Face and body Analysis Natural Computer Interaction

REST API iMX6 - PC (Controller)

Contents

1 Overview

This document describes the REST API between the iMX6 and the PC.
Section 4 describes which interfaces are used in which context / use case, while 5 and 6 describes the technical interface in detail.
In this document, the iMX6 Hardware will have the IP address 192.168.0.10:8000 while the PC hardware will have the IP address 192.168.0.20:9000.

2 General comments

- the *Action* tag in each of the transfered JSON structures is technically not needed, but helps when debugging
- every time an interface is called a TRACE event (6.1) should be send back to for debugging purposes.

3 Return Values for all REST API calls

The REST API supports these return value:

200	OK	request successful
403	Forbidden	page is called, but the server was not initialized before
400	Bad request	request has invalid input parameters
404	Not found	called page not found
501	Not Implemented	called page found, but function has to be implemented
503	Service Unavailable	call can not be completed successfully because of internal issues - more details optionally in returned JSON structure

Each API calls may return these values:

Result Code	Result	Additional Comments
200	call successful	
403	server was not initialized	
400	call failed - further information via LoggingEvent	
501	during development phase	

Additional values are listed separately at each function.

4 Use Cases / Context

This section describes very roughly, which commands are used in with use case / context.

4.1 Initialize

PC ⇒ iMX6	Initialize	(section 5.1 page 4)
PC ⇒ iMX6	SetLogLevel	(section 5.2 page 4)
iMX6 ⇒ PC	LoggingEvent	(section 6.1 page 14)
PC ⇒ iMX6	SetupGazeWorldModel	(section 5.12 page 11)
PC ⇒ iMX6	DefineGazeRegions	(section 5.13 page 12)
PC ⇒ iMX6	DefineGazeGestureEvents	(section 5.14 page 13)

4.2 Enrollment

PC ⇒ iMX6	EnumEnrollmentModules	(section 5.7 page 6)
PC ⇒ iMX6	StartEnrollment	(section 5.8 page 7)
PC ⇒ iMX6	EnrollmentDataEvent	(section 6.3 page 18)

4.3 Profile handling

PC ⇒ iMX6	EnumEnrollmentModules	(section 5.3 page 5)
PC ⇒ iMX6	CreateUserProfile	(section 5.4 page 5)
PC ⇒ iMX6	ActivateUserProfile	(section 5.5 page 6)
PC ⇒ iMX6	DeleteUserProfile	(section 5.6 page 6)

4.4 Use case handling

PC ⇒ iMX6	EnumUseCaseModules	(section 5.9 page 9)
PC ⇒ iMX6	StartUseCase	(section 5.10 page 10)
PC ⇒ iMX6	StopUseCase	(section 5.11 page 11)
PC ⇒ iMX6	UseCaseEvent	(section 6.2 page 14)

5 iMX6 Server

5.1 Initialize Server

5.1.1 IN

URL: *http://192.168.0.10:8000/Initialize/*
Body:

```
{"Action": "InitializeServer",
 "CallbackServer": "192.168.0.20:9000"
 "OnLoggingEventURL": "/LoggingEvent/"
 "OnEnrollmentEventURL": "/EnrollmentDataEvent/"
}
```

OnLoggingEventURL is the call back URL for logging events on the server running on the client side (see section 6.1).
OnEnrollmentEventURL is the call back URL for the steps during the enrollment phase (see section 6.3).

