Information Extraction, Real-Time Processing and DW2.0 in Operational Business Intelligence

Malu Castellanos, Umeshwar Dayal, Song Wang, and Gupta Chetan

Hewlett-Packard Laboratories
Palo Alto, CA, USA
firsname.lastname@hp.com

Abstract. In today's enterprise, business processes and business intelligence applications need to access and use structured and unstructured information to extend business transactions and analytics with as much adjacent data as possible. Unfortunately, all this information is scattered in many places, in many forms; managed by different database systems, document management systems, and file systems. Companies end up having to build one-of-a-kind solutions to integrate these disparate systems and make the right information available at the right time and in the right form for their business transactions and analytical applications. Our goal is to create an operational business intelligence platform that manages all the information required by business transactions and combines facts extracted from unstructured sources with data coming from structured sources along the DW2.0 pipeline to enable actionable insights. In this paper, we give an overview of the platform functionality and architecture focusing in particular in the information extraction and analytics layers and their application to situational awareness for epidemics medical response.

1 Introduction

Today, organizations use relational DBMSs to manage (capture, store, index, search, process) structured data needed for on-line transaction processing (OLTP)[1] applications. However, typical business transactions (billing, order processing, accounts payable, claims processing, loan processing, etc.) need both structured and unstructured information (contracts, invoices, purchase orders, insurance claim forms, loan documents, etc.). This information is scattered in many places and managed by different database, document management, and file systems [1]. This makes it difficult to find all the information relevant to a business transaction, leading to increased transaction cost or loss of revenue. A good example is hospital billing (Fig. 1), where massive amounts of information (i.e., physician notes, file records, forms) are generated by different processes (admission, lab tests, surgery, etc.) during a patient's hospital stay. This immense amount of unstructured data spread across different entities needs to be captured, sorted, reconciled, codified and integrated with other structured data to process billing transactions. Unfortunately, the lack of a platform to extract structured data from these unstructured documents and integrate it with other structured data

[1] Also for on-line analytics processing (OLAP) but it is not the focus of this paper.

S. Kikuchi, S. Sachdeva, and S. Bhalla (Eds.): DNIS 2010, LNCS 5999, pp. 33–45, 2010.

causes errors in the billing process. These errors result in a 30% revenue loss according to an internal reliable source working with a major hospital[2]. Another example is loan processing, where the inability to make all of the relevant information available to business transactions has led to mismanagement of the whole process. In these cases, companies have to build one-of-a-kind solutions to integrate disparate systems and make the right information available at the right time in the right form for business users.

