CHAPTER 4

API データ型

この章では、SCMS SCE Subscriber API で使用されるさまざまな API データ型について説明します。

サブスクライバ ID

SCE Subscriber API のほとんどのメソッドでは、入力パラメータとしてサブスクライバ ID を使用する必要があります。サブスクライバ ID は、サブスクライバ名または CM MAC（メディア アクセス制御）アドレスを表すストリングです。ここでは、サブスクライバ ID のフォーマット ルールについて説明します。

サブスクライバ名では、*大文字と小文字を区別*します。使用できる文字数は最大 64 文字です。34（"）、39（'）、96（`）を除く 32 ～ 126（両端を含む）の ASCII コードで出力可能な文字はすべて使用できます。

次に例を示します。

```
String subID1="john";
String subID2="john@yahoo.com";
```

ネットワーク ID のマッピング

ネットワーク ID は、SCE デバイスが特定のサブスクライバ レコードと関連付けるネットワーク識別子です。たとえば IP アドレスは、ネットワーク ID マッピングの典型的な例です。現在のところ、IP アドレス、IP 範囲、VLAN のマッピング タイプが SCE でサポートされています。

NetworkID クラスは、さまざまなタイプのサブスクライバ ネットワーク ID を表します。

API は次のサブスクライバ マッピング タイプをサポートします。

- IP アドレスまたは IP 範囲
- VLAN タグ

（注）　同じサブスクライバに IP アドレス /IP 範囲と VLAN タグを同時に使用することはできません。

ネットワーク ID を含むサブスクライバ操作を使用している場合、発信側は NetworkID パラメータを指定する必要があります。

NetworkID クラス コンストラクタの定義は、次のとおりです。

```
public NetworkID(String mapping,short mappingType) throws Exception
public NetworkID(String[] mappings,short[] mappingTypes) throws Exception
```

NetworkID コンストラクタのパラメータは、次のとおりです。

- **java.lang.String** マッピング ID、またはマッピング ID の配列
- short マッピング タイプ、またはマッピング タイプの配列

配列を渡す場合は、MappingType 配列に、マッピング配列と同数の要素、または単一の要素が含まれている必要があります。

- 配列に複数の要素が含まれている場合は、NetworkID.TYPE_IP または NetworkID.TYPE_VLAN 定数を使用します。
- 単一配列要素の場合は、NetworkID.ALL_IP_MAPPINGS または NetworkID.ALL_VLAN_MAPPINGS 定数を使用します。

IP アドレス マッピングの指定

IP アドレスの文字列フォーマットには、一般的に次のような 10 進表記法が使用されています。

```
IP-Address=[0-255].[0-255].[0-255].[0-255]
```

例

- `216.109.118.66`

  IP アドレスのマッピングの型は、クラス NetworkID で指定されます。
- `com.scms.common.NetworkID.TYPE_IP`

  com.scms.common.NetworkID.ALL_IP_MAPPINGS は、マッピング ID 配列内のすべてエントリが IP マッピングであることを指定します。

IP 範囲マッピングの指定

IP 範囲の文字列フォーマットは、10 進表記法の IP アドレスおよびビット マスク内の **1** の数を表す 10 進数です。**IP-Range=[0-255].[0-255].[0-255].[0-255]/[0-32]**.

例

- **10.1.1.10/32** はフル マスクの IP 範囲、つまり正規の IP アドレスです。
- **10.1.1.0/24** は 24 ビット マスクの IP 範囲で、**10.1.1.0** ～ **10.1.1.255** の範囲のアドレスすべてを表します。

（注） IP 範囲のマッピングの型は、IP アドレスのマッピングの型と同じです。

VLAN タグ マッピングの指定

VLAN タグ マッピングの文字列フォーマットは、次の範囲の 10 進数です。**[2-2046]**

VLAN マッピングの型は、**com.scms.common.NetworkID** クラスで指定されます。

- IP アドレスのマッピングの型は、クラス NetworkID で指定されます。
- `com.scms.common.NetworkID.TYPE_VLAN`
- **com.scms.common.NetworkID.ALL_VLAN_MAPPINGS** は、マッピング ID 配列内のすべてエントリが VLAN マッピングであることを指定します。

