分野間連携データ基盤

詳細設計書

(提供者編)

第4.0版



変更来歴

|  |  |  |  |
| --- | --- | --- | --- |
| # | 版数 | 発行年月日 | 変更内容 |
| 1 | 第1.0版 | 2020/07/7 | 新規作成 |
| 2 | - | 2020/11/30 | 来歴管理I/F(旧 原本性保証I/F)に関する記述の修正  契約確認処理の関連の記述の修正  認証認可I/Fの追加 |
| 3 | - | 2021/01/29 | 来歴管理I/Fで通信先をコンフィグから取得するように変更し、エラーコード09003Eを削除。 |
| 4 | 第2.0版 | 2021/4/06 | ディレクトリ／ファイル一覧を「詳細設計書\_別紙14\_提供者コネクタ\_ディレクトリ構成」に移動 |
| 5 | 第3.0版 | 2021/11/19 | * NGSI来歴対応を反映。認証認可I/Fの機能追加。契約管理の機能追加。 * 章構成などをはじめとした詳細設計書レイアウトの調整 * 基本設計書との文言ゆれを調整 |
| 6 | 第4.0版 | 2022/6/30 | * 認証認可に関する仕様調整の適用 * 章構成などをはじめとした詳細設計書レイアウトの調整 * 詳細設計書間での文言ゆれを調整 * 認証認可I/Fを認可I/Fに変更 |

目次

[変更来歴 １-２](#_Toc108540587)

[目次 １-１](#_Toc108540588)

[１． はじめに 1](#_Toc108540589)

[１．１ ドキュメント体系 1](#_Toc108540590)

[１．２ 適用範囲 1](#_Toc108540591)

[２． コンテナ構成 2](#_Toc108540592)

[３． 機能一覧 3](#_Toc108540593)

[４． コネクタメインサブシステム 4](#_Toc108540594)

[４．１ 内部仕様 4](#_Toc108540595)

[４．１．１ システム構成 4](#_Toc108540596)

[４．１．２ 公開インタフェース 7](#_Toc108540597)

[４．１．３ 内部データ一覧 7](#_Toc108540598)

[４．１．４ コンフィグ定義 7](#_Toc108540599)

[４．２ 機能詳細 8](#_Toc108540600)

[４．２．１ コネクタメイン機能 処理一覧 8](#_Toc108540601)

[４．２．２ 処理フロー 9](#_Toc108540602)

[４．２．３ 出力ログ 15](#_Toc108540603)

[５． 認可I/Fサブシステム 16](#_Toc108540604)

[５．１ 内部仕様 16](#_Toc108540605)

[５．１．１ システム構成 16](#_Toc108540606)

[５．１．２ 公開インタフェース 17](#_Toc108540607)

[５．１．３ コンフィグ定義 17](#_Toc108540608)

[５．２ 機能詳細 18](#_Toc108540609)

[５．２．１ 認可I/F機能 処理一覧 18](#_Toc108540610)

[５．２．２ 処理フロー 19](#_Toc108540611)

[５．２．３ 出力ログ 20](#_Toc108540612)

[６． カタログ検索I/Fサブシステム 21](#_Toc108540613)

[６．１ 内部仕様 21](#_Toc108540614)

[６．１．１ システム構成 21](#_Toc108540615)

[６．１．２ 公開インタフェース 21](#_Toc108540616)

[６．１．３ 内部データ一覧 21](#_Toc108540617)

[６．２ 機能詳細 22](#_Toc108540618)

[６．２．１ カタログ検索I/F機能 処理一覧 22](#_Toc108540619)

[６．２．２ 処理フロー 22](#_Toc108540620)

[６．２．３ 出力ログ 22](#_Toc108540621)

[７． データ交換I/Fサブシステム 24](#_Toc108540622)

[７．１ 内部仕様 24](#_Toc108540623)

[７．１．１ システム構成 24](#_Toc108540624)

[７．１．２ 公開インタフェース 24](#_Toc108540625)

[７．１．３ 内部データ一覧 24](#_Toc108540626)

[７．２ 機能詳細 25](#_Toc108540627)

[７．２．１ データ交換I/F機能 処理一覧 25](#_Toc108540628)

[７．２．２ 処理フロー 25](#_Toc108540629)

[７．２．３ 出力ログ 26](#_Toc108540630)

[８． 来歴管理I/Fサブシステム 27](#_Toc108540631)

[８．１ 内部仕様 27](#_Toc108540632)

[８．１．１ システム構成 27](#_Toc108540633)

[８．１．２ 公開インタフェース 27](#_Toc108540634)

[８．１．３ 内部データ一覧 27](#_Toc108540635)

[８．２ 機能詳細 28](#_Toc108540636)

[８．２．１ 来歴管理I/F機能 処理一覧 28](#_Toc108540637)

[８．２．２ 処理フロー 29](#_Toc108540638)

[８．２．３ 出力ログ 30](#_Toc108540639)

