Towards a Big Data Analytics Framework for IoT and Smart City Applications

Chapter · January 2014

DOI: 10.1007/978-3-319-09177-8_11

4 authors, including:

Martin Strohbach
AGT International
44 PUBLICATIONS **812** CITATIONS
SEE PROFILE

Evangelos Gazis
Huawei Technologies
77 PUBLICATIONS **1,358** CITATIONS
SEE PROFILE

Towards a Big Data Analytics Framework for IoT and Smart City Applications

Martin Strohbach, Holger Ziekow, Vangelis Gazis, Navot Akiva

AGT International
Hilpertstrasse 35, 64295 Darmstadt, Germany

e-mail: {mstrohbach, hziekow, vgazis, nakiva}@agtinternational.com

Abstract. An increasing amount of valuable data sources, advances in Internet of Things and Big Data technologies as well as the availability of a wide range of machine learning algorithms offers new potential to deliver analytical services to citizens and urban decision makers. However there is still a gap in combining the current state-of-the art in an integrated framework that would help reducing development costs and enable new kind of services. In this chapter we show how such an integrated Big Data analytical framework for Internet of Things and Smart City application could look like. The contributions of this chapter are threefold: (1) we provide an overview of Big Data and Internet of Things technologies including a summary of their relationships, (2) we present a case study in the smart grid domain that illustrates the high level requirements towards such an analytical Big Data framework, and (3) we present an initial version of such a framework mainly addressing the volume and velocity challenge. The findings presented in this chapter are extended results from the EU funded project BIG and the German funded project PEC.

1 Introduction

In times of increasing urbanization, local decision makers must be prepared to maintain and increase the quality of life of a growing urban population. For instance, there are major challenges related to minimizing pollution, managing traffic as well as making efficient use of scarce energy resources. For instance, in regard to congested traffic conditions, the Confederation of British Industries estimates that the cost of road congestion in the UK is GBP 20 billion (i.e., USD 38 billion) annually. In addition to challenges related to the efficient use of natural and manmade resources ensuring the health and safety of urban citizens, e.g. in the context of large events or supporting law enforcement are key concerns of a modern smart city.

In order to address these challenges urban decision makers as well as citizens will need the capacity to make the right assessment of urban situations based on correct data, and, more importantly, they will need the key information contained in the data to assist them in their decision processes.

adfa, p. 1, 2011.

As put by Neelie Kroes, EU commissioner for the Digital Agenda, data is the new gold, meaning that data is a valuable resource that can be mined for creating new values. In the context of smart cities, there is an abundance of data sources that can be mined by applying data analytics techniques and generate value by offering innovative services that increase citizens' quality of life. Data may be provided by all stakeholders of a Smart City [10], i.e. the society represented by citizens and businesses and governments represented by policy makers and administrations.

On one hand data sources may include traditional information held by public bodies (Public Sector Information, PSI) including anonymous data such as cartography, meteorology, traffic and any kind of statistics data as well as personal data, e.g. from public registries, inland revenues, health care, social services etc. [67].

On the other hand citizens themselves create a constant stream of data in and about cities by using their smartphones. By using apps like Twitter and Facebook, or apps provided by the city administration, they leave digital traces related to their activities in the physical city that has the potential to create valuable insights for urban planners.

With the advent of deployed sensor systems such as mobile phone networks, camera networks in the context of intelligent transportation systems (ITS) or smart meters for metering electricity usage, new data sources are emerging that are often discussed in the context on the Internet of Things (IoT), i.e. the extension of the internet to virtually every artifact of daily life by the use of identification and sensing technologies.

Thus we can summarize that the key components required for Smart City application are available: 1) an abundance of data sources, 2) infrastructure, networks, interfaces and architectures are being defined in the IoT and M2M community, 3) a vast range of Big Data technologies are available that support the processing of large data volumes, and 4) there is ample and wide knowledge about algorithms as well as toolboxes [62] that can be used to mine the data.

Despite all the necessary conditions for a Smart City are met, there is still a lack of an analytical framework that pulls all these components together such that services for urban decision makers can easily be developed.

In this chapter we address this need by proposing an initial version of such an analytical framework that we derived based on existing state-of-the art, initial findings from our participation in the publicly funded projects Big Data Public Private Forum (BIG) [7] and Peer Energy Cloud (PEC) [46] as well as our own experiences with analytical applications for Smart Cities.

The European Project BIG is a Coordinated Support Action (CSA) that seeks to build an industrial community around Big Data in Europe with the ultimate goal to develop a technology roadmap for Big Data in relevant industrial sectors. As part of this effort the BIG project gathers requirements on Big Data technologies in industry driven working groups. Groups relevant for Smart Cities include health, public sector, energy and the transport working group. The project has released both an initial version of requirements in these and other sectors [67] as well as a set of technical white papers providing an overview of the state of the art in Big Data technologies [31]. In this chapter we draw from these results of the BIG project and complement it with the

findings from a concrete use case in the energy sector as carried out in the PEC project.

The remainder of this chapter is structured as follows: section 2 provides background about the technical challenges associated with the Big Data and Internet of Things topics. Section 3 summarizes the state of the art in Big Data technologies. In section 4 we elaborate on the concrete Big Data challenges that need to be addressed in the context of Smart City applications. Section 5 presents a case study from the smart grid domain that demonstrates how we applied big data analytics in a realistic setting. In section 6 we extend the analytics presented in this case study towards an initial big data analytics framework. In section 7 we report on our lessons learned. In section 8 we summarize further research directions required to extend the framework fully implement it. Finally section 9 concludes this chapter.

2 Background: Big Data and the Internet of Things

In this section we describe the technologies that are concerned with connecting everyday artefacts, i.e. the Internet of Things, and relate them to Big Data technologies that address the challenges of managing and processing large and complex data sets. For an extensive discussion and definition of the term Big Data we refer to the respective report of McKinsey [37].

2.1 The Various Faces of Big Data

Although managing and processing large data sets is not fundamentally new, during the past years a range of technologies have emerged that facilitate the efficient storage and processing of big data sets. While Big Data technologies such as Map Reduce [14] and Hadoop [63] are the result of big Internet companies such as Google and Yahoo!, the need to handle and process large data sets is quickly extending to other sectors. For instance, the amount of various patient data in the health sector offers new opportunities for better treatments [67] and the advent of smart meters, allows utility provider to better cope with the instabilities of the grid caused by renewable energy sources [33]. From a technological perspective Big Data challenges and technologies can best be described along the so-called 3 V's: Volume, Velocity, and Variety [32].

The Volume Challenge. The Volume challenge refers to storing, processing and quickly accessing large amounts of data. While it is hard to quantify the boundary for a volume challenge, common data sets in the order of hundreds of Terabyte or more are considered to be big. In contrast to traditional storage technologies such as relational data base management systems (RDBMS), new Big Data technologies such as Hadoop are designed to easily scale with the amount of data to be stored and processed. In its most basic form, the Hadoop system uses its Hadoop Distributed File System (HDFS), to store raw data. Parallel processing is facilitated by means of its Map Reduce framework that is highly suitable for solving any embarrassingly parallel

processing problems. With Hadoop it is possible to scale by simply adding more processing nodes to the Hadoop cluster without the need to do any reprogramming as the framework takes care of using additional resources as they become available.

Summarizing the trends in the volume challenge one can observe a paradigm shift with respect to the way the data is handled. In traditional database management systems the database design is optimized for the specific usage requirements, i.e., data is pre-processed and only the information that is considered relevant is kept. In contrast, in a truly data-driven enterprise that builds on Big Data technologies, there is awareness that data may contain value beyond the current use. Thus a master data set of the raw data is kept that allows data scientists to discover further relationships in the data, relationships that may reside beyond the requirements of today. As a side effect it also reduce the costs of human error such as erroneous data extraction or transformation.

The Velocity Challenge. Velocity refers to the fact that data is streaming into the data infrastructure of the enterprise at a high rate and must be processed with minimal latency. To this end, different technologies are applicable, depending on the amount of state and complexity of analysis [57]. In cases where only little state is required (e.g., maintaining a time window of incoming values), but complex calculations need to be performed over a temporally scoped subset of the data, Complex Event Processing (CEP) engines (see section 3.3) offer efficient solutions for processing incoming data in a stream manner. In contrast, when each new incoming data set needs to be related to a large number of previous records, but only simple aggregations and value comparisons are required, noSQL databases offer the necessary write performance. The required processing performance can then be achieved by using streaming infrastructures such as Storm [56] or S4 [51].

