televic 制御プロトコル案内書

rev.2024-12-27　株式会社オーディオブレインズ

目次（ハードウェア/ソフトウェア別 参照先リスト）

ハードウェア	ソフトウェア/バージョン	プロトコル	参照先
Plixus MME Plixus AE-R	CoCon（Plixus version 6.x）	API	CoCon API
		カメラコントロール	Plixus 6.x カメラコントロール
	Confero（Plixus version 7.x）	API	Confero API
		カメラコントロール	Camera Protocols for Confero & D-Cerno AE
Confidea WAP G4 *スタンドアロン動作	Confero	API	Confero API
		カメラコントロール	Camera Protocols for Confero & D-Cerno AE
D-Cerno AE	-	API	D-Cerno Customer REST API
		カメラコントロール	Camera Protocols for Confero & D-Cerno AE
D-Cerno CUR	-	API	D-Cerno CUR API
		カメラコントロール	D-Cerno CUR カメラコントロール

※上記にないプロダクトの資料は別途お問い合わせください

【補足】カメラコントロールプロトコルについて

televicの現行の会議システムはAPIとは別にカメラコントロール用のプロトコルを持ち、会議マイクオン/オフのトリガーステータスをUDP（もしくはTCP）で制御器に送信することが可能です。

【補足】：CoCon（Plixus version 6.x）、 Confero（Plixus version 7.x）について

Plixusは、要件に応じて専用ソフトウェア(CoCon)互換バージョンか Web GUIコントロール(Confero)バージョンのいずれかを選択します。詳細は別途お問い合わせください。

CoCon API（Plixus 6.x/CoCon Server）

プロトコル資料（バージョン別、英文、外部リンク）

- CoCon API Description:
  - CoCon API 6.9 - EN
  - CoCon API 6.8 - EN
  - CoCon API 6.7 - EN
  - CoCon API 6.6 - EN
  - CoCon API 6.5 - EN
  - CoCon API 6.4 - EN
  - CoCon API 6.3 - EN
  - CoCon API 6.2 - EN
  - CoCon API 6.1 - EN
  - CoCon API 6.0 - EN

補足：REST/JSON

- HTTP API(ポート8890)、REST/JSONフォーマット、Long polling方式です。
  Plixusエンジンがサーバー、制御器がクライアントです。
  - ”Server to client”が Connect するとプッシュされるコマンドです。
  - ”Client to server”が制御器から送信するコマンドです。

補足：API module compatibility

- プロトコル資料の「API module compatibility」リストのチェックボックスを確認してください。
  「Plixus Core」にチェックのある項目はPlixusエンジンをターゲットに制御が可能な項目で、「CoCon for Plixus」 にチェックのある項目はCoCon Server（サーバーPC）をターゲットに制御が可能な項目です。
- 例：SetState（マイクオンオフ）は「Plixus Core」に該当しPlxusエンジンを直接制御可能です

API Module	Cocon for Plixus Core	Plixus Core
4.3.2.12 GetAllSeats	☑	☑
4.3.2.13 GetBooths	☑	☑
4.3.2.14 EditSeat	☑	☑
4.3.2.15 GetAllUnits	☑	☑
4.3.2.16 SetOperatingMode	☑	☑
4.3.2.17 GetOperatingMode	☑	☑
4.3.2.18 ClearAllMeetingAndDelegateData	☑	☐
4.3.2.19 SetSeatPriority	☑	☐
4.3.3 Microphone		
4.3.3.1 SetState	☑	☑
4.3.3.2 SetMicrophoneMode	☑	☑