# はじめに

## ドキュメント体系

本書はシステムの内部設計を記載する。構成ファイル一覧を表 １．１‑１に記載する。

表 １．１‑１詳細設計書構成ファイル一覧

|  |  |
| --- | --- |
| # | ドキュメント名 |
|  | 詳細設計書\_共通編.docx |
|  | 詳細設計書\_提供者編.docx |
|  | 詳細設計書\_利用者編.docx |
|  | 基本設計書\_別紙2\_コンフィグパラメータ一覧.xlsx |
|  | 詳細設計書\_別紙1\_メッセージ一覧.xlsx |
|  | 詳細設計書\_別紙7\_提供者\_コネクタメイン.html |
|  | 詳細設計書\_別紙8\_提供者\_カタログ検索IF.html |
|  | 詳細設計書\_別紙9\_提供者\_データ交換IF(CADDE).html |
|  | 詳細設計書\_別紙10\_提供者\_認可I/F.html |
|  | 詳細設計書\_別紙11\_提供者\_来歴管理I/F.html |
|  | 詳細設計書\_別紙13\_提供者コネクタ\_ディレクトリ構成.xlsx |

## 適用範囲

本書は2022年3月版の詳細設計を対象とする。

# コンテナ構成

提供者側コネクタは下図の構成で構築する。

コネクタメイン(提供者側)

データ提供I/F

(HTTPS)

コンテナ

コンテナ内サブシステム

HTTPS通信

凡例：

カタログ検索 I/F

カタログ検索I/F

(HTTPS CKAN))

データ交換I/F

データ提供I/F

(FTP)

データ提供I/F

(HTTPS NGSI)

他社開発コンテナ内サブシステム

共通編記載サブシステム

来歴管理I/F

認証認可I/F

図 ２．１‑１ コンテナ構成

# 機能一覧

機能一覧を記載する。

表 ３‑１詳細設計書機能一覧

| # | 機能 | 概要 | 備考 |
| --- | --- | --- | --- |
|  | コネクタメイン | * データ連携のシナリオに基づきコネクタ内の他機能を呼び出す | － |
|  | 認可I/F | * 認可サーバに対して、トークン確認のリクエストを送信し、利用者が正しいCADDEユーザかどうか認証する * 認可サーバ対して、認可確認のリクエストを送信し、取得対象のデータに対して、取得可能か確認する | － |
|  | カタログ検索I/F | * 利用者コネクタと通信(HTTPS)を確立し、データ提供者のCKANからカタログ詳細情報を検索し取得し、検索結果を返却する | － |
|  | データ交換I/F | * 利用者コネクタと通信(HTTPS)を確立し、データ提供者のデータ管理からデータを取得し、返却する | － |
|  | データ提供I/F（NGSI） | * データ管理(NGSIサーバ)からNGSIにてデータを取得する | － |
|  | データ提供I/F（FTP） | * データ管理(FTPサーバ)からFTPにてデータを取得する * FTPサーバのID、パスワードは、提供者コネクタ内で設定する | － |
|  | データ提供I/F（HTTPS） | * データ管理(Webサーバ)からHTTPS経由でデータを取得する * WebサーバのID、パスワードは、提供者コネクタ内で設定する | － |
|  | 来歴管理I/F | * 来歴登録されたデータ取得時、来歴管理サービスに対して、来歴登録(送信)を行う * 限定提供データ(契約有)のデータ取得時、契約管理サービスに対して、データの送信通知を行う | 来歴管理\_機能設計書(2020年度版 1.0版).pdf  [2020/09/04版]の4.来歴管理I/F APIに基づいて作成 |

# コネクタメインサブシステム

## 内部仕様

### システム構成

システム構成を記載する。

コネクタメインコンテナ

カタログ検索I/F

コンテナ

カタログ

詳細検索

認証認可I/F

コンテナ

カタログ詳細

検索要求

検索結果

1. 認可トークン取得
2. 認可トークン検証
3. 認可確認

提供者側コネクタ Dockerコンテナ群

確認結果

カタログ

検索I/F

(HTTPS CKAN)

カタログ詳細

検索要求

検索結果

凡例:

: 対象サブシステム

: その他システム

: HTTPS通信

: 関数呼び出し

図 ４．１．１‑１ カタログ検索(詳細検索)システム構成

コネクタメインコンテナ

凡例:

: 対象サブシステム

: その他システム

: 通信

データ

交換I/F

コンテナ

データ

交換

要求

データ

交換要求

提供者側コネクタ Dockerコンテナ群

データ管理

公開

CKAN

ファイルサーバ

リソース検索要求

交換実績記録用リソースID

データ取得

コンテキスト情報(JSON)

ファイル(CSV, GeoJSON, etc)

認証認可I/F

コンテナ

1. 認可トークン取得
2. 認可トークン検証
3. 認可確認

各確認結果

来歴管理I/F

コンテナ

1. 送信履歴登録
2. データ証憑通知（送信）

各識別情報

コンテキスト情報

(JSON)

ファイル

(CSV, GeoJSON, etc)

図 ４．１．１‑２ データ交換システム構成(1/2)

コネクタメインコンテナ

データ

交換I/F

コンテナ

データ交換

要求

提供者側コネクタ Dockerコンテナ群

凡例:

: 対象サブシステム

: その他システム

: 通信

: 関数呼び出し

**データ取得詳細**

データ提供

I/F(HTTPS)

リソース提供識別手段が

"file/http"の場合

データ交換

(HTTPS)

ファイル

データ提供

I/F(FTP)

リソース提供識別手段が"file/ftp"の場合

データ交換

(FTP)

ファイル

データ交換

I/F(HTTPS NGSI)

リソース提供識別手段が"api/ngsi"の場合

データ交換

(HTTPS NGSI)

