

API連携による組み込みシステムへのAI/IoT機能の実装

常三島技術部門
情報システムグループ

辻 明典 (TSUJI Akinori)

1. はじめに

AIやIoT技術の発展に伴い、様々な分野で業務の効率化や自動化が実現されている。これまで特定の機能に特化していた組み込みシステムにもAIやIoT技術が導入され、より高度な情報処理、ネットワーク連携が急速に進んでいる。本報告では、API連携を用いて組み込みシステムにAI/IoT機能を実装する方法について述べる。API連携により、クラウド上の多様なウェブサービスを利用できるようになり、組み込みシステムの機能を容易に拡張でき、新たな価値の創出が期待できる。

2. 概要

組み込みシステムにAI/IoT機能を実装するには、いくつかの課題がある。組み込みシステムは一般にプロセッサの演算能力、メモリ容量、消費電力に制限がある。一方で、AIモデルの実装には膨大な計算量を要し、そのままでは組み込みシステムへの搭載は困難である。また、IoTデバイスからの大量のデータを記録し、リアルタイムで処理するには十分な処理能力と記憶領域が不可欠である。さらに、ネットワーク通信を伴うシステムでは、通信環境の不安定さによる遅延や切断がシステムの誤動作を引き起こすリスクがある。セキュリティ面では、外部からの不正アクセスやデータ漏洩の危険性があるため慎重な対策が求められる。このように、AI/IoT機能を組み込みシステムに搭載するには、ハードウェア、ソフトウェア、ネットワーク、ストレージ、セキュリティ等、多岐に渡る技術の統合が必須となる。そのため、システム全体の設計および開発が複雑化し、開発期間の長期化やコストの増大につながる課題が顕在化する。

これらの課題解決には、クラウドコンピューティングを活用したAPI (Application Programming Interface) 連携が有効である。APIは、異なるソフトウェアやサービス同士を相互に

接続するためのインタフェースである。APIの利用により、組み込みデバイスはデータの収集や簡易な処理に専念でき、より複雑な演算や学習はクラウド上のAIサービスやデータ処理サービスに委譲できる。また、クラウド上のデータベースとのAPI連携により、大量のデータを効率的に蓄積・管理できる。セキュリティ対策が施されたクラウドサービスの利用は、デバイス単体でセキュリティ対策をするより安全なシステムを構築が可能である。

3. API連携による組み込みシステムの構成

組み込みシステムにAI/IoT機能を実装するには、API連携によるウェブサービスの利用が重要である。AI/IoT機能を実行するウェブサービスにアクセスすることで、組み込みデバイスのみで実現が困難で高度な処理が可能となる。ここでは、API連携による組み込みシステムの基本構成と実装が必要な機能について述べる。

3. 1 システム構成

図1にAPI連携による組み込みシステムの構成を示す。システムは、組み込みデバイス、プロキシ、API、AI/IoT機能を実行するウェブサービスより構成される。組み込みデバイス

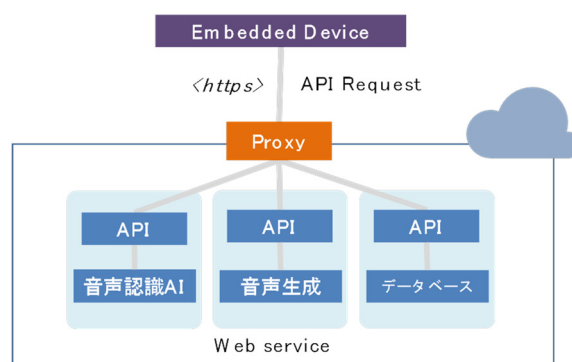


図1 API連携による組み込みシステムの構成（ウェブサービスを自由に組み合わせ利用できる。）

は、公開されたAPIを利用することで、音声認識AI、音声生成、およびデータベース機能を実装できる。クラウド上の各々のサービスは独立したウェブサービスとして実行されており、ウェブサービス同士をAPI連携して組み合わせることで、一つのアプリケーションを構築できる。セキュリティ強化のため、外部からのAPIアクセスはプロキシ経由で行われ、通信路はHTTPSで暗号化されている。

