Project Acronym:	**LeanBigData**
Project Title:	**Ultra-Scalable and Ultra-Efficient Integrated and Visual Big Data Analytics**
Project Number:	**619606**
Instrument:	**STREP**
Call Identifier:	**ICT-2013-11**

D9.12 Insight to LeanBigData

Work Package:	*WP9 - Exploitation, Industrial Awareness, Dissemination*
Due Date:	31/01/2016
Submission Date:	31/01/2016
Start Date of Project:	01/02/2015
Duration of Project:	36 Months
Organisation Responsible for Deliverable:	UPM
Version:	1.0
Status:	Final
Author(s):	Ricardo Jimenez (ed) LeanXcale All partners
Reviewer(s):	
Nature:	☒ R – Report ☐ P – Prototype ☐ D – Demonstrator ☐ O – Other
Dissemination level:	☒ PU - Public ☐ CO - Confidential, only for members of the consortium (including the Commission) ☐ RE - Restricted to a group specified by the consortium (including the Commission Services)
Project co-funded by the European Commission within the Seventh Framework Programme (2007-2013)	

Revision history

Version	Date	Modified by	Comments
0.1	10/01/2016	Ricardo Jimenez (LeanXcale)	First revision D9.11
0.2	17/01/2016	Ricardo Jimenez (LeanXcale)	First round of contributions
0.3	29/01/2016	Ricardo Jimenez (LeanXcale)	Second round of contributions
1.0	31/01/2016	Ricardo Jimenez (LeanXcale)	Final version

Table of Contents

Index of Figures

Abbreviations and acronyms

tbd	To be defined
SOA	Service Oriented Architecture
CEP	Complex Event Processing
OLTP	On Line Transactional Processing
OLAP	On Line Analytical Processing

1. Introduction

1.1. Purpose of the document

This document aims at providing the vision, objectives and progress of the project for a technical audience interested in the project outcome.

1.2. Intended audience

The target readers are the general public with technical background and interested in following up the project outcome.

2. LeanBigData Motivation, Vision and Goals

LeanBigData aims at addressing three open challenges in big data analytics:

1. The cost, in terms of resources, of scaling big data for streaming and static data sources;
2. The lack of integration of existing big data management technologies and their high response time;
3. The insufficient end-user support leading to extremely lengthy big data analysis cycles.

Over the last years there has been a lot of progress on the scalability of big data. Google, Facebook, and Amazon already process massive amounts of data. However, the techniques used for processing these large amounts of data are extremely inefficient, consuming a tremendous amount of resources resulting in a very high total cost of ownership (TCO). The amount of resources used to process data is becoming an important concern due to the fact that public cloud data centres are becoming one of the biggest consumers of energy; 2% of the electricity produced in the US is consumed nowadays by cloud data centres. World-wide data centres consume about 1.3% of the electricity produced.

Integrating different technologies over the same set of data requires a large effort, it is ad-hoc, and increases development cost for analytics. Multiple technologies are integrated mostly via an extraction-transform-load process that results in reading and writing the whole database in a periodic manner, typically daily. This widespread approach has two important problems: It affects the QoS of the production database and it is extremely costly. In some sense, although scalable, big data analytics tend to operate mostly in batch mode resulting in poor support for business processes: For instance, the Google web search engine used map-reduce jobs to process the results of crawling the web to generate a new web index from scratch. This resulted in delays of weeks from the crawling a URL until it appeared as a result in a web search. Today, Google uses a transactional index that is continuously updated and the delay has been reduced to minutes. Big data analytics still suffer from such issues in most domains.

Finally, the end-user of big data analytics is facing today long cycles of data analysis: Long cycle to discover relevant facts in data (problems, issues, alarms, etc.) that require fast reaction; Long cycle (hours or days) to get the results of large analytical queries; Long cycle to visualize the result of ad-hoc queries due to requires programmatic effort; Long cycle to interact with the visualizations until they serve the final business process.

LeanBigData is addressing these challenges by:

- Architecting and developing three **resource-efficient** Big Data management systems typically involved in Big Data processing: a novel transactional NoSQL key-value data store, a distributed complex event processing (CEP) system, and a distributed SQL query engine. The efficiency of these systems is one of the main innovations of the project. To achieve this we will remove main overheads at all levels: in the data managers (garbage collection, multi-versioning, multi-threading contention, networking, management of shared resources), in the use of underlying hardware (memory hierarchies and NUMA architectures, multi-cores, storage subsystem), in the operating system and virtualization layer, and by taking into account emerging storage technologies and trends in non-volatile memories.
- Providing **an integrated big data platform** with these three main technologies used for big data, NoSQL, SQL, and Streaming/CEP that will improve response time for unified analytics over multiple sources of data avoiding the inefficiencies and delays introduced by existing ETL-type approaches. To achieve this we will use fine-grain intra-query and intra-operator parallelism that will lead to sub-second response times for queries over static and streaming big data.

- Supporting an **end-to-end big data analytics solution** enhancing the lifecycle of data analytics by: 1) automated discovery of anomalies and root cause analysis that will provide end-users with a starting point at time 0; 2) Supporting data scientists to manipulate the result set of analytical queries in an agile way by means of a visual and interactive interface to discover insights by enabling an easy declarative manipulation of the results sets.

LeanBigData delivers a Big Data platform that is ultra-efficient, improving today’s best effort systems by at least one order of magnitude in efficiency, reducing the amount resources required to process a set of data or allowing us to process more data with the same amount of resources as today. LeanBigData will scale efficiently to 10,000s of cores. Finally, LeanBigData will demonstrate these results in a cluster with 1,000+ cores in four real industrial use cases with real data, paving the way for deployment in the context of realistic business processes.

LeanXcale is a spinoff of the project that has already been incorporated to exploit commercially some of the core technology developed in the project.

3. Overview and Progress Beyond the State of the Art

3.1. Global Architecture

Figure 1 presents the holistic view of the LeanBigData platform. The data management subsystems are depicted on the bottom part. The key value data store sits on top of the storage layer (bottom right). The storage layer aims at providing a highly efficient IO path to store data persistently. The key value data store acts as storage engine. The storage layer is seen as local storage by the nodes, but it provides a distributed shared storage system with high availability provisions. The key value data store allows reading and writing data as key-value pairs at very high rates. Persistency is achieved by storing data in the storage layer. The key-value data store is fully ACID, that is, it is transactional. Transactional support is depicted on the bottom left.

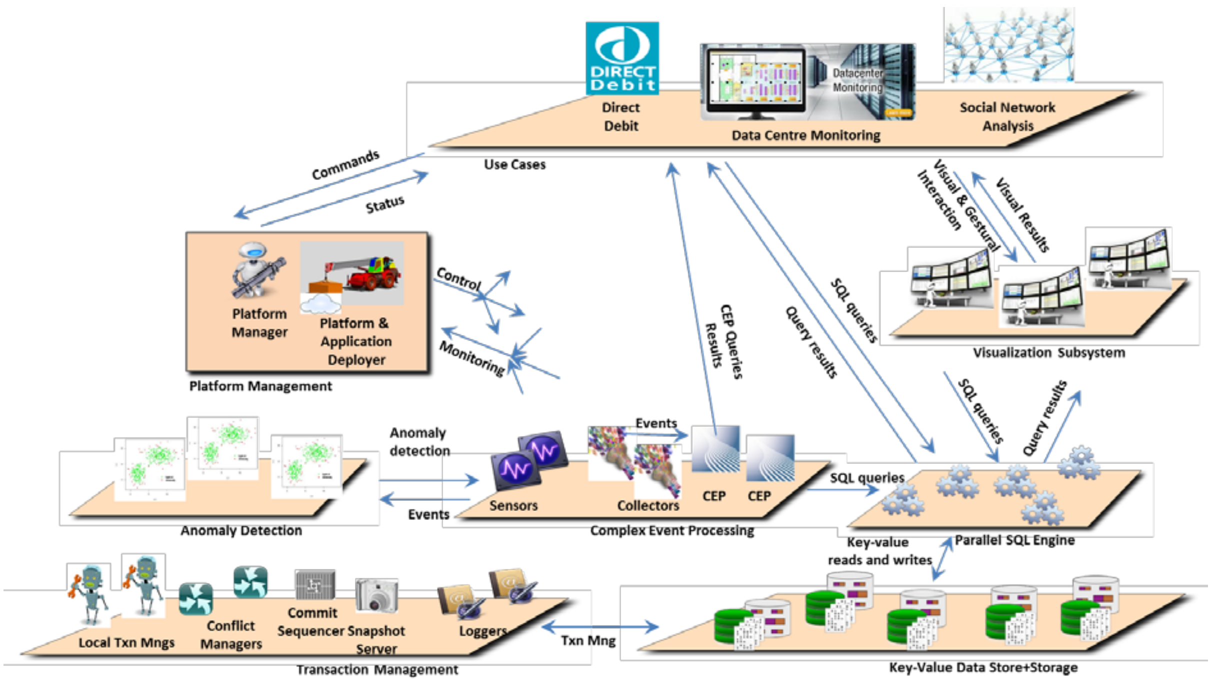


Figure 1: Holistic View of LeanBigData Platform

The transactional support decomposes the ACID properties and scales each of them separately, but in a composable way. The transactional management is integrated with the key-value data store, therefore providing a full ACID key-value data store. The integration is performed in a highly decoupled way. The key-value data store provides multi-versioning capabilities. It enables the transactional layer to determine from which snapshot data should be read and with which timestamp updated tuples should be labelled. All the transactional logic is encapsulated as a wrapper of the key-value data store client proxy.

The transactional support scales the different ACID properties (Atomicity, Consistency, Isolation, Durability) independently but in a composable way. Atomicity is provided by local transaction managers. Local transactional managers are collocated with the key-value data store clients.

The SQL processing layer provides both OLTP and OLAP capabilities over the operational data. That is, it enables to perform updates over the data and analytical queries over the current version of the operational data. The SQL layer is totally stateless and only stores session data. It is also parallel, that is, a query can be run by all SQL engine instances to share the work across instances and be able to reduce the latency of heavy analytical queries. The SQL engine can also be used as an OLTP engine with many different instances to be able to scale OLTP query processing.