The Variety Challenge. In a data-driven economy the objective is to maximize the business value by considering all available data. From a technical perspective one could formulate that problem by evaluating a function over all accessible data sets [38]. In practice, however, this approach must confront the challenge of heterogeneous data sources ranging from unstructured textual sources (e.g., social media data) to the wide disparity in the formats of sensor data. Traditionally this challenge is addressed by various forms of data integration. In the context of Big Data there is however a new dimensions to the integration challenge which is the amount of different data sources that need to be integrated. Social media, open (governmental) data sources [12], [21], and data platforms [64] and markets [11], result in a data ecosystem of significant variation. As the integration of new data sources requires manual work to understand the source schema, to define the proper transformations and to develop data adapters, existing approaches do not scale effectively.

Veracity. Apart from the original 3V's described above, an almost inexhaustible list of Big Data V's are discussed. For instance veracity relates to the trust and truthfulness of the data. Data may not fully be trusted, because of the way it has been acquired, for instance by unreliable sensors or imperfect natural language extraction

algorithms, or because of human manipulation. Assessing and understanding data veracity is a key requirement when deriving any insights from data sets.

Visualization. Visualization of big data is particularly important for data scientists that try to discover new patterns in the data that can exploited for creating new business value, e.g. by creating new services by combining seemingly unrelated data sets.

2.2 Internet of Things

Over the last decade, there has been a growing research interest in the "Internet of Things" – a disruptive technology, according to the US National Intelligence Council [59]. Despite its recent popularity, the term "Internet of Things" was actually first heard of in the previous century. The original definition envisioned a world where computers would relieve humans of the Sisyphean burden of data entry by automatically recording, storing and processing in a proper manner all the relevant information about the things involved in human activities [29]. Henceforth, and depending on the viewpoint, different understandings and definitions of what the "Internet of Things" is about have been reported in the literature [2][39][26]. The European Commission envisions it as an integrated part of the Future Internet where "Things having identities and virtual personalities operating in smart spaces using intelligent interfaces to connect and communicate within social, environmental, and user contexts [28]". The use of standard technologies in the World Wide Web to instrument the "Internet of Things" is frequently referred to as the "Web of Things".

Prominent players in all the major ICT markets (i.e., information technology, data networking, telecommunications, etc.) have publicly acknowledged the challenges brought on and the potential entailed by the Internet of Things (IoT) and Machine to Machine Communications (M2M). For instance, NEC expects that the M2M market will expand to approximately JPY 330 billion by 2015, while the Big Data market will expand to JPY 630 billion by 2017 and will exceed JPY 1 trillion by 2020. The 58.5% annual growth observed in 2010 in deployed M2M devices (as quantified via M2M SIM cards) in EU27 stands as market evidence in support of these estimates.

M2M Standardization. Making rapid progress over the last couple of years, the ETSI Technical Committee (TC) on Machine to Machine (M2M) communications has recently published its first version of M2M specifications. The objective is to define the end-to-end system architecture that enables integration of a diverse range of M2M devices (e.g., sensors, actuators, gateways, etc.) into a platform that exposes to applications a standardized interface for accessing and consuming the data and services rendered through these (typically last mile) devices [16]. To this end, ETSI M2M standards define the architecture, interfaces, protocols and interaction rules that govern the communication between M2M compliant devices.

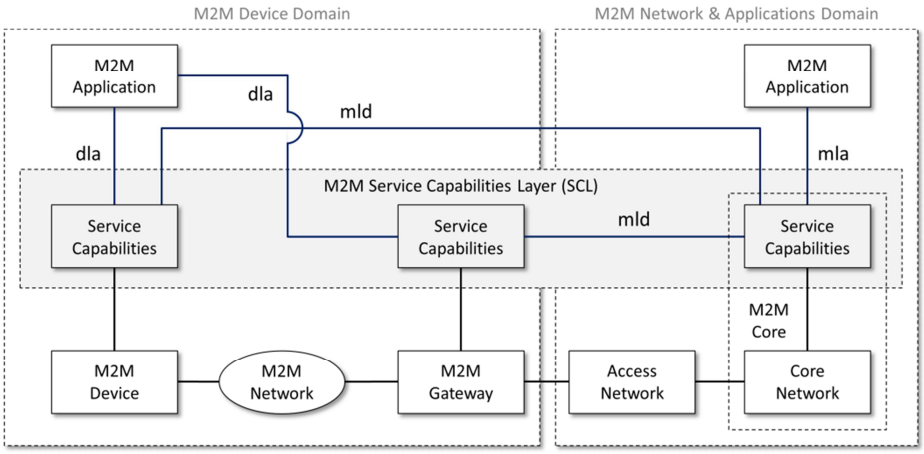

Fig. 1. The ETSI M2M architecture.

The M2M logical architecture under development in ETSI comprises two high level domains:

1. The Network and Application (NA) domain, composed of the following elements:
   - M2M Access Network (AN) providing for communication between the Device Domain and the Core Network (CN).
   - M2M Core Network (CN) providing for IP connectivity and the associated control functions to accommodate roaming and network interconnection.
   - M2M Service Capabilities (SC) providing functions shared by M2M applications by exposing selected infrastructure functionalities through network interfaces whilst hiding realization details.
   - M2M Applications running the actual application logic.

2. The Device (D) domain, composed of the following elements:
   - M2M Gateway using M2M SC for interconnecting to the NA domain and interworking M2M Devices to it. The M2M Gateway may also run M2M applications.
   - M2M Device that runs M2M applications using M2M SC functions and connecting to the NA domain through any of the following modes:
   - Direct, where the AN provides connectivity.
   - Proxy, where the M2M Gateway provides connectivity by acting as a proxy.
   - M2M Area Network (ArN) interconnecting M2M Devices and M2M Gateways through a field-specific networking technology, e.g., Power Line Communications (PLC), KNX, M-BUS, etceteras.

Management plane functions include network management functions (i.e., functions for managing fault, configuration, accounting, performance and security aspects) for the AN and CN domains, as well as management functions specific to M2M. An M2M application may use any combination of the Service Capabilities available in

the D and NA domains. These Service Capabilities are accessed through the following reference points:

- mIa, for the NA domain, allowing access and use of Service Capabilities therein.
- dIa, for the D domain, allowing an M2M application residing in an M2M host (i.e., Device or Gateway) to access and use different Service Capabilities in the same M2M host. When the M2M host is an M2M Device, access and use of different Service Capabilities in an M2M Gateway is supported also.
- mId, for the communication between M2M Service Capabilities residing in different M2M domains.

Over these references, resource management procedures adopt the RESTful style for the exchange and update of data values on the basis of CRUD (Create, Read, Update, Delete) and NE (Notify, Execute) primitives.

The role of wireless technologies in ETSI M2M is that of connectivity with minimal infrastructure investment both in the Device domain and the Network and Applications domain.

On a global scale, the oneM2M Partnership Project, established by seven of the world's leading information and communications technology (ICT) Standards Development Organizations (SDOs) is chartered to the efficient deployment of M2M systems. To this end, existing ETSI M2M standards are to be transferred to oneM2M and ratified as global M2M standards.

Smart Cities. By amassing large numbers of people, urban environments have long exhibited high population densities and now account for more than 50% of the world's population [58]. With 60% of the world population projected to live in urban cities by 2025, the number of megacities (i.e., cities with a minimum population of 10 million people) is expected to increase also. It is estimated that, by 2023, there will be 30 megacities globally.

Considering that cities currently occupy 2% of global land area, consume 75% of global energy resources and produce 80% of global carbon emissions, the benefit of even marginally better efficiency in their operation will be substantial [58]. For instance, the Confederation of British Industries estimates that the cost of road congestion in the UK is GBP 20 billion (i.e., USD 38 billion) annually. In London alone, introduction of an integrated ICT solution for traffic management resulted in a 20% reduction of street traffic, 150 thousand tons of CO_2 less emissions per year and a 37% acceleration in traffic flow [18].

Being unprecedentedly dense venues for the interactions – economic, social and of other kind – between people, goods and services, megacities also entail significant challenges. These relate to the efficient use of resources across multiple domains (e.g., energy supply and demand, building and site management, public and private transportation, healthcare, safety and security, etc.). To address these challenges, a more intelligent approach in managing assets and coordinating the use of resources is envisioned, based on the pervasive embodiment of sensing and actuating technologies throughout the city fabric and supported by ubiquitous communication networks and

the ample processing capacity of data centers. The umbrella term Smart City [68] refers to the application of this approach in any of six dimensions:

- Smart economy
- Smart mobility
- Smart environment
- Smart people
- Smart living
- Smart governance

By aggregating data feeds across these domains and applying data processing algorithms to surface the dominant relationships in the data, the situational awareness of the Smart City at the executive level becomes possible. For instance, by leveraging its open data initiative, the city of London provides a dashboard application demonstrating the kind of high-level oversight achievable by cross-silo data integration and the use of innovative analytic applications [35].

