A PROPOSAL OF NOSQL ENABLED LOGISTICS INFORMATION SYSTEM FRAMEWORK

Fares Zaidi, Laurent Amanton, Eric Sanlaville

A PROPOSAL OF NOSQL ENABLED LOGISTICS INFORMATION SYSTEM FRAMEWORK

Fares Zaidi
Normandie Univ, UNIHAVRE, LITIS, 76600 Le Havre, France

Laurent Amanton
Normandie Univ, UNIHAVRE, LITIS, 76600 Le Havre, France

Eric Sanlaville
Normandie Univ, UNIHAVRE, LITIS, 76600 Le Havre, France

{fares.zaidi, laurent.amanton, eric.sanlaville}@univ-lehavre.fr

Abstract

Supply Chains and Logistics have a growing importance in global economy. Supply chains over the world are very heterogeneous and each one can both produce and receive massive amounts of structured and unstructured data in real-time, which are usually generated by information systems, smart objects or manually by humans. This heterogeneity is due to Logistics Systems components and processes that are developed by different modelling methods and running on many platforms.

In this paper we attempt to identify current challenges and integration issues between separately designed *Information and Communication Technology* systems and we propose a distributed NoSQL based Logistics Information System architecture that facilitate real-time cooperation between stakeholders in a multi-actor environment. We included also a brief discussion on perspectives and future scope of work.

Key words:

Logistics, Supply Chain Management, Distributed Information Systems, Big Data Analytics, NoSQL databases.

1. Introduction

Supply chains are the backbone of the global economy. Nowadays, research and innovation have become a major asset for improving sustainability and performance of supply chains. Needing for design, manage, and operate these complex global supply chains continues to grow. Basically, Supply Chain Management (SCM) is the art or science of getting products from manufacturers to customers. This definition can also be extended to service industries like tourism and health care which have some of the most sophisticated Supply Chains.

Managing supply chains involves dealing with four flows: the flow of physical products, the flow of money, the flow of information, and the reverse flow of products at the end of their life for recycling, re-manufacturing or disposal. In our study, we focus on the flow of information or data. Logistic networks are heterogeneous and fragmented since each stakeholder has his own Logistic Information System, usually called *Enterprise Resource Planning* (ERP). Thus, several challenges have arisen. Firstly, the most part of generated and exchanged data is unstructured and therefore difficult to store, process or analyse with legacy tools. Secondly, the competition between logistics actors or companies involves a lack of information sharing, cooperation and reactivity to risks. Although these actors have often divergent interests and values, they also need each other's support. Hence, it seems more than ever necessary to define a new paradigm for *Logistics Information Systems* (LIS). This paradigm consists of enhancing the flexibility and scalability of the LIS, handling massive and heterogeneous data, enabling a better real-time cooperation between stakeholders and facing reactively to risks while taking into account data and confidentiality issues. It relies on the emerging NoSQL databases and Big Data management tools to accomplish this task.

The rest of this paper is organized as follows. Section 2 presents Logistic Information Systems and NoSQL Databases. Then, our NoSQL based LIS architecture is described in section 3. Finally, Section 4 concludes the paper and highlights future scope of work.

2. Related work

2.1 Supply chains and Logistics Information Systems

Supply chain as defined by Cristopher [1] is "*a network of organizations that are involved through upstream and downstream linkages in the different processes and activities that produce value in the form of products and services in the hands of the ultimate customer*".

In fact, Supply Chains are composed of multiple stakeholders like suppliers, manufacturers, carriers, distributors and customers. As shown in Figure 1, Supply Chain deals with four flows, which are physical, money, data and reverse flows. Furthermore, all stakeholders or organizations have Plan, source and deliver activities, and most of them perform make activity, except some ones (like carriers or distributors).

their internal actions or processes like make, move and store using Internal Supply Chain Management processes (ISCM) [3]. These different functions needs to communicate with each other in order to exchange data easily, securely, correctly and in real-time in some cases.

Supply chains are very heterogeneous and complex, they involve multiple players which need instantaneous communications for B2B (Business to Business), B2C (Business to Customer) and M2M (Machine to Machine). Hence, ICT systems are the wise solution to enable these communications.

