Learning Analytics Platform, towards an open scalable streaming solution for education

Nicholas Lewkow
McGraw-Hill Education
281 Summer St.
Boston, MA
nicholas.lewkow
@mheducation.com

Neil Zimmerman
McGraw-Hill Education
281 Summer St.
Boston, MA
neil.zimmerman
@mheducation.com

Mark Riedesel
McGraw-Hill Education
281 Summer St.
Boston, MA
mark.riedesel
@mheducation.com

Alfred Essa
McGraw-Hill Education
281 Summer St.
Boston, MA
alfred.essa
@mheducation.com

ABSTRACT

Next generation digital learning environments require delivering *just-in-time feedback* to learners and those who support them. Unlike traditional business intelligence environments, streaming data requires resilient infrastructure that can move data at scale from heterogeneous data sources, process the data quickly for use across several data pipelines, and serve the data to a variety of applications. As a solution to this problem, we have designed and deployed into production the Learning Analytics Platform (LAP), which can ingest data from different education systems using standardized IMS Caliper events. The education events are triggered by student and instructor activity within Caliper instrumented learning systems. Once sent to the LAP, events are transformed and stored in a data store where they can be used for student, educator, and administrator visualizations as well as education driven analytics research. Two McGraw-Hill Education platforms, Connect, used for higher education, and Engrade, for K-12, are currently instrumented to send the LAP event data which in turn feeds visualizations for educational insight. Future plans for the LAP include collection of education event data from a wide variety of proprietary and open source education platforms, computational engines for predictive analytics, and an open API for third-party analytics using LAP data.

Keywords

Learning Analytics, Event Processing, Heterogeneous Data, Streaming Data, Parallel Architecture

1. INTRODUCTION

It is the goal of next generation digital learning systems to use big data and analytics to advance learning outcomes. These next generation systems should be able to provide just-in-time feedback to students and educators with an aim to increase the efficiency and effectiveness of digital education. Further, with the increasing instrumentation of all digital media, digital learning environments should be instrumented in a way that allows important education data to be collected in a standardized fashion for both real-time and after-the-fact (batched) data analysis. This task requires large scale processing of streaming data utilizing massively parallel architectures which may ingest, store, and analyze data in real-time.

Our solution to this problem is the Learning Analytics Platform (LAP). The LAP is designed to ingest educational data from present and future education platforms in the form of standardized events using the IMS Caliper spec [2]. Two existing education platforms, Connect for higher education and Engrade for K-12, have already been instrumented to create and ship Caliper education events to the LAP. Once ingested by the LAP, the data is transformed and stored for building of visualizations used for educational reports [4]. These 'insights' include several real-time statistics for students and educators including time-spent, outcomes, submission times near due dates, attendance, and class performance comparisons. Additionally, messages indicating negative trends, such as repeatedly starting assignments near or after the due date, are also presented to the user.

To meet the requirement of providing just-in-time feedback to students and educators, an analytics platform must also be have a parallel architecture which may effectively ingest streaming data. There are several proposed requirements for a streaming data architecture, including the ability to handle data imperfections, generate predictable outcomes, and to guarantee data safety and availability [5]. Additionally, the architecture should be automatically scalable and fault tolerant for both software and hardware failures, particularly, it must not lose any event data under any circum-

stance. Our architecture has the additional requirements of ingesting data from heterogeneous sources and performing data transformations using different data pipelines. The LAP fulfills all of the above requirements in its current version with further refinements and additions planned for the near future.

Details about the LAP are discussed in the following sections including information about the standardized data format which was used, IMS Caliper events, as well as specifics on the LAP architecture and performance. Information regarding future versions of the LAP is also discussed followed by concluding remarks.

2. STANDARDIZED CALIPER EVENTS

With the continual adoption of digital education systems a global standard for educational event data, which is generated from a large diversity of heterogeneous systems, has become increasingly sought after. While there has been advancement in this area by the Tin Can API [1], the IMS Global Learning Consortium has proposed a schema-driven solution to this problem with their Caliper event spec [2]. JSON-LD (linked data) is used for the Caliper events as a way to link specific, normalized fields within a set of events [3]. Using the Caliper events, data from heterogeneous learning systems can be created, transmitted, and collected for analysis in a global and standardized fashion.