The footprint of our current cities' impact is growing at 8% annually, which means it more than doubles every 10 years. Thus not surprisingly, NIKKEI estimates that USD 3.1 trillion will be invested globally in Smart City projects over the next 20 years [69].

Relationship to Big Data. The popularity of data mashup platforms, as evident today for human-to-machine and machine-to-human information, is expected to extend to machine-to-machine information [16]. Data generated in the context of machine-to-machine communication are typically not constrained by the processing capacities of human entities in terms of volume, velocity, and variety. Particularly in regard to velocity, the ongoing deployment of a large number of smart metering devices and their supporting infrastructures across urban areas increases the percentage of frequently updated small volume data in the overall data set of the Smart City. Thus M2M data exchanges in the context IoT applications for a Smart City impact upon the requirements of data handling through an increase in the volume, variety and velocity of the data.

Considering the trinity of IoT, M2M and Smart Cities from the standpoint of cloud technologies, it becomes apparent that the scalability to a large number of M2M devices (i.e., sensors, actuators, gateways) and data measurements will be a prime (non-functional) application requirement. It is, therefore, apparent, that IoT, M2M and Smart Cities are, from a requirements perspective, right at the core of what Big Data technologies provides.

Increasing urbanization and M2M deployment bring on significant increases in the data generated by IoT applications deployed in the Smart City fabric. For instance, the London Oyster Card data set amounts to 7 million data records per day and a total of 160 million data records per month [4]. Given that a Smart City generates a wide spectrum of data sets of similar – and even larger – size, challenges characteristic of Big Data arise in collecting, processing and storing Smart City data sets.

3 State of the Art

In this section we describe state of the art of Big Data according focusing on the volume and velocity challenge. For this section, we describe the state of the art mainly from an industrial perspective, i.e. we provide examples of available technologies that represent technologies that can also be used in a productive environment.

3.1 Big Data

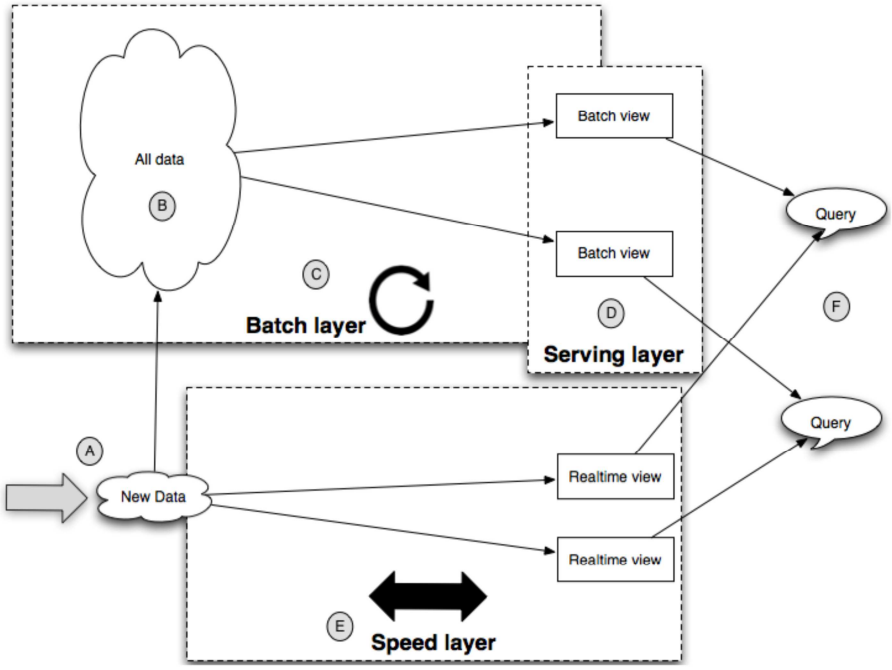

Fig. 2. Lambda Architecture (Source: Big Data – Principles and best practices of scalable realtime data systems, ISBN 9781617290343 [38])

As described in the previous section, the advances in IoT infrastructures, combined with the social demands for more efficient resource usage in densely populated urban environments, introduces data-related challenges that are at the focus of Big Data toolsets. The latter include technologies and tools that support solving the volume challenge. A range of noSQL databases are able to keep up with high update rates. Programming frameworks such as MapReduce [14] help processing large data in batches. And Stream processing infrastructure such as Storm [56] and S4 [51] provide support for scalable processing of high velocity data. In practice both technologies are required in order to design a low latency query system. Marz and Warren have de-

vised the term Lambda architecture as an architectural pattern that defines the interplay between batch and speed layer in order to provide low latency queries [38].

New data (A) is provided both to the batch and speed layer as depicted in Fig. 2. The batch layer (C) stores all incoming data in its raw format as master data set (B). It is also responsible for running batch jobs that create batch views in a serving layer (D) that is optimized for efficient querying (F). The batch layer is optimized for processing large amount of data, e.g. by using the MapReduce framework and may require hours to process data. Consequently, the batch view will not be updated between repeated executions of the batch jobs and the corresponding data cannot be included in the result set of a query to the serving layer.

The speed layer (E) addresses this information gap by providing a real-time view on the data that has arrived since the last executed batch job. This way an application will always have up-to-date information by querying both the serving and speed layer.

3.2 Batch Processing

In order to cope with the volume challenge so-called noSQL databases are gaining in popularity. Those databases can be classified according to their scalability with respect to data size and suitability to model complex data relationships [43]. With decreasing ability to handle data size and increasing capability to handle complex relationships, we can distinguish between key-value stores, columnar stores, document databases, and graph databases.

Key-value stores are databases that scale easily, but are only able to capture simple relationships, i.e. a key and its value. Columnar stores in contrast store data in columns and are thus better suited for queries that access only small portions of high dimensional data. Document databases are able to store even more complex data, but typically do not scale as well. Graph databases in contrast can easily model any relationship, but do not scale as easily.

The actual processing of data uses different computational frameworks such as Hadoop's MapReduce [63], Hama's Bulk Synchronous Processing framework [53], or graph processing frameworks [24]. They are designed to process large volumes of data in batches, i.e. in regular intervals. As a consequence these technologies are not suitable to process data in real-time.

3.3 Real-time analytics

It is commonly acknowledged that, in a lot of economically significant application domains, the value of information decreases as it ages. That is, the more recently the data (and the respective information drawn from it) has been acquired, the more valuable it is. The challenge of processing high velocity data in real-time calls for dedicated solutions that can handle data in motion. Today a number of solutions exist that are designed for executing complex logic over continuous data flows with high performance. These solutions are typically referred to as stream processing or complex event processing systems. The terms stream processing and event processing developed independently and one may argue for some differences in the underlying philos-

ophies. However, the terms are increasingly used interchangeably and the corresponding solutions follow similar principles. We will subsequently only use the term Complex Event Processing (CEP) and subsume stream processing under this term.

CEP engines are designed for implementing logic in the form of queries or rules over continuous data flows. Typically they include a high level declarative language for the logic definition with explicit support for temporal constructs. The defined logic is executed by the engine using processing techniques that are optimized for continuous data flows. In contrast to batch-driven processing that is triggered on request, CEP engines process incoming data continuously in an event-driven manner. Another difference to batch systems is that CEP engines do not persist information. That is, they operate on temporal windows or synopsis of the incoming data in memory. Consequently the scope of this analytics is live data or data about the recent past as opposed to long term analysis of recorded information.

The need for CEP technology is rooted in various domains that required fast analysis of incoming information. Examples can be found among others in the finance domain where CEP technologies are used for applications like algorithmic trading or the detection of credit card fraud [27]. In these cases CEP is very well suited due to applications requiring fast analysis of high volume data streams involving temporal patterns. For instance, a credit card fraud may be detected if multiple transactions are executed in short time from far apart locations. Other application domains for CEP include fields like logistics [60], business process management [61], and security [22]. Also, the IoT domain has sparked a range of applications that require event-driven processing and is one of the drivers for CEP technology. Specifically the field of sensor networks has early on let to development of systems that are designed for processing in an event-driven manner (e.g. TinyDB [36], Aurora/Borealis [**Error! Reference source not found.**]).

Over the last years a number of CEP solutions have emerged in academia as well as in industry. Some of the known early academic projects are TelegraphCQ [8], TinyDB [36], STREAM [41], Aurora/Borealis [**Error! Reference source not found.**] and Padres [30]. The research initiates were followed by (or directly led to) the emergence of several startups in this domain. For instance the company StreamBase is based on the Aurora/Borealis project and results from the STREAM project fueled the startup Coral8 and the CEP solution of Oracle. Other vendors software vendors like Microsoft and IBM have created CEP solution based on their own internal research projects (CEDR [3], System S [**Error! Reference source not found.**]). In addition several major vendors have strengthened or established CEP capabilities through acquisitions in recent years. Some examples to name are the acquisition of Apama by Software AG, StreamBase Systems by TIBCO, and Sybase by SAP.

