Complex event processing distributed architecture for Massive Open Online Courses

Sakina Benarbia
Université Internationale de Rabat, Faculty of Computing and logistics, TicLab
Université Mohammed V Ecole Mohammadia d' Ingénieurs, Rime
Rabat, Morocco
sakina.benarbia@uir.ac.ma

Nabih Alaoui
Université Internationale de Rabat, Faculty of Computing and logistics, TicLab
Rabat, Morocco
nabih.alaoui@uir.ac.ma

Samir Bennani
Université Mohammed V Ecole Mohammadia d' Ingénieurs, Rime
Rabat, Morocco
sbennani@emi.ac.ma

***Abstract*—Big data technologies are becoming widely used, not only for recording but also for analyzing human generated data. Indeed, the quick technological evolutions of the recent years have reached many fields especially education where the use of connected computing devices (e.g. smart devices, computers, servers, etc.) is continuously growing. The generated data in the education field is becoming extremely voluminous, especially in eLearning, (e.g. Massive Open Online Courses (Moocs)). In this regard, real time data processing has become one of the main challenges in Moocs, as data coming from diversified sources must be processed with respect to semantics. As such, this paper investigates the use of Semantic Complex Event processing in the analysis of the data generated through Moocs. This paper also presents a distributed complex event processing system for learning activities in Moocs.**

Keywords— Big Data; Semantic web; e-learning; event processing; complex event processing; Mooc;

I. INTRODUCTION

Nowadays, a great volume of information is generated by human activities, this amount of information comes out from various sources in an infinity types of filed such as education, health, and driving. There for, a wide variety of technologies has been created and adapted to analyse and manipulate these types of information which are called big data. These techniques are issued from several fields including statistics, computer science, applied mathematics, and economics. This means that an organization that intends to derive value from big data has to adopt a flexible, multidisciplinary approach.

In our case the focus is on Massive Online Open Courses (MOOCs). Moocs are a fundamental departure from traditional eLearning systems. MOOCs are based on an open learning plate forms to vast amounts of users: the leading MOOC platforms are COURSERA, IVERSITY and EDX. A single course enrollment in MOOCs can range between 10,000 to 200,000 students, therefore, Moocs provides a potentially rich venue for large scale digital data (e.g., student course comments, temporal and geo-location data, etc.).

There are two closely related research fields devoted to enhancing teaching and learning through the study of the learners' behavior: learning analytics and educational data mining EDM. As definition for learning analytics "the measurement, collection, analysis and reporting of data about learners and their contexts, for purposes of understanding and optimizing learning and the environments which it occurs", while EDM is dedicated to and applying computerized methods to detect patterns in large collections of educational data that would then be hard or impossible to analyze due to the enormous volume of data within which they exist. Both disciplines rely on collecting learners' data from different sources to apply their different techniques [1]. In our research work, the analyses are about the huge amount of generated data in real time, we are not using EDM or Learning analytics. In our case we will use a CEP (complex event processing) system, Complex Event Processing is a technique for tracking, analyzing, and processing data as events happen. This information is processed and communicated based on business rules and processes. The information is extracted from distributed message-based systems that allow system users to specify the information that is of interest to them. High speed data processing is needed to maintain stream processing using CEP.

The objective of this paper is to present a CEP distributed system for learner activities in Moocs. We will remind what we have already done in functional model and continue to the distributed system.

This paper is structured as follows. The second section presents our Functional semantic complex event model. The third section describes a distributed complex event processing system for learner activities. And we will conclude in the fourth section.

III. RELATED WORKS

Information on a learner in Moocs generate complex event from a lot of sources (videos, forum interactions, clickstreams, assignments, search), which lead us to the complex event processing systems to deal with such events. As presented in our first paper a lot of work has been dons in this area.

First, C-SPARQL is a language of continuous queries over streams of RDF data. C-SPARQL is a continuity of SPARQL, developed to support physical and logical time calculations in stream using the Resource Description Framework data (semantic information).

Second, ETALIS is an open-source complex event processing system that can detect complex events in real time. It can perform reasoning over streaming events with respect to background knowledge. ETALIS implements two languages: ETALIS Language for Events, and Event Processing SPARQL.

Thirds, EP-SPARQL [8] is an extension of SPARQL relaying on CONSTRUCT and recursive queries, it uses the

978-1-5386-9493-0/18/$31.00 ©2018 IEEE

binary operators SEQ, EQUALS, OPTIONALSEQ, and EQUALSOPTIONAL used to combine graph patterns in streaming.

fourth, CQELS is Continuous Query Evaluation over Linked Streams. This engine focuses on continuous queries is a result of implementing caching technologies and heuristic tools for storing middle query results. CQELS demonstrate higher performance in terms of speed than C-SPARQL or ETALIS.

