AITC
先端ＩＴ活用推進コンソーシアム

クラウド・テクノロジー研究部会

GAEとAWSを利用したFacebookデータ収集のプロトタイプ

2012年08月30日

株式会社イーグル
菅井 康之

背景

- ProjectLAでソーシャルデータを利用する
  - SNSの一つとして、Facebookが対象
  - Facebookからデータ収集するプロトタイプを作成

プロトタイプ作成にあたり利用した
いくつかのサービスについて
奮闘記をお届けします

Facebook API

- Facebookでは、TwitterのようなPublicなタイムライン(stream)が存在しない
  - ユーザを狙い撃ちした情報収集しか出来ない
    - タグや発言内容での検索は出来ない
  - ユーザの情報にアクセスするには、OAuthによる認可が必要である
    - ユーザのIDさえ分かればOAuthなしでもアクセス出来るが、参照可能な情報に限りがある

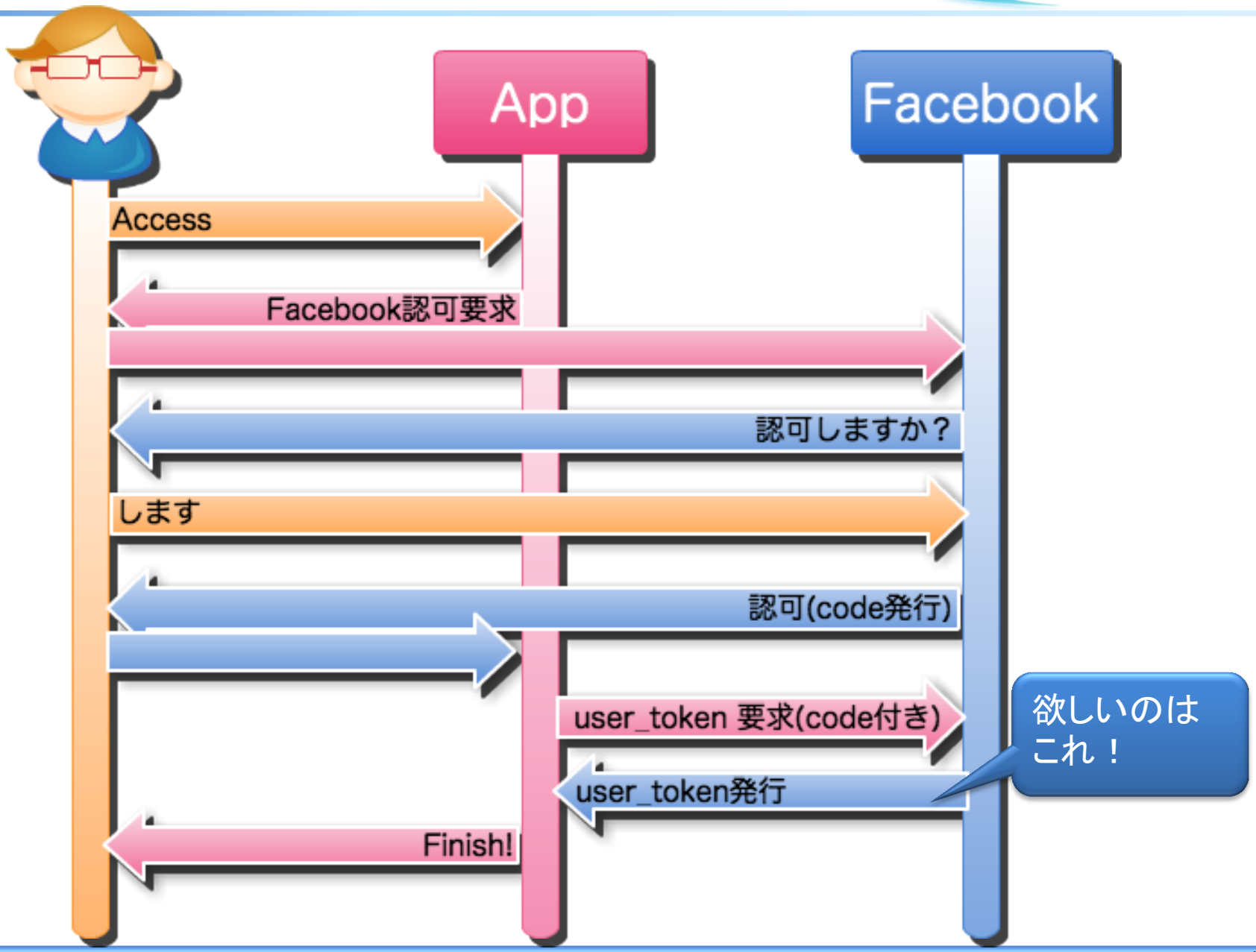
- Graph API or FQLによる情報取得
  - どちらも同じような情報獲得
  - 今回は主にGraph APIを利用

