Available online at www.sciencedirect.com

ScienceDirect

Procedia CIRP 52 (2016) 18 – 23

www.elsevier.com/locate/procedia

Changeable, Agile, Reconfigurable & Virtual Production

A Hyperconnected Manufacturing Collaboration System Using the Semantic Web and Hadoop Ecosystem System

Hsiao-Kang Lin[a]*, Jennifer A. Harding[b], Chun-I Chen[c]

[ac]Department of Industrial Management, I-Shou University, Kaohsiung, Taiwan ROC
[b]Wolfson School of Mechanical, Electrical and Manufacturing Engineering, Loughborough University, Loughborough, Leicestershire LE11 3TU, UK

* Corresponding author. Tel.: ＋886-7-657-7711 Ext.5514; fax: +886-7-657-8536. *E-mail address:* hklin@isu.edu.tw

Abstract

With the explosive growth of digital data communications in synergistic operating networks and cloud computing service, hyperconnected manufacturing collaboration systems face the challenges of extracting, processing, and analyzing data from multiple distributed web sources. Although semantic web technologies provide the solution to web data interoperability by storing the semantic web standard in relational databases for processing and analyzing of web-accessible heterogeneous digital data, web data storage and retrieval via the predefined schema of relational / SQL databases has become increasingly inefficient with the advent of big data. In response to this problem, the Hadoop Ecosystem System is being adopted to reduce the complexity of moving data to and from the big data cloud platform. This paper proposes a novel approach in a set of the Hadoop tools for information integration and interoperability across hyperconnected manufacturing collaboration systems. In the Hadoop approach, data is "**E**xtracted" from the web sources, "**L**oaded" into a set of the NoSQL Hadoop Database (HBase) tables, and then "**T**ransformed" and integrated into the desired format model with Hive's schema-on-read. A case study was conducted to illustrate that the Hadoop Extract-Load-Transform (ELT) approach for the syntax and semantics web data integration could be adopted across the global smartphone value chain.

Keywords: Hyperconnected Manufacturing Collaboration System; Hadoop Ecosystem System; Hadoop Database (HBase); Hadoop Hive; Extract-Load-Transform (ELT).

1. Introduction

In recent years, manufacturing systems have undergone profound changes, driven by the emerging hyperconnected manufacturing systems that are increasingly relying on knowledge enabled networks and cloud–based services. The knowledge-based collaboration projects, in particularly SYNERGY, provides the necessary technological infrastructure for supporting enterprise collaboration that enables the sharing of knowledge within Virtual Organizations (VOs) to the mutual benefit of the VO partners [1]. A Collaboration Moderator (CM) is a major component of the SYNERGY project. A CM is a specialist application for supporting individual collaboration partners by raising awareness of issues affecting items of interest identified by the members of VO. The prototype CM was implemented as a web-based and service-oriented software infrastructure. It is an integration platform that utilizes web services standards to support a Service-Oriented Architecture (SOA) within VOs. The information and knowledge is stored using semantic web technology. A common schema - Virtual Organization Knowledge Base (VOKB) is stored using a pre-agreed schema within the collaborative VOs for knowledge sharing [2, 3]. However, the participating partners in the VO need to transform or map their origin database schema into the VOKB to facilitate semantic data integration and interoperability.

More recently, the Hadoop Ecosystem System has been adopted on the cloud for web data integration [4, 5]. This paper proposes a novel approach in the form of a set of the Hadoop tools for information integration across a hyperconnected manufacturing collaboration system, without the need for mapping into the common / mediated model. In the course of the SYNERGY project, during research into the CM concept and VOKB modelling, the VOKB database was designed based on HBase tables, HDFS and Hive in Microsoft Azure cloud environment to create a scalability and high performance platform for dealing with heterogeneous web data. A case study was conducted to illustrate that the Hadoop for hyperconnected manufacturing collaboration system integration could be adopted across the global smartphone value chain. The paper is organized as follows: Section 2 loads VOKB ontology model to RDF syntax and designs logical VOKB HBase tables. Section 3 shows the Hadoop Extract-Load-Transform (ELT) approach for the syntax and semantics web data integration. Concluding comments are provided in section 4.