3. 2 システムの動作

システムの動作フローを図2に示す。まず、組み込みデバイスがユーザ入力やセンサデータの取得を行う。次に、取得したデータを基に、プロキシサーバ経由で必要なAPIにリクエストを送信する。プロキシサーバはAPIリクエストを検証し、該当するAPIに転送を行う。ウェブサービスはリクエストに基づいて処理を行い、その結果をプロキシサーバ経由で組み込みデバイスに返す。組み込みデバイスは返された結果に基づいてアクチュエータの制御やユーザに情報の表示を行う。

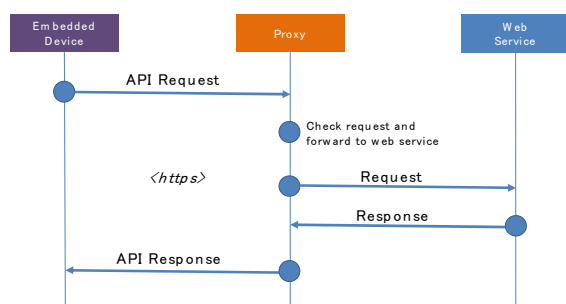


図2 システムの動作（組み込みデバイスのAPIリクエストとプロキシ経由のウェブサービスへのアクセス）

3. 3 組み込みデバイス

組み込みデバイスは、ユーザとのインタフェースとなり、センサデータの収集や音声の入出力等を行う。具体的には、音声やタッチパネル、ボタンなどのユーザ入力を受け付け、各種センサのデータを収集する。APIリクエストとして、センサで取得したデータやユーザ入力に基づいて、各APIに対してリクエストを送信する。APIから返された結果を解析し、それに応じた処理を実行する。一般にAPIで送受信されるデータは、JSONやXMLなどの構造化された軽量のデータである。組み込みデバイス

はAPIリクエストの発行とウェブサービスからの結果を受けるだけで良いため、通信機能を搭載したマイコンであれば特に高性能なプロセッサや大容量のメモリを必要としない。

3. 4 プロキシサーバ

プロキシサーバは、組み込みデバイスとウェブサービス間の通信を中継し、APIアクセスのセキュリティ強化を行う。アクセス制御として、認証されたデバイスからのリクエストのみを許可し、不正アクセスを防止する。また、通信路をHTTPSにより暗号化し、盗聴や改ざんを防ぐ。さらに、複数のサービスへのリクエストを分散することで、システム全体の負荷を軽減できる。頻繁にアクセスされるデータをキャッシュすることで応答時間の短縮を図ることも可能である。

3. 5 APIとウェブサービス

APIは、組み込みデバイスとウェブサービス間の通信インタフェースである。APIリクエストはHTTPS経由で行われ、HTTPメソッド（GET、POST、PUT、DELETE）によりウェブサービスにリクエストを送ることができる。例えば、音声認識AIのAPIは、デバイスからの音声データを受け付け、文字起こししたテキストデータを結果として回答する。APIの利用により、組み込みデバイス単体で実装が困難な生成AIの実装やデータベースの処理が容易に実行できる。また、ウェブサービスは組み込みデバイスに依存がなく、各々独立して開発ができる利点がある。

4. API連携による組み込みシステムのAI/IoT機能の実装

前章の基本構成に基づいて組み込みシステムにAI/IoT機能を実装した。システム構成を図3に示す。本システムは、昨年度構築したデータ分析基盤を拡張したものである^[1]。ウェブサービスとして、音声生成、音声認識、データベース機能を提供し、それぞれ組み込みデバイスからAPIにより利用可能である。次に、本システムを構成する組み込みデバイス、プロキシの実装、デバイスの認証、APIの実装について述べる。