5.1.2 OUT

no out message - only result code

5.2 Set log level

5.2.1 IN

URL: *http://192.168.0.10:8000/SetMinLogLevel/*
Body:

```
{"Action" : "SetLogLevel",
 "MinLogLevel" : "<Minimal log level>"
}
```

Valid values for *MinLogLevel* are the TRACE, DEBUG, INFO, WARN, ERROR, FATAL (like in NLOG)

5.2.2 OUT

no out message - only result code

5.3 Enum user profiles

5.3.1 IN

URL: *http://192.168.0.10:8000/EnumUserProfiles/*
Body: currently optional - if more data has to be passed, it must be added here

```
{"Action": "EnumUserProfiles",
}
```

5.3.2 OUT

```
{"Action": "EnumUserProfilesResult",
"ProfileCount" : 3,
  "Profiles" : [
    "<Profile ID1>",
    "<Profile ID2>",
    "<Profile ID3>"
  ]
}
```

5.4 Create user profile

5.4.1 IN

URL: *http://192.168.0.10:8000/CreateUserProfile/*
Body:

```
{"Action" : "CreateUserProfile",
  "ClientAppUserID" : "<client user ID>"
}
```

<client user ID> is an arbitrary string that is unique on client side

5.4.2 OUT

```
{"Action": "CreateUserProfileResult",
  "ClientAppUserID" : "<client app user ID>",
  "FANCIUserID" : "<FANCI user ID>"
}
```

<FANCI user ID> is the ID that is used to store the data of the new user on the IMX6.

5.5 Activate user profile

5.5.1 IN

URL: *http://192.168.0.10:8000/ActivateUserProfile/*
Body:

```
{"Action" : "ActivateUserProfile",
  "FANCIUserID" : "<FANCI user ID>"
}
```

5.5.2 OUT

no out message - only result code

Result Code	Result	Additional Comments
200	Profile is activated	
400	Profile not activated - FANCIUserID unknown	

5.6 Delete user profile

5.6.1 IN

URL: *http://192.168.0.10:8000/DeleteUserProfile/*
Body:

```
{"Action" : "DeleteUserProfile",
  "FANCIUserID" : "<FANCI user ID>"
}
```

5.6.2 OUT

no out message - only result code

Result Code	Result	Additional Comments
200	Profile is activated	
400	Profile not deleted - FANCIUserID unknown	

5.7 Enum enrollment modules

5.7.1 IN

URL: *http://192.168.0.10:8000/EnumEnrollmentModules/*
Body: - currently optional - if more data has to be passed, it must be added here

```
{"Action": "EnumEnrollmentModules",
}
```

5.7.2 OUT

```
{"Action": "EnumEnrollmentModulesResult",
 "ModulesCount" : 3,
 "Modules" : [
   {
     "Module" : "Gaze",
     "StartEnrollmentUrl" : "http://192.168.0.10:8000/enroll_gaze"
     "Status" : "OK",
   },
   {
     "Module" : "Voice",
     "StartEnrollmentUrl" : "http://192.168.0.10:8000/enroll_voice"
     "Status" : "ERROR",
   },
   {
     "Module" : "Face",
     "StartEnrollmentUrl" : "http://192.168.0.10:8000/enroll_face"
     "Status" : "OK",
   }
 ]
}
```

Supported options for *Module* depend on the iMX6 hardware
Supported options for *Status* are: OK, ERROR. Enrollment of modules in Status ERROR can not be started. The Face module is doing enrollment for both Authentication and Emotion.

5.8 Start enrollment of a module

5.8.1 IN

URL: *http://192.168.0.10:8000/StartEnrollmentUrl/* - depends on the enrollment module - from *EnumEnrollmentModules*
Body: *Content* - currently optional - if more data has to be passed, it must be added here

```
{"Action": "StartEnrollment",
  "Module": "<module name>",
  "Parameter1": "<value1>",
  "Parameter2": "<value2",...
}
```

Valid values for *<module name>* are the modules returned in *EnumEnrollmentModules*. *ParameterX* and the corresponding value *<valueX>* are module specific.

5.8.2 OUT

```
{"Action": "StartEnrollmentResult",
  "Module": "<module name>",
  "Status": "<status>",
  "Message": "<module specific message>",
  "Timestamp": "<timestamp>
}
```

Supported options for *<module name>* are the modules returned in *EnumEnrollmentModules*.
Supported options for *<status>* are: OK, ERROR.
Supported options for *<module specific message>* are specific to the module.
<timestamp> is the timestamp in milliseconds of the iMx6 server. If the status is ERROR, then the call has returned 503.
If the status is OK, then enrollment has started, and the call has returned 200.

