

パーソナルデータ連携モジュール 利用設定手順書

2022年11月30日

日本電気株式会社



改版履歴

版	作成日	変更内容
1.0	2022/10/1	新規作成
1.1	2022/11/30	「1.2 関連ドキュメント」の「表 1-1-1 関連ドキュメント」の記載を変更 Block コード、アクターコードの共通マニフェスト対応 ・ 3.3 PF 設定 : 追加 PxR-Block の EKS マニフェスト作成および起動 ・ 3.8 PF 設定 : EKS マニフェストアクターコード設定

目次

1	はじめに	5
1.1	本手順書の位置付け	5
1.2	関連ドキュメント	5
1.3	前提条件	5
1.4	バージョン	5
1.5	環境構成	6
1.6	表記方法	6
1.6.1	コマンド入力方法	6
1.6.2	API 入力方法	7
1.6.3	PxR-Block 名の定義	8
2	PxR の起動・停止	9
2.1	PxR の起動方法	9
2.2	PxR の停止方法	9
2.3	PxR-BLOCK の再起動方法	10
3	PxR-Block の追加方法	11
3.1	追加 PxR-BLOCK の BLOCK カタログ作成	12
3.1.1	Tips. カタログの特定方法	15
3.2	PF 設定：追加 PxR-BLOCK の DB 構築	17
3.3	PF 設定：追加 PxR-BLOCK の EKS マニフェスト作成および起動	17
3.4	追加 PxR-BLOCK の割り当て	20
3.5	アクター認定申請	22
3.6	アクター認定承認、クライアント証明書の発行	25
3.7	クライアント証明書のアップロード	28
3.8	PF 設定：EKS マニフェストアクターコード設定	30
4	サービス設定	31
4.1	グローバル設定（BLOCK 共通設定）	31
4.2	運営メンバーの追加	39
5	PxR-Block の削除方法	42
5.1	アクターの終了	43

5.1.1	Region 終了	44
5.1.2	Region 利用者 ID 連携の解除	47
5.1.3	APP 利用者 ID 連携の解除	49
5.1.4	Region に参加している APP の離脱	52
5.1.5	流通制御による利用者 ID 連携情報の確認	55
5.1.6	アクター認定解除	57
5.2	PF 設定 : POD 停止、EKS マニフェスト削除	59
5.3	PF 設定 : DB 削除	61

1 はじめに

1.1 本手順書の位置付け

パーソナルデータ連携モジュール 利用手順書（以降、本書）は、パーソナルデータ連携モジュール（以降、本モジュール）を利用するにあたり、本モジュールの起動・停止方法、PxR-Block の設定項目の設定方法、PxR-Block の追加・削除方法について記載したものです。

1.2 関連ドキュメント

本手順書に関連するドキュメントを以下に示します。

表 1-1-1 関連ドキュメント

ドキュメント名
パーソナルデータ連携モジュール 構築ガイド
パーソナルデータ連携モジュール アプリケーション開発ガイド

1.3 前提条件

前提条件を以下に示します。

- ・インターネットへアクセス可能であること。
- ・「パーソナルデータ連携モジュール 構築ガイド」を前提に、本モジュールを構築していること。

1.4 バージョン

別紙「パーソナルデータ連携モジュール 構築ガイド」1.4 を参照のこと。

1.5 環境構成

本手順で説明する環境構成イメージを下图に示します。

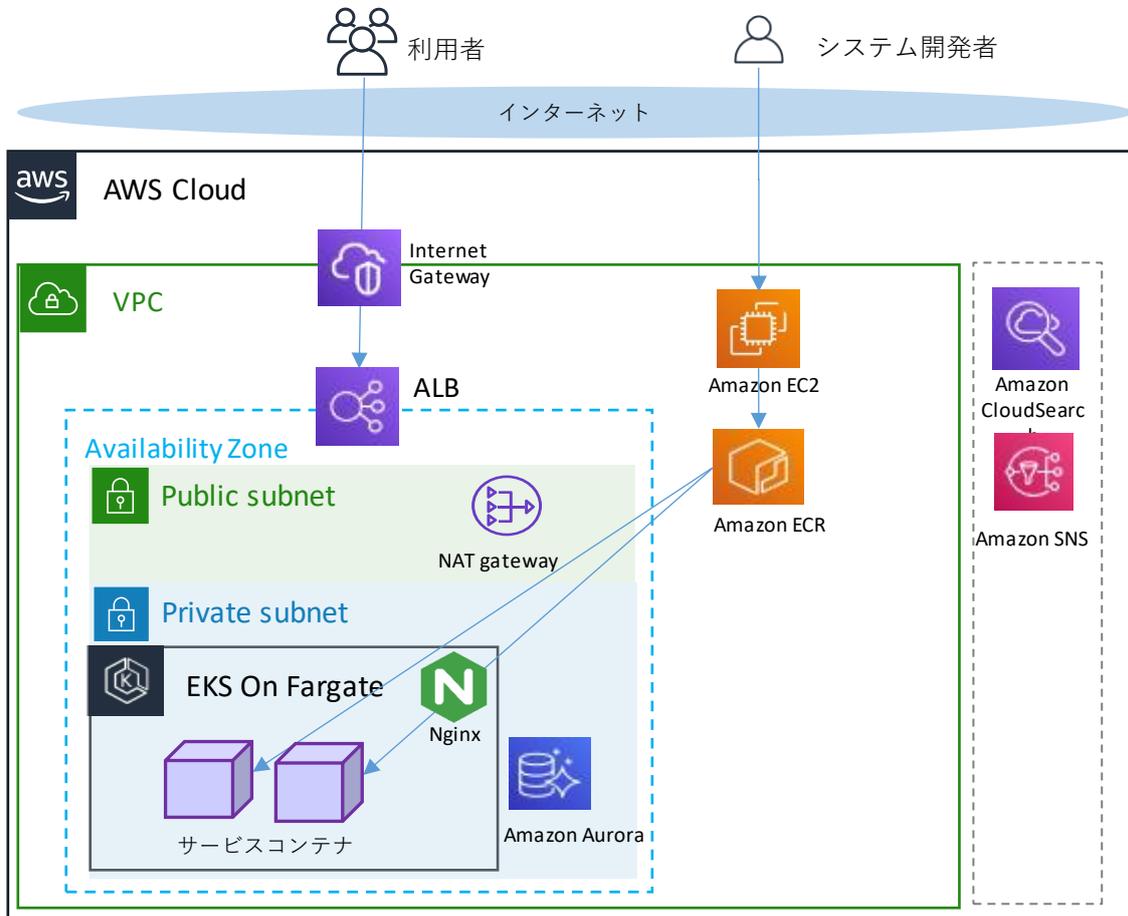


図 1-1 AWS を使用した環境構成イメージ

1.6 表記方法

1.6.1 コマンド入力方法

(例) : Linux コマンド入力

```
# kubectl apply -f . -R  
# kubectl rollout restart -n pxr deployments/<追加 Block の deployment 名>
```

※行頭の # はプロンプトであり、以降のコマンドを入力する。

赤字部分は修正箇所を示し、<> 内はユーザー設定値、もしくは API 取得値を示す。

(例) : ファイル編集

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: application000001-operator-service-container-config-map
  namespace: pxr
data:
  config.json: |
    {
      "session_expire": 168,
      "login_code_expire": 10,
      "initial_password_expire": 7,
      "cookie_base_name": "operator_type%s_session",
      "catalog_url": "http://localhost:3003/pxr-block-proxy/?block=1000401&path=/catalog",
      "catalog_ext_name": "xxx-healthcare-consortium",
      "ca_url": "http://localhost:3012/certification-authority",
      "block": {
        "_value": "<PxR-Blockコード>",
        "_ver": 1
      },
      "actor": {
        "_value": null,
        "_ver": null
      }
    }
    ~~~以降省略~~~
```

※赤字部分は修正箇所を示し、<>内はユーザー設定値、もしくはAPI取得値を示す。

1.6.2 API 入力方法

(例) : API パス

サービス名称	API 名称	メソッドおよびパス
カタログサービス	更新	PUT /catalog/ext/{code}

API パス使用例 : <https://<ドメイン>/catalog/catalog/ext/{code}>

※ドメインの後ろに API パスをつなげて使用する。

(例) : API レスポンス

```
{
  "catalogItem": {
    "ns": "catalog/ext/{ext_name}/setting/actor-own/{actor_type}/actor_{actor_code}",
    "name": "設定の名称",
    "_code": {
      "_value": "対象のコード",
      "_ver": "対象のバージョン"
    },
  },
  "inherit": {
    "_value": "継承元カタログコード",
    "_ver": "継承元カタログバージョン"
  },
}
```

※赤字部分はユーザー設定値、もしくは Response で取得できる設定値を示す。

1.6.3 PxR-Block 名の定義

PxR-Block 論理名と物理名は以下前提で説明します。

表 1-2 PxR-Block 論理名と物理名対比表

PxR-Block 論理名	PxR-Block 物理名 (Kubernetes Pod 名、DB 名)	補足
PxR-Root-Block	root-api ※例	構築ガイドで作成する PxR-Block ※物理名は任意
APP-Block	application000001-api ※例	「3 PxR-Block の追加方法」で作成する。 ※物理名は任意
Region-Root-Block	region000001-api ※例	

2 PxR の起動・停止

本章では、本モジュールの起動・停止・再起動方法について記載する。

2.1 PxR の起動方法

以降に、PxR の開始手順を記載する。

1. DB (AWS Aurora) を起動する。

```
# aws rds start-db-cluster --db-cluster-identifier <DB クラスター>
```

2. Kubernetes の Pod 起動を行う。

Scale コマンドの `--replicas` に値を設定して PxR-Root-Block および追加 PxR-Block の Pod を起動する。

※Pod の冗長化を行う場合は、`--replicas` の数を増やす。

```
# kubectl scale deploy/root-api --replicas=1 -n pxr
# kubectl scale deploy/<開始対象の PxR-Block の物理名> --replicas=1 -n pxr
```

※起動したい Pod の deployment を行う。

3. Kubernetes の Pod ステータス一覧を確認し、Running (動作中) に変われば起動完了となる。

※Pod ステータス一覧レスポンス例

```
# kubectl get Pod -n pxr
NAME                                READY  STATUS  RESTARTS  AGE
application000001-api-df4dbb5f5-pgxnw  9/9    Running  0          13h
region000001-api-6884664976-zzbgg      7/7    Running  0          13h
root-api-86df7cffb9-btrsc              13/13  Running  0          13h
```

2.2 PxR の停止方法

以降に、PxR の停止手順を記載する。

1. Kubernetes の deploy 一覧を確認する。

※Pod ステータス一覧レスポンス例

```
# kubectl get deploy -n pxr
NAME                                READY  UP-TO-DATE  AVAILABLE  AGE
application000001-api  1/1      1            1          75d
region000001-api      1/1      1            1          75d
root-api              1/1      1            1          75d
```

2. Kubernetes の Pod 停止を行う。

Scale コマンドで `--replicas=0` を設定して、Pod の Undeploy を行う。

```
# kubectl scale deploy/root-api --replicas=0 -n pxr
# kubectl scale deploy/<停止対象の Pxr-Block の物理名> --replicas=0 -n pxr
```

※すべての Pod を停止する。

3. DB (AWS Aurora) を停止する。

```
# aws rds stop-db-cluster --db-cluster-identifier <DB クラスター名>
```

4. Kubernetes の Pod ステータス一覧を確認し、停止していることを確認する。

※Pod ステータス一覧レスポンス例

```
# kubectl get Pod -n pxr
No resources found in pxr namespace.
```

2.3 PxR-Block の再起動方法

以降に PxR-Block の再起動手順を記載する。

1. 対象 Pod の PxR-Block を再起動する。

Kubernetes の Restart コマンドで再起動実行する。