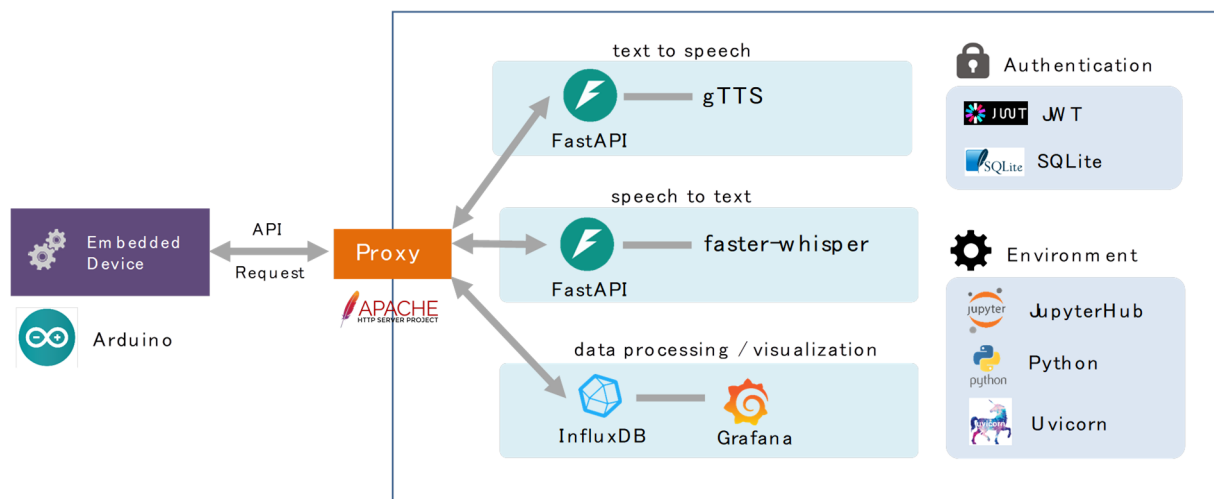


図3 オープンソースソフトウェアに基づくAPI連携による組み込みシステムの実装（組み込みデバイスに、音声生成、音声認識、データベース機能を提供し、プロキシ経由で各APIの利用ができる。）

4. 1 組み込みデバイスの実装

組み込みデバイスとして、無線Wi-Fi機能を搭載したマイコン（Espressif社、ESP32-WROOM-32D）を用いた。表1にデバイスの仕様、図4にデバイスの外観を示す。音声入力にMEMSマイク、音声出力にD級アンプ、スピーカーとして、ユーザによる音声入出力に対応した。また、ユーザ入力用のスイッチ、表示用の液晶ディスプレイを備えている。ソフトウェア開発にはArduinoを用いた。

表1 組み込みデバイスの仕様

品名	規格	備考
マイコン	Espressif, ESP-WROOM-32D	IEEE802.11/b/g/n
MEMSマイク	Sipeed, MSM261S4030H	音声入力, I ² S
D級アンプ	Analog Devices, MAX98357A	音声出力, I ² S
スピーカー	3W, 8Ω	音声出力
空気質センサ	Bosch, BME680	センサ, I ² C
照度センサ	Rohm, BH1750	センサ, I ² C
液晶ディスプレイ	2.4inch, 320x240	表示, SPI
スイッチ	タクトスイッチ(黒) 3個	ユーザ入力

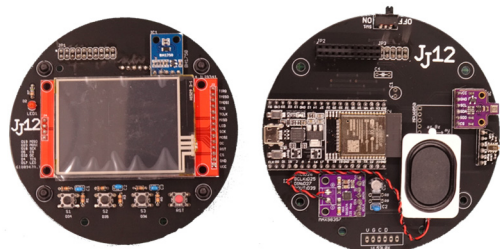


図4 組み込みデバイスの外観（液晶ディスプレイボードとプロセッサ・センサボードに分かれており連結可能）

4. 2 プロキシの実装

プロキシにはWebサーバApacheを用いた^[2]。プロキシはリバースプロキシとして動作し、クライアントからのリクエストを内部のウェブサーバに中継を行う。リバースプロキシは表2のルールに従って、デバイスのAPIリクエストを各ウェブサービスに転送する設定を行った。

表2 プロキシサーバとウェブサービス

サービス名	プロキシサーバ(エンドポイント)	ウェブサーバ
音声生成	https://<hostname>/tts	http://localhost:9000/tts
音声認識AI	https://<hostname>/stt	http://localhost:9001/stt
データベース	https://<hostname>/db	http://localhost:8086/