コンテキスト

情報

データ管理

／ファイルサーバ

ファイル

ファイル

コンテキスト

情報

ファイル

ファイル

コンテキスト

情報

リクエスト

リクエスト

リクエスト

図 ４．１．１‑３ データ交換システム構成(2/2)

### 公開インタフェース

「詳細設計書\_別紙7\_提供者\_コネクタメイン.html」を参照すること。

### 内部データ一覧

以下の内部データを保持する。

表４．１．３-１ 内部データ一覧

|  |  |  |
| --- | --- | --- |
| # | データ名 | 概要 |
|  | url\_ authorization | 提供者コネクタ 認可I/Fの認可トークン取得、認可トークン検証、認可確認に接続するためのURL |
|  | url\_eventwithhash\_sent | 提供者コネクタ 来歴管理I/Fの送信履歴登録要求に接続するためのURL |

### コンフィグ定義

コンフィグの一覧は「基本設計書\_別紙2\_コンフィグパラメータ一覧.xlsx」を参照すること。

## 機能詳細

### コネクタメイン機能 処理一覧

表 ４．２．１‑１ インタフェース情報

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| # | 処理名 | 概要 | URL | 関数名 | メソッド |
|  | カタログ検索 （詳細検索） | 詳細CKANを利用し、カタログ情報を取得する。APIユーザが指定した条件に従いカタログ情報を取得する。 | /cadde/api/v4/catalog | search | GET |
|  | データ交換 | 契約情報の確認を行い、データ管理からファイル取得を行う。  ファイル取得後、来歴管理I/Fに送信履歴要求を行う。 | /cadde/api/v4/file | files | GET |

### 処理フロー

#### カタログ検索(詳細検索)

1. パラメータとして、クエリストリング、認証トークンを取得する。認証トークンが取得できない場合は独自Exceptionを発生させる。
2. クエリストリングをログ出力する。
3. 各コンフィグファイルから以下を取得する。コンフィグから情報が取得できない場合、独自Exceptionを発生させる。
   * ckan.json
     1. カタログサイト(詳細)アクセスURL (detail\_ckan\_url)
     2. 認可設定（authorization）
   * connector.json
     1. CADDEユーザID（提供者）(provider\_id)
4. 認証トークンがNoneでない場合、認証トークン、CADDEユーザID（提供者）、提供者コネクタID、提供者側コネクタのシークレットを引数として、認可I/Fの認可トークン取得を呼び出す。
5. 実行結果を確認し、次のいずれかの処理を行う。
   * + ステータスコードが「200」の場合は、レスポンスのヘッダ" auth\_token "に設定されている情報を認可トークンとして取得する。
     + ステータスコードが「200」以外の場合は、該当するステータスコードとメッセージを設定し、独自Exceptionを発生させる。
6. 認可トークン、CADDEユーザID（提供者）、提供者コネクタID、提供者側コネクタのシークレットを引数として認可I/Fの認可トークン検証を呼び出す。
7. 実行結果を確認し、次のいずれかの処理を行う。
   * ステータスコードが「200」の場合は、レスポンスヘッダの"consumer\_id"に設定されている情報をCADDEユーザID（利用者）として取得する。CADDEユーザID（利用者）が取得できなかった場合は、独自Exceptionを発生させる。
   * ステータスコードが「200」以外の場合は、該当するステータスコードとメッセージを設定し、独自Exceptionを発生させる。
8. 認可設定の取得値がtrueの場合、CADDEユーザID（提供者）、提供者コネクタID、提供者側コネクタのシークレット、リソースURL（本処理フローではカタログサイト(詳細)アクセスURLをリソースURLとする）、認可トークンを引数として認可I/Fの認可確認を呼び出す。
9. 実行結果を確認し、次のいずれかの処理を行う。
   * ステータスコードが「200」の場合、認可の有効性を確認できたものとし、処理を継続する。
   * ステータスコードが「200」以外の場合は、独自Exceptionを発生させる。
10. カタログサイト(詳細)アクセスURLの末尾に"/api/3/action/package\_search"を追加して、対象カタログサイトURLとする。この時、取得したカタログサイト(詳細)アクセスURLの末尾が"/"の場合は、追加する文言を"api/3/action/package\_search"とする。
11. クエリストリング、対象カタログサイトURLを引数としてカタログ検索I/F(HTTPS CKAN)を呼び出す。

* カタログ検索I/Fで例外が発生した場合は該当するステータスコードとメッセージをレスポンスに設定する。
* カタログ検索I/Fが成功した場合はステータスコードに「200」を、ボディに検索結果文字列を設定する。