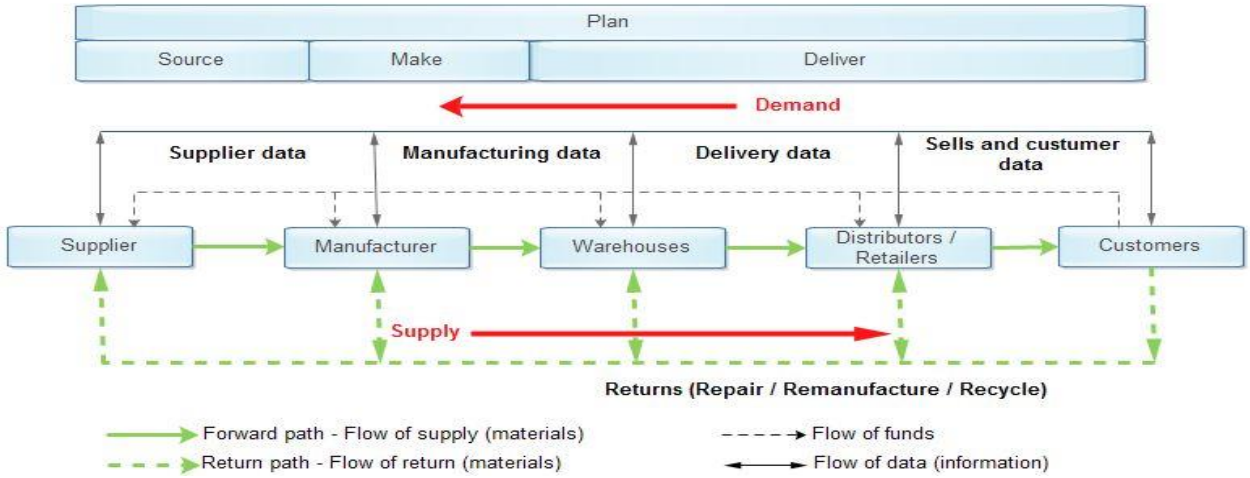

Figure 1: General Supply Chain (adapted from [4])

Organizations and their activities within the supply chain are integrated thanks to Plan activity [2]. If we look at stakeholders of a supply chain with a macro process view we can see that each member of the Supply Chain has three main components or modules, as illustrated in Figure 2 below.

Logistic Information Systems represent the data side of Supply Chains, their main objective is to manage information flows through SCs.

Figure 2: Macro perspective of a company in a Supply Chain (adapted from [2])

Companies connects to their suppliers with source activities (like acquiring raw materials) using Supplier Resource Management (SRM) module, and to their customers with deliver activities (like delivery of goods) using Customer Resource Management (CRM) module. They also operate

Managing data and information is overriding to deal with the other flows, i.e. physical, money and reverse flows. Hence, Logistics Information

Systems have an important role in Supply Chain Management since they are the hub of interconnection between different flows and their related platforms.

There are three classes of ICT systems related to logistics:

- Enterprise Resource Planning (ERP): represents the main database for firm activity.
- Supply Chain Planning: consists of Product Lifecycle Management (PLM), scheduling and production Planning.
- Supply Chain Execution: Warehouse Management Systems (WMS), Manufacturing Execution Systems (MES) and Transportation Management Systems (TMS).

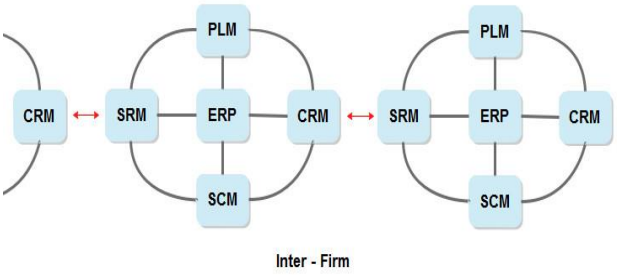

Figure 3: General Inter-firms interaction

Figure 3 presents the interconnection between Logistics companies. We can see that The Customer Relationship Management module of each company interacts with the Supplier Relationship management module of the other. This interconnection may results in integration issues among separately designed systems, in particular real-time managing of information flows. H. Wortmann and al. [5] organized interviews with 9 distribution companies specialised in Cross-Docking and they noted the following problems:

- GPS systems used to track merchandise and packages can't provide information about causes of delays and other issues. Hence, planners tend to communicate with drivers by email or mobile in order to be informed about transportation state and eventual issues and solutions.