Face For Face enrollment, the specific input parameters are the following:

- *FANCIUserID* the id of the user to enroll.
- *DurationSeconds* the amount of time used for enrollment. If not specified it is default to 5 seconds.

The output *<module specific message>* is one of the following:

- *SUCCESS* if the *Status* is *OK*.
- *SERVER_NOT_INITIALIZED* if the *Status* is *ERROR* because the server was not initialized.
- *SERVER_ERROR* if the *Status* is *ERROR* because the server had an error.
- *MISSING_RequiredParameter* if the *Status* is *ERROR* because the *RequiredParameter* was not set.
- *WRONG_RequiredParameter* if the *Status* is *ERROR* because the *RequiredParameter* was not valid.
- *NO_FACE_DETECTED* if the *Status* is *ERROR* because no face was detected.
- *NOT_ENOUGH_FACE_DETECTED* if the *Status* is *ERROR* because not enough faces were detected.
- *MULTIPLE_FACES_DETECTED* if the *Status* is *ERROR* because multiple faces were detected at the same time.
- *INTERNAL_ERROR* if the *Status* is *ERROR* because one of the authentication or emotion core library failed.
- *CAMERA_BUSY* if the *Status* is *ERROR* because the camera is already used by some other process.
- *CAMERA_FAILED* if the *Status* is *ERROR* because the camera could not be opened.
- *UNKNOWN_ERROR* if the *Status* is *ERROR* because there was an unknown error.

5.9 Enum use case modules

5.9.1 IN

URL: *http://192.168.0.10:8000/EnumUseCaseModule/*
Body: *Content* - currently optional - if more data has to be passed, it must be added here

```
{"Action": "EnumUseCaseModules",
}
```

5.9.2 OUT

```
{"Action": "EnumUseCaseModulesResult",
  "UseCaseCount" : 4,
  "UseCases" : [
    {
      "UseCase" : "GestureDetection",
      "Status" : "OK",
      "Enabled" : "true",
      "StartUrl" : "http://192.168.0.10:8000/start_gesturedetection"
      "StopUrl" : "http://192.168.0.10:8000/stop_gesturedetection"
      "ModuleDetails" :
        {
           "SupportedGestures" : ["WipeLeft", "WipeRight", "WipeUp" ],
           "SupportedPostures" : ["ThumpUp", "ThumpDown"],
           "..." : "..."
        }
    },
    {
      "Module" : "GazeTracking",
      "Status" : "OK",
      "Enabled" : "true",
      "StartUrl" : "http://192.168.0.10:8000/start_gazetracking"
      "StopUrl" : "http://192.168.0.10:8000/stop_gazetracking"
      "ModuleDetails" :
        {
          "TrackerURLs" : [
              "tobii-ttp://is4be-030105141715",
              "tobii-ttp://is4be-030105042734"
              ],
          "..." : "..."
        }
    },
    {
      "Module" : "Emotion",
      "Status" : "OK",
      "Enabled" : "true",
      "ModuleDetails" :
        {
          "..." : "..."
        }
    },
    {
      "Module" : "Authentication",
      "Status" : "OK",
      "Enabled" : "false",
      "StartUrl" : "http://192.168.0.10:8000/start_authentication"
      "StopUrl" : "http://192.168.0.10:8000/stop_authentication"
      "ModuleDetails" :
        {
          "..." : "..."
        }
    },
    {
      "Module" : "EmotionDetection",
      "Status" : "ERROR",
      "Enabled" : "false",
      "StartUrl" : "http://192.168.0.10:8000/start_emotiondetection"
      "StopUrl" : "http://192.168.0.10:8000/stop_emotiondetection"
      "ModuleDetails" :
        {
          "..." : "..."
        }
    }
  ]
}
```

Supported options for *Module* depend on the iMX6 hardware
Supported options for *Status* are: OK, ERROR.
Supported options for *Enabled* are: false, true.
The structure of *ModuleDetails* is sensor dependent - this is not JSON conform, but we will deal with that.