#### データ交換

1. HTTPリクエストヘッダとして、リソースURL、リソース提供手段識別子、認証トークン、NGSIオプションを取得する。認証トークン、NGSIオプションが取得できない場合はNoneを設定する。
2. リソースURL、リソース提供手段識別子、認証トークン、NGSIオプションをログ出力する。
3. 各コンフィグファイルから以下を取得する。コンフィグから情報が取得できない場合、独自Exceptionを発生させる。
   * ckan.json
     1. カタログサイト(公開)アクセスURL (release\_ckan\_url)
     2. カタログサイト(詳細)アクセスURL (detail\_ckan\_url)
   * connector.json
     1. CADDEユーザID（提供者） (provider\_id)
     2. 提供者コネクタID (provider\_connector\_id)
     3. 提供者側コネクタのシークレット (provider\_connector\_secret)
   * http.json（リソース提供手段識別子が"file/http"の場合）
     1. 認可情報（authorization）
        + リソースURLより認可識別URL（url）を生成し、キーにする
          1. 認可設定（enable）
     2. 取引市場利用（contract\_management\_service）
        + リソースURLより取引市場利用識別URL（url）を生成し、キーにする
          1. 取引市場利用制御値（enable）
   * ftp.json（リソース提供手段識別子が"file/ftp"の場合）
     1. 認可情報（authorization）
        + リソースURLより認可識別URL（url）を生成し、キーにする
          1. 認可設定（enable）
     2. 取引市場利用（contract\_management\_service）
        + リソースURLより取引市場利用識別URL（url）を生成し、キーにする
          1. 取引市場利用制御値（enable）
   * ngsi.json（リソース提供手段識別子が"api/ngsi"の場合）
     1. 認可情報（authorization）
        + リソースURLより認可識別URL（url）を生成し、キーにする
          1. 認可設定（enable）
     2. 取引市場利用（contract\_management\_service）
        + リソースURLより取引市場利用識別URL（url）を生成し、キーにする
          1. 取引市場利用制御値（enable）
4. NGSIオプションが設定されている場合は"key:value,・・・"形式から辞書型に変換する。変換に失敗した場合は、独自Exceptionを発生させる。
5. 認証トークンがNoneでない場合、以下の認可に関する処理を行う。
6. 認証トークン、CADDEユーザID（提供者）、提供者コネクタID、提供者側コネクタのシークレットを引数として、認可I/Fの認可トークン取得を呼び出す。
7. 実行結果を確認し、次のいずれかの処理を行う。
   * + ステータスコードが「200」の場合は、レスポンスのヘッダ" auth\_token "に設定されている情報を認可トークンとして取得する。
     + ステータスコードが「200」以外の場合は、該当するステータスコードとメッセージを設定し、独自Exceptionを発生させる。
8. 認可トークン、CADDEユーザID（提供者）、提供者コネクタID、提供者側コネクタのシークレットを引数として認可I/Fの認可トークン検証を呼び出す。
9. 実行結果を確認し、次のいずれかの処理を行う。
   * + ステータスコードが「200」の場合は、レスポンスヘッダの"consumer\_id"に設定されている情報をCADDEユーザID（利用者）として取得する。
     + ステータスコードが「200」以外の場合は、該当するステータスコードとメッセージを設定し、独自Exceptionを発生させる。
10. 認可設定の取得値がtrueの場合、以下を行う。
    1. CADDEユーザID（提供者）、提供者コネクタID、提供者側コネクタのシークレット、リソースURL、認可トークンを引数として認可I/Fの認可確認を呼び出す。
    2. 実行結果を確認し、次のいずれかの処理を行う。
       * ステータスコードが「200」の場合、レスポンスの以下ヘッダに値が設定されていれば、記載の通りにデータを取得する。値のない場合はＮoneを設定する。
         1. "x-cadde-contract-id"は、取引IDとしてデータを取得する。
         2. "x-cadde-contract-type"は、契約形態としてデータを取得する。
         3. "x-cadde-contract-management-service-url"は、契約管理サービスURLとしてデータを取得する。
       * ステータスコードが「200」以外の場合は、独自Exceptionを発生させる。
11. リソースURLから詳細検索CKANを逆引き検索し、交換実績記録用リソースIDを取得する。
12. "/api/3/action/resource\_search"を、カタログサイト(公開)アクセスURL、カタログサイト(詳細)アクセスURLの末尾に追加する。この時、取得したカタログサイト(公開)アクセスURL、カタログサイト(詳細)アクセスURLの末尾が"/"の場合は、追加する文言を"api/3/action/resource\_search"とする。カタログサイト(詳細)アクセスURLがNoneの場合は、カタログサイト(詳細)アクセスURLに対しては実施しない。
13. リソースURLを用いて、以下の通り検索条件を作成する。  
    検索条件= "?query=url:" + リソースURL
14. 検索条件、カタログサイト(公開)アクセスURLを引数としてカタログ検索を行う。カタログ検索で例外が発生した場合は該当するステータスコードとメッセージをレスポンスに設定する。カタログ検索I/Fが成功した場合は、結果をカタログサイト(公開)検索結果として保持する。
15. カタログサイト(詳細)アクセスURLがNoneでない場合は検索条件、カタログサイト(詳細)アクセスURLを引数としてカタログ検索を行う。カタログ検索で例外が発生した場合は該当するステータスコードとメッセージをレスポンスに設定する。カタログ検索が成功した場合は、結果をカタログサイト(詳細)検索結果として保持する。カタログサイト(詳細)アクセスURLがNoneの場合はカタログサイト(詳細)検索結果にNoneを設定する。
16. カタログサイト(公開)検索結果、カタログサイト(詳細)検索結果を確認する。いずれも、検索結果が0件もしくは、Noneの場合は、独自Exceptionを発生させる。
17. 取得したカタログサイト(公開)検索結果、カタログサイト(詳細)検索結果を以下の通り確認し、交換実績記録用リソースIDを取得する。
    * 1. 取得したカタログサイト(公開)検索結果から1件分のカタログを取り出す。
      2. カタログの交換実績記録用リソースIDを確認し、以下の判定ごとに処理を行う。
         + カタログに交換実績記録用リソースIDが設定されている場合、これを取得する。交換実績用リソースIDが設定されていない場合は、何も処理を行わない。
         + 暫定的に交換実績記録用リソースIDとなりうる交換実績記録用リソースID（以降、暫定交換実績記録用リソースIDと称す）がNoneかつ、交換実績記録用リソースIDが設定されている場合は、暫定交換実績記録用リソースIDに交換実績記録用リソースIDの値を設定する。
         + 暫定交換実績記録用リソースIDがNoneでないかつ、交換実績記録用リソースIDが設定されている場合は、暫定交換実績記録用リソースIDと交換実績記録用リソースIDを比較し、同じ値の場合は何も処理を行わない。異なる値の場合は独自Exceptionを発生させる。
      3. 検索結果が2件以上の場合、(ア)に戻り、検索結果として抽出されたカタログ全てを判定する。全てのカタログを確認し終わった場合は、暫定交換実績記録用リソースIDを交換実績記録用リソースIDとして設定する。（本項目の処理結果の例をに示す）

