Dynamic and Scalable Real-time Analytics in Logistics
Combining Apache Storm with Complex Event Processing for Enabling New Business Models in Logistics

Benjamin Gaunitz, Martin Roth and Bogdan Franczyk
Information Systems Institute, Leipzig University, Grimmaische Straße 12, Leipzig, Germany

Keywords: Real-time Analytics, Logistics, Complex Event Processing.

Abstract: In this paper we present an approach for an information system which is capable of processing and analysing vast amounts of data. In addition to Big Data solutions we do not focus on ex post batch processing but on online stream processing. We use Apache Storm in combination with Complex Event Processing to provide a scalable and dynamic event-driven information system, providing logistics businesses with relevant information in real-time to increase their data and process transparency.

1 INTRODUCTION

For logistics companies, especially for providers of courier, express and parcel services, it is essential to deliver goods and services in a highly time critical environment to satisfy customer needs. Those companies that are able to react as fast as possible to changing conditions will have benefits in the market competition by reducing costs and increasing efficiency. Modern logistics companies produce an enormous amount of data every day by tracking millions of shipments around the world as well as the fleet delivering these shipments. For example in 2013 2.66 billion shipments were handled by Courier, Express and Parcel services in Germany (BIEK, 2014). With the rise of the Internet of Things and the enhancement of sensor technology it is not only possible to track shipments location-wise but also monitoring sensor data such as temperature, orientation and acceleration resulting in an exponential growth of the amount of data which is produced by each tracked object. In addition to that, external sources such as traffic or weather services also generate business-relevant data which has to be processed.

The field of Big Data is well-known and has been widely covered in the recent years. However in the context of Big Data, data is analysed after a business process is finished. The results and insights are available ex post and in terms of logistics cannot be used to affect currently running transports.

(Hackathorn, 2003) introduces the term ‘action distance’ which describes the distance between the occurrence of a business relevant event and the moment when any action is taken. The reasons for this distance are diverse and range from a spatial to a social gap. As time passes, the value of the business event decreases, hence the smaller the action distance, the higher is the value of the business event and therefore the benefit for the company.

This results in the urge of minimising the latency between the information and the action and is summarised under the term Fast Data (Mishne et al., 2013). This advancement from Big Data to Fast Data creates a need for real-time analytics systems.

With real-time analytics logistics companies are able to react proactive to problems occurring during the transport. Therefore, logistics providers are not only able to answer question like “Where is my shipment?” but also “Where will my shipment be on a certain point of time?”.

The following are upcoming trends in logistics which require a broad usage of real-time analytics approaches in the future.

Real-time routing: Logistic chains consist of different steps. Usually the last step, the delivery to the recipient, is the most expensive for logistics provider. Increasing the efficiency on this last leg can reduce costs significantly. (Jeske et al., 2013) defines one scenario where delivery trucks are rerouted in real-time. The route of each truck is calculated constantly according to the delivery sequence and the

Gaunitz B., Roth M. and Franczyk B..
Dynamic and Scalable Real-time Analytics in Logistics - Combining Apache Storm with Complex Event Processing for Enabling New Business Models in Logistics.
DOI: 10.5220/0005467602890294
In *Proceedings of the 10th International Conference on Evaluation of Novel Approaches to Software Engineering* (ENASE-2015), pages 289-294
ISBN: 978-989-758-100-7

current traffic situation. The goal of this approach is to decrease idle time and therefore increase the utilization of the fleet and the delivery quality. Also customer satisfaction can be improved by providing arrival times and increasing on-time delivery. In order to enable real-time rerouting an appropriate IT system has to be implemented which is capable of handling data from multiple input sources such as traffic services, ERP systems (delivery sequence) and fleet management systems and execute complex calculations (e.g. Travelling Salesman Problem) on this data with extremely low latency.

Synchromodality: Synchromodality is an advancement of multi-, inter- and co-modality with the objective to use any available form of transportation at any time and point in the logistics chain and also switch freely between them (van Stijn et al., 2013). The ideal transport mode can be selected based on a set of defined criteria. These could be for example transport cost, transport speed, load factor or, with the rising importance of green logistics, carbon dioxide emissions. To enable synchromodality seamlessly and successfully within the logistics chain it is necessary to improve data transparency between all participants. This includes detailed information about the transported freight as well as about the load factor on any transport vehicle which is and can be used potentially.