```
# kubectl rollout restart -n pxr deployments/<PxR-Block 物理名> -n pxr
```

2. Kubernetes の Pod ステータス一覧を確認し、Running (動作中) に変われば起動完了となる。

※Pod ステータス一覧レスポンス例

```
# kubectl get Pod -n pxr
application000001-api-df4dbb5f5-pgxnw 10/10 Running 0 13h
region000001-api-6884664976-zzbgg 7/7 Running 0 13h
root-api-86df7cffb9-btrsc 17/17 Running 0 13h
```

3 PxR-Block の追加方法

本章では本モジュールにおける、PxR-Block を追加する手順について記載する。

本モジュールにおける PxR-Block 追加は、以下フロー図の流れで追加処理を行う。

※PxR-Block の追加は、以下の構築手順でシステム全体を停止せずに追加処理が可能

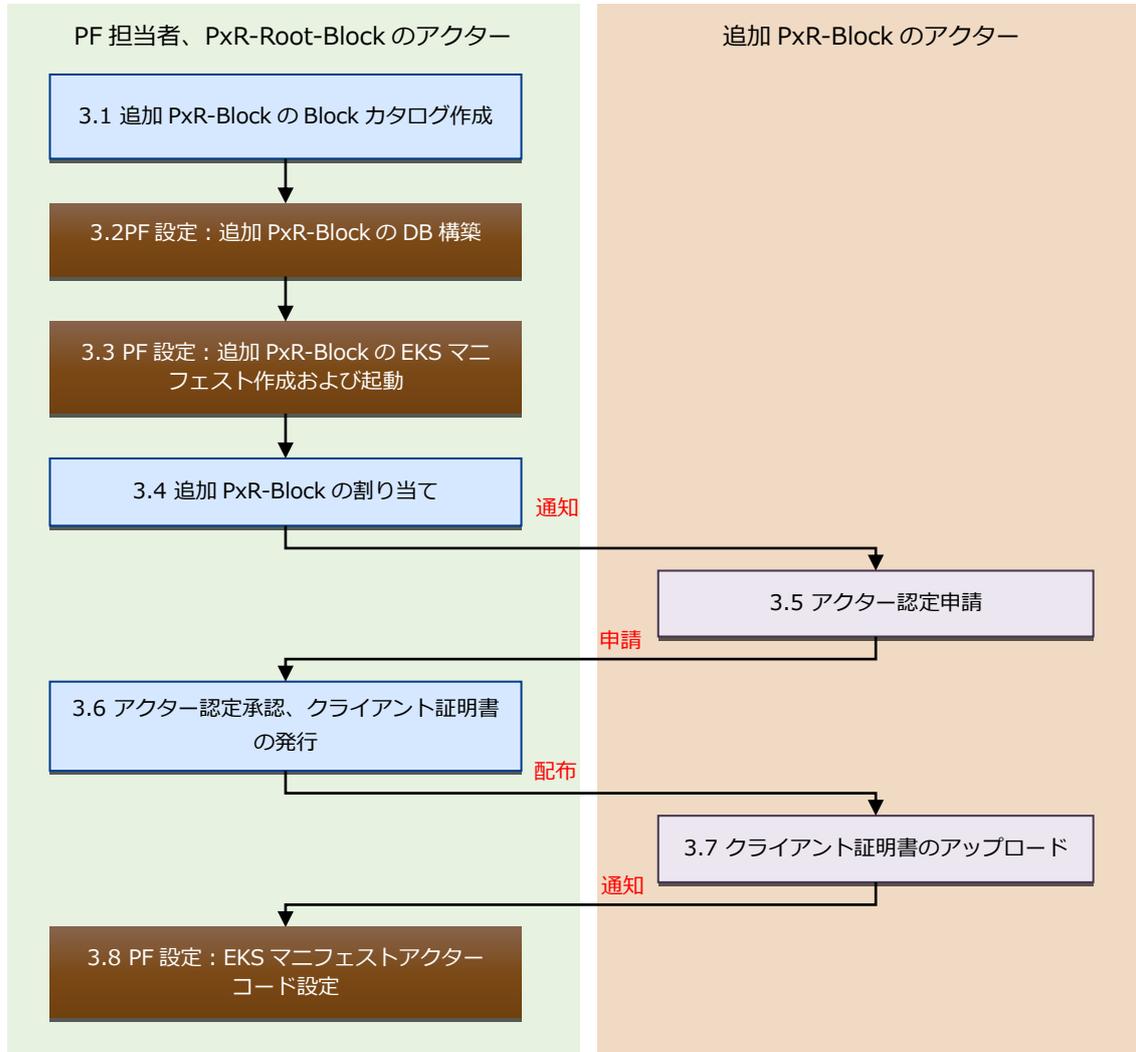


図 3-1 PxR-Block 追加フロー

3.1 追加 PxR-Block の Block カタログ作成

PxR-Root-Block のアクターが、新規 Block のカタログ作成、Block コードの取得を行う。

※PxR-Root-Block にログインセッションを確立して実施する。

1. PxR-Root-Block にログインする。

サービス名称	API 名称	メソッドおよびパス
オペレーターサービス	ログイン	POST /operator/login

リクエスト

```
{
  "type": 3,
  "loginId": "ログイン ID",
  "hpassword": "ハッシュパスワード"
}
```

※ハッシュパスワードは、パスワードを、「構築ガイド」の 2.4 マニフェスト作成、適用で Configmap に設定したソルト値とハッシュ化回数でハッシュ化 (SHA-256) したパスワード

ログイン後の API 使用について

ログイン後に各 API を使用するためにはログイン時に返却される「sessionId」と「XSRF-TOKEN」を設定する必要がある。

a. sessionId

ログイン API のレスポンス内で各 API を使用するための「sessionId」が返却される。

例：ログイン API のレスポンス

```
{
  "sessionId": "cab7561a298146422016cece0e892c3e93287b98bd24357efebcb2d29195eada",
  "operatorId": 1,
  "type": 3,
  "loginId": "ログイン ID",
  "name": "管理者",
  "auth": --- 省略 ---,
  "lastLoginAt": "2022-06-21T15:53:43.226+0900",
  "passwordChangedFlg": true,
  "loginProhibitedFlg": false,
  "attributes": {
    "smsAuth": false
  },
  "block": {
    "_value": 1000401,
    "_ver": 1
  },
  "actor": {
    "_value": 1000431,
    "_ver": 1
  }
}
```

各 API を使用する際にリクエストの cookie に以下の形式で設定する。

```
cookie: operator_type3_session="発行された sessionId"
```

b. XSRF-TOKEN

ログイン API のレスポンスの set-cookie で各 API を使用するための「XSRF-TOKEN」が返却される。

例：

```
set-cookie: XSRF-TOKEN=wNS5VFAK-tryomg1zs1omhd7tk2fkgifmrkg;
```

各 API を使用する際にリクエストの header に以下の形式で設定する。

```
x-xsrf-token: "発行された XSRF-TOKEN"
```

2. PxR-Root-Block でカタログ作成 API を実行する。

本手順により各アクターに割り当てられる Block のカタログを作成する。

サービス名称	API 名称	メソッドおよびパス
カタログサービス	更新	PUT /catalog/ext/{code}

リクエスト

```
{
  "catalogItem": {
    "ns": "catalog/ext/{ext_name}/block/{actor_type}",
    "name": "ブロック名称",
    "_code": null,
    "inherit": {
      "_value": "継承元カタログコード",
      "_ver": "継承元カタログバージョン"
    },
    "description": {
      "title": null,
      "section": [
        {
          "title": "ブロック概要-タイトル",
          "content": [
            {
              "sentence": "ブロック概要-本文"
            }
          ]
        }
      ]
    }
  },
  "template": {
    "value": [
      {
        "key": "actor-type",
        "value": "アクターのタイプ"
      }
    ]
  }
}
```

```

    "key": "assigned-organization",
    "value": null
  },
  {
    "key": "assignment-status",
    "value": "unassigned"
  },
  {
    "key": "base-url",
    "value": "ポータル URL"
  },
  {
    "key": "first-login-url",
    "value": "ポータルの初回ログイン URL"
  },
  {
    "key": "id",
    "value": "識別用 ID"
  },
  {
    "key": "pxr-portal-first-login-url",
    "value": null
  },
  {
    "key": "service-name",
    "value": "サービス名称"
  }
}
]
},
"attribute": null
}

```

名称	説明
catalog/ext/{ext_name}/block/{actor_type}	<ul style="list-style-type: none"> ・ { ext_name } 設定されている ext 名に置き換える。 <ul style="list-style-type: none"> ・ { actor_type } Block を使用する対象のアクターのタイプに置き換える。 アプリケーション P : app 領域運営 SP : region-root
ブロック名称	作成対象 Block の名称を設定する。
継承元カタログコード	継承元となるカタログのコードを設定する。 (「3.1.1Tips. カタログの特定方法 (P15)」を参照し、NS「catalog/model/block/{actor_type}」で取得する)
継承元カタログバージョン	継承元となるカタログのバージョンを設定する。 (「3.1.1Tips. カタログの特定方法 (P15)」を参照し、NS「catalog/model/block/{actor_type}」で取得する)
ブロック概要-タイトル	作成対象 Block の概要分のタイトルを設定する。
ブロック概要-本文	作成対象 Block の概要分の本文を設定する。
アクターのタイプ	Block を使用する対象のアクターのタイプを設定する。 アプリケーション P : app 領域運営 SP : region-root
ポータルの URL	対象のアクター用ポータルのベース URL を設定する。 例) https://[base-url]/portal/login
ポータルの初回ログイン URL	PxR-Block 以外は null が設定される。

識別用 ID	Block の固有 ID を設定する。※他 Block との同名不可
サービス名称	PxR-Block のサービス名を設定する。※他 Block との同名不可 例 : region000001-service

API のレスポンスから、作成されたカタログのコードを確認する。

レスポンス例

```

{
  "catalogItem": {
    "ns": "catalog/model/unit/si/length",
    "name": "cm",
    "description": "センチメートル",
    "_code": {
      "_value": 100046,
      "_ver": 1
    },
    "inherit": null
  },
  "template": {
    "_code": {
      "_value": 1,
      "_ver": 1
    },
    "additionalProp1": {}
  },
  "prop": [
    {
      "key": "index",
      "type": "string"
    },
    {
      "key": "format",
      "type": {
        "of": "item",
        "_code": {
          "value": -9,
          "ver": 1
        },
        "candidate": null
      }
    }
  ]
}

```

カタログコード

カタログバージョン

3.1.1 Tips.カタログの特定方法

NS 検索 API からカタログを取得する。

サービス名称	API 名称	メソッドおよびパス
カタログサービス	取得	GET /catalog?ns={NS} ※PxR-Root-Block 以外の PxR-Block で PxR-Root-Block の API を使用する場合は、proxy を経由するため以下のパスになります。 PUT /pxr-block-proxy/pxr-block-proxy/&block=\$pxr-root-block&path=/catalog?ns={NS}

※{NS}は検索対象のカタログにより異なる。

レスポンス例