Next to purely commercial offerings the market includes offerings that are available as open source solutions. Example include engines like Esper [19], ruleCore [50] or Siddhi [55]. Noticeable additions to the open source domains are the solutions Storm [56] and S4 [51]. These solutions are not classical CEP engines in the sense that they do not provide a dedicated query language. In contrast, Storm and S4 provide event processing platforms that are focusing on support for distributing of logic to achieve scalability.

4 The Big Smart City Integration Challenges

In this section we describe in brief the challenges related towards an integrated solution for scalable analysis of Smart City data sources. We consider these challenges mainly from an integration point of view along two dimensions. First there is the question how batch and stream processing should be integrated in a modern Smart City environment (section 4.1). And second, there is the challenge how the variety of data source should be handled in order to efficiently deliver new services and analyze these data sets as a whole rather than in isolation (section 4.2). As social media sources provide a potentially rich source of information, we describe them separately (section 4.3).

4.1 Integration of batch and stream processing

New kind of smart city application will greatly benefit from elaborate analytical algorithms. For instance in order to make prediction about traffic patterns, crime [47], diseases or energy consumption, it is necessary to first learn patterns from the data, e.g. using statistical or machine learning algorithms. In a second step, the model is applied to new data making decisions based on the model such as the future energy consumption.

As the learned models do not need to be adapted with every incoming data set, state-of-the art big data batch processing approaches are suitable to learn models on large data sets. Vice versa stream processing approaches are suitable to evaluate new data in real-time.

Note that this is different from Marz' and Warren's main application of the Lambda architecture with applications in mind that require a scalable and robust system design, for querying large amounts of data with low latencies as for instance required for real-time Business Intelligence dashboard visualizations. These kind of applications require essentially the same logic in the batch and stream layer whereas in model-based analysis the logic differs.

The challenge is therefore to devise distributed version of known model learning algorithms and implement them on common distributed processing frameworks such as MapReduce [14]. As with any application of statistical or machine learning model, the main challenge is to find the right features. In particular it is necessary to define interactions between the batch and speed layers required to access the model. However, the biggest challenge is finding the right combination of technologies that allow for a scalable and robust design. On top of that, the design must ensure that the interactions defined for accessing the model do not run counter to the realization of efficient and scalable processing.

4.2 Integration of heterogeneous data sources

The spectrum of IoT data sources includes sensor data, product databases, or data extracted from the web, including social media. Different data sources model data in different ways and use different protocols and interfaces for communication. To avoid

forcing each IoT application to understand a multitude of different data models, including the encoding and semantics of the consumed data as well as the protocol used to access it, the IoT platform must accommodate many different data models by supporting extensions and a continuum in the evolution of data models and render those to the application in a standard, semantically enriched format. This will also enable better integration between data sources thus facilitating the analysis over disparate data sets.

It is understood that IoT applications vary in terms of the data sets they employ in their function and in the non-functional requirements (e.g., reliability, scalability, etc.) imposed upon their operation. Therefore, selecting the solution technology set that renders the intended function while meeting the non-functional requirements of the application domain will be an important concern. For instance, some IoT applications will require real-time processing of frequently updated small volume structured data sets, while other will require complex analytic operations on large volumes of infrequently updated but semantically enriched unstructured data sets. The wide range of operational requirements entailed by this disparity, suggests that the proper instrumentation of the data processing stage will be a paramount concern for IoT applications in Smart Cities. Such instrumentation matters need to be addressed in conjunction with instrumentation options arising from section 4.1 above.

4.3 Natural Text and Social Media Analysis

Nowadays, social networks are widely accessible via mobile devices such as smart phones making them a rich source for monitoring citizen's behaviors and sentiments in real time. Social networks such as Twitter, Facebook, Foursquare etc. provide access to various location-based information reported by their users, though usually in an unstructured format.

Integrating social sensors data with the IoT could be highly beneficial in several aspects:

1. Contextualization of physical phenomena by providing a subjective context signaling (i.e., explanation) on top of physical events detected by physical sensors
2. Calibration of noisy events detected by physical sensors, by providing an additional supportive signaling of the same events by the social sensors.
3. Detection of 'below the radar' events by combining both subjective and objective sensors data, which otherwise would not have been detected using each separately.

The initial challenge when considering the integration of social networks reports as sensorial insights is the extraction of such signals. Extracted sensorial data might include peoples' stated opinions and statements regarding a particular sentiment, topic of interests, reported facts about one's self (e.g., illness, vacation, event attendance) and various additional subjective reports. Natural language processing (NLP) along with Machine Learning algorithms are applied for extracting the relevant signals from the data posted by the users of the social medium.

An additional challenge is the reliability/credibility of the social sensors. Naturally, the model-based inferred signals are attached with a certainty level produced by the

model, where handling the data uncertainty is not trivial. Moreover, the reporting user could also be attached with a credibility score which measures the overall worthiness of considering its data in general.

The sparsity of geo-tagged data in social network data is another challenge when considering its value for integration with the IoT. For example, only 1% of all Twitter messages are explicitly geo-tagged by the users. Recent work suggests several methodologies to overcome this challenge by inferring the users' location based on its context [13] [17].

Finally, there is a need for an architecture that combines both batch and stream processing over social data, for achieving a couple of goals:

1. Enabling the combination of offline-modeling over vast amounts of data and applying the resulting model over streaming data in real time. Most of the complex semantic analysis tasks, as for instance Sentiment Analysis [34] [45] require batch modeling. Feature extraction could be done in real time over a data stream by applying a sliding time window (e.g. 20 seconds) over measures as terms' frequency or TF-IDF [52]. For addressing the challenge of evaluating models in data streams where the data distribution changes constantly, a sliding window kappa-based measure was proposed [6].
2. Analyzing data of all social sensor types, either streaming (e.g. Twitter) or non-streaming (e. g. blog posts) using the same architecture.

5 Case Study: Smart Grid Analytics

In this section we discuss a case study from the smart grid domain that illustrates the application of big data on smart home sensor data. The case is taken from Peer Energy Cloud (PEC) project [46] that runs a smart grid pilot in a German city. The pilot includes installations of smart home sensors in private homes that measure energy consumption and power quality such as power voltage and frequency at several power outlets in each home. Each sensor takes measurements every two seconds and streams the results into a cloud-based infrastructure that runs analytics for several different use cases. In this section we discuss the technical details of three scenarios that we implemented in our labs using data from deployed sensors:

1. **Power quality analytics** – shows the benefits of big data batch processing technologies.
2. **Real-time grid monitoring –** shows the benefits of in-stream analytics.
3. **Forecasting energy demand –** shows the need to combine both batch and stream processing.

5.1 Power Quality Analytics

Power quality analytics addresses the identification of problem areas within the distribution grid. The fine grained sensing allows for detecting power quality issues in the last mile of the grid, down to the household level. For instance, voltage fluctuations

can be detected and compared between houses or streets. This enables to pinpoint hotspots of power quality problems and identify root causes. Fig. 3 below shows exemplary the statistics that identify problem hotspots (left) and an analysis of the root cause (right). Spatio-temporal clustering of power quality anomalies can provide operators with insights about the circumstances when power quality issues arise.
Clustering analysis may be further extended adding additional dimensions such as weather conditions, or periodic time attributes like season or time of the day. Together, these analysis support exploratory investigation of root causes and planning of counter measures.

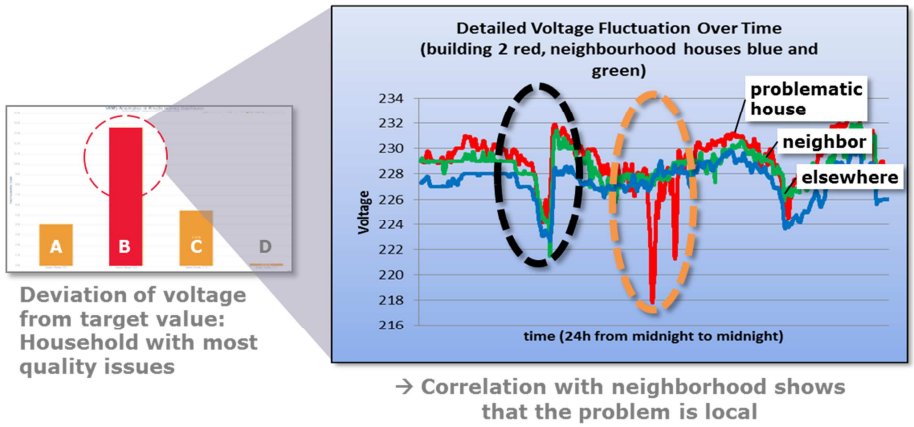

Fig. 3. Analysis of voltage deviations.