Last, INSTANS is an incremental engine for near-real-time processing of complex, layered, heterogeneous events. It focuses on continuous query. To process stream RDF INSTANS uses the Rete algorithm. As seen in all presented works functions of stream processing were added to SPARQL query process, systems are focusing on-stream RDF data. Moocs learner activities need more than that to be deduced because, besides to streaming, data generated is coming from many kinds of resources to be processed it needs reasoning component in streaming environment.

II. ONLINE LEARNING: FUNCTIONNAL SEMANTIC COMPLEX EVENT MODEL

In our first paper we have presented a functional complexes event processing system for learner activities in Moocs, our model adapts existing CEP functions to the case of Moocs learner Data which come from distinct types of sources, not only streaming.

In the Fig1. we present our Macro Model, the proposed system relay on a CEP engine that will be detailed in the next points.

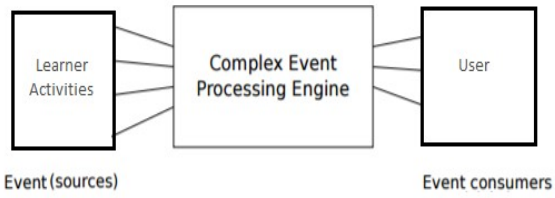


Fig. 1. Semantic Complex event processing macro model

Functional comlex event processing model :

To deal with real-time analysis our system relies on semantic CEP model respecting the following rules: (i) every new event is deduced and generated from relationship between detected events coming from various sources, and (ii) a complex event is generated by interactions between detected events. Our model relays on event sources based on a remote protocol, a CEP engine and a result as described in Fig1.

As presented in Fig2. The system proposed uses a CEP engine relaying on complex event processing system that is composed of four main engines and agents, we list below the main functions of each one:

a) Event Receiver: receives stream RDF events and forward them to the Event performer.

b) Rule engine: describe events using rules. The rules are written in XML format.

c) Event performer: performs the event in the way it was defined in the rule engine.

d) Event Forwarder: is generally connected to database or webserver, and it forwards the CEP event to user.

So, our model will be presented as follows

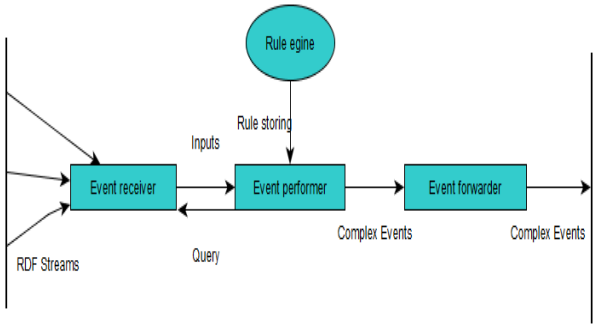


Fig. 2. Functional semantic complex event processing model for learner activities

The most important sections to describe are those related to data extraction, event operations and dataflows.

1) Data extraction:

In our system data will be extracted by using SPARQL query for event rule definition, and we do this using SPARQL tags and time domain tags in the event tags.

2) Event operations:

Similarly, to the data extraction phase, we will use tags for describing operations between events to generate the complex events. Event performer sends SPARQL and time domain query to the event receiver for managing queries. Event Forwarder sends the user the complex event, which can be connected to database driver or web server.

3) Data flows:

To deal with massive data steaming, the components of the proposed complex event processing model are distributed on a distributed environment. Therefore, in the proposed model flows are done in the following way: events are first detected as a RDF stream. Then, the Event performer catches relevant query data to confirm whether they are satisfied to the user's query or not, to made and forward the complex events.

III. ONLINE LEARNING: DISTRIBUTED EVENT PROCESSING SYSTEM FOR REAL TIME DATA ANALYSIS

To support a wide range of semantic data and queries processing there are many RDF Stream solutions such as CQLES, C-SPARQL, INSTANS, ETALIS, EPSPARQL that provide various operators to deal with the challenges of duplicated events detection and near real-time events aggregation. Some of them provide only classic data stream operators such as windows and sequences, and other offer a real-time scalable processing of complex events, linking then dynamic datasets.

Thus, one of the most crucial elements in the implementation of complex event processing system is scalability of architecture to ensure the required performance for Big Data Analytics. No matter how much hardware is raised (CPU or main memory) the real time processing is assured to exceed the amount computer's ability to calculate.

As the amounts of generated data can vary heavily it is necessary that the information system is scalable. But vertical scalability is limited by the resources that can be added to a machine, thus horizontal scalability – adding machines to a cluster is essential.