```
[
```

```

{
  "catalogItem": {
    "ns": "catalog/model/setting/actor-own/app",
    "name": "setting",
    "_code": {
      "_value": 100068,
      "_ver": 1
    },
    "inherit": null,
    "description": "アプリケーションプロバイダーによるアクター個別設定の定義です。"
  },
  "template": {
    "_code": {
      "_value": 168,
      "_ver": 1
    },
    "use_multifactor_authentication": null,
    "use_user_information": null
  },
  "prop": [
    {
      "key": "use_multifactor_authentication",
      "type": {
        "of": "boolean",
        "cmatrix": null
      },
      "description": "2要素認証",
      "isInherit": false
    },
    {
      "key": "use_user_information",
      "type": {
        "of": "boolean",
        "cmatrix": null
      },
      "description": "利用者管理情報使用設定",
      "isInherit": false
    }
  ],
  "value": null,
  "attribute": null
}
]

```

カタログコード

カタログバージョン

3.2 PF 設定 : 追加 PxR-Block の DB 構築

1. 追加する PxR-Block の DB を構築する。

別紙：『構築ガイドの 2.3 DB 構築』を参照して、追加する Block の DB 接続ユーザーの作成、DB 作成、スキーマの作成、テーブルの作成を実施する。

2. 追加する PxR-Block に運営メンバー（初期ユーザー）を登録する。

別紙：『構築ガイドの 2.7 初期ユーザー登録』を参照して、追加する PxR-Block の初期ユーザーを登録する。

※初期ユーザーから他運営メンバーの追加方法（参照：「4.2 運営メンバーの追加」）

3.3 PF 設定 : 追加 PxR-Block の EKS マニフェスト作成および起動

追加 PxR-Block の EKS マニフェストの作成および適用手順を記載する。

1. 追加 PxR-Block のマニフェストファイルの作成・設定を行う。
 - a. Configmap/<ベース PxR-Block>/*.yaml を<追加 PxR-Block 名>ディレクトリにコピーして、<追加 PxR-Block 名>に変更する。
 - ・コピーしたディレクトリの*.yaml ファイル内の文字列<ベース PxR-Block 名>を<追加 PxR-Block 名>に変換する。

対象ファイル：別紙『構築ガイドの 2.1 資材入手』を参照のこと。

APP-Block の場合：application000001

Region-Root-Block の場合：region000001

サンプル：operator-service-container.yaml ※他全ファイル、全箇所を対象に変更する

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: <ベース PxR-Block 名>-operator-service-container-config-map
  namespace: pxr
data:
  config.json: |
    {
      "session_expire": 168,
      "login_code_expire": 10,
    }
  ~~~ 以降省略 ~~~
```

Grep 置換方法例

```
# sed -i -e 's/<ベース PxR-Block 名>/<追加 PxR-Block 名>/' <対象ファイル>
```

- b. 同様に deployment/<ベース PxR-Block 名>-deployment.yaml をファイルコピーして、<追加

PxR-Block 名> に変更する。

- ・コピーした deployment.yaml のファイル名を <追加 PxR-Block 名> -deployment.yaml に変更する。

- ・ファイル内の文字列 <ベース PxR-Block 名> を <追加 PxR-Block 名> に変換する。

Grep 置換方法例

```
# sed -i -e 's/<ベース PxR-Block 名>/<追加 PxR-Block 名>/' <対象ファイル>
```

c. 同様に service/<ベース PxR-Block 名> -service.yaml をファイルコピーして、に <追加 PxR-Block 名> に変更する。

- ・コピーした service.yaml のファイル名を <追加 PxR-Block 名> -service.yaml に変更する。

- ・ファイル内の文字列 <ベース PxR-Block 名> を <追加 PxR-Block 名> に変換する。

Grep 置換方法例

```
# sed -i -e 's/<ベース PxR-Block 名>/<追加 PxR-Block 名>/' <対象ファイル>
```

d. 構築手順で作成した Ingres /pxr-ingress.yaml に <追加 Block> の Ingress 設定を追加する。

対象ファイル：別紙『構築ガイドの 2.1 資材入手』で示す ingres/pxr-ingress.yaml

サンプル：pxr-ingress.yaml ※最終行に <追加 PxR-Block> の Ingress 設定を追加する。

※ドメインとなる「XXXXX. me.uk」は構築時に使用したドメインを記載する。

```
- host: root.XXXXX. me.uk
  http:
    paths:
      - path: /*
        backend:
          serviceName: root-service
          servicePort: 80

- host: <追加 PxR-Block> .XXXXX. me.uk
  http:
    paths:
      - path: /*
        backend:
          serviceName: <追加 PxR-Block> -service
          servicePort: 80
```

2. DB 構築時に設定した DB 設定値を、各コンテナの yaml ファイルに設定する。

configmap/<追加 PxRBlock>/の各コンテナ yaml ファイルの ormconfig.json 定義部分を DB 構築のときに設定したデータベース名、スキーマ名、ユーザー名、パスワードを設定する。※追加 Block の Configmap 全コンテナ yaml ファイルに設定する。

別紙：『構築ガイドの 2.5 マニフェスト作成、適用』の手順 3 ormconfig.json 編集を参照

3. configmap/common-configmap.yaml を編集し、追加 PxR-Block コードを設定する。

<PxR-Block コード>、_Ver は「3.1 追加 PxR-Block の Block カタログ作成」で確定した Block コード、および ver を使用する。

“actor”の“_value”と“_ver”は null を設定してください。

```
~~~以前省略~~~
apiVersion: v1
kind: ConfigMap
metadata:
  name: application000001-common-config-map
  namespace: pxr
data:
  block-common-conf.json: |
    {
      "block": {
        "_value": 1000407,
        "_ver": 1
      },
      "actor": {
        "_value": 1000436,
        "_ver": 1
      }
    }
}
---
apiVersion: v1
kind: ConfigMap
metadata:
  name: application000004-common-config-map
  namespace: pxr
data:
  block-common-conf.json: |
    {
      "block": {
        "_value": <PxR-Blockコード>,
        "_ver": 1
      },
      "actor": {
        "_value": null,
        "_ver": null
      }
    }
}
~~~以降省略~~~
```

base_block の定義をコピー追加して、Blockコード、アクターコードの設定を行う。

4. Kubernetes の Configmap、deployment、Service、Ingress の yaml 適用、Pod の起動を行う。

```
# kubectl apply -f . -R
```

※上記コマンドで実行ディレクトリ配下の yaml の更新が行われる。

追加 PxR-Block の deployment.yaml の yaml 適用で Pod が起動されない場合は、「2.1 PxR の起動方法」で追加 PxR-Block の起動を行う。

5. Kubernetes の Pod ステータスを確認して、追加 PxR-Block が、Running（動作中）に変われば起動完了となる。

※Pod ステータス一覧レスポンス例

```
# kubectl get Pod -n pxr
application000001-api-df4dbb5f5-pgxnw 10/10 Running 0 13h
region000001-api-6884664976-zzbgg 7/7 Running 0 13h
root-api-86df7cffb9-btrsc 17/17 Running 0 13h
```

3.4 追加 PxR-Block の割り当て

PxR-Root-Block のアクターが、追加した PxR-Block の割り当てを行う。

※PxR-Root-Block にログインセッションを確立して実施する。

1. PxR-Root-Block にログインする。

サービス名称	API 名称	メソッドおよびパス
オペレーターサービス	ログイン	POST /operator/login

リクエスト

```
{
  "type": 3,
  "loginId": "ログイン ID",
  "hpassword": "ハッシュパスワード"
}
```

※ハッシュパスワードは、パスワードを、「構築ガイド」の 2.4 マニフェスト作成、適用で Configmap に設定したソルト値とハッシュ化回数でハッシュ化 (SHA-256) したパスワード

2. PxR-Root-Block でカタログ変更 API を実行する。

本手順により Block カタログを割当状態に変更する。

サービス名称	API 名称	メソッドおよびパス
カタログサービス	更新	PUT /catalog/ext/{code}

リクエスト (block 追加時のリクエスト時の内容を引き継いでください)

```
{
  "catalogItem": {
    "ns": "catalog/ext/{ext_name}/block/{actor_type}",
    "name": "ブロック名称",
    "_code": {
      "_value": "作成時に確認したカタログコード",
      "_ver": "作成時に確認したカタログバージョン"
    },
    "inherit": {
      "_value": "継承元カタログコード",
      "_ver": "継承元カタログバージョン"
    },
    "description": {
      "title": null,
      "section": [
        {
          "title": "ブロック概要-タイトル",
          "content": [
            {
              "sentence": "ブロック概要-本文"
            }
          ]
        }
      ]
    }
  }
}
```

```

"template": {
  "value": [
    {
      "key": "actor-type",
      "value": "アクターのタイプ"
    },
    {
      "key": "assigned-organization",
      "value": "Block 割り当てを行う組織名称"
    },
    {
      "key": "assignment-status",
      "value": "assigned"
    },
    {
      "key": "base-url",
      "value": "ポータル URL"
    },
    {
      "key": "first-login-url",
      "value": "ポータルの初回ログイン URL"
    },
    {
      "key": "id",
      "value": "識別用 ID"
    },
    {
      "key": "pxr-portal-first-login-url",
      "value": null
    },
    {
      "key": "service-name",
      "value": "サービス名称"
    }
  ]
},
"attribute": null
}

```

※赤字部分を変更する。

- 追加 PxR- Block のアクターに、URL 情報、初期ログイン ID、初期パスワードの連絡、および「3.5 アクター認定申請」の実施連絡を行う。

3.5 アクター認定申請

追加 PxR-Block のアクターが、アクター認定申請を行う。

※追加 PxR-Block にログインセッションを確立して実施する。

1. アクター認定申請を行う追加 Block にログインする。

サービス名称	API 名称	メソッドおよびパス
オペレーターサービス	ログイン	POST /operator/login

リクエスト

```
{
  "type": 3,
  "loginId": "ログイン ID",
  "hpassword": "ハッシュパスワード"
}
```

※ハッシュパスワードは、パスワードを、「構築ガイド」の 2.4 マニフェスト作成、適用で Configmap に設定したソルト値とハッシュ化回数でハッシュ化 (SHA-256) したパスワード値

2. 追加 PxR-Block でアクター認定申請を行う。

サービス名称	API 名称	メソッドおよびパス
カタログ更新サービス	申請	POST /catalog-update/actor ※proxy を経由するため以下のパスになります。 POST /pxr-block-proxy/pxr-block-proxy/?block=\$pxr-root-block&path=/catalog-update/actor

リクエスト

```
{
  "approvalActor": {
    "_value": "PxR-Root-Block のアクターコード",
    "_ver": "PxR-Root-Block のアクターバージョン"
  },
  "actorCatalog": {
    "catalogItem": {
      "ns": "catalog/ext/{ext_name}/actor/{actor_type}",
      "name": "組織名",
      "description": "組織の概要",
      "_code": null,
      "inherit": {
        "_value": "継承元カタログコード",
        "_ver": "継承元カタログバージョン"
      }
    }
  },
  "template": {
    "prop": null,
    "value": [
      {
        "key": "breakaway-flg",
        "value": false
      },
      {
        "key": "category",

```

```

    "value": null
  },
  {
    "key": "information-site",
    "value": null
  },
  {
    "key": "main-block",
    "value": [
      {
        "key": "_value",
        "value": "該当組織の Block コード"
      },
      {
        "key": "_ver",
        "value": 1
      }
    ]
  },
  {
    "key": "other-block",
    "value": null
  },
  {
    "key": "region",
    "value": null
  },
  {
    "key": "statement",
    "value": [
      {
        "key": "title",
        "value": "組織ステートメント"
      },
      {
        "key": "section",
        "value": [
          {
            "key": "title",
            "value": "組織概要タイトル"
          },
          {
            "key": "content",
            "value": [
              {
                "key": "sentence",
                "value": "組織概要本文"
              }
            ]
          }
        ]
      }
    ]
  },
  {
    "key": "status",
    "value": null
  },
  {
    "key": "trader-alliance",
    "value": null
  }
]
}, "inner": null,
"attribute": null
},

```

