録されているJPEG データの登録を削除します。

5.7.3. パラメータ

const char* camid	カメラID (ASCII8 文字以内)
uint32_t id	fclRegister/fclRegisterFromFile の呼び出しで返された id

5.7.4. 返り値

成功したときは TRUE、失敗したときは FALSE を返します。

5.7.5. 処理シーケンス

4.5.5.の処理シーケンス参照。

5.8. fc1NetVersion

5.8.1. インターフェース

char* fc1NetVersion(void);

5.8.2. 機能

ライブラリがビルドされた日付を返します。

5.8.3. パラメータ

なし

5.8.4. 返り値

ライブラリがビルドされた日付を返します。

5.9. fclNetRegisterJpegHandler

5.9.1. インターフェース

```
void fclNetRegisterJpegHandler( const char* camid,
    void (*_fclJpegHandler)(uint32_t frame_no,const void*,int len));
```

5.9.2. 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、パラメータに NULL を指定します。

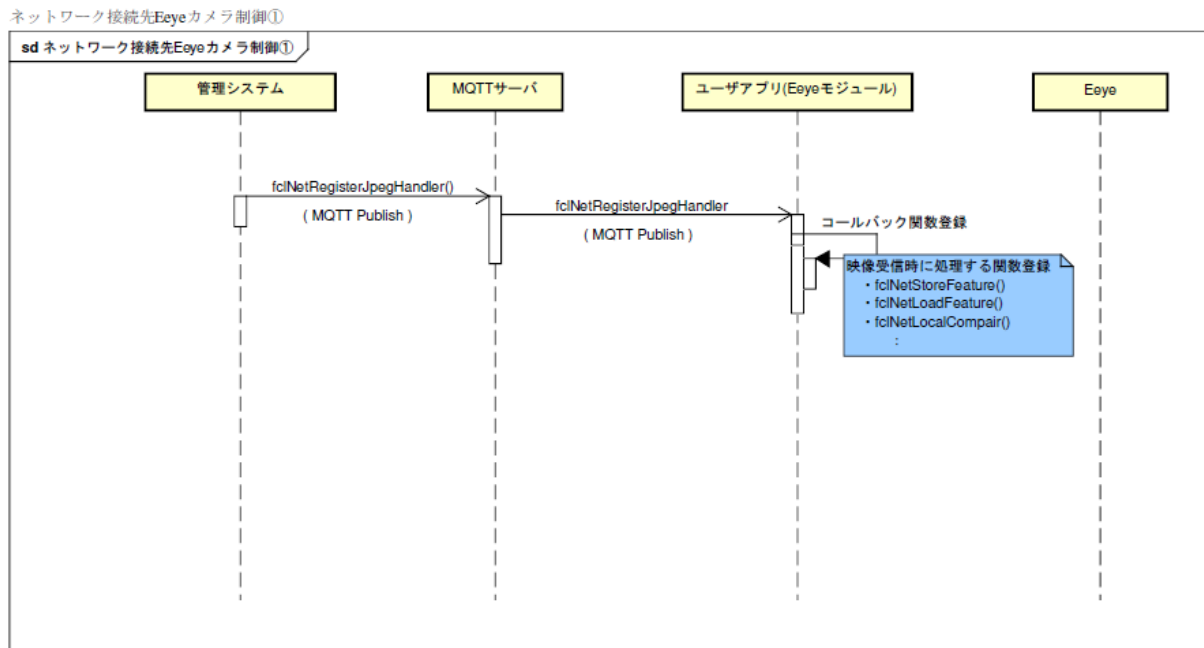
5.9.3. パラメータ

const char* camid	カメラID (ASCII8 文字以内)
void (*_fclJpegHandler)(uint32_t frame_no,const void*,int len)	登録するコールバック関数

5.9.4. 返り値

なし

5.9.5. 処理シーケンス



5.10. fclNetStoreFeature

5.10.1. インターフェース

```
int fclNetStoreFeature(const char* camid,FEATURE* features,int feature_count);
```