表 ４．２．２‑１ 取得したカタログによる交換実績用リソースIDの設定例

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| パターンNo | 交換実績用リソースID | | | |
| カタログ1件目 | カタログ2件目 | カタログ3件目 | 最終取得値 |
| 1 | ABCD | ABCD | ABCD | ABCD |
| 2 | ABCD | (設定なし) | ABCD | ABCD |
| 3 | ABCD | ABCD | WXYZ | 独自Exception発生 |
| 4 | ABCD | 設定なし | WXYZ | 独自Exception発生 |
| 5 | 設定なし | 設定なし | 設定なし | None |

1. リソースURL、CADDEユーザID（利用者）、NGSIオプションをヘッダに設定して、リソース提供手段識別子に対応するコネクタ共通データ提供I/Fの処理を呼び出す。リソース提供手段識別子の各対応は後述の通りとする。リソース提供手段識別子が"api/ngsi"、"file/ftp"、 "file/http"以外の文字列の場合は、独自Exceptionを発生させる。なお、戻り値で取得したファイルデータは保持する。データ取得時にExceptionが発生した場合は、対応するステータスコードを設定してレスポンスを返す。
   * リソース提供手段識別子が"file/http"の場合  
     データ提供I/F(HTTPS)
   * リソース提供手段識別子が"file/ftp"の場合  
     データ提供I/F(FTP)
   * リソース提供手段識別子が"api/ngsi"の場合  
     データ提供I/F(HTTPS NGSI)
2. 取得対象データの取引市場利用制御値が”true”の場合、契約有のデータとしてデータ証憑通知（送信）を行う。
   1. 取引IDがNoneの場合、独自Exceptionを発生させる。
   2. 取得したデータよりハッシュアルゴリズムがSHA512のハッシュ値を生成する。
   3. CADDEユーザID（提供者) 、CADDEユーザID(利用者)、取引ID、ハッシュ値、契約管理サービスURL、認証トークンを引数として来歴管理I/Fのデータ証憑通知（送信）にHTTPリクエストを発行する。
   4. 実行結果を確認し、応答のステータスコードが「200」以外の場合は、独自Exceptionを発生させる。
3. 交換実績記録用リソースIDがNoneでないかつ、CADDEユーザID（利用者）がNoneでない場合、来歴登録が必要なデータとして送信履歴登録を行う。送信履歴登録を呼び出さなかった場合は、識別情報に空文字("")を設定する。
   1. CADDEユーザID（提供者）、CADDEユーザID（利用者）、交換実績記録用リソースID、認証トークンをヘッダに設定して、来歴管理I/Fの送信履歴登録にHTTPリクエストを発行する。
   2. 実行結果を確認し、次のいずれかの処理を行う。
      * 応答のステータスコードが「200」の場合は、戻り値として識別情報、来歴管理サービスURLを取得する。
      * 応答のステータスコードが「200」以外の場合は、独自Exceptionを発生させる。
4. レスポンスを以下の通りに生成する。
   1. ステータスコードを200に設定する。
   2. レスポンスボディにサブシステムから取得した戻り値を設定する。
   3. 以下のレスポンスヘッダに以下を設定する。値が取得できない場合は空文字("")を設定する。
      * x-cadde-provenance：識別情報
      * x-cadde-provenance-management-service-url：来歴管理サービスURL
   4. 取得対象データの取引市場利用制御値が”true”の場合、以下のレスポンスヘッダに値を設定する。値が取得できない場合、取得対象データの取引市場利用制御値が”faulse”の場合は空文字("")を設定する。
      * x-cadde-contract-id：取引ID
      * x-cadde-contract-type：契約形態
      * x-cadde-contract-management-service-url：契約管理サービスURL

### 出力ログ

独自Exceptionに付与するエラーメッセージおよび出力ログの情報は「詳細設計書\_別紙1\_メッセージ一覧.xlsx」を参照すること。

# 認可I/Fサブシステム

## 内部仕様

### システム構成

システム構成を記載する。

凡例:

