平成 29 年度補正予算　産業データ共有促進事業費補助金

製造プラットフォームオープン連携事業

サブシステムプロトタイプ実装仕様

（ＨＣＳ／ＨＣＴ編）

Ver.1.2

2019 年 3 月 1 日

一般社団法人

インダストリアル・バリューチェーン・イニシアティブ

目次

1. はじめに

1.1 対象範囲

本ドキュメントは、2018 年度「製造プラットフォームオープン連携事業」のシステム実装基本要件仕様[1] に基づき、デモ用システム内における HCT および HCS の仕様に関して記載しています。

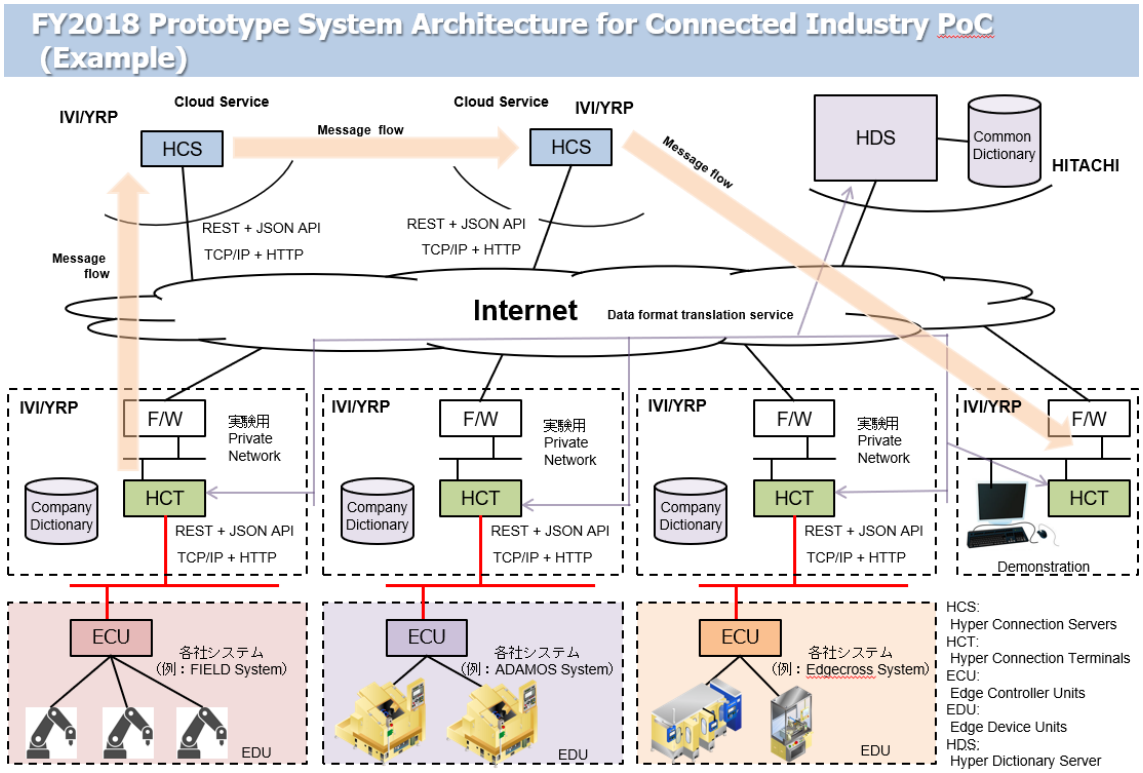

図 1 システムアーキテクチャ

1.2 変更履歴

版数	作成日	作成者	改版内容
0.01	2018.9.6	UNL 坂	初版作成
0.02	2018.9.19	UNL 坂	随時追加
0.03	2018.10.16	UNL 坂	随時追加
0.04	2018.10.18	UNL 新堂	章構成を変更
0.05	2018.10.19	UNL 坂	体裁調整
0.06	2018.10.24	UNL 坂	取引プロファイルおよび一部 API を修正
0.07	2018.10.25	UNL 坂	取引プロファイルおよび一部 API を修正
1.0	2018.11.9	UNL 坂	API/TDP を一部修正、動作例を追加
1.1	2018.11.22	UNL 坂	取引プロトコルを一部修正、API を修正/追加
1.11	2018.12.7	UNL 坂	動作例を追加
1.12	2019.1.15	UNL 坂	各箇所における誤りを修正
1.2	2019.2.26	UNL 坂	最終版に向けてコメント等を削除、動作例を追記

1.3 構築方針

HCS、HCT、HDS は、機能の実行のために必要十分なソースコードを MIT ライセンスポリシーのもとで GitHub に公開し、オープンソースとして開発します。

1.4 ソースコード

本仕様書に対応したソースコードは、以下で公開されています。
https://github.com/ivi-ciof/

2. 基本事項

2.1 動作条件

以下に HCT を動作させる環境について、示します。

カテゴリ	名称	メーカー	型式	バージョン
ハード	コンピュータ	Lenovo	ThinkPad X270 20HN0010JP	
		CPU: Corei3 以上、メモリ: 4GB 以上、Ethernet ポート有(RJ-45)		
ソフト	OS	Microsoft	Windows10 Pro	1709
	バーチャルマシン		Docker	18.06.1-ce
	HTTP サーバ		Nginx (Docker image)	1.15.5
	開発環境	-	Ruby	2.5.1p57
		-	Puma	3.12.0
		-	Rails	5.2.1
		-	PostgreSQL	10.5

以下に HCS を動作させる環境について、示します。

カテゴリ	名称	メーカー	型式	バージョン
ソフト	クラウドサービス	Amazon	AWS	
	バーチャルマシン		Docker	
	HTTP サーバ		Nginx	
	開発環境	-	Ruby	
		-	Puma	
		-	Rails	
		-	PostgreSQL	

2.2 通信プロトコル

HTTP(Hypertext Transfer Protocol)/1.1 [3] に準拠します。

2.3 パラメータの形式

通信に用いる文字コードは UTF-8(BOM なし)とします。
要求パラメータおよび応答パラメータは原則 JSON 形式とします。

2.4 HTTP ステータスコード

HCS/HCT API は、処理結果を示すために次に示す標準的な HTTP のステータスコードを使用します。

ステータスコード	説明
200	Success リクエストが適切に処理されたことを示します。
400	Bad request リクエストヘッダ、クエリパラメータ、またはリクエストボディが不正であることを示します。
500	Server error サーバで内部エラーが発生したことを示します。

2.5 ユーザ認証

ECU による HCT および HCS への接続には、RFC7235[4] の規定に基づき、BASIC 認証のパラメータを HTTP の Authorization ヘッダに含めます。
本デモにおいては、ユーザ名およびパスワードは、あらかじめ決められた以下のものを用います。
ユーザ名：CIOF
パスワード：pciof

2.6 ID 生成

ID の生成方針は、「サブシステムプロトタイプ実装仕様（HDS/HCM 編)」[2] に従い、以下の通りとします。

- 各種データ(共通データ辞書、変換マップ、等)を表す ID は、データの登録用 API を提供する側(サーバ側)が、登録処理時に採番します。
- HCS/HCT/HCM/HDS/ECU/EDU 等のモジュールやユーザなど、登録用 API のない場合は決めうちとします。
- ID は 32 桁(固定長)の 16 進(0～9、a～f)とします。
  - 先頭 4 桁　：登録用 API を提供するモジュール固有(決めうち)の値
  - 後半 28 桁　：モジュール内でユニーク性を担保

No.	モジュール名	データ名	先頭 4 桁	備考
1	HCT	TCP	2000	本 ID はグローバルでユニークとします
2		TDP	2001	
3		TSP	2002	
4		ECU	2003	
5		EDU	2004	
6		DCR	2005	
7		message	2006	
8		DCM	2007	
9	HCS	HCT	1000	

2.7 データ構造

各 HCT 間で送受信するデータは、以下の通りです。
本節では、メッセージヘッダおよび本体のデータ構造について記載します。

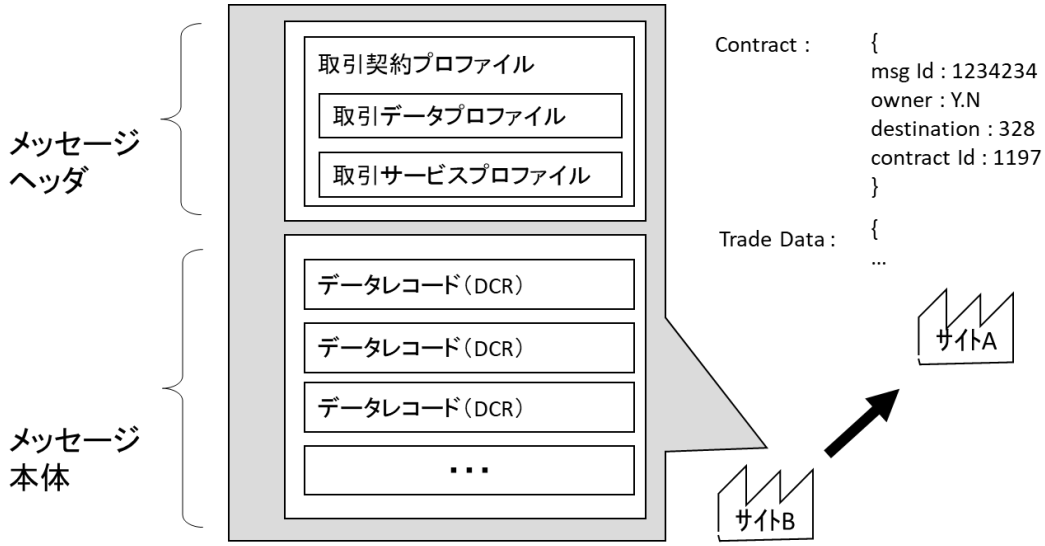

図 2　メッセージのデータ構造概略

2.7.1　TCP（**取引契約プロファイル**：Trade Contract Profile）

以下に、取引契約プロファイルのデータフォーマットを示します。

No.	項目名	説明	データ型
1	tcp_id	取引契約プロファイル ID	String
2	tcp_name	取引契約プロファイル名	String
3	owner_id	事業者 ID(データ送信者)	String
4	owner_name	事業者名	String
5	created_date	作成日(YYYY/MM/DD)	String
6	author	作成者	String
7	user_id	取引先 ID(データ利用者)	String
8	user_name	取引先名	String
9	data_dictionary_id	共通辞書 ID	String
10	data_dictionary_name	共通辞書名	String
11	trigger	開始トリガ（提供者 or 利用者）	String
12	start_event	開始イベント	String
13	event_class	イベント区分	String
14	transaction_start_condition	取引開始条件	String
15	start_date	開始日(YYYY/MM/DD)	String
16	transaction_end_condition	取引終了条件	String
17	end_date	終了日(YYYY/MM/DD)	String
18	owner_hct_id	事業者ターミナル ID	String
19	owner_unit_id	事業者実施ユニット ID(データ提供するユニット ID)	String
20	owner_unit_class	事業者ユニット区分(ECU/EAU)	String

21	owner_edu_id	事業者実施 EDU ID	String
22	user_hct_id	取引先ターミナル ID	String
23	user_unit_id	取引先実施ユニット ID(データ利用するユニット ID)	String
24	user_unit_class	取引先ユニット区分(ECU/EAU)	String
25	user_edu_id	取引先実施 EDU ID	String
26	owner_tdp_id	事業者の取引データプロファイル ID	String
27	user_tdp_id	取引先の取引データプロファイル ID	String
28	owner_tsp_id	事業者のサービスプロファイル ID	String
29	user_tsp_id	取引先のサービスプロファイル ID	String
30	request_parameter	リクエストパラメータリスト(PULL 型の取引の場合にテキストで条件指定をする)	String[]
31	storage_restriction	保存制限をテキストで指定	String
32	use_restriction	利用制限をテキストで指定	String
33	modified_restriction	改変制限をテキストで指定	String
34	disclosure_restriction	開示制限をテキストで指定	String
35	other_conditions	契約条件・課金条件などをテキストで指定	String

例：


```
{
  "tcp_id":"tcp01",
  "tcp_name":"TCP サンプル",
  "owner_id":"owner01",
  "owner_name":"株式会社テスト",
  "created_date":"2018/10/29",
  "author":"田中太郎",
  "user_id":"user01",
  "user_name":"テスト工務店",
  "data_dictionary_id":"sdd01",
  "data_dictionary_name":"テスト用共有辞書",
  "trigger":"提供者",
  "start_event":"開始イベント",
  "event_class":"イベント区分",
  "transaction_start_condition":"取引開始条件",
  "start_date":"2018/01/01",
  "transaction_end_condition":"取引終了条件",
  "end_date":"2020/12/31",
  "owner_hct_id":"hct01",
  "owner_unit_id":"ecu01",
  "owner_unit_class":"ECU",
  "owner_edu_id":"edu01",
```



```
  "user_hct_id":"hct11",
  "user_unit_id":"ecu11",
  "user_unit_class":"EDU",
  "user_edu_id":"edu11",
  "owner_tdp_id":"tdp01",
  "user_tdp_id":"tdp11",
  "owner_tsp_id":"tsp01",
  "user_tsp_id":"tsp11",
  "request_parameter":"リクエストパラメータ",
  "storage_restriction":"保存制限",
  "use_restriction":"利用制限",
  "modified_restriction":"改変制限",
  "disclosure_restriction":"開示制限",
  "other_conditions":"その他の条件"
}
```