5.10.2. 機能

fclNetRegister,fclNetRegisterFile 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;
```

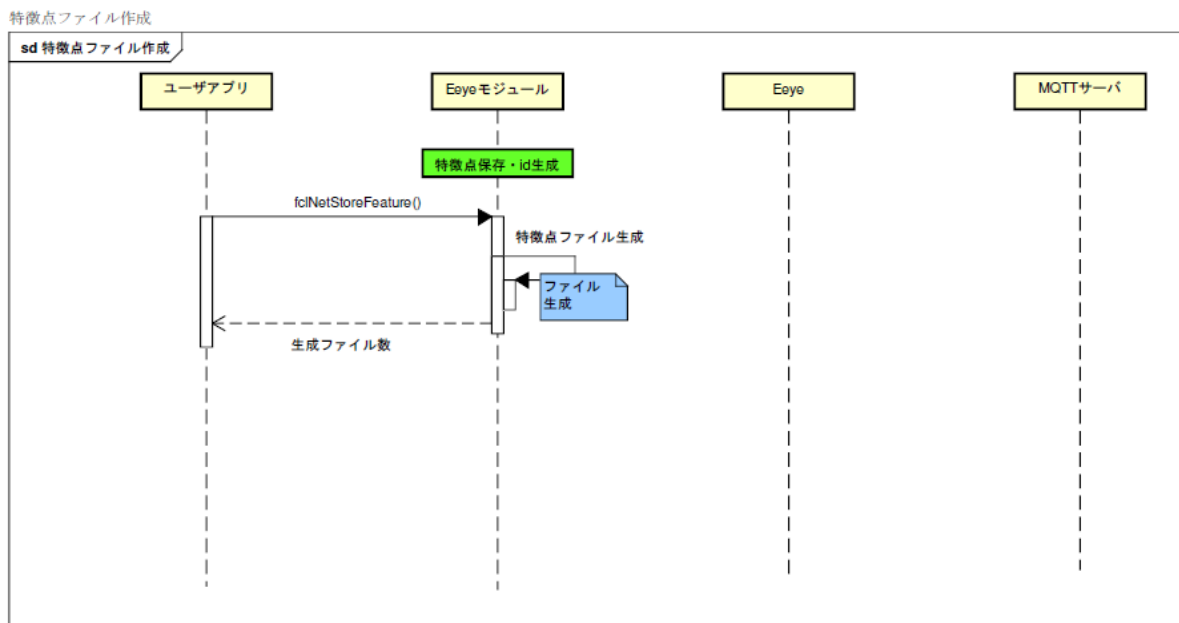
5.10.3. パラメータ

const char* camid	カメラID (ASCII8 文字以内)
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

5.10.4. 返り値

書き出された要素数

5.10.5. 処理シーケンス



5.11. fclNetLoadFeature

5.11.1. インターフェース

```
int fclNetLoadFeature(const char* camid,FEATURE* features,int feature_count);
```

5.11.2. 機能

fclNetStoreFeature 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;
```