5.10 Start use case

5.10.1 IN

URL: *http://192.168.0.10:8000/StartUseCaseUrl/* - depends on the use case - from *Enum use case modules*
Body: *Content*

```
{"Action": "StartUseCase",
  "CallbackUrl" : "http://192.168.0.20/UseCaseEvent/UseCaseSpecificUrl",
  <Use Case Dependent Settings>
}
```

5.10.2 OUT

no out message - only result code

5.11 Stop use case

5.11.1 IN

URL: *http://192.168.0.10:8000/StopUseCaseUrl/* - depends on the use case - from *Enum use case modules*
Body: *Content*

```
{"Action": "StopUseCase",
}
```

5.11.2 OUT

no out message - only result code

5.12 Setup gaze world model

5.12.1 IN

URL: *http://192.168.0.10:8000/SetupGazeWorldModel/*
POST parameter: *Content*

```
{"Action" : "SetupGazeWorldModel",
 "TrackerCount" : 2
 "Trackers" : [
   {
     "active" : "true",
     "tracker_url" : "tobii-ttp://is4be-030105141715",
     "tracker_to_monitor_x_offset_mm" : 0,
     "monitor_width_mm" : 292.6,
     "monitor_height_mm" : 109.7,
     "monitor_width_px" : 1024,
     "monitor_height_px" : 768,
     "tracker_angle_y_degree" : 0,
     "position" : "center"
   },
   {
     "active" : "true",
     "tracker_url" : "tobii-ttp://is4be-030105042734",
     "tracker_to_monitor_x_offset_mm" : -132.5,
     "monitor_width_mm" : 264.5,
     "monitor_height_mm" : 165.2,
     "monitor_width_px" : 800,
     "monitor_height_px" : 600,
     "tracker_angle_y_degree" : -10,
     "position" : "right"
   }
 ]
}
```

5.12.2 OUT

no out message - only result code

5.13 Define gaze regions

5.13.1 IN

URL: *http://192.168.0.10:8000/DefineGazeRegions/*
Body:

```
{"Action" : "DefineGazeRegions",
 "InteractorCount" : 2,
 "Interactors" : [
   {
     "InteractorID" : <InteractorID>,
     "ScreenPosition" : <ScreenPosition>,
     "RectShapeRegionCount" : 2,
     "RectShapeRegions" : [
       {
         "X" : 10, "Y" : 10, "W" : 10, "H" : 10,
         "Weight" : 64
       },
       {
         "X" : 30, "Y" : 30, "W" : 10, "H" : 10,
         "Weight" : 64,
       }
      ],
     "CircleShapeRegionCount" : 2,
     "CircleShapeRegions" : [
            {
              "X" : 40, "Y" : 40, "R" : 10,
              "Weight" : 64
            },
            {
              "X" : 50, "Y" : 50, "R" : 30,
              "Weight" : 64,
            }
          ],
     "ZOrder" : 100
   },
   {
     "InteractorID" : <InteractorID>,
     "RectShapeRegionsCount" :1,
     "RectShapeRegions" : [
       {
         "X" : 10, "Y" : 10, "W" : 10, "H" : 10,
         "Weight" : 64
       }
     ],
     "CircleShapeRegionsCount" : 0,
     "CircleShapeRegions" : [],
     "ZOrder" : 50
   }
 ]
}
```

A call of *DefineGazeRegions* always defined updates the *complete* gaze region set.
The *InteractorID* is defined by the client application.
The *ScreenPosition* is defined in the gaze world model.
The *Weight* is a score that influences how likely a region is to be selected. This is useful if two regions are close to each other, to avoid "flickering" between the two.