2. Loads VOKB Ontology Model into RDF and Designs VOKB HBase tables

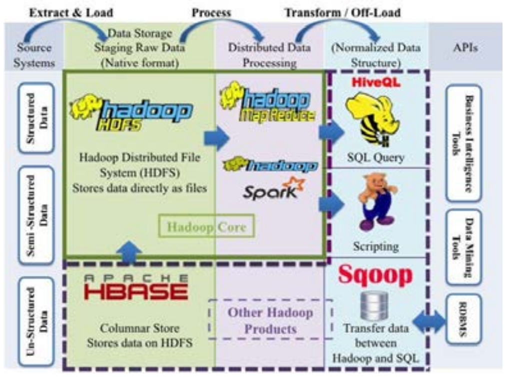
According to the Wiki (http://en.wikipedia.org/wiki/Semantic_Web）, "The semantic web aims at converting the current web, dominated by unstructured and semi-structured documents into a "web of data"." The semantic web stack builds on the W3C's Web Ontology Language (OWL), Resource Description Framework (RDF), and RDF schema (RDFS). These are the standard for conceptual description or modeling of information that is implemented in web documents. The RDF and relational / SQL databases mapping for storage, retrieval data and data interchange via the predefined common schema has become increasingly challenging. The heart of the SQL challenge is a process in database and especially in data warehousing known as "Extract, Transform & Load (ETL) ". Vassiliadis et al. [6] illustrated that an ETL system consists of three main functional steps: extraction, transformation, and loading.

- Extraction step is for extracting data from homogeneous or heterogeneous data sources,
- Transformation step involves cleaning, filtering, validating and applying business rules - which is for the customization and integration of the information coming from multiple sources into a common format.
- Loading is the final target step for subsequent analysis,

A traditional ETL architecture normally requires a transformation (intermediate) step between the extraction and the loading. Many researchers [7-10] have attempted to fill the role of the intermediate step of the ETL process by ETL tools, ontology, and semantic web technologies. Simitsis et al. [7] addressed ETL tools as pieces of software responsible for the extraction of data from several sources, their cleansing, customization and insertion into a data warehouse. Skoutas et al. [8] used ontologies for the conceptual design of ETL processes. Lin and Harding [9, 10] proposed semantic web technologies for the integration of the different manufacturing system engineering information through mapping into the mediated / common model and used the Web Ontology Language (OWL) as a middleware with data conversion rules to provide a mapping logic to automatically derive the transformations from the source attributes to the attributes belonging to the mediated / common format.

According to the Intel white paper in Big Data Analytics [11], none of middleware solutions is cheap or simple, and their cost and complexity are compounded by big data. When the source data sets are large, fast, and unstructured, traditional ETL can become the bottleneck, because it is too complex to develop, too expensive to operate, and takes too long to execute. The Hadoop provides an alternative approach in which data is "**E**xtracted" from the sources, "**L**oaded" into the HBase database, and then "**T**ransformed" and integrated into the desired format in Hive - (ELT).

The Apache Hadoop Ecosystem is an open source distributed software platform and consists of various components. There are two primary components at the core of Apache Hadoop for storing and processing data - the Hadoop Distributed File System (HDFS) and the MapReduce (Spark) parallel processing framework. The Hadoop Ecosystem brings additional functionality for loading column-oriented database, improving data access, transforming and off-loading data, such as HBase, Sqoop, Pig and Hive …etc, illustrated in figure 1.


Extract & Load
Process
Transform / Off-Load
Source Systems
Data Storage Staging Raw Data (Native format)
Distributed Data Processing
(Normalized Data Structure)
APIs
Structured Data
Semi-Structured Data
Un-Structured Data
hadoop HDFS
Hadoop Distributed File System (HDFS) Stores data directly as files
hadoop MapReduce
hadoop
Spark
Hadoop Core
HiveQL
SQL Query
Scripting
APACHE HBASE
Columnar Store Stores data on HDFS
Other Hadoop Products
Sqoop
Transfer data between Hadoop and SQL
Business Intelligence Tools
Data Mining Tools
RDBMS
Figure 1: Apache Hadoop Ecosystem

In this paper, the RDF and the HBase table are used to establish a data integration platform for the Virtual Organization Knowledge Base (VOKB), defined by the SYNERGY project. A VO is a short-term association with a specific goal of acquiring and exploiting a business opportunity. However, despite the short-term nature of the VO, to operate successfully partners in a VO must share knowledge and information to a significant degree. In addition, there is a body of knowledge, which is relevant to the VO as a whole, regarding structural, and operational aspects of the VO. Again mutual understanding of this knowledge amongst partners is essential to the success of the VO.