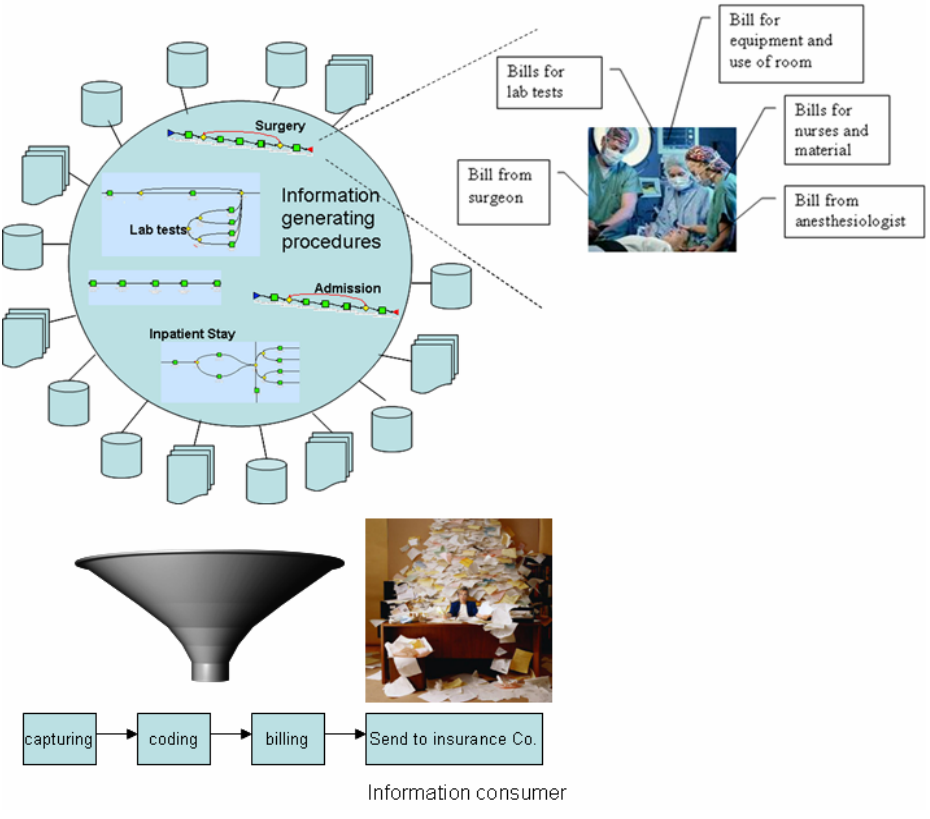

Fig. 1. Hospital billing operational environment

The problem described before is exacerbated by today's competitive and highly dynamic environment where analyzing data to understand how the business is performing, to predict outcomes and trends, to improve the effectiveness of business processes and to detect and understand the impact of external events on the internal business operations has become essential. In fact, the next wave of competitive differentiation will be based on the ability to harness unprecedented amounts of data into actionable information. Analytics will give data-driven companies a powerful

[2] Due to confidentiality we cannot give the name of the hospital.

advantage in customer service, new product development, risk profiling, real-time pricing, pattern recognition, fraud detection and many other examples. The value of analytics is in enabling actions based on insights ("actionable insights") derived from information extracted from all sorts of data sources, that is, structured or unstructured and internal or external. Moreover, the trend towards operational BI where the speed to insight is critical to improve the effectiveness of business operations with minimal latency, is forcing companies to analyze structured and unstructured information as soon as it is generated, that is, as data streams. A good example is the case of epidemics awareness for medical response where hospitals analyze incident reports of incoming patients and news feeds about the onset of epidemics and compare them to determine the effect that the epidemics event is having on the hospital. This allows the hospital to be prepared to deal with the epidemics by ordering supplies, scheduling resources, etc. Once more, the lack of a platform that supports the required functionality, (analytics in this case) on the combination of extracted information forces companies to adopt ad-hoc solutions that are rather difficult to build, very costly, limited and often not even satisfactory.

The combination of streaming data with stored data on one dimension and structured and unstructured data in another dimension gives an enormous competitive advantage to companies that have the technology to exploit this information in a timely manner.

Our goal is to create an operational business intelligence platform for all data required by business transactions and BI applications. This is done in response to the need that organizations have to adopt technology that enables them to a) incorporate unstructured data into their business transactions, b) to gain "real-time" actionable insight derived from information that is extracted from structured and unstructured data sources, and c) doing all this in an integrated, scalable and robust way.

The rest of the paper is structured as follows: Section 2 presents the layered architecture of our operational BI platform. Section 3 describes the role of information extraction in DW 2.0, the information extraction pipeline and its fit into textual-ETL that is inherent to the DW 2.0 architecture. Section 4 illustrates the operation of the IE and analytics layers and their interaction with a concrete application to contractual situational awareness where real-time processing is of essence. Section 5 gives a brief overview of related work and Section 6 presents our conclusions.

2 Platform Architecture

Our operational business intelligence platform has two main functions:

(a) It integrates structured and unstructured data to extend the scope of business transactions with adjacent data, for a better support of business processes' underlying business operations.

(b) It integrates structured and unstructured data to extend the scope of BI applications and analytics with adjacent data to give the users a 360^{o} data visibility.

The components of the platform are shown in Figure 2.

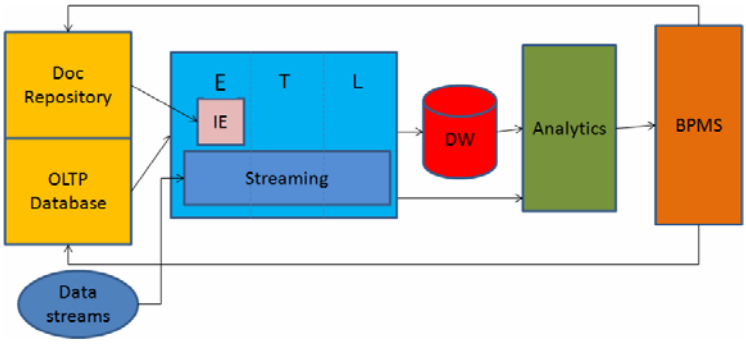

Fig. 2. Architectural components

OLTP DBMS - A massively parallel DBMS is often deployed in OLTP environments that require scalability and very high availability where its fault tolerant nature delivers the stability to guarantee that transactions are completed. The idea is to extend these benefits to all operational information –structured and unstructured - required by business transactions.

