法政大学情報メディア教育研究センター研究報告　*Vol.31*　2017 年

IT を利用した教育システムに関する最新動向の調査報告

A Report about Latest Trends of Education Support Systems Using IT

藤井　聡一朗 [1)]
Soichiro Fujii

[1)] 法政大学情報メディア教育研究センター

Recently, many technology standards about education support systems are published such as Learning Tools Interoperability(LTI), xAPI, Caliper Analytics, etc. These standards enable to make new type of education support systems. One of those are called Next Generation Digital Learning Environment(NGDLE) or LearningOS. In this report I show results of investigations about latest trends of education support systems like NGDLE and technical standards used to making new type of education support systems.

Keywords : NGDLE, LMS, e-Learning, LTI, Caliper Analytics

1. はじめに

近年の IT の進歩により多くの教育機関で IT を利用したシステムの導入が進んできている．高等教育では Moodle や Sakai，Canvas といった Learning Management System(LMS)が広く使われるようになってきたが，近年ではクラウド上で提供されるサービスを大学や個々の教員が利用することが増えてきた．また教育支援システムに関する技術標準も広く利用されるようになり，新規のものも多く発表されている．そういった流れの中，旧来の LMS の考えとは異なった次世代の教育プラットフォームが提唱されている．その代表的なものが Next Generation Digital Learning Environment(NGDLE)[1]だ．

NGDLE は文字どおり IT を用いた次世代の教育環境を表すバスワードであり，近年の IT サービスのクラウド化やビッグデータを用いた Learning Analytics の導入などを視野に入れ考案されたプラットフォームである．NGDLE の特徴は標準規格を用いた分散型システムであることだ．各機能は Learning Tools Interoperability(LTI)[2]に対応したプラグイン形式で実装される．LTI を採用することにより，サードパーティのクラウドサービスとして提供される LTI ツールとの連携が可能になる．また，Learning Analytics に対応するために xAPI[3]や Caliper Analytics[4]のような標準規格を使って学習履歴をデータとして収集する．学習履歴の収集にこのような標準規格を用いることで，各組織で解析結果を表示するダッシュボードや解析エンジンなどを共有できるというメリットがある．

こういった IT を利用した教育システムに関する最新動向を把握するために NGDLE で用いられる技術標準について調査を実施した．この報告書ではその調査結果を報告する．

2. 調査対象とした技術標準

今回は下記の技術標準について，それぞれを組み込んだプロトタイプシステムの開発を行い調査を実施した．

- LTI
- Caliper Analytics

Caliper Analytics と役割が重複するため今回は調査の対象から除外したが，xAPI は NGDLE を構成する重要な技術標準の一つである．これからそれぞれの標準規格の概要と実装について説明する．

原稿受付 2017 年 3 月 25 日
発行　　2017 年 6 月 30 日

2.1 LTI

LTI は国際的な標準化団体である IMS Global Learning Consortium(IMS GLC)[5]が策定する教育支援プラットフォームの連携のための標準規格である. LTI では LTI Consumer と LTI Provider という二つの役割が定義されており，LTI Consumer は LTI Provider を外部ツールとして利用することができる. LTI 連携によって LTI Consumer と LTI Provider 間ではユーザ情報や科目情報などを共有することができる.

NGDLE では本体に LTI Consumer の機能を持ち，各種のサブモジュールを LTI Provider として実装する構成が提案されている. このような構成をとることにより，柔軟な機能追加が可能になり，クラウドサービスとしてサードパーティから提供される機能も組み込むことができる.

LTI 機能の実装は IMS GLE の GitHub[6]上でライブラリが公開されている. 今回は各技術標準の調査の一環として LTI Provider 機能を有するプロトタイプシステムを開発した. 今回開発したシステムの使用言語は Java を用いたのでライブラリも GitHub 上に公開されている Java 用のもの[7]を利用したが，ライブラリは各種言語のものが用意されている.