: 対象サブシステム

: その他システム

: HTTPS通信

コネクタメイン

コンテナ

認証認可I/F

コンテナ

認可サーバ

提供者側コネクタ Dockerコンテナ群

認可トークン

認可トークン取得

認可トークン

認可トークン検証

CADDEユーザID

（利用者）

認可確認

取引ID

契約形態

認可確認

取引ID

契約形態

認可トークン取得

認可トークン検証

CADDEユーザID

（利用者）

図 ５．１．１‑１ システム構成

### 公開インタフェース

「詳細設計書\_別紙10\_提供者\_認可I/F.html」を参照すること。

### コンフィグ定義

「基本設計書\_別紙2\_コンフィグパラメータ一覧.xlsx」を参照すること。

## 機能詳細

### 認可I/F機能 処理一覧

表 ５．２．１‑１ インタフェース情報

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| # | 処理名 | 概要 | URL | 関数名 | メソッド |
|  | 認可トークン取得 | 認可サーバに対して、トークンフェデレーションリクエストを行い、認可トークンを返す。 | /token\_federation | token\_federation | GET |
|  | 認可トークン検証 | 認可サーバに対して、トークンイントロスペクションをリクエストし、CADDEユーザID（利用者）を取得する。 | /token\_introspect | token\_introspect | GET |
|  | 認可確認 | 認可サーバに対して、認可確認をリクエストし、取引ID、契約形態を取得する。 | /token\_contract | token\_contract | GET |

### 処理フロー

#### 認可トークン取得

1. HTTPリクエストヘッダから、認証トークン、CADDEユーザID（提供者）、提供者コネクタID、提供者側コネクタのシークレットを取得する。
2. 認証トークン、提供者コネクタID、提供者側コネクタのシークレットをメッセージコードとともにログ出力する。
3. コンフィグファイルから以下を取得する。コンフィグから情報が取得できない場合、独自Exceptionを発生させる。
   * authorization.json
     1. 認可サーバアクセスURL(authorization\_server\_url)
4. 認可サーバにPOST通信でHTTPリクエストを発行する。リクエストのbody部に以下の値を設定する。  
   token=(認証トークン) & client\_id=(提供者コネクタID)& resource\_url=(リソースURL)
5. 実行結果を確認し、次のいずれかの処理を行う。

* HTTPステータスコードが「2xx」以外の場合、独自Exceptionを発生させる。
* HTTPステータスコードが「2xx」の場合、レスポンスbody部の" auth-token"の値を取得し、認可トークンとする。レスポンスにHTTPステータスコード「200」、ヘッダ" x-cadde-auth-token"に認可トークンを設定し、応答を返す。

#### 認可トークン検証

1. HTTPリクエストヘッダとして、認可トークン、CADDEユーザID（提供者）、提供者コネクタID、提供者側コネクタのシークレットを取得する。
2. 認証トークン、提供者コネクタID、提供者側コネクタのシークレットをメッセージコードとともにログ出力する。
3. コンフィグファイルから以下を取得する。コンフィグから情報が取得できない場合、独自Exceptionを発生させる。
   * authorization.json
     1. 認可サーバアクセスURL(authorization\_server\_url)
4. 認可サーバにPOST通信でHTTPリクエストを発行する。リクエストのbody部に以下の値を設定する。  
   client\_id=(提供者コネクタID)&client\_secret=(提供者側コネクタのシークレット)&token=(認可トークン)
5. 実行結果を確認し、次のいずれかの処理を行う。

* 実行結果でエラーが発生している場合は、独自Exceptionを発生させる。エラーが発生しているかの判定は、実行結果のHTTPステータスコードが「2xx」以外の場合にエラーと判定する。
* HTTPステータスコードが「2xx」かつレスポンスbody部の"active"の値が"true"でなかった場合、独自Exceptionを発生させる。
* HTTPステータスコードが「2xx」かつ、レスポンスbody部の"active"の値が"true"の場合、レスポンスbody部の"username"の値を取得し、CADDEユーザID（利用者）とする。レスポンスにHTTPステータスコード「200」、ヘッダ"x-cadde-consumer-id"にCADDEユーザID（利用者）を設定し、応答を返す。

#### 認可確認

1. HTTPリクエストヘッダとして、CADDEユーザID（提供者）、提供者コネクタID、提供者側コネクタのシークレット、リソースURL、認可トークンを取得する。
2. CADDEユーザID（提供者）、提供者コネクタID、提供者側コネクタのシークレット、リソースURL、認可トークンをメッセージコードとともにログ出力する。
3. コンフィグファイルから以下を取得する。コンフィグから情報が取得できない場合、独自Exceptionを発生させる。
   * authorization.json
     1. 認可サーバアクセスURL(authorization\_server\_url)
4. 認可サーバにPOST通信でHTTPリクエストを発行する。リクエストのbody部に以下の値を設定する。  
   token=(認可トークン) & client\_id=(提供者コネクタID) &client\_secret=(提供者側コネクタのシークレット) & resource\_url=(リソースURL)
5. 実行結果を確認し、次のいずれかの処理を行う。
   * HTTPステータスコードが「2xx」以外の場合、独自Exceptionを発生させる。

* HTTPステータスコードが「2xx」の場合、以下のレスポンスヘッダ部の値を取得し、レスポンスヘッダとする。レスポンスにHTTPステータスコード「200」とし、レスポンスヘッダと併せて応答を返す。
  1. 取引ID  
     レスポンスヘッダ"x-cadde-contract-id"とする。
  2. 契約形態  
     レスポンスヘッダ"x-cadde-contract-type"とする。
  3. 契約管理サービスURL  
     レスポンスヘッダ"x-cadde-contract-management-service-url"とする。