Document Repository - Unstructured data comes in many forms: scanned or electronic documents, email, audio, video, Web pages and many others. Our platform initially focuses on text-based documents, such as Word or PDF files. Text documents are sent to the platform repository along with any accompanying metadata. The metadata is stored in the search database and the document is parsed to populate the inverse index database that supports search functionality. All of the metadata about a document is stored in the DBMS tables and the actual document is treated as a Binary Large Object (BLOB), stored in the Large Object Storage subsystem.

Extract-Transform-Load (ETL) – The back-end of the architecture is a data integration pipeline for populating the data warehouse by extracting data from distributed and usually heterogeneous operational sources (OLTP database); cleansing, integrating and transforming the data; and loading it into the data warehouse. The integration pipeline supports batch and streaming processing as well as structured and unstructured data. The latter is known as textual ETL and is an essential component of the DW 2.0 architecture (see Section 3).

Information Extraction (IE) – It handles the "E" part of (textual) ETL of unstructured data sources. It provides the ability to automatically read through textual documents, retrieve business-relevant information (e.g., doctor's name, surgical procedure, symptoms), and present it in a structured fashion to populate the data warehouse so that searching and accessing this information is substantially simplified. Before extraction ever happens, a document model that reflects the structure of the documents and a domain model that specifies the items to be extracted need to be defined [2]. Also a subset of documents is manually tagged with the target information according to the domain model (the use of a GUI greatly simplifies this process). Once tagged, the documents go through a normalization process that may involve stemming, correcting misspellings, and eliminating stop words. Then, the documents are transformed into

different representations that suit the input requirements of specialized algorithms that use these documents to learn extraction models (see Section 3). Once models are learned they are applied to production documents to mine target data from text. Notice that this is the data used to extend the scope of business transactions and BI applications with structured data extracted from unstructured sources.

Datawarehouse (DW) – It is a repository designed to facilitate reporting and analysis through a variety of front-end querying, reporting and analytic sources. It typically consolidates data from operational databases but in our platform it also consolidates structured data extracted from unstructured sources. It is also implemented on a parallel DBMS (in fact, it might be the same DBMS that runs the OLTP Database and the Document Repository.

Analytics – They consume data from structured sources along with data that the IE component extracted from unstructured ones. Moreover, since data may come from internal and external sources and may take the form of stored or streaming data, this layer needs to incorporate analytic algorithms that are capable of simultaneously dealing with streams of varying speeds and static data. This enables users to get better insight in a timely manner. In section 4 we introduce one of our streaming analytics techniques.

BPMS Layer - The business process management system (BPMS) controls transactions, requests to the Document Repository and analytics. At different steps of a business process OLTP transactions are carried out and it is these transactions whose scope is augmented by the data extracted by the information extraction layer in such a way that unstructured data is processed as part of the normal business processes.

3 Information Extraction and DW 2.0

Data warehousing began in the 1980s as a way to reduce users' frustration with their inability to get integrated, reliable, accessible data. On-line applications had no appreciable amount of historical data because they jettisoned their historical data as quickly as possible in the name of high performance. Thus, corporations had lots of data and very little information. For 15 years, people built different manifestations of data warehouses where the focus was on structured data extracted from OLTP databases. Increasingly, enterprises have come to realize that for business intelligence applications supporting decision making, information derived from unstructured and semi-structured data sources is also needed. By some accounts, over 80% of an enterprise's information assets are unstructured, mainly in the form of text documents (contracts, warranties, forms, medical reports, insurance claims, policies, reports, customer support cases, etc.) and other data types such as images, video, audio, and other different forms of spatial and temporal data. This led to the evolution of the underlying architecture of the data warehouse into what Bill Inmon has called DW 2.0 [3]. The core idea of DW 2.0 is the integration of structured and unstructured data. Thus, one of the main differences between the first generation of data warehouses and DW 2.0 is that in the latter unstructured data is a valid part of the data warehouse. Unstructured data exists in various forms in DW 2.0: actual snippets, edited words

and phrases and extracted structured data. The latter being the interesting one here as described below.