Facebook OAuth

- OAuth認証準備
  - Facebook Developerでアプリケーション登録
    - https://developers.facebook.com/apps

  - APP ID, Secret Keyを獲得

Facebook OAuth認可の流れ



OAuth用サイト

- プロトタイプ作成にあたり、ユーザを受け付けるためのサイトが必要
  - GoogleAppEngineを使用
    - 立ち上げっぱなしにしたい
    - リダイレクト先としてのドメイン名は固定したい
      - 再起動の度にドメインが変わられると困る
    - さくっと作りたい
  - 取得したトークン情報はGAEのDataStoreに格納
    - ログ以外のファイル出力は不可
  - トークンを溜め込んだ後に、エクスポート出来ない事に気付く。。
    - DataStoreエクスポート用のページを個別に作成

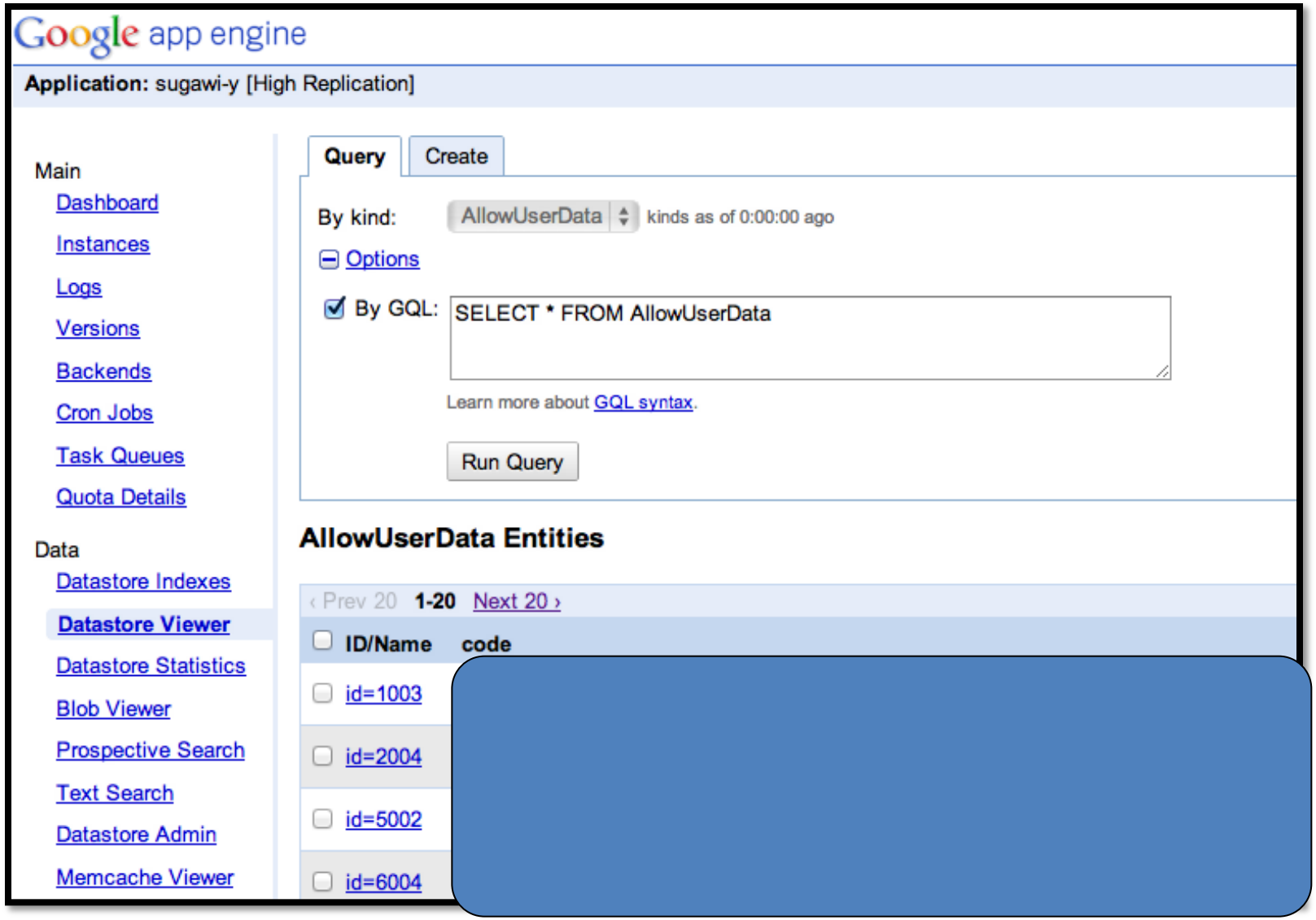
GAE Source(AccessToken取得)

```
Object code = req.getParameter("code");
if(code == null) {
    //Facebook認可用リダイレクト
    resp.sendRedirect("http://www.facebook.com/dialog/oauth?"
            + "client_id=" + APP_ID
            + "&scope=user_status" //,status_update"
            + "&redirect_uri=" + REDIRECT_URL);
    return;
}

//アクセストークン取得要求
String accessToken =
        sendMessage(
                "https://graph.facebook.com/oauth/access_token?"
                        + "client_id=" + APP_ID
                        + "&redirect_uri=" + REDIRECT_URL
                        + "&client_secret=" + SECRET_KEY
                        + "&code=" + code.toString());

//データストアに格納
AllowUserData data = new AllowUserData(accessToken, code.toString());
PersistenceManager pm = pmFactory.getPersistenceManager();
try {
    pm.makePersistent(data);
} finally {
    pm.close();
}

resp.setContentType("text/plain");
resp.getWriter().println("Thank you for your support!!");
```

```
private static final PersistenceManagerFactory pmFactory
        = JDOHelper.getPersistenceManagerFactory("transactions-optional");
```

```
@PersistenceCapable(identityType = IdentityType.APPLICATION)
public class AllowUserData {

    @PrimaryKey
    @Persistent(valueStrategy = IdGeneratorStrategy.IDENTITY)
    private Long id;

    @Persistent
    private Date date;

    @Persistent
    private String userToken;

    @Persistent
    private String code;

    public AllowUserData(String userToken, String code) {
        this.userToken = userToken;
        this.code = code;
        this.date = Calendar.getInstance(
                TimeZone.getTimeZone("Asia/Tokyo")).getTime();
    }
}
```