ネットワーク ID マッピングの例

単一の IP アドレスを持つ NetworkID を作成します。

```
NetworkID nid = new NetworkID("1.1.1.1",NetworkID.TYPE_IP)
```

IP アドレス範囲を持つ NetworkID を作成します。

```
NetworkID nid = new NetworkID("1.1.1.1/24",NetworkID.TYPE_IP)
```

複数の IP アドレスを持つ NetworkID を作成します。

```
NetworkID nid = new NetworkID(new String[]{"1.1.1.1","2.2.2.2","3.3.3.3"},
NetworkID.ALL_IP_MAPPINGS)
```

単一の VLAN アドレスを持つ NetworkID を作成します。

```
NetworkID nid = new NetworkID("23",NetworkID.TYPE_VLAN)
```

SCA BB サブスクライバのポリシー プロファイル

ポリシー プロファイルには、サブスクライバ ポリシー情報が記述されています。ポリシー プロファイルは一般に、ポリシー パッケージで識別される静的に定義されたポリシーと、動的な性質を持つことがある一連のサブスクライバ ポリシー プロパティの、2 つの主要パートで構成されます。パッケージ ID はポリシー パッケージを識別します。サブスクライバ トラフィックに適用されるルールの大部分は、パッケージ ID から取得されます。

SCA BB 内のサブスクライバ ポリシー プロパティは、サブスクライバによって生成されたネットワーク トラフィックに対する SCE の分析や反応に影響する一組の**キー値**です。

プロパティの詳細については、『*Cisco Service Control Application Suite for Broadband User Guide*』を参照してください。

SCA BB バージョン 3.0 には次のプロパティがあります。

- packageId — サブスクライバのパッケージ ID を定義します。
- monitor — このサブスクライバのトランザクションごとに Raw Data Record（RDR）を発行するかどうかを指定します。

PolicyProfile クラス

API には、API 操作に必要なサブスクライバ ポリシー プロファイルをフォーマット化するための PolicyProfile クラスがあります。

次のメソッドは、ポリシー プロパティの配列に基づいて PolicyProfile クラスを構築します。

```
public PolicyProfile(String[] policy)
```

（注） 配列内の各ストリングは、次のように符号化する**必要があります**。

```
property_name=property_value
```

次のメソッドを使用すると、上記フォーマットに従って、プロファイルにポリシー プロファイルを追加できます。

```
public void addPolicyProperty(String policyProperty)
```

（注） このメソッドはパフォーマンスを高めるために最適化されていません。パフォーマンスを最大にするには、PolicyProfile コンストラクタを使用します。

例

```
PolicyProfile pp = new PolicyProfile(new
         String[]{"packageId=22", "monitor=1"})
```

サブスクライバ クォータ

SCA BB でクォータ プロビジョニングを実行するには、サブスクライバ クォータ バケットを使用します。サブスクライバごとに 16 のバケットがあり、各バケットを容量またはセッション数で定義できます。サブスクライバが特定のサービスを利用すると、消費した容量またはセッション数が、いずれかのバケットから減らされます。各サービスで使用するバケットは、SCA BB コンソール を使用して汎用ポリシー定義内で定義されたサービス設定によって決まります。ボリューム バケットの消費量は L3 キロ バイト単位でカウントされ、セッション バケットの消費量はセッション数でカウントされます。たとえば、閲覧と電子メール サービスはバケット番号 1 から、P2P サービスはバケット番号 2 から、それぞれクォータを消費するものとし、それ以外のサービスはどれも特定のバケットに対応付けないように定義することができます。

クォータ バケットは次のコンポーネントで構成されます。

- バケット ID — 定義済みポリシー内の定義に従うバケットの一意の ID（**String**）。有効値の範囲は [1-16] です。
- バケット値 — クォータ バケット値（**long**）