5.13.2 OUT

no out message - only result code

5.14 Define gaze + gesture events

5.14.1 IN

URL: *http://192.168.0.10:8000/DefineGazeGestureEvents/*
Body

```
{"Action" : "DefineGazeGestureEvents",
 "EventsCount" : 3
 "Event" : [
    {
      "EventID" : "<EventID1>",
      "InteractorID" : "Button1",
      "Gesture" : "HighFive",
    },
    {
      "EventID" : "<EventID2>",
      "InteractorID" : "Button2",
      "Gesture" : "HighFive",
    },
    {
      "EventID" : "<EventID3>"
      "InteractorID" : "Button1",
      "Gesture" : "WipeLeft",
    },
  ]
}
```

5.14.2 OUT

no out message - only result code

6 PC Server (server running on the client for callbacks)

6.1 Logging / Warning / Errors

6.1.1 IN

URL: *http://192.168.0.20:9000/LoggingEvent/*
Body:

```
{"Action": "LoggingEvent",
 "Level": "<log level>",
 "Message" : "<message>",
 "Description" : "<description>"
}
```

Valid values for *<log level>* are the TRACE, DEBUG, INFO, WARN, ERROR, FATAL (like in NLOG)
<message> is a short description of the logging event.
<description> is an optional detailed description of the logging event.

6.1.2 OUT

no out message - only result code

6.2 Use Case Event

6.2.1 IN

URL: *http://192.168.0.20:9000/UseCaseEventUrl/*
The URL is the *CallbackUrl* given when starting the use case (section 5.10). Body:

```
  {"Action": "UseCaseEvent",
  "Module": "<module name>",
  "Timestamp": "<timestamp>",
  "EventData" :
  {
    <event data structure - depends on <module name>>
  }
}
```

Valid values for *<module name>* are the *Module* values returned by *EnumUseCaseModules* (section 5.9)
The value of *EventData* depends on the sensor module which triggered the event.
timestamp is the timestamp of the iMX6 server.

Authentication

For the authentication use case, five events can be generated. The json returned is one of the following:

```
{"Action": "UseCaseEvent",
 "Module": "Authentication",
 "Timestamp": "<timestamp>",
 "EventData" : {
   "Name" : "Engagement"
  }
}
```

or

```
{"Action": "UseCaseEvent",
 "Module": "Authentication",
 "Timestamp": "<timestamp>",
 "EventData" : {
   "Name" : "NewUserID",
   "User" : "<FANCIUserID>",
   "Score" : "<AuthenticationScore>"
  }
}
```

or

```
{"Action": "UseCaseEvent",
 "Module": "Authentication",
 "Timestamp": "<timestamp>",
 "EventData" : {
   "Name" : "NoFace"
  }
}
```

or

```
{"Action": "UseCaseEvent",
 "Module": "Authentication",
 "Timestamp": "<timestamp>",
 "EventData" : {
   "Name" : "Impostor",
   "Score" : "<AuthenticationScore>"
  }
}
```

or

```
{"Action": "UseCaseEvent",
 "Module": "Authentication",
 "Timestamp": "<timestamp>",
 "EventData" : {
   "Name" : "SpoofingAttempt",
   "User" : "<FANCIUserID>",
   "Score" : "<AuthenticationScore>"
  }
}
```

where *FANCIUserID* is the ID of the user who was authenticated, or whom someone tried to spoof; and *AuthenticationScore* is a float value in $[-1.0, 1.0]$ is the score output.

Mood

```
{"Action": "UseCaseEvent",
 "Module": "MoodChange",
 "Timestamp": "<timestamp>",
 "EventData" : {
   "Name" : "<NewMoodName>"
  }
}
```

where *NewMoodName* is one of *NewMoodAnger*, *NewMoodDisgust*, *NewMoodFear*, *NewMoodHappiness*, *NewMoodSadness* or *NewMoodSurprise*.