The complex event processing (CEP) subsystem enables to deploy and process continuous queries that process streams of events. The CEP subsystem can interact with the stored data via SQL to either read or write data. Data collectors are used to interface between the CEP subsystem and systems to extract data from.

The visualization subsystem provides a visual workbench for data scientist to perform data analysis. Analytical queries can be submitted against the parallel SQL engine and the result sets then can be manipulated through a visual and gestural interface to transform them as needed to extract insights and visualize them.

The platform manager enables to deploy the platform and monitor it while running, as well as reconfiguring it when needed.

The use cases validate the LeanXcale platform to stress it with real-world problems. The first use case deals with large scale data centre monitoring. The second use case focuses on e-advertising. The third use case addresses social network analysis applied to sentiment analysis around events or meetings in a city. The fourth use case aims at finding fraud in direct debit operations in the banking area.

3.2. Storage

Over the last few years there have been dramatic technological changes in the I/O path, mainly driven by NAND-FLASH-based storage devices that are becoming commonplace. However, current technology projections are that we are approaching another disruptive change to the I/O hierarchy with the use of emerging, byte addressable Non-volatile Memories (NVM). There is a number of other technologies [1] that are candidates for bridging the gap between non-persistent byte-addressable DRAM and persistent, block-addressable storage, with STT-RAM, PCM, and FeRAM being currently the most likely candidates to bridge this gap. Such technologies will not only dramatically reduce I/O latency and increase I/O throughput, but will also require fundamental architectural changes to the storage subsystem: Storage (and NVM) will need to be placed closer to compute servers to take advantage of lower latencies, the boundary with non-persistent DRAM will have to be re-evaluated in a fundamental manner, and the software stack will need to adapt to new capabilities and limitations to a larger extend compared to NAND-FLASH SSDs [2,3,4,5].

With the advent of big-data, storage architectures and the storage stack in general are undergoing fundamental changes to keep up with demand and requirements of new applications and services. At the storage system level problem faced today by infrastructures can be categorized in three broad areas:

1. Storage acceleration
2. Storage convergence
3. Storage isolation

Storage acceleration is the area that has received over the past few years a tremendous amount of attention, mainly due to the advent of FLASH-based storage device technologies. Research and industry have strived over the past years to integrate these new technologies in the I/O path and build faster and more efficient storage systems that results at lower total cost of ownership (TCO). This has required tremendous work at the device, systems, and event application level and despite the progress that has been achieved, a number of issues still remain open, making this a fertile area for research.

However, using technology projections, we believe that new persistent storage technologies will appear in the future that has fundamentally different properties both from disks and NAND-FLASH-based SSDs. These technologies are collectively called non-volatile memories (NVM) and they are high-density, low-latency, persistent memories, with however, different types of characteristics and access methods. Projections indicate that, similar to SSDs, these technologies will need to be integrated in the I/O path creating a deep storage hierarchy from application memory, all the way to slow but large capacity disk drives. Based on our past experience from SSDs, such technologies will not be straight-forward to integrate in the I/O stack, since they fundamentally change the way storage is accessed, e.g. via variable size units or event at user-space without the need to go through the operating system kernel.

Storage convergence refers to bringing storage and computation closer both in the data center and HPC. Traditionally, persistent storage has been placed behind a storage-area-network (SAN) for scaling and management purposes. SANs help manage large amounts of storage in a “centralized” manner, resulting in improved performance and reduced TCO for scalable storage, despite the high cost of the SAN itself. With current technology trends for it becomes important that storage moves closer to the compute nodes. Although these tradeoffs are generally complex in a real system, consider a platform that uses for storage an NVM device with access latency below 1 usec. It does not make sense to place such a device behind a SAN, when the network latency itself is an order of magnitude higher. In addition, storage convergence is projected to provide better locality and significant benefits for increased reliability.

However, realizing storage convergence is a significant architectural shift for storage systems. It requires fundamentally different approaches to storing, caching, replicating, and moving data around. For instance, caching storage in local write back cache is not safe, as a failure can result in data loss and corruption.

Storage isolation is emerging as one of the central problems for storage infrastructures of the future. Recent work and results show that shared access to storage and contention that is induced event when independent applications access the same storage devices (but not the same data) results in loss of efficiency at the device and I/O path level. As a consequence providers and infrastructure manager systems prefer to reduce shared access to storage resources by over-provisioning resulting in increased costs. Problems related to storage interference will intensify in the future given the mandated shift towards consolidated systems. Local storage devices will need to be shared by both local and remote applications for storing and accessing data, which will result at increased interference and loss of performance determinism at multiple levels: devices, network, and software I/O stack.

LeanBigData focuses on the first of these two issues, storage acceleration and storage convergence, and designs a shared storage substrate that will pave the way for satisfying the needs of future applications.

At the data storage and access layer, LeanBigData takes a new approach. Typical systems today rely on three main layers for the required functionality:

- A local file system that provides space allocation and recovery.
- A distributed filesystem, usually HDFS, that provides a global namespace for files and data replication. This file-system also allows efficient sequential accesses to data.
- A key-value store that provides lookup operations over unordered data, typically in the form of (key,value) pairs.

These layers result in inefficiencies, mainly high CPU and memory use, which eventually hinders data processing capacity significantly. LeanBigData proposes a more efficient alternative that offers the same functionality in a leaner manner:

- A shared storage layer, directly ontop of the physical devices offers a global namespace, space allocation, replication, and recovery.

- A key-value store that runs directly on top of this shared storage layer, manages (key-value) pairs allowing both efficient sequential and lookup operations.

To realize this vision, the shared storage substrate of LeanBigData includes several innovations:

- Allows data to be placed anywhere on the system, without requiring indirection steps, after data is placed.
- Offers near-native performance for local data and optimizes performance of accesses to remote data.
- Offers fast replicated writes, without additional round-trips.
- Scales performance with all resources: cores, storage devices, network links.
- Provides elasticity in terms of both devices and servers at a fine granularity.
- Provides per volume configurable fault tolerance and deals with partitions in a simple but practical manner.

References

[1] Burr, G.W.; Kurdi, B.N.; Scott, J.C.; Lam, C.H.; Gopalakrishnan, K.; Shenoy, R.S., "Overview of candidate device technologies for storage-class memory," *IBM Journal of Research and Development*, vol.52, no.4.5, pp.449,464, July 2008. doi: 10.1147/rd.524.0449

[2] *Why is SSS (solid state storage) Crucial to the Data Center?* Jim Handy, Analyst, Objective Analysis. Keynote presentation at 2014 Storage Industry Summit (NVM). San Jose, CA. 28 January 2014. Available online at http://www.snia.org/nvmsummit

[3] S. Swanson and A. M. Caulfield, “Refactor, Reduce, Recycle: Restructuring the I/O Stack for the Future of Storage,” Computer, vol. 46, no. 8, pp. 52– 59, 2013.

[4] J. Ousterhout, P. Agrawal, D. Erickson, C. Kozyrakis, J. Leverich, D. Mazi`eres, S. Mitra, A. Narayanan, G. Parulkar, M. Rosenblum, S. M. Rumble, E. Stratmann, and R. Stutsman, “The case for ramclouds: Scal- able high-performance storage entirely in dram,” SIGOPS Operating Sys- tems Review, vol. 43, no. 4, Jan. 2010.

S. Peter, J. Li, I. Zhang, D. R. K. Ports, T. Anderson, A. Krishnamurthy, M. Zbikowski, and D. Woos, “Towards high-performance application-level storage management,” in Proceedings of the 6th USENIX Workshop on Hot Topics in Storage and File Systems, 2014.

3.3. Key Value Data Store

The LeanBigData platform uses as storage engine for the data a novel key-value data store. Current key-value data store technologies are still based on traditional server architectures that do not deliver high levels of efficiency, because they had their roots in a world without multiple CPUs and core, without NUMA memory, and without modern kinds of storage technologies. The goal in LeanBigData is to design a brand new key-value data store technology conceived to exploit efficiently the current hardware and storage architecture.

The new key-value data store aims at reducing the main costs and shortcomings that current key-value data stores have:

- Lack of transactional consistency.
- Overhead of distribution.

- Inefficient multi-versioning support.
- Cost of serialization.
- Cost of garbage collection.
- Cost of IO.

Another important innovation pursued by the new key-value data store is how to attain non-intrusive elasticity. That is, how to bear continuous reconfiguration due to dynamic load balancing and elasticity decisions without affecting the peak performance of the data store while is being reconfigured.

3.4. Ultra-Scalable Transactions

One of the main core features of the LeanBigData is its transactional support. The transactional support guarantees data consistency in both the advent of failures and concurrent accesses. Transactions are a critical feature of data stores. Firstly, without transactions, when there is a failure, the database becomes inconsistent. For instance, if a client is performing a transfer of funds between two accounts, and the money is withdrawn from the origin account but there is a failure before depositing the money in the second account the database has lost its consistency. Obviously this is not admissible for the operational database of any organization. Secondly, without transaction, when there are concurrent accesses the data can lose its coherence. A recent example has been some BitCoin companies that were based on a non-transaction NoSQL data store and millions of dollars were stolen, not due to a security hole, but through the inconsistencies that a non-transactional data store allows. This resulted in being able to withdraw from an account the amount it contained many times in parallel before the balance of the account was updated. This was done massively in parallel enabling to steal millions of dollars.

LeanBigData is fully transactional avoiding these serious problems that non-transactional data stores, such as NoSQL data stores, pose. More technically, it fully provides the so-called ACID properties. These properties are Atomicity, Consistency, Isolation and Durability. Atomicity guarantees that the database accesses within a transaction are all-or-nothing in the advent of failures. In the above examples it guarantees that a bank transfer either is fully performed or if there is a failure is like the bank transfer has not been done at all. Consistency is the property guaranteed by the applications, basically, they have to guarantee that any transaction that takes as initial state a consistent database after the transaction completed with success, the database is in a final consistent state. That is, the logic of applications is correct. Isolation is the property that guarantees the data consistency in the advent of concurrent accesses. It has to guarantee that the final result is equivalent to a serial execution of the transactions. Finally, Durability guarantees that the updates from a transaction completed with success (i.e. committed) are never lost even in the advent of failures.