GAE DataStore格納イメージ

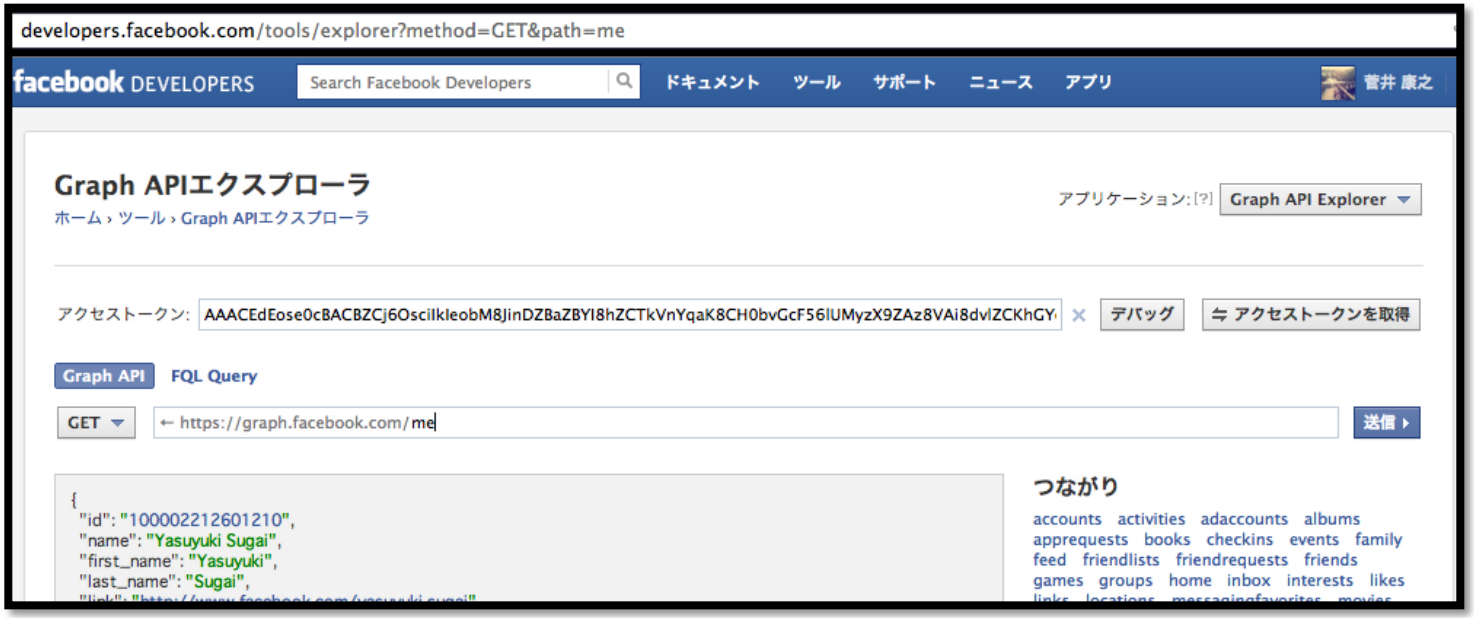


Facebook データアクセス

- GAEで溜め込んだUserTokenを使用して、情報へのアクセスを行う
  - 取得出来る情報・・・（いっぱい。。）
    - accounts activities adaccounts albums apprequests books checkins events family feed friendlists friendrequests friends games groups home inbox interests likes links locations messaging favorites movies music mutualfriends notes notifications outbox payments permissions photos picture posts scores statuses tagged television updates videos
  - 基本的には何でも取得可能
    - アプリケーションのスコープとユーザの許可次第
  - Endpointにアクセスすると、最新25件分のデータがJSON形式で返却
    - 次データへのURLも含まれているため、遡ってデータ取得を行うことも可能
  - 制限は？
    - 公式な仕様は見つけられず。。。
      - Forumを漁るとこんな記述が「600 calls per 600 seconds, per token & per IP」
      - こんなエラーも用意されてるっぽい「Facebook.GraphAPIError: (#613) Calls to stream have exceeded the rate of 600 calls per 600 seconds」

お世話になった ツール

- スコープやEndpoint、データ構造をいちいち確認するのは大変。。。(スコープも盛りだくさん)
  - Graph APIエクスプローラ
    - http://developers.facebook.com/tools/explorer



- スコープ毎にどこまで参照可能か試す事が出来る
- レスポンスもJSONのまま表示されるため、データ構造を確認する事が可能

スコープを色々与えてみると

Privateな情報にもアクセス可能

今回使用するEndpoint

- statuses(近況)
  - https://graph.facebook.com/me/statuses
  - 日時,メッセージ,コメント, いいね！,etc..
- checkins
  - https://graph.facebook.com/me/checkins
  - 日時, メッセージ,場所,Geolocation, コメント, いいね！,etc..
- links
  - https://graph.facebook.com/me/links
  - 日時, メッセージ,サムネイル画像(URL),リンク先URL,リンクタイトル,コメント, いいね！,etc..