Fourth-party Logistics Provider: In fourth-party logistics (4PL) the logistics provider (4PLP) does not have any assets and does not fulfil any transport at all. The 4PLP acts as a coordinator for several logistics provider (Nissen and Bothe, 2002). As many partners are used during the transport process it is essential that in a case of delivery quality failure the relevant transport partner is identified and conceivably the transport process is reorganised to ensure a reliable delivery, regarding time and quality. In this scenario each shipment is tracked concerning their geo location and certain sensor values, such as ambient temperature, acceleration and orientation. Before the transport is started, a service level agreement is created which defines the quality of the transport and timeslots for each leg. The continuous real-time tracking of shipments creates the possibility for logistics companies to react to nonconformities instantaneously.

(Ellis, 2014) defines five components of real-time architectures: Collection, Data Flow, Processing, Storage and Delivery. As all of those are essential, we will focus on the processing of data.

This paper presents an approach for a distributed information system which will be capable to analyse and process these vast amounts of data in real-time, combining Apache Storm and Complex Event Processing.

The paper is structured as follows: Section 2 describes the requirements of a real-time system architecture regarding the above mentioned trends. Related research which has been conducted in the area of real-time analytics and logistics is presented in Section 3. In Section 4 we present and explain the technical foundation which we used for our idea. The real-time analytics system we designed is described in Section 5.

2 REQUIREMENTS ANALYSIS

After the emergence of Big Data in recent years the next-level approach is summarised under the term Fast Data. It is an evolution from batch to stream processing. While for batch processing the analysed data is definite, for stream processing the analysed data is infinite. This creates the need for new technologies and techniques to analyse this stream of data.

The upcoming trends in the last section show the importance of a small action distance otherwise these trends will not be realisable. Real-time routing for example where the dispatcher receives the information after the transport has finished is not valuable. This results in a need for low latency to process the information quickly to retain its value.

As logistics business operates 24/7 all around the world and low latency is essential for high information value a highly stable information system is needed, otherwise the latency will increase and the information value decrease.

Additionally in all three future trends a large amount of data is produced. In real-time routing it is the data of the delivery fleet, for synchromodality it is the data about the utilisation of each usable means of transport and in 4PLP it is even each shipment which creates data. 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

Another requirement is data transparency to create process transparency. Especially in 4PL and

synchromodal logistics each business partner needs to introduce information systems to increase the data transparency of its operations, e.g. utilisation data being available system wide. The barrier for integration has to be low as small and medium enterprises are essential for successful 4PL and synchromodal logistics.

3 RELATED WORK

The combination of real-time analytics and logistics and the benefits and opportunities resulting of this combination have not been researched widely yet.

(Jeske et al., 2013) presents DHL's view on Big Data. Current and future scenarios in logistics are presented partly relying on real-time analytics such as real-time routing (as described in Section 1) and crowd-based pick-up and delivery, where any person can deliver shipments on the go. Many other use cases for logistics are presented as well which can create value-added services for logistics businesses. The publication offers a business view; technical implementations or considerations are not presented.

In (Toshniwal et al., 2014), published by Twitter employees, the usage of Apache Storm by Twitter is outlined. Twitter runs "several hundreds of topologies" on "hundreds of servers (spread across multiple data centres)" producing "several" billions events and terabytes of data per day. Apache Storm is used for processing the content in various streams and also for machine learning algorithms. They especially highlight the stable behaviour of Storm during runtime, the low latency (~1ms) and the high availability (99.9%). On the other hand it is stated that a dynamic re-optimisation of the topologies is one of the issues considered for future research which we also identified as a disadvantage (see Section 5). Our approach includes a proposal for bypassing this disadvantage by using Complex Event Processing in addition to Apache Storm (see Section 5). Despite the paper is not about the domain of logistics it is a good overview of the capabilities of real-time analytics and the technical foundations of Apache Storm.

(Isoyama et al., 2012) has examined technical issues that emerge from Distributed Complex Event Processing in clusters which we describe in Section 5.2. They also propose a possible implementation to solve these issues, though a complete solution is not presented. The result still lacks the possibility of full scalability of Complex Event Processing.

(Schwegmann et al., 2013) use real-time analytics in combination with event-driven Business Activity Monitoring. The paper is about the usage of Complex Event Processing with a Business Process Management System to calculate KPIs in real-time which are used to make predictions about future process behaviour by applying statistical evaluations to these KPIs. While the same assumption of minimising the action distance to increase the business value is taken as basis, the main focus is on creating future predictions which we do not regard in this paper

4 TECHNICAL FOUNDATIONS

Businesses are event-driven. They have to react to unpredictable external and internal events (Bruns and Dunkel, 2010). Our approach is embedded in an Event-Driven Architecture (EDA) to process these business relevant events. In an EDA the components themselves are event-driven. This means that communication between them is done with the usage of events, small data objects containing information. The business data is transmitted continuously to the system in form of an infinite event stream. The event stream is the foundation of an EDA.