- In addition to the previous problem, GPS systems are not integrated with stakeholders' transactional systems (like ERP or WMS), which implies that the last ones are not always up to date in real-time.

- Decision Support Systems (DSS) are not reliable as they are fed by transactional systems which can store bypassed data.

We can see here that there is a vicious circle created by issues related to GPS, transactional and decision support systems. Hence, it is necessary to break this circle and improve real-time cooperation and integration between heterogeneous ICT systems. For these reason, we proposed the solution detailed in section 4.

2.2 NoSQL Databases

Non-relational, open source and distributed databases also known as Not Only SQL or NoSQL are new generation databases that can handle massive amounts of data in real-time, whether structured, semi-structured or unstructured data [6]. They appeared by 2007 when Google and Amazon engineers designed BigTable [7] and DynamoDB [8] respectively.

NoSQL data stores are schema-less, i.e. their schema is dynamic and flexible, this is very useful to store and handle a variety of data structures, unlike legacy SQL or Relational Database Management Systems (RDBMS) that have a rigid schema; Moreover, these latter are based on ACID properties (Atomicity, Consistency, Isolation, Durability). Firstly, Atomicity requires that each transaction must be indivisible or atomic, if one slice of a transaction fails, then the whole transaction fails. Secondly Consistency ensures that the data stored in a cluster must be identical into all nodes, when a write is performed to one node, this write must be copied to all other nodes before responding to a request. Thirdly, Isolation stipulates that each transaction has to be executed as if it was the only one in the system, and the result of concurrent execution of transactions should be the same as if they were executed sequentially. Finally, Durability ensures that once a transaction is committed, it will be stored in a non-volatile memory before responding to a request, in order to avoid data-loss. All these constraints complicate the introduction of additional nodes (horizontal scalability) and therefore make traditional RDBMS clusters hard to scale. Table 1 summarizes comparison between SQL and NoSQL data stores.

SQL	NoSQL
- Relational - SQL language - ACID transactions - Hard scalability - Not adapted to big data - Rigid schema - Structured data	- Non-Relational - No common language - BASE - Horizontal scalability - Performance/ Big data - Schema-less/ Flexible - Structured and unstructured data

Table 1: SQL vs NoSQL data stores

Thus, to increase write operations capacity and improve cluster scalability, NoSQL data stores relax ACID requirements by using CAP theorem (Consistency, Availability, Partition tolerance) and

BASE model (Basically Available, Soft-State, and Eventual Consistency).

CAP theorem has been proposed by Eric Brewer in 2000 [9] and formally proven by Seth Gilbert and Nancy Lynch in 2002 [10]. It states that at a specific given moment *t*, a distributed system can only guarantee two of these three constraints:

- Data Consistency (C): All system nodes have always the same view of the data
- Availability (A): guarantee that each client can always read and write
- Partition tolerance (P): the system works well despite physical network partitions

BASE model [11] improves scalability by relaxing consistency constraints (Eventual Consistency) [12] [13], this model is optimistic. Conversely, ACID Model is pessimistic, i.e. it guarantees consistency whatever the circumstances.

NoSQL Databases are classified into four categories or families [14]:

- Key/value databases: key/value pairs can be stored in tables. this data structure reduces data access complexity to O(1)
- Column databases: data are stored in columns, unlike classic RDBMSs which stock data in rows.
- Document databases: they use documents called JSON (JavaScript Object Notation) to store data. These documents can be nested or grouped together into a set called Collection.
- Graph databases: information are stored into graphs, more exactly data are stored into nodes, and relationship between these nodes appears on edges. Thus, it is possible to query the database using common algorithms of graph theory. Low level layer of these databases can be key/value or document type.