The information extraction layer of our platform architecture serves for the purpose of extracting structured data from unstructured sources. The goal is to make this data available to BI applications and analytics and to OLTP applications that carry out business operations. For illustration, imagine a company that has a BI application that creates user profiles and makes purchase recommendations based on these profiles. To improve the accuracy of a profile, it may be important to consider not just the transactional data in the various operational databases involved in the ordering process, but also to look at the content of web pages that the user has browsed, his reviews or blog postings, and any contracts he may have with the company. This example makes evident that unstructured data provides valuable contextual information for more informed operational decision making.

To incorporate unstructured data into the DW architecture (i.e., DW 2.0), it is not sufficient to merely store the data as is (i.e., text) into the warehouse. Rather, it is important to extract useful information from the unstructured data and turn it into structured data that can be stored, accessed and analyzed efficiently along with other structured data. However, this is not an easy task. Take, for example, the text in customer reviews, which are written in an ad-hoc manner. The lack of structure makes it hard to find the product and the features referred to in the review, and especially which features the customer likes and which he doesn't like.

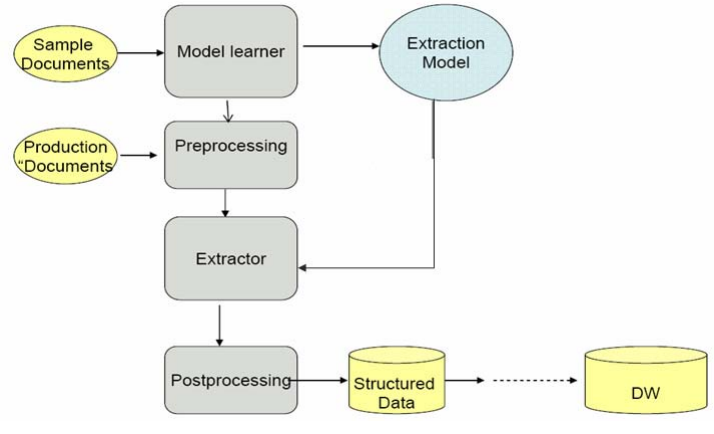

Fig. 3. Information extraction pipeline for unstructured data

Numerous *information extraction* (IE) techniques have been developed that try to learn models for the retrieval of relevant entities, relationships, facts, events, etc. from text data [4]. Some of the most popular techniques are based on rule learning [5], Hidden Markov Models [6] and more recently Conditional Random Fields [7]. We experimented with these techniques but did not obtain good results so we developed our own technique based on genetic algorithms (GA) [8] to learn the most relevant combinations of prefixes and suffixes of tagged instances of the role-based entity types of interest. To this end, a bag of terms is built from all the prefixes in the

context of the tagged entities in the training set. Another bag is built from their suffixes. For example, given the tagged sentence *symptoms appeared on* <expirationDate> *May 31, 2009*, </expirationDate> *after returning from a trip toMexico….* The terms "*symptoms*", "*appeared*", "*on*" are added to a bag of prefixes of the role-based entity type "SymptomsOnsetDate" whereas the terms "*after*", "*returning*", "*from*", "*trip*", "*Mexico*" are added to its bag of suffixes. The bags are then used to build individuals with *N* random prefixes and *M* random suffixes in the first generation and for injecting randomness in the off-springs in later generations. Since only the best individuals of each generation survive, the fitness of an individual is computed from the number of its terms (i.e., prefixes and suffixes) that match the context terms of the tagged instances. The best individual in a pre-determined number of iterations represents a context pattern given by its terms and is used to derive an extraction rule that recognizes entities of the corresponding type. The GA is run iteratively to obtain more extraction rules corresponding to other context patterns. The process ends after a given number of iterations or when the fitness of the new best individual is lower than a given threshold. The rules are validated against an unseen testing set and those with the highest accuracy (above a given threshold) constitute the final rule set for the given role-based entity type.

Figure 3 shows the pipeline of information extraction from text data sources that is built into our platform's IE layer with the goal of extracting relevant data from a collection of documents; e.g., contract number, customer, and expiration date from contracts. Whatever the information extraction algorithm used, the source data always needs to be pre-processed to get rid of noise, transform it to the representation required by the extraction method, etc before the actual extraction takes place. In addition, the output also needs to be post-processed to gather the structured data in the form of attribute-value pairs, which can then be transformed and loaded into the data warehouse. The pipeline of tasks to extract, transform and load data from unstructured sources into a data warehouse so that it can be integrated into the structured world corresponds to what Inmon has named *textual-ETL* [9]. The pipeline involves numerous operations from which those belonging to IE correspond to the 'E' (extract) part of ETL. The extracted data from the unstructured sources often relates to data in structured sources so it is staged into a landing area and is then loaded into the data warehouse, where the information from both structured and unstructured sources is consolidated (e.g., contract data is related to customer data) for use by BI applications.