Caliper events strive to create a generalized framework that can be utilized by all types of learning events ranging from a student using an interactive education tool, such as a learning game, to an educator recording attendance in their class. The Caliper events are based on the data triple of **"Actor" - "Verb" - "Object"**. As an example, the event for a student submitting an assessment (homework, quiz, test, etc.) would have the **Actor** be the student, the **Verb** be the submission of an assessment, and the **Object** would contain information on the assessment being submitted, potentially how long it took to complete.

In order to utilize the Caliper event spec, learning platforms must be instrumented to create events when actions occur by either students or educators. Currently the Connect and Engrade systems are instrumented to to create Caliper events when actions occur, and to send the events to the LAP. Instrumentation is unique to the system in question and greatly depends on how that system's data is stored. In the case of Connect, Caliper events are created through a series of database triggers when actions are taken by students. The database triggers are automated to create the Caliper events using tables from their system databases when new information is passed to the system from a user. Future plans include instrumentation of several new external systems, allowing for increasingly rich data in the LAP for analysis and visualization.

3. ARCHITECTURE

Increasing instrumentation of sensors and digital media requires streaming analytics architectures to analyze data in real-time. Attention was paid to the development and design of the LAP architecture to ensure that it met all requirements of a parallel streaming system containing both a data store and an analytics engine. Key features include auto-scaling fundamental architecture components with varying load, fault tolerance for both hardware and software failures, and the ability to process data and have it available for output API access immediately.

In the simplest of descriptions, the LAP is designed to:

1. Receive learning events from external applications through an ingestion API.
2. Send raw events directly to long-term storage.
3. Validate each event for expected fields and types.
4. Process the events, which requires application dependent data transformations.
5. Store those events in the data store according to application dependent schema.
6. Query the data store and perform transformations and aggregations as needed for the output API.

Figure 1 displays a high level view of the LAP architecture with learning events from three separate external applications coming into the LAP. Once the events are received by the LAP they are transformed according to an application specific schema and stored in the data store. When a user requests an insight visualization, the LAP output API is called by the external application. This triggers the results and analytics service to query the data store, aggregate and transform the data as needed, and pass it to the insight service where the data is used to build the appropriate insight visualization which is then passed to the user.

The ingestion API and collection service are configured to receive IMS Caliper events sent from external applications through sensors which have to be implemented in those external applications. To maximize performance over high-overhead HTTP, several events are sent simultaneously in an event container. The number of events sent in a single container can range from one to tens of thousands. Once a container of events is received it is immediately sent to a long term storage system for backup purposes. The container is then opened and the events within are validated, transformed into an application specific schema, and sent to the data store.

The data store utilizes the non-relational database MongoDB, which allows for great flexibility in data model schema. Events from external applications are not guaranteed to come into the LAP in chronological order so the data model schemas were developed to create a deterministic data store state from events coming in arbitrary order. The data model employed for the current two applications consists of a two schema model; one document type holds the student level data and the other holds the class level data. Building of the insight visualizations then requires the results service querying N student documents for a class with N students, and 1 document per class for a total of $N + 1$ documents.

The insight service is built external to the LAP to provide the user with the requested analytical visualization. Once the data store is queried and the appropriated documents are