ここで、音声認識AIを利用する場合には、デバイスからAPIのエンドポイントである「https://<hostname>/stt」に対してリクエストを送る。リクエストは内部のウェブサーバ「http://localhost:9001/stt」に転送され、音声認識された文字列がデバイスに返される。プロキシの使用により、外部からはSSL/TLSで暗号化されたHTTPS経由のアクセスとなるため、マイコンにプロキシサーバのサーバ証明書の実装が必要になる。ここでは、あらかじめブラウザでルート証明書(root.crt)をダウンロードし、プログラムにヘッダファイルとして埋め込んで使用した。この際、証明書の有効期間を確認しておく。終了時刻を超えると認証ができなくなるので注意が必要である。

4. 3 デバイスの認証

外部にAPIが公開されているため、不正なデバイスからAPIが利用される可能性がある。そこで、特定のデバイスからのみAPIを利用できるようデバイスの認証を導入した。デバイスの認証にはBearer認証を用いた。Bearer認証はアクセストークンを利用するRFC7519で標準化された認証方式である。アクセストークンは、次のPythonコードを用いてデバイス名と秘密鍵を設定して生成した。ここで、生成されたトークンの例を示す。

```
from jose import jwt
device_id = "<デバイス名>"
SECRET_KEY = "<opensslコマンドで生成>"
token = jwt.encode({"device_id": device_id},
                    SECRET_KEY, algorithm="HS256")
```

(生成されたトークンの例)

```
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJkZXZpY2VfaWQiOiJpb3QtMDEifQ.80K5zf
GXb7BeLrRg7JmEeP9RBul8hu5Ra2BsWi-Ga4
```

実際には、デバイス名、秘密鍵（opensslコマンドで生成）、およびトークンの有効期限をSQLiteデータベースに保存しておき、データベースから読み出した値をデバイスの認証に使用した。認証されたデバイスは、APIリクエストのAuthorizationヘッダにアクセストークンを付加して送信する。ウェブサービス側で、デバイス名、秘密鍵の認証、有効期限の確認が行われ、問題がなければリクエストを受け付け、実行結果をデバイスに返す。アクセストークンは、HTTPS経由で送信されるため、盗聴や改竄のセキュリティリスクは少ない。

4. 4 APIウェブサービスの実行

ウェブサービスとして、音声生成、音声認識、およびデータベースサーバを構築した。音声生成サーバは、PythonライブラリgTTSを用いて、与えられたテキストから音声を作成する機能を提供する。gTTSはGoogle translateを使用しているため生成回数等に制限があるが、多くの言語に対応している^[3]。音声認識サーバは、Faster-whisperを用いてデバイスで録音された音声を認識して文字おこしする機能を提

供する。Faster-whisperは、OpenAIのWhisperモデルをCTranslate2という高速推論エンジンを用いて再構築したものであり認識に係る時間が速いのが特徴である^[4]。データベースサーバにはInfluxdbを用いた^[5]。データベースの作成・削除、データベースへのデータの読み込み書き込みを行う機能を提供する。

音声生成、音声認識サービスは、Pythonスクリプトで記述し、FastAPIライブラリを使用してAPIを作成した。FastAPIは、PythonでAPIを構築するためのWebフレームワークである^[6]。FastAPIを用いると、SwaggerUIによりAPIドキュメントを自動で生成できる。開発したPythonスクリプトをウェブサービスとして起動するために、Python用ASGI WebサーバであるUvicornを用いた^[7]。ここで、音声生成サービスのPythonスクリプトがmain.pyのとき、次のようにホスト名、ポート番号を指定してウェブサーバを実行できる。

```
$ uvicorn main:app --host <ip address> --port
<port number>
```

これにより、指定のIPアドレス、ポート番号でウェブサービスが起動する。SwaggerUIにより、ブラウザで「<http://<ip address>:<port number>/docs>」にアクセスすると、APIドキュメントを閲覧でき、APIリクエストのテストができる。

データベースInfluxdbサービスは、アクセストークンを生成できるUIがあるため、事前にトークンを生成して使用した。Influxdbに記録したデータの可視化にはGrafanaを用いた。アクセストークンは、次のコマンドで生成できる。

```
$ influx auth create --org <my-org> --all-
access
```

ここでは、グループOrganizationに所属するユーザに、すべてのアクセス権を与えるトークンを生成している。Influxdbではグループごとに、データベースへの読み込み書き込み、ダッシュボード表示の権限の設定ができる。デバイスは、ここで生成されたアクセストークンを認証に用いて、クラウド上のデータベースへのデータの読み書き込みを行った。