Each database from the previous families can be used to operate with specific applications accordingly to several real constraints and needs, for example graph based databases are suitable to be used to store relationship between users in social networks. In a multi-actor logistic environment, data is generally unstructured and distributed over the supply chain, and the system needs to be scalable. So the ideal is to use a NoSQL database. After literature review [14] [15], we decided to use the most popular and powerful Nosql data store, MongoDB.
MongoDB is a document based NoSQL database, it respects the CP (Consistency and Partition tolerance) properties of the CAP theorem. MongoDB can scale horizontally with automated partitioning, also called Sharding, where each shard represents a replica set, and each replica set is composed of one Master (primary) and several slaves (secondaries). Master and slaves are considered as Mongo Daemon instances (mongod); thus, all writes go to the Master by default, so, the master-slave replication is used to ensure a high availability [6]. Figure 4 represents the general framework of MongoDB.

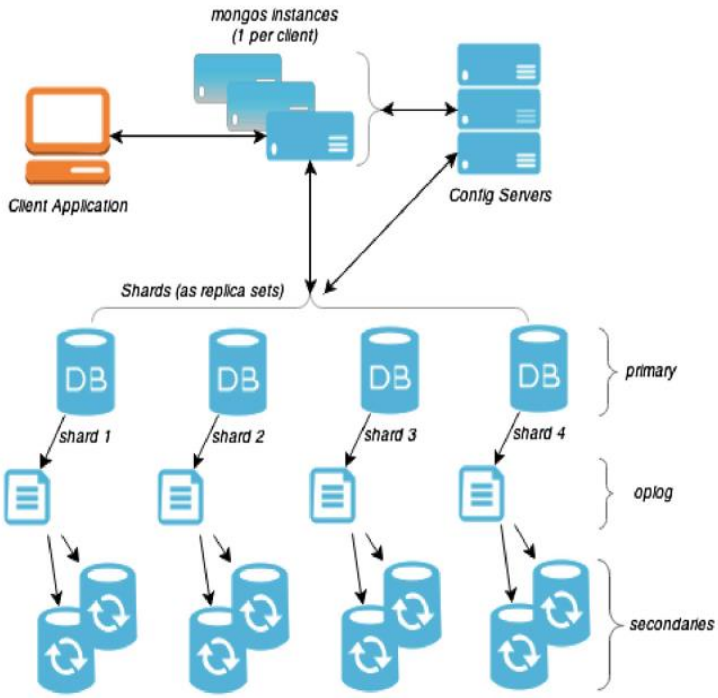

Figure 4: MongoDB framework [16]

MongoDB Shard (MongoS) represents the client side of MongoDB, it is required to distribute a data of a sharded collection across a sharded clusters, this process is called Balancing. MongoS acts also as a router in order to link between the customer's application layer and the sharded cluster in order to determine the location of the data.

Oplog (Operations LOG) is a special collection that keeps record of all operations received by the Master, then slaves copy and apply these operations in an asynchronous process in order to maintain the current state of the database. Oplog's size is 5% of physical memory in Linux and windows systems.

The next section will describe how we used MongoDB to empower our Logistic Information System framework.

3. Approach

Our case study concerns hospitals supply chain, in which we have to deal with the following constraints:

- Stakeholders over the supply chain are geographically distributed, and each one

has a specific need to some information in real-time, for example the customer (hospital) needs to get delivery date and possible delivery ways of medications packages, food or covers; The carrier needs to know the position of its trucks and road traffic state from an external source like internet; The supplier would like to find the carrier that proposes the cheapest and fastest shipping service, etc.

- They should get necessary information without calling or emailing each other.

- Data flow is generated over the supply chain by an increasing number of stakeholders and customers, and potentially millions of sensors and tracked objects. The most part of this data is unstructured and then need a flexible schema data store.

According to these constraints, we thought to build a platform that automates and standardizes data exchange between stakeholders, and breaks obstacles in order to improve cooperation.

The Logistics Information System architecture that we propose relies on MongoDB NoSQL database, as shown in Figure 5. The aim of this framework is to handle massive real-time and distributed data exchange and requests generated in a multi-actor logistics environment and provide logistics stakeholders like suppliers, carriers, customers or manufacturers with relevant information that they need at the right time.