The challenge in textual-ETL consists in identifying how to abstract all the above tasks into operators that can be used to design, optimize and execute these flows in the same way as for structured data.

Using a uniform approach for ETL of structured and unstructured data makes it possible to use the same data warehouse infrastructure. The question is whether there should be separate pipes for the structured and unstructured data in the ETL flows or only one. Having separate pipes (as shown in Figure 4a) is conceptually simpler to design. However, having a single pipe with two separate extraction stages but a single integrated transformation-load stage (as shown in Figure 4b) creates better opportunities for end-to-end optimization.

In the next section we present an application of our operational BI platform. Specifically, we describe how information extraction and analytics help to get actionable insight in a timely manner in the context of epidemics awareness for hospital response.

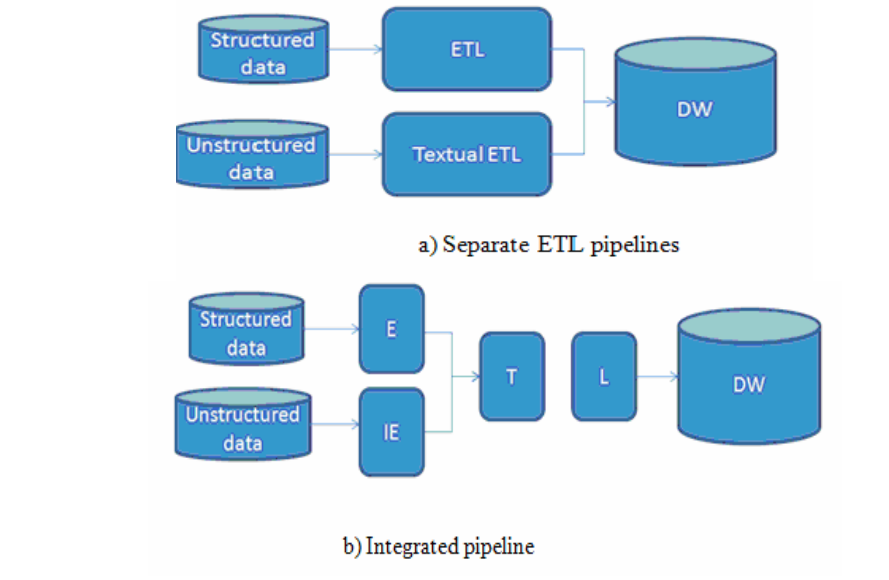

Fig. 4. ETL Pipeline for Structured and Unstructured Data

4 An Application: Situational Awareness

An important function of our operational BI platform is extracting and correlating different sources of unstructured data. Here we describe how correlating unstructured stored data with unstructured streaming data from the web enables situational awareness. We apply it to the medical domain where hospital needs to be alert about epidemics onsets and how they are affecting the patients arriving at the hospitals. This allows the hospitals to be prepared to deal with the epidemic by ordering medical supplies, scheduling resources, etc.

Data buried in incident reports of patients arriving at a hospital and in articles of news feeds provide valuable information that when extracted and correlated can provide actionable insight that can enable the hospital to be prepared for an epidemic situation. It is easy to imagine the complexity and unfeasibility of manually correlating news feeds with patient incident reports. Not only is the amount of news articles immense (and the amount of reports might be quite large as well) but the streams arrival rate might be too fast to cope with. Furthermore, it is practically impossible to keep track of the many details of the incidents in the reports to correlate them to the news articles about the epidemics onset which would require analyzing every report and every news article.

Our platform facilitates this by using novel techniques to extract relevant information from disparate sources of unstructured data (in this case the patient incident reports and the news feeds) and determine which elements (i.e., documents) in one stream are correlated to which elements in the other stream. It also has the capability to do inner correlation to compute reliability scores of the information extracted from fast streams. The need for this feature can easily be seen for news streams: the more

news articles report on a given event, the more reliable it is that the event indeed occurred. Consequently, as the streams are being processed, correlations are updated by factoring in a reliability score.