The big data challenges in this use cases arises due to the high data volumes. Every household produces almost 2 million energy related measurements a day. This number has to be multiplied by the number of households that use the technology and accumulates over time. For instance, about 200,000 million voltage measurements per month would be available in a full rollout in the small pilot city of Saarlouis.

The query logic for power quality analytics is relatively simple but poses challenges due the high data volumes. Experiments in the pilot project revealed operational challenges already when implementing the analytics for the first four pilot households. Already at this limited scope relational databases required tweaking to handle the queries. However, using the MapReduce programming model and the Hadoop framework it was straight forward to implement the power quality analytics in a scalable way. This is because the underlying analytics problem is of embarrassingly parallel nature and hence very well suited for parallel processing. For instance, it is trivial to partition the data for analyzing household specific voltage fluctuations by household and the Hadoop based implementation achieved close to linear scalability.

5.2 Real-time Grid Monitoring

The second use case – real-time grid monitoring – is about providing live insights into the current state of the electrical grid. Industrial control systems typically provide measures of the grid down to the level of secondary substations. With smart homes it now becomes possible to monitor the distribution grid on a level that is not covered by current infrastructures. For instance, power quality measures and the consumed power can be observed for each customer in real-time. This allows detecting critical states and aid responses on a local level. For instance, customers may selectively get demand response signals to temporally adapt their consumption in order to avoid overload situations.

The big data challenges in this use cases arises due to the high data volumes and velocity of the data. In the PEC pilot, every household produces about 18 sensor measurements per second. A full rollout in the pilot city would result in about 360,000 new measurements every second. Continuously inferring an accurate live state of the grid poses challenges to the throughput and latency of the analytics system. The addressed analytics for power quality and consumption analysis are characterized by incremental updates as well as temporal aggregates. Specifically for such a setting, stream processing and CEP technologies provide an answer to these challenges. Inferring the live state only requires computations over latest sensor information. Thus, the state that is needed for processing is relatively small. CEP engines keep the state in memory and thereby enable high throughput. We found that CEP engines are suitable to (a) support the query logic for live analysis power quality measures and power consumption and (b) to provide the required throughput. Using the open source CEP engine Esper [19] we could run the required analytics for thou-sands of households in parallel on a single machine. The performance depends on implementation details of the specific analysis. However, the processing paradigm of CEP significantly eased the development of high throughput analytics over the pilot data.

5.3 Forecasting Energy Demand

The third use case – forecasting energy demand – is an important element in demand side management solutions that are under investigation within the PEC pilot. Demand side management takes the approach to balance the grid, not only by adapting the production but also the consumption. Being able to predict the consumption on a household level allows taking proactive measures to influence demand, e.g. by sending demand response signals that ask consumers to reduce load [44]. In the context of the PEC project we are investigating mechanisms to continuously predict load on household level. The underlying concept is to (a) build a prediction model based on recorded sensor data and (b) to apply the model in real-time based on using the latest sensor measurements.

The big data challenge in this use case is twofold. The first challenge arises for building prediction models based on high volumes of sensor records. The second challenge is to apply these models in the stream, using high velocity sensor measurements.

To build prediction models, a large spectrum of candidate algorithms may be applied and tuned to predict household specific electricity demand. For instance, in [66] we used support vector machines and neural networks. In experiments we observed error reductions in load predictions between compared to persistence predictors of up to 33% (see [66] for details). A straight forward way to implement the model learning in a scalable way is to scale out by partitioning data and processing along households. This approach works to learn arbitrary prediction models in batch mode. To apply the model in real-time, one needs to continuously extract the model features from the incoming energy measurements and call the learned models. Partitioning can be done along households and supported by frameworks for stream processing. In [65] we describe an instantiation of the concept based on a combination of Esper [19] and Weka [62]. With this approach we are able to make low latency real-time forecast for 1000 households on a single machine (see [65] for details).

Suitable technologies exist for the challenges of model learning as well as real-time application of the prediction models. However, no off-the-self solutions to our knowledge directly meet the twofold challenges of this use case. Instead, a combination of big data technologies in required. We discuss such a combination in the following section.

5.4 Lessons Learned

Throughout the pilot project PEC we are gaining first-hand experience into operational aspects of IoT applications with big data. These experiences underpin many of the considerations described in the preceding sections. Specifically, we discuss (1) experience with practical use of Hadoop, (2) implications of using relational schemas, (3) operational challenges in an uncontrolled environment for sensor deployment, (4) and the challenge of combining batch and stream processing.

Using Hadoop Eased Development. By using the Hadoop framework, we found that the benefits of out-of-the box scalability materialized very early in the project. Even for rather simple analytics and after only a few month of data collection the development team struggled to make the corresponding queries scale sufficiently on relational databases. However, using Hadoop it was straight forward to achieve sufficient scalability and performance, without the need tune the implementation. This does not mean that solutions based on relational databases could not have achieved the required performance and scalability. Yet, the burden for the development teams was significantly lower using Hadoop.

Denormalization Helped with Operational Challenges. Regarding (2), the use of relational schemas, we found that denormalization helps with several operational challenges when handling time series of sensor data. Storing each sensors value with all related metadata (e.g. deployment location) makes it trivial to keep the metadata consistent with the measurement. This is especially helpful in an evolving system environment. Throughout the project we found that initially assumed functional de-

pendencies between sensors and metadata entries did not hold anymore as the system use cases expanded. For instance, we initially assumed that sensor deployments reside within one household. However, the need to move the same hardware to multiple locations arose later in the project. By storing the metadata along with the sensor values, it is trivial to ensure that each sensor recording can always be analysed in the context of the information data was correct during measurement time.

Uncontrolled Environments Require Robust Analytics. Regarding (3), operational challenges for sensor deployments in an uncontrolled environment, we found that this factor has major implications on developing analytics. The part of the system that is outside the realm of control is naturally exposed to unavoidable distortions and externally induced disruptions. This challenge arises in most IoT scenario and is therefore typical for this domain. In the PEC project the sensors are deployed in private households. Distortions due to power outages, accidental disconnection of the sensors, or simply mishandling of the system are among incidents that must be expected. It is therefore that analytics solution must cope with some degree of errors and uncertainty in the input data.

Missing best practices for combining batch and stream processing. Regarding (4), the challenge of combining batch and stream processing, we found that the application of existing big data technologies is straight forward when doing batch and stream processing in isolation. Both worlds have matured tool chains that work to a large degree out of the box. However, the design space for combining batch and stream processing is more open and best practices are less explored. While adapters for data exchange exist, the details of the interplay between batch and stream processing leaves significant effort to development. For instance, we found that the need for batch driven model learning and stream driven application of the models reoccurs in many use cases. However, the most suitable technologies for these tasks do not provide off-the-shelf support for this integrated scenario.

6 Unified Big Data Processing

As a first step towards addressing the Volume and Velocity challenge in the context of Smart Cities, we devised an Analytical Stream Processing framework for handling large quantities of IoT related data. It extends the basic Lambda architecture by supporting statistical and machine based model learning in batches and its use in the streaming layer.

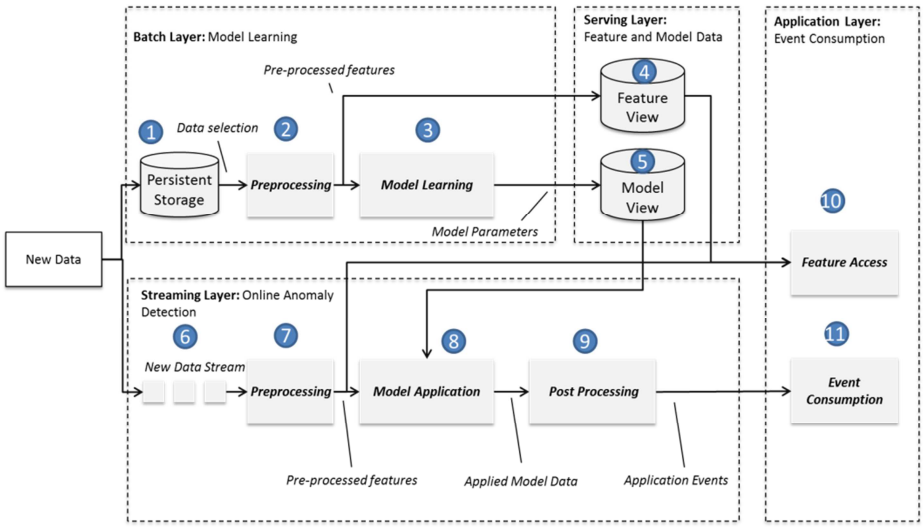

Fig. 4. Big Data Analytical Stream Processing Framework

Fig. 4 shows an initial draft of our framework. New data is being dispatched both to batch layer and stream processing layer. The main responsibility of the batch layer is to calculate models that characterize the incoming data, e.g. by describing re-occurring patterns creating a prediction model. For instance, the batch layer can realize the learning of models for households specific load prediction in the PEC project. The model is provided in the serving layer. Further to the basic lambda architecture, it is important to note that the serving layer must also serve the streaming layer that needs the model data in order to provide its analytical results to the application layer. Online load prediction in the PEC project is an example for such a situation. Here the speed layer extracts features (i.e. temporal aggregates) from the incoming load measurements and calls the previously learned prediction models to obtain a prediction value.