However, nobody knew how to scale-out transactional processing. This has resulted in modern approaches such as NoSQL data stores to dismiss transactions in favour of being able to scale out. In LeanBigData one of the main innovations is that we are able to scale-out transactions to very large scales (100s of nodes). So unlike any existing data management platform LeanBigData is able to guarantee full data consistency and be able to scale to large levels. This has been achieved by a brand new approach to scale transactional processing. This approach is based on decomposing the ACID properties to minimal features that can be scaled out independently, but in a composable manner.

Additionally, LeanBigData is interested not only in scaling out transactional processing, but being able to scale it efficiently. That is, in reducing the number of nodes required to process a given workload. In order to scale efficiently a number of sources of overhead are being dealt with:

- Context changes across thread.
- Inter-thread synchronization.
- Java garbage collection overhead.
- The cost of distribution.
- The cost of multi-versioning.
- The cost of IO.

Current results show that before introducing optimizations to deal with the above sources of overheads, the system is able to scale linearly with a linear resource usage. The system has been benchmarked with the TPC-C benchmark workload that is the most exigent in terms of realistic update transactional workloads.

3.5. Real-Time Big Data

Analytical systems aim at answer business oriented questions, such as: “What is the total sales amount for year 2014?” Real-time analytics means first the answer is obtained fast, from an interactive system. A multi-dimensional model produces faster answers when compared with the normal approach in transactional systems, as dimensional modelling promotes data aggregation. But real-time analytics means also that results have to be obtained from fresh data, ideally, the one being produced by OLTP.

In detail, OLAP systems need to perform three types of operations, namely: consolidation, drill-down and slicing & dicing. The consolidation operations require data to be aggregated. This is made possible through selecting the right granularity regarding the necessary dimensions within the data warehouse in order to compute a measure. Usually this sort of data aggregations allows collecting an overview and then navigating through the details. Picking up on the previous sales example, the business question could be tailored to only regard the sales respecting a given product. The drill-down operation would change the granularity of a possible "product" dimension so that the result would be at the level of a single product. Finally, the slice & dice operation allows navigating through the dimensional model and relate several dimensions, extracting only relevant sub-sets of data from the dimensions.

OLAP systems are usually categorized according to the following taxonomy:

- ROLAP: These systems allow the execution of analytical processing over relational storage. The relational database in such systems, besides holding tables for the desired dimensions also adds further tables that enable querying aggregated data. The main advantage of ROLAP systems is that they do not limit possible business questions, as no restrictions on the dimensions of the OLAP cube are imposed. However, as queries are answered straight out of the operational database, the transactional operation may suffer delays and throughput penalties.
- MOLAP: This definition stands for Multi-dimensional OLAP that replaces the relational database with a multi-dimensional storage. This requires the OLAP cube to be pre-computed. MOLAP systems present great performance for query execution that is due to the indexing and caching capabilities in multi-dimensional systems. However, MOLAP also has major negative points due to the need for data processing through the use of the ETL process, but also due to difficulties related with the ability to efficiently perform queries over dimensions with high cardinality.

The relational approach has the advantage, in principle, of allowing OLTP and OLAP operations to be executed in the same query engine. Currently, there are two main trends in the path to scale analytics on relational data.

The first follows the Map-Reduce [1] approach with improvements to bring it closer to SQL semantics. The standard SQL Map-Reduce approach introduced by Hadoop [2] is a programming model based in two functions: the *map* and *reduce* functions. With the *map* function, users generate an intermediate set of key/value pairs from the data set, while the *reduce* function merges all the intermediate values associated with each key. The Map-Reduce approach is targeted to big data sets and currently offers several advantages over the use of parallel database systems in what regards to storage system independence and elastic adaptation to variable resources and demands. For a comprehensive survey of the key differences between Map-Reduce and parallel databases, the interested reader in kindly referred to [3]. Projects like BigQuery [4], Tenzing [5] or Hive [6] provide a SQL interface over a Map-Reduce framework and a scalable key-value store. BigQuery allows only a subset of SQL operators that allow basic data aggregation and projection. Hive maps operators like equi-joins or unions to Map-Reduce jobs. Tenzing also relies on Map-Reduce to provide a query language closer to SQL. Such projects are OLAP oriented and as a consequence require ETL procedures that may generate duplicated data.

The second trend is based on leveraging parallel-distributed databases that fully support SQL semantics, thus making it more expressive when compared with the former Map-Reduce approach for normalized data sets. In particular, to perform data aggregation, these systems need to access the same tabular structure as the transactional system, but in a different pattern. A single aggregation primitive (for example a sum) needs to go over all the occurrences (rows) of a given attribute (column) and compute the aggregated value. For this reason, if the analytical system were to use the same row-oriented strategy as the transactional system, that operation would generate a significant larger number of calls to the I/O system. Employing a columnar layout of data allows the occurrences of a single attribute to be stored in consecutive data blocks.

By using such approach, it is expected a reduction of the amount of I/O calls to roughly a third when compared with performing columnar aggregations over row-oriented storage. Moreover, using a columnar store also simplifies type encoding, as all the entire data stream shares the same data type. The advantages just presented can also be transported to operations that are only executed in-memory, as analytical queries usually only read subset of columns on a table. Therefore, solutions as MonetDB [7], Vertica [8], or GreenPlum [9] are strict OLAP oriented projects, lacking the ability to perform OLTP workloads and relying on the ETL approach.

The LeanBigData parallel-distributed SQL engine integrates transactional processing support, to allow processing of typical workloads in a production database (OLTP workload), with analytical queries typical of a data warehouse/big data solution (OLAP workload). It is based on an existing centralized engine (Apache Derby) and its extension to the Derby Distributed Query Engine (DQE), thus the current effort has focused on analytical workloads and on designing and implementing appropriate extensions for parallel distributed processing.

The general approach of the LeanBigData parallel SQL query engine is to provide intra-query parallel processing with symmetric workers in the Derby Distributed Query Engine (DQE). To support scalable analytical processing, its design focuses on aggregation operations over very large data sets. First, it adds SQL standard support for window aggregation to Derby DQE. Second, it focus on scale-out by offering intra-operator parallelism that can be the most effective precisely in aggregation operations. It is thus a hybrid approach to OLAP, combining a relational system with scalable partitioning and shuffling similar to Map-Reduce that can be used interactively.

Preliminary performance evaluation of the current prototype, that includes a query optimizer that transforms sequential query plans for parallel-distributed execution, as well as a communication middleware based on Remote Direct Memory Access using the InfiniBand *Verbs* interface, has shown that scale-out can in fact be achieved by linearly improving the execution time for queries such as those included in the TPC-H industry standard benchmark.

3.6. Elastic Complex Event Processing

LeanBigData also brings a scalable and elastic complex event processing engine (CEP) integrated with the platform. The incorporation of the CEP engine enables to correlate in real-time massive event/data streams with the operational data stored in the LeanBigData platform. This is especially interesting in the case of Internet of Things (IoT) and Machine-to-Machine (M2M) applications.

In LeanBigData the main focus to advance scalable CEP technology with respect the state of the art has been on increasing its efficiency in scaling and on providing non-intrusive elasticity. The CEP aims at scaling with the number of deployed queries, the complexity of the deployed queries and the stream volumes to be managed by the deployed queries.

3.7. Anomaly Detection

In the context of the LeanBigData project, input data in the different case studies are streams. Streams are processed through the CEP and analysed searching for potential anomalies. The module for Anomaly Detection inside the platform is a logical unit inside the CEP, containing the CEP operators used for Anomaly Detection and Root-cause Analysis. *Figure 2* shows the architecture of the platform where the CEP and the Anomaly Detection module have been marked.

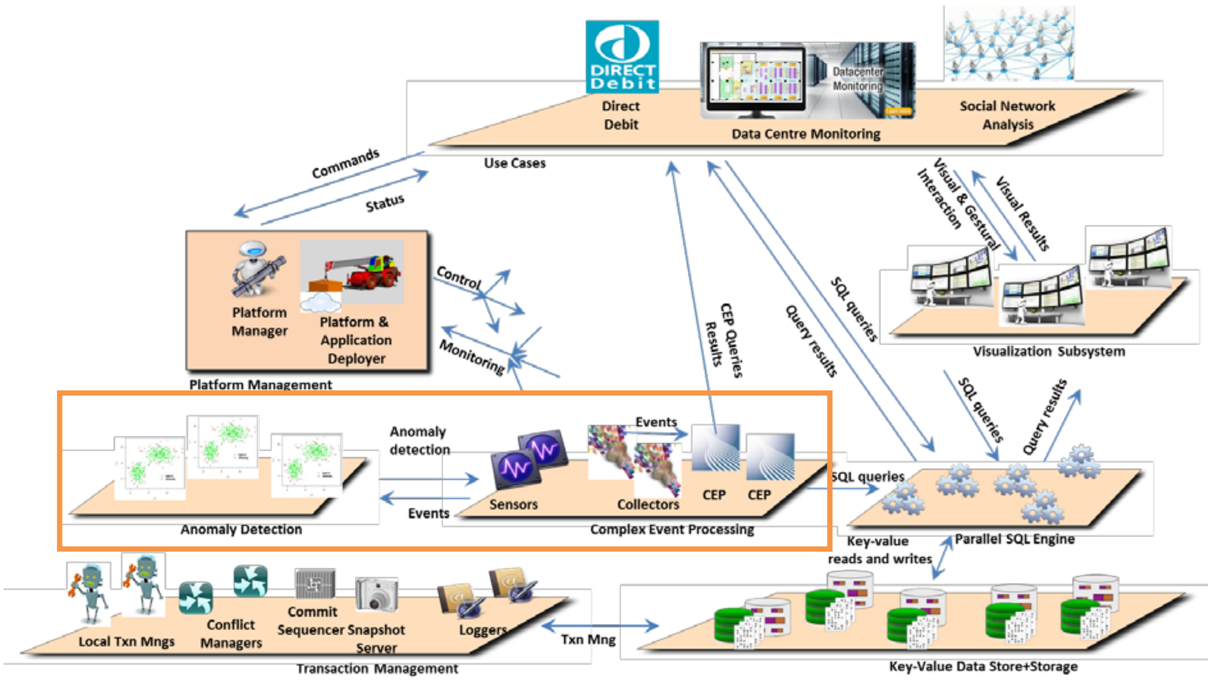


Figure 2 – Architecture of the LeanBigData platform where the location of the anomaly detection and root-cause analysis operators is indicated by an orange box.