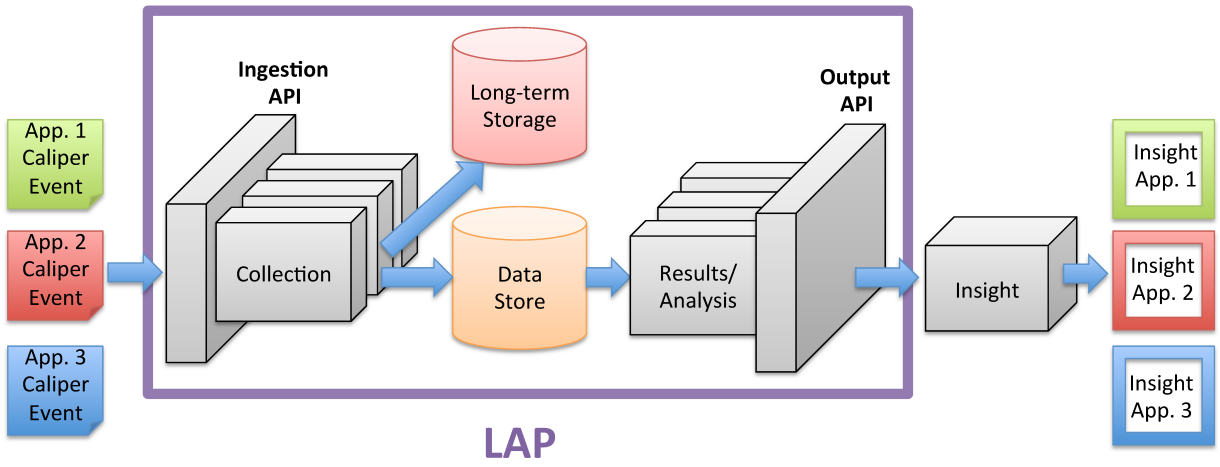

Figure 1: High level view of the LAP architecture. Learning events are passed into the LAP through the ingestion API and visualizations 'insights' are produced through the insight service calling the LAP's output API. Internally, the LAP consists of a collection service, a data store, long-term storage, and a results service, which also performs analytics. Several instances of the collection and results services run in parallel.

returned, they are aggregated, transformed, and returned to the output API to provide the client with data to build the visualization. This process is very lightweight and fast, allowing for quick feedback and response to the user who has just finished an assessment.

The technology used for the LAP was chosen to be lightweight, reliable, and adaptable for future system iterations. The collection, results, and insight services were all built using Node.js which allows for asynchronous communication run in parallel across several service instances. The analytical visualizations were built using JavaScript with AngularJS and D3.js frameworks which provide fast, responsive, and interactive interfaces that are customizable. The data store was built using MongoDB with a three member replica set, allowing for highly available data and low latency access for both read and write operations. Amazon Web Services (AWS) were used to host the LAP utilizing load balancing technologies routed to EC2 instances. An AWS S3 bucket was used for long-term storage of the raw event containers. Details pertaining to the performance of the LAP as well as future plans for the system follow.

4. PERFORMANCE

Performance is very important to the success of the LAP. It is imperative that the LAP be able to ingest data from several external systems during their peak times simultaneously in a manner which does not delay the real-time analysis on the output of the system.

In testing the performance of the LAP, two main points within the system were identified as the potential bottlenecks. The first of these points is the ingestion of events from external applications into the LAP while the second is the querying, aggregation, and analysis done by the results service prior to returning data to the output API.

For the current two external systems sending data to the LAP, our anticipated peak load is around 0.1 MB/sec. While this load is not particularly large, future plans include ingesting events from many more external systems, so the desired performance should easily be at least ten times larger at around 1 MB/sec. To test the performance from event reception to data store insertion, an automated script was developed which creates 10 threads with each sending a series of containers with varying numbers of events to the LAP running three instances of the collection service. The processing time, from data being sent from an external application to be inserted into the data store, was then measured as a function of number of events per container. Figure 2 displays the results of this test with collection rates as MB/sec and the number of events per container ranging from 1 to 1000. The results shown in Figure 2 are informative for a few reasons. First, it is clear that high collection rates into the system requires more than one event to be sent per container. In particular, the LAP can process 0.1 MB/sec or more if the events are sent with at least 4 per container. To reach our desired bandwidth of ten times our current peak, 1 MB/sec, requires sending about 75 events per container or more with three collection instances. The second realization from Figure 2 is that the collection rate of the system somewhat flattens out between 500 and 1000 events per container, making it less efficient to process these larger containers. This effect ultimately has to be weighed against the network bandwidths in sending events from external applications and has not yet been tested. It should also be mentioned again that these performance tests were done with three instances of the collection service. Increased rates could always be achieved by increasing the number of collection service instances, but these tests were done to determine collection rates for a static number of instances.