2.7.2 TDP（取引データプロファイル：Trade Data Profile）

以下に、取引データプロファイルのデータフォーマットを示します。

No.	項目名	説明	データ型
1	tdp_id	取引データプロファイル ID	String
2	tdp_name	取引データプロファイル名	String
3	dcm_class	区分(利用 or 提供)	String
4	transaction_start_condition	取引開始条件	String
5	start_date	開始日(YYYY/MM/DD)	String
6	transaction_end_condition	取引終了条件	String
7	end_date	終了日(YYYY/MM/DD)	String
8	storage_restriction	保存制限をテキストで指定	String
9	use_restriction	利用制限をテキストで指定	String
10	modified_restriction	改変制限をテキストで指定	String
11	disclosure_restriction	開示制限をテキストで指定	String
12	dtm_id	辞書変換マップ ID のリスト	String[]

例：


```
{
  "tdp_id":"tdp01",
  "tdp_name":"株式会社テストのサンプル TDP",
  "dcm_class":"利用",
  "transaction_start_condition":"取引開始条件",
  "start_date":"2018/01/01",
  "transaction_end_condition":"取引終了条件",
  "end_date":"2020/12/31",
  "storage_restriction":"保存制限なし",
  "use_restriction":"利用制限なし",
  "modified_restriction":"改変制限なし",
```



```
 "disclosure_restriction":"開示制限なし",
 "dtm_id": ["dtm01"]
}
```

2.7.3 TSP（取引サービスプロファイル：Trade Service Profile）

以下に、取引サービスプロファイルのデータフォーマットを示します。

No.	項目名	説明	データ型
1	tsp_id	取引サービスプロファイル ID	String
2	tsp_name	取引サービスプロファイル名	String
3	service_dictionary_id	個別サービス辞書 ID	String
4	service_dictionary_name	個別サービス辞書名	String
5	dcm_id	データ ID	String
6	dcm_name	データ名	String
7	process_id	プロセス ID	String
8	arrangement_id	配置 ID	String
9	data_class	区分(利用 or 提供)	String
10	process_name	プロセス名	String
11	category_id	カテゴリ ID	String
12	category_name	カテゴリ名	String
13	unit_id	ユニット(ECU/EAU/EDU)ID	String
14	unit_name	ユニット(ECU/EAU/EDU)名	String
15	hct_id	ユニットが属するターミナル ID	String
16	hct_name	ユニットが属するターミナル名	String
17	transaction_start_condition	取引開始条件	String
18	start_date	開始日(YYYY/MM/DD)	String
19	transaction_end_condition	取引終了条件	String
20	end_date	終了日(YYYY/MM/DD)	String
21	event_list	起動/終了イベントリスト	JsonObject[]
22	data_restriction_list	関連データの制約リスト	JsonObject[]
23	secondary_disclosure_list	データ 2 次開示先リスト	JsonObject[]

起動/終了イベントリストのデータフォーマットを示します。

No.	項目名	説明	データ型
1	event_id	イベント ID	String
2	event_name	イベント名	String
3	event_class	区分(利用 or 提供)	String
4	event_ archive	記録	String

関連データの制約リストのデータフォーマットを示します。

No.	項目名	説明	データ型
1	dcm_id	データ ID	String
2	dcm_name	データ名	String
3	dcm_class	区分(利用 or 提供)	String
4	dcm_restriction	制約をテキストで記載	String

データ 2 次開示先のデータフォーマットを示します。

No.	項目名	説明	データ型
1	unit_id	ユニット ID	String
2	unit_name	ユニット名	String
3	disclosure_level	開示レベルをテキストで記載	String
4	access_restriction	アクセス制限をテキストで記載	String

例：

```
{
  "tsp_id":"tsp01",
  "tsp_name":"株式会社テストのサンプル TSP ",
  "service_dictionary_id":"add01",
  "service_dictionary_name":"株式会社テストのサンプル個別辞書",
  "dcm_id":"dcm01",
  "dcm_name":"温湿度計",
  "process_id":"process01",
  "arrangement_id":"arrangement01",
  "data_class":"利用",
  "process_name":"温湿度計のデータ入手プロセス",
  "category_id":"category01",
  "category_name":"サンプルカテゴリ",
  "unit_id":"ecu01",
  "unit_name":"株式会社テストの ECU",
  "hct_id":"hct01",
  "hct_name":"株式会社テストの HCT",
  "transaction_start_condition":"取引開始条件",
  "start_date":"2018/01/01",
  "transaction_end_condition":"取引終了条件",
  "end_date":"2020/12/31",
  "event_list": [
    {
      "event_id": "event01",
      "event_name": "イベント名",
      "event_class": "利用",
      "event_ archive": "イベントに関する記録"
    }
  ],
  "data_restriction_list":[
```

```
    {
      "dcm_id":"dcm01",
      "dcm_name":"制約に関するデータ",
      "dcm_class":"利用",
      "dcm_restriction":"ここに制約を記載する"
    }
  ],
  "secondary_disclosure_list":[
    {
      "unit_id":"",
      "unit_name":"",
      "disclosure_level":"なし",
      "access_restriction":"なし"
    }
  ]
}
```

2.7.4 データレコード

データレコード(DCR)のデータフォーマットについて、以下に示します。

No.	項目名	説明	データ型
1	dcr	キーとバリューの組のリスト	JsonObject[]

例

```
{
  "dcr": [{"気温":"12.3"}, {"相対湿度":"60"}]
}
```

3. HCT の機能

HCT は、以下の機能を持ちます。

3.1 プロファイルの管理

HCT は、自身が関連するプロファイルを管理します。プロファイルは、データ取引の内容および方法を示す情報です。データ提供側とデータ利用側の双方が、1 つのデータ取引に関するプロファイルを別々に持ち、データ取引契約 ID で一意に対応づけられています。プロファイルは、データ取引契約フェーズで作成され、それぞれの内容は、対応するが異なっており、相手側のプロファイルの内容を直接その形式で知ることはできません。したがって、HCT は、ある一つの取引契約プロファイル ID に対して、一つの取引データプロファイルと一つの取引サービスプロファイルを紐づけておくことになります。
プロファイルのデータフォーマットについては、2.7 節を参考にしてください。

3.2 サンプルデータの保持

取引契約フェーズにおいて、サンプルデータの受け渡しを行いますが、このデータを HCT は保持します。サンプルデータは、HCM からのリクエストにより、登録されます。
以下にサンプルデータのデータフォーマットを示します。

No.	項目名	説明	データ型
1	sample_data_id	サンプルデータの ID	String
2	tcp_id	サンプルデータの登録先 TCP	String
3	sample_data_dcr	サンプルデータの DCR	String[]

3.3 配下の ECU と EDU の管理

HCT は、自分と同一のネットワークにつながる ECU と EDU のリストを持ちます。また、それぞれの EDU に対して実行可能な PCM のリストを持ちます。
EDU や、それらが実行可能な PCM のリストは、ECU-HCT 間 API で登録・抹消します。

3.4 サブスクライブリストの管理

HCT は、プッシュ型通知をする場合に、通知先のリストを管理します。通知先の登録および抹消は、HCM を経由して取引契約プロファイルとして登録されます。

3.5 辞書変換処理

HCT は、通信データに対して辞書変換を行います。各取引における、各 DPD に対して変換内容を HDS に照会し、結果を用いて辞書変換します。

3.6 データの暗号/復号化

HCT は、インターネット上の HCS にデータを送信する際に、特別なデータの暗号化を行わず、汎用的な HTTPS 通信によって、通信路のセキュリティを担保します。

4. HCS の機能

本デモにおいては､最終的なシステム構成とはことなり､HCS は 1 台のサーバとして実装します。すなわち、本デモで実装する HCS は、すべての HCT からのデータを受け取り保持し、すべての HCT からの要求にしたがって、必要なデータを提供します。

4.1 事業者の管理

HCS は通常、ある事業者によって管理されます。本デモにおいては、HCS は 1 台のみでの運用となるため、事業者は登録しません。

4.2 配下の HCT の管理

HCS は自分の配下の HCT のリストを持ちます。HCT のリストは、HCT-HCS 間 API で登録・抹消します。

4.3 取引メッセージの保持

HCS は HCT から受信した取引メッセージを保持します。

4.4 取引メッセージの配信

HCT から取引メッセージの配信要求があれば、それに従って必要な取引メッセージを配信し、配信後のメッセージは、消去し、それ以降保持しません。

5. HCM-HCT 間 API リファレンス

HCM-HCT 間の API リファレンス仕様として、以下を規定します。

・プロファイル登録/変更/削除/取得
・ECU/EAU からサンプルデータ取得
・サンプルデータ登録/削除
・他サイト HCT からサンプルデータ取得
・他サイト HCT の提供可能データ取得
・HCM リクエスト登録
・HCM リクエスト取得

5.1 プロファイル登録

<table>
<tr><td>API 名称</td><td>プロファイル登録</td></tr>
<tr><td>処理概要</td><td>HCT へ新規プロファイルを登録し、ID を発行します。
TCP/TDP/TSP は、その組で一度に登録しても相互に紐づけられることはありません。プロファイル間の紐づけは、プロファイルの更新を別途行うことで実現します。
(TCP/TDP/TSP データフォーマット内で)各 ID を指定した場合、その ID が HCT 内に存在しない場合に限り、指定された ID で新規プロファイルを登録します。存在した場合は、エラーとなります。</td></tr>
<tr><td>HTTP メソッド</td><td>POST</td></tr>
<tr><td>URL パス</td><td>/hct/api/v1/profiles</td></tr>
<tr><td>要求パラメータ</td><td>メッセージボディ部に、JSON 形式で以下のパラメータを与えます。
{
"tcp": "省略(TCP データフォーマット)",
"tcp_temporary_registration_flag": false,
"tdp": "省略(TDP データフォーマット)",
"tdp_temporary_registration_flag": false,
"tsp": "省略(TSP データフォーマット)",
"tsp_temporary_registration_flag": false
}
<table><tr><th>属性</th><th>型</th><th>説明</th></tr><tr><td>tcp</td><td>JsonObject</td><td>(任意)登録する TCP</td></tr><tr><td>tcp_temporary_registration_flag</td><td>Boolean</td><td>(任意)登録する TCP の仮登録フラグ</td></tr><tr><td>tdp</td><td>JsonObject</td><td>(任意)登録する TDP</td></tr><tr><td>tdp_temporary_registration_flag</td><td>Boolean</td><td>(任意)登録する TDP の仮登録フラグ</td></tr><tr><td>tsp</td><td>JsonObject</td><td>(任意)登録する TSP</td></tr><tr><td>tsp_temporary_registration_flag</td><td>Boolean</td><td>(任意)登録する TSP の仮登録フラグ</td></tr></table></td></tr>
<tr><td>応答</td><td>2 章 HTTP プロトコルに準じます</td></tr>
<tr><td>応答ヘッダ</td><td>Content-Type: application/json</td></tr>
<tr><td>応答パラメータ</td><td>メッセージボディ部に、以下のパラメータを返します。
{
"tcp_id": "tcp01",
"tdp_id": "tdp01",
"tsp_id": "tsp01"</td></tr>
</table>

}

属性	型	説明
tcp_id	String	(任意)登録した TCP の ID
tdp_id	String	(任意)登録した TDP の ID
tsp_id	String	(任意)登録した TSP の ID