### 出力ログ

独自Exceptionに付与するエラーメッセージおよび出力ログの情報は「詳細設計書\_別紙1\_メッセージ一覧.xlsx」を参照すること。

# カタログ検索I/Fサブシステム

## 内部仕様

### システム構成

システム構成を記載する。

凡例:

: 対象サブシステム

: その他システム

: HTTPS通信

カタログ

検索I/F

コンテナ

カタログ

検索I/F

コンテナ

コネクタ

メイン

コンテナ

利用者側コネクタ

Dockerコンテナ群

カタログ

検索

検索結果

提供者側コネクタ Dockerコンテナ群

カタログ

検索

検索結果

図 ６．１．１‑１ システム構成

### 公開インタフェース

「詳細設計書\_別紙8\_提供者\_カタログ検索IF.html」を参照すること。

### 内部データ一覧

以下の内部データを保持する。

表 ４．４‑１ 内部データ一覧

|  |  |  |
| --- | --- | --- |
| # | データ名 | 概要 |
|  | url\_main\_search | 提供者コネクタ コネクタメインの詳細カタログ検索要求に接続するためのURL |

## 機能詳細

### カタログ検索I/F機能 処理一覧

表 ６．２．１‑１ インタフェース情報

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| # | 処理名 | 概要 | URL | 関数名 | メソッド |
|  | カタログ検索 | 詳細検索リクエストを受け付け、メイン制御に処理を依頼する。 | /cadde/api/v4/catalog | search | GET |

### 処理フロー

#### カタログ検索

1. クエリストリングとして検索条件、HTTPリクエストのヘッダから認証トークンを取得する。認証トークンが取得できない場合は、Noneを設定する。
2. クエリストリングをメッセージコードとともにログ出力する。
3. クエリストリングとコネクタメインのURL(固定)から、以下の通り検索用URLを作成する。  
   検索用URL = コネクタメインのURL + クエリストリング
4. クエリストリング、認証トークンをヘッダに設定し、検索用URL を用いてコネクタメインのカタログ検索(詳細検索) へHTTPリクエストを発行する。
5. 検索結果を確認し、次のいずれかの処理を行う。

* 検索結果のステータスコードが「200」でない場合は、CKANから返却されたステータスコードを設定し、ボディにエラーメッセージコードを設定して応答を返す。
* 検索結果のステータスコードが「200」の場合、ステータスコードに「200」を設定し、ボディに検索結果文字列を設定して応答を返す。

### 出力ログ

独自Exceptionに付与するエラーメッセージおよび出力ログの情報は「詳細設計書\_別紙1\_メッセージ一覧.xlsx」を参照すること。

# 

# データ交換I/Fサブシステム

## 内部仕様

### システム構成

システム構成を記載する。

凡例:

: 対象サブシステム

: その他システム

: HTTPS通信

データ

交換I/F

コンテナ

データ交換I/F

コンテナ

コネクタ

メイン

コンテナ

利用者側コネクタ

Dockerコンテナ群

データ交換

ファイルもしくは

JSONと識別情報

データ交換

ファイルもしくはJSONと識別情報

提供者側コネクタ Dockerコンテナ群

図 ７．１．１‑１ システム構成

### 公開インタフェース

「詳細設計書\_別紙9\_提供者\_データ交換IF(CADDE).html」を参照すること。

### 内部データ一覧

以下の内部データを保持する。

表７．１．３‑１ 内部データ一覧

|  |  |  |
| --- | --- | --- |
| # | データ名 | 概要 |
|  | url\_main\_file | 提供者コネクタ コネクタメインのデータ取得I/F(HTTPS CADDE)に接続するためのURL |

## 機能詳細

### データ交換I/F機能 処理一覧

表 ７．２．１‑１ インタフェース情報

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| # | 処理名 | 概要 | URL | 関数名 | メソッド |
|  | データ交換 | CADDEリクエストを受け付け、コネクタメインに処理を依頼する。 | /cadde/api/v4/file | files | GET |

### 処理フロー

#### データ交換

1. HTTPリクエストヘッダとして、リソースURL、リソース提供手段識別子、NGSIオプション、認証トークンを取得する。認証トークンを取得できない場合は、認証トークンをNoneに設定する。
2. リソースURL、リソース提供手段識別子、NGSIオプションをメッセージコードとともにログ出力する。
3. リソースURL、リソース提供手段識別子、NGSIオプション、認証トークンをヘッダに設定してコネクタメインのデータ交換にHTTPリクエストを発行する。
4. 実行結果を確認し、次のいずれかの処理を行う。

* エラーが発生している場合は、独自Exceptionを発生させる。エラーが発生しているかの判定は、実行結果のHTTPステータスコードが「2xx」以外の場合にエラーと判定する。
* HTTPステータスコードが「2xx」かつリソース提供手段識別子が"api/ngsi"の場合は実行結果からコンテキスト情報、レスポンスヘッダ情報を取得し、HTTPステータスコード「200」をレスポンスに設定して、応答を返す。レスポンスはflaskライブラリを利用して作成する。なお、レスポンスには以下を含める。
  1. レスポンスヘッダの"eventIdentifier"に実行結果から取得した識別情報を設定する。
  2. レスポンスヘッダに以下を設定する。値が取得できない場合は空文字("")を設定する。
     1. 識別情報（x-cadde-provenance）
     2. 来歴管理サービスURL（x-cadde-provenance-management-service-url）
     3. 取引ID（x-cadde-contract-id）
     4. 契約形態（x-cadde-contract-type）
     5. 契約管理サービスURL（x-cadde-contract-management-service-url）
* HTTPステータスコードが「2xx」かつリソース提供手段識別子が"api/ngsi"でない場合は実行結果からファイル名、レスポンスヘッダ情報を取得し、HTTPステータスコード「200」をレスポンスに設定して、応答を返す。なお、レスポンスには以下を含める。
  1. レスポンスヘッダの"attachment\_filename"に実行結果から取得したファイル名に設定する。
  2. レスポンスヘッダの"eventIdentifier"に実行結果から取得した識別情報を設定する。
  3. レスポンスヘッダに以下を設定する。値が取得できない場合は空文字("")を設定する。