補足：Connect/Notification

- 「Server to Client」に該当するコマンドはPlixus/CoConのイベントを制御クライアントへステータスポーリングするコマンドです。
    - ステータスポーリングを取得するために Connect（ロングポーリングのスタート）を送る必要があります。
    - ステータスポーリングをひとつ取得するため Notification（通知をもらうコマンド）をひとつ送る必要があります。
    - 「Server to Client」に該当するコマンドはPlixus/CoConのイベント制御もしくはステータスのGETコマンドで、Connect および Notification を必要としません。
    - 「Server to Client」「Client to Server」のセッションは、異なるスレッドを使用した独立したものである必要があります。単一のスレッドによる処理だと、ステータスポーリングを待つセッションにイベント制御コマンドが送信され、そのコマンドが処理されません。
- Subscribe, Unsubscribe により、必要のないイベントカテゴリの通知を受信しないようにフィルタリングすることが可能です。（デフォルトでは全てのモジュールからの通知が受信されます。）
  制御器が不要なパケットの受信や処理に時間を費やさないよう、このフィルタリングを利用した効率的なプログラムを実施してください。

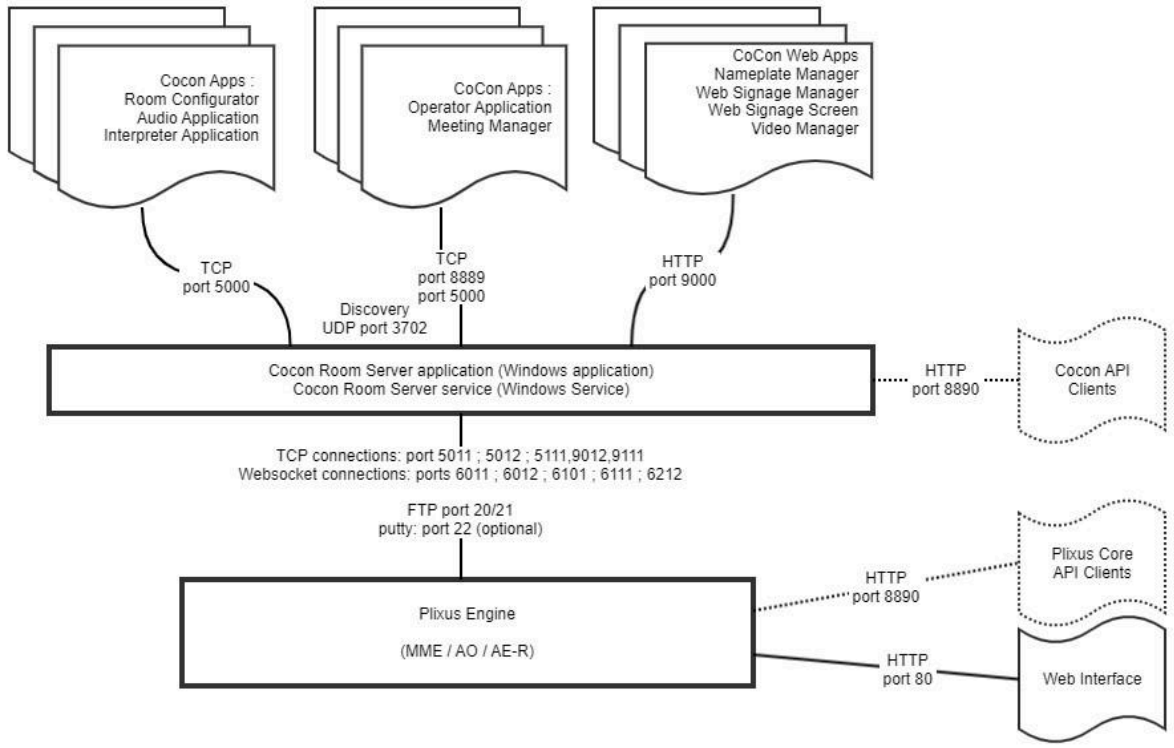
補足：CoCon API Test Tool

- CoCon API Test Tool は、CoCon APIのターミナルソフトウェアです。
  CoConソフトウェアセット（CoCon Server/CoConクライアントアプリケーション）の一部で、下記URLからダウンロードが可能です。
  https://www.televic.com/en/conference/support/software-updates
    - [インストール先ディレクトリ]￥CoCon API Test Tool￥CoCon API Test Tool.exe　を起動するとCoCon API Test Toolのウィンドウが現れます。
    - Communication Logには、テストツール⇔主装置 間のコマンド送受が履歴されます。
    - Base URL に、http://[制御対象のIPアドレス]:8890/CoCon/　を入力します。（制御対象はPlixusエンジンもしくはCoConサーバー（PC)です）
    - “Client to Server”コマンドのテスト：URI Commandに「～CoCon/」以下のコマンドを入力しRequestをクリックするとコマンドが送信されます。例えば SetState を使ってSeat1をオンする場合、
      Microphone/SetState/?State=On&SeatNr=1　を入力しRequestボタンをクリックします。
    - “Server to Client”コマンドのテスト（Connect/Notification セッションのテスト）：Connectボタンをクリックすると Connect が制御のターゲットに送信されます。Connect が成功しAPI制御が成立するとIDが返却され、ステータス取得が可能になります。ステータスをひとつ取得するたびに、Test Toolから自動的に Notification がひとつ送信されます。