The two major components that we use in our system are Apache Storm and Complex Event Processing which we will explain in the following section.

4.1 Apache Storm

Apache Storm was developed by BackType, which was later acquired by Twitter, to analyse the users' click behaviour in real-time and soon became an Apache Top-Level project (Ellis, 2014). Apache Storm is scalable out of the box and can easily be set up on clusters or cloud services like Amazon's EC2.

Basically Apache Storm works like Hadoop. With Hadoop it is possible to analyse large amounts of data within a distributed cluster. Each node within the cluster analyses a small set of the data, the results of all nodes are being merged after the analysis process (Map-Reduce algorithm) (Dean and Ghemawat, 2008). Just as Hadoop, Apache Storm is horizontally scalable and able to process large amounts of data in a distributed cluster. But unlike Hadoop, which is designed for analysing data ex post in fixed sized batches, Apache Storm is event-driven and analyses and processes data in form of an (infinite) event stream in real-time thus making results available immediately.

The basic components of Apache Storm are spouts and bolts. Spouts are the sources of data

which is emitted in form of tuples. Tuples and events can be used interchangeably in this context. For example these tuples can be data from traffic, weather, location or sensor tracking services. Bolts are the components which process and evaluate the tuples. If needed, bolts can emit their calculated results for further evaluation to other bolts.

These two components are combined in so called topologies. Topologies are directed-acyclic graphs which define the flow of events from the sources (spouts) to the sink passing one or multiple nodes (bolts). The foundation of each topology is a cluster of physical or virtual machines. Depending on the configuration and the work load Storm will create multiple instances of each bolt and distribute them across the cluster to ensure a fast event processing.

It is possible to add and remove machines to the Apache Storm cluster to easily adapt to changing event flow rates. Also crashed machines will not be a problem as bolts are replicated throughout the cluster and the event flow is rerouted to the existing machines. For further technical details see (Ellis, 2014; Toshniwal *et al.*, 2014; Apache Storm, 2014)
There are also two major disadvantages of Apache Storm regarding the above defined requirements. The first one is that topologies cannot be changed once set up and running. The current workaround is to shutdown a topology and restart an updated version. As this workaround can last up to a few minutes, it will result in losing events and / or increasing latency significantly.
The other disadvantage is the programmatical evaluation of events within the bolts i.e. using conditional expressions. This can result in code that is hard to read, understand and manage and since it is hard-coded it is conceivably one of the reasons a dynamic update of the evaluation is not possible within the bolts.

4.2 Complex Event Processing

CEP is the main component for event processing in an Event Driven Architecture (Bruns and Dunkel, 2010). According to (Luckham, 2002) it is a software technology which is used to analyse causal, temporal and spatial relationships between events in real-time. So in CEP events are not evaluated separately but rather in their context. (Bruns and Dunkel, 2010) defines three essential elements of CEP components: event model, event rules and the event engine, respectively in the terms of (Luckham *et al.*, 2011): event schema, rules and event processing agent.

The event model defines the events which contain information that is business relevant e.g. a sensor event containing the current ambient temperature. The event model is not further regarded in this paper.CEP components use rules to evaluate the events and their relations to each other. In most CEP components the event rules are defined declarative with the usage of a Domain Specific Language, the Event Processing Language (EPL). This EPL is used to create SQL-like statements containing the events defined in the event model and the conditions they should apply to. The third element is the CEP engine. In combination with the event rules the CEP engine analyses the event stream in real-time for patterns defined in the EPL. CEP engines are able to analyse event streams from multiple sources and can handle millions of events per second depending on their hardware and configuration. Examples for CEP components are *Esper*, *Drools*, *Microsoft StreamInsight* and *TIBCO BusinessEvents* & *Streambase*.

Current CEP solutions are not scalable in a full extent. While vertical scaling – adding resources to a machine – is easily possible, limitations exist for horizontal scaling – adding machines to a cluster (see Section 5.2). As vertical scaling is limited by the resources (CPU, memory, HDD) that can be managed by one machine, the maximum amount of ICA8Rm9udFN0eWxlPg0KICAgICAgICA8TmV1d as well, so support for horizontal scaling is required.

4.3 Benefits of Integrating Complex Event Processing in Apache Storm

The combination of Apache Storm and CEP results in the following benefits. Apache Storm is a good basis for distributed computing as it is horizontally scalable by default and fault-tolerable. With the usage of a CEP engine we are able to bypass the disadvantages and additionally use the horizontal scaling feature provided by Storm which is currently not possible with CEP *engines* (see Section 5.2).