5 API連携による組み込みシステムの動作検証

構築したAPI連携による組み込みシステムへのAI/IoT実装について動作検証を行った。組み込みデバイスに、音声認識API、音声生成APIを利用して、音声によるデバイスの操作を行った。同時に、並列でセンサデータをデータベースAPIにより書き込みを行った。

5.1 音声によるデバイスの操作

API連携によるAI/IoT機能の実装を検証するために、音声によるデバイスの操作を行った。デバイス操作のシナリオを図5のように作成して実行した。シナリオは次のとおりである。デバイスに「LEDをONにしてください。」と音声入力を行う。音声認識サービスに対してAPIリクエストで音声データ(WAV形式)を送り、この音声を文字おこした結果をJSONテキストでデバイスに返した。デバイスは、認識した文字列を解析し、「LED」、「ON」の両方が含まれていれば、「デバイスに実装したLEDを点灯」した。さらに、デバイスからテキスト「LEDをONしました。」を音声生成サービスにAPIリクエストで送り、テキストが音声に変換され、音声データ(MP3形式)としてデバイスが受け取り、生成された音声をデバイスのスピーカーで再生した。ここで、音声認識サービスへのAPIリクエストは、次のようにHTTPリクエストのPOSTメソッドで送った。

```
POST /stt HTTP/1.1
Host: <hostname>
Content-Type: multipart/form-data
<Body> WAV形式データ
```

ここでは、URI (/stt)、ホスト名を指定して、音声データを添付して送信した。これらの音声処理と並列して、別プロセッサで1分毎にセンシングをし、APIリクエストでデータベースにセンサデータの書き込みを行った。20台のデバイスで同時にシナリオを実行したところ、音声認識、音声生成に数秒の遅延が発生することがあった。遅延の原因は、音声認識AI・音声生成の処理時間がかかること以外に、録音音声(WAV形式)のサイズが大きいため送信に時間を要していたことが考えられる。

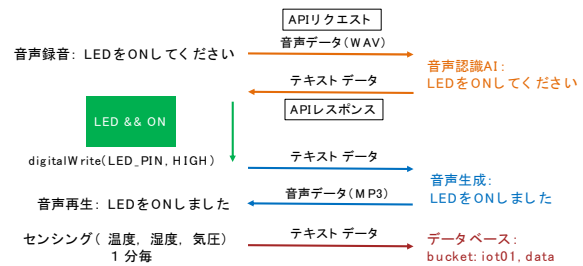


図5 音声によるデバイス操作のシナリオ

6. まとめ

本稿では、API連携により組み込みシステムにAI/IoT機能を実装する方法について述べた。APIを利用することで、組み込みシステムでは実装が困難な音声認識AI、音声生成、データベース機能を追加することができた。APIの利用に際しては、ユーザにアクセストークンを発行し、デバイス認証としてBearer認証を行う仕組みを導入した。さらに、通信路をHTTPS経由で暗号化し、プロキシサーバ経由でアクセス制限することでセキュリティ強化を図った。

API連携により、音声認識、音声生成機能を自由に組み合わせ使用できることで、新たなユーザインタフェースの提供が可能となった。また、センサデータについて、データベースにAPI経由でアクセスすることで、サーバ上のストレージにデータの記録・更新・削除、可視化が容易にできるようになり、デバイス自体に必要なメモリ領域を削減できた。さらに、組み込みシステムに最適なAI/IoTサービスを構築することで、リソース制限のある組み込みシステムに付加価値を与えることが可能となった。今後、AIサービスとして対話型AIの実装やサーバリソース拡張を図る予定である。

参考文献

- [1] 辻明典, “AIの活用を目的としたIoTデータ分析基盤の構築と運用”, 徳島大学技術支援部技術報告, No. 7, pp. 13-17, 2024.
- [2] Apache, <https://httpd.apache.org/>
- [3] gTTS, <https://pypi.org/project/gTTS/>
- [4] Faster-whisper, <https://github.com/SYSTRAN/faster-whisper>
- [5] InfluxDB, <https://www.influxdata.com/>
- [6] FastAPI, <https://fastapi.tiangolo.com/>
- [7] Uvicorn, <https://www.uvicorn.org/>