5.2 プロファイル変更

<table>
<tr><td>API 名称</td><td>プロファイル変更</td></tr>
<tr><td>処理概要</td><td>HCT に登録済みの TCP/TDP/TSP を変更します。
各取引プロファイル内の ID を指定することで、書き換え対象を選択します。データフォーマット内に呼び出し側で紐づける ID を記載することで TCP と TDP/TSP を紐づけすることができます。</td></tr>
<tr><td>HTTP メソッド</td><td>PUT</td></tr>
<tr><td>URL パス</td><td>/hct/api/v1/profiles</td></tr>
<tr><td>要求パラメータ</td><td>メッセージボディ部に、JSON 形式で以下のパラメータを与えます。
{
"tcp": "省略(TCP データフォーマット)",
"tcp_temporary_registration_flag": false,
"tdp": "省略(TDP データフォーマット)",
"tdp_temporary_registration_flag": false,
"tsp": "省略(TSP データフォーマット)",
"tsp_temporary_registration_flag": false
}
<table>
<tr><th>属性</th><th>型</th><th>説明</th></tr>
<tr><td>tcp</td><td>JsonObject</td><td>(任意)変更する TCP</td></tr>
<tr><td>tcp_temporary_registration_flag</td><td>Boolean</td><td>(任意)変更する TCP の仮登録フラグ</td></tr>
<tr><td>tdp</td><td>JsonObject</td><td>(任意)変更する TDP</td></tr>
<tr><td>tdp_temporary_registration_flag</td><td>Boolean</td><td>(任意)変更する TDP の仮登録フラグ</td></tr>
<tr><td>tsp</td><td>JsonObject</td><td>(任意)変更する TSP</td></tr>
<tr><td>tsp_temporary_registration_flag</td><td>Boolean</td><td>(任意)変更する TSP の仮登録フラグ</td></tr>
</table></td></tr>
<tr><td>応答</td><td>2 章 HTTP プロトコルに準じます</td></tr>
<tr><td>応答ヘッダ</td><td>Content-Type: application/json</td></tr>
<tr><td>応答パラメータ</td><td>メッセージボディ部に、以下のパラメータを返します。
{
"tcp_id": "tcp01",
"tdp_id": "tdp01",
"tsp_id": "tsp01"
}
<table>
<tr><th>属性</th><th>型</th><th>説明</th></tr>
<tr><td>tcp_id</td><td>String</td><td>(任意)変更した TCP の ID</td></tr>
</table></td></tr>
</table>

	tdp_id	String	(任意)変更した TDP の ID
	tsp_id	String	(任意)変更した TSP の ID

5.3 プロファイル削除

<table>
<tr><td>API 名称</td><td>プロファイル削除</td></tr>
<tr><td>処理概要</td><td>HCT に登録済みの TCP を削除します。
TCP を削除することで、紐づいた TDP/TSP も削除されます。</td></tr>
<tr><td>HTTP メソッド</td><td>DELETE</td></tr>
<tr><td>リクエスト URL</td><td>/hct/api/v1/profiles/<tcp_id></td></tr>
<tr><td>要求パラメータ</td><td>属性 型 説明
tcp_id String (必須)削除したい TCP の ID</td></tr>
<tr><td>応答</td><td>2 章 HTTP プロトコルに準じます</td></tr>
<tr><td>応答ヘッダ</td><td>Content-Type: application/json</td></tr>
<tr><td>応答パラメータ</td><td>メッセージボディ部に、以下のパラメータを返します。
{
"tcp_id": "tcp01"
}
<table>
<tr><th>属性</th><th>型</th><th>説明</th></tr>
<tr><td>tcp_id</td><td>String</td><td>(必須)削除した TCP の ID</td></tr>
</table>
</td></tr>
</table>

5.4 プロファイル取得

<table>
<tr><td>API 名称</td><td>プロファイル取得</td></tr>
<tr><td>処理概要</td><td>各 ID で指定した TCP/TDP/TSP を取得します。ID を指定しない場合は、HCT に登録済みのすべての TCP/TDP/TSP を応答します。</td></tr>
<tr><td>HTTP メソッド</td><td>GET</td></tr>
<tr><td>リクエスト URL</td><td>/hct/api/v1/profiles?tcp_id=<tcp_id> or tdp_id=<tdp_id> or tsp_id=<tsp_id></td></tr>
<tr><td>要求パラメータ</td><td>
<table>
<tr><th>属性</th><th>型</th><th>説明</th></tr>
<tr><td>tcp_id</td><td>String</td><td>(任意)取得したい TCP の ID</td></tr>
<tr><td>tdp_id</td><td>String</td><td>(任意)取得したい TDP の ID</td></tr>
<tr><td>tsp_id</td><td>String</td><td>(任意)取得したい TSP の ID</td></tr>
</table>
</td></tr>
<tr><td>応答</td><td>2 章 HTTP プロトコルに準じます</td></tr>
<tr><td>応答ヘッダ</td><td>Content-Type: application/json</td></tr>
<tr><td>応答パラメータ</td><td>メッセージボディ部に、以下のパラメータを返します。
{
"tcps": [
{
省略(TCP データフォーマット),
"tcp_temporary_registration_flag": false
},
...
],
"tdps": [
{
省略(TDP データフォーマット),
"tdp_temporary_registration_flag": false
},
...
],
"tsps": [
{
省略(TSP データフォーマット),
"tsp_temporary_registration_flag": false
},
...
]
}
<table>
<tr><th>属性</th><th>型</th><th>説明</th></tr>
<tr><td>tcp</td><td>JsonObject</td><td>(任意)取得した TCP</td></tr>
<tr><td>tcp_temporary_registration_flag</td><td>Boolean</td><td>(任意)取得した TCP の仮登録フラグ</td></tr>
</table>
</td></tr>
</table>

	tdp	JsonObject	(任意)取得した TDP	
	tcp_temporary_registration_flag	Boolean	(任意)取得した TDP の仮登録フラグ	
	tsp	JsonObject	(任意)取得した TSP	
	tcp_temporary_registration_flag	Boolean	(任意)取得した TSP の仮登録フラグ	

5.5 ECU/EAU からサンプルデータ取得

<table>
<tr><td>API 名称</td><td>ECU/EAU からサンプルデータ取得</td></tr>
<tr><td>処理概要</td><td>ecu_id および edu_id で指定された対象から dcr_number 数分だけサンプルデータを取得する。</td></tr>
<tr><td>HTTP メソッド</td><td>GET</td></tr>
<tr><td>リクエスト URL</td><td>/hct/api/v1/sample_data?ecu=<ecu_id>&edu=<edu_id>
&dcr_number=<dcr_number></td></tr>
<tr><td>要求パラメータ</td><td>
<table>
<tr><th>属性</th><th>型</th><th>説明</th></tr>
<tr><td>ecu_id</td><td>String</td><td>(必須)サンプル取得対象となる EDU を管理する ECU の ID</td></tr>
<tr><td>edu_id</td><td>String</td><td>(必須)サンプル取得対象となる EDU の ID</td></tr>
<tr><td>dcr_number</td><td>Int</td><td>(必須)取得したいサンプル DCR 数</td></tr>
</table>
</td></tr>
<tr><td>応答</td><td>2 章 HTTP プロトコルに準じます</td></tr>
<tr><td>応答ヘッダ</td><td>Content-Type: application/json</td></tr>
<tr><td>応答パラメータ</td><td>
メッセージボディ部に、以下のパラメータを返します。
{
dcrs:[
{
"dcr": [{"気温":"12.3"}, {"相対湿度":"60"}]
},
{
...
}
...
]
}
<table>
<tr><th>属性</th><th>型</th><th>説明</th></tr>
<tr><td>dcr</td><td>String[]</td><td>(必須)取得したサンプルデータ</td></tr>
<tr><td>dcrs</td><td>JsonObject[]</td><td>(必須)取得したサンプルデータのリスト</td></tr>
</table>
</td></tr>
</table>

5.6 サンプルデータ登録

<table>
<tr><td>API 名称</td><td>サンプルデータ登録</td></tr>
<tr><td>処理概要</td><td>tcp_id で指定された取引におけるサンプルデータを登録します。サンプルデータは共通データ辞書の DCM で登録されます。ひとつの tcp_id に対してサンプルはひとつの dcr として持ちます。再登録で上書きされます。</td></tr>
<tr><td>HTTP メソッド</td><td>POST</td></tr>
<tr><td>リクエスト URL</td><td>/hct/api/v1/sample_data</td></tr>
<tr><td>要求パラメータ</td><td>メッセージボディ部に、JSON 形式で以下のパラメータを与えます。
{
"sample_data": [
{
"tcp_id": "tcp01",
"dcr": [{"気温":"12.3"}, {"相対湿度":"60"}]},
{
"tcp_id": "tcp02",
"dcr": [{"気温":"12.5"}, {"相対湿度":"40"}]},
…
]
}
<table>
<tr><th>属性</th><th>型</th><th>説明</th></tr>
<tr><td>tcp_id</td><td>String</td><td>(必須)サンプル登録する TCP の ID</td></tr>
<tr><td>dcr</td><td>String[]</td><td>(必須)サンプルデータのリスト</td></tr>
</table>
</td></tr>
<tr><td>応答</td><td>2 章 HTTP プロトコルに準じます</td></tr>
<tr><td>応答ヘッダ</td><td>Content-Type: application/json</td></tr>
<tr><td>応答パラメータ</td><td>メッセージボディ部に、以下のパラメータを返します。
{
"sample_data_id": [
"sample01",
"sample02",
…
]
}
<table>
<tr><th>属性</th><th>型</th><th>説明</th></tr>
<tr><td>sample_data_id</td><td>String[]</td><td>(必須)登録したサンプルデータの ID のリスト</td></tr>
</table>
</td></tr>
</table>

5.7 サンプルデータ削除

<table>
<tr><td>API 名称</td><td>サンプルデータ削除</td></tr>
<tr><td>処理概要</td><td>tcp_id で指定された取引におけるサンプルデータを削除します。</td></tr>
<tr><td>HTTP メソッド</td><td>DELETE</td></tr>
<tr><td>リクエスト URL</td><td>/hct/api/v1/sample_data?tcp=<tcp_id></td></tr>
<tr><td>要求パラメータ</td><td><table><tr><th>属性</th><th>型</th><th>説明</th></tr><tr><td>tcp_id</td><td>String</td><td>(必須)サンプルを削除する TCP の ID</td></tr></table></td></tr>
<tr><td>応答</td><td>2 章 HTTP プロトコルに準じます</td></tr>
<tr><td>応答ヘッダ</td><td>Content-Type: application/json</td></tr>
<tr><td>応答パラメータ</td><td>メッセージボディ部に、以下のパラメータを返します。
{
"sample_data_id": "sample01"
}<table><tr><th>属性</th><th>型</th><th>説明</th></tr><tr><td>sample_data_id</td><td>String</td><td>(必須)削除したサンプルデータの ID</td></tr></table></td></tr>
</table>

5.8 サンプルデータ取得

<table>
<tr><td>API 名称</td><td>サンプルデータ取得</td></tr>
<tr><td>処理概要</td><td>tcp_id で指定された取引におけるサンプルデータを取得します。</td></tr>
<tr><td>HTTP メソッド</td><td>GET</td></tr>
<tr><td>リクエスト URL</td><td>/hct/api/v1/sample_data?tcp=<tcp_id></td></tr>
<tr><td>要求パラメータ</td><td><table><tr><th>属性</th><th>型</th><th>説明</th></tr><tr><td>tcp_id</td><td>String</td><td>(必須)サンプルを取得する TCP の ID</td></tr></table></td></tr>
<tr><td>応答</td><td>2 章 HTTP プロトコルに準じます</td></tr>
<tr><td>応答ヘッダ</td><td>Content-Type: application/json</td></tr>
<tr><td>応答パラメータ</td><td>メッセージボディ部に、以下のパラメータを返します。
{
"tcp_id": "tcp01",
"dcr": [{"気温":"12.3"}, {"相対湿度":"60"}]},
}<table><tr><th>属性</th><th>型</th><th>説明</th></tr><tr><td>tcp_id</td><td>String</td><td>(必須)サンプルデータが紐づく TCP の ID</td></tr><tr><td>dcr[]</td><td>String[]</td><td>(必須)サンプルデータのリスト</td></tr></table></td></tr>
</table>

5.9 他サイト HCT からサンプルデータ取得

<table>
<tr><td>API 名称</td><td>他サイト HCT からサンプルデータ取得</td></tr>
<tr><td>処理概要</td><td>tcp_id で指定された取引において、hct_id で指定したサンプルデータを取得します。</td></tr>
<tr><td>HTTP メソッド</td><td>GET</td></tr>
<tr><td>リクエスト URL</td><td>/hct/api/v1/sample_data?
tcp=<tcp_id>&hct=<hct_id></td></tr>
<tr><td>要求パラメータ</td><td><table><tr><th>属性</th><th>型</th><th>説明</th></tr><tr><td>tcp_id</td><td>String</td><td>(必須)取得したいサンプルをもつ TCP の ID</td></tr><tr><td>hct_id</td><td>String</td><td>(任意)取得したいサンプルをもつ HCT の ID</td></tr></table></td></tr>
<tr><td>応答</td><td>2 章 HTTP プロトコルに準じます</td></tr>
<tr><td>応答ヘッダ</td><td>Content-Type: application/json</td></tr>
<tr><td>応答パラメータ</td><td>メッセージボディ部に、以下のパラメータを返します。
{
"tcp_id": "tcp01",
"dcr": [{"気温":"12.3"}, {"相対湿度":"60"}]
}<table><tr><th>属性</th><th>型</th><th>説明</th></tr><tr><td>tcp_id</td><td>String</td><td>(必須)取得したサンプルをもつ TCP の ID</td></tr><tr><td>dcr[]</td><td>String[]</td><td>(必須)サンプルデータのリスト</td></tr></table></td></tr>
</table>