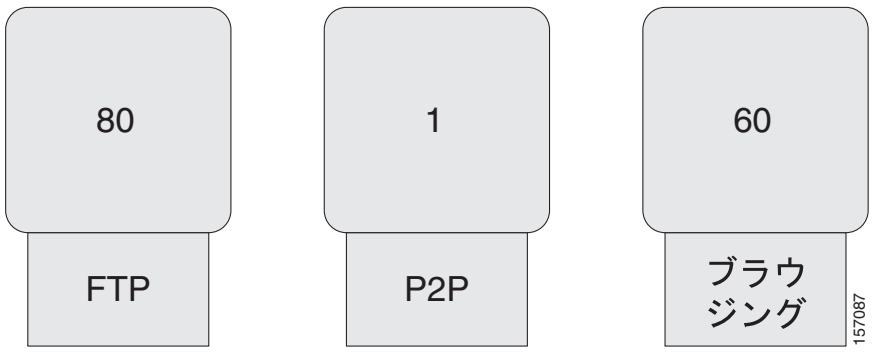
クォータ操作によって、サブスクライバのクォータ バケットは動的に変更されます。クォータ操作には 2 つのタイプがあります。

- ADD_QUOTA_OPERATION — SCE プラットフォーム上のバケットの現在値に、新しいクォータ値を加算します。
- SET_QUOTA_OPERATION — SCE プラットフォーム上のクォータ バケット値を新しい値で交換します。

例

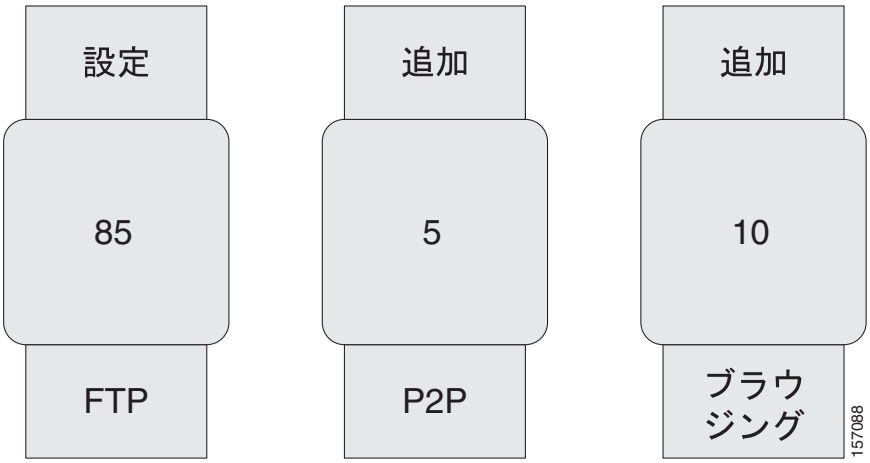
SCE のサブスクライバ A のクォータの現在値は、次のとおりです。

図 4-1　サブスクライバ クォータ — 現在値



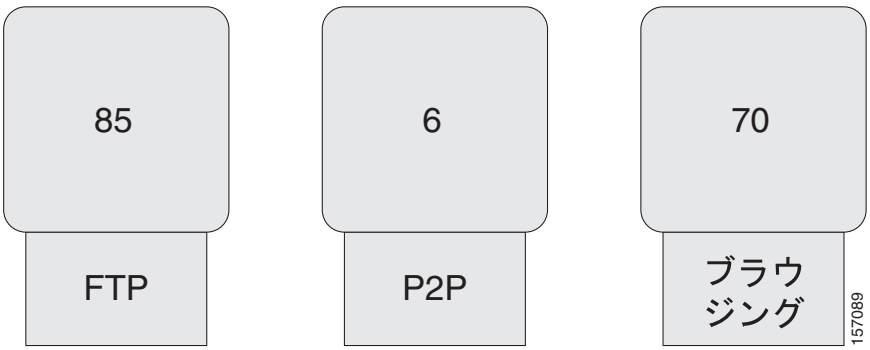
既存クォータに次のアクションを適用するとします。

図 4-2　サブスクライバ クォータ — アクションの適用



クォータ アクションを実行すると、次のようになります。

図 4-3　サブスクライバ クォータ — 結果



サブスクライバ クォータの詳細については、『*Cisco Service Control Application for Broadband User Guide*』を参照してください。