ベストプラクティス

- 不必要なHTTP接続の追加を避けて、既存の接続を維持するようにしてください。
- `Connect` で返されたIDを用いて、 `Notification` リクエストをできるだけすぐに行ってください。このリクエストは以下のような結果になります：
    - IDが無効な場合（`HTTP 400`）:直ちに新しい接続を開く必要があります。（主装置が再起動されたケースなど）
    - メッセージが送信されない場合（`HTTP 408`）:30秒後にタイムアウトとなるため、速やかに新しい通知リクエストを送信します。
    - タイムアウト前にメッセージが送信される場合:メッセージを送信・処理した後、速やかに新しい通知リクエストを開きます。
- `Notification` メッセージを処理する専用のスレッドを用意し、処理の漏れがないようにします。また、制御器のロジックやUI処理は `Notification` を処理するスレッドとは別のスレッドで処理し、迅速な通信を確保してください。

参考：占有ポート



Confero API（Plixus 7.x/Confidea G4）

プロトコル資料（バージョン別、英文、外部リンク）

- Confero API Documentation
  - CRP 7.11 - Plixus | G4
  - CRP 7.10 - EN
  - CRP 7.9 - EN
  - CRP 7.8 - EN
  - CRP 7.4 - EN
  - CRP 7.3 - EN
  - CRP 7.2 - EN

補足：コマンド形式

- 伝送には HTTP/HTTPS を用います。Plixusエンジン/Confidea WAP G4が サーバー、制御器がクライアントとなります。
- REST/JSONフォーマット、Long polling方式を採用しております。
- HTTPS 暗号化通信を利用するにはサーバーに証明書をアップロードする必要があります。証明書の発行については Swagger UI の説明を参照してください。
  - 証明書がない場合、通信はHTTP で、ポート番号は `9080` です。
  - 証明書がある場合、通信はHTTPSで、ポート番号は `9443` です。

補足：Swagger UI

- Confero API仕様は「Swagger UI」にて記述されています（英文）
  Google Chrome等のWebブラウザに以下のアドレスを入力して Swagger UIにアクセスしてください。
  http://{サーバーIP}/openapi/
- Swagger UI のAPIターミナル機能を使用して制御の確認が可能です。
  Swagger UIの「Authorize」ボタンをクリックするとトークン入力画面が現れるので、トークンを入力して Authorize します。
  これによりSwagger UIから生成される各種HTTPコマンドのヘッダーにトークンがセットされます。

補足：APIアクセスキー/トークン

- Confero API アクセスには HTTPヘッダーにセキュリティトークン（アクセスキー）をセットする必要があります。
  （トークン様式一例： `08b64f33-01f2-400a-865b-af552da01c94` ）
- トークンは Plixusエンジン/WAP G4 の Web GUI 設定にて発行します。詳細は Swagger UI の説明を参照してください。