5.10他サイト HCT の提供可能データ取得

<table>
<tr><td>API 名称</td><td>他サイト HCT の提供可能データ取得</td></tr>
<tr><td>処理概要</td><td>hct_id で指定した HCT 内において、プッシュ型取引にて提供できるデータを含むプロファイル(TCP のうち、trigger が提供者で user_id が null のもの)を取得します。</td></tr>
<tr><td>HTTP メソッド</td><td>GET</td></tr>
<tr><td>リクエスト URL</td><td>/hct/api/v1/available_data?hct=<hct_id></td></tr>
<tr><td>要求パラメータ</td><td>
<table>
<tr><th>属性</th><th>型</th><th>説明</th></tr>
<tr><td>hct_id</td><td>String</td><td>(必須)問い合わせ先 HCT の ID</td></tr>
</table>
</td></tr>
<tr><td>応答</td><td>2 章 HTTP プロトコルに準じます</td></tr>
<tr><td>応答ヘッダ</td><td>Content-Type: application/json</td></tr>
<tr><td>応答パラメータ</td><td>メッセージボディ部に、以下のパラメータを返します。
{
"tcps": [
{省略},
{省略},
...
]
"hct_id": "hct01"
}
<table>
<tr><th>属性</th><th>型</th><th>説明</th></tr>
<tr><td>tcps</td><td>String[]</td><td>(必須)提供可能なデータに関する TCP のリスト</td></tr>
<tr><td>hct_id</td><td>String</td><td>(必須)問い合わせ先 HCT の ID</td></tr>
</table>
</td></tr>
</table>

5.11HCM リクエスト登録

<table>
<tr><td>API 名称</td><td>HCM リクエスト登録</td></tr>
<tr><td>処理概要</td><td>他サイトの HCM にリクエストを送信します。</td></tr>
<tr><td>HTTP メソッド</td><td>POST</td></tr>
<tr><td>リクエスト URL</td><td>/hct/api/v1/hcm_requests</td></tr>
<tr><td>要求パラメータ</td><td>メッセージボディ部に、JSON 形式で以下のパラメータを与えます。
{
"source_hct_id":"hct01",
"destination _hct_id":"hct02",
"request_parameter":(日立製作所様指定)
}
<table><tr><th>属性</th><th>型</th><th>説明</th></tr><tr><td>source_hct_id</td><td>String</td><td>(必須)送信元 HCT の ID</td></tr><tr><td>destination_hct_id</td><td>String</td><td>(必須)送信先 HCT の ID</td></tr><tr><td>request_parameter</td><td>JsonObject</td><td>(必須)指定パラメータ</td></tr></table></td></tr>
<tr><td>応答</td><td>2 章 HTTP プロトコルに準じます</td></tr>
<tr><td>応答ヘッダ</td><td>Content-Type: application/json</td></tr>
<tr><td>応答パラメータ</td><td>なし</td></tr>
</table>

5.12HCM リクエスト取得

<table>
<tr><td>API 名称</td><td>HCM リクエスト取得</td></tr>
<tr><td>HTTP メソッド</td><td>GET</td></tr>
<tr><td>リクエスト URL</td><td>/hct/api/v1/hcm_requests</td></tr>
<tr><td>要求パラメータ</td><td>なし</td></tr>
<tr><td>処理概要</td><td>他サイトの HCM からのリクエストを取得します。</td></tr>
<tr><td>応答</td><td>2 章 HTTP プロトコルに準じます</td></tr>
<tr><td>応答ヘッダ</td><td>Content-Type: application/json</td></tr>
<tr><td>応答パラメータ</td><td>メッセージボディ部に、以下のパラメータを返します。
[
{
"source_hct_id":"hct01",
"request_parameter":(日立製作所様指定)
},
...
]
<table>
<tr><th>属性</th><th>型</th><th>説明</th></tr>
<tr><td>source_hct_id</td><td>String</td><td>(必須)送信元 HCT の ID</td></tr>
<tr><td>request_parameter</td><td>JsonObject</td><td>(必須)指定パラメータ</td></tr>
</table></td></tr>
</table>

6. ECU-HCT 間 API リファレンス

ECU-HCT 間の API リファレンス仕様として、以下を規定します。

・HCT への ECU 登録
・HCT への EDU 登録
・HCT から ECU 登録抹消
・HCT から EDU 登録抹消
・取引 ID の検索・取得
・データ・要求の送信
・データ・要求の検索・取得
・サンプルデータの送信
・HCS への HCT 登録

6.1 HCT への ECU 登録

<table>
<tr><td>API 名称</td><td>HCT への ECU 登録</td></tr>
<tr><td>処理概要</td><td>HCT へ新規 ECU を登録します。</td></tr>
<tr><td>HTTP メソッド</td><td>POST</td></tr>
<tr><td>リクエスト URL</td><td>/hct/api/v1/ecus</td></tr>
<tr><td>要求パラメータ</td><td>メッセージボディ部に、JSON 形式で以下のパラメータを与えます。
{
"ecu_name": "FANUC_ROBOECU",
}
<table>
<tr><th>属性</th><th>型</th><th>説明</th></tr>
<tr><td>ecu_name</td><td>String</td><td>(必須)登録する ECU 名</td></tr>
</table>
</td></tr>
<tr><td>応答</td><td>2 章 HTTP プロトコルに準じます</td></tr>
<tr><td>応答ヘッダ</td><td>Content-Type: application/json</td></tr>
<tr><td>応答パラメータ</td><td>メッセージボディ部に、以下のパラメータを返します。
{
"ecu_id": "ecu01",
"ecu_key": "12345678"
}
<table>
<tr><th>属性</th><th>型</th><th>説明</th></tr>
<tr><td>ecu_id</td><td>String</td><td>(必須)登録した ECU の割当 ID</td></tr>
<tr><td>ecu_key</td><td>String</td><td>(必須)登録した ECU に対する key コード(MD5)</td></tr>
</table>
</td></tr>
</table>

6.2 HCT への EDU 登録

<table>
<tr><td>API 名称</td><td>HCT への EDU 登録</td></tr>
<tr><td>処理概要</td><td>HCT へ新規 EDU を登録します。
その際､その EDU で対応可能な PCM/ECM リストもあわせて登録します。
（現状、新規登録のみ可能。PCM/ECM リストの編集は不可）</td></tr>
<tr><td>HTTP メソッド</td><td>POST</td></tr>
<tr><td>リクエスト URL</td><td>/hct/api/v1/edus</td></tr>
<tr><td>要求パラメータ</td><td>メッセージボディ部に、JSON 形式で以下のパラメータを与えます。
{
"edu_name": "FANUC_DRILL",
"pcm_id": ["pcm01", "pcm02"],
"ecm_id": ["ecm01", "ecm02"]
}
<table>
<tr><th>属性</th><th>型</th><th>説明</th></tr>
<tr><td>edu_name</td><td>String</td><td>(必須)登録する EDU 名</td></tr>
<tr><td>pcm_id</td><td>String</td><td>(任意)動作可能な PCMID のリスト</td></tr>
<tr><td>ecm_id</td><td>String</td><td>(任意)動作可能な ECMID のリスト</td></tr>
</table></td></tr>
<tr><td>応答</td><td>2 章 HTTP プロトコルに準じます</td></tr>
<tr><td>応答ヘッダ</td><td>Content-Type: application/json</td></tr>
<tr><td>応答パラメータ</td><td>メッセージボディ部に、以下のパラメータを返します。
{
"edu_id": "edu01",
}
<table>
<tr><th>属性</th><th>型</th><th>説明</th></tr>
<tr><td>edu_id</td><td>String</td><td>(必須)登録した EDU の割当 ID</td></tr>
</table></td></tr>
</table>

6.3 HCT から ECU の登録抹消

<table>
<tr><td>API 名称</td><td>HCT から ECU の登録抹消</td></tr>
<tr><td>処理概要</td><td>HCT から ecu_id で指定した ECU を抹消します。</td></tr>
<tr><td>HTTP メソッド</td><td>DELETE</td></tr>
<tr><td>リクエスト URL</td><td>/hct/api/v1/ecus</td></tr>
<tr><td>要求パラメータ</td><td>メッセージボディ部に、JSON 形式で以下のパラメータを与えます。
{
"ecu_id": "ecu01",
"ecu_key": "12345678"
}
<table>
<tr><th>属性</th><th>型</th><th>説明</th></tr>
<tr><td>ecu_id</td><td>String</td><td>(必須)登録した ECU の割当 ID</td></tr>
<tr><td>ecu_key</td><td>String</td><td>(必須)登録した ECU に対する key コード(MD5)</td></tr>
</table></td></tr>
<tr><td>応答</td><td>2 章 HTTP プロトコルに準じます</td></tr>
<tr><td>応答ヘッダ</td><td>Content-Type: application/json</td></tr>
<tr><td>応答パラメータ</td><td>メッセージボディ部に、以下のパラメータを返します。
{
"ecu_id: "ecu01"
}
<table>
<tr><th>属性</th><th>型</th><th>説明</th></tr>
<tr><td>ecu_id</td><td>String</td><td>(必須)抹消した ECU の割当 ID</td></tr>
</table></td></tr>
</table>

6.4 HCT から EDU の登録抹消

<table>
<tr><td>API 名称</td><td>HCT から EDU の登録抹消</td></tr>
<tr><td>処理概要</td><td>HCT から edu_id で指定した EDU を抹消します。</td></tr>
<tr><td>HTTP メソッド</td><td>DELETE</td></tr>
<tr><td>リクエスト URL</td><td>/hct/api/v1/edus</td></tr>
<tr><td>要求パラメータ</td><td>メッセージボディ部に、JSON 形式で以下のパラメータを与えます。
{
"edu_id": "edu01"
}
<table>
<tr><th>属性</th><th>型</th><th>説明</th></tr>
<tr><td>edu_id</td><td>String</td><td>（必須）登録した EDU の割当 ID</td></tr>
</table>
</td></tr>
<tr><td>応答</td><td>2 章 HTTP プロトコルに準じます</td></tr>
<tr><td>応答ヘッダ</td><td>Content-Type: application/json</td></tr>
<tr><td>応答パラメータ</td><td>メッセージボディ部に、以下のパラメータを返します。
{
"edu_id": "edu01"
}
<table>
<tr><th>属性</th><th>型</th><th>説明</th></tr>
<tr><td>edu_id</td><td>String</td><td>（必須）抹消した EDU の割当 ID</td></tr>
</table>
</td></tr>
</table>

6.5 取引契約プロファイル ID の検索・取得

<table>
<tr><td>API 名称</td><td>取引契約プロファイル ID の検索・取得</td></tr>
<tr><td>処理概要</td><td>URL の引数に指定された EDU の組に合致する TCP の ID のリストを返します。
source_edu_id と destination_edu_id が省略された場合は、ecu_id に紐づく ECU が関連するすべての tcp_id を応答します。</td></tr>
<tr><td>HTTP メソッド</td><td>POST</td></tr>
<tr><td>リクエスト URL</td><td>/hct/api/v1/tcps</td></tr>
<tr><td>要求パラメータ</td><td>メッセージボディ部に、JSON 形式で以下のパラメータを与えます。
{
"ecu_id": "ecu01",
"ecu_key": "12345678",
"source_edu_id":"edu01",
"destination_edu_id":"edu02"
}
<table>
<tr><th>属性</th><th>型</th><th>説明</th></tr>
<tr><td>ecu_id</td><td>String</td><td>(必須)登録した ECU の割当 ID</td></tr>
<tr><td>ecu_key</td><td>String</td><td>(必須)登録した ECU に対する key コード(MD5)</td></tr>
<tr><td>source_edu_id</td><td>String</td><td>(任意)提供元 EDU の ID</td></tr>
<tr><td>destination_edu_id</td><td>String</td><td>(任意)送信先 EDU の ID</td></tr>
</table></td></tr>
<tr><td>応答</td><td>2 章 HTTP プロトコルに準じます</td></tr>
<tr><td>応答ヘッダ</td><td>Content-Type: application/json</td></tr>
<tr><td>応答パラメータ</td><td>メッセージボディ部に、以下のパラメータを返します。
{
"tcp_id": ["tcp01", "tcp02", "tcp04"]
}
<table>
<tr><th>属性</th><th>型</th><th>説明</th></tr>
<tr><td>tcp_id</td><td>String</td><td>TCP の ID</td></tr>
</table></td></tr>
</table>

6.6 データ・要求の送信

<table>
<tr><td>API 名称</td><td>データ・要求の送信</td></tr>
<tr><td>処理概要</td><td>tcp_id に紐づいた取引における送信対象へ、message を送信します。連結データ送信の場合に、連結すべき message がそろわなかった場合は、それらの message を破棄します。（タイムアウトを 10 分に設定しています）</td></tr>
<tr><td>HTTP メソッド</td><td>POST</td></tr>
<tr><td>リクエスト URL</td><td>/hct/api/v1/tcps/<tcp_id></td></tr>
<tr><td>要求パラメータ</td><td>メッセージボディ部に、JSON 形式で以下のパラメータを与えます。
{
"message": [{"温度":"12.3"}, {"湿度":"60"}]
}