```
"isDraft": true
}
```

名称	説明
PxR-Root-Block のアクターコード	申請先の PxR-Root-Block カタログのコードを設定する。 (「3.1.1Tips.カタログの特定方法 (P15)」を参照し、NS「catalog/ext/{ext_name}/actor/pxr-root」で取得する)
PxR-Root-Block のアクターバージョン	申請先の PxR-Root-Block カタログのバージョンを設定する。 (「3.1.1Tips.カタログの特定方法 (P15)」を参照し、NS「catalog/ext/{ext_name}/actor/pxr-root」で取得する)
catalog/ext/{ext_name}/actor/{actor_type}	<ul style="list-style-type: none"> ・ { ext_name } 設定されている ext 名に置き換える。 <ul style="list-style-type: none"> ・ { actor_type } 対象のアクターのタイプに置き換える。 アプリケーション P : app 領域運営 SP : region-root
組織名	申請対象アクターの組織名を設定する。
組織の概要	申請対象アクターの概要を設定する。
継承元カタログコード	継承元となるカタログのコードを設定する。 (「3.1.1Tips.カタログの特定方法 (P15)」を参照し、NS「catalog/model/actor/{actor_type}」で取得する)
継承元カタログバージョン	継承元となるカタログのバージョンを設定する。 (「3.1.1Tips.カタログの特定方法 (P15)」を参照し、NS「catalog/model/actor/{actor_type}」で取得する)

3. PxR-Root-Block のアクターにアクター認定申請を実施したことを連絡する。

3.6 アクター認定承認、クライアント証明書の発行

PxR-Root-Block のアクターが、認定申請に対する認定承認、およびクライアント証明書のダウンロードを行う。※クライアント証明書は申請したアクターに配布する。

※PxR-Root-Block にログインセッションを確立して実施する。

1. Root-Block にログインする。

サービス名称	API 名称	メソッドおよびパス
オペレーターサービス	ログイン	POST /operator/login

リクエスト

```
{
  "type": 3,
  "loginId": "ログイン ID",
  "hpassword": "ハッシュパスワード"
}
```

※ハッシュパスワードは、パスワードを、「構築ガイド」の 2.4 マニフェスト作成、適用で Configmap に設定したソルト値とハッシュ化回数でハッシュ化 (SHA-256) したパスワード

2. PxR-Root-Block でアクター認定申請通知を取得する。

承認対象の通知の ID を取得する。

サービス名称	API 名称	メソッドおよびパス
通知サービス	取得	GET /notification/?is_send=false&is_unread=false&is_approval=false&type=1&num=0 ※proxy を経由するため以下のパスになります。 /pxr-block-proxy/pxr-block-proxy/?path=notification%2F%3Fis_send%3Dfalse%26is_unread%3Dfalse%26is_approval%3Dfalse%26type%3D1%26num%3D0

レスポンス例

```
[
  {
    "id": 1,
    "type": 1,
    "title": "アクター認定申請",
    "content": "アクター認定の申請です。認定または否認をお願いします。",
    "attribute": {},
    "category": {
      "_value": 117,
      "_ver": 1
    },
    "from": {
      "blockCode": "申請元の Block コード",
      "operatorId": 1,
      "actor": {
        "_value": null,
        "_ver": null
      }
    },
    "approval": {
```

```

"operatorId": 0,
"status": 1,
"approvalAt": "2022-00-00T00: 00: 00. 000+0900",
"expirationAt": "2022-00-00T00: 00: 00.000+0900"
},
"readAt": "2022-00-00T00: 00: 00. 000+0900",
"sendAt": "2022-00-00T00: 00: 00. 000+0900",
"is_transfer": false
}
]

```

3. アクター認定申請を承認する。

サービス名称	API 名称	メソッドおよびパス
通知サービス	承認要求への承認	PUT /notification/approval'

リクエスト

```

{
  "id": "取得した通知 ID",
  "status": 1,
  "comment": "任意のコメント"
}

```

4. クライアント証明書を発行するための必要情報を取得する。

サービス名称	API 名称	メソッドおよびパス
カタログ更新サービス	申請取得	GET /catalog-update/actor?code={block_code}&ver=1&actorType={actor_type}&approved=true ※proxy を経由するため以下のパスになります。 /pxr-block-proxy/pxr-block-proxy/?path=%2Fcatalog-update%2Ffactor%3Fcode%3D{block_code}%26ver%3D1%26actorType%3D{actor_type}%26approved%3Dtrue

- ・ {block_code}
Block 作成時に確認したカタログコードに置き換える。
- ・ {actor_type}
対象のアクターのタイプに置き換える。
アプリケーション P : app
領域運営 SP : region-root

レスポンス (attributes の中身がクライアント証明書を発行するための必要情報)

```

{
  "id": 1,
  "type": 1,
  "approvalActor": {
    "_value": xxxxxx,
    "_ver": 1
  },
  "actorCatalog": --- 省略 ---,
  "expireAt": "2022-00-00T00:00: 00.000+0900",
}

```

```

"isDraft": false,
"status": 1,
"authCode": "0TxGntOS60M3",
"applicantDate": "2022-00-00T00:00: 00.000+0900",
"comment": null,
"approver": "rootuser",
"approvalAt": "2022-00-00T00:00: 00.000+0900'",
"attributes": {
  "actorCode": 1001020,
  "serialNo": "XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX",
  "fingerPrint": "XX:XX:XX:XX:XX:XX:XX:XX"
}
}

```

5. 認定したアクターのアクターコードを取得する。

サービス名称	API 名称	メソッドおよびパス
カタログサービス	取得	GET /catalog?ns={NS}

※{NS}は catalog/ext/{ext_name}/actor/{actor_type}

・ { ext_name }

設定されている ext 名に置き換える。

・ { actor_type }

対象のアクターのタイプに置き換える。

アプリケーション P : app

領域運営 SP : region-root

レスポンス例

```

[
  {
    "catalogItem": {
      "ns": " catalog/ext/{ext_name}/actor/{actor_type}",
      "name": "カタログ名称",
      "_code": {
        "_value": xxxxxxxx,
        "_ver": x
      },
      "inherit": {
        "_value": xx,
        "_ver": x
      },
      "description": "カタログ概要"
    },
    "template": --- 省略 ---,
    "prop": --- 省略 ---,
    "value": --- 省略 ---
    "attribute": null
  }
]

```

6. PxR-Root-Block でクライアント証明書をダウンロードする。

サービス名称	API 名称	メソッドおよびパス
認証局サービス	クライアント証明書取得	GET /certification-authority/client/{serialNo}/{fingerPrint} ※proxy を経由するため以下のパスになります。 /pxr-block-proxy/pxr-block-proxy/?path=/certification-authority/client/{serialNo}/{fingerPrint}

- ・ {serialNo}

必要情報の attributes 内 serialNo に置き換える。

- ・ {ingerPrint}

必要情報の attributes 内 fingerPrint に置き換える。

レスポンスの内容（クライアント証明書）を保存する。

7. 追加 PxR-Root-Block のアクターに、クライアント証明書の配布および証明書アップロードの実施を行うよう連絡する。

3.7 クライアント証明書のアップロード

追加 PxR-Block のアクターが、配布されたアクター証明書をアップロードする。

※PxR-Root-Block から配布されたクライアント証明書を使用する。

1. クライアント証明書をアップロードするため、追加 PxR-Block にログインする。

サービス名称	API 名称	メソッドおよびパス
オペレーターサービス	ログイン	POST /operator/login

リクエスト

```
{
  "type": 3,
  "loginId": "ログイン ID",
  "hpassword": "ハッシュパスワード"
}
```

※ハッシュパスワードは、パスワードを、「構築ガイド」の 2.4 マニフェスト作成、適用で Configmap に設定したソルト値とハッシュ化回数でハッシュ化（SHA-256）したパスワード

2. クライアント証明書をアップロードする。

サービス名称	API 名称	メソッドおよびパス
証明書管理サービス	証明書保存	POST /certificate-manage/ ※proxy を経由するため以下のパスになります。 POST /pxr-block-proxy/pxr-block-proxy/?block=\$pxr-root-block&path=/certificate-manage/

※リクエストは「3.6 アクター認定承認、クライアント証明書の発行」で保存した内容（クライアント証明書）を使用する。

3. クライアント証明書のアップロードが完了したことを、PxR-Root-Block のアクターに連絡する。

3.8 PF 設定 : EKS マニフェストアクターコード設定

確定したアクターコードを EKS マニフェストに設定する。

1. Configmap/common-configmap.yaml を編集し、追加した PxR-Block のアクターコードを設定する。

<アクターコード>、_Ver は、「3.6 アクター認定」で取得した actor_code を設定する。

```
~~~以前省略~~~
apiVersion: v1
kind: ConfigMap
metadata:
  name: application000004-common-config-map
  namespace: pxr
data:
  block-common-conf.json: |
    {
      "block": {
        "_value": 1001020,
        "_ver": 1
      },
      "actor": {
        "_value": <アクターコード>,
        "_ver": 1
      },
    }
  }
~~~以降省略~~~
```

2. Kubernetes の Configmap の適用、Pod の再起動を行う。

```
# kubectl apply -f . -R
# kubectl rollout restart -n pxr deployments/<追加 Block の deployment 名>
```

3. Kubernetes の Pod ステータス一覧から追加 PxR-Block が、Running（動作中）に変われば再起動完了となる。

※Pod ステータス一覧レスポンス例

```
# kubectl get Pod -n pxr
application000004-api-df4dbb5f5-pgxnw 10/10 Running 0 13h
region000001-api-6884664976-zzbgg 7/7 Running 0 13h
root-api-86df7cffb9-btrsc 17/17 Running 0 13h
```

4 サービス設定

本章では、本モジュールにおけるグローバル設定の変更、運営メンバーの追加の手順について記載する。

※グローバル設定の変更は、初期カタログ設定値から変更する場合に実施する。

4.1 グローバル設定（Block 共通設定）

グローバル設定を初期カタログ設定値から変更する場合に、PxR-Root-Block にログインセッションを確立して実施する。

1. PxR-Root-Block にログインする。

サービス名称	API 名称	メソッドおよびパス
オペレーターサービス	ログイン	POST /operator/login

リクエスト

```
{
  "type": 3,
  "loginId": "ログイン ID",
  "hpassword": "ハッシュパスワード"
}
```

※ハッシュパスワードは、パスワードを「構築ガイド」の 2.4 マニフェスト作成、適用で Configmap に設定したソルト値とハッシュ化回数でハッシュ化（SHA-256）したパスワード

ログイン後の API 使用について

ログイン後に各 API を使用するためにはログイン時に返却される「sessionId」と「XSRF-TOKEN」を設定する必要がある。

a. sessionId

ログイン API のレスポンス内で各 API を使用するための「sessionId」が返却される。

例：ログインのレスポンス

```
{
  "sessionId": "cab7561a298146422016cece0e892c3e93287b98bd24357efebcb2d29195eada",
  "operatorId": 1,
  "type": 3,
  "loginId": "ログイン ID",
  "name": "管理者",
  "auth": --- 省略 ---,
  "lastLoginAt": "2022-06-21T15:53:43.226+0900",
  "passwordChangedFlg": true,
  "loginProhibitedFlg": false,
  "attributes": {
    "smsAuth": false
  },
  "block": {
    "_value": 1000401,
    "_ver": 1
  },
  "actor": {
    "_value": 1000431,
```

```

    "_ver": 1
  }
}

```

各 API を使用する際にリクエストの cookie に以下の形式で設定する。

```
cookie: operator_type3_session="発行された sessionId"
```

b. XSRF-TOKEN

ログイン API のレスポンスの set-cookie で各 API を使用するための「XSRF-TOKEN」が返却される。

例：

```
set-cookie: XSRF-TOKEN=wNS5VFAK-tryomg1zs1omhd7tk2fkgjfmrk;
```

各 API を使用する際にリクエストの header に以下の形式で設定する。

```
x-xsrf-token: "発行された XSRF-TOKEN"
```

2. PxR-Root-Block でカタログ変更 API を実行する。

本手順により各アクターに割り当てられる Block のカタログを変更する。

サービス名称	API 名称	メソッドおよびパス
カタログサービス	更新	PUT /catalog/ext/{code}

リクエスト (template の内容を設定する)