ここでは、サブスクライバ クォータ管理操作などのために API が提供するクラスについて説明します。

SCAS_BB_Quota

SCAS_BB_Quota クラスは、すべてのコールバック関数で QuotaListenerEx インターフェイスが使用するクォータ インターフェイスを実装します。「QuotaListenerEx インターフェイス クラス」(p.5-10) を参照してください。

次のメソッドは、ID および値の配列に基づいて SCAS_BB_Quota を構築します。

```
public SCAS_BB_Quota (String[] bucketIDs,
long[] bucketValues)
```

次のメソッドは、ID および値の配列、プロファイル ID、理由、およびタイムスタンプに基づいて SCAS_BB_Quota を構築します。

```
public SCAS_BB_Quota (String[] bucketIDs,
long[] bucketValues,
int quotaProfileId,
int reason,
long timestamp)
```

次のメソッドでは、クォータ バケットの ID を取得できます。

```
public String[] getBucketIDs()
```

次のメソッドでは、クォータ バケットの値を取得できます。

```
public long[] getBucketValues()
```

quotaProfileId パラメータはクォータ プロファイルの ID、つまり パッケージ ID です。次のメソッドでは、クォータ プロファイル ID を取得できます。

```
public int getQuotaProfileId()
```

reason パラメータはクォータ ステータス イベントにのみ関連し、3 つの有効値を取ります。

- 0 — 設定時間（2 分おきなど）に達しました。
- 1 — サブスクライバ ログアウトによってクォータ ステータス イベントがトリガーされました。
- 2 — パッケージ変更によってクォータ ステータス イベントがトリガーされました。

次のメソッドでは、理由を取得できます。

```
public int getReason()
```

timestamp パラメータには、イベントが生成された（SCE 内の）時刻が含まれます。このパラメータは、1970 年 1 月 1 日 00:00 GMT（グリニッジ標準時）からの秒数で計算されます。

次のメソッドでは、タイムスタンプを取得できます。

```
public long getTimestamp()
```

SCAS_BB_QuotaOperation

SCAS_BB_QuotaOperation クラスは QuotaOperation インターフェイスを実装します。このインターフェイスは、ログイン操作（「ログオン操作」[p.5-20] を参照）やクォータ更新操作（「quotaUpdate 操作」[p.5-29] を参照）など、サブスクライバのクォータを含むサブスクライバ プロビジョニング操作に使用されます。

次のメソッドは、ID、値、およびアクションの配列に基づいて SCAS_BB_QuotaOperation を構築します。

```
public SCAS_BB_QuotaOperation (String[] IDs,
long[] values,
short[] actions)
```

次のメソッドでは、クォータ バケットの ID を取得できます。

```
public String[] getBucketIDs()
```

次のメソッドでは、クォータ バケットの値を取得できます。

```
public long[] getBucketValues()
```

次のメソッドでは、クォータ バケットのアクションを取得できます。

```
public short[] getBucketActions()
```

バルク操作のデータ型

それぞれ独自のパラメータを持つ複数のサブスクライバに同じメソッドを実行する場合は、バルク クラスおよびバルク操作を使用します。API にはバルク クラスがあり、バルク操作の結果処理や、SCE からの一括通知に使用できます。バルク クラスは **loginBulk** や **logoutBulk** などのバルク メソッドに渡されます。

次に、バルク操作を使用する場合の注意事項を示します。

- すべてのバルク クラスは共通の BulkBase クラスから継承されます。
- SCE のメモリ制限により、バルク サイズのエントリ数は最大 100 に制限されています。

バルク イテレータ

BulkBase クラスには、バルクに含まれるデータを表示するためのイテレータがあります。

次に バルク イテレータの構文を示します。

```
Iterator getIterator()
```

このイテレータを使用すると、さまざまな通知内で SCE から受信されたバルクに対して反復操作を実行したり（logoutBulkIndication、loginPullBulkResponseIndication など）、操作に失敗した場合に、各操作で使用されたデータを検査したりできます。