補足：Notificationについて

- `Notification` はデバイスのイベントを制御クライアントへステータスポーリングするコマンドです。ステータスポーリングをひとつ取得するために、`Notification` （通知をもらうコマンド）を送る必要があります。

Plixus/CoConのイベント制御、もしくはステータスのGETコマンドは `Notification` を必要としません。

- `include-filter` オプションにより必要なEventのみ取得することが可能です。
  Eventはmoduleによって、任意に選択可能です。
  制御器が不要なパケットの受信や処理に時間を費やさないよう、このフィルタリングを利用した効率的なプログラムを実施してください。
  - `Audio`：システム全体の一般的なオーディオ設定を構成するモジュール
  - `Discussion`：マイクの状態変更や会議設定の変更・取得を行うためのオペレータ機能を持つモジュール
  - `Meeting`：会議のコントロール用モジュール
  - `Recording`：録音をサポートするシステムで録音を制御するモジュール
  - `Room`：部屋の設定にアクセスするためのモジュール
  - `Wireless`：ワイヤレスシステムの無線機能の取得と制御を行うモジュール
  - `WirelessCoupling`：有線システムとワイヤレスシステムを連携させるためのモジュール

- `Notification` のイベントストア内の各イベントには一意で連続的な識別子（`id`）が割り当てられており、イベントが発生した順序を維持します。
  `minimum-id` オプションを利用して、クライアントが最後に受信したイベントIDの次のIDを指定することで、未処理の新しいイベントのみを確実に取得することが可能です。

  具体例：
  クライアントアプリケーションが `id=100` までのイベントを処理したとします。この場合、`minimum-id` の使用方法は以下の通りです。

  `minimum-id` **を指定しないリクエスト：**
  `GET /api/notification/events?include-filter=Audio,Discussion`
  結果： `minimum-id` を指定しない場合、APIは最新のイベントを返すか、新しいイベントを待機しますが、接続再確立前に発生したイベントを見逃す可能性があります。

  `minimum-id` **を指定したリクエスト：**
  `GET /api/notification/events?minimum-id=101&include-filter=Audio,Discussion`
  結果： APIは `id >= 101` のイベントを順に返します。これにより、最後に処理したイベント（`id = 100`）以降のすべてのイベントを確実に受け取ることができます。

ベストプラクティス

- 不必要なHTTP接続の追加を避け、既存の接続を維持するようにしてください。
- `Notification` メッセージを処理する専用のスレッドを用意し、処理の漏れがないようにします。また、制御器のロジックやUI処理は `Notification` を処理するスレッドとは別のスレッドで処理し、迅速な通信を確保します。

D-Cerno Customer REST API

プロトコル資料（バージョン別、英文、外部リンク）

- CRP 1.3.0 - EN

補足：コマンド形式

- 伝送には HTTP/HTTPS を用います。D-Cerno AEがサーバー、制御器がクライアントとなります。
- REST/JSONフォーマット、Long polling方式を採用しております。
- HTTPS 暗号化通信を利用するにはサーバーに証明書をアップロードする必要があります。証明書の発行については Swagger UI の説明を参照してください。
    - 証明書がない場合、通信はHTTP で、ポート番号は `9080` です。
    - 証明書がある場合、通信はHTTPSで、ポート番号は `9443` です。

補足：Swagger UI

- API仕様は「Swagger UI」にて記述されています（英文）
  Google Chrome等のWebブラウザに以下のアドレスを入力して Swagger UIにアクセスしてください。
  http://{サーバーIP}/openapi/
- Swagger UI のAPIターミナル機能を使用して制御の確認が可能です。
  Swagger UIの「Authorize」ボタンをクリックするとトークン入力画面が現れるので、トークンを入力して Authorize します。
  これによりSwagger UIから生成される各種HTTPコマンドのヘッダーにトークンがセットされます。