```

{
  "catalogItem": {
    "ns": "catalog/ext/{ext 名}/setting/global",
    "name": グローバル設定のカタログ名称,
    "_code": null,
    "inherit": {
      "_value": 継承元カタログコード,
      "_ver": 継承元カタログバージョン
    },
    "description": グローバル設定のカタログ概要
  },
  "template": {
    "value": [
      {
        "key": "_code",
        "value": [
          {
            "key": "_value",
            "value": 1000374
          },
          {
            "key": "_ver",
            "value": 1
          }
        ]
      }
    ]
  }
}

```

```

    "key": "account-lock-count",
    "value": 6
  },
  {
    "key": "account-lock-release-time",
    "value": [
      {
        "key": "type",
        "value": "minute"
      },
      {
        "key": "value",
        "value": 30
      }
    ]
  },
  {
    "key": "book-open-code-expiration",
    "value": [
      {
        "key": "type",
        "value": "minute"
      },
      {
        "key": "value",
        "value": 10
      }
    ]
  },
  {
    "key": "book-open-notification-interval",
    "value": [
      {
        "key": "type",
        "value": "day"
      },
      {
        "key": "value",
        "value": 1
      }
    ]
  },
  {
    "key": "book_create_sms_message",
    "value": "%s?ID=%s パスワードは次のメッセージでお送りします"
  },
  {
    "key": "book_deletion_pending_term",
    "value": [
      {
        "key": "type",
        "value": "day"
      },
      {
        "key": "value",
        "value": 14
      }
    ]
  },
  {
    "key": "data_download_term_expiration",
    "value": [
      {
        "key": "type",
        "value": "day"
      }
    ]
  },

```

```

    {
      "key": "value",
      "value": 14
    }
  ],
},
{
  "key": "identity-verification-expiration",
  "value": [
    {
      "key": "type",
      "value": "day"
    },
    {
      "key": "value",
      "value": 7
    }
  ]
},
{
  "key": "login_sms_message",
  "value": "Your code is %s"
},
{
  "key": "management_id_format",
  "value": "^(?=.*[A-Za-z])(?=.*¥¥d)[A-Za-z¥¥d]{8,}$"
},
{
  "key": "management_password_similarity_check",
  "value": true
},
{
  "key": "min_period_for_platform-tou_re-consent",
  "value": [
    {
      "key": "type",
      "value": "day"
    },
    {
      "key": "value",
      "value": 7
    }
  ]
},
{
  "key": "min_period_for_region-tou_re-consent",
  "value": [
    {
      "key": "type",
      "value": "day"
    },
    {
      "key": "value",
      "value": 7
    }
  ]
},
{
  "key": "one-time-login-code-expiration",
  "value": null
},
{
  "key": "open_book_automatically",
  "value": true
},
{

```

```

"key": "password-expiration",
"value": [
  {
    "key": "type",
    "value": "day"
  },
  {
    "key": "value",
    "value": 90
  }
]
},
{
"key": "password-generations-number",
"value": 4
},
{
"key": "personal_account_delete",
"value": false
},
{
"key": "personal_disassociation",
"value": true
},
{
"key": "personal_share_basic_policy",
"value": false
},
{
"key": "personal_two-step_verification",
"value": true
},
{
"key": "presigned_url_expiration",
"value": [
  {
    "key": "type",
    "value": "second"
  },
  {
    "key": "value",
    "value": 30
  }
]
},
{
"key": "pxr_id_format",
"value": "^(?=.*[A-Za-z])(?=.*¥¥d)[A-Za-z¥¥d]{8,}$"
},
{
"key": "pxr_id_password_format",
"value": "^(?=.*[A-Za-z])(?=.*¥¥d)[A-Za-z¥¥d]{12,}$"
},
{
"key": "pxr_id_password_similarity_check",
"value": true
},
{
"key": "pxr_id_prefix",
"value": ""
},
{
"key": "pxr_id_suffix",
"value": ""
},
{

```

```

"key": "region-tou_re-consent_notification_interval",
"value": [
  {
    "key": "type",
    "value": "day"
  },
  {
    "key": "value",
    "value": 3
  }
]
},
{
"key": "search_target_ns",
"value": [
  {
    "key": "name",
    "value": "データカテゴリ"
  },
  {
    "key": "ns",
    "value": "catalog/ext/{extName}/attribute/category/data"
  }
]
},
{
"key": "service_category_for_data_category",
"value": [
  {
    "key": "service",
    "value": [
      {
        "key": "_value",
        "value": 1000065
      },
      {
        "key": "_ver",
        "value": 1
      }
    ]
  },
  {
    "key": "dataCategory",
    "value": [
      {
        "key": "_value",
        "value": 1000137
      },
      {
        "key": "_ver",
        "value": 1
      }
    ]
  }
]
},
{
"key": "service_category_for_workflow",
"value": [
  {
    "key": "service",
    "value": [
      {
        "key": "_value",
        "value": 1000065
      }
    ]
  }
]
}

```

```

    {
      "key": "_ver",
      "value": 1
    }
  ],
},
{
  "key": "workflow_p",
  "value": [
    {
      "key": "_value",
      "value": 1000438
    },
    {
      "key": "_ver",
      "value": 1
    }
  ]
},
{
  "key": "workflow",
  "value": [
    {
      "key": "_value",
      "value": 1000481
    },
    {
      "key": "_ver",
      "value": 1
    }
  ]
}
],
},
{
  "key": "session-expiration",
  "value": [
    {
      "key": "type",
      "value": "hour"
    },
    {
      "key": "value",
      "value": 3
    }
  ]
},
{
  "key": "use_app-p",
  "value": false
},
{
  "key": "use_id_connect",
  "value": false
},
{
  "key": "use_region_service_operation",
  "value": false
},
{
  "key": "use_share",
  "value": false
},
{
  "key": "use_app-p",
  "value": true
}

```

```

    }
  ]
}, "attribute": null
}

```

表 4-1 説明 (リクエストの template の項目)

設定項目	説明
継承元カタログコード	継承元となるカタログのコードを設定する。 初期カタログから変更がない場合は「160」 (「3.1.1Tips. カタログの特定方法 (P15)」を参照し、NS「catalog/model/setting/global」で取得する)
継承元カタログバージョン	継承元となるカタログのバージョンを設定する。 初期カタログから変更がない場合は「1」 (「3.1.1Tips. カタログの特定方法 (P15)」を参照し、NS「catalog/model/setting/global」で取得する)
account-lock-count	PxR-Block ログイン時のパスワード入力ミスを何回まで許容するかを指定する。
account-lock-release-time	アカウントロック解除までの時間を指定する。
book_create_sms_message	My-Condition-Book 開設時に送信される SMS メッセージ内容を指定する。
book_deletion_pending_term	My-Condition-Book の削除保留期間を指定する。
identity-verification-expiration	本人性確認コードの有効期限を指定する。
login_sms_message	個人による PxR-Root-Block ログイン時に送信される SMS メッセージ内容を指定する。
management_id_format	運営メンバーの ID フォーマットを正規表現で指定する。
management_id_format_errormessage	運営メンバーの ID 入力時のエラーメッセージを指定する。
management_password_format	運営メンバーのパスワードフォーマットを正規表現で指定する。
management_password_similarity_check	運営メンバーのパスワード類似性チェック有無を指定する。
min_period_for_platform-tou_re-consent	プラットフォーム利用規約の再同意期限の最低期間を指定する。
min_period_for_region-tou_re-consent	リージョン利用規約の再同意期限の最低期間を指定する。
one-time-login-code-expiration	個人による PxR-Root-Block ログイン時のワンタイムログインコードを発行してからの有効期限を指定する。
password-expiration	運営メンバー・個人のパスワード有効期限を指定する。
password-generations-number	運営メンバー・個人のパスワード更新時について、過去何世代まで同じパスワードを許可しないかを指定する。
personal_account_delete	個人による自身のアカウント削除を許可するかを指定する。
personal_disassociation	個人の連携解除の使用可否を指定する。
personal_share_basic_policy	個人の共有の基本方針可否設定を指定する。
personal_two-step_verification	個人による PxR-Root-Block ログイン時の 2 段階認証解除を許可するかを指定する。
pxr_id_format	PxR-ID (個人の ID) フォーマットを指定する。
pxr_id_format_errormessage	PxR-ID (個人の ID) フォーマットエラーメッセージを指定する。
pxr_id_password_format	PxR-ID (個人の ID) のパスワードフォーマットを指定する。
pxr_id_password_similarity_check	PxR-ID のパスワード類似性チェックの有無を指定する。

pxr_id_prefix	PxR-ID（個人の ID）（個人の ID）の prefix を指定する。
pxr_id_suffix	PxR-ID（個人の ID）の suffix を指定する。
region-tou_re-consent_notification_interval	領域運営サービスプロバイダーのリージョン利用規約通知間隔を指定する。
session-expiration	セッション有効期限を指定する。
use_app-p	アプリケーションプロバイダーの使用可否を指定する。
use_region_service_operation	リージョンサービスの運用有無の設定を指定する。
use_share	ドキュメント共有の使用設定を指定する。

4.2 運営メンバーの追加

PxR-Block にメンバー追加権限を持つユーザー（初期ユーザー）を使い、他運営メンバーを追加する手順について記載する。

※本手順は対象 PxR-Block を起動した状態で実施する。

1. 運営メンバーを追加する対象の Block にログインする。

サービス名称	API 名称	メソッドおよびパス
オペレーターサービス	ログイン	POST /operator/login

リクエスト

```
{
  "type": 3,
  "loginId": "ログイン ID",
  "hpassword": "ハッシュパスワード"
}
```

※ハッシュパスワードは、パスワードを、「構築ガイド」の 2.4 マニフェスト作成、適用で Configmap に設定したソルト値とハッシュ化回数でハッシュ化（SHA-256）したパスワード

2. PxR-Root-Block でカタログ変更 API を実行する。

本手順により各アクターの設定カタログを変更する。

サービス名称	API 名称	メソッドおよびパス
オペレーターサービス	追加	POST /operator/

リクエスト（流通制御の場合）

```
{
  "type": 3,
  "loginId": "ログイン ID",
  "hpassword": "ハッシュパスワード",
  "name": "名称",
  "attributes": null,
  "roles": null,
  "auth": {
    "member": {
      "add": true,

```

```

    "update": true,
    "delete": true
  },
  "book": {
    "create": true
  },
  "actor": {
    "application": true,
    "approval": true
  },
  "catalog": {
    "create": true
  },
  "setting": {
    "update": true
  }
}
}
}

```

リクエスト (領域運営の場合)

```

{
  "type": 3,
  "loginId": "ログイン ID",
  "hpassword": "ハッシュパスワード",
  "name": "名称",
  "attributes": null,
  "roles": null,
  "auth": {
    "member": {
      "add": true,
      "update": true,
      "delete": true
    },
  },
  "actor": {
    "application": true,
    "approval": true
  },
  "app-wf-user": {
    "create": true
  },
  "catalog": {
    "create": true
  },
  "join": {
    "application": true,
    "approval": true
  },
  "alliance": {
    "application": true,
    "approval": true
  },
  "setting": {
    "update": true
  }
}
}
}

```

リクエスト (アプリケーションの場合)

```

{
  "type": 3,
  "loginId": "ログイン ID",
  "hpassword": "ハッシュパスワード",
  "name": "名称",

```