Facebook データサンプル①

- statuses


```
{
"id":"333826936701057",
"from":{"name":"Yasuyuki Sugai","id":"100002212601210"},
"message":"Facebook APIテスト",
"updated_time":"2012-07-19T06:50:18+0000",
"likes":{"data":[{"id":"100002212601210","name":"Yasuyuki Sugai"}],
"paging":{"next":"https://xxxx"}},
"comments":
    {"data":[
        {
          "id":"333826936701057_2032625",
          "from":{"name":"Yasuyuki Sugai","id":"100002212601210"},
          "message":"これはコメント",
          "can_remove":true,
          "created_time":"2012-07-19T06:50:31+0000",
          "like_count":1,"user_likes":true,"likes":1
        }
    ],
    "paging":{"next":"https://xxx"}
  }
}
```

Facebook データサンプル②

- checkins


```
{
"id":"333829446700806",
"from":{"name":"Yasuyuki Sugai","id":"100002212601210"},
"message":"Facebook APIテスト チェックイン",
"place":
  {
    "id":"125987910786452",
    "name":"東京オペラシティ",
    "location":
        {
          "street":"西新宿3-20-2",
          "city":"Shinjuku-ku",
          "state":"Tokyo",
          "country":"Japan",
          "zip":"160-0023",
          "latitude":35.682824038082,"longitude":139.68706621217
        }
  },
"application":{"name":"Facebook for Android","namespace":"fbandroid","id":"350685531728"},
"created_time":"2012-07-19T07:06:14+0000"
}
```

Facebook データサンプル③

- links

{
"id":"322946857797935",
"from":{"name":"Yasuyuki Sugai","id":"1000022126012l0"},
"message":"Facebook API テスト links",
"picture":"http://external.ak.fbcdn.net/xxx",
"privacy":
{
"description":"Friends of Friends",
"value":"FRIENDS_OF_FRIENDS"
},
"link":"http://dailynews.yahoo.co.jp/fc/sports/soccer_oympic_team...",
"name":"U-23 監督采配に意図みえず?",
"icon":"http://static.ak.fbcdn.net/rsrc.php/v2/yD/r/aS8ecmYRys0.gif",
"created_time":"2012-07-19T07:07:51+0000"
}

データ格納先

- 取得したデータの格納先が必要
  - Facebookだけでなく、他のSNS情報も同じところに溜め込みたい
    - 各SNSから取得したデータを構造そのままに格納する
    - 単純なデータ構造、データ量は膨大に増えていく可能性も
    - トランザクション制御も要らず、ポンポンデータを放り込みたい
  - AWSのdynamodbに格納することに
    - データ量は無制限
      - 今回のようなデータ量の予測が難しい貯めこみ系に向いている（拡張性）
    - スループットを調整可能
      - データ量が少ない間はそこまで性能は求めない
      - データが増えてきた時にスループットを上げたい（効率化）
      - 小規模スタートにも向いている
    - 分散配置＆SSD
      - SNS、ユーザ増による書き込み負荷にも耐えられそう（I/O）

IAMの設定

- AWSを複数人で利用する際には、IAMの設定が必要
  - メインアカウントのcredential(access_key, secret)を渡すわけにはいかない
    - 課金サービスが使いたい放題になってしまう
  - IAMを使用する事で、個別にcredentialを払い出す事が可能
  - 利用可能なサービスやリソースのアクセス制限等を細かく指定することが可能・・・なはず。。。
    - 癖があるので、試行錯誤が必要

IAMの試行錯誤（失敗）

Show Policy Cancel

```
{
  "Statement": [
    {
      "Sid": "Stmt1344253861640",
      "Action": [
        "dynamodb:BatchGetItem",
        "dynamodb:DeleteItem",
        "dynamodb:ListTables",
        "dynamodb:DescribeTable",
        "dynamodb:GetItem",
        "dynamodb:PutItem",
        "dynamodb:Query",
        "dynamodb:Scan",
        "dynamodb:UpdateItem",
        "dynamodb:UpdateTable"
      ],
      "Effect": "Allow",
      "Resource": [
        "arn:aws:dynamodb:ap-northeast-1:*:*"
      ]
    }
  ]
}
```