The Anomaly Detection and Root-cause Analysis operators that have been implemented inside this module respond to the following requirements by the case studies:

- **Need for detecting discrepancies with respect to a model created a priori**: this is the case for instance in the Data Centre case study, led by CA Technologies, in which different models are used to generate predictions of expected values for temperature and energy consumption for the servers in the datacentre. The anomaly detector implemented in this use case compares actual data with predictions to monitor the validity of the model and is able to cope with different sources of noise.

For the Banking case study, led by SyncLab, the model built was obtained by applying novel process mining techniques to the debit transactions and discrepancies were scored to improve the fraud detection capabilities of the classifier under development in this case study.

- **Need for detecting bursty words in tweets**, for the Social Network case study, led by ATOS. The implementation of this operator consists of a parallelization of an online bursty word detection algorithm that can handle large amounts of data.
- **Need for generating the data that is necessary to recalibrate the model**: this is particularly necessary in the Data Centre case study. Anomalies, which by definition are short-lived, are ignored for this purpose, but durable changes raise an alarm indicating that the model is no longer valid and the system generates the data necessary to recalibrate the model.
- **Need for generating hypotheses of the root causes that may have generated a particular anomaly**: In the Data Centre case study a change in the temperature in a server may be caused by a change in another server that transfers heat to the first one. The implementation for the RCA operator takes the output of the anomaly detectors and uses a hierarchical decomposition of the diagnosing procedure to meet the challenges of diagnosing a very large system in a small amount of time, and the fact that anomaly detectors act as noisy sensors.

This is necessary in the three case studies. In the Data Centre case study a change in the temperature in a server may be caused by a change in another server that transfers heat to the first one. These types of hypothesis should be generated by the system for future RCA. In the Social Network case study, the most frequent words related to a change in a particular word frequency may contain conceptual information that provides hints about the cause of that change (or anomaly). Root-cause analysis will be implemented with rules defined on top of graphs, which is a novel application of these structures in this area.

Based on this observation, three novel generic CEP operators will be created that can be summarized as it follows:

- **Model Discrepancy Detector** (MDD): operator to detect anomalies comparing actual data with different types of models.
- **Recalibration Model Data Generator** (RMDG): operator to generate new data to update a model that is obsolete, after a durable change in the characteristics of input data.
- **Graph-Based Root Cause Analysis** (GBRCA): operator to provide hypotheses to help users on a generic RCA process. This operator will be based on the representation of a system through a graph

3.8. Visual Analytics Workbench

The workbench deals with the higher layers of the LeanBigData platform. It supports end users in their decisions by providing automated data pre-processing, visualization of query results, and exploration of these visualizations.

A number of Key R& D goals has been achieved:

1. Querying and post-processing through configurable workflow

A data processing workflow is a configurable acyclic directed graph where nodes represent either data sources or data operators. These are linked by edges to define the process sequence. This approach is commonly used in advanced analytics tools but less common in traditional BI. We aim to provide a workflow that includes a subset of SQL operators to simplify the query process as well as additional data processing and reporting operators. A user may click on any node and execute the process up to that point to get a preview of the overall function. Outputs of the query can be downloaded or saved to a data base. Outputs of one query may be re-used in another query.

2. Minimal data exchange over network

Current approaches to data processing workflows assume that data will be loaded onto a local hard-drive for processing and/or visualization. The scale of big data means that data transfer is prohibitive, which requires model building to be moved to the data. This has implications for both GUI and data processing activities. From a GUI perspective, data representations need to be based on meta-data, summary statistics/or and sub-samples. From a data processing perspective, workflows will be transformed into queries that will be sent to the backend of the LBD platform via JDBC for processing

3. Browser-Based, touch & gesture friendly GUI

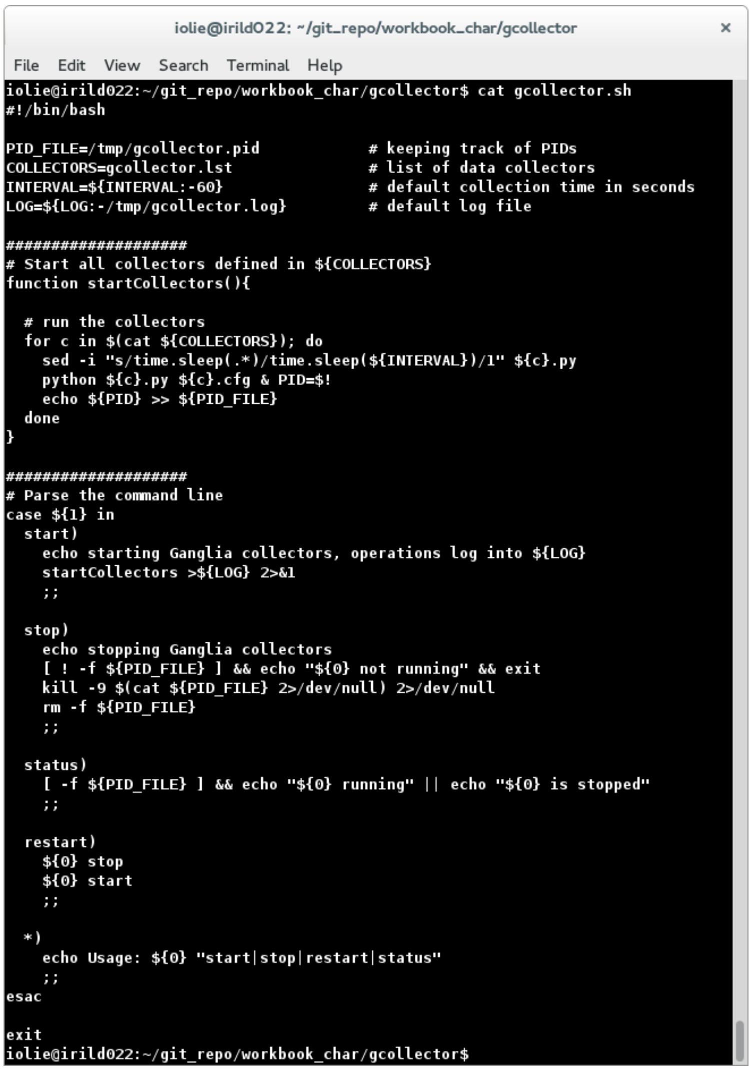
Current advanced analytics workflow GUIs have been designed for native desktop applications and assume a WIMP (Windows, Icon, Mouse, Pointer) mode of interaction. We aim to develop a browser-based workflow constructor that is compatible with touchscreen and gestural interactions.

The visualization system will consist of client and server-based components. Client side components will be consist of responsive web-based GUI’s and Visualizations to support query building and chart configuration and a gesture recognition component across multiple devices. Server side components will manage query workflow execution, metadata management and storage of query results and statistics.

The system will use novel human-computer interaction techniques that will support visualisation manipulation on multi-touch and gestural displays. Novel approaches to the visualization of data structures will be explored. This work will design visual and gestural semantics that will enable an end user to generate queries, filter data sets, apply different visualisation types, navigate large scale visualisation and integrate/aggregate results. In addition to SQL queries the visual interface will support the deployment of data analysis operators such as clustering.

3.9. Data sensors

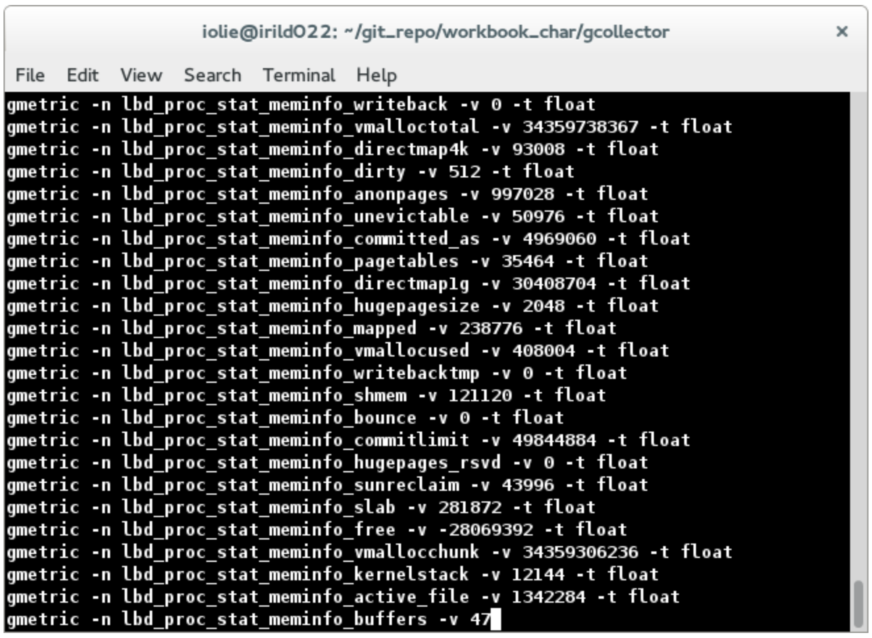
A range of data sensors were developed for the platform manager and use cases. Evaluations of C++, Python and Bash implementations shown the flexibility to add sensors in LBD. Data sensors use an open framework in which individual modules can be plugged into the platform, providing multiple dispatching mechanisms. The instrumentation and monitoring has been implemented as a sensor based service that can be easily instantiated and managed by external programs or services, such as the scripts that manage the execution of experiments. The agent implements the typical management functions of a service such as start, stop, restart and configuration related methods, see *Figure 3*. The valid operations are self-explanatory, for instance “gcollector.sh start” or “gcollector.sh stop”.