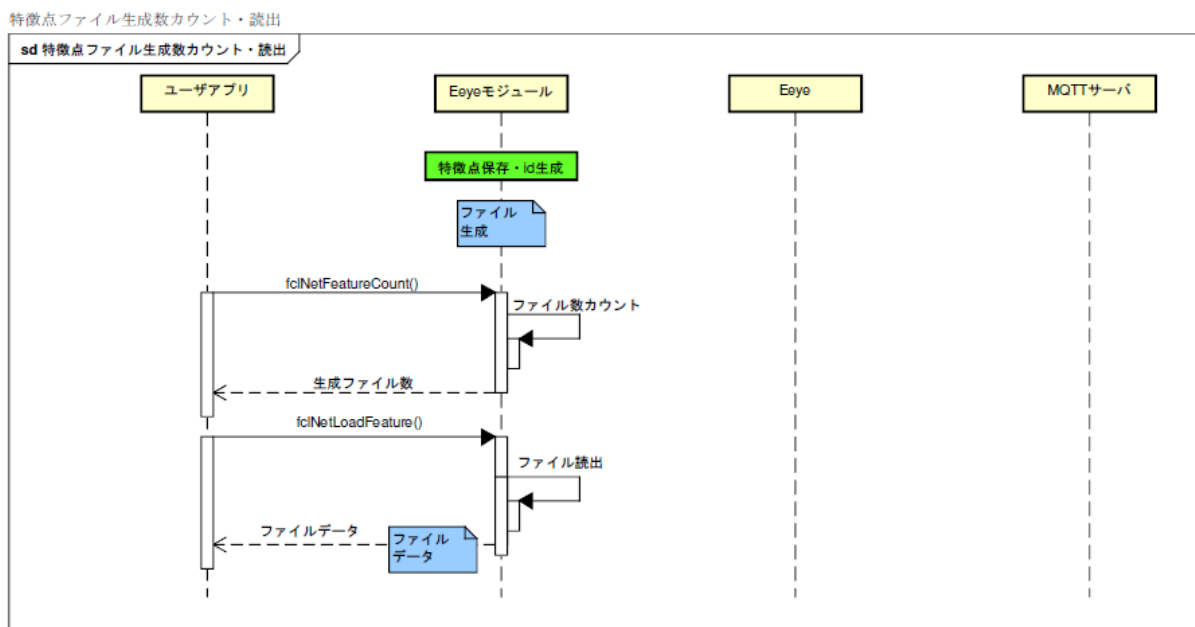
5.11.3. パラメータ

const char* camid	カメラID (ASCII8 文字以内)
FEATURE* features	特徴点バッファ
int feature_count	要素数

5.11.4. 返り値

読みこまれた要素数

5.11.5. 処理シーケンス



5.12. fclNetFeatureCount

5.12.1. インターフェース

```
int fclNetFeatureCount(const char* camid);
```

5.12.2. 機能

fclNetRegister, fclNetRegisterFile で登録した件数を返します。

5.12.3. パラメータ

const char* camid カメラID (ASCII8 文字以内)

5.12.4. 返り値

fclNetRegister, fclNetRegisterFile で登録した件数を返します。

5.12.5. 処理シーケンス

4.11.5.の処理シーケンス参照。

5.13. fc1NetRegisterFaceDataHandler

5.13.1. インターフェース

```
void fc1NetRegisterFaceDataHandler( const char* camid,
    void (*faceDataHandler)(FaceData* buf,uint8_t* jpegbuf,int jpeglen));
```

5.13.2. 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。

5.13.3. パラメータ

const char* camid	カメラID (ASCII8 文字以内)
void (*faceDataHandler)(FaceData*buf,uint8_t* jpegbuf,int jpeglen)	登録するコールバック関数