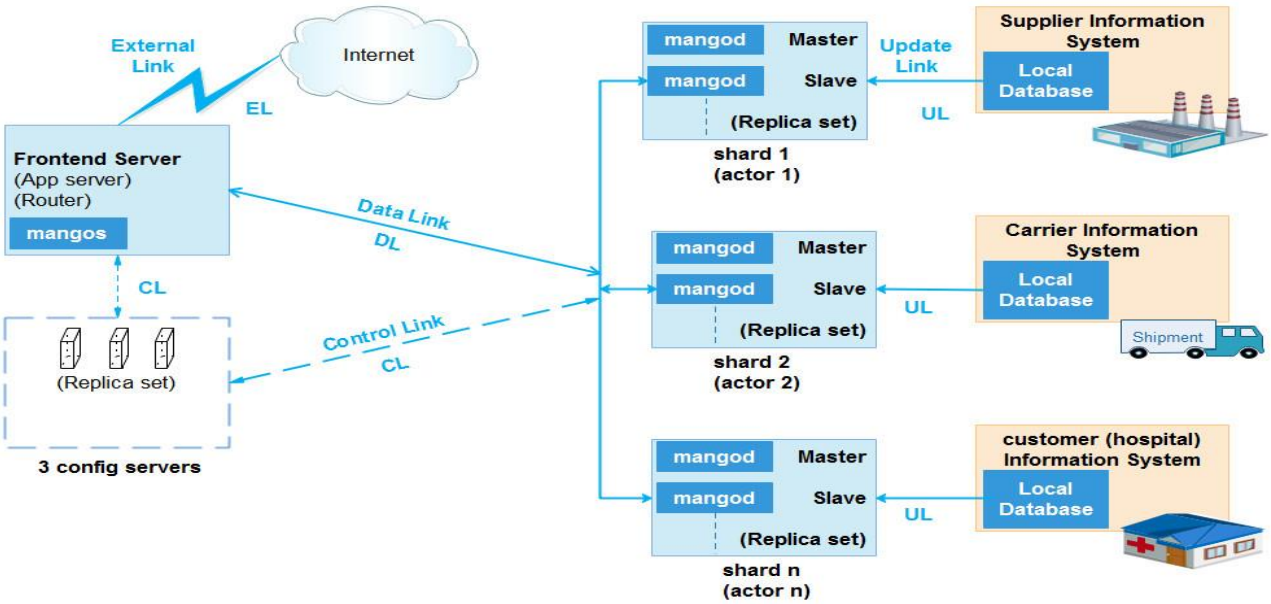

Figure 5: NoSQL Logistics Information System Architecture

The first component of the framework is the *frontend server* or application server, wherein *MongoS* (MongoDB Shard) is installed. Mongos is a routing service for mongoDB shard configurations which aims to link between the application layer and the sharded cluster in order to determine the location of the data. When mongos receives a query, it communicate with the config server to get the information needed to forward the query to the right shard (replica set).

The second component is the *Config Server*, which serves to store the mapping or metadata that links requested data with the shard that contains it. In production environment, sharded clusters have exactly 3 config servers to ensure high availability and redundancy.

The third component is composed of shards which are used to store stakeholders' shared data. A single shard is usually composed of a replica set with one Master and many slaves to ensure availability and redundancy. Each shard is connected to a local database of a logistic actor in order to collect necessary information via periodically updates.

Communication between the previous three main components is performed using data, control, update and also external links:

- Data link (DL) allows data forwarding and broadcasting in the global supply chain
- Control link (CL) allows transmission of requests and logistics network mapping between all framework components
- Update link (UL) enables feeding the shards of the distributed database
- External link (EL) collects information from the web on demand, which may be weather or traffic information, and so on

Data confidentiality aspect is also very important, the idea here is to provide management of the

Frontend server to a trusted third party. Data shared by stakeholders to feed shards constitute the minimum necessary to operate the system. No one has access to others information; When an actor needs an information, he sends a request to the frontend server, this one will contact the config servers via Control Links to get the location of information, afterwards it get necessary data from all related shards or external sources; thus, it aggregates the data and uses the result to respond to the initial request via Data Link.