According to (Ellis, 2014) real-time analytics systems have exactly these three features:

- Low Latency
- Horizontal Scalability
- High Availability

The most of complex event processing engines are based on a vertically scalable architecture that consists of adding resources to the nodes of the cluster, typically involving the addition of processors (or cores), memory or disks. However, toward implementing of a complex event processing system that can process Big Data and support real-time analytics, the system should handle a horizontally scalable architecture that consist of adding more nodes to the cluster to allow scale-up of performance.

The framework Hadoop is focused on batch processing for big data analytics. In many cases such as indexing web this model still sufficient, but in our case processing real-time information from highly dynamic sources is required. Consequently, too main frameworks are generally used:

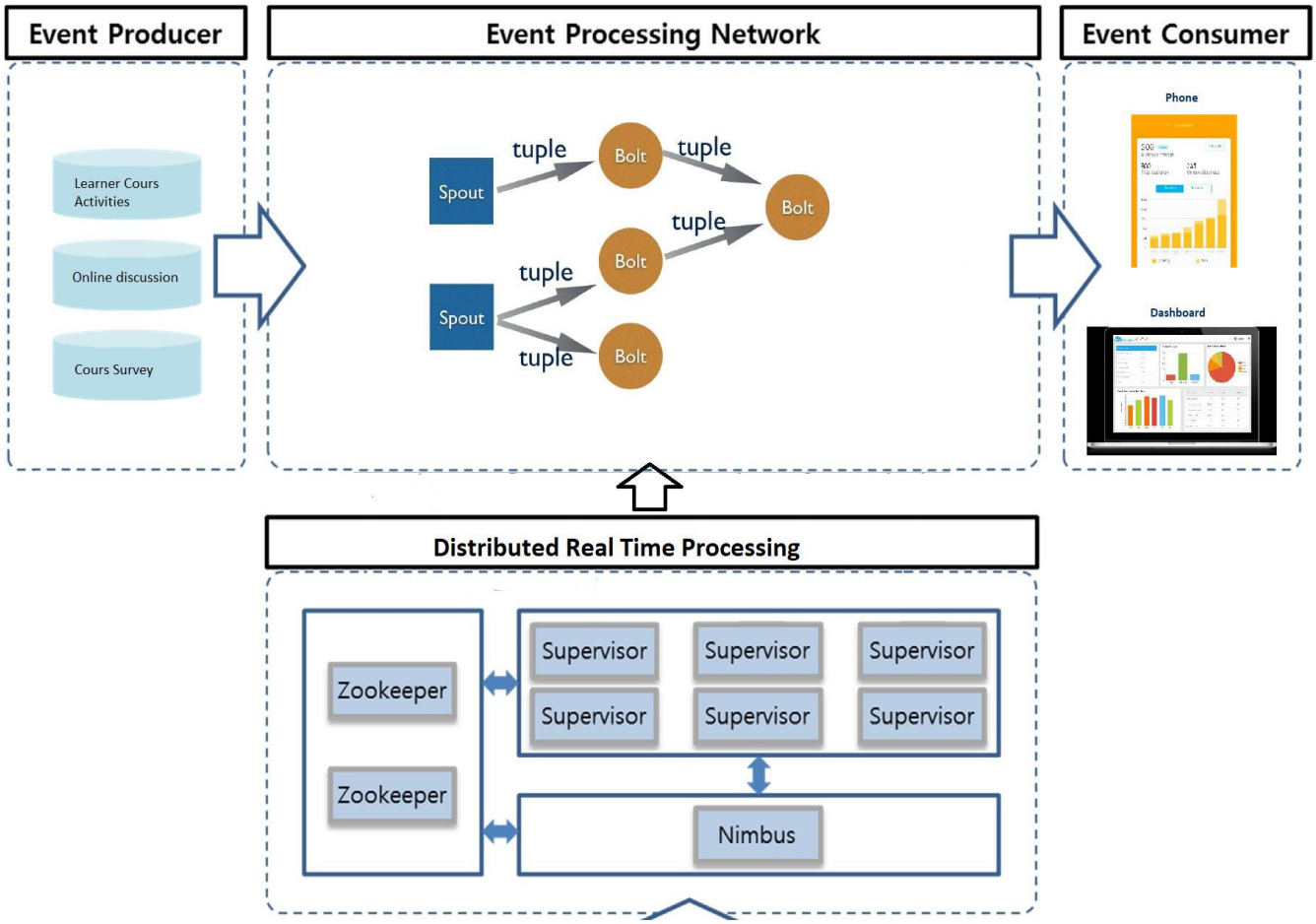
- S4[6] (Simple Scalable Streaming System) is a distributed stream processing engine inspired by the MapReduce model. S4 was designed to solve problems in the context of search applications that use data mining and machine learning algorithms.
- Storm [7] is commonly used for a wide range of applications, processing high volume data with low response times. In storm processing the huge amount of information in real-time is achieved by distributing the workload across multiple machines

The advantage of using frameworks such as S4 or Storm is that it allows implementing a distributed process characterized by stability while assigning the complexity of distributed/parallel processing and construction of a recovery mechanism to the framework. Even if S4 framework can be used for real-time event analytics, it potentially will lead to message loss due to its recovery mechanism, which relies on the check point technique to support fault-tolerance.
Then, using the proposed learner Moocs activities model, we have chosen Storm as the framework for Realtime analysis of the data. We will use storm framework to build a real-time, complex event processing architecture respecting the following conditions:

- Preventing message loss,
- Fault-tolerant,
- Horizontal scalability.