As stated above CEP engines include the ability to register rules which are used to evaluate incoming events. In contrary to Apache Storm these rules, which are comparable to the processing algorithm within the bolts, are expressed using EPL statements. The rules within the CEP engine can be changed dynamically and are immediately applied to all events which are received after the update. Therefore giving us the ability of reacting to changing circumstances or focuses of interest in terms of the business relevant data while being flexible by adding and removing resources depending on the workload.

5 APPROACH

This section presents and evaluates our approach for an information system being able to process and analyse vast amounts of data in real-time with being extendable and changeable at any time during operations.

5.1 Scenario

The foundation of our system is an Apache Storm topology with multiple internal and external data sources, which could be shipments, means of transport or traffic services (see Figure 1). The amount of data emitted by these sources is the event stream thus containing the business relevant data. The spouts (Spout 1 to Spout i) are the connectors to the event stream and emit the data in form of tuples to the topology. The spouts are the starting point of the event flow within our Apache Storm topology. The events in form of tuples are then passed to the bolt in tier 1. The *events* are evaluated according to the EPL statements. Either the results are directly forwarded to the connected systems or are being passed to next tier for further evaluation. It is recommended to use general rules in the lower tiers and get more specific the higher the bolt is inside the topology. This will ensure that uninteresting events are already filtered in the beginning and reduce the load on the more complex statements. The architecture is comparable to an array of sieves for filtering sand where the holes in the sieve become smaller the further the sand flows through the array. However we are not interested what is left in the sieves but rather what flows through.

As already stated, in Apache Storm a bolt is a template for handling events. Storm will create several instances of one bolt according to the load and configuration parameters. In our model each instance of a bolt creates and starts a CEP engine. Events will only flow to one instance of a bolt so we need to assure that each CEP instance in one tier has exact the same statements implemented the holes in our sieve. To do so, we need another component, the statement repository. The statement repository contains all the rules for one tier / bolt which will be used to evaluate the incoming events. Whenever a new instance of a bolt is created the CEP engine, which is created with this bolt, will pull all the current statements from the repository and vice versa subscribe itself for updates to the repository. This subscription enables us to pass new or changed statements to all CEP engines immediately and in order with that realising the requirement of fast adaption to changing environments as mentioned in the beginning of this paper. This feature giving executives, users and analysts to change rules on the fly and let them, the domain experts, decide what is interesting.

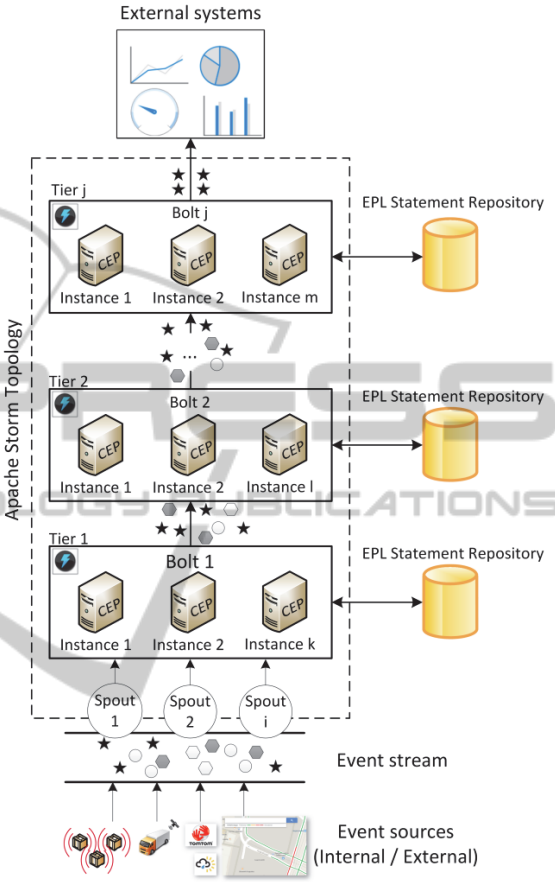

Figure 1: Apache Storm in Combination with Complex Event Processing.