iolie@irild022: ~/git_repo/workbook_char/gcollector
File Edit View Search Terminal Help

```
iolie@irild022:~/git_repo/workbook_char/gcollector$ cat gcollector.sh
#!/bin/bash

PID_FILE=/tmp/gcollector.pid            # keeping track of PIDs
COLLECTORS=gcollector.lst               # list of data collectors
INTERVAL=${INTERVAL:-60}                # default collection time in seconds
LOG=${LOG:-/tmp/gcollector.log}         # default log file

####################
# Start all collectors defined in ${COLLECTORS}
function startCollectors(){

  # run the collectors
  for c in $(cat ${COLLECTORS}); do
    sed -i "s/time.sleep(.*)/time.sleep(${INTERVAL})/1" ${c}.py
    python ${c}.py ${c}.cfg & PID=$!
    echo ${PID} >> ${PID_FILE}
  done
}

####################
# Parse the command line
case ${1} in
  start)
    echo starting Ganglia collectors, operations log into ${LOG}
    startCollectors >${LOG} 2>&1
    ;;

  stop)
    echo stopping Ganglia collectors
    [ ! -f ${PID_FILE} ] && echo "${0} not running" && exit
    kill -9 $(cat ${PID_FILE} 2>/dev/null) 2>/dev/null
    rm -f ${PID_FILE}
    ;;

  status)
    [ -f ${PID_FILE} ] && echo "${0} running" || echo "${0} is stopped"
    ;;

  restart)
    ${0} stop
    ${0} start
    ;;

  *)
    echo Usage: ${0} "start|stop|restart|status"
    ;;
esac

exit
iolie@irild022:~/git_repo/workbook_char/gcollector$
```

Figure 3. GCollector, a shell script in Bash that initialises the Ganglia data sensors.

Upon starting the data collection, the formatted output is dispatched to the defined backend, eg the Ganglia (UDP) or LBD (TCP) backend. We can inspect the log file for debugging reasons or to repeat all sampled metrics (redo action) and environment. *Figure 4* illustrates a typical output of the log option. Each line is self-contained and calls the Ganglia dispatcher, *gmetric*, with the metric namespace and associated values. Lines starting with “gmetric” can be filtered out to rerun the operations.


iolie@irild022: ~/git_repo/workbook_char/gcollector
File Edit View Search Terminal Help

```
gmetric -n lbd_proc_stat_meminfo_writeback -v 0 -t float
gmetric -n lbd_proc_stat_meminfo_vmalloctotal -v 34359738367 -t float
gmetric -n lbd_proc_stat_meminfo_directmap4k -v 93008 -t float
gmetric -n lbd_proc_stat_meminfo_dirty -v 512 -t float
gmetric -n lbd_proc_stat_meminfo_anonpages -v 997028 -t float
gmetric -n lbd_proc_stat_meminfo_unevictable -v 50976 -t float
gmetric -n lbd_proc_stat_meminfo_committed_as -v 4969060 -t float
gmetric -n lbd_proc_stat_meminfo_pagetables -v 35464 -t float
gmetric -n lbd_proc_stat_meminfo_directmap1g -v 30408704 -t float
gmetric -n lbd_proc_stat_meminfo_hugepagesize -v 2048 -t float
gmetric -n lbd_proc_stat_meminfo_mapped -v 238776 -t float
gmetric -n lbd_proc_stat_meminfo_vmallocused -v 408004 -t float
gmetric -n lbd_proc_stat_meminfo_writebacktmp -v 0 -t float
gmetric -n lbd_proc_stat_meminfo_shmem -v 121120 -t float
gmetric -n lbd_proc_stat_meminfo_bounce -v 0 -t float
gmetric -n lbd_proc_stat_meminfo_commitlimit -v 49844884 -t float
gmetric -n lbd_proc_stat_meminfo_hugepages_rsvd -v 0 -t float
gmetric -n lbd_proc_stat_meminfo_sunreclaim -v 43996 -t float
gmetric -n lbd_proc_stat_meminfo_slab -v 281872 -t float
gmetric -n lbd_proc_stat_meminfo_free -v -28069392 -t float
gmetric -n lbd_proc_stat_meminfo_vmallocchunk -v 34359306236 -t float
gmetric -n lbd_proc_stat_meminfo_kernelstack -v 12144 -t float
gmetric -n lbd_proc_stat_meminfo_active_file -v 1342284 -t float
gmetric -n lbd_proc_stat_meminfo_buffers -v 47
```

Figure 4. Sample output of data sensors dispatching to Ganglia.

The data collector agent can be seen as a self-contained Ganglia sensor; and is responsible for the agent's individual instantiation, execution and runtime management. The current implementation of the restart command will stop and destroy all the instances of all loaded data sensors; will restart the agent configuration; restart the configured modules, instantiate and execute them again.

3.10. Platform manager

When sensors dispatch data to the Ganglia backend, the operators should be able to distinguish the source or origin of metrics. It creates problems for large, distributed systems. The data sensors, together with the platform manager have an end-to-end solution for the gathering, dispatching and visualisation of metrics. Data sensors follow a hierarchical namespace, already seen in *Figure 4*. Metrics have a hierarchical order given by,

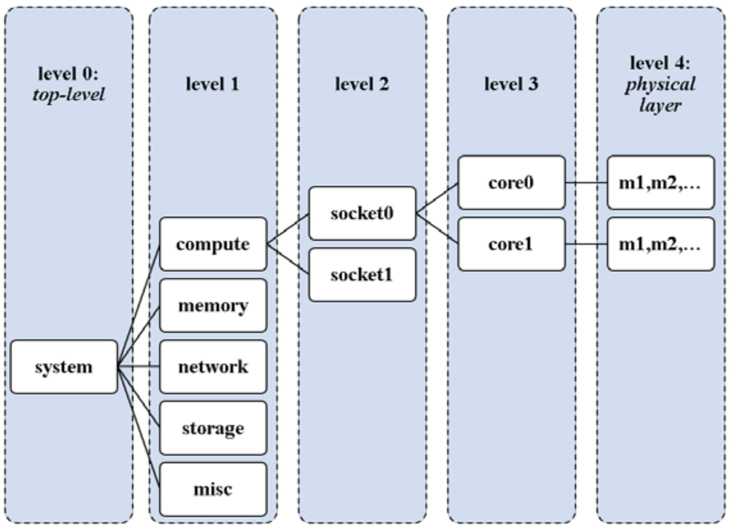


Figure 5. Hierarchical namespace associated to data sensors.

Raw data receives a timestamp before dispatching the payload to the data sink, and it is expected to return a data map containing metrics and their values. For instance, the prefix “proc.net” refers to “/proc/net” in Linux. There is however the inclusion of the prefix “LBD” to facilitate the access to LBD related metrics. For instance, when the platform manager (Ganglia) opens, the very first window (shown in Figure 6) contains the summary across multiple timescales:

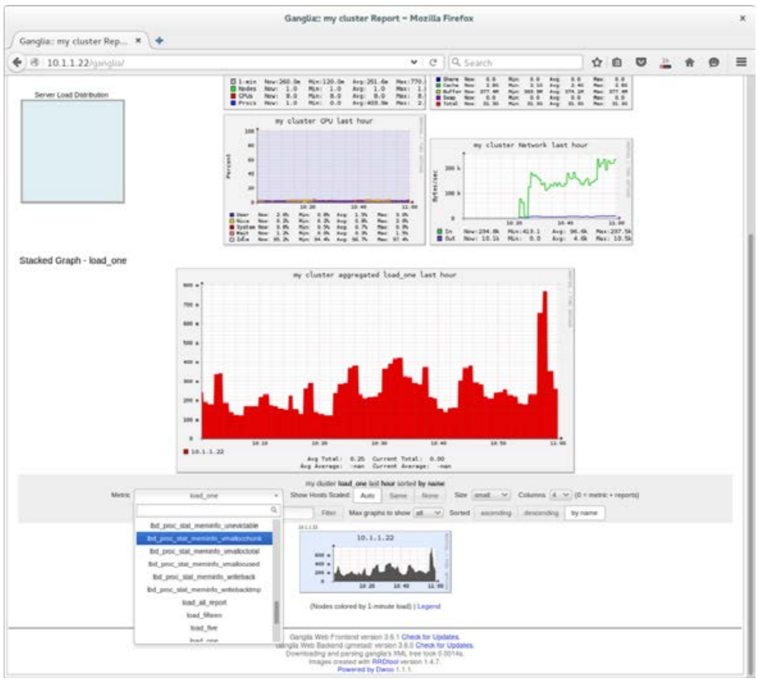
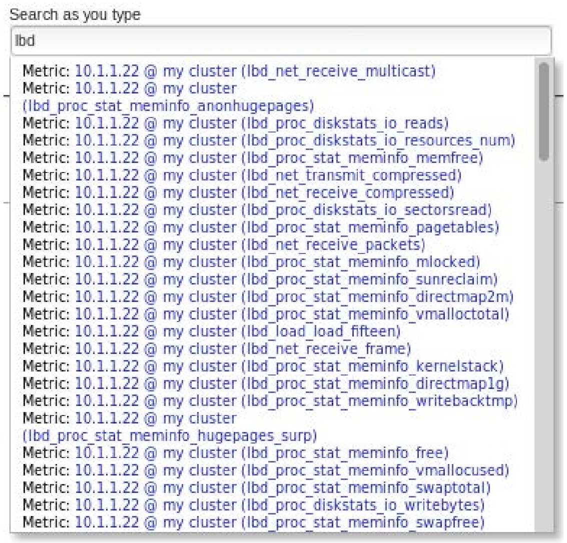

Ganglia:: my cluster Report - Mozilla Firefox
Server Load Distribution
my cluster CPU last hour
my cluster Network last hour
Stacked Graph - load_one
my cluster aggregated load_one last hour
10.1.1.22

Figure 6. Ganglia

Ganglia uses RRD to present summaries over long periods of time (see Figure 6). The menu allows users to display data for the last: hour, 2hr, 4hr, day, week, month and year. While the frontend should show something similar to Figure 6, and LBD metrics are accessible via a search box, as shown in Figure 7. Research and development efforts around the LBD Platform Manager were centred on *manageability*, *extensibility*, *scalability*, and *analytics* aspects. After evaluating a range of commercial and open-source monitoring solutions we found the current state-of-the-art has to be extended if one covers all of these goals. Ganglia, however, was identified as being the closest match to the purposes of LBD, as a base point. It required customisation to achieve all key requirements in the data sensors, storage and backend.