イテレータにはデータを取得するための次のメソッドがあります。

```
public Object next()
public boolean hasNext()
```

next() メソッドは SubscriberData オブジェクトを返します。

SubscriberData クラスは、バルクに含まれる単一サブスクライバの情報を取得する場合に使用します。

SubscriberData

SubscriberData クラスは、特定のサブスクライバに実行できるすべての操作を表します。SubscriberData クラスには情報を取得するための次のユーティリティ メソッドが含まれています。

```
public String getSubscriberID()
public String getAnonymousID()
public String[] getMappings()
public short[] getTypes()
public boolean getAdditiveFlag()
```

ここでは、各 API 操作で使用できるさまざまなバルク データの型について説明します。

Login_BULK クラス

このクラスはサブスクライバのバルクを表し、**loginBulk** 操作に必要なデータをすべて含みます。

コンストラクタ

データで満たされた **Login_BULK** を構築するには、次のコンストラクタを使用します。

```
public Login_BULK(String[] subscriberIDs,
NetworkID[] networkIDs,
boolean[]networkIDsAdditive,
PolicyProfile[] policy,
QuotaOperation[] quota)
```

パラメータ

subscriberID — サブスクライバの一意の ID。サブスクライバ ID のフォーマットについては、「サブスクライバ ID」（p.4-2）を参照してください。

networkID — サブスクライバのネットワーク ID。詳細は「ネットワーク ID のマッピング」（p.4-2）を参照してください。

networkIDAdditive — このフラグが TRUE に設定されている場合は、サブスクライバの既存の networkID に、指定した networkID が追加されます。それ以外の場合は、指定した networkID で既存の networkID が置き換えられます。

policy — サブスクライバのポリシー プロファイル。詳細は「SCA BB サブスクライバのポリシー プロファイル」（p.4-4）を参照してください。

quota — サブスクライバのクォータ。詳細は「サブスクライバ クォータ」（p.4-5）を参照してください。

空の Login_BULK を構築するには、次のメソッドを使用します。

```
public Login_BULK()
```

addBulkEntry メソッド

バルクにエントリを追加するには、次のメソッドを使用します。

```
public void addBulkEntry(String subscriberID,
NetworkID networkID,
boolean networkIdsAdditive,
PolicyProfile policy,
QuotaOperation quota)
```

パラメータ

subscriberID — サブスクライバの一意の ID。サブスクライバ ID のフォーマットについては、「サブスクライバ ID」（p.4-2）を参照してください。

networkID — サブスクライバのネットワーク ID。詳細は「ネットワーク ID のマッピング」（p.4-2）を参照してください。

networkIDAdditive — このフラグが TRUE に設定されている場合は、サブスクライバの既存の networkID に、指定した networkID が追加されます。それ以外の場合は、指定した networkID で既存の networkID が置き換えられます。

policy — サブスクライバのポリシー プロファイル。詳細は「SCA BB サブスクライバのポリシー プロファイル」（p.4-4）を参照してください。

quota — サブスクライバのクォータ。詳細は「サブスクライバ クォータ」（p.4-5）を参照してください。

例

Login_BULK オブジェクトの使用方法：例

次に、Login_BULK オブジェクトの使用例を示します。

```
// バルク構築用のデータをすべて用意します。
String[] names = new String[5];
NetworkID[] mappings = new NetworkID[5];
boolean[] additive = new boolean[5];
PolicyProfile[] policy = new PolicyProfile[5];

for (int i=0; i<5; i++)
{
names[i]="sub_"+i;
mappings[i] = new NetworkID(”1.1.1.”+i,NetworkID.TYPE_IP);
additive[i] = true;
policy[i] = new PolicyProfile(new String[]{"packageId="+(i+1)});
}
// バルク オブジェクトを構築します。
Login_BULK bulk = new Login_BULK(names,mappings,additive,policy,null);
// これで loginBulk 操作でバルク オブジェクトを使用できます。
sceApi.loginBulk(bulk,null);
```