<table><tr><td>属性</td><td>型</td><td>説明</td></tr><tr><td>tcp_id</td><td>String</td><td>送信データの TCP の ID</td></tr><tr><td>message[]</td><td>JsonObject[]</td><td>送信したい実データ。
null にすると送信先 EDU に対する送信要求としてとらえられる。</td></tr></table></td></tr>
<tr><td>応答</td><td>2 章 HTTP プロトコルに準じます</td></tr>
<tr><td>応答ヘッダ</td><td>Content-Type: application/json</td></tr>
<tr><td>応答パラメータ</td><td>メッセージボディ部に、以下のパラメータを返します。
{
"error_id": "0"
}
<table><tr><td>属性</td><td>型</td><td>説明</td></tr><tr><td>error_id</td><td>String</td><td>エラーID。0 は正常。その他未定義。</td></tr></table></td></tr>
</table>

6.7 データ・要求の検索・取得

<table>
<tr><td>API 名称</td><td>データ・要求の検索・取得</td></tr>
<tr><td>処理概要</td><td>ecu_id で指定した宛先の message を応答する。</td></tr>
<tr><td>HTTP メソッド</td><td>POST</td></tr>
<tr><td>リクエスト URL</td><td>/hct/api/v1/message</td></tr>
<tr><td>要求パラメータ</td><td>メッセージボディ部に、JSON 形式で以下のパラメータを与えます。
{
"ecu_id": "ecu01",
"ecu_key": "12345678"
}
<table>
<tr><td>属性</td><td>型</td><td>説明</td></tr>
<tr><td>ecu_id</td><td>String</td><td>(必須)登録した ECU の割当 ID</td></tr>
<tr><td>ecu_key</td><td>String</td><td>(必須)登録した ECU に対する key コード(MD5)</td></tr>
</table>
</td></tr>
<tr><td>応答</td><td>2 章 HTTP プロトコルに準じます</td></tr>
<tr><td>応答ヘッダ</td><td>Content-Type: application/json</td></tr>
<tr><td>応答パラメータ</td><td>メッセージボディ部に、以下のパラメータを返します。
{
"source_hct_id": "hct01",
"destination_ecu_id": "ecu01",
"response": [
{
"destination_edu_id": "edu01",
"message": [
{"気温":"12.3"},
{"湿度":"60"}
]
},
{
"destination_edu_id": "edu02",
"message": [
{"気温":"12.5"},
{"湿度":"68"}
]
},
...
]</td></tr>
</table>

}

属性	型	説明
source_hct_id	String	(必須)提供元 HCT の ID
destination_ecu_id	String	(必須)送信先 ECU の ID
destination_edu_id	String	(任意)送信先 EDU の ID
response	JsonObject[]	(任意)応答内容
message	JsonObject[]	(任意)メッセージ内容

6.8 サンプルデータの送信

<table>
<tr><td>API 名称</td><td>サンプルデータの送信</td></tr>
<tr><td>処理概要</td><td>HCM からのサンプルデータ要求に従って、ECU/EAU がサンプルデータを送信する。</td></tr>
<tr><td>HTTP メソッド</td><td>POST</td></tr>
<tr><td>リクエスト URL</td><td>/hct/api/v1/tcps/samples</td></tr>
<tr><td>要求パラメータ</td><td>メッセージボディ部に、JSON 形式で以下のパラメータを与えます。
{
"source_ecu_id": "ecu01",
"source_edu_id": "edu01",
"message": [{"温度":"12.3"}, {"湿度":"60"}]
}

<table>
<tr><th>属性</th><th>型</th><th>説明</th></tr>
<tr><td>source_ecu_id</td><td>String</td><td>(必須)提供元 ECU の ID</td></tr>
<tr><td>source_edu_id</td><td>String</td><td>(任意)提供元 EDU の ID</td></tr>
<tr><td>message[]</td><td>JsonObject[]</td><td>送信したい実データ。
null にすると送信先 EDU に対する送信要求としてとらえられる。</td></tr>
</table>
</td></tr>
<tr><td>応答</td><td>2 章 HTTP プロトコルに準じます</td></tr>
<tr><td>応答ヘッダ</td><td>Content-Type: application/json</td></tr>
<tr><td>応答パラメータ</td><td>メッセージボディ部に、以下のパラメータを返します。
{
"error_id": "0"
}
<table>
<tr><th>属性</th><th>型</th><th>説明</th></tr>
<tr><td>error_id</td><td>String</td><td>エラーID。0 は正常。その他未定義。</td></tr>
</table>
</td></tr>
</table>

6.9 HCS への HCT 登録

<table>
<tr><td>API 名称</td><td>HCS への HCT 登録</td></tr>
<tr><td>処理概要</td><td>HCS へ新規 HCT を登録します。再呼び出しした場合は、現在の HCT-ID が応答され、再登録されることはありません。</td></tr>
<tr><td>HTTP メソッド</td><td>POST</td></tr>
<tr><td>リクエスト URL</td><td>/hct/api/v1/hcts</td></tr>
<tr><td>要求パラメータ</td><td>メッセージボディ部に、JSON 形式で以下のパラメータを与えます。
{
"hct_name": "MEL_HCT"
}
属性 型 説明
hct_name String (必須)登録する HCT 名</td></tr>
<tr><td>応答</td><td>2 章 HTTP プロトコルに準じます</td></tr>
<tr><td>応答ヘッダ</td><td>Content-Type: application/json</td></tr>
<tr><td>応答パラメータ</td><td>メッセージボディ部に、以下のパラメータを返します。
{
"hct_id": "hct01",
"hct_key": "12345678"
}
属性 型 説明
hct_id String (必須)登録した HCT の割当 ID
hct_key String (必須)登録した HCT に対する key コード(MD5)</td></tr>
</table>

6.10HDS パスワード登録

<table>
<tr><td>API 名称</td><td>HDS パスワード登録</td></tr>
<tr><td>処理概要</td><td>HDS に接続する際のパスワードを登録します。1 つの HCT に対して 1 つの HDS パスワードを登録します。すでに登録された状態で登録すると、上書きされます。これはデモのみの仕様です。</td></tr>
<tr><td>HTTP メソッド</td><td>POST</td></tr>
<tr><td>リクエスト URL</td><td>/hct/api/v1/hds_password</td></tr>
<tr><td>要求パラメータ</td><td>メッセージボディ部に、JSON 形式で以下のパラメータを与えます。
{
"hds_password": "password"
}
<table><tr><td>属性</td><td>型</td><td>説明</td></tr><tr><td>hds_password</td><td>String</td><td>(必須)登録するパスワード</td></tr></table></td></tr>
<tr><td>応答</td><td>2 章 HTTP プロトコルに準じます</td></tr>
<tr><td>応答ヘッダ</td><td>Content-Type: application/json</td></tr>
<tr><td>応答パラメータ</td><td>なし</td></tr>
</table>

7. HCT-HCS 間 API リファレンス

HCT-HCS 間の API リファレンス仕様として、以下を規定します。

・HCS への HCT 登録
・HCS からの HCT 登録抹消
・データ・要求の送信
・データ・要求の検索・取得

7.1 HCS への HCT 登録

<table>
<tr><td>API 名称</td><td>HCS への HCT 登録</td></tr>
<tr><td>処理概要</td><td>HCS へ新規 HCT を登録します。</td></tr>
<tr><td>HTTP メソッド</td><td>POST</td></tr>
<tr><td>リクエスト URL</td><td>/hcs/api/v1/hcts</td></tr>
<tr><td>要求パラメータ</td><td>メッセージボディ部に、JSON 形式で以下のパラメータを与えます。
{
"hct_name": "MEL_HCT",
}
<table>
<tr><th>属性</th><th>型</th><th>説明</th></tr>
<tr><td>hct_name</td><td>String</td><td>(必須)登録する HCT 名</td></tr>
</table>
</td></tr>
<tr><td>応答</td><td>2 章 HTTP プロトコルに準じます</td></tr>
<tr><td>応答ヘッダ</td><td>Content-Type: application/json</td></tr>
<tr><td>応答パラメータ</td><td>メッセージボディ部に、以下のパラメータを返します。
{
"hct_id": "hct01",
"hct_key": "12345678"
}
<table>
<tr><th>属性</th><th>型</th><th>説明</th></tr>
<tr><td>hct_id</td><td>String</td><td>(必須)登録した HCT の割当 ID</td></tr>
<tr><td>hct_key</td><td>String</td><td>(必須)登録した HCT に対する key コード(MD5)</td></tr>
</table>
</td></tr>
</table>

7.2 HCS から HCT の登録抹消

<table>
<tr><td>API 名称</td><td>HCS から HCT の登録抹消</td></tr>
<tr><td>処理概要</td><td>HCS から hct_id で指定した HCT を抹消します。</td></tr>
<tr><td>HTTP メソッド</td><td>DELETE</td></tr>
<tr><td>リクエスト URL</td><td>/hcs/api/v1/hcts</td></tr>
<tr><td>要求パラメータ</td><td>メッセージボディ部に、JSON 形式で以下のパラメータを与えます。
{
"hct_id": "hct01"
}
<table>
<tr><th>属性</th><th>型</th><th>説明</th></tr>
<tr><td>hct_id</td><td>String</td><td>（必須）登録した HCT の割当 ID</td></tr>
</table>
</td></tr>
<tr><td>応答</td><td>2 章 HTTP プロトコルに準じます</td></tr>
<tr><td>応答ヘッダ</td><td>Content-Type: application/json</td></tr>
<tr><td>応答パラメータ</td><td>メッセージボディ部に、以下のパラメータを返します。
{
"hct_id": "edu01"
}
<table>
<tr><th>属性</th><th>型</th><th>説明</th></tr>
<tr><td>hct_id</td><td>String</td><td>（必須）抹消した HCT の割当 ID</td></tr>
</table>
</td></tr>
</table>

7.3 データ・要求の送信

<table>
<tr><td>API 名称</td><td>データ・要求の送信</td></tr>
<tr><td>処理概要</td><td>tcp の情報と dcr[]の情報を紐づけて保存する。</td></tr>
<tr><td>HTTP メソッド</td><td>POST</td></tr>
<tr><td>リクエスト URL</td><td>/hcs/api/v1/message</td></tr>
<tr><td>要求パラメータ</td><td>メッセージボディ部に、JSON 形式で以下のパラメータを与えます。
{
"tcp": {省略},
"dcr": [{"気温":"12.3"}, {"相対湿度":"60"}]
}
<table>
<tr><th>属性</th><th>型</th><th>説明</th></tr>
<tr><td>tcp</td><td>JsonObject</td><td>(必須)TCP そのもの</td></tr>
<tr><td>dcr[]</td><td>JsonObject[]</td><td>(必須)送信するデータレコード</td></tr>
</table></td></tr>
<tr><td>応答</td><td>2 章 HTTP プロトコルに準じます</td></tr>
<tr><td>応答ヘッダ</td><td>Content-Type: application/json</td></tr>
<tr><td>応答パラメータ</td><td>メッセージボディ部に、以下のパラメータを返します。
{
"error_id": "0"
}
<table>
<tr><th>属性</th><th>型</th><th>説明</th></tr>
<tr><td>error_id</td><td>String</td><td>エラーID。0 は正常。その他未定義。</td></tr>
</table></td></tr>
</table>

7.4 データ・要求の検索・取得

<table>
<tr><td>API 名称</td><td>データ・要求の検索・取得</td></tr>
<tr><td>処理概要</td><td>destination_hct_id で指定した宛先の message を応答する。</td></tr>
<tr><td>HTTP メソッド</td><td>POST</td></tr>
<tr><td>リクエスト URL</td><td>/hcs/api/v1/message</td></tr>
<tr><td>要求パラメータ</td><td>メッセージボディ部に、JSON 形式で以下のパラメータを与えます。
{
"hct_id": "hct01",
"hct_key": "12345678",
"destination_hct_id": "hct02",
"destination_ecu_id": "ecu02"
}
<table>
<tr><th>属性</th><th>型</th><th>説明</th></tr>
<tr><td>hct_id</td><td>String</td><td>(必須)登録した HCT の割当 ID</td></tr>
<tr><td>hct_key</td><td>String</td><td>(必須)登録した HCT に対する key コード(MD5)</td></tr>
<tr><td>destination_hct_id</td><td>String</td><td>(必須)送信先 HCT の ID</td></tr>
<tr><td>destination_ecu_id</td><td>String</td><td>(必須)送信先 ECU の ID</td></tr>
</table></td></tr>
<tr><td>応答</td><td>2 章 HTTP プロトコルに準じます</td></tr>
<tr><td>応答ヘッダ</td><td>Content-Type: application/json</td></tr>
<tr><td>応答パラメータ</td><td>メッセージボディ部に、以下のパラメータを返します。
{
"source_hcs_id": "hcs01",
"destination_hct_id: "hct02",
"destination_ecu_id: "ecu02",
"message": [
{
"tcp_id": "tcp01",
"dcr": [{"気温":"12.3"}, {"相対湿度":"60"}]
},
...
]
}
<table>
<tr><th>属性</th><th>型</th><th>説明</th></tr>
<tr><td>source_hcs_id</td><td>String</td><td>(必須)提供元 HCS の</td></tr>
</table></td></tr>
</table>