Search as you type

```
lbd
Metric: 10.1.1.22 @ my cluster (lbd_net_receive_multicast)
Metric: 10.1.1.22 @ my cluster
(lbd_proc_stat_meminfo_anonhugepages)
Metric: 10.1.1.22 @ my cluster (lbd_proc_diskstats_io_reads)
Metric: 10.1.1.22 @ my cluster (lbd_proc_diskstats_io_resources_num)
Metric: 10.1.1.22 @ my cluster (lbd_proc_stat_meminfo_memfree)
Metric: 10.1.1.22 @ my cluster (lbd_net_transmit_compressed)
Metric: 10.1.1.22 @ my cluster (lbd_net_receive_compressed)
Metric: 10.1.1.22 @ my cluster (lbd_proc_diskstats_io_sectorsread)
Metric: 10.1.1.22 @ my cluster (lbd_proc_stat_meminfo_pagetables)
Metric: 10.1.1.22 @ my cluster (lbd_net_receive_packets)
Metric: 10.1.1.22 @ my cluster (lbd_proc_stat_meminfo_mlocked)
Metric: 10.1.1.22 @ my cluster (lbd_proc_stat_meminfo_sunreclaim)
Metric: 10.1.1.22 @ my cluster (lbd_proc_stat_meminfo_directmap2m)
Metric: 10.1.1.22 @ my cluster (lbd_proc_stat_meminfo_vmalloctotal)
Metric: 10.1.1.22 @ my cluster (lbd_load_load_fifteen)
Metric: 10.1.1.22 @ my cluster (lbd_net_receive_frame)
Metric: 10.1.1.22 @ my cluster (lbd_proc_stat_meminfo_kernelstack)
Metric: 10.1.1.22 @ my cluster (lbd_proc_stat_meminfo_directmap1g)
Metric: 10.1.1.22 @ my cluster (lbd_proc_stat_meminfo_writebacktmp)
Metric: 10.1.1.22 @ my cluster
(lbd_proc_stat_meminfo_hugepages_surp)
Metric: 10.1.1.22 @ my cluster (lbd_proc_stat_meminfo_free)
Metric: 10.1.1.22 @ my cluster (lbd_proc_stat_meminfo_vmallocused)
Metric: 10.1.1.22 @ my cluster (lbd_proc_stat_meminfo_swaptotal)
Metric: 10.1.1.22 @ my cluster (lbd_proc_diskstats_io_writebytes)
Metric: 10.1.1.22 @ my cluster (lbd_proc_stat_meminfo_swapfree)
```

Figure 7. LDB Data Sensors

By typing the prefix “LBD” in the Ganglia frontend, a list of all data sensors developed for LBD is displayed (see Figure 7). Ganglia comes with a rudimentary set of data sensors out of the box, which have been extended. New data sensors have been implemented and are now being integrated with the visualisation subsystem. The dispatcher “gmetric” is the main component that data sensors employ. The Ganglia backend basically reads and stores the metrics in RRD format, which is the one that keeps summaries across days, weeks and months.

4. Industrial Use Cases

4.1. Cloud Data Centre Monitoring

Modern IT management systems employ variety of models for monitoring and managing large IT infrastructures. These models range from relatively simple event-condition-action rules to sophisticated simulation and multivariate regression. Models may also represent the relationship between different elements in the system. These models reflect normal, or expected, behaviour of managed environment. They are employed for detecting anomalies or changes, when behaviour of managed environment departs significantly from the expected behaviour given by a model, or for prediction in order to anticipate behaviour of the managed environment under changing load conditions and over time. The combination of graph models with other types of models is considered to be a powerful tool for root cause analysis.

Often these models are created using domain knowledge of experts. Another method of model creation involves process of training using monitoring or experiment data collected over a period of time. Once a model is created, the environment that produced training data might change significantly, and that would require model revision. For example, it might happen that the managed environment was patched to a new version of software. It is also possible that there was a change in a number of components within the managed environment, or there was a significant change in the way users use the managed environment. In all these cases a revision of a model is necessary.

In the Cloud Data Centre Monitoring case study, the produced models will be used for predicting energy consumption and temperature in a real life cloud environment, instrumented with CA products. Given a specific workload and a specific server, we may be able to predict the energy consumption and temperature of that particular server when running this workload.

However, conditions may change in dynamic environments and the models may require to be reviewed. The need for a model review can be derived from the monitoring and configuration data. For example, in a case where the environment was patched to a new version of software, monitoring data would reflect a change in behaviour that may not return to the previously observed pattern that was reflected in the training data used to create the model.

We will distinguish between two types of events:

- **Sporadic anomalies**: these are unexpected sporadic changes in the behaviour of the system that are not durable in time. Sporadic anomalies should be ignored in our case study, since models do not need to be recalibrated because the system behaviour has actually not changed. It is important to remark that, sporadic anomalies do not alter the model of what is normal, but they need to be identified and their cause and consequence need to be understood. Sporadic Anomalies can disrupt a system. In our case study we will need to explore the system when an anomaly is detected.
- **Durable changes**: are those changes that are reflected through durable alterations in behaviour of the system. Durable changes require model recalibration.

Therefore, in this case study we use both anomaly and change detection systems. These systems identify time of change and set of affected components so that a new set of data can be created after a durable change is detected, for the purpose of training a new version of the model. Moreover, when a new anomaly or change is detected an automatic root cause analysis process starts to help the analyst group several anomalies to reduce the cognitive load and to provide insight on the cause of those anomalies. Such diagnosing process is challenging due to the size of the monitored system as well as the noisiness associated to alarms in an information-poor environment, typical from non-intrusive monitoring tools.

The purpose of our work in LeanBigData is to explore novel extensions of the functionality of current monitoring tools by using efficient big data storage and analysis functionalities. Specifically, we plan to use LeanBigData technology to improve our capabilities to recalibrate our models depending on the data collected from real systems, by using anomaly and change detection techniques and root cause analysis. This goes beyond the state of the art because there is currently no industrial data centre monitoring tool that is able to build and keep updated a per-server and per-server-type energy and thermal models of the data centre, and information valuable for capacity planning, what-if analysis and planning of new facilities.

Since some of the decisions involved in this process might need human intervention (for instance to decide if an observed anomaly corresponds to a seasonal effect), we have created new visualizations and explored new HCI possibilities in collaboration with other partners in the project to improve the management of data centres to assist the users of these models.

4.2. Social Media Analytics

Social networks are becoming the most dynamic and easy-to-use mechanism to publish and exchange user-generated content. People use social networks for a variety of purposes, such as expressing opinions, spreading rumours, announcing gatherings, advertising products, etc. Among all social networks, Twitter has become the de facto source for real-time social media analytics due to a combination of factors: Twitter provides a platform for public short messages exchange used by billions of people, and a powerful open search and streaming APIs Twitter is an invaluable data source for a variety of purposes, including monitoring political elections.

The Social Network analytics case study is dealing with the challenging issue of using social media as complementary input for politicians and other actors in the prediction of elections outcomes, identification of relevant topics for voters, measure voter’s opinion during a political debate, etc. To that extent, the use case gathers data from social networks (mainly Twitter) related to a political party or event (debate, election) and monitors and analyses the activity, providing new insights about electorate opinion.

To that end a solution for social networks gathering and analysis based on the content of dedicated data channels will be developed. A data channel will be “listening” different set of conversations in Twitter (or other social networks), meaning that the data channels implement configurable user queries (based on keywords, hashtags, locations, etc.) to gather data (tweets) associated to the channel. The users are able to set up several data channels for different purposes or events, providing that the search limits provided by the APIs of the social networks (i.e. the limits of the public Twitter search and/or streaming APIs) are respected. As data channels depend on the source of the data gathered, the concept of data pool is introduced, as a stream aggregating related data with different sources. In the same way as data channels, the user will be also able to set up several data pools for different purposes. In that way, the analysis of the data is able to discriminate the data in order to analyse it in very flexible ways.

In the scope of LeanBigData, the case study is focused in the monitoring of an general election process, including pre campaign and debates, as a proof-of-concept of the validity of the approach in a politics business scenario.

Finally, it is important to point out that the case study delivers a couple of implementations of the software: one using big data “traditional” technologies and a second one offering the same results using LeanBigData components. Therefore, besides the functional aspects and the clear business oriented approach of the case studies, the main goal of this particular case study is to serve as a benchmark on how LeanBigData could hopefully help to develop, ease the development and improve the performance of this type of applications.