```
"attributes": null,  
"roles": null,  
"auth": {  
  "member": {  
    "add": true,  
    "update": true,  
    "delete": true  
  },  
  "catalog": {  
    "create": true  
  },  
  "actor": {  
    "application": true  
  },  
  "app-wf-user": {  
    "create": true  
  },  
  "join": {  
    "application": true,  
    "approval": true  
  }  
}  
}
```

5 PxR-Block の削除方法

本モジュールにおける PxR-Block 削除は、以下フロー図の流れで削除処理を行う。

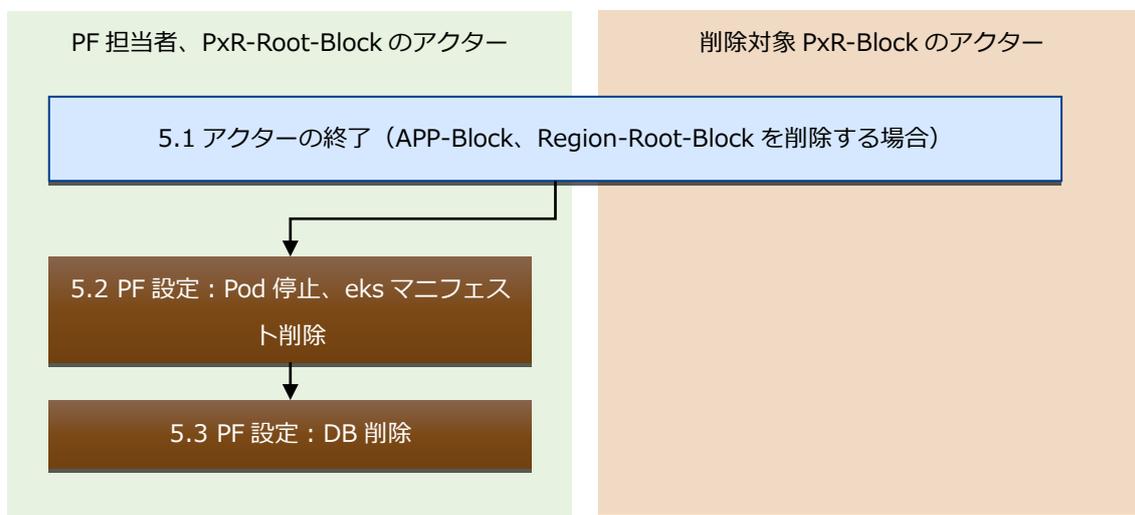


図 5-1 PxR-Block 削除フロー

5.1 アクターの終了

各 PxR-Block を削除する準備として Region 参加停止、利用者 ID 連携解除、アクター認定解除を行う。
 ※削除対象が PxR-Root-Block の場合は、全サービスを終了するためすべての PxR-Block の Pod 停止、eks マニフェストの削除、DB 削除を行うため、アクターの終了は実施不要である。

削除対象が Region-Root-Block の場合、下図の流れでアクターの終了手続きを行う。

※APP-Block のアクター終了を行う場合は、「5.1.3 APP 利用者 ID 連携の解除（APP-Block 終了時のみ）
 ～5.1.6 アクター認定解除（削除対象 APP-Block）」を実施する。

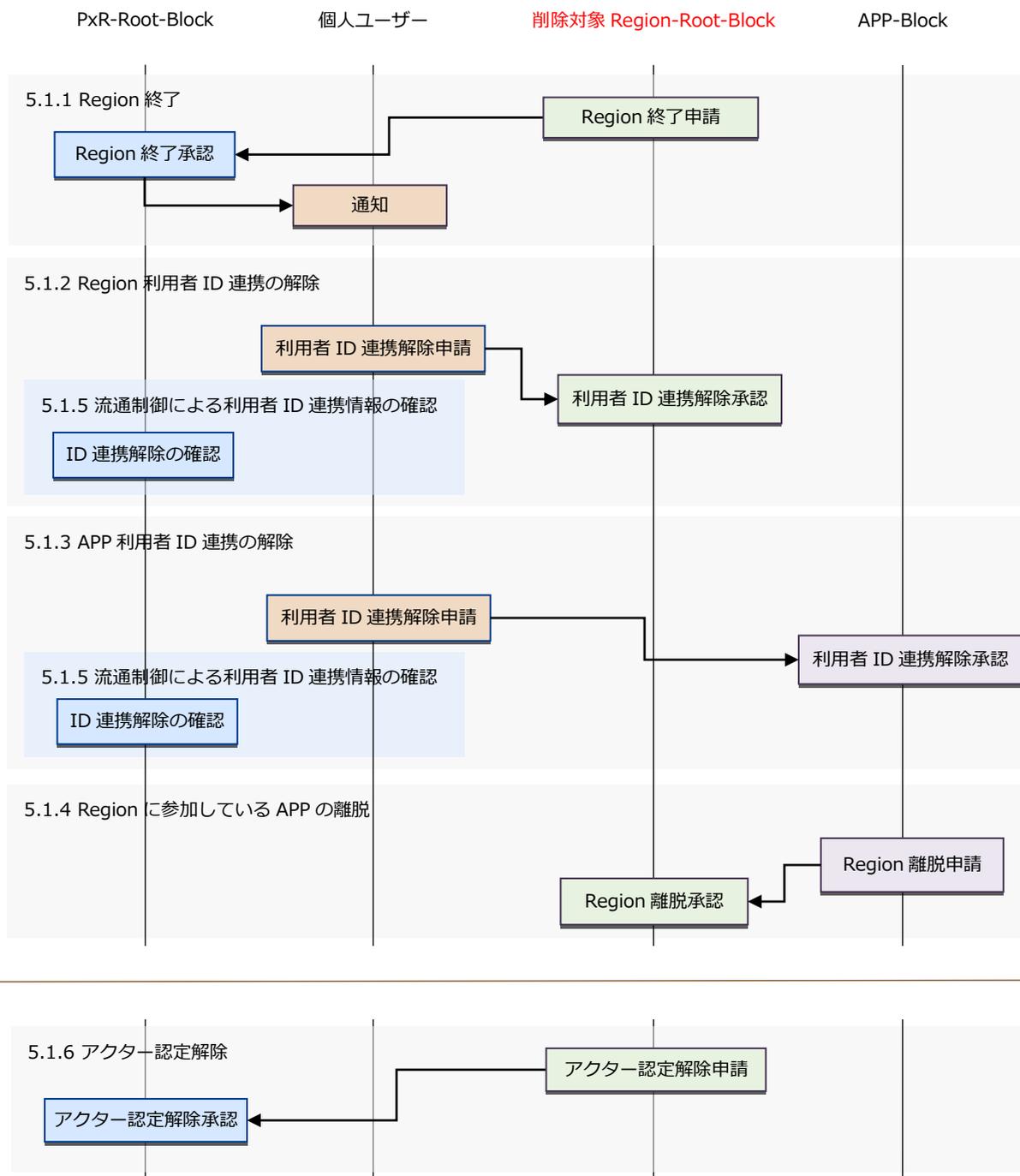


図 5-2 アクターの終了フロー（Region-Root-Block 編）

5.1.1 Region 終了

削除対象の Region-Root-Block から Region 終了申請を行い、PxR-Root-Block で承認する。

※削除対象の Region-Root-Block で作成しているすべての Region の終了処理を行う。

1. Region 終了申請

a. 削除対象の Region-Root-Block にログインする。

サービス名称	API 名称	メソッドおよびパス
オペレーターサービス	ログイン	POST /operator/login

リクエスト

```
{
  "type": 3,
  "loginId": "ログイン ID",
  "hpassword": "ハッシュパスワード"
}
```

※ハッシュパスワードは、パスワードを、「構築ガイド」の 2.4 マニフェスト作成、適用で Configmap に設定したソルト値とハッシュ化回数でハッシュ化 (SHA-256) したパスワード

b. カタログを取得して、組織で作成した Region の一覧を取得する。

サービス名称	API 名称	メソッドおよびパス
カタログサービス	取得	GET /catalog/{code} ※proxy を経由するため以下のパスになります。 POST /pxr-block-proxy/pxr-block-proxy/?block=\$pxr-root-block&path=/catalog/{code}

※{code}は該当の領域運営アクターのカatalogコードです

レスポンス例

```
{
  "catalogItem": {
    "ns": "catalog/ext/xxxxx/actor/region-root",
    "name": "領域運営アクター",
    "_code": {
      "_value": 1000432,
      "_ver": 4
    },
    "inherit": {
      "_value": 49,
      "_ver": 1
    },
    "description": "領域運営アクターの概要"
  },
  "template": {
    "_code": {
      "_value": 1000432,
      "_ver": 4
    },
    "breakaway-flg": false,
    "category": null,
    "information-site": null,
    "main-block": {
```

```

    "_value": 1000402,
    "_ver": 1
  },
  "other-block": null,
  "region": [
    {
      "_value": 1000451,
      "_ver": 1
    },
    {
      "_value": 1000452,
      "_ver": 1
    }
  ],
  "statement": --- 省略 ---,
  "status": [
    {
      "status": "certified",
      "by": {
        "_value": 1000431,
        "_ver": 1
      },
      "at": "2020-01-01T00:00:00.000+0900"
    }
  ],
  "trader-alliance": [
    {
      "_value": 1000435,
      "_ver": 1
    }
  ]
],
"prop": --- 省略 ---,
"value": --- 省略 ---,
"attribute": null
}

```

組織の持つ Region のコード

c. Region 終了申請を実施する。

サービス名称	API 名称	メソッドおよびパス
カタログ更新サービス	Region 終了申請	POST /region/status/end ※proxy を経由するため以下のパスになります。 POST /pxr-block-proxy/pxr-block-proxy/?block=\$pxr-root-block&path=/region/status/end

リクエスト

```

{
  "regionCode": {
    "_value": 終了する Region のカタログコード,
    "_ver": 終了する Region のカタログバージョン
  },
  "endDate": "終了日(2022-01-01T11:11:11.000+0900)",
  "requestComment": "任意のコメント"
}

```

※終了日に設定した日時に Region が終了する

2. Region 終了申請の承認

a. PxR-Root-Block にログインする。

サービス名称	API 名称	メソッドおよびパス
オペレーターサービス	ログイン	POST /operator/login

リクエスト

```
{
  "type": 3,
  "loginId": "ログイン ID",
  "hpassword": "ハッシュパスワード"
}
```

※ハッシュパスワードは、パスワードを、「構築ガイド」の 2.4 マニフェスト作成、適用で Configmap に設定したソルト値とハッシュ化回数でハッシュ化（SHA-256）したパスワード

b. Region 終了申請の通知 ID を取得する。

サービス名称	API 名称	メソッドおよびパス
通知サービス	取得	GET /notification/?is_send=false&is_unread=false&is_approval=false&type=1&num=0 ※proxy を経由するため以下のパスになります。 POST /pxr-block-proxy/pxr-block-proxy/?path=/notification/?is_send=false&is_unread=false&is_approval=false&type=1&num=0

レスポンス例

```
[
  {
    "id": 34,
    "type": 1,
    "title": "Region 終了申請",
    "content": "Region 終了を要求します",
    "attribute": {
      "region": 1001059,
      "actor": 1001021,
      "wf": [
        1001054
      ]
    },
    "category": {
      "_value": 120,
      "_ver": 1
    },
    "from": {
      "blockCode": 1001018,
      "operatorId": 34,
      "actor": {
        "_value": 1001021,
        "_ver": 1
      }
    },
    "approval": {
      "operatorId": 0,
      "status": 1,
      "approvalAt": "2022-06-09T14:24:26.600+0900",
      "expirationAt": "2022-06-16T14:22:40.038+0900"
    },
    "readAt": null,
    "sendAt": "2022-06-09T14:22:40.039+0900",
    "is_transfer": false
  }
]
```

```
}  
]
```

c. Region 離脱申請を承認する。

サービス名称	API 名称	メソッドおよびパス
通知サービス	承認要求への承認	PUT /notification/approval ※proxy を経由するため以下のパスになります。 POST /pxr-block-proxy/pxr-block-proxy/?block=\$pxr-root-block&path=/notification/approval

リクエスト