Login_BULK の操作：例

次の例は、Login_BULK オブジェクトの別の操作方法を示しています。

```
// 空のバルクを構築します。
Login_BULK bulk = new Login_BULK ();
// addBulkEntry メソッドを使用してバルクにデータを設定します。
for (int i=0; i<20; i++)
{
String name ="sub_"+i;
NetworkID mappings = new NetworkID(i+1);
boolean additive = true;
PolicyProfile policy = new PolicyProfile(
new String[]{"packageId="+(i+1)});
QuotaOperation quota = new SCAS_BB_QuotaOperation(
new String[]{“1”,”2”,”3”},
new long[]{80,80,0}
new short[]{SCAS_BB_QuotaOperation.ADD_QUOTA_OPERATION,
SCAS_BB_QuotaOperation.ADD_QUOTA_OPERATION,
SCAS_BB_QuotaOperation.SET_QUOTA_OPERATION});
bulk.addBulkEntry(name,mappings,additive,policy,quota);
}
// これで loginBulk 操作でバルク オブジェクトを使用できます。
sceApi.loginBulk(bulk,null);
```

SubscriberID_BULK クラス

入力するサブスクライバ ID のみを必要とする **logoutBulkIndication** コールバック関数では、SubscriberID_BULK クラスを使用します。「logoutBulkIndication コールバック メソッド」（p.5-10）を参照してください。

コンストラクタ

サブスクライバ ID データを含む **SubscriberID_BULK** を構築するには、次のコンストラクタを使用します。

```
public SubscriberID_BULK(String[] subscriberIDs)
```

空の **SubscriberID_BULK** を構築するには、次のメソッドを使用します。

```
public SubscriberID_BULK()
```

パラメータ

subscriberID — サブスクライバの一意の ID。サブスクライバ ID のフォーマットについては、「サブスクライバ ID」（p.4-2）を参照してください。

addBulkEntry メソッド

SubscriberID バルクにエントリを追加するには、次のメソッドを使用します。

```
addBulkEntry(String subscriberID)
```

パラメータ

subscriberID — サブスクライバの一意の ID。サブスクライバ ID のフォーマットについては、「サブスクライバ ID」（p.4-2）を参照してください。

NetworkAndSubscriberID_BULK クラス

次の操作では、サブスクライバ ID および NetworkID を必要とするバルク操作に NetworkAndSubscriberID_BULK クラスを使用します。

- **getSubscribersBulkResponse** コールバック（「LoginPullListener インターフェイス クラス」[p.5-8] を参照）
- **logoutBulk** 操作（「logoutBulk 操作」[p.5-25] を参照）
- **networkIDUpdateBulk** 操作（「networkIdUpdateBulk 操作」[p.5-27] を参照）

コンストラクタ

SubscriberID および NetworkID データを含む **NetworkAndSubscriberID_BULK** を構築するには、次のコンストラクタを使用します。

```
public NetworkAndSubscriberID_BULK(String[] subscriberIDs,
NetworkID[] networkIDs,
boolean[] netIdAdditive)
```

空の **NetworkAndSubscriberID_BULK** を構築するには、次のメソッドを使用します。

```
public NetworkAndSubscriberID_BULK()
```

パラメータ

subscriberID — サブスクライバの一意の ID。サブスクライバ ID のフォーマットについては、「サブスクライバ ID」（p.4-2）を参照してください。

networkID — サブスクライバのネットワーク ID。詳細は「ネットワーク ID のマッピング」（p.4-2）を参照してください。

networkIDAdditive — このフラグが TRUE に設定されている場合は、サブスクライバの既存の networkID に、指定した networkID が追加されます。それ以外の場合は、指定した networkID で既存の networkID が置き換えられます。

addBulkEntry メソッド

バルクにエントリを追加するには、次のメソッドを使用します。

```
addBulkEntry(String subscriberID,
NetworkID networkID,
boolean netIdAdditive)
```