			ID	
	destination_hct_id	String	(必須)送信先 HCT の ID	
	destination_ecu_id	String	(任意)送信先 ECU の ID	
	message	JsonObject[]	(任意)メッセージ内容	
	tcp_id	String	(必須)TCP の ID	
	dcr[]	String[]	(必須)送信データレコード	

8. 付録

8.1 動作例

本システム（HCT/HCS を含む本デモにかかわるシステムのすべて）において、デモを実施する場合に、「取引契約フェーズ」と「取引実施フェーズ」の大きく 2 つのフェーズがあります。
ここでは、HCT および HCS を用いて通信をするための「通信準備」、「取引契約フェーズ」、「取引実施フェーズ」の動作例を記載します。本動作例の中で示す ID 等は、最終版の ID 割り当て番号とは異なり、説明のために便宜的に使用している ID となりますので、ご注意ください。

8.1.1 通信準備

ここでは、HCT を実験用の PC へインストールしてからの準備について記載します。

(1) はじめに、HCT を HCS へ登録します。登録は、HCT に対して以下のようなリクエストを行います。

リクエスト

```
POST /hct/api/v1/hcts HTTP/1.1
Host: example.org
Content-Type: application/json; charset=UTF-8
Authorization: Basic Q0lPRjpwY2lvZg==

{
  "hct_name": "SAMPLE_HCT"
}
```

レスポンス

```
HTTP/1.1 200 OK
Date: Thu, 18 Oct 12:34:56 JST
Content-Type: application/json; charset=UTF-8
Content-Length: xxx
Connection: close
```

```
{
    "hct_id": "10000000000000000000000000000031",
    "hct_key": "sff4Bs6WY5K6a60G5HynJQ=="
}
```

(2) HCT の ID が発行されたら、それをユビキタス・ネットワーキング研究所の坂(tomoki.ban@ubin.jp)および日立製作所の伊藤様(tetsu.ito.fn@hitachi.com)にメールで連絡します。(この手順はデモのための暫定的な手順となります)

(3) 日立製作所様にて各サイトの HCT-ID とパスワードを HDS に手動で登録し、メールで連絡します。(この手順はデモのための暫定的な手順となります)

(4) HCT に対して HDS のパスワードを以下のリクエストで登録します。(この手順はデモのための暫定的な手順となります)

リクエスト

```
POST /hct/api/v1/hds_password HTTP/1.1
Host: example.org
Content-Type: application/json; charset=UTF-8
Authorization: Basic Q0lPRjpwY2lvZg==
Cache-Control: no-cache

{
  "hds_password": "(日立製作所様より連絡のあった文字列)"
}
```

レスポンス

```
HTTP/1.1 204 OK
Date: Thu, 18 Oct 12:34:56 JST
Content-Type: application/json; charset=UTF-8
Content-Length: xxx
Connection: close
```

これで HDS へのアクセスが可能となりました。

(5) 続いて、HCT に対して ECU(EAU)を登録します。

リクエスト

```
POST /hct/api/v1/ecus HTTP/1.1
Host: example.org
Content-Type: application/json; charset=UTF-8
Authorization: Basic Q0lPRjpwY2lvZg==

{
```

```
  "ecu_name": "SAMPLE_ECU"
}
```

レスポンス

```
HTTP/1.1 200 OK
Date: Thu, 18 Oct 12:34:56 JST
Content-Type: application/json; charset=UTF-8
Content-Length: xxx
Connection: close

{
    "ecu_id": "20030000000000000000000000000003",
    "ecu_key": "cMUy6TZqXh5hM+hOB4+OoQ=="
}
```

(6) 続いて、HCT に対して EDU を登録します。

リクエスト

```
POST /hct/api/v1/edus HTTP/1.1
Host: example.org
Content-Type: application/json; charset=UTF-8
Authorization: Basic Q0lPRjpwY2lvZg==

{
"edu_name": "SAMPLE_EDU",
"pcm_id": ["pcm01", "pcm02"],
"ecm_id": ["ecm01", "ecm02"]
}
```

レスポンス

```
HTTP/1.1 200 OK
Date: Thu, 18 Oct 12:34:56 JST
Content-Type: application/json; charset=UTF-8
Content-Length: xxx
Connection: close

{{
    "edu_id": "20040000000000000000000000000002"
}
```

これで通信する(取引契約フェーズに移行する)準備が完了しました。ECU(EAU)および EDU が複数ある場合は、上記手順を繰り返して登録してください。また、ここで入手した ECU KEY は、API によって必要になりますので ECU 内に保持する必要があります。

8.1.2 取引契約フェーズ

ここでは、共通辞書（SDD)、個別辞書（ADD)の登録と照会、辞書変換マップ（DTM）の登録、取引プロファイル(TCP, TDP, TSP)の登録を行います。
具体例については、「サブシステムプロトタイプ実装仕様（HDS/HCM 編)」[2] を参照してください。
フロー中において、HCT は、以下の API を提供します。

・プロファイル登録/変更/削除/取得
・サンプルデータ登録/削除
・他サイト HCT からサンプルデータ取得
・他サイト HCT の提供可能データ取得
・HCM リクエスト登録
・HCM リクエスト取得

12/7(金)現在においては、HDS が公開されておりませんので、取引プロファイルを HCM(画面)経由で登録できません。そのため、以下のような手順で通信を行う 2 台の HCT に対して取引プロファイルを登録する必要があります。ここでは、HCT1 と HCT2 を通信させることを前提にセットアップします。

(1) はじめに HCT1 に対して TCP/TDP/TSP を登録します。なお、各 ID(HCT, UNIT, EDU, TDP, TSP)の設定は、実際の環境に合わせる必要がありますので、注意してください。

リクエスト

```
POST /hct/api/v1/profiles HTTP/1.1
Host: example.org(HCT1 のアドレスとしてください)
Content-Type: application/json; charset=UTF-8
Authorization: Basic Q0lPRjpwY2lvZg==

{
  "tcp": {
    "tcp_id": "20000000000000000000000000000001",
    "tcp_name": "TCP サンプル",
    "owner_id": "owner01",
    "owner_name": "株式会社テスト",
    "created_date": "2018/10/29",
    "author": "田中太郎",

    "user_name": "テスト工務店",
```

```
    "data_dictionary_id": "sdd01",
    "data_dictionary_name": "テスト用共有辞書",
    "trigger": "利用者",
    "start_event": "開始イベント",
    "event_class": "イベント区分",
    "transaction_start_condition": "取引開始条件",
    "start_date": "2018/01/01",
    "transaction_end_condition": "取引終了条件",
    "end_date": "2020/12/31",
    "owner_hct_id": "10000000000000000000000000000035",
    "owner_unit_id": "20030000000000000000000000000001",
    "owner_unit_class": "ECU",
    "owner_edu_id": "20040000000000000000000000000001",
    "user_hct_id": "10000000000000000000000000000036",
    "user_unit_id": "20030000000000000000000000000001",
    "user_unit_class": "ECU",
    "user_edu_id": "20040000000000000000000000000001",
    "owner_tdp_id": "20010000000000000000000000000001",
    "user_tdp_id": "20010000000000000000000000000001",
    "owner_tsp_id": "20020000000000000000000000000001",
    "user_tsp_id": "20020000000000000000000000000001",
    "request_parameter": "リクエストパラメータ",
    "storage_restriction": "false",
    "use_restriction": "利用制限",
    "modified_restriction": "改変制限",
    "disclosure_restriction": "開示制限",
    "other_conditions": "その他の条件"
  },
  "tcp_temporary_registration_flag": "false",
  "tdp": {
    "tdp_id": "20010000000000000000000000000001",
    "tdp_name": "株式会社テストのサンプル TDP",
    "dcm_class": "利用",
    "transaction_start_condition": "取引開始条件",
    "start_date": "2018/01/01",
    "transaction_end_condition": "取引終了条件",
    "end_date": "2020/12/31",
    "storage_restriction": "保存制限なし",
```



```
    "use_restriction": "利用制限なし",
    "modified_restriction": "改変制限なし",
    "disclosure_restriction": "false",
    "drm_id": [
      "drm01"
    ],
    "dtm_id": [
      "dtm01"
    ]
  },
  "tdp_temporary_registration_flag": "false",
  "tsp": {
    "tsp_id": "2002000000000000000000000000001",
    "tsp_name": "株式会社テストのサンプル TSP ",
    "data_dictionary_id": "add01",
    "data_dictionary_name": "株式会社テストのサンプル個別辞書",
    "dcm_id": "dcm01",
    "dcm_name": "温湿度計",
    "process_id": "process01",
    "arrangement_id": "arrangement01",
    "data_class": "利用",
    "process_name": "温湿度計のデータ入手プロセス",
    "category_id": "category01",
    "category_name": "サンプルカテゴリ",
    "unit_id": "ecu01",
    "unit_name": "株式会社テストの ECU",
    "hct_id": "hct01",
    "hct_name": "株式会社テストの HCT",
    "transaction_start_condition": "取引開始条件",
    "start_date": "20180101",
    "transaction_end_condition": "取引終了条件",
    "end_date": "2020/12/31",
    "event_list": [
      {
        "event_id": "event01",
        "event_name": "イベント名",
        "event_class": "利用",
        "event_ archive": "イベントに関する記録"
```

```
        }
      ],
      "data_restriction_list": [
        {
          "dcm_id": "dcm01",
          "dcm_name": "制約に関するデータ",
          "dcm_class": "利用",
          "dcm_restriction": "ここに制約を記載する"
        }
      ],
      "secondary_disclosure_list": [
        {
          "unit_id": "",
          "unit_name": "",
          "disclosure_level": "なし",
          "access_restriction": "なし"
        }
      ]
    },
    "tsp_temporary_registration_flag": "false"
}
```

レスポンス

```
HTTP/1.1 200 OK
Date: Thu, 18 Oct 12:34:56 JST
Content-Type: application/json; charset=UTF-8
Content-Length: xxx
Connection: close

{
      "tcp_id": "20000000000000000000000000000001",
      "tdp_id": "20010000000000000000000000000001",
      "tsp_id": "20020000000000000000000000000001"
}
```

(2) 次に HCT2 に対しても HCT1 と同一の TCP を登録します（設定する ID 等以外は、完全に同一のため、リクエストとレスポンスを省略します)。

これで取引契約フェーズでの設定は完了です。

8.1.3 取引実施フェーズ

取引実施フェーズでは、リクエストの送信、データレコード（DCR)の送信、辞書変換マップの照会、変換の実施を行います。「取引実施フェーズ」においては、プッシュ型通信、プル型通信の二つの通信方法があり、加えて通常通信、DCM 連結通信の二つのパターンがあります。以下、それぞれの例を示します。

8.1.3.1 プッシュ型、通常通信

図 3 のように、HCT1 所属 ECU1 配下の EDU11 から HCT2 所属 ECU2 配下の EDU21 にメッセージ(温度 12.3 度、湿度 60%)を HDS(本デモでは 1 台)経由で送信する場合の例を以下に示します。

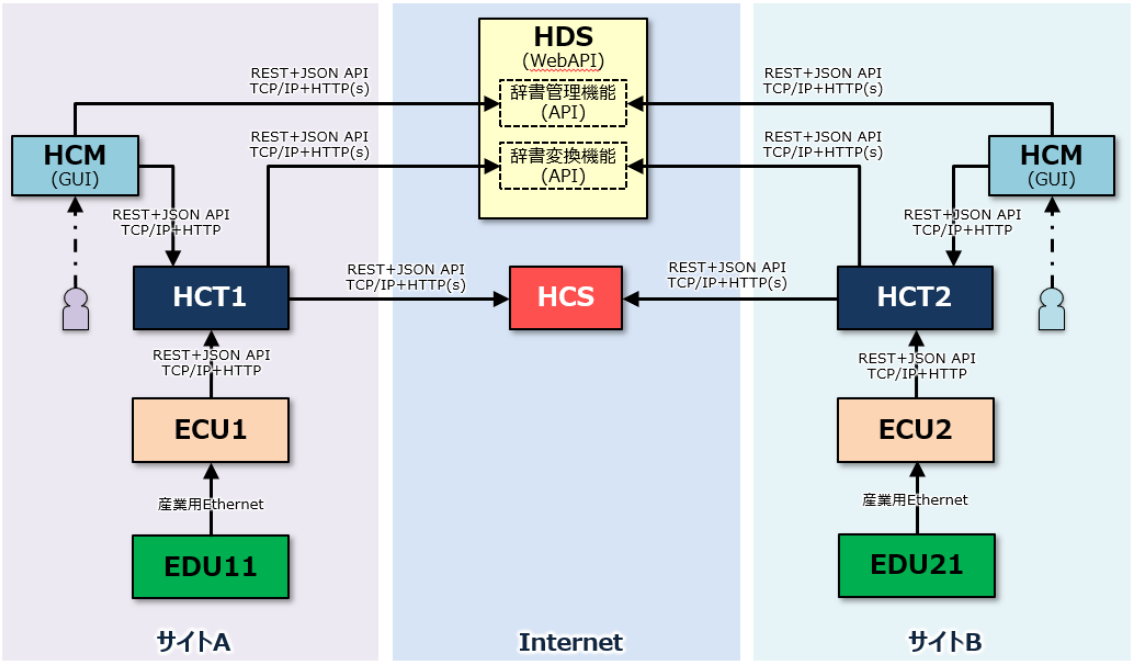

図 3 取引例のシステム構成

(1) ECU1 は、HCT1 に対して、該当する取引契約プロファイルを入手します。
ここでは、取引契約プロファイル"tcp01"が得られるとします。

リクエスト

```
GET /hct/api/v1/tcps?source=edu11&destination=edu21 HTTP/1.1
Accept: application/json
Authorization: Basic xxxx
Host: example.org
```