```
{  
  id: "取得した通知 ID",  
  status: 1,  
  comment: "任意のコメント"  
}
```

3. Region に参加している利用者に Region 終了の通知 (SMS) がされる。

5.1.2 Region 利用者 ID 連携の解除

削除対象の Region-Root-Block に連携されている利用者 ID の連携解除を行う。(「5.1.1 Region 終了」により対象の個人に通知される)

※すべての連携済み利用者 ID に対して連携解除を行う。

1. 利用者 ID 連携解除申請

a. 利用者 ID 連携の解除申請のため、PxR-Root-Block に個人アカウントでログインする。

サービス名称	API 名称	メソッドおよびパス
オペレーターサービス	ログイン	POST /operator/ind/login

リクエスト

```
{  
  "type": 0,  
  "loginId": "ログイン ID",  
  "hpassword": "ハッシュパスワード"  
}
```

※ハッシュパスワードは、パスワードを、「構築ガイド」の 2.4 マニフェスト作成、適用で Configmap に設定したソルト値とハッシュ化回数でハッシュ化 (SHA-256) したパスワード

b. PxR-Root-Block で個人が利用者 ID 連携の解除申請を実施する。

サービス名称	API 名称	メソッドおよびパス
本人性確認サービス	本人性確認コード発行	POST /identity-verify/code ※proxy を経由するため以下のパスになります。

		/pxr-block-proxy/pxr-block-proxy/pxr-block-proxy/?path=/identity-verify/code
--	--	--

リクエスト

```
{
  "actor": {
    "_value": 連携解除申請を送付する組織のカタログコード,
    "_ver": 連携解除申請を送付する組織のカタログバージョン
  },
  "region": {
    "_value": 連携解除申請を送付する Region のカタログコード,
    "_ver": 連携解除申請を送付する Region のカタログバージョン
  },
  "app": null,
  "wf": null
}
```

2. 利用者 ID 連携解除申請の承認

a. Region-Root-Block にログインする。

サービス名称	API 名称	メソッドおよびパス
オペレーターサービス	ログイン	POST /operator/login

リクエスト

```
{
  "type": 3,
  "loginId": "ログイン ID",
  "hpassword": "ハッシュパスワード"
}
```

※ハッシュパスワードは、パスワードを、「構築ガイド」の 2.4 マニフェスト作成、適用で Configmap に設定したソルト値とハッシュ化回数でハッシュ化 (SHA-256) したパスワード

b. 利用者 ID 連携の解除申請の通知を取得する。

承認対象の通知の ID を取得する。

サービス名称	API 名称	メソッドおよびパス
通知サービス	取得	GET /notification/?is_send=false&is_unread=false&is_approval=false&type=1&num=0 ※proxy を経由するため以下のパスになります。 POST /pxr-block-proxy /pxr-block-proxy/?block=\$pxr-root-block&path=/notification/?is_send=false&is_unread=false&is_approval=false&type=1&num=0

レスポンス例

```
[
  {
    "id": 34,
    "type": 1,
    "title": "利用者 ID 連携解除",
    "content": "利用者 ID 連携解除を要求します",
    "attribute": {
      "region": 1001059,
      "actor": 1001021,
    }
  }
]
```

通知 ID

```

    "wf": [
      1001054
    ],
  },
  "category": {
    "_value": 120,
    "_ver": 1
  },
  "from": {
    "blockCode": 1001018,
    "operatorId": 34,
    "actor": {
      "_value": 1001021,
      "_ver": 1
    }
  },
  "approval": {
    "operatorId": 0,
    "status": 1,
    "approvalAt": "2022-06-09T14:24:26.600+0900",
    "expirationAt": "2022-06-16T14:22:40.038+0900"
  },
  "readAt": null,
  "sendAt": "2022-06-09T14:22:40.039+0900",
  "is_transfer": false
}
]

```

c. Region-Root-Block で利用者 ID 連携解除申請を承認する。

サービス名称	API 名称	メソッドおよびパス
通知サービス	承認要求への承認	PUT /notification/approval ※proxy を経由するため以下のパスになります。 PUT /pxr-block-proxy/pxr-block-proxy/?block=\$pxr-root-block&path=/notification/approval

リクエスト

```

{
  id: "取得した通知 ID",
  status: 1,
  comment: "任意のコメント"
}

```

5.1.3 APP 利用者 ID 連携の解除

削除対象の APP-Block に連携されている利用者 ID の連携解除を行う。

※すべての連携済み利用者 ID に対して連携解除を行う。

1. 利用者 ID 連携解除申請

a. 利用者 ID 連携の解除申請のため、PxR-Root-Block に個人アカウントでログインする。

サービス名称	API 名称	メソッドおよびパス
オペレーターサービス	ログイン	POST /operator/ind/login

リクエスト

```
{
  "type": 0,
  "loginId": "ログイン ID",
  "hpassword": "ハッシュパスワード"
}
```

※ハッシュパスワードは、パスワードを、「構築ガイド」の 2.4 マニフェスト作成、適用で Configmap に設定したソルト値とハッシュ化回数でハッシュ化 (SHA-256) したパスワード

b. PxR-Root-Block で個人が利用者 ID 連携の解除申請を実施する。

サービス名称	API 名称	メソッドおよびパス
本人性確認サービス	本人性確認コード発行	POST /identity-verify/code ※proxy を経由するため以下のパスになります。 /pxr-block-proxy/pxr-block-proxy/?path=/identity-verify/code

リクエスト

```
{
  "actor": {
    "_value": "連携解除申請を送付する組織のカタログコード",
    "_ver": "連携解除申請を送付する組織のカタログバージョン"
  },
  "app": {
    "_value": "連携解除申請を送付するアプリケーションのカタログコード",
    "_ver": "連携解除申請を送付するアプリケーションのカタログバージョン"
  },
  "wf": null
}
```

2. 利用者 ID 連携解除申請の承認

a. APP-Block にログインする。

サービス名称	API 名称	メソッドおよびパス
オペレーターサービス	ログイン	POST /operator/login

リクエスト

```
{
  "type": 3,
  "loginId": "ログイン ID",
  "hpassword": "ハッシュパスワード"
}
```

※ハッシュパスワードは、パスワードを、「構築ガイド」の 2.4 マニフェスト作成、適用で Configmap に設定したソルト値とハッシュ化回数でハッシュ化 (SHA-256) したパスワード

b. 利用者 ID 連携の解除申請の通知を取得する。

承認対象の通知 ID を取得する。

サービス名称	API 名称	メソッドおよびパス
通知サービス	取得	GET /notification/?is_send=false&is_unread=false&

		is_approval=false&type=1&num=0 ※proxy を経由するため以下のパスになります。 POST /pxr-block-proxy/pxr-block-proxy/?block=\$pxr-root-block&path=/notification/?is_send=false&is_unread=false&is_approval=false&type=1&num=0
--	--	--

レスポンス例

```
[
  {
    "id": 34,
    "type": 1,
    "title": "利用者 ID 連携解除",
    "content": "利用者 ID 連携解除を要求します",
    "attribute": {
      "region": 1001059,
      "actor": 1001021,
      "wf": [
        1001054
      ]
    },
    "category": {
      "_value": 120,
      "_ver": 1
    },
    "from": {
      "blockCode": 1001018,
      "operatorId": 34,
      "actor": {
        "_value": 1001021,
        "_ver": 1
      }
    },
    "approval": {
      "operatorId": 0,
      "status": 1,
      "approvalAt": "2022-06-09T14:24:26.600+0900",
      "expirationAt": "2022-06-16T14:22:40.038+0900"
    },
    "readAt": null,
    "sendAt": "2022-06-09T14:22:40.039+0900",
    "is_transfer": false
  }
]
```

通知 ID

c. アプリケーションが利用者 ID 連携解除申請を承認する。

サービス名称	API 名称	メソッドおよびパス
通知サービス	承認要求への承認	PUT /notification/approval ※proxy を経由するため以下のパスになります。 PUT /pxr-block-proxy/pxr-block-proxy/&block=\$pxr-root-block&path=/notification/approval

リクエスト

```
{
  id: "取得した通知 ID",
  status: 1,
  comment: "任意のコメント"
}
```

5.1.4 Regionに参加しているAPPの離脱

削除対象 Region-Root-Block に参加している APP-Block から Region 離脱申請を行い、Region-Root-Block で承認する。

※参加しているすべての Region の離脱処理を行う。

1. Region 離脱申請

a. 削除対象の APP-Block にログインする。

サービス名称	API 名称	メソッドおよびパス
オペレーターサービス	ログイン	POST /operator/login

リクエスト

```
{
  "type": 3,
  "loginId": "ログイン ID",
  "hpassword": "ハッシュパスワード"
}
```

※ハッシュパスワードは、パスワードを、「構築ガイド」の 2.4 マニフェスト作成、適用で Configmap に設定したソルト値とハッシュ化回数でハッシュ化 (SHA-256) したパスワード

b. カタログを取得し、参加している Region コードを特定する。

サービス名称	API 名称	メソッドおよびパス
カタログサービス	取得	GET /catalog/{code} ※proxy を経由するため以下のパスになります。 POST /pxr-block-proxy/pxr-block-proxy/?block=\$pxr-root-block&path=/catalog/{code}

※{code}は該当のアプリケーションのカタログコードです。

レスポンス例

```
{
  "catalogItem": {
    "ns": "catalog/ext/xxxxx/actor/app/actor_1000436/application",
    "name": "アプリケーション名",
    "_code": {
      "_value": 1000461,
      "_ver": 1
    },
    "inherit": {
      "_value": 41,
      "_ver": 1
    },
    "description": "アプリケーション概要"
  },
  "template": {
    "_code": {
      "_value": 1000461,
      "_ver": 1
    },
    "information-site": null,
    "redirect_url": null,
    "region-alliance": [
      {

```

参加している Region のコード

```

    "_value": 1001087,
    "_ver": 1
  },
  {
    "_value": 1001094,
    "_ver": 1
  }
],
"share": [
  {
    "_value": 1000463,
    "_ver": 1
  },
  {
    "_value": 1000464,
    "_ver": 1
  }
],
"store": [
  {
    "_value": 1000467,
    "_ver": 1
  }
]
},
"prop": --- 省略 ---,
"value": --- 省略 ---,
"attribute": null
}

```

c. Region 離脱申請を実施する。

サービス名称	API 名称	メソッドおよびパス
カタログ更新サービス	取得	POST /catalog-update/join/remove ※PxR-Root-Block 以外の PxR-Block で PxR-Root-Block の API を使用する場合は、proxy を経由するため以下のパスになります。 POST /pxr-block-proxy/pxr-block-proxy/?block=\$pxr-root-block&path=/catalog-update/join/remove

リクエスト

```

{
  "region": {
    "code": 参加している Region のカタログコード,
    "version": 参加している Region のカタログバージョン
  },
  "actor": {
    "code": 自組織のカタログコード,
    "version": 自組織のカタログバージョン,
    "app": [
      {
        "code": アプリケーションのカタログコード,
        "version": アプリケーションのカタログバージョン
      }
    ]
  },
  "isDraft": false
}

```

2. Region 離脱承認

a. Region-Root -Block にログインする。

サービス名称	API 名称	メソッドおよびパス
オペレーターサービス	ログイン	POST /operator/login

リクエスト

```
{
  "type": 3,
  "loginId": "ログイン ID",
  "hpassword": "ハッシュパスワード"
}
```

※ハッシュパスワードは、パスワードを、「構築ガイド」の 2.4 マニフェスト作成、適用で Configmap に設定したソルト値とハッシュ化回数でハッシュ化（SHA-256）したパスワード

b. Region 離脱申請の通知 ID を取得する。

サービス名称	API 名称	メソッドおよびパス
通知サービス	取得	GET /notification/?is_send=false&is_unread=false&is_approval=false&type=1&num=0 ※PxR-Root-Block 以外の PxR-Block で PxR-Root-Block の API を使用する場合は、proxy を経由するため以下のパスになります。 POST /pxr-block-proxy/pxr-block-proxy/?path=/notification/?is_send=false&is_unread=false&is_approval=false&type=1&num=0

レスポンス例