The process follows several steps:

i. First, the IE layer extracts relevant data from the patient incident reports. This includes symptoms and history facts like places where the patient has been, people who have been in contact with the patient, date when the symptoms appeared, etc. Notice that this is not a simple recognition of words or entities like a simple date, instead it is entity recognition at a higher semantic level where it is necessary to make distinctions like start date of symptoms versus other dates in the report (e.g., date of visit to the hospital). Given that there is ample variability in the way reports are written, it is necessary to learn models that recognize these kind of semantic entities which we call *role-based-entities*.
ii. Second, a categorizer in the Analytics layer classifies the articles from news feeds (e.g., New York Times RSS feeds) into interesting (e.g., "epidemics") and non-interesting (the others) categories.
iii. From those articles in the interesting categories the IE layer extracts relevant data about the location of the onset, the symptoms, the transmission channel, etc. Again, this is a high semantic level of entity recognition.
iv. Inner correlations of the descriptor formed from the extracted data with descriptors of previous articles within a time window are computed in the Analytics layer to derive reliability scores.
v. Finally the similarity of reports and epidemics related articles is measured using the extracted information as features and extending them along predefined hierarchies. The similarity is computed in the Analytics layer in terms of hierarchical neighbors using fast streaming data structures called Hierarchical Neighborhood Trees (HNT).

The *neighbor* concept is used to measure the similarity of categorical data. If two entities (incident report and news article) have the same value for a categorical variable (e.g. both have fever for a symptom variable), then they are neighbors and will be put into a same node in the HNT. If they have different values (e.g., vomiting and diarrhea) then they will be put into different nodes in the HNT and thus they are neighbors only at the level of their lowest common ancestor (e.g., stomach upset). Based on HNT, we thus can specify a scale based neighborhood relationship which measures the level in the HNT tree of the closest common ancestor of the tags. The hierarchical neighborhoods are critical to finding correlations between the reports and the news items. For example, assume a report doesn't mention fever by name but indicates that the patient has chills and a news article talks about high body temperature. The chills belong to a hierarchy where one of its ancestors is fever, and similarly high body temperature belongs to the hierarchy that contains fever as an ancestor. In this case, the report and the news article are neighbors at the level of fever. As a result we not only learn that these are neighbors but also that they are related through "fever". Once correlations are obtained, relevance scores are derived by factoring in the reliability scores computed before. For details on HNTs the interested reader is referred to [10].

The steps described above are done in two phases. The first one, corresponding to the very top part in Figure 5, is off-line and is domain-specific; it is where models are learned to extract information from documents and to classify them. This phase is preceded by a specification step where the user defines through a GUI the entities to be extracted from the documents and other relevant domain information like document categories of interest. We have developed flexible models that learn regularities in the textual context of the entities to be extracted while allowing some degree of variability. The models are provided with knobs to tune according to quality requirements, for example, to tradeoff accuracy for performance.