2.2 Caliper Analytics

Caliper Analytics は Learning Analytics のための標準規格で，LTI と同様に IMS GLE が策定している.類似の標準規格に xAPI があるが，今回は Caliper Analytics を調査対象とした. Caliper Analytics は Learning Analytics 全般を扱うための規格とされているが，現在のバージョンでは学習履歴のデータ形式の定義にとどまっている.

Caliper Analytics も IMS GLE の GitHub 上にライブラリが公開されており，学習履歴を整形してデータストアへ送信するための API などが提供されている. 今回は検証のためプロトタイプに学習履歴を格納する機能を実装した. 実装言語は前述した通り Java なので Java 用のライブラリ[8]を利用している.

Caliper Analytics の学習履歴を格納する場所はイベントストアと呼ばれ，GitHub 上にはその実装も提供されており，プロトタイプにて利用しようと試みたが，あまりメンテナンスされていないようでうまくセットアップできなかったため今回は MongoDB を用いて自作した.

3 プロトタイプシステムの開発

今回は検証のため前述した標準規格を用いて簡単なプロトタイプシステムを実装して開発手法と動作について確認した. ここではその実装の詳細について解説する.

今回開発したプロトタイプシステムの概要は下記のとおりである.

- 言語: Java
- フレームワーク: Spring Boot
- LTI バージョン: 1.0
- LTI ライブラリ: basiclti-util-java
- Caliper ライブラリ: caliper-java-public

開発したプロトタイプシステムは LTI Consumer として用意した Sakai と連携させ，自作のイベントストアに学習履歴を格納するテストを実施し正常に動作することを確認した. それぞれの実装の詳細について下記に示す.

3.1 LTI Provider(LTI1.0)の実装

実装方法の詳細は basiclti-util-java の GitHub 上に記載されているのでここでは概要について説明する. ライブラリでは Spring フレームワークと AspectJ, maven を利用しているため，それらをセットアップした上で機能を実装した. 以下にその実装手順と該当箇所のソースコードを示す.

① pom.xml ファイルの依存先に basiclti-util を追加する.

```
<dependency>
        <groupId>org.imsglobal</groupId>
        <artifactId>basiclti-util</artifactId>
        <version>1.1.2</version>
</dependency>
```

② LTI Consumer との接続のための key と secret を管理する KeyService を追加する.(今回はプロトタイプのため定数を返しているが，本来は key に対応した secret を返す必要がある.)

```
@Service
public class MockKeyService implements LtiKeySecretService
{
    public String getSecretForKey(String key) {
```

```
        return "secret";
    }
}
```

③　AspectJ を設定する.

```
@Configuration
@EnableAspectJAutoProxy
public class AOPConfig {
          @Autowired
          private LtiKeySecretService keyService;
          @Bean
          public LtiLaunchVerifier ltiLaunchVerifier() {
                    return                                  new
LtiLaunchVerifier(keyService, new LtiOauthVerifier());
          }
}
```

④　LTI のエントリーポイントのコントローラを実装する. LTI Consumer にはこのリクエスト先を起動 URL として設定する.

```
@Lti
@RequestMapping(value = "/lti", method = RequestMethod.POST)
public String ltiEntry(HttpServletRequest request,
 LtiVerificationResult result) {
  if(!result.getSuccess()){
    return "error";
  } else {
    String  uid =
     request.getParameter("lis_person_sourcedid");
    String familyName =
     request.getParameter("lis_person_name_family");
    String givenName =
     request.getParameter("lis_person_name_given");
    System.out.println(
     result.getLtiLaunchResult().getUser().getId());
    return "success";
  }
}
```

3.2 Caliper Analytics の実装

Caliper Analytics の実装については LTI のようにcaliper-java-public のGitHub 上には詳細が記載されていない. ライブラリの使用方法を確認するために今回はプロジェクト内にあるテストコードを参考にしたため独自の解釈と実装が多く含まれる点はご容赦いただきたい. 今回はテストとして Caliper Analytics で定義されているSessionEvent を生成し, イベントストアに格納した. 以下にその実装手順と該当箇所のソースコードを示す.