4.3. Financial Analytics

The Single Euro Payments Area (or “SEPA” for short) is where more than 500 million citizens, over 20 million businesses and European public authorities can make and receive payments in euro under the same basic conditions, rights and obligations, regardless of their location.
The overall gains expected from SEPA for all stakeholders has been evaluated at €21.9 billion per year by PWC in 2014 confirming a Cap Gemini study of 2008 evaluating these benefits at € 123 billion cumulated over 6 years.
SEPA adopts the ISO 20022 standard, a multi-part International Standard prepared by ISO Technical Committee TC68 Financial Services.
The recent adoption of the Single Euro Payments Area (SEPA), which follows the European Union (EU) payments integration initiative (http://ec.europa.eu/finance/payments/sepa), has moved more attention to the mechanisms to avoid frauds in banking/financial transactions.
As far as an SDD (SEPA Direct Debit) transaction is concerned, the SEPA standard has simplified a lot the payment process, while moving the consequences of a fraud from the user to the bank. In particular in an SDD transaction one person/company withdraws funds from another person's bank account. Formally, the person who directly draws the funds ("the payee or creditor") instructs his or her bank to collect (i.e., debit) an amount directly from another's ("the payer or debtor") bank account designated by the payer and pay those funds into a bank account designated by the payee.
Typically examples of SDD transactions are services that requiring recurrent payments, such as pay per view TV, energy distribution, credit card etc.
To set up the process, the creditor has to acquire an SDD mandate from the debtor and advise his/her bank about that mandate. Each time it will be needed, the Creditor sends a direct debit request (with amount specification) to his/her bank that will start the process to request the specified amount on the Debtor’s bank account.
The debtor only has to provide the signature of the mandate and the debtor could not receive communications about the SDD request. The debtor has no prior acknowledgement about the direct debit being charged to his bank account. Typically, the creditor sends a receipt to the debtor using a best effort service, so no guarantee is provided about delivery time and delivery itself. The debtor will only have access to the direct debit after the transaction has been completed. He/she can identify an unauthorized SDD amount only when it receives its bank statement.
Of course this exposes the debtor to a number of possible frauds. For this reason, with SEPA, in case of error/fraud with an SDD, a debtor can request for a refund until: 8 weeks after the SDD deadline or 13 months for unauthorized SDD (no or revoked mandate).

The Financial Banking application monitors the debit request to try to recognize unauthorized debit, thus providing an anti-fraud system helping the financial institution to reduce costs due to frauds. Unauthorized debits are typically due to the Identity Theft as shown in Figure 8.

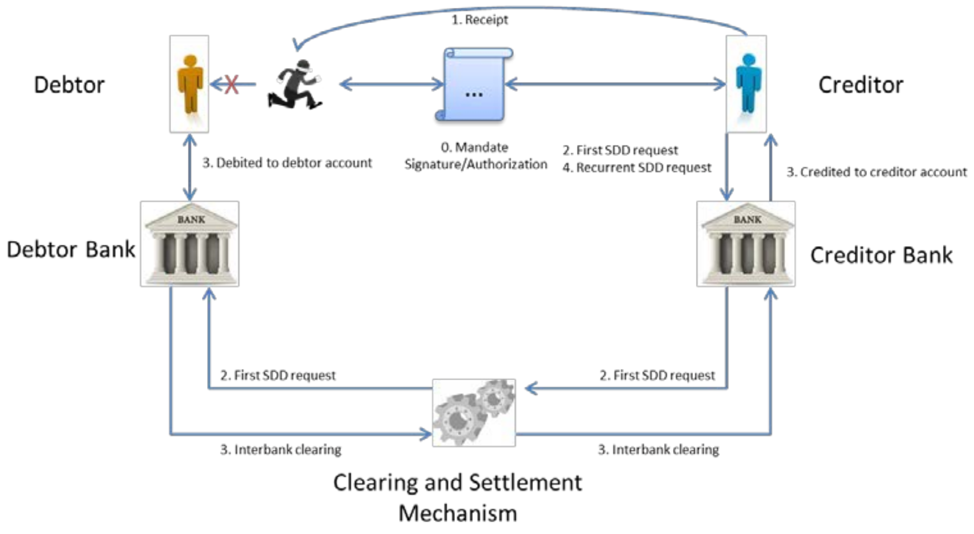


Figure 8 - SDD unauthorized

As depicted in Figure 8, the identity theft occurs at the beginning of the process, when the SDD mandate is being created. The SDD mandate is authorized not by the debtor, but by someone impersonating the debtor. This illegal behavior is also known as Mandate Fraud. For instance, this identify thief will benefit from the product/service being provided and charged by the creditor but will not pay for the service. The debtor will be charged for a product service that did not acquire and, will detect the fraud after the direct debit is performed.
Even in countries where a number of controls are enforced on the authorization set up process, the problem of direct debit fraud is extensive. A recent study (http://www.niceactimize.com/index.aspx?page=news372) highlights that more than half of businesses would be willing to switch their custom to a financial institution if it could provide better fraud prevention than their current provider.
Detecting frauds in SEPA Direct Debit transactions requires the ability of performing complex correlations – in a reliable and timely fashion - on data flows that are amazingly large in terms of volumes and highly heterogeneous in their format. The inherent complexity of the problem, combined to the fact that the SEPA standard has been adopted only recently, has resulted in the inability of the anti-fraud market of delivering efficient solutions (some partial solutions exist, but their scope is limited to the monitoring of credit card operations). Financial use case's approach to address this problem is to create a debtor profile (in terms of interests, visited places, etc.) by continuously processing multiple data flows, and to detect identity thefts by means of deviations of the observed profile from the debtor profile (a deviation is, as an example, a discrepancy between the debtor's interests and the service being purchased in an SDD transaction). This solution cannot be implemented – at the scale that is needed – on top of existing technologies. The high scalability of the LeanBigData platform, combined with the three different data management technologies - specifically: the improved SQL (for querying), anomaly detection (to identify anomalous requests), and CEP (for correlating and filtering events) allowed SyncLab to efficiently implement the proposed solution in terms of performance and cost. In *Figure 9* a screenshot of the developed prototype used to the visualization of the Debtor profile is shown

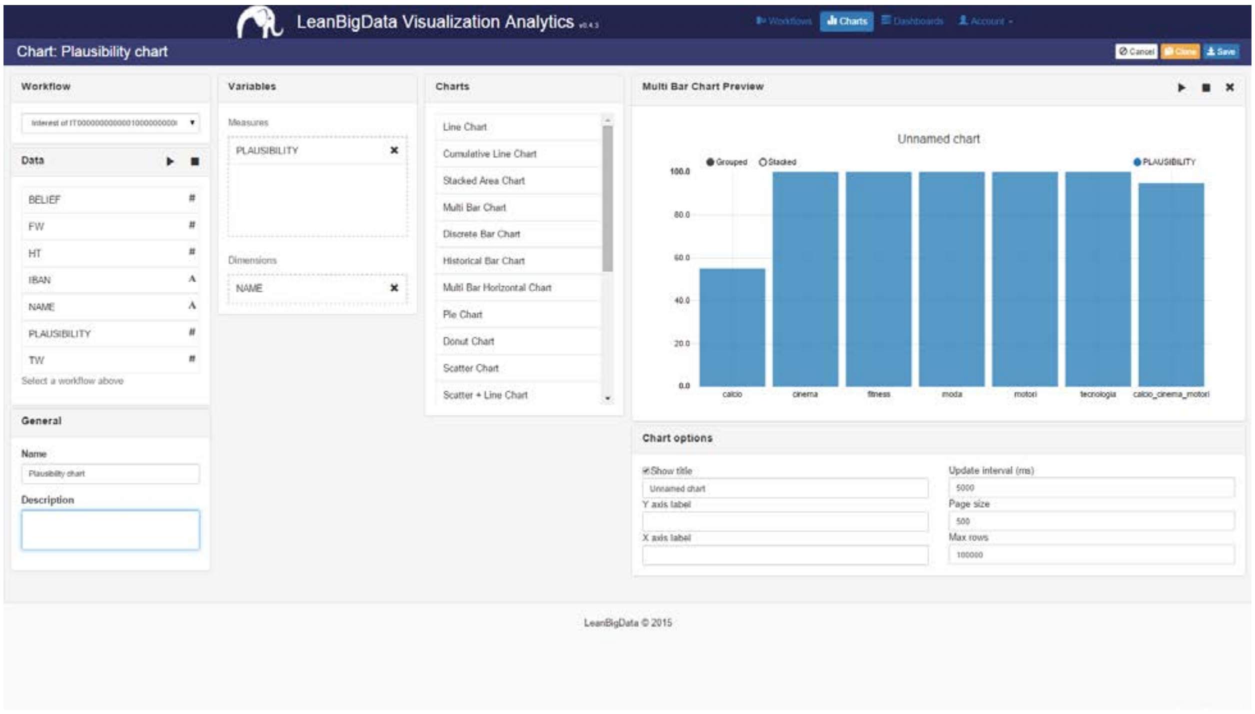


Chart: Plausibility chart

Workflow: Interest of IT0000000000001000000000

Measures: PLAUSIBILITY

Dimensions: NAME

Name: Plausibility chart

Description:

☑ Show title: Unnamed chart

Y axis label:

X axis label:

Update interval (ms): 5000

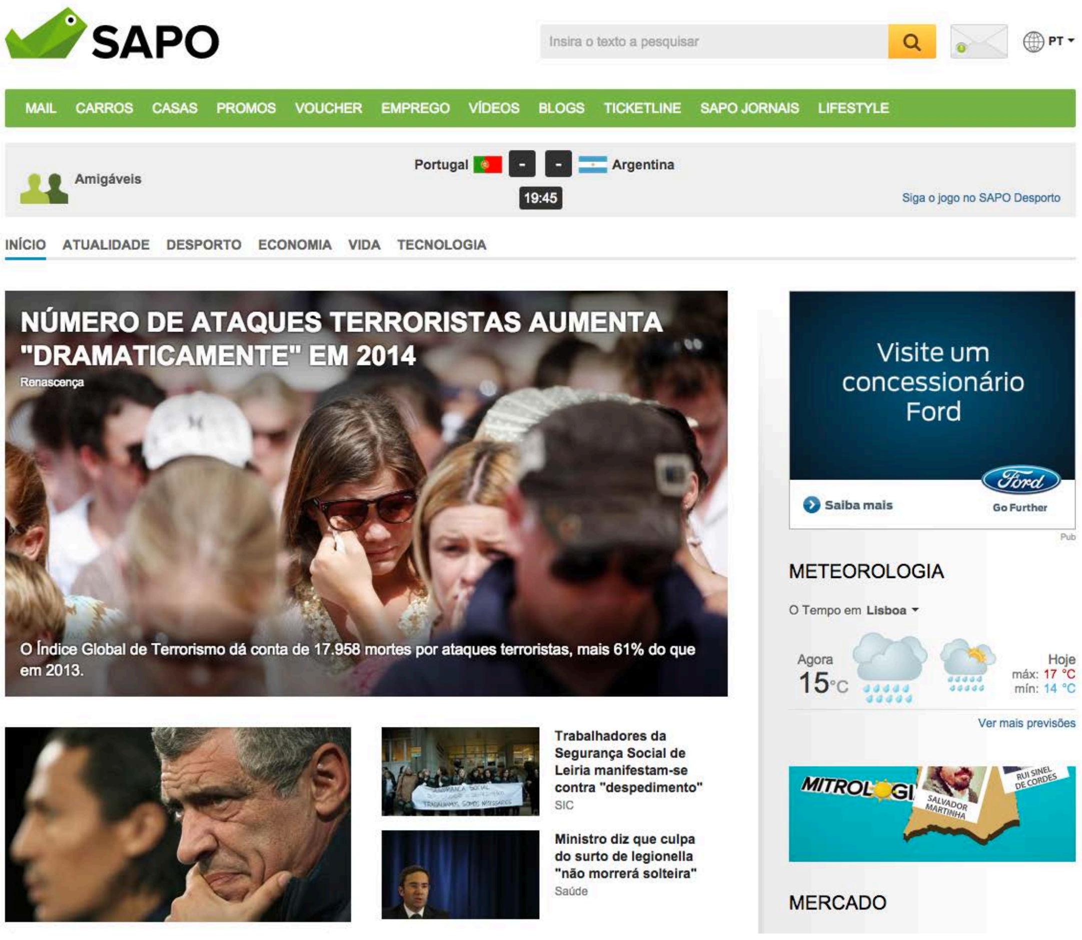
Page size: 500

Max rows: 100000

Figure 9 : *Debtor profile visualization*

4.4. Targeted Advertisement

SAPO, the internet technology department at Portugal Telecom, responsible for the most visited online portal in Portugal with associated services competing directly with Google and Yahoo services in the country, runs a vast big data and analytics infrastructure which tops up the data coming from the portal with data from the quadruple-play telecommunications business where PT is market leader in all sectors: mobile, internet, landline and TV.


SAPO

Insira o texto a pesquisar

PT

MAIL CARROS CASAS PROMOS VOUCHER EMPREGO VÍDEOS BLOGS TICKETLINE SAPO JORNAIS LIFESTYLE

Amigáveis

Portugal - - Argentina

19:45

Siga o jogo no SAPO Desporto

INÍCIO ATUALIDADE DESPORTO ECONOMIA VIDA TECNOLOGIA

NÚMERO DE ATAQUES TERRORISTAS AUMENTA "DRAMATICAMENTE" EM 2014

Renascença

O Índice Global de Terrorismo dá conta de 17.958 mortes por ataques terroristas, mais 61% do que em 2013.

Trabalhadores da Segurança Social de Leiria manifestam-se contra "despedimento"

SIC

Ministro diz que culpa do surto de legionella "não morrerá solteira"

Saúde

Visite um concessionário Ford

Saiba mais

Go Further

Pub

METEOROLOGIA

O Tempo em Lisboa

Agora 15°C

Hoje máx: 17 °C mín: 14 °C

Ver mais previsões

MITROLOGIA

MERCADO

Figure 1. SAPO Homepage

As part of PT’s Internet business, SAPO sells multiplatform ads online covering the whole spectrum of web, mobile and TV. Similarly to other industry standards, such as Google AdSense and AdWords, SAPO allows advertisers to define their own campaigns, set their campaign goals and budget, their choice of paid words, as well as many other constraints including geographic and demographic of the targeted customers.

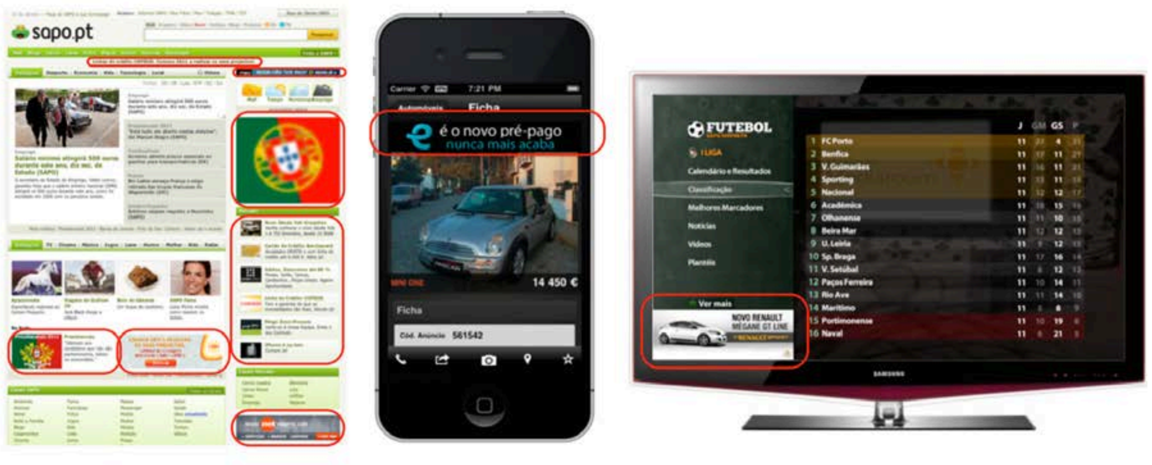


Figure 2. Ads being served on three platforms: web, mobile, TV.

Recent trends point to a maximisation of convergence synergies between all the distribution channels by allowing profiling to happen across the whole spectrum of data so that ads are served in a much more targeted fashion. An example of an intricate target ad strategy, would be to use EPG (Electronic Programming Guide) information and the customer's own use of the set top box to infer the program you are watching and to deliver ads on your laptop at home which are related to the programmes you are watching (e.g. a Mercedes Benz ad if you are watching Top Gear). A plethora of such examples exists.

Decisions on which ads to show in which client need to be made in a fraction of a second and should be informed by all the batch processing associated with profiling. To cope with these large streams of data SAPO currently uses a hodgepodge of big data technologies currently leading to high communication overheads and undesired operational complexity. The goal of this case study in the project is to make use of the LeanBigData platform to improve efficiency around the management of the existing data to allow faster and better queries at the database level and also to improve cross-domain analytics made possible through the convergence of the various data streams while enormously simplifying the application by relying on a single database, LeanBigData, providing combined OLTP and OLAP capabilities.

With LeanBigData PT explores novel extensions of the functionality of current third party and in-house tools by using efficient big data storage and analysis functionalities. In particular, PT plans to use LeanBigData technology to improve its capabilities to make online advertising campaigns more efficient and to recalibrate our profile clustering. With LeanBigData PT aims to explore novel extensions of the functionality of current third party and in-house tools by using efficient big data storage and analysis functionalities. In particular, PT plans to use LeanBigData technology to improve our capabilities to make online advertising campaigns more efficient and to recalibrate our profile clustering.

5. Conclusions

The LeanBigData project has made substantial progress during its second year. All the individual subsystems have been implemented. A first version of the platform has been completed by integrating the first batch of subsystems.

The use cases have finalized their initial implementation on top of the first version of the integrated platform. The feedback received from the use cases enabled to refine the architecture of LeanBigData.

During the last year the final version of the platform will be completed as well as the use cases and a full evaluation and validation of the platform will be performed.

6. References

[1] J. Dean and S. Ghemawat, “MapReduce: Simplified Data Processing on Large Clusters,” *Commun. ACM*, pp. 107-113, 2008.

[2] K. Shvachko, H. Kuang, S. Radia and R. Chansler, “The Hadoop Distributed File System,” in *Proceedings of the 2010 IEEE 26th Symposium on Mass Storage Systems and Technologies (MSST),* Washington, DC, USA, 2010.

[3] J. Dean and S. Ghemawat, “MapReduce: A Flexible Data Processing Tool,” *Commun. ACM,* vol. 53, pp. 72-77, 2010.

[4] “Bigquery,” 2011. [Online]. Available: http://code.google.com/apis/bigquery/.

[5] B. Chattopadhyay, L. Lin, W. Liu, S. Mittal, P. Aragonda, V. Lychangina, Y. Kwon and M. Wong, “Tenzing A SQL Implementation On The MapReduce Framework,” in *Proceedings of VLDB,*, 2011.

[6] A. Thusoo, J. Sarma, N. Jain, Z. Shao, P. Chakka, N. Zhan, S. Antony, H. Liu and R. Murthy, “Hive-a petabyte scale data warehouse using hadoop,” in *IEEE 26th International Conference on Data Engineering (ICDE),*, 2010.

[7] “MonetDB,” [Online]. Available: https://www.monetdb.org/Home.

[8] “Vertica,” [Online]. Available: http://www.vertica.com/.

[9] “GreenPlum,” [Online]. Available: http://www.pivotal.io/big-data/pivotal-greenplum-database.

[10] G. W. Burr, B. N. Kurdi, J. C. Scott, C. H. Lam, K. Gopalakrishnan and R. S. Shenoy, “Overview of candidate device technologies for storage-class memory,” *IBM Journal of Research and Developement,* vol. 52, pp. 449-464, 2008.

[11] S. Swanson and A. M. Caulfiled, “Refactor, Reduce, Recycle: Restructuring the I/O Stack for the Future of Storage,” 2013.

[12] J. Handy, “Objective Analysis Storage Industry Summit (NVM),” 28 January 2014. [Online]. Available: http://www.snia.org/nvsummit.

[13] J. Ousterhout, P. Agrawal, D. Erickson, C. Kozyrakis, J. Leverich, D. Mazi`eres, S. Mitra, A. Narayanan, G. Parulkar, M. Rosenblum, S. M. Rumble, E. Stratmann and R. Stutsman, “The case for ram clouds: Scalable high-performance storage entirely in dram,” in *SIGOPS Operating Systems Review*, 2010.