Gesture

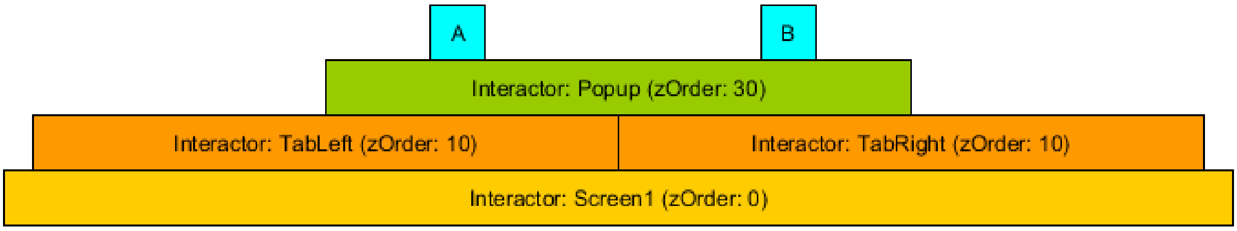
```
{"Action": "UseCaseEvent",
 "Module": "Gesture",
 "Timestamp": "<timestamp>",
 "EventData" : {
    "Name" : "NewHandGesture",
    "Gesture" : "<GestureOrPostureName>",
  }
}
```

where *GestureOrPostureName* is one of *BIG_5, THUMB_UP, V_LETTER, FIRST_3, FINGERS4, PHONE, STOP, WAVE, SWIPE_DOWN, SWIPE_UP, SWIPE_LEFT* or *SWIPE_RIGHT*. These postures are described in SoftKinetic Close Interaction Library.

Gaze

```
{"Action": "UseCaseEvent",
 "Module": "Gaze = ",
 "Timestamp": "<timestamp>",
 "EventData" : {
    "Name" : "GazeData",
    "Interactor" : "<InteractorID>",
    "Event" : "Enter",
    "InteractorStackCount" : 3,
    "InteractorStack" : [
      "<InteractorID>",
      "<InteractorIDBelow1>",
      "<InteractorIDBelow2>"
    ]
  }
}
```

Valid values for *<InteractorID>* and *InteractorStack* are the values uses in *DefineGazeRegions* (section 5.13)
Valid values for *Event* are the *Enter* and *Leave*
InteractorStack will return the stack of all interactors in the order of the zOrder.


A
B
Interactor: Popup (zOrder: 30)
Interactor: TabLeft (zOrder: 10)
Interactor: TabRight (zOrder: 10)
Interactor: Screen1 (zOrder: 0)

p.e.

- if you gaze at *A* the value of Interactor should be *Popup* and the value of InteractorStack should be *Popup, TabLeft, Screen1*
- if you gaze at *A* the value of Interactor should be *Popup* and the value of InteractorStack should be *Popup, TabRight, Screen1*

Gaze-Gesture-Fusion

```
{"Action": "UseCaseEvent",
 "Module": "GestureGesture",
 "Timestamp": "<timestamp>",
 "EventData" : {
   "Name" : "GazeGestureData",
   "EventID" : "<EventID>",
 }
}
```

6.2.2 OUT

no out message - only result code

6.3 Enrollment Data Event

6.3.1 IN

URL: *http://192.168.0.20:9000/EnrollmentDataEvent/*
GET parameter: *Module*=<module name>
Body:

```
{"Action": "EnrollmentDataEvent",
  "Module": "<module name>",
  "Progress" : <Progress as integer value 0..100>,
  "Step" : "<Tag of the enrollment step>"
}
```

The valid values for *Step* depend on the value of *Module*:

- Gaze
  - LookAtGazeCalibrationPostition1
  - LookAtGazeCalibrationPostition2
  - ....
  - LookAtGazeCalibrationPostitionX

  The gaze calibration point coordinates, numbering and amount are entered in the gaze world model.

6.3.2 OUT

no out message - only result code