パラメータ

subscriberID — サブスクライバの一意の ID。サブスクライバ ID のフォーマットについては、「サブスクライバ ID」（p.4-2）を参照してください。

networkID — サブスクライバのネットワーク ID。詳細は「ネットワーク ID のマッピング」（p.4-2）を参照してください。

networkIDAdditive — このフラグが TRUE に設定されている場合は、サブスクライバの既存の networkID に、指定した networkID が追加されます。それ以外の場合は、指定した networkID で既存の networkID が置き換えられます。

LoginPullResponse_BULK クラス

このクラスはサブスクライバのバルクを表し、**loginPullResponseBulk** メソッドに必要なデータをすべて含みます。

コンストラクタ

関連データを含む **LoginPullResponse_BULK** を構築するには、次のコンストラクタを使用します。

```
public LoginPullResponse_BULK(String[] anonymousSubscriberIDs,
String[] subscriberIDs,
NetworkID[] networkIDs,
boolean[] networkIdsAdditive,
PolicyProfile[] policy,
QuotaOperation[] quota)
```

空の **LoginPullResponse_BULK** を構築するには、次のメソッドを使用します。

```
public LoginPullResponse_BULK()
```

- パラメータ（p.4-13）

パラメータ

anonymousSubscriberID — アノニマス サブスクライバの ID。この ID は loginPullRequest/loginPullBulkRequest 通知に格納されて、SCE から送信されます（「loginPullRequest コールバック メソッド」[p.5-8] および「loginPullRequestBulk コールバック メソッド」[p.5-9] を参照）。詳細は「サブスクライバ統合モデル」（p.2-3）を参照してください。

subscriberID — サブスクライバの一意の ID。サブスクライバ ID のフォーマットについては、「サブスクライバ ID」（p.4-2）を参照してください。

networkID — サブスクライバのネットワーク ID。詳細は「ネットワーク ID のマッピング」（p.4-2）を参照してください。

networkIDAdditive — このフラグが TRUE に設定されている場合は、サブスクライバの既存の networkID に、指定した networkID が追加されます。それ以外の場合は、指定した networkID で既存の networkID が置き換えられます。

policy — サブスクライバのポリシー プロファイル。詳細は「SCA BB サブスクライバのポリシー プロファイル」（p.4-4）を参照してください。

quota — サブスクライバのクォータ。詳細は「サブスクライバ クォータ」（p.4-5）を参照してください。

addBulkEntry メソッド

バルクにエントリを追加するには、次のメソッドを使用します。

```
public addBulkEntry(String anonymousSubscriberID,
String subscriberID,
NetworkID networkID,
boolean networkIdAdditive,
PolicyProfile policy,
QuotaOperation quota)
```

パラメータ

anonymousSubscriberID — アノニマス サブスクライバの ID。この ID は loginPullRequest/loginPullBulkRequest 通知に格納されて、SCE から送信されます（「loginPullRequest コールバック メソッド」[p.5-8] および「loginPullRequestBulk コールバック メソッド」[p.5-9] を参照）。詳細は「サブスクライバ統合モデル」（p.2-3）を参照してください。

subscriberID — サブスクライバの一意の ID。サブスクライバ ID のフォーマットについては、「サブスクライバ ID」（p.4-2）を参照してください。

networkID — サブスクライバのネットワーク ID。詳細は「ネットワーク ID のマッピング」（p.4-2）を参照してください。

networkIDAdditive — このフラグが TRUE に設定されている場合は、サブスクライバの既存の networkID に、指定した networkID が追加されます。それ以外の場合は、指定した networkID で既存の networkID が置き換えられます。

policy — サブスクライバのポリシー プロファイル。詳細は「SCA BB サブスクライバのポリシー プロファイル」（p.4-4）を参照してください。

quota — サブスクライバのクォータ。詳細は「サブスクライバ クォータ」（p.4-5）を参照してください。

PolicyProfile_BULK クラス

updatePolicyProfileBulk 操作ではこのクラスを使用して、サブスクライバ ID およびサブスクライバ ポリシー プロファイルのバルクを表します。