Storm is a distributed and parallel framework based on a master node called Nimbus, Zookeepers and workers. Nimbus is a daemon that runs on the master node of Storm cluster. It is responsible for distributing the code among the worker nodes, assigning input data sets to machines for processing and monitoring for failures. Supervisor manages worker processes to complete the tasks assigned by Nimbus. The topology in storm is contained of Spout and Bolt, with Spout read tuples off a messaging framework and emit them as stream of messages, and Bolt acting in a lot of roles such as a filtering, functions, aggregations, joins, talking to databases. Zookeeper monitors the working node status and helps the supervisor to interact with the nimbus. It is responsible to maintain the state of nimbus and supervisor.

In the aim of analyzing a learner activity, the distributed



events are processed as defined in Fig. 3. Under the event producer part event are produced by enrolling for a course, taking the course when it begins or after that in offline process, taking the assignment related to the course and being evaluated by the teacher or the system on knowledge, uploading the contents authorized and participating in an online discussion about the course or directly contact the teacher, or participating to a feedback surveys about some course are each handled by Spouts. After that bolts starts performing event-processing of data set coming from spots; each result is sent to the appropriate Bolt, which is responsible of complex event detection. Thus, each Spout and Bolt are allocated to a proper cluster node by the Storm framework.

IV. CONCLUSION

Producing, transmitting and continuously querying RDF Streams is the aim of semantic CEP engines. As such, they miss the functions required for reasoning activities coming from diverse sources with respect to background ontology. In this paper, a functional semantic CEP model for Moocs learner detection activities is proposed to overcome those issues.

The proposed model is still at the step of functional thinking. In this paper, we have defined tags and XML rule format using RDF stream processing, and we have proposed a CEP distributed architecture. Future work will complete this model by presenting the sample of data that can be used and test it on a real data environment, we will describe detail related to Stream RDF input, Rule registration, rule parsing, and storing, SPARQL query, Triple storage and management. Also, this model could be updated to and extended to other RDF stream processing such health-care or IoT systems.

REFERENCES

[1] Iria Estévez-Ayres, Jesús Arias Fisteus, Carlos Delgado-Kloos. Lostrego: A distributed stream-based infrastructure for the real-time gathering and analysis of heterogeneous educational data.Journal of Network and Computer Applications,2017

[2] Davide Francesco, Barbieri Daniele, Braga Stefano, Ceri Emanuele, Della Valle, Michael Grossniklaus. Querying RDF Streams with C-SPARQL, ACM SIGMOD Record archive Volume 39 Issue 1, March 2010, Pages 20-26

[3] Mikko Rinne, Esko Nuutila, and Seppo T¨orm¨a. INSTANS: High-Performance Event Processing with Standard RDF and SPARQL, Proceeding ISWC-PD'12 Proceedings of the 2012th International Conference on Posters & Demonstrations Track - Volume 914 Pages 101-104, November 11 - 15, 2012.

[4] Darko Anicic, Sebastian Rudolph, Paul Fodor, Nenad Stojanovic, Stream Reasoning and Complex Event Processing in ETALIS, The Journal of Systems and Software. Semantic Web 1(2009)1–5. IOS Press

[5] D. Anicic, P. Fodor, S. Rudolph, and N. Stojanovic, EPSPARQL: A unified language for event processing and stream reasoning, in: Proc. of the 20th International Conference on World Wide Web (WWW'11), S. Srinivasan, K. Ramamritham, A. Kumar, M.P. Ravindra, E. Bertino, and R. Kumar, eds, ACM, New York, USA, 2011, pp. 635–644.

[6] Leonardo Neumeyer, Bruce Robbins, Anish Nair, Anand Kesari. S4: Distributed Stream Computing Platform, 2010 IEEE International Conference on Data Mining Workshops

[7] A. Toshniwal, S.Taneja, A.Shukla, K.Ramasamy, J.M. Patel*, S.Kulkarni,J.Jackson, K.Gade, M.Fu, J.Donham, N.Bhagat, S.Mittal, D.Ryaboy. Storm @Twitter, Proceedings of the 2014 ACM SIGMOD International Conference on Management of Data Pages 147-156