レスポンス

```
HTTP/1.1 200 OK
Date: Thu, 18 Oct 12:34:56 JST
```

```
Content-Type: application/json; charset=UTF-8
Content-Length: xxx
Connection: close

{
   "tcp_id": ["tcp01"]
}
```

(2)　ECU1 は、HCT1 に対して、温度と湿度に関するメッセージを送信します。

リクエスト

```
POST /hct/api/v1/tcps/tcp01 HTTP/1.1
Accept: application/json
Date: Thu, 18 Oct 12:34:56 JST
Content-Type: application/json; charset=UTF-8
Authorization: Basic xxxx
Host: example.org
Content-Length: xxx

{"message": [{"温度":"12.3"}, {"湿度":"60"}]}
```

レスポンス

```
HTTP/1.1 200 OK
Date: Thu, 18 Oct 12:34:56 JST
Content-Type: application/json; charset=UTF-8
Content-Length: xxx
Connection: close
```

(3)　データレコードを受け取った HCT1 は、HCT1 内に管理されている取引契約プロファイル ID "tcp01"の内容を参照し、関連する取引データプロファイル ID "tdp01"を入手します。続けて、"tdp01"に関連する辞書変換マップ ID のリスト" dtm01"を入手します。

(4)　HCT1 は、入手した辞書変換マップ ID のリスト"dtm01"を用いて、HDS に DTM を問い合わせます。

リクエスト

```
GET /hds/api/dtms/dtm01 HTTP/1.1
Accept: application/json
Authorization: Basic xxxx
Host: example.org
```

レスポンス

```
HTTP/1.1 200 OK
Date: Thu, 18 Oct 12:34:56 JST
Content-Type: application/json; charset=UTF-8
Content-Length: xxx
Connection: close

 {
    "dtm_id": "dtm01",
    "source_data_dictionary_id": "add01",
    "source_dcm_id": "dcm01",
    "source_drm_id": null,
    "destination_data_dictionary_id": "sdd01",
    "destination_dcm_id": "dcm51",
    "destination_drm_id": null,
    "ptm_list": [
       {"source_dpd_name": "温度", "destination_dpd_name": "気温"},
       {"source_dpd_name": "湿度", "destination_dpd_name": "相対湿度"}
    ]
 }
```

(5)　HCT1 は、上記 DTM のうち PTM リストを用いて、データ項目を変換します。

(6)　変換後、HCT1 は HCS に対して、次のようなメッセージを送信します。

リクエスト

```
POST / hcs/api/v1/message HTTP/1.1
Accept: application/json
Date: Thu, 18 Oct 12:34:56 JST
Content-Type: application/json; charset=UTF-8
Authorization: Basic xxxx
Host: example.org
Content-Length: xxx

{
     "tcp": {省略},
    "dcr": [{"気温":"12.3"}, {"相対湿度":"60"}]
}
```

レスポンス

```
HTTP/1.1 200 OK
Date: Thu, 18 Oct 12:34:56 JST
Content-Type: application/json; charset=UTF-8
Content-Length: xxx
Connection: close
```

(7)　HCS は、自身内部に取引契約プロファイル情報と DCR を紐づけて保存します。

(8)　（その後、非同期に）HCT2 は、ECU2 からの以下のリクエストを受けます。

リクエスト

```
GET /hct/api/v1/message?destination=ecu02 HTTP/1.1
Accept: application/json
Authorization: Basic xxxx
Host: example.org
```

(9)　リクエストを処理する際に、HCT2 は、HCS に対して、HCT2 宛のメッセージを以下の Json 形式のファイルで取得します。

リクエスト

```
GET /hcs/api/v1/message?destination_hct=hct02&destination_ecu=ecu02 HTTP/1.1
Accept: application/json
Authorization: Basic xxxx
Host: example.org
```

レスポンス

```
HTTP/1.1 200 OK
Date: Thu, 18 Oct 12:34:56 JST
Content-Type: application/json; charset=UTF-8
Content-Length: xxx
Connection: close

{
   "source_hcs_id": "hcs01",
   "destination_hct_id: "hct02",
   "destination_ecu_id: "ecu02",
   "message": [
       {
        "tcp_id": "tcp01",
        "dcr": [{"気温":"12.3"}, {"相対湿度":"60"}]
```

```
      }
  ]
}
```

（10） HCT2 は、自身が管理している取引契約プロファイル ID "tcp01"の内容を参照し、関連する取引データプロファイル ID "tdp11"を入手します。続けて、"tdp11"に関連する辞書変換マップ ID のリスト" dtm11"を入手します。

（11） 入手した辞書変換マップ ID のリスト"dtm11"を用いて、HDS に以下の通り問い合わせることで、Json 形式のファイルで DTM が応答されます。

リクエスト

```
GET /hds/api/dtms/dtm11 HTTP/1.1
Accept: application/json
Authorization: Basic xxxx
Host: example.org
```

レスポンス

```
HTTP/1.1 200 OK
Date: Thu, 18 Oct 12:34:56 JST
Content-Type: application/json; charset=UTF-8
Content-Length: xxx
Connection: close

{
  "dtm_id": "dtm11",
  "source_data_dictionary_id": "sdd01",
  "source_dcm_id": "dcm51",
  "source_drm_id": null,
  "destination_data_dictionary_id": "add11",
  "destination_dcm_id": "dcm11",
  "destination_drm_id": null,
  "ptm_list": [
    {"source_dpd_name": "気温", "destination_dpd_name": "気温"},
    {"source_dpd_name": "相対湿度", "destination_dpd_name": "湿度"}
  ]
}
```

（12） HCT2 は、上記 DTM のうち PTM リストを用いて、データ項目を変換します。
変換後のデータおよびデータ項目
[{"気温":"12.3"}, {"湿度":"60"}]
（13） 変換後、メッセージを EDU2 に対して応答します。

レスポンス

```
HTTP/1.1 200 OK
Date: Thu, 18 Oct 12:34:56 JST
Content-Type: application/json; charset=UTF-8
Content-Length: xxx
Connection: close

{
  "source_hct_id": "hct01",
  "destination_ecu_id": "ecu02",
  "response": [
    {
      "destination_edu_id": "edu21",
      "message": [
          {"気温":"12.3"},
          {"湿度":"60"}
          ]
    }
  ]
}
```

8.1.3.2 プル型、通常通信

プル型通信の場合は、利用者からの送信要求によって、提供者はメッセージを送信します。システム構成は、図 3 と同じであるとします。したがって、ECU2 からの送信要求を起点に、HCT1 所属 ECU1 配下の EDU11 から HCT2 所属 ECU2 配下の EDU21 にメッセージ(温度 12.3 度、湿度 60%)を HDS(本デモでは 1 台)経由で送信します。

(1) ECU2 は、定められた取引契約プロファイルに従い、HCT2 に対して提供者にメッセージの送信要求をします。この取引の TCP ID は tcp02 とします。

リクエスト

```
POST /hct/api/v1/tcps/tcp02 HTTP/1.1
Accept: application/json
```

```
Date: Thu, 18 Oct 12:34:56 JST
Content-Type: application/json; charset=UTF-8
Authorization: Basic xxxx
Host: example.org
Content-Length: xxx

{
    “message”: null
}
```

レスポンス

```
HTTP/1.1 200 OK
Date: Thu, 18 Oct 12:34:56 JST
Content-Type: application/json; charset=UTF-8
Content-Length: xxx
Connection: close

{
    “error_id”: “0”
}
```

(2)　HCT2 は、HCS に対してメッセージの送信要求を送信します。

リクエスト

```
POST / hcs/api/v1/message HTTP/1.1
Accept: application/json
Date: Thu, 18 Oct 12:34:56 JST
Content-Type: application/json; charset=UTF-8
Authorization: Basic xxxx
Host: example.org
Content-Length: xxx

{
    “message”: null
}
```

レスポンス

```
HTTP/1.1 200 OK
Date: Thu, 18 Oct 12:34:56 JST
```

```
Content-Type: application/json; charset=UTF-8
Content-Length: xxx
Connection: close
```

(3)　　（その後、非同期に）HCT1 は、ECU1 からの以下のリクエストを受けます。

リクエスト

```
GET /hct/api/v1/message?destination=ecu01 HTTP/1.1
Accept: application/json
Authorization: Basic xxxx
Host: example.org
```

(4)　　リクエストを処理する際に、HCT1 は、HCS に対して、HCT1 宛のメッセージを以下の Json 形式のファイルで取得します。

リクエスト

```
GET /hcs/api/v1/message?destination_hct=hct01&destination_ecu=ecu01 HTTP/1.1
Accept: application/json
Authorization: Basic xxxx
Host: example.org
```

レスポンス

```
HTTP/1.1 200 OK
Date: Thu, 18 Oct 12:34:56 JST
Content-Type: application/json; charset=UTF-8
Content-Length: xxx
Connection: close

{
   "source_hcs_id": "hcs01",
   "destination_hct_id: "hct01",
   "destination_ecu_id: "ecu01",
   "message": [
        {
         "tcp_id": "tcp02",
         "dcr": null
        }
   ]
}
```

(5)　　HCT1 は、メッセージを ECU1 に対して応答します。

レスポンス

```
HTTP/1.1 200 OK
Date: Thu, 18 Oct 12:34:56 JST
Content-Type: application/json; charset=UTF-8
Content-Length: xxx
Connection: close

{
   "source_hct_id": "hct01",
   "destination_ecu_id": "ecu01",
   "response": [
      {
         "destination_edu_id": "edu11",
         "message": null
      }
   ]
}
```

(6)　　ECU1 は、上記の要求に対して、メッセージを送信します。

(7)　　以後の流れは、8.1.3.1 の(2)以降と同様ですので、省略します。

8.1.3.3 プル型、DCM 連結の通信

ここでは、プル型における DCM 連結通信の場合の通信例について説明します。図 4 のように、HCT1 所属 ECU1 配下の EDU11 と EDU12 から HCT2 所属 ECU2 配下の EDU21 にメッセージ(温度 12.3 度、湿度 60%、空調動作状態)を HDS(本デモでは 1 台)経由で送信する場合の例を以下に示します。なお、EDU11 は、温湿度計であり、温度 12.3 度、湿度 60%のメッセージを送信し、EDU12 は、空調設備であり、空調動作状態を送信します。

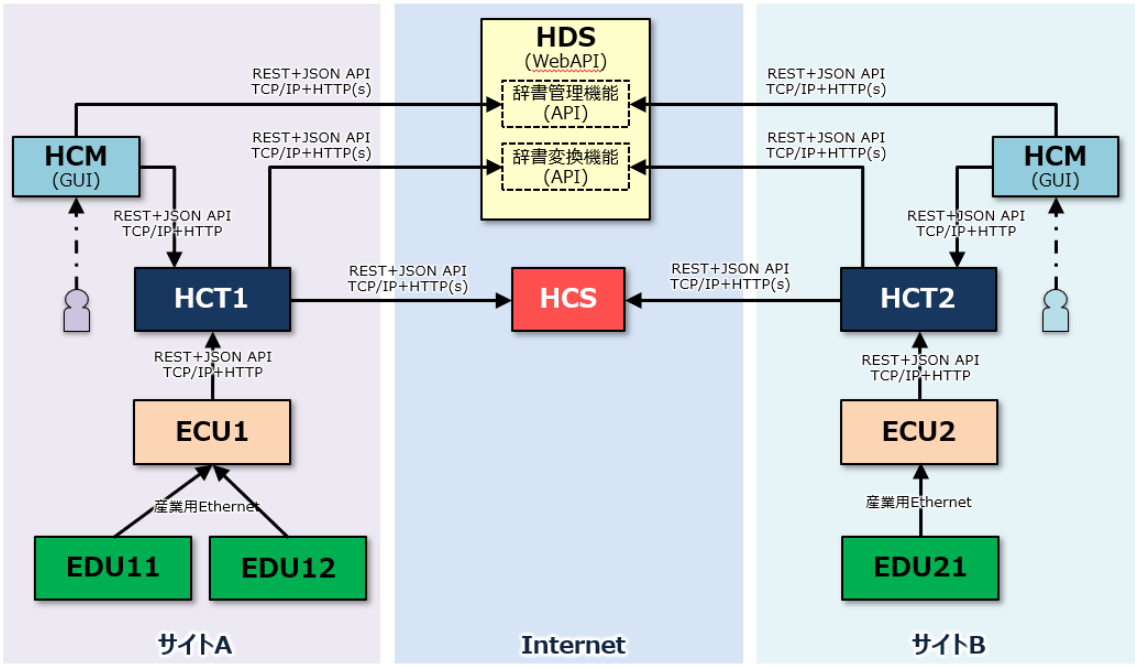

図 4　取引例のシステム構成

(1)　　ECU2 は、定められた取引契約プロファイルに従い、HCT2 に対して提供者にメッセージの送信要求をします。この取引の TCP ID は tcp03 とします。