1. 識別情報（x-cadde-provenance）
2. 来歴管理サービスURL（x-cadde-provenance-management-service-url）
3. 取引ID（x-cadde-contract-id）
4. 契約形態（x-cadde-contract-type）
5. 契約管理サービスURL（x-cadde-contract-management-service-url）

### 出力ログ

独自Exceptionに付与するエラーメッセージおよび出力ログの情報は「詳細設計書\_別紙1\_メッセージ一覧.xlsx」を参照すること。

# 来歴管理I/Fサブシステム

## 内部仕様

### システム構成

システム構成を記載する。

凡例:

: 対象サブシステム

: その他システム

: HTTPS通信

コネクタメイン

コンテナ

来歴管理

I/F

コンテナ

来歴管理

エージェント

送信履歴登録

識別情報

送信履歴登録

識別情報

提供者側コネクタ Dockerコンテナ群

データ証憑

通知(送信)

識別情報

データ証憑

通知(送信)

識別情報

図 ８．１．１‑１ システム構成

### 公開インタフェース

「詳細設計書\_別紙11\_提供者\_来歴管理I/F(CADDE).html」を参照すること。

### 内部データ一覧

以下の内部データを保持する。

表８．１．３‑１ 内部データ一覧

|  |  |  |
| --- | --- | --- |
| # | データ名 | 概要 |
|  | url\_eventwithhash\_sent | 提供者コネクタ 来歴管理I/Fの送信履歴登録に接続するためのURL |

## 機能詳細

### 来歴管理I/F機能 処理一覧

表 ８．２．１‑１ インタフェース情報

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| # | 処理名 | 概要 | URL | 関数名 | メソッド |
|  | 送信履歴登録 | 来歴管理に送信履歴登録を行い、識別情報を取得する。 | /eventwithhash/sent | sent | GET |
|  | データ証憑通知 （送信） | 契約管理サービスにデータ証憑通知（送信）を行う。 | /eventwithhash/voucher | voucher | GET |

### 処理フロー

#### データ証憑通知（送信）

1. HTTPリクエストヘッダとして、CADDEユーザID（提供者）、CADDEユーザID（利用者）、取引ID、ハッシュ値、契約管理サービスURL、認証トークンを取得する。
2. CADDEユーザID（提供者）、CADDEユーザID（利用者）、取引ID、ハッシュ値、契約管理サービスURLをメッセージコードとともにログ出力する。認証トークンについてはログ出力しない。
3. CADDEユーザID（提供者）、CADDEユーザID（利用者）、取引ID、ハッシュ値、認証トークンをヘッダとして、取得した契約管理サービスURLへHTTPリクエストを行い、契約管理サービスを呼び出す。
4. 実行結果を確認し、次のいずれかの処理を行う。
   * エラーが発生している場合は、独自Exceptionを発生させる。
   * 通知に成功した場合、レスポンスにHTTPステータスコード「200」を設定し、応答を返す。

#### 送信履歴登録

1. HTTPリクエストヘッダとして、CADDEユーザID（提供者）、CADDEユーザID（利用者）、交換実績記録用リソースID、認証トークンを取得する。
2. CADDEユーザID（提供者）、CADDEユーザID（利用者）、交換実績記録用リソースIDをメッセージコードとともにログ出力する。認証トークンについてはログ出力しない。
3. コンフィグファイルから以下を取得する。コンフィグから情報が取得できない場合、独自Exceptionを発生させる。
   * provenance.json
     1. 来歴管理I/FへのアクセスURL(provenance\_management\_api\_url)
4. 来歴管理サービスの履歴登録APIにAPI定義から自動生成したクライアントを使用してHTTPリクエストを発行する。body部を以下の通り設定する。  
   {  
    "cdldatamodelversion": "2.0",

"cdleventtype": "Sent",  
 "dataprovider": (CADDEユーザID（提供者）),  
 "datauser": (CADDEユーザID（利用者）),  
 "cdlpreviousevents": (交換実績記録用リソースID)  
}

1. 実行結果を確認し、次のいずれかの処理を行う。

* エラーが発生している場合は、独自Exceptionを発生させる。
* 登録に成功した場合、イベント識別子(cdleventid)を取得し、識別情報とする。レスポンスにHTTPステータスコード「200」、レスポンスの以下ヘッダに値を設定し、応答を返す。
  1. "x-cadde-provenance"：識別情報
  2. "x-cadde-provenance-management-service-url"：来歴管理サービスURL

### 出力ログ

独自Exceptionに付与するエラーメッセージおよび出力ログの情報は「詳細設計書\_別紙1\_メッセージ一覧.xlsx」を参照すること。