6.1 Model Learning in the Batch Layer

In the batch layer new data is collected in persistent storage such as the Hadoop File System (HDFS) or a NoSQL database (1). The model learning process consists of two main steps as commonly used by machine learning algorithms. They are executed in regular time intervals in order to adapt to changing patterns in the data. These three steps are pre-processing (2) and model learning (3). We briefly describe these processing steps and outline how they can be applied to large data sets.

Pre-Processing. In the pre-processing step raw data is processed in order to obtain a list of features that is suitable for applying the respective model learning algorithms. For raw sensor data this typically includes further sub-stages such as a sampling stage, data cleaning, feature extraction and noise filtering. The sampling stage outputs

a list of equally sampled measurements. This is particularly important for event-based sources. Data cleaning may include removal or substitution of erroneous sensor data. The feature extraction stage outputs a time series of one or multiple features that provide a suitable input to the model learning step. A feature can be any value that is calculated out of the raw sensor measurements including the raw measurement itself or applying a function over several types of raw measurements. Finally the noise filtering stage smoothens the extracted features, e.g. by applying a moving average or median filter.

Model Learning. In the model learning step the extracted features are used to calculate the actual model. The model parameters representing the model depend on the used algorithms. In our use case above this would include temporal aggregates of household and device specific load measurements. For a statistical model the model parameters would represent the parameters of a statistical distribution function such as the mean and standard deviation for a normal distribution. The model would also include thresholds on the variance based on which an anomaly is considered to be detected.

Scalable Processing. In the context of processing large amounts of data it is important to parallelize the processing steps discussed above. The realization of the parallelization depends to a large degree on the actual data and algorithms used. Due to the simplicity of the MapReduce framework it is beneficial to describe the batch layer processing in terms of map and reduce steps.

In order to apply the MapReduce framework to a model learning problem, we first need to consider whether the problem is embarrassingly parallel, i.e. whether the input data can be split into independent parts for which the algorithms can be applied. If there is a large number of spatial or temporal groups of data records for which a model needs to be learned, applying the MapReduce framework is an obvious procedure. For instance for the load prediction use case discussed above, we can create our load model for each household independently mapping individual measurement to a household and calculating the model for each household in the reduce step.

In other cases it may be necessary to rewrite the algorithm in a distributed way. Chu et al. describe how a special class of machine learning algorithms can be rewritten so that their execution can be sped up by using the MapReduce framework [9]. For more complex problems, it may be necessary to redesign the algorithm, potentially sacrificing optimal solutions or use other parallel programming frameworks such as Apache Hama [53] that provides an implementation of Bulk Synchronous Processing technique.

6.2 Viewing data patterns in the Serving Layer

Similar to the basic Lambda architecture the serving layer provides a view of the output data generated by the batch jobs. In the context of our analytical streaming

framework such batch views includes a feature view (4) that contains the pre-processed feature values prepared for fast access by applications.

The model view (5) is an important specialization of the serving layer. It contains the model parameters as calculated in the model learning step. The size of the model view is typically considerably smaller than the original data sets. For instance, the size of a statistical model that characterizes load distributions of individual households aggregates data over the period that is used for model learning.

In contrast to the basic Lambda architecture, the model views are primarily used by the speed layer and not the application layer. Different design options exist for this integration. Two fundamental options are to (a) leave the model in the serving layer and (b) to load the model into the speed layer. In both options the speed layer extracts model inputs from the live data stream and calls the model. In option (a), the speed layer sends the extracted model input to the serving layer and gets the result of the model application in return. This option has the advantage that the model must not be managed in the memory of the speed layer. The drawback is that calls to the serving layer may increase latency and reduce throughput. In option (b), the model is loaded into the memory of the speed layer one the corresponding streaming logic is instantiated this option has the advantage that the model application avoids the overhead of making external calls but – dependent on the model size – can also cause resource problems with respect to the main memory.

6.3 In-Stream Analytics in the Streaming Layer

The streaming layer receives the same stream of new data (6) as the batch layer. Its main purpose is to analyze the incoming data stream in real-time. As a first step the data is pre-processed (7) in order to receive the features out of the raw data stream. This calculation is functionally equivalent to the pre-processing step in the batch layer (2). Thus it may be beneficial to re-use the logic in the batch layer, in particular if complex processing over time series data is being performed. In this case, the query languages offered by complex event processing engines may often be better suited to formulate the logic than a hand-crafted code or even SQL [20]. This can for instance be achieved by invoking the CEP queries in the reduce step of a MapReduce job [40].

In the next step the model in the serving layer is accessed in order to apply the model on the pre-processed features calculated in the previous step (8). In the case of statistical anomaly detection, this could for instance involve computing the deviation from the expected value as stored in the model for the matching time period and throwing an event if the threshold is exceeded. For instance, to detect power quality anomalies one may compare current voltage fluctuations against a statistical model of typically observed fluctuations.

As the processing in the speed layer is often time critical, it may be necessary to avoid further latencies introduced by accessing the stored model in the serving layer. This could for instance be achieved by providing an event source in the streaming layer that accesses the corresponding parts of the model or directly loading the model into the CEP engine and manually updating it. However, if the model is large and

requires complex update strategy it is more beneficial to query an in memory database, e.g. using Redis [48].

The final step in the speed layer is the post-processing of applied model data (8). This step may be necessary to reduce the number of false positives. In the case of anomaly detection for instance, it is often not desired to propagate a singular anomaly event that may result from measurement errors or other inaccuracies in the data or model. Thus, an anomaly may only be indicated if it is consecutively detected over a certain time.

Scalable Processing. In order to cope with a large number of events it is necessary to scale out the processing logic to multiple machines. This can for instance be achieved by creating a Storm [56] topology that defines how stream based data is being processed. The partitioning of data can then be defined in a similar was in the batch processing layer. For instance if the data can easily spatially be separated it is possible to use Storm's field grouping capability to ensure that data of a spatial partition is always processed by the same task. The processing logic itself can then be implemented using modern complex event processing engines such as Esper [19] benefiting from their performance and query languages.

6.4 Accessing the data in the application Layer

As in the basic Lambda architecture the application layer is accessing the data both from the serving and speed layer. The application uses the data from the batch and streaming layer in two ways. First, the streaming data can be used to provide a real time view of the features calculated in the batch layer (10). As in the basic lambda architecture this can be useful to provide a real-time dashboard view, for instance reflecting the current state of the power grid. Second, the application may only consume events generated in the stream layer that are based on applying incoming data to the model as described above (11). This way applications and human operators can receive events about detected anomalies or continuous predictions about energy consumption.

7 Further Research Directions

For further work we plan to extend our analytical framework and apply it to other domains in the context of Smart Cities. This includes in particular adding more machine learning algorithms for the batch layer and the corresponding logic for the streaming layer.

In this chapter we have focused on an analytical framework for processing large volumes of data in real-time, i.e. we addressed mainly the volume and velocity challenge. As we extend our work to different data sets and application domains, it will however be increasingly important to cope with the variety of data sources and their data.

It is therefore a key requirement for the proposed analytical framework that both the model learning algorithms as well as the stream logic can be applied uniformly across different data sets and application domains. On one hand this will maximize the value that can be extracted from available data sets and on the other hand the processing chain can then easily be applied to new data sets, thus saving effort, time and costs during the development process.

A future research direction is therefore to extend the analytical framework with the necessary mechanisms to achieve such uniform processing. This could for instance be realized by a meta data model on which the corresponding logic operates. As we see the application of this analytical framework mainly in the context of Smart Cities and the Internet of Things, an entity-based framework that naturally models real-world entities such as sensors, people, buildings, etc. [23]. Such an information model along with a corresponding architecture has been defined by the IoT-A project [5].