```
[
  {
    "id": 34,
    "type": 1,
    "title": "参加離脱",
    "content": "参加離脱を要求します",
    "attribute": {
      "region": 1001059,
      "actor": 1001021,
      "wf": [
        1001054
      ]
    },
    "category": {
      "_value": 120,
      "_ver": 1
    },
    "from": {
      "blockCode": 1001018,
      "operatorId": 34,
      "actor": {
        "_value": 1001021,
        "_ver": 1
      }
    },
    "approval": {
      "operatorId": 0,
      "status": 1,
      "approvalAt": "2022-06-09T14:24:26.600+0900",
      "expirationAt": "2022-06-16T14:22:40.038+0900"
    }
  },
]
```

```

    "readAt": null,
    "sendAt": "2022-06-09T14:22:40.039+0900",
    "is_transfer": false
  }
]

```

c. Region-Root-Block で Region 離脱申請を承認する。

サービス名称	API 名称	メソッドおよびパス
通知サービス	承認要求への承認	PUT /notification/approval ※PxR-Root-Block 以外の PxR-Block で PxR-Root-Block の API を使用する場合は、proxy を経由するため以下のパスになります。 PUT /pxr-block-proxy/pxr-block-proxy/?block=\$pxr-root-block&path=/notification/approval

リクエスト

```

{
  id: "取得した通知 ID",
  status: 1,
  comment: "任意のコメント"
}

```

5.1.5 流通制御による利用者 ID 連携情報の確認

PxR-Root-Block から個人の情報を取得して、解除されていない利用者 ID 連携がないか確認を行う。
必要であれば流通制御から利用者 ID 連携解除を行う。

1. 利用者 ID 連携情報の確認
 - a. PXR-Root-Block にログインする。

サービス名称	API 名称	メソッドおよびパス
オペレーターサービス	ログイン	POST /operator/login

リクエスト

```

{
  "type": 3,
  "loginId": "ログイン ID",
  "hpassword": "ハッシュパスワード"
}

```

※ハッシュパスワードは、パスワードを、「構築ガイド」の 2.4 マニフェスト作成、適用で Configmap に設定したソルト値とハッシュ化回数でハッシュ化 (SHA-256) したパスワード

- b. 利用者 ID 連携情報の確認

サービス名称	API 名称	メソッドおよびパス
Book 管理サービス	My-Condition-Book 一覧取得	POST /book-manage/search ※proxy を経由するため以下のパスになります。

		POST /pxr-block-proxy/pxr-block-proxy/?path= /book-manage/search
--	--	--

リクエスト

```
{
  "pxrId": null,
  "createdAt": null,
  "offset": 何件目の Book から取得を開始するか (数字) ,
  "limit": 開始から何件 Book を取得するか (数字)
}
```

レスポンス例

```
[
  {
    "pxrId": "pxr_user_test001",
    "status": 0,
    "attributes": {
      "key": "value"
    },
    "cooperation": [
      {
        "actor": {
          "_value": 1000439,
          "_ver": 1
        },
        "region": null,
        "app": null,
        "wf": {
          "_value": 1000501,
          "_ver": 1
        },
        "userId": "aaaaaaa",
        "startAt": null,
        "status": 0
      }
    ],
    "userInformation": --- 省略 ---
  }
]
```

利用者 ID 連携情報

※利用者 ID 連携が残っていれば「5.1.2 Region 利用者 ID 連携の解除」または「5.1.3 APP 利用者 ID 連携の解除」から利用者 ID 連携を解除してください。

5.1.6 アクター認定解除

削除対象の Block からアクター認定解除申請を行い、PxR-Root-Block で承認する。

1. アクター認定解除申請

a. 削除対象の Block にログインする。

サービス名称	API 名称	メソッドおよびパス
オペレーターサービス	ログイン	POST /operator/login

リクエスト

```
{
  "type": 3,
  "loginId": "ログイン ID",
  "hpassword": "ハッシュパスワード"
}
```

※ハッシュパスワードは、パスワードを、「構築ガイド」の 2.4 マニフェスト作成、適用で Configmap に設定したソルト値とハッシュ化回数でハッシュ化 (SHA-256) したパスワード

b. アクター認定解除申請を実施する。

サービス名称	API 名称	メソッドおよびパス
カタログ更新サービス	取得	POST /catalog-update/actor/remove ※proxy を経由するため以下のパスになります。 POST /pxr-block-proxy/pxr-block-proxy/?block=\$pxr-root-block&path=/catalog-update/actor/remove

リクエスト

```
{
  "migrationActorCode": null,
  "isDraft": false
}
```

2. アクター認定解除申請の承認

a. PxR-Root-Block にログインする。

サービス名称	API 名称	メソッドおよびパス
オペレーターサービス	ログイン	POST /operator/login

リクエスト

```
{
  "type": 3,
  "loginId": "ログイン ID",
  "hpassword": "ハッシュパスワード"
}
```

※ハッシュパスワードは、パスワードを、「構築ガイド」の 2.4 マニフェスト作成、適用で Configmap に設定したソルト値とハッシュ化回数でハッシュ化 (SHA-256) したパスワード

b. アクター認定解除申請の通知 ID を取得する。

サービス名称	API 名称	メソッドおよびパス
通知サービス	取得	GET /notification/?is_send=false&is_unread=false&is_approval=false&type=1&num=0 ※proxy を経由するため以下のパスになります。 /pxr-block-proxy/pxr-block-proxy/?path=/notification/?is_send=false&is_unread=false&is_approval=false&type=1&num=0

レスポンス例

<pre>[{ "id": 34, "type": 1, "title": "アクター認定解除, "content": "アクター認定解除を要求します", "attribute": { "region": 1001059, "actor": 1001021, "wf": [1001054] }, "category": { "_value": 120, "_ver": 1 }, "from": { "blockCode": 1001018, "operatorId": 34, "actor": { "_value": 1001021, "_ver": 1 } }, "approval": { "operatorId": 0, "status": 1, "approvalAt": "2022-06-09T14:24:26.600+0900", "expirationAt": "2022-06-16T14:22:40.038+0900" }, "readAt": null, "sendAt": "2022-06-09T14:22:40.039+0900", "is_transfer": false }]</pre>	<div style="background-color: #003366; color: white; padding: 5px; display: inline-block;">通知 ID</div>
---	--

c. アクター認定解除申請を承認する。

サービス名称	API 名称	メソッドおよびパス
通知サービス	承認要求への承認	PUT /notification/approval

リクエスト

<pre>{ id: "取得した通知 ID", status: 1, comment: "任意のコメント" }</pre>

5.2 PF 設定 : Pod 停止、eks マニフェスト削除

削除対象 PxR-Block の Pod の停止、および eks マニフェストの削除を行う。

※PxR-Root-Block を削除する場合は、すべての PxR-Block を対象に実施する。

1. Deployment 削除

a. Kubernetes の deployment 一覧を取得する。

```
# kubectl get deploy -n pxr
```

b. レスポンスは、下記のように表示される。

```
NAME                READY  UP-TO-DATE  AVAILABLE  AGE
application000001-api  1/1    1            1           75d
region000001-api     1/1    1            1           75d
root-api             1/1    1            1           75d
```

c. 削除対象の PxR-Block を Kubernetes の Deployment 削除を実行する。

```
# kubectl delete deploy -n pxr <deployment NAME>
```

※<deployment NAME>は、一覧から指定する。

2. Service 削除

a. Service 一覧を取得する。

```
# kubectl get svc -n pxr
```

b. レスポンスは、下記のように表示される。

```
NAME                TYPE        CLUSTER-IP  EXTERNAL-IP  PORT(S)        AGE
application000001-service ClusterIP   172.20.80.243 <none>       80/TCP,443/TCP 103d
application000002-service ClusterIP   172.20.64.91  <none>       80/TCP,443/TCP 103d
application000003-service ClusterIP   172.20.174.34 <none>       80/TCP,443/TCP 103d
```

c. 削除対象の PxR-Block を Kubernetes の Service 削除コマンドを実行する。

```
# kubectl delete svc -n pxr <service NAME>
```

※<service NAME>は、Service 一覧から取得した Service を指定する。

3. Ingress 削除

a. pxr-ingress.yaml を編集し、削除対象 Block の host 定義を削除する。

例) 下記二重取り消し線部分 [**application000002**]

```

---
apiVersion: extensions/v1beta1
kind: Ingress
~省略~
- host: application000001.pxrtest4.pxrtest.me.uk
  http:
    paths:
      - path: /*
        backend:
          serviceName: application000001-service
          servicePort: 80
host: application000002.pxrtest4.pxrtest.me.uk
http:
paths:
path: /*
backend:
serviceName: application000002-service
servicePort: 80

```

b. Kubernetes に Ingress を適用する。

```
# kubectl apply -f pxr-ingress.yaml
```

4. 削除対象 Block の yaml ファイルを削除する。

削除対象ファイル

- manifest/deployment/<削除 PxR-Block>-deployment.yaml
- manifest/services/<削除 PxR-Block>-service.yaml
- manifest/configmap/<削除 PxR-Block>/ ※ディレクトリ、ファイル削除

例) <削除 PxR-Block> : 「**application000002**」

```

# cd ./manifest
# rm -rvf .//*application000002*.yaml
# rm -rvf ./configmap/application000002

```

5. Yaml ファイルの適用を行う。

```
# kubectl apply -f . -R
```

6. Kubernetes の Pod ステータス一覧から削除した PxR-Block の Pod が表示されていないならば削除完了となる。

※Pod ステータス一覧レスポンス例

```

# kubectl get Pod -n pxr
application000001-api-df4dbb5f5-pgxnw 10/10 Running 0 13h
region000001-api-6884664976-zzbgg 7/7 Running 0 13h
root-api-86df7cffb9-btrsc 17/17 Running 0 13h

```

5.3 PF 設定 : DB 削除

削除対象 Block の DB を削除する。※対象 Block のデータを消去する場合に実施する。

※PxR-Root-Block を削除する場合は、全 DB を消去する。

本作業は、対象の DB を削除するにはコマンドを実行するロールがスーパーユーザーか、データベースの所有者権限で実行する。

1. 対象 Block のデータベースを削除する。

コマンド : DB の削除

```
DROP DATABASE <削除対象のデータベース名>;
```

2. 対象 Block の DB ユーザーを消去する。※Block 追加時に作成した全 DB ユーザーを削除

※原則、データベースの削除を実行すると、紐づけられたユーザーも一緒に削除される。DB ユーザーを確認して、必要の無い DB ユーザーが残っていれば、本対応を実施する。

コマンド : DB ユーザーの確認

```
select username from pg_user;
```

コマンド : DB ユーザーの削除

```
DROP USER <削除対象プレフィックス>_access_control_user;  
DROP USER <削除対象プレフィックス>_access_manage_user;  
DROP USER <削除対象プレフィックス>_book_manage_user;  
DROP USER <削除対象プレフィックス>_binary_manage_user;  
DROP USER <削除対象プレフィックス>_book_operate_user;  
DROP USER <削除対象プレフィックス>_catalog_user;  
DROP USER <削除対象プレフィックス>_catalog_update_user;  
DROP USER <削除対象プレフィックス>_certification_authority_user;  
DROP USER <削除対象プレフィックス>_certificate_manage_user;  
DROP USER <削除対象プレフィックス>_identify_verify_user;  
DROP USER <削除対象プレフィックス>_notification_user;  
DROP USER <削除対象プレフィックス>_operator_user;  
DROP USER <削除対象プレフィックス>_block_proxy_user;  
DROP USER <削除対象プレフィックス>_ctoken_ledger_user;  
DROP USER <削除対象プレフィックス>_local_ctoken_user;
```