①　caliper-java プロジェクトを maven でローカルリポジトリにインストールし, pom.xml に依存先を追加する. (LTI と違ってインストールが必要な点に注意)

```
<dependency>
    <groupId>org.imsglobal.caliper</groupId>
    <artifactId>caliper-java</artifactId>
    <version>1.0.0</version>
</dependency>
```

②　学習履歴をイベントストアへ送信する CaliperEventService を作成する. このServiceではセンサーの初期化とイベントの生成を行なっている.

```
@Service
public class CaliperEventService {
private static String HOST = "http://requestb.in/14stje71";
private static String API_KEY = "FEFNtMyXRZqwAH4svMakTw";
private Sensor<String> sessionSensor = new Sensor("sensorid?");
private SoftwareApplication edApp =
 SoftwareApplication.builder().id("edAapp:peas2_be").build();

public CaliperEventService() {
          Options opts = new Options();
          opts.setHost(HOST);
          opts.setApiKey(API_KEY);
          sessionSensor.registerClient("sessionSensor",
           new Client("clientid?", opts));
}

public void sendSessionEvent(String uid,
 String name, String sessionId) {
          Person actor = buildPerson(uid, name);
          DateTime now = DateTime.now();
          SessionEvent sessionEvent = SessionEvent.builder()
           .actor(actor)
           .action(Action.LOGGED_IN.getValue())
           .object(edApp)
           .target(MediaLocation.builder()
            .id("https://mm16.media.hosei.ac.jp/peas2_be/")
            .name("Top Page")
            .build())
          .generated(Session.builder()
           .id("peas2_be/session-"+sessionId)
           .name("session-"+sessionId)
           .actor(actor)
           .dateCreated(now)
           .startedAtTime(now)
           .build())
          .build();
          sessionSensor.send(sessionSensor, sessionEvent);
}

private Person buildPerson(String uid, String name) {
          return Person.builder()
           .id("user:peas2_be/"+uid)
           .name(name)
           .build();
}
```

③　LTI のエントリーポイントにイベント格納のコードを埋め込む.

```
caliperEventService.sendSessionEvent(uid,
 "Hosei Taro", session.getId());
```

4　まとめ

この報告書では IT を利用した教育システムに関する最新動向について調査し，次世代の教育プラットフォームとして提唱されている NGDLE とその関連技術について調査した．NGDLE の核となる標準技術は LTI と Caliper Analytics(または xAPI)であり，今回はそれらを用いたプロトタイプを開発することにより詳細を調査した．各標準技術については GitHub 上にライブラリが公開されているものの，そのメンテナンス状況や開発ドキュメントの充実は未だ充分なレベルとは言えない．今回開発したプロトタイプではイベントストアに自作のものを利用したが，OpenLRS[9]や Learning Locker[10]などの標準規格に対応したオープンソースのものが徐々に公開されてきている．プロトタイプで実装した Learning Analytics 面の機能はデータを格納するのみにとどまったが，今後は解析エンジンやダッシュボードなどもイベントストアのようにオープンソースソフトウェアとして公開されるものが増えていくと予想される．

参考文献

[1]NGDLE, http://www.imsglobal.org/tags/ngdle, (参照 2017-06-21)
[2]LTI, https://www.imsglobal.org/activity/learning-tools-interoperability, (参照 2017-06-21)
[3]xAPI, https://www.adlnet.gov/xapi/, (参照 2017-06-21)
[4]Caliper Analytics, http://www.imsglobal.org/activity/caliperram, (参照 2017-06-21)
[5]IMS Global Learning Consortium, http://www.imsglobal.org, (参照 2017-06-21)
[6]IMS GLC GitHub, https://github.com/IMSGlobal, (参照 2017-06-21)
[7]basiclti-util-java, https://github.com/IMSGlobal/basiclti-util-java, (参照 2017-06-21)
[8]caliper-java-public, https://github.com/IMSGlobal/caliper-java-public, (参照 2017-06-21)
[9]OpenLRS, https://www.apereo.org/projects/openlrs, (参照 2017-06-21)
[10]Learning Locker, https://learninglocker.net, (参照 2017-06-21)

謝辞

本研究は JSPS 科研費 JP15K00493 の助成を受けたものです．