8 Conclusions

In this chapter we have proposed an initial draft of a Big Data analytical framework for IoT and Smart City applications. The framework is based on existing state-of-the art, initial findings from our participation in the publicly funded projects BIG [7] and PEC [46] as well as our own experiences with analytical applications for Smart Cities. Our work is motivated by the fact that key components such as data, sources, algorithms, IoT architectures and Big Data technologies are available today, but still there is a lot of effort required to put them into operational value.

A significant part of this effort is due to missing standards (cf. for instance the SQL query language for relational databases) and wide variety of different technologies in the Big Data domain as well as the required integration effort. But the application of advanced analytical computation at scale and speed also requires considerable design effort and experience. We believe that an analytical Big Data framework along with appropriate toolboxes can add significant value both to required development effort and insights that can be derived from the data. While there are Big Data machine learning libraries such as Mahout [42], as well as frameworks for model learning on top of Hadoop [48], we are not aware of fully integrated analytical frameworks that combine model learning and stream processing.

In order to fully benefit from the framework its overall design and associated toolboxes need to support a variety of data sources and algorithms. While this requirement does not change the high level architecture of the framework, such extensions do have significant impact on the interface level and overall design of the individual processing components. A carefully planned and sound conceptual design as well as pragmatic implementation decisions will be an important enabler to reduce development costs and create innovative services in the IoT and Smart City domain.

Acknowledgements. This work has partly been funded by the EU funded project Big Data Public Private Forum (BIG), grant agreement number 318062 and by the Peer Energy Cloud project which is part of the Trusted Cloud Program funded by the Ger-

man Federal Ministry of Economics and Technology. We would also like to thank Max Walther, who implemented the batch-driven power quality analytics MapReduce jobs as well as Alexander Bauer and Melanie Hartmann who supported us with the design of the analytical models. The original publication is available at www.springerlink.com.

References

1. Abadi, D. J., et al.: The Design of the Borealis Stream Processing Engine. In: CIDR, vol. 5, pp. 277-289 (2005)
2. L. Atzori, A. Iera, and G. Morabito.: The internet of things: A survey. Computer Networks, vol. 54, no. 15, pp. 2787 – 2805 (2010).
3. Barga, R. S., et al.: Consistent streaming through time: A vision for event stream processing. arXiv preprint cs/0612115 (2006).
4. Batty, M., "Smart Cities and Big Data", http://www.spatialcomplexity.info/.
5. Bauer, M., Bui, N., Giacomin, P., Gruschka, N., Haller, S., Ho, E., Kernchen, R., Lischka, M., Loof, J.D., Magerkurth, C. Meissner, S., Meyer, S., Nettsträter, A., Lacalle, F.O., Segura, A.S., Serbanati, A., Strohbach, M., Toubiana, V., Walewski, J. W.: IoT-A Project Deliverable D1.2 – Initial Architectural Reference Model for IoT. (2011). Available at http://www.iot-a.eu/public/public-documents/d1.2/view, last accessed 18/09/2013.
6. Bifet, A., Frank, E.: Sentiment knowledge discovery in twitter streaming data. In Discovery Science , pages 1-15. Springer, (2010).
7. BIG Project Website, http://www.big-project.eu/, last accessed 19/09/2013.
8. Chandrasekaran, S., et al.: TelegraphCQ: continuous dataflow processing. In: ACM SIGMOD international conference on Management of data, pp. 668-668. ACM (2003)
9. Chu, C.-T., Kim, S. K., Lin, Y.A., Yu, Y.Y., Bradski, G., Ng, A.Y., Olukotun, K.: Map-Reduce for Machine Learning on Multicore. In: Schölkopf, B., Platt, J.C., Hoffman, T. (eds.), Advances in Neural Information Processing Systems 19 (NIPS 2006), pp. 281-288. MIT Press, Cambridge, MA (2007).
10. Correia, Z.P.: Toward a Stakeholder Model for the Co-Production of the Public Sector Information System. Information Research. 10(3), paper 228 (2005). Available at http://InformationR.net/ir/10-3/paper228.html, last accessed 27/02/2013.
11. DataMarket, http://datamarket.com/, last accessed 21/09/2013.
12. Data.gov, http://www.data.gov/, last accessed 21/09/2013.
13. Davis, J.R., Clodoveu A., et al.: Inferring the Location of Twitter Messages based on User Relationships. Transactions in GIS, vol. 15, no. 6: pp. 735-751 (2011).
14. Dean, J., Ghemawat, S.: MapReduce: Simplified Data Processing on Large Clusters. Communications of the ACM. 51(1), 1–13 (2008). doi:10.1145/1327452.1327492.
15. Directive 2003/98/EC of the European Parliament and of the Council of 17 November 2003 on the re-use of public sector information, available at http://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=CONSLEG:2003L0098:20130717:EN:PDF, last accessed 13/01/2014.,
16. M. Dohler, "Machine-to-Machine Technologies, Applications & Markets", 27th IEEE International Conference on Advanced Information Networking and Applications (AINA) (2013).
17. Dredze, M., Paul, M. J., Bergsma, S., Tran, H.: Carmen: A Twitter Geolocation System with Applications to Public Health. (2013).
18. The Economist, "Running out of road", November 2006.

19. EsperTech, http://esper.codehaus.org, last accessed 22/09/2013.
20. Etzion, O.: On Off-Line Event Processing (2009). Event Processing Thinking Online Blog. Available at http://epthinking.blogspot.de/2009/02/on-off-line-event-processing.html, last accessed 17/09/2013.
21. European Open Data Portal, http://open-data.europa.eu/, last accessed 21/09/2013.
22. Farroukh, A., Sadoghi, M., Jacobsen, H-A.: Towards vulnerability-based intrusion detection with event processing. In: 5th ACM international conference on Distributed event-based system, pp. 171-182 ACM (2011).
23. Gazis, V. , Strohbach, M., Akiva, N., Walther, M.: A Unified View on Data Path Aspects for Sensing Applications at a Smart City Scale. In: IEEE 27th International Conference on Advanced Information Networking and Applications Workshops (WAINA2013), pp. 1283-1288. IEEE Computer Society, Barcelona (2013). doi:10.1109/WAINA.2013.66.
24. Gedik, B., Andrade, H., Wu, K. L., Yu, P. S., Doo, M.: SPADE: the system s declarative stream processing engine. In: ACM SIGMOD international conference on Management of data, pp. 1123-1134, ACM (2008).
25. Giraph Project, http://giraph.apache.org/, last accessed 21/09/2013.
26. Gubbi, J., Buyya, R., Marusic, S. and Palaniswami, M., "Internet of things (IoT): A vision, architectural elements, and future directions", Future Generation Computer Systems, 2013.
27. Hinze, A., Sachs, K., & Buchmann, A.: Event-based applications and enabling technologies. In: Third ACM International Conference on Distributed Event-Based Systems (2009)
28. INFSO D.4 Networked Enterprise & RFID INFSO G.2 Micro & Nanosystems, "Internet of Things in 2020 – A roadmap for the Future", September 2008, report available at http://www.smart-systems-integration.org/public/internet-of-things.
29. ITU, "The Internet of Things," 2005.
30. Fidler, E., Jacobsen, H. A., Li, G., Mankovski, S.: The PADRES Distributed Publish/Subscribe System. In: FIW, pp. 12-30 (2005).
31. van Kasteren, T., Ravkin, H., Strohbach, M., Lischka, M., Tinte, M., Pariente, T., Becker, T., Ngonga, A., Lyko, K., Hellmann, S., Morsey, M., Frischmuth, P., Ermilov, I., Martin, M., Zaveri, A., Capadisli, S., Curry., E., Freitas, A., Rakhmawati, N.A., ul Hassan, U., Iqbal, A.: BIG Project Deliverable D2.2.1 – First Draft of Technical White Papers (2013). Available at http://big-project.eu/deliverables, last accessed 19/09/2013.
32. Laney, D. 3D Data Management: Controlling Data Volume, Velocity and Variety. Meta Group Research Report, 2001, available at http://blogs.gartner.com/doug-laney/files/2012/01/ad949-3D-Data-Management-Controlling-Data-Volume-Velocity-and-Variety.pdf, last accessed 21/09/2013.
33. Leeds, D., J., THE SOFT GRID 2013-2020: Big Data & Utility Analytics for Smart Grid, GTM Research Report, 2012, available at http://www.greentechmedia.com/research/report/the-soft-grid-2013, last accessed 21/09/2013.
34. Liu, B.: Sentiment analysis and opinion mining. In Synthesis Lectures on Human Language Technologies, 5(1):1-167. (2012).
35. The London Dahsboard, http://data.london.gov.uk/london-dashboard, last accessed 21/09/2013.
36. Madden, S.R., Franklin, M.J., Hellerstein, J.M., Hong, W.: TinyDB: An acquisitional query processing system for sensor networks. .ACM Transactions on Database Systems , vol. 30, pp. 122-173, (2005).
37. Manyika, J., Chui, M., Brown, B., Bughin, J., Dobbs, R., Roxburgh, C., Hung Byers, A.: Big data: The next frontier for innovation, competition, and productivity. McKinsey Global Institute. Available at http://www.mckinsey.com/insights/business_technology/big_data_the_next_frontier_for_innovation, last accessed 20/08/2013.

38. Marz, N., Warren, J.: A new paradigm for Big Data. In Big Data – Principles and best practices of scalable real-time data systems, to appear, Chapter 1, Manning Publications Co. Available at http://www.manning.com/marz/, ISBN 9781617290343, last accessed 16/08/2013.
39. Miorandi, S. Sicari, F. D. Pellegrini, and I. Chlamtac, "Internet of things: Vision, applications and research challenges," Ad Hoc Networks, vol. 10, no. 7, pp. 1497 – 1516, 2012.
40. Microsoft BI Team, Big Data, Hadoop and StreamInsight™, available at http://blogs.msdn.com/b/microsoft_business_intelligence1/archive/2012/02/22/big-data-hadoop-and-streaminsight.aspx, last accessed 09/09/2013.
41. Rajeev, M., Widom, J., Arasu, A., Babcock, B., Babu, S., Datar, M., Manku, G., Olston, C., Rosenstein, J., Varma, R.: Query processing, approximation, and resource management in a data stream management system. In: CIDR Conference, pp. 1-16 (2002).
42. Owen, S., Anil, R. Dunning, T., and Ellen Friedman, Mahout in Action. Manning Publications Co., ISBN 9781935182689 (2011).
43. Neubauer, P.: Neo4j and some graph problems. Available at http://www.slideshare .net/peterneubauer/neo4j-5-cool-graph-examples-4473985?from_search=2, last accessed 21/09/2013.
44. Palensky, P., Dietrich, D.: Demand side management: Demand response, intelligent energy systems, and smart loads. Transactions on Industrial Informatics, IEEE, vol. 7(3), pp 381-388, (2011) .
45. Pang, B. and Lee, L.: Opinion mining and sentiment analysis. Foundations and trends in information retrieval, 2(1-2), pp.1-135 (2008).
46. Peer Energy Cloud project website, http://www.peerenergycloud.de/, last accessed 19/09/2013.
47. PredPol, http://www.predpol.com/, 21/09/2013.
48. Radoop, http://www.radoop.eu/, last accessed 16/12/2013.
49. Redis Project, http://redis.io/topics/faq, last accessed 17/09/2013.
50. ruleCore, http://www.rulecore.com, last accessed 22/09/2013.
51. S4 Project, http://incubator.apache.org/s4/, last accessed 16/08/2013.
52. Salton, G. and Buckley, C.: Term-Weighting Approaches in Automatic Text Retrieval. Information Processing and Management, vol. 24, no. 5, pp. 513-523 (1988).
53. Seo, S., Yoon, E.J., Kim, J., Jin, S., Kim, J.-S., Maeng, S.: HAMA: An Efficient Matrix Computation with the MapReduce Framework. In: 2nd IEEE International Conference on Cloud Computing Technology and Science (CloudCom 2010), pp. 721-726, IEEE Computer Society (2013).
54. Shigeru, O.: M2M and Big Data to Realize the Smart City. NEC Technical Journal, vol. 7, no. 2 (2012).
55. Siddhi CEP - The Complex Event Processing Engine, http://siddhi.sourceforge.net, last accessed 22/09/2013.
56. Storm Project, http://storm-project.net/, last accessed 16/08/2013.
57. Stonebraker, M.: What Does 'Big Data' Mean? (Part 3). BLOG@ACM (2012). Available at http://cacm.acm.org/blogs/blog-cacm/157589-what-does-big-data-mean-part-3/fulltext, last accessed 16/08/2013.
58. United Nations, World Urbanization Prospects 2011 Revision (2011).
59. US National Intelligence Council.: Disruptive Civil Technologies: Six Technologies with Potential Impacts on US Interests out to 2025. http://www.fas.org/irp/nic/disruptive.pdf, last accessed 20/12/2013.

60. Wang, F. et al.: Bridging physical and virtual worlds: complex event processing for RFID data streams. Advances in Database Technology-EDBT 2006, Springer Berlin Heidelberg, pp. 588-607 (2006).
61. Weidlich, M., Ziekow, H., Mendling, J., Günther, O., Weske, M., Desai, N.: Event-based monitoring of process execution violations. In: Business Process Management, pp. 182-198. Springer, Berlin Heidelberg, (2011)
62. Hall M., Frank E., Holmes G., Pfahringer B., Reutemann P., Witten I. H.: The WEKA Data Mining Software: An Update. SIGKDD Explorations, vol. 11, pp. 10-18 (2009).
63. White, T.: Hadoop: The Definitive Guide, O'Reilly (2012).
64. Xively, https://xively.com/, last accessed 21/09/2013.
65. Ziekow H., Doblander C., Goebel C., Jacobsen H.-A.: Forecasting Household Electricity Demand with Complex Event Processing: Insights from a Prototypical Solution. Middleware Conference, Beijing, China (2013).
66. Ziekow H., Goebel C., Strüker J., Jacobsen H.-A.: The Potential of Smart Home Sensors in Forecasting Household Electricity Demand. IEEE International Conference on Smart Grid Communications (SmartGridComm2013), Vancouver, Canada (2013).
67. Zillner, S., Rusitschka, S., Munné, R., Lippell, H., Lobillo Vilela, F., Hussain, K., Becker., T., Jung., R., Paradowski, D., Huang, Y.: BIG Project Deliverable D2.3.1 – First Draft of Sector's Requisites (2013). Available at http://big-project.eu/deliverables, last accessed 19/09/2013.
68. European Smart Cities project, available at http://www.smart-cities.eu/model.html.
69. Smart City Week 2012 international conference and exhibition report, Oct. 29 – Nov. 2 available at http://scw.nikkeibp.co.jp/2013/docs/SCW2012_Conference_Report_vol3.pdf.

Index of Terms

Glossary of Terms

Batch layer. Refer to the corresponding processing layer of the *Lambda architecture* [38] that processes data in batches.

Batch processing. Refers to processing methods that collect data over a certain amount of time and then process several data sets as a whole.

Big Data. We use the term *Big Data* both to refer to ”datasets whose size is beyond the ability of typical database software tools to capture, store, manage, and analyze” [37].

Big Data analytics framework. An architectural reference model including architectural principles and a set of tools suitable for analytical challenges that require model learning and near real-time decision making based on large data sets.

In-Stream processing. Refers to methods that process data continuously (cf *batch processing*).

CEP. *See* Complex Event Processing.

Complex Event Processing. Refers to processing methods that are tailored for application of logic over continuous data flows. Complex event processing derives complex events from more simple events.

ITS. *See* Intelligent Transportation Systems.

Intelligent Transportation Systems. A distributed architecture based on ICT components (e.g., sensors, actuators, servers, etc.) and M2M communications that is used to manage and control a transportation infrastructure.

Internet of Things. Vangelis?

IoT. *See* Internet of Things.

Internet of Things.

Lambda architecture. A high level architecture for Big Data systems defined by Marz and Warren [38].

M2M. *See* Machine-to-machine communication.

Machine-to-machine communication. A communication capacity whereby a set of distributed components (e.g., sensors, actuators, servers, etc.) communicate to support a computational process in realizing its established policy and where the process itself has no human participation (i.e., in terms of neither a provider of inputs to the process nor a consumer of outputs of the process).

Model learning. Summarizes algorithms that characterize large data sets by considerably smaller data sets.

noSQL. Refers to data storage systems that use non-relational data models.

REST. *See* Representational State Transfer.

Representational State Transfer. An application model whereby behavior is structured on the basis of elementary operations (e.g., create, update, delete, etc.) upon distinct resources whose identity is represented in a URL notation.

SCL. *See* Service Capability Layer.

Service Capability Layer. A set of services offered to applications utilizing M2M capacities that is organized through a layer of software (e.g., on top of middleware platform).

SIM. *See* Subscriber Identity Module.

Subscriber Identity Module. An embedded component that provides cryptographic mechanisms (e.g., authentication, authorization, etc.) of accessing the identity information of a subscriber to a service (e.g., a cellular mobile data service) and the profile information associated to that subscription.

SQL. Standard Query Language. Refers to the standardized query language for relational database management systems.

Stream processing. Refers to technologies that are tailored for applying processing logic over continuous streams of linearly ordered data with a given schema.

Streaming layer. Refers to the corresponding processing layer of the *Lambda architecture* [38] that uses *stream processing* for processing data continuously.