リクエスト

```
POST /hct/api/v1/tcps/tcp03 HTTP/1.1
Accept: application/json
Date: Thu, 18 Oct 12:34:56 JST
Content-Type: application/json; charset=UTF-8
Authorization: Basic xxxx
Host: example.org
Content-Length: xxx

{
    “message”: null
}
```

レスポンス

```
HTTP/1.1 200 OK
Date: Thu, 18 Oct 12:34:56 JST
Content-Type: application/json; charset=UTF-8
Content-Length: xxx
Connection: close

{
```

```
    “error_id”: “0”
}
```

(2)　　HCT2 は、HCS に対してメッセージの送信要求を送信します。

リクエスト

```
POST / hcs/api/v1/message HTTP/1.1
Accept: application/json
Date: Thu, 18 Oct 12:34:56 JST
Content-Type: application/json; charset=UTF-8
Authorization: Basic xxxx
Host: example.org
Content-Length: xxx

{
    “message”: null
}
```

レスポンス

```
HTTP/1.1 200 OK
Date: Thu, 18 Oct 12:34:56 JST
Content-Type: application/json; charset=UTF-8
Content-Length: xxx
Connection: close
```

(3)　　（その後、非同期に）HCT1 は、ECU1 からの以下のリクエストを受けます。

リクエスト

```
GET /hct/api/v1/message?destination=ecu01 HTTP/1.1
Accept: application/json
Authorization: Basic xxxx
Host: example.org
```

(4)　　リクエストを処理する際に、HCT1 は、HCS に対して、HCT1 宛のメッセージを以下の Json 形式のファイルで取得します。

リクエスト

```
GET /hcs/api/v1/message?destination_hct=hct01&destination_ecu=ecu01 HTTP/1.1
Accept: application/json
Authorization: Basic xxxx
Host: example.org
```

レスポンス

```
HTTP/1.1 200 OK
Date: Thu, 18 Oct 12:34:56 JST
Content-Type: application/json; charset=UTF-8
Content-Length: xxx
Connection: close

{
   "source_hcs_id": "hcs01",
   "destination_hct_id: "hct01",
   "destination_ecu_id: "ecu01",
   "message": [
        {
         "tcp_id": "tcp03",
         "dcr": null
        }
   ]
}
```

(5)　　HCT1 は、メッセージを ECU1 に対して応答します。

レスポンス

```
HTTP/1.1 200 OK
Date: Thu, 18 Oct 12:34:56 JST
Content-Type: application/json; charset=UTF-8
Content-Length: xxx
Connection: close

{
   "source_hct_id": "hct01",
   "destination_ecu_id": "ecu01",
   "response": [
      {
         "destination_edu_id": "edu11",
         "message": null
      }
   ]
```

```
}
```

(6)　　ECU1 は、上記の要求に対して、メッセージを送信しますが、この取引においては、EDU11 および EDU12 のそれぞれから非同期にメッセージが送信される想定です。はじめに、EDU11 から ECU1 へメッセージを送信します。

リクエスト

```
POST /hct/api/v1/tcps/tcp03 HTTP/1.1
Accept: application/json
Date: Thu, 18 Oct 12:34:56 JST
Content-Type: application/json; charset=UTF-8
Authorization: Basic xxxx
Host: example.org
Content-Length: xxx

{"message": [{"温度":"12.3"}, {"湿度":"60"}]}
```

レスポンス

```
HTTP/1.1 200 OK
Date: Thu, 18 Oct 12:34:56 JST
Content-Type: application/json; charset=UTF-8
Content-Length: xxx
Connection: close
```

(7)　　HCT1 は、tcp03 に対する TDP（ここでは tdp03 とする）内の dtm_id(ここでは dtm03 とする)を確認し、DTM の内容について HDS に以下の通り問い合わせます。

リクエスト

```
GET /hds/api/v1/dtms/dtm03 HTTP/1.1
Accept: application/json
Authorization: Basic xxxx
Host: example.org
```

レスポンス

```
HTTP/1.1 200 OK
Date: Thu, 18 Oct 12:34:56 JST
Content-Type: application/json; charset=UTF-8
Content-Length: xxx
Connection: close
```

```
{
    "dtm_id": "dtm03",
    "source_data_dictionary_id": "add01",
    "source_dcm_id": "dcm03",
    "source_drm_id": "drm03",
    "destination_data_dictionary_id": "sdd01",
    "destination_dcm_id": "dcm53",
    "destination_drm_id": "drm53",
    "is_tentative": false,
    "ptm_list": [
      {"source_dpd_name": "温度", "destination_dpd_name": "気温"},
      {"source_dpd_name": "湿度", "destination_dpd_name": "相対湿度"}
    ]
}
```


(8)　　HCT1 は、source_drm_id および destination_drm_id が null でないことを確認することで、他のメッセージが必要であると判断し、このタイミングでは HCS に対してメッセージを送りません。

(9)　　（続けて、非同期に）EDU12 から HCT1 へメッセージを送信します。

リクエスト

```
POST /hct/api/v1/tcps/tcp03 HTTP/1.1
Accept: application/json
Date: Thu, 18 Oct 12:34:56 JST
Content-Type: application/json; charset=UTF-8
Authorization: Basic xxxx
Host: example.org
Content-Length: xxx

{"message": [{"空調動作状態":"通常運転"}]}
```

レスポンス

```
HTTP/1.1 200 OK
Date: Thu, 18 Oct 12:34:56 JST
Content-Type: application/json; charset=UTF-8
Content-Length: xxx
Connection: close
```

(10)　HCT1 は、(7)で HDS より取得した DTM の drm_id を用いて以下の通り HDS に対して DRM を問い合わせます。

リクエスト

```
GET /hds/api/v1/data_dictionaries/add01/drms/drm03 HTTP/1.1
Accept: application/json
Authorization: Basic xxxx
Host: example.org
```

レスポンス

```
HTTP/1.1 200 OK
Date: Thu, 18 Oct 12:34:56 JST
Content-Type: application/json; charset=UTF-8
Content-Length: xxx
Connection: close

{
    "drm_id": "drm03",
    "data_dictionary_id": "add01",
    "dcm_id": "dcm03",
    "dpd_name_list": ["空調動作状態"],
    "target_dcm_id": "dcm04",
    "target_dpd_name_list": ["動作状態"],
    "additional_dpd_name_list": null
}
```

(11)　これにより HCT1 は、すべてのメッセージがそろったと判断し、このタイミングでは HCS に対してメッセージを送ります。

リクエスト

```
POST / hcs/api/v1/message HTTP/1.1
Accept: application/json
Date: Thu, 18 Oct 12:34:56 JST
Content-Type: application/json; charset=UTF-8
Authorization: Basic xxxx
Host: example.org
Content-Length: xxx
```

```
{
"tcp": {省略},
    "dcr": [{"気温":"12.3"}, {"相対湿度":"60"}, {"動作状態":"通常運転"}]
}
```

レスポンス

```
HTTP/1.1 200 OK
Date: Thu, 18 Oct 12:34:56 JST
Content-Type: application/json; charset=UTF-8
Content-Length: xxx
Connection: close
```

(12)　以下、他で説明した手順と同様となるため省略します。

8.2 参照規定

[1] 一般社団法人インダストリアル・バリューチェーン・イニシアティブ。「製造プラットフォームオープン連携事業システム実装基本要件仕様（案）」、2018。
[2] 日立製作所。「サブシステムプロトタイプ実装仕様(HDS/HCM 編)」、2018。
[3] R.Fielding and J.Reschke. Hypertext Transfer Protocol (HTTP/1.1): Message Syntax and Routing, 2014. RFC 7230, http:/tools.ietf.org/html/rfc7230.
[4] R. Fielding and J.Reschke. Hypertext Transfer Protocol (HTTP/1.1): Authorization, 2014. RFC 7235, https:/tools.ietf.org/html/rfc7235.

8.3 用語定義

用語	意味
ADD	Actual Data Dictionally（個別データ辞書） 個々のサイトやターミナル（HCT）あるいはターミナル下のコントロラ（ECU）で利用するすべての DCM をリストとしてもつもの。
DCM	Data Component Model（データモデル） 辞書に登録するデータの単位。E-R 図における Entity、UML のクラス図におけるクラス，RDB のスキーマにおけるテーブルまたはビューに相当する。

用語	意味
DCC	Data Category（データカテゴリ） データモデル（DCM）をカテゴリ分類したもの。ここで定義したカテゴリは、カテゴリ辞書（SCD）に登録され利用される。
DCR	Data Component Record（データレコード） DCM に沿って登録された具体的なデータ内容。
DPD	Data Property Definition（データ定義項目） DCM を構成する項目であり、この単位でデータ要素が設定される。RDB におけるフィールドに相当する。
DTM	Dictionary Translation Map（辞書変換マップ） 個別辞書、共通などに対応したそれぞれの変換元、変換先の DCM を対応づけたもの。変換元、変換先が連結された 1 つ以上の DCM である場合には、多対の関係となる。
EAU	Edge Application Unit（エッジアプリケーション） 各サイト内で、データを活用したアプリケーションを提供する装置。クラウド上で実行しないオンプレミス型の業務アアリはすべてこれに該当する。
ECC	Event and Condition Category（イベントカテゴリ） イベントモデル（ECM）のカテゴリを示したもの。すべての ECM はいずれかのイベントカテゴリ（ECC）に属する。
ECM	Event and Condition Model（イベントモデル） サイバー上での事象の定義単位。PCM のトリガとなる。ECM は、 PCM の処理によって定義される場合、DCM の特定の値の状態で定義される場合、フィジカル世界の事象に対応づけて定義する場合がある。
ECU	Edge Control Unit（エッジコントローラ） 各サイト内で、それぞれの目的に応じて利用するデータを管理しデータを処理制御する装置。HCT を介して、他のサイトからデータを取得あるいは提供する。
EDU	Edge Device Unit（エッジデバイス） 物理的な世界からデータを取得するセンサー、あるいはデータを物理的な世界に適用するアクチュエータに付随して、データの起点と終点となる機器。
HCM	Hyper Connection Manager（連携マネージャ） 異なるサイト間で、ECU や EAU が連携する場合に、データ流通の内容や利用方法、辞書の利用や変換方式などを設定するための UI を提供する。
HCS	Hyper Connection Server（連携サーバ） インターネット上に配置され、下の連携ターミナルとの通信を行うと共に、他のサーバとのとの通信により、ヘテロなサイト間連携を可能とする。
HCT	Hyper Connection Terminal（連携ターミナル） 各サイトのプライベートなネットワーク内に位置し、ローカル IP アドレスを持つ。外部のインターネットとは、あらかじめ設定した HCS とのみ通信する。

用語	意味
HDS	Hyper Dictionary Server（辞書サーバ） 共通辞書、個別および辞書変換テーブルを管理し、登録や修正検索などに対応する。個別辞書をサイトごとに管理しつつ、共通辞書の改変を支援する。
PCC	Process Component Category（プロセスカテゴリ） プロセスモデル（PCM）のカテゴリを示したもの。すべての PCM はいずれかのカテゴリ（PCC）に属する。
PCM	Process Component Model（プロセスモデル） サイバー上の処理の単位。特定の DCM の内容をもとに、あらかじめ定めれた手順に従った計算を行い、特定の DCM の内容を操作する。
PTM	Property Translation Map（項目変換マップ） 辞書変換において DCM がもつ項目（DPD）間の対応付けを示すもの。
SDD	Specific Category Dictionary（カテゴリ辞書） データカテゴリ（DCC)、プロセスカテゴリ（PCC)、イベントカテゴリ（ECC）をまとめたもの。複数のカテゴリ辞書が定義できるが、辞書間の変換の定義はできない。
TCP	Trade Contract Profile（取引契約プロファイル） ２つのサイト間で、データ流通の形式や方法、契約内容を定めたもの。データの保存方法、権利の帰属、課金方法、禁止事項なども含む。
TDP	Trade Data Profile（取引データプロファイル） メッセージ送信時あるいは受信時にメッセージ内で利用しているデータ定義（DCM）について共通辞書または個別辞書で記述したもの。リクエストされたデータの DCM は、共通辞書上では複数の DCM の連結で表現される場合などは、その構造も示す。
TSP	Trade Service Profile（取引サービスプロファイル） データを提供する側、および利用する側でのプロセス（PCM）および関連するイベント（ECM）の内容を定義したもの。また、実際に実行する ECU、EAU、EDU なども示す。TSP 上で定義した DCM は、HCM で照会可能とする。