- コンストラクタ（p.4-14）
- addBulkEntry メソッド（p.4-14）

コンストラクタ

関連データを含む **PolicyProfile_BULK** を構築するには、次のコンストラクタを使用します。

```
public PolicyProfile_BULK(String[] subscriberIDs, PolicyProfile[] policy)
```

空の **PolicyProfile_BULK** を構築するには、次のメソッドを使用します。

```
public PolicyProfile_BULK()
```

パラメータ

subscriberID — サブスクライバの一意の ID。サブスクライバ ID のフォーマットについては、「サブスクライバ ID」（p.4-2）を参照してください。

policy — サブスクライバのポリシー プロファイル。詳細は「SCA BB サブスクライバのポリシー プロファイル」（p.4-4）を参照してください。

addBulkEntry メソッド

バルクにエントリを追加するには、次のメソッドを使用します。

```
public addBulkEntry(String subscriberID,  PolicyProfile policy)
```

パラメータ

subscriberID — サブスクライバの一意の ID。サブスクライバ ID のフォーマットについては、「サブスクライバ ID」（p.4-2）を参照してください。

policy — サブスクライバのポリシー プロファイル。詳細は「SCA BB サブスクライバのポリシー プロファイル」（p.4-4）を参照してください。

Quota_BULK クラス

次の操作ではこのクラスを使用して、サブスクライバ ID およびクォータ バケットのバルクを表します。

- getQuotaStatusBulk 操作（バケット ID のみを指定）
- quotaStatusBulkIndication コールバック メソッド
- quotaDepletedBulkIndication コールバック メソッド
- quotaBelowThresholdIndication コールバック メソッド

コンストラクタ

関連データを含む **Quota_BULK** を構築するには、次のコンストラクタを使用します。

```
public Quota_BULK(String[] subscriberIDs, Quota[] subscribersQuota)
```

空の Quota_BULK を構築するには、次のメソッドを使用します。

```
public Quota_BULK()
```

パラメータ

subscriberID — サブスクライバの一意の ID。サブスクライバ ID のフォーマットについては、「サブスクライバ ID」（p.4-2）を参照してください。

quota — サブスクライバのクォータ。詳細は「サブスクライバ クォータ」（p.4-5）を参照してください。

addBulkEntry メソッド

バルクにエントリを追加するには、次のメソッドを使用します。

```
public addBulkEntry(String subscriberID,Quota quota)
```

パラメータ

subscriberID — サブスクライバの一意の ID。サブスクライバ ID のフォーマットについては、「サブスクライバ ID」（p.4-2）を参照してください。

quota — サブスクライバのクォータ。詳細は「サブスクライバ クォータ」（p.4-5）を参照してください。

QuotaOperation_BULK クラス

QuotaUpdateBulk 操作およびログイン操作ではこのクラスを使用して、サブスクライバ ID およびサブスクライバ クォータ操作のバルクを表します。

コンストラクタ

関連データを含む **QuotaOperation_BULK** を構築するには、次のコンストラクタを使用します。

```
public QuotaOperation_BULK(String[] subscriberIDs,
QuotaOperation[]quotaOperations)
```

空の QuotaOperation_BULK を構築するには、次のメソッドを使用します。

```
public QuotaOperation_BULK()
```

パラメータ

subscriberID — サブスクライバの一意の ID。サブスクライバ ID のフォーマットについては、「サブスクライバ ID」（p.4-2）を参照してください。

quotaOperation — サブスクライバのクォータに対して実行するクォータ処理。詳細は「サブスクライバ クォータ」（p.4-5）を参照してください。

addBulkEntry メソッド

バルクにエントリを追加するには、次のメソッドを使用します。

```
addBulkEntry(String subscriberID, QuotaOperation quotaOperation)
```

パラメータ

subscriberID — サブスクライバの一意の ID。サブスクライバ ID のフォーマットについては、「サブスクライバ ID」（p.4-2）を参照してください。

quotaOperation — サブスクライバのクォータに対して実行するクォータ処理。詳細は「サブスクライバ クォータ」（p.4-5）を参照してください。