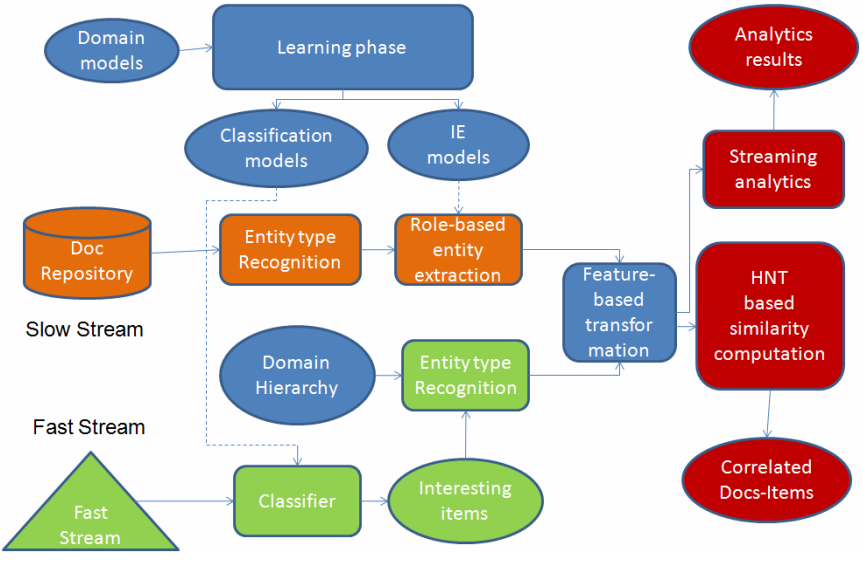

Fig. 5. IE & Analytics layers interaction

In the second phase, corresponding to the rest of Figure 5, the models learned in the previous phase are applied to classify documents and extract information from them. This can be done either off-line for a static document collection like a contracts repository or on-line for slow text streams like the contracts stream and for fast text streams like news feeds tweets or blogs. To speed up the search for role-based entities as well as to improve the precision of the results, the IE layer first uses entity recognizers to identify appropriate entity types. For example, to extract the date that symptoms appeared from a patient report it first recognizes entities of type date in the report and only on the textual neighborhood (surrounding words) of these entities it applies the extraction models learned in the previous phase to recognize the symptoms onset date. For the external fast stream the documents are first classified to discard those that are irrelevant (i.e, those that are not about epidemics) to avoid wasting time in attempting to extract information from them.

Once the information is extracted in terms of concepts (i.e., role-based entities) from the incident reports and the news items, the Analytics layer finds correlations using HNTs. Each concept belongs to one or more concept hierarchies and an HNT (Hierarchical Neighborhood Tree) is a tree based structure that represents all these hierarchies. Every document in the slow stream is converted to a feature vector (i.e., descriptor) where every feature is one of the extracted concepts. This document now coded as a multi-dimensional point is inserted into the HNTs for all the dimensions in the predefined hierarchies. For example suppose a report contains the concept "fever" and "Mexico" (e.g., the person visited Mexico). Then, it is coded as a two dimensional vector, where fever belongs to the "Symptoms" hierarchy and "Mexico", belongs to the "Location" hierarchy. In other words for the dimension "Symptoms" the value is "fever" and for the dimension "Location", the value is "Mexico". As a result of this process the reports in the slow stream are stored as multidimensional points in the HNTs. Likewise, when an "interesting" (i.e., "epidemics) news article comes in, it is similarly converted to a multidimensional point and inserted into the HNTs. The reports in each level of the hierarchy (for each of the dimensions) are the neighbors of the news item. For example, assume there is a news article n_1, which contains the concept "Honduras". The report (slow stream document) containing the concept "Mexico" is a neighbor at the level of "political region" to n_1. There are several advantages to using HNTs: (i) Mathematically, it allows us to overcome the problem of defining distances over categorical data such as regions, (ii) Inserts and deletes are very fast with the help of simple indexing schemes, (iii) It facilitates finding correlations at different abstraction levels, thus we can use the levels to quantify the "nearness", meaning points that are in the same node at a higher level are "less near" than the points which are in the same node at a lower level in the hierarchy.

The "nearness" is measured by the level of the lowest common ancestor. If news item n_i and a patient incident report c_k are neighbors in multiple HNTs they are more similar. This can be quantified with the following equation: $S_{i,k} = \prod_{j=1}^{d} s_{i,k,j}$, Where $S_{i,k}$ is the similarity between news item n_i and a report c_k and $s_{i,k,j}$ is the scale at which news item n_i and a report c_k are neighbors in HNT d. Here d corresponds to the dimension. This whole process is efficient as explained in [10].

Once we have computed the scale based similarity using the equation above, we compute the '*top k*' reports that are most correlated to the news. These reports represent the best candidates of patients affected by the epidemics (they match characteristics of the epidemic in terms of symptoms, place of origin, pre-conditions, etc). They make possible to derive information about the spread and effect that the epidemic onset is having in the area. With this valuable information, the hospital gains awareness of how the situation is developing and can get prepared accordingly.

In a nutshell, the goal of the IE and analytics layers of the platform is to reduce the time and effort to build data flows that integrate structured and unstructured data, slow and fast streams, and analyze (through correlation in our epidemics example) this combination of data in near-real-time to provide awareness of situations with potential impact on the enterprise business operations.

As part of this effort we are developing algorithms for different functions, including information extraction and analytics, to be wrapped as operators of a library used to build data flows. In addition, we are developing techniques to optimize these flows with respect to different quality metrics in addition to performance (we call this, QoX-based optimization) [11].