4. Conclusion and future scope of work

In this paper we proposed a NoSQL based Logistics Information System framework in order to improve cooperation between logistic stakeholders and manage global supply chain data in real-time. Then, each actor can send requests to the frontend server and directly get related information without accessing to others sensitive data. We really believe that cooperation and information sharing between stakeholders is the solution for more improved LIS. In future work, we will design a prototype to handle Hospitals Supply Chain.

Acknowledgment

this work was financed by French government, region of Normandy, and European Union (European Regional Development Fund) within CLASSE and PERFAD projects.

References

[1] M. Christopher, *Logistics and Supply Chain Management: Strategies for Reducing Cost and Improving Service Financial*. London: Taylor & Francis, 1998.

[2] S. Stephens, “Supply Chain Operations Reference Model Version 5.0: A New Tool to Improve Supply Chain Efficiency and Achieve Best Practice,” *Inf. Syst. Front.*, vol. 3, no. 4, pp. 471–476, Dec. 2001.

[3] S. Biswas and J. Sen, “A Proposed Architecture for Big Data Driven Supply Chain Analytics,” Social Science Research Network, Rochester, NY, SSRN Scholarly Paper ID 2803901, Jun. 2016.

[4] S. Biswas and J. Sen, “A Proposed Framework of Next Generation Supply Chain Management Using Big Data Analytics,” 2016.

[5] Hans Wortmann, A. A. Alblas, P. Buijs, and K. Peters, “Supply Chain Integration for Sustainability Faces Sustaining ICT Problems,” in *Advances in Production Management Systems. Sustainable Production and Service Supply Chains*, 2013, pp. 493–500.

[6] K. Grolinger, W. A. Higashino, A. Tiwari, and M. A. Capretz, “Data management in cloud environments: NoSQL and NewSQL data stores,” *J. Cloud Comput. Adv. Syst. Appl.*, vol. 2, no. 1, p. 22, Dec. 2013.

[7] F. Chang *et al.*, “Bigtable: A distributed storage system for structured data,” *ACM Trans. Comput. Syst. TOCS*, vol. 26, no. 2, p. 4, 2008.

[8] G. DeCandia *et al.*, “Dynamo: amazon’s highly available key-value store,” *ACM SIGOPS Oper. Syst. Rev.*, vol. 41, no. 6, pp. 205–220, 2007.

[9] E. A. Brewer, “Towards robust distributed systems,” in *PODC*, 2000, vol. 7.

[10] S. Gilbert and N. Lynch, “Brewer’s Conjecture and the Feasibility of Consistent, Available, Partition-tolerant Web Services,” *SIGACT News*, vol. 33, no. 2, pp. 51–59, juin 2002.

[11] A. Fox, S. D. Gribble, Y. Chawathe, E. A. Brewer, and P. Gauthier, “Cluster-based Scalable Network Services,” in *Proceedings of the Sixteenth ACM Symposium on Operating Systems Principles*, New York, NY, USA, 1997, pp. 78–91.

[12] D. Pritchett, “BASE: An Acid Alternative,” *Queue*, vol. 6, no. 3, pp. 48–55, mai 2008.

[13] W. Vogels, “Eventually Consistent,” *Commun ACM*, vol. 52, no. 1, pp. 40–44, Jan. 2009.

[14] C. He, “Survey on NoSQL Database Technology,” *J. Appl. Sci. Eng. Innov. Vol*, vol. 2, no. 2, 2015.

[15] S. S. Nyati, S. Pawar, and R. Ingle, “Performance evaluation of unstructured NoSQL data over distributed framework,” in *2013 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, 2013, pp. 1623–1627.

[16] G. Haughian, R. Osman, and W. J. Knottenbelt, “Benchmarking Replication in Cassandra and MongoDB NoSQL Datastores,” in *Database and Expert Systems Applications*, S. Hartmann and H. Ma, Eds. Springer International Publishing, 2016, pp. 152–166.