The VOKB therefore describes the terminology in the VO knowledge. For example, A **Virtual Organisation (VO)** has VOPartners each of which is an **Enterprise** collaborating with other enterprises who are in turn **VOPartner**s in the same VO. A VO has **VOActivity** and a common goal to carryout a **VOProject** in order to deliver a **VOProduct**. The details of the VOKB ontology structures can be found in SYNERGY project deliverable reports[3]. As mentioned earliest, RDF is a standard model for information interchange on the Web. The aim of this paper is to represent RDF/XML assertions in an HBase database and discover how to store and process the data. Therefore, the defined abstraction of the VOKB Ontology is converted into RDF/XML and is illustrated in figure 2.

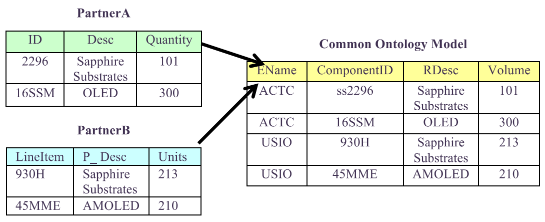
```
<?xml version="1.0"?>
<!DOCTYPE rdf:RDF [
  <!ENTITY owl "http://www.w3.org/2002/07/owl#" >
  <!ENTITY xsd "http://www.w3.org/2001/XMLSchema#" >
  <!ENTITY rdfs "http://www.w3.org/2000/01/rdf-schema#" >
  <!ENTITY rdf "http://www.w3.org/1999/02/22-rdf-syntax-ns#" >]>
<rdf:RDF xmlns="http://www.w3.org/2002/07/owl#"
   xml:base="http//www.w3.org/2002/07/owl"
   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
   xmlns:owl="http://www.w3.org/2002/07/owl#"
   xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
   xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">
  <Ontology rdf:about="http://www.semanticweb.org /ontologies/VOKB"/>
<Class rdf:about="http://www.semanticweb.org /ontologies/VOKB#Facility"/>
   <Class rdf:about="http://www.semanticweb.org /ontologies/VOKB#Enterprise">
      <rdfs:subClassOf rdf:resource="http://www.semanticweb.org /ontologies/VOKB#Facility"/>
   </Class>
   <Class rdf:about="http://www.semanticweb.org /ontologies/VOKB#VirtualOrganisation">
      <rdfs:subClassOf rdf:resource="http://www.semanticweb.org /ontologies/VOKB#Facility"/>
   </Class>

   <Class rdf:about="http://www.semanticweb.org /ontologies/VOKB#BusinessOpportunity">
      <rdfs:subClassOf rdf:resource="http://www.semanticweb.org /ontologies/VOKB#VirtualOrganisation"/>
   </Class>
   <Class rdf:about="http://www.semanticweb.org /ontologies/VOKB#CollaborationPattern">
      <rdfs:subClassOf rdf:resource="http://www.semanticweb.org /ontologies/VOKB#VirtualOrganisation"/>
   </Class>
   <Class rdf:about="http://www.semanticweb.org /ontologies/VOKB#ModeratorKnowledge">
      <rdfs:subClassOf rdf:resource="http://www.semanticweb.org /ontologies/VOKB#VirtualOrganisation"/>
   </Class>
   <Class rdf:about="http://www.semanticweb.org /ontologies/VOKB#VOPartner">
      <rdfs:subClassOf rdf:resource="http://www.semanticweb.org /ontologies/VOKB#VirtualOrganisation"/>
   </Class>
   <Class rdf:about="http://www.semanticweb.org /ontologies/VOKB#VOActivity">
      <rdfs:subClassOf rdf:resource="http://www.semanticweb.org /ontologies/VOKB#VirtualOrganisation"/>
   </Class>

   <Class rdf:about="http://www.semanticweb.org /ontologies/VOKB#VOProject">
      <rdfs:subClassOf rdf:resource="http://www.semanticweb.org /ontologies/VOKB#VOActivity"/>
   </Class>
   <Class rdf:about="http://www.semanticweb.org /ontologies/VOKB#VOProduct">
      <rdfs:subClassOf rdf:resource="http://www.semanticweb.org /ontologies/VOKB#VOProject"/>
   </Class>
</rdf:RDF>
```

Figure 2: VOKB in RDF

The first step when modelling a semantic web with HBase involves mapping the RDF to the HBase tables. To store the RDF/XML descriptions of the VOKB, the classes in RDF to HBase tables, such as, Facility, Enterprise, Virtual Organizations (VOs), VOPartner, VOProject, VOProduct, VOProduction and VO Production_Resource (VOPR) …etc. This paper only illustrates the details of the HBase tables that demonstrate how these tables support distributed semantics web interoperability. The VO Production_Resource (VOPR) HBase table will be described in section 3.2.