ほんとはtable/project_la
にしたい。。

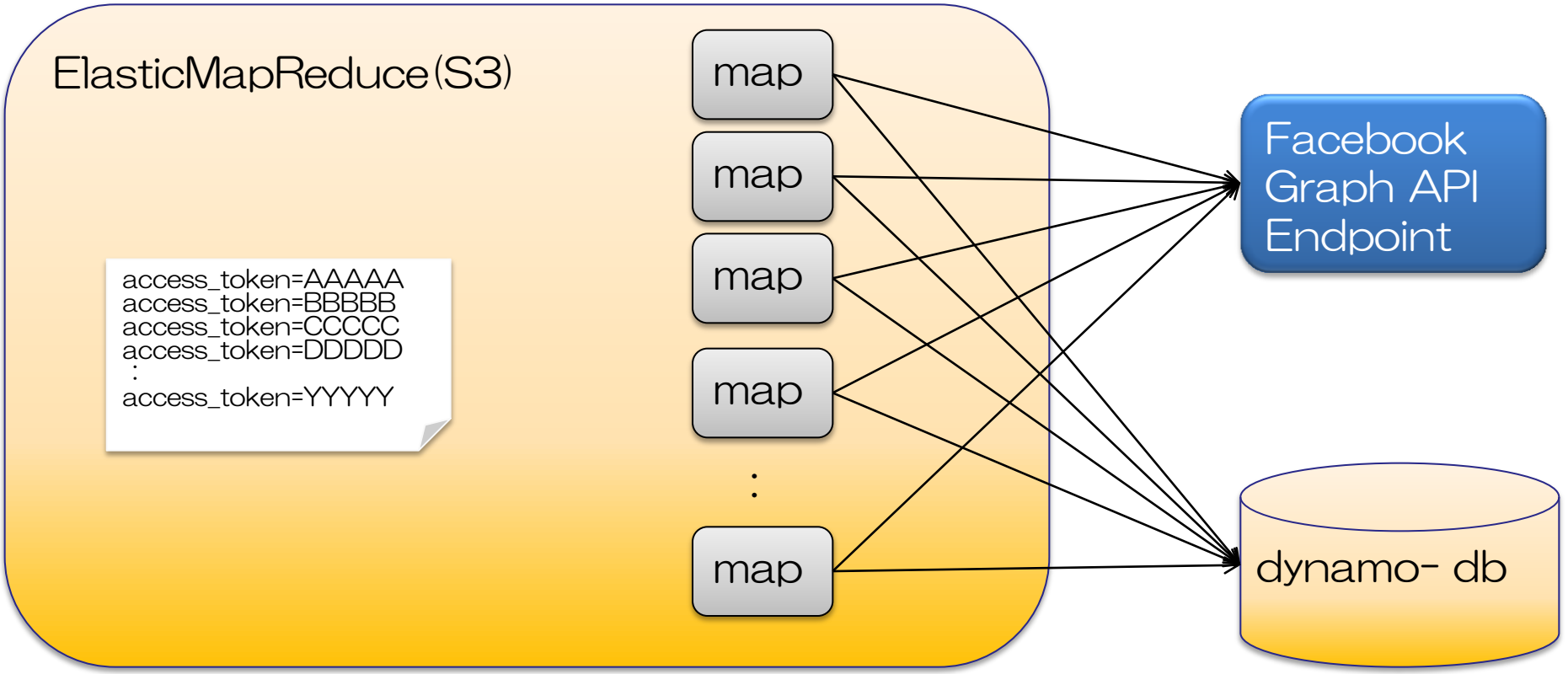
Show Policy

```
{
  "Statement": [
    {
      "Sid": "Stmt1346152780282",
      "Action": [
        "dynamodb:ListTables"
      ],
      "Effect": "Allow",
      "Resource": [
        "*"
      ]
    }
  ]
}
```