補足：APIアクセスキー/トークン

- Confero API アクセスには HTTPヘッダーにセキュリティトークン（アクセスキー）をセットする必要があります。
  （トークン様式一例： `08b64f33-01f2-400a-865b-af552da01c94` ）
- トークンは D-Cerno AE の WEB GUI 設定にて発行します。詳細はSwagger UI の説明を参照してください。

補足：Notificationについて

- `Notification` はデバイスのイベントを制御クライアントへステータスポーリングするコマンドです。ステータスポーリングをひとつ取得するために、`Notification` （通知をもらうコマンド）を送る必要があります。
  Plixus/CoConのイベント制御、もしくはステータスのGETコマンドは `Notification` を必要としません。
- Notificationは `include-filter` オプションにより必要なEventのみ取得することが可能です。Eventはmoduleによって、任意に選択可能です。
  制御器が不要なパケットの受信や処理に時間を費やさないよう、このフィルタリングを利用した効率的なプログラムを実施してください。

    - `Audio`: 一般的なシステム全体のオーディオ設定を行うモジュール。
    - `Device`: デバイス関連情報を取得するモジュール
    - `Discussion`: マイクの状態を変更したり取得したり、会議設定を変更するためのオペレータ機能を持つモジュール
    - `Notification`: 会議コントローラーから非同期イベントを取得するモジュール
    - `Room`: 会議室設定にアクセスするモジュール
    - `System`: 座席の並び替えを管理するモジュール

- `Notification` のイベントストア内の各イベントには一意で連続的な識別子（`id`）が割り当てられており、イベントが発生した順序を維持します。
  `minimum-id` オプションを利用して、クライアントが最後に受信したイベントIDの次のIDを指定することで、未処理の新しいイベントのみを確実に取得することが可能です。

  具体例：
  クライアントアプリケーションが `id=100` までのイベントを処理したとします。この場合、`minimum-id` の使用方法は以下の通りです。

  `minimum-id` **を指定しないリクエスト：**
  `GET /api/notification/events?include-filter=Audio,Discussion`
  結果： `minimum-id` を指定しない場合、APIは最新のイベントを返すか、新しいイベントを待機しますが、接続再確立前に発生したイベントを見逃す可能性があります。

  `minimum-id` **を指定したリクエスト：**
  `GET /api/notification/events?minimum-id=101&include-filter=Audio,Discussion`
  結果： APIは `id >= 101` のイベントを順に返します。これにより、最後に処理したイベント（`id = 100`）以降のすべてのイベントを確実に受け取ることができます。

ベストプラクティス

- 不必要なHTTP接続の追加を避け、既存の接続を維持するようにしてください。
- `Notification` メッセージを処理する専用のスレッドを用意し、処理の漏れがないようにします。また、制御器のロジックやUI処理は `Notification` を処理するスレッドとは別のスレッドで処理し、迅速な通信を確保してください。

Camera Protocols for Confero & D-Cerno AE

プロトコル資料（英文、外部リンク）

- Camera Protocols for Confero & D-Cerno AE – EN

Plixus 6.x カメラコントロール

プロトコル資料（英文、外部リンク）

- Camera Protocols for Confero & D-Cerno AE – EN

  補足： Plixus 6.x で利用可能なプロトコルは「TLVCAM1」「TLVCAM2」で、プロトコル仕様はConferoと同じのため Conferoの資料を参照してください。また「T-CAM」プロトコルは利用できません。

D-Cerno CUR API

プロトコル資料（英文、外部リンク）

- https://audiobrains.com/data/televic/others/d-cerno_cur_api.pdf

D-Cerno CUR カメラコントロール

プロトコル資料（英文、外部リンク）

- 製品マニュアルの「カメラコントロール」の章を参照してください
  https://audiobrains.com/data/televic/manual/d-cerno.pdf

AUDIO BRAINS

株式会社オーディオブレインズ
〒216-0033 神奈川県川崎市宮前区宮崎649-3 電話：044-888-6761
（受付時間：10:00～18:00 土日祝日・弊社休業日を除く）
https://audiobrains.com/