5.2 Evaluation and Further Research

Currently the analysis of Complex Event Processing in a distributed environment has a major drawback. We have to distinguish between two different EPL rule categories: stateless and stateful. Stateless rules need information only from the current event which is evaluated. No information about preceding events is necessary to process the rule. E.g. "every EventA(temperature > 30)". Contrarily stateful rules need information about other events. E.g. "Event A followed by Event B". This category includes sliding windows, patterns and abundance detection of events. While it is not a problem to process stateless rules in a distributed system, the processing of stateful rules is complicated. Each event is evaluated by

exactly one of multiple CEP engines within each tier. Each CEP engine has its own context which is not visible to the other instances. Stateful rules need in contrary to stateless rules, additional context information. To successfully evaluate events, the context has to be shared between all event processors. (Isoyama et al., 2012) proposes a solution to adjust the distribution of events throughout the cluster. This is not a full scale solution of the problem of context sharing within distributed CEP engine. The most useful approach for our scenario would be the implementation of an external shared context state which according to (Isoyama et al., 2012) will result in high overloads and performance issues when the separate CEP engine instances will access the shared context state. Further research in this area has still to be conducted.

6 CONCLUSIONS

In this paper we presented an approach for an information system being capable of analysing vast amounts of data in real-time and thus enabling businesses to react immediately by reducing the action distance. The system we presented is completely based on open source software. Because of missing licence fees it is possible for small and medium logistics enterprises to implement and use real-time analytics for their operational work to offer value-added services such as real-time tracking or SLA monitoring and therefore increasing the business profit and improve customer satisfaction.

ACKNOWLEDGEMENTS

The work presented in this paper was funded by the German Federal Ministry of Education and Research under the project LogiLeit (BMBF 03IPT504A).

REFERENCES

Apache Storm (2014), “storm”, available at: https://storm.apache.org/ (accessed 16 January 2015).

BIEK (2014), *Anzahl der Sendungen von Kurier-, Express- und Paketdiensten (KEP) in Deutschland in den Jahren 2000 bis 2013 (in Millionen), Statista - Das Statistik-Portal*.

Bruns, R. and Dunkel, J. (2010), *Event-Driven Architecture*, Springer, Berlin, Heidelberg.

Dean, J. and Ghemawat, S. (2008), “MapReduce. Simplified Data Processing on Large Clusters”, *Communications of the ACM*, Vol. 51 No. 1, pp. 107–113.

Ellis, B. (2014), *Real-Time Analytics,* 1st ed., Wiley, Indianapolis.

Hackathorn, R. (2003), “Minimizing Action Distance”, available at: http://www.tdan.com/view-articles/5132/ (accessed 19 January 2015).

Isoyama, K., Kobayashi, Y., Sato, T., Kida, K., Yoshida, M. and Tagato, H. (2012), “A scalable complex event processing system and evaluations of its performance”, in *Proceedings of the 6th ACM International Conference on Distributed Event-Based Systems*, ACM, New York, NY, pp. 123–126.

Jeske, M., Grüner, M. and Weiß, F. (2013), *Big Data in Logistics*, Troisdorf, Germany.

Luckham, D., Schulte, R., Adkins, J., Bizarro, P., Jacobsen, H.-A., Mavashev, A., Michelson, B.M. and Niblett, P. (2011), *Event Processing Glossary: Version 2.0*.

Luckham, D.C. (2002), *The Power of Events*, Addison-Wesley, Boston.

Mishne, G., Dalton, J., Li, Z., Sharma, A. and Lin, J. (2013), “Fast data in the era of big data”, in Ross, K. and Data, ACM Special Interest Group on Management of (Eds.), *Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data*, ACM, New York, NY, USA, pp. 1147–1158.

Nissen, V. and Bothe, M. (2002), “Fourth Party Logistics – ein Überblick”, *Logistik Management*, Vol. 4 No. 1, pp. 16–26.

Schwegmann, B., Matzner, M. and Janiesch, C. (2013), “A Method and Tool for Predictive Event-Driven Process Analytics”, in Alt, R. and Franczyk, B. (Eds.), *Proceedings of the 11th International Conference on Wirtschaftsinformatik, Leipzig*, Univ., Leipzig.

Toshniwal, A., Donham, J., Bhagat, N., Mittal, S., Ryaboy, D., Taneja, S., Shukla, A., Ramasamy, K., Patel, J.M., Kulkarni, S., Jackson, J., Gade, K. and Fu, M. (2014), “Storm@twitter”, in *SIGMOD '14: Proceedings of the 2014 ACM SIGMOD International Conference on Management of Data,* ACM, New York, NY, USA, pp. 147–156.

van Stijn, E., Hesketh, D., Tan, Y.-H., Klievink, B., Overbeek, S., Heijmann, F., Pikart, M. and Buterly, T. (2013), “The Data Pipeline”, in United Nations Economic Commission for Europe (Ed.), *Connecting International Trade: Single Windows and Supply Chains in the Next Decade*, United Nations, New York and Geneva, pp. 158–183.