これが無いとConsoleで参照
できなくなる。。

データ収集 実行環境

- ユーザ数(=access_token数)増加を考慮し、AWSのEMR上に実行環境を構築



Source①

```
List<String> targetEndpointList = new ArrayList<String>();
targetEndpointList.add("statuses");
targetEndpointList.add("links");
targetEndpointList.add("checkins");

for(String endpoint : targetEndpointList) {
    String requestUrl = "https://graph.facebook.com/me/" + endpoint + "?" + userToken;
    do {
        //情報取得
        String response = FacebookUtil.sendMessage(requestUrl);
        Map decodeRes = JSON.decode(response, Map.class);
        for(Map feed : ((List<Map>)decodeRes.get("data"))) {

            AbstractFacebookData data = AbstractFacebookData.parse(endpoint, feed);
            if(!isTargetData(data.getMessage())) {
                continue; //#aitc3以外の投稿
            }
            for(FacebookComment comment : data.getComments()) { //コメントも対象チェック(#aitc3)
                if(!isTargetData(comment.getMessage())) {
                    data.deleteComment(comment.getId()); //収集対象外のコメントは除去してJSON組み直し
                }
            }

            //Dynamo格納用Object
            DataStore store = new DataStore();
            store.setId(data.getId());
            store.setTime(data.getTime());
            store.setData(data.toString());
            store.setNote(endpoint);

            synchronized(mapper) {
                mapper.save(store); //Dynamo格納 Insert AND Update
            }
        }

        requestUrl = getNextPageUrl(decodeRes); //Paging
    } while(requestUrl != null);
}
```

Facebookデータアクセス、
Dynamo格納部分のみ
抜粋（一部簡略）

Source②+dynamo格納イメージ

AITC 先端ＩＴ活用推進コンソーシアム

```
private static void init() throws IOException  {
    AWSCredentials credentials = new PropertiesCredentials(
            AmazonDynamoDBSample.class.getResourceAsStream("AwsCredentials.properties"));

    AmazonDynamoDBClient dynamoDB = new AmazonDynamoDBClient(credentials);
    dynamoDB.setEndpoint("http://dynamodb.ap-northeast-1.amazonaws.com"); //Asia Pacific Tokyo
    mapper = new DynamoDBMapper(dynamoDB); //static DynamoDBMapper mapper;
}
```

```
@DynamoDBTable(tableName = "project_la")
public class DataStore {
    private String id;
    private String time;
    private String source;
    private String data;
    private String note;

    public DataStore() {
        this.source = "facebook";
    }

    @DynamoDBHashKey
    public String getId(){ return id; }
    public void setId(String id){ this.id = id; }

    public String getTime(){ return time; }
    public void setTime(String time){ this.time = time; }

    public String getSource(){ return source; }
    public void setSource(String source){ this.source = source; }

    public String getData(){ return data; }
    public void setData(String data){ this.data = data; }

    public String getNote(){ return note; }
    public void setNote(String note){ this.note = note; }
}
```

Services ▾ Edit Shortcut ▾

DynamoDB: Explore Table: project_la

Region: Asia Pacific (Tokyo) ▾

List Tables | **Browse Items**

Scan / Get | Go | New Item | Edit Item | Copy to New | Delete Item

id	time	source	data	note
"338292329587851"	"2012/08/02 02:04:00"	"facebook"	"{"id":"338292329587851","from":{"name":"Yasuyuki Sugai","id":"100002212601210"},"message":"4種目終了。 日本人による金争いになるとは思わなかった #aitc3","updated_time":"2012-08-01T17:04:00+0000"}"	"statuses"

課題

- APIの仕様変更に依存する
  - Facebookでは7月にuser_tokenを無期限で使用出来るoffline_accesssを廃止に
    - user_tokenの有効期限が60日に
    - RefreshTokenするにはユーザ認可が必要になる模様。。
  - 随時情報収集をしないといけない
    - 仕様を理解するためには自分で手を動かしてみる
- dynamodbでは、ごっそり取得に適さない？
  - Key指定の検索は早いが、全データごっそり取る場合にはそれなりに時間がかかりそう（当たり前だけど）
    - 1回のscanで1Mまでしか取得できない
    - Range keyを上手く活用することで回避できるか？
    - 最近Binaryにも対応したらしい

AITC
先端ＩＴ活用推進コンソーシアム

ご清聴ありがとうございました

END