The second potential bottleneck in the LAP is the querying of the results service and the load on the output API. Currently the LAP is in a trial mode with tens of thousands of users. For this relatively low number of users the load on the output API is not a major concern and detailed performance testing has not yet been done. The implementation

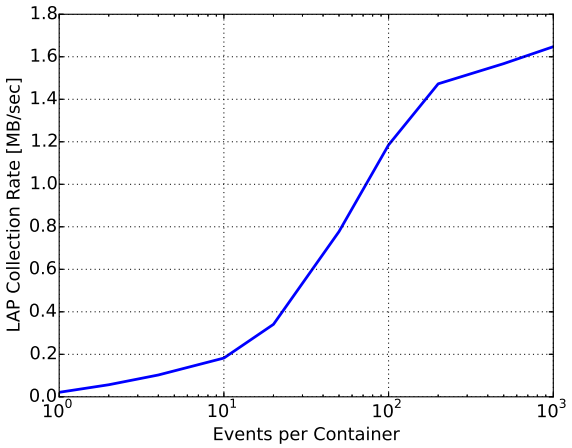

Figure 2: The collection rate of Caliper events, in [MB/sec], processed by the LAP is shown as a function of number of events per container. 10 parallel threads were used to send data to the LAP during this test, simulating the load from several external systems sending events simultaneously. The system tested had 3 instances of the collection service operating in parallel.

of auto-scaling within AWS, however, should easily handle large spikes in front end usage during peak hours.

5. FUTURE VERSIONS

The current LAP is the first iteration of a production system. It supports two external education systems, can support more than ten times the anticipated peak load, and has a modest analytics layer within its architecture. The next version of the LAP is currently being developed with the goal of supporting many more external systems with totals of tens of millions of users. In addition to increased user load, the future plans for the LAP include a substantial computational layer, opening up the possibility for richer analytics, as well as an open API for third party analytics to be done using LAP data.

The initial success of the current version of the LAP has led to plans for instrumentation of several new educational systems so their data may be ingested by the LAP. To be able to handle the increased numbers of users with the LAP, several changes and additions are needed. The most drastic of these changes is switching to a completely AWS system, fully utilizing Amazon cloud technologies [6]. Moving the LAP to a full AWS stack will allow for massive scalability and the ability to store data, perform analytics, and give support to millions of students, educators, and researchers. Further, future versions of the LAP will also have the ability to perform more advanced analytics including predictive analytics, machine learning algorithms, and large distributed calculations and aggregations. Incorporating a distributed calculation layer into the LAP will allow for a richer set of analytics to be performed and thus give the ability for deeper insight into large educational data sets.

Implementation of an open API for data consumption and analytics by third parties is also planned for the LAP. One of the intended features of the LAP is the lack of PII data held within the data store. The de-identified of the LAP data allows for third parties to ingest and do analysis on our data without concern for privacy. Creating an open API for the LAP will help push the fields of learning analytics and educational science by allowing researchers greater access to student and educator data.

6. CONCLUSIONS

We have built a platform able to ingest, store, and analyze data from external learning applications in a scalable fashion. Two existing applications, Connect and Engrade have been instrumented to create and send standardized Caliper learning events to the LAP. Once received, the learning events are transformed within customized data pipelines and stored within a fast data store, implemented using non-relational MongoDB. Analysis is done on the stored learning events, creating visualizations of educational insight for students and educators. The architecture of the LAP allows for just-in-time feedback with the insight visualizations to its users. The current version of the LAP is able to process and store education events ten times faster than is required for peak usage by the current two applications interfaced with the LAP. Future versions are planed for the LAP and will include a complete backend stack which is hosted by AWS and able to auto-scale across the entire platform. Additionally, an advanced computational layer and open API are planed for future versions of the LAP. It is our vision that present and future iterations of the LAP may provide the analysis, high quality educational data, and predictive analytics to drive the next generation of education.

7. ACKNOWLEDGMENTS

This paper is based on work supported by the McGraw-Hill Education Digital Platform Group (MHE DPG). We would like to extend our appreciation for all the help and the informational support provided by the Analytics team at DGP and the MHE CDO Stephen Laster. Despite provided support, any opinions, findings, conclusions or recommendations expressed in this paper are those of the authors and do not necessarily reflect positions or policies of the company.

8. REFERENCES

[1] Experience API Version 1.0.0. Technical report, Advanced Distributed Learning, 2013.

[2] Learning Measurement for Analytics. Technical report, IMS GLOBAL Learning Consortium, 2013.

[3] JSON for Linking Data, 2015. http://json-ld.org/.

[4] J. Feild. Improving student performance using nudge analytics. In *Educational Data Mining*, 2015.

[5] M. Stonebraker, U. Cetintemel, and S. Zdonik. The 8 Requirements of Real-Time Stream Processing. *SIGMOD Record*, 34(4), 2005.

[6] E. Swensson. Big Data Analytics Options on AWS. 2014.