3. Hadoop ELT for the Syntax and Semantic Web Data Integration

We built a test case involving development of a cloud-based infrastructure for creating, operating, evolution and eventually dissolving of a virtual organization in the smartphone industry. Lin and Harding [9, 10] observed that VO project team members in different parts of the world, each worked using their own preferred terminology. However, when people are brought together from different groups or companies, two common types of problem can occur in communications that share and exchange information. Firstly, that the same term is being applied to different concepts (semantic problem) and secondly, that different terms may be used to denote the same entity (syntax problem). Figure 3, for example, shows that the column heading "ID" used by Partner A has the same meaning as "LineItem" used by PartnerB: These two different identifiers exist within different models, but mean the same thing. For the Common Ontology Model they propose the single term "ComponentID" that is stored on pre-agreed schema within the collaborative VOs, whilst simultaneously sharing information through the transforming / mapping mechanisms of the common ontology model.



ID	Desc	Quantity
2296	Sapphire Substrates	101
16SSM	OLED	300

LineItem	P_ Desc	Units
930H	Sapphire Substrates	213
45MME	AMOLED	210

EName	ComponentID	RDesc	Volume
ACTC	ss2296	Sapphire Substrates	101
ACTC	16SSM	OLED	300
USIO	930H	Sapphire Substrates	213
USIO	45MME	AMOLED	210

Figure 3: Predefined schema transformed into the common ontology model for data integration

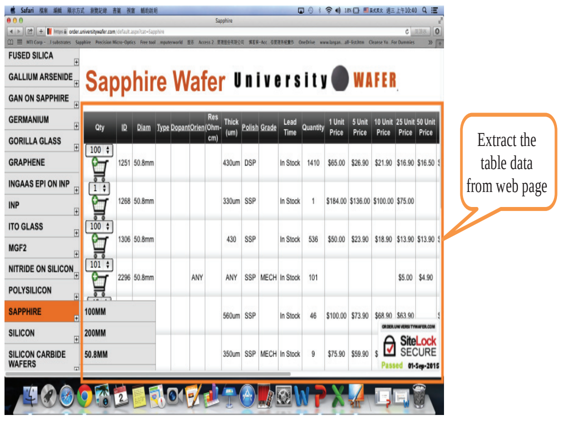
This section aims to show how the Hadoop ELT approach for the syntax and semantics web data integration could be adopted across the global smartphone value chain, without the need for mapping into the mediated model. The operation of the Hadoop ELT process is now demonstrated through the example of new VO projects / new collaborative production in a manufacturing resource e-planning task, using Microsoft Azure HDInsight Service. It is a cloud PaaS (platform as a service) that deploys and provisions Hadoop clusters for the HDFS / MapReduce software framework and related projects such as HBase, Pig, Sqoop and Hive in a simpler, more scalable, and cost-efficient environment. Three main steps of the Hadoop ELT process are: Extract data from the web, load the extracted data into the HBase tables and transform and off-load data by Hadoop Hive.

3.1. Extract data from the web

The growing popularity of hyperconnected manufacturing collaboration systems on the World Wide Web has resulted in a huge amount of information sources via the Internet. It is necessary to analyse this big volume of data and extract useful information from the web. Ferrara et al. [12] provided a simple classification framework in which existing Web Data Extraction (WDE) applications are grouped into two main classes, namely applications at the Enterprise level and at the Social Web level. At the Enterprise level, WDE techniques emerge as a key tool to perform data analysis in business and competitive intelligence systems as well as for business process reengineering. At the Social Web level, WDE techniques allow gathering of the large amounts of structured data that is being continuously generated and disseminated by Web 2.0, Social Media and Online Social Network users and this offers unprecedented opportunities to analyze human behavior at a very large scale. WDE is a process of retrieving unstructured (such as user comments, and web logs), semi-structured data (such as tables or itemized and enumerated lists) or social media data (from sites such as Facebook, LinkedIn, and Twitter) from web pages in HTML, XML, RDF format …etc. and importing it into a structured data system like a database or spreadsheet.

In addition, Chang et at.[13] pointed out that the Internet presents large amounts of potentially useful information, it is usually heterogeneous and lacking in structure. To automate the translation of input pages into structured data, considerable effort has been directed towards the area of Information Extraction (IE). Unlike Information Retrieval (IR), which concerns how to identify relevant documents from a document collection. IE produces structured data ready for post-processing, which is crucial to many applications of searching and web mining tools. Moreover, various approaches [12-14] have been proposed to efficiently collect the data with limited human effort. Recently, several tools have been developed that allow for