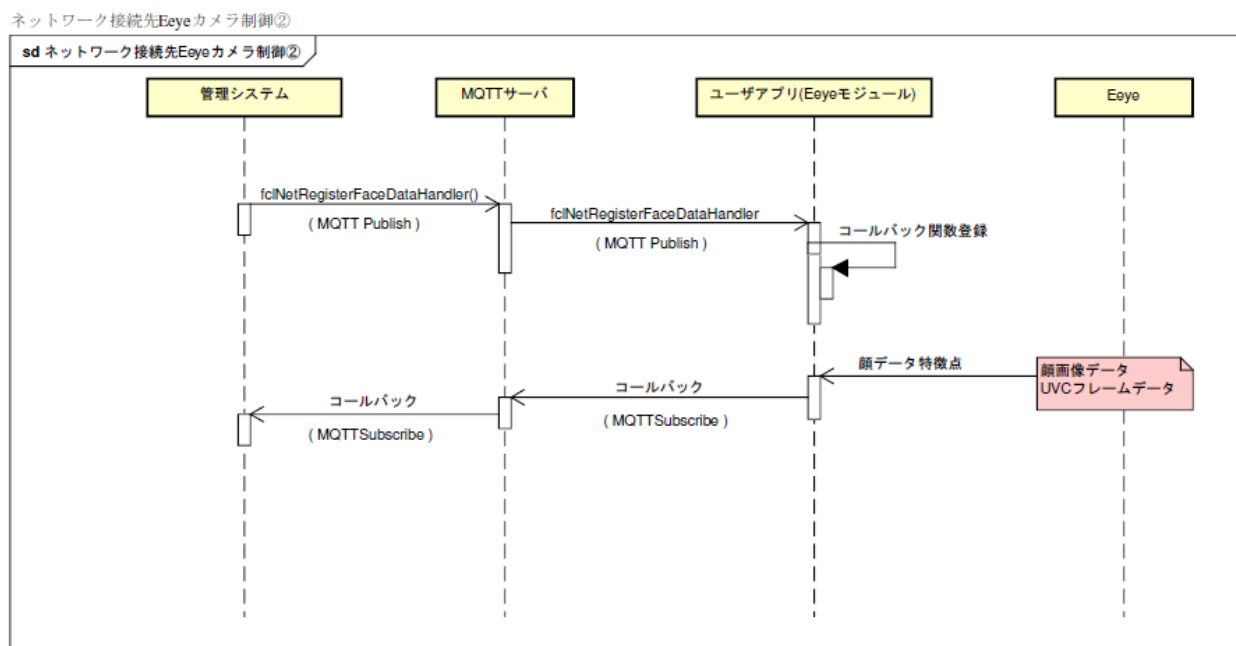
コールバックされるときのパラメータは以下のようになります。

FaceData*buf	特徴データ
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

5.13.4. 返り値

なし

5.13.5. 処理シーケンス



5.14. fclNetCompair

5.14.1. インターフェース

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

5.14.2. 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。

5.14.3. 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

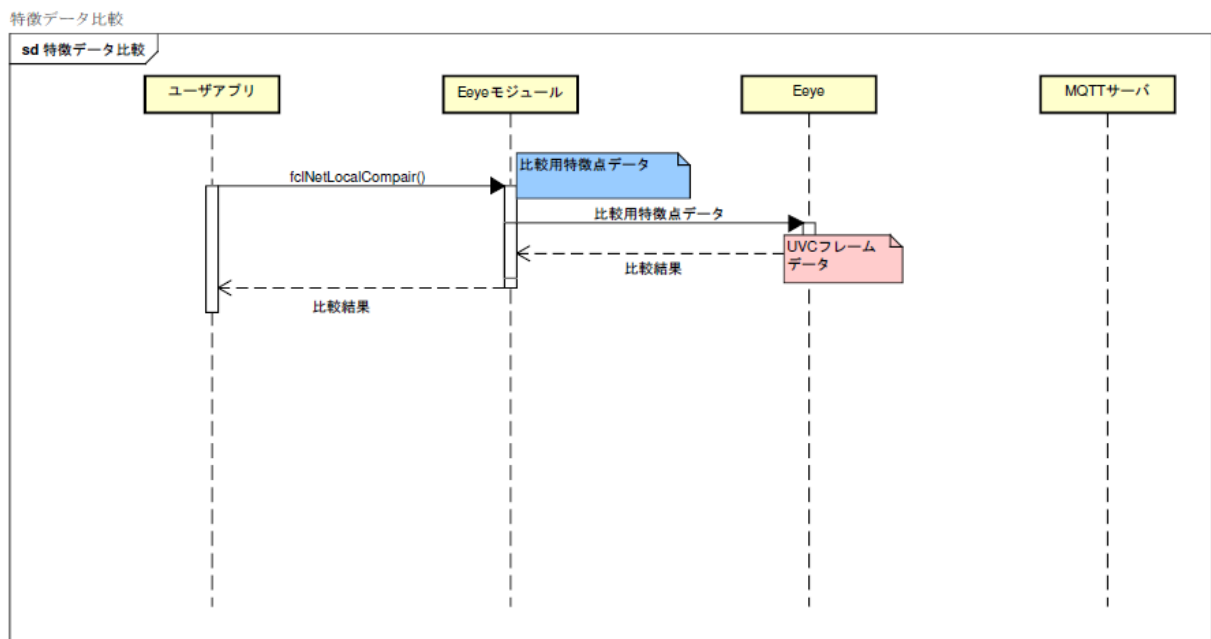
5.14.4. 返り値

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。

5.14.5. 処理シーケンス



5.15. fclNetGetFaceData

5.15.1. インターフェース

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

5.15.2. 機能

JPEG データからFaceData を取り出します。

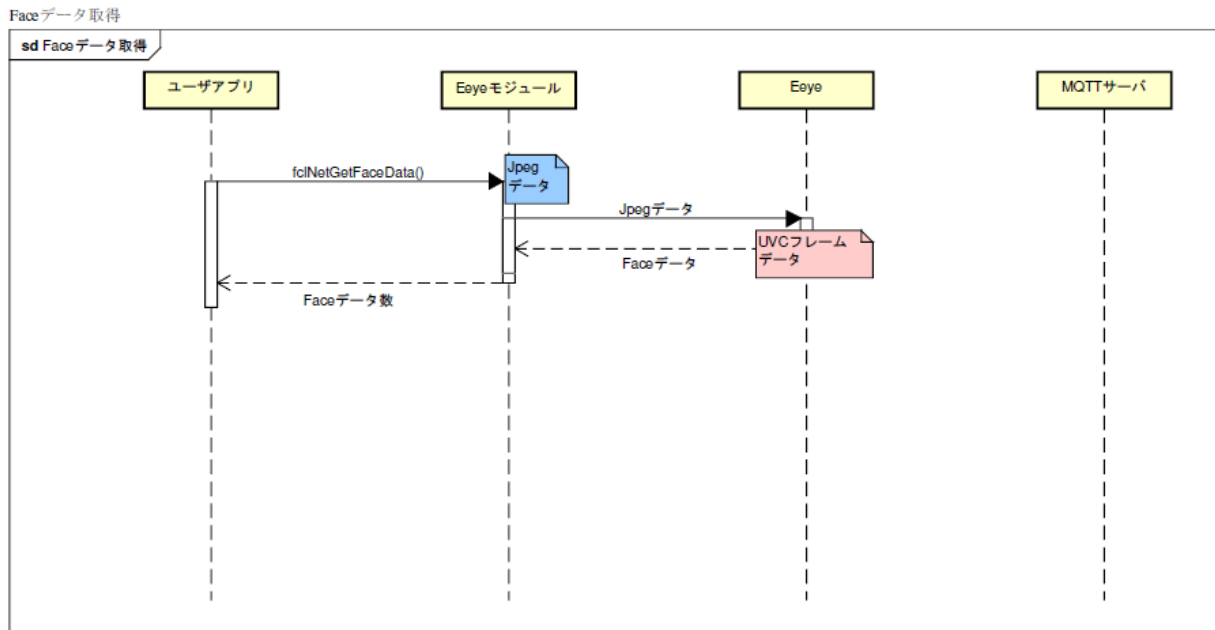
5.15.3. パラメータ

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)

5.15.4. 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。

5.15.5. 処理シーケンス



6. FaceData の定義

FaceData の定義は以下の通りです。

```
typedef struct _FaceData{
    int32_t track_id;
    char snap_type[16];

    int32_t rect_left;
    int32_t rect_top;
    int32_t rect_right;
    int32_t rect_bottom;

    float blur;
    float pose_roll;
    float pose_yaw;
    float pose_pitch;

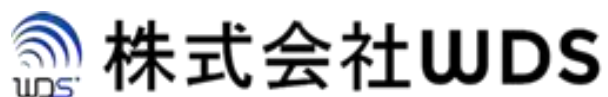
    uint8_t feature[2048];
    float score;
    float liveness;

    int32_t attr_glasses;
    int32_t attr_mouth_open;
    int32_t attr_gender;
    int32_t attr_age;
    int32_t attr_beauty;
    int32_t attr_eye_close;
    int32_t attr_smile;
} 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



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