5 Related Work

Efforts have been made in this space in both industry and academia. There are OLTP database systems like Oracle, document management systems such as Documentum™, workflow management systems like FileNet™, Text Mining products such as PolyAnalyst™ and Attensity™, Extract-Transform-Load products like Informatica™. However we have found no solution that encompasses all of these functionalities in a single platform that manages structured and unstructured data, integrating them seamlessly to provide actionable insight for optimizing business operations.

Information extraction at the high semantic level done here is very challenging and requires training learning models. Although many proposals for such models exist [4], our experiments revealed that they do not work well for large documents nor they are flexible enough to cope well with variability in the text surrounding the information to be extracted (even if they claim the opposite). At the end, a lot of manual tuning needs to be done. This led us to decide creating our own algorithms for role-based entity extraction. In contrast, for plain entity extraction there are commercial and open-source frameworks and suites with recognizers well trained on vast collections of text or even manually created like ThingFinder [12] which is used in the IE layer for plain entity recognition.

Distance based clustering approaches are most relevant to our streaming correlation work. For high dimension data, distance computation can be expensive. In such cases a subspace projection approach can be used. Aggarwal [13] uses this approach for outlier detection. However, none of the approaches are designed for streaming data. A recent approach [14] computes distance based outliers over streaming windowed data. However, they do not consider the case of heterogeneity of fast streams and slow streams. In this paper we propose a novel and efficient solution with HNT to measure the correlation among the patients incident reports in a slow stream and news items in a fast stream based on categorical distances.

6 Conclusions

We have presented our operational business intelligence platform where information extraction, real-time processing and analytics are the core functionalities to achieve the goal of providing actionable insight in a timely manner for improving business operations and at the same time to extend the scope of transactional processes with data from unstructured sources. Our work is unique in that it integrates seamlessly all the necessary components into a single platform, making it easy to develop applications that benefit from the insight obtained. We have shown an application where unstructured data from patients incident reports and from streaming news articles is

extracted and analyzed in real-time to enable situational awareness of epidemic onsets for hospitals to be proactive in being prepared to deal with the epidemic.

Currently some components of the architecture have been implemented but we still have a long way to complete the development of algorithms and techniques for the rest of the platform components.

References

[1] Halevy, A.: Why your Data Won't Mix. Association for Computing Machinery Queue Magazine (October 2005)
[2] Castellanos, M., Dayal, U.: FACTS: An Approach to Unearth Legacy Contracts. In: Proc. First International Workshop on Electronic Contracting (WEC 2004), San Diego, CA (July 2004)
[3] Inmon, W.H., Strauss, D., Neushloss, G.: DW 2.0: The Architecture for the Next Generation for Data Warehousing. Morgan Kauffman, Burlington (2008)
[4] Sarawagi, S.: Information Extraction. Foundations and Trends in Databases 1(3), 261–377 (2008)
[5] Soderland, S.: Learning Information Extraction Rules for Semi-Structured and Free Text. Machine Learning 34(1-3), 233–272 (1999)
[6] Freitag, A.M.: Information Extraction with HMM Structures Learned by Stochastic Optimization. In: Proc. 17th National Conference on Artificial Intelligence. AI Press (2000)
[7] Peng, F., McCallum, A.: Accurate Information Extraction from Research Papers using Conditional Random Fields. In: HLT-NAACL, pp. 329–336 (2004)
[8] Michalewicz, Z.: Genetic Algorithms + Data Structures = Evolution Programs, 3rd edn. Springer, Heidelberg (1996)
[9] Inmon, W.H., Nesavich, A.: Tapping into Unstructured Data: Integrating Unstructured Data and Textual Analytics into Business Intelligence. Morgan Kaufmann, San Francisco (2007)
[10] Castellanos, M., Gupta, C., Wang, S., Dayal, U.: Leveraging Web Streams for Contractual Situational Awareness in Operational BI. To appear in Proc. EDBT Workshops, BEWEB 2010 (2010)
[11] Dayal, U., Castellanos, M., Simitsis, A., Wilkinson, K.: Data Integration Flows for Business Intelligence. In: Proc. EDBT (2009)
[12] Business Objects Thing Finder Language Guide and Reference. Business Objects an SAP Company (2009)
[13] Aggarwal, C., Han, J., Yu, P.S.: A framework for projected clustering of high dimensional data streams. In: Proceedings of the 30th VLDB Conference (2004)
[14] Angiulli, F., Fassetti, F.: Detecting distance-based outliers in streams of data. In: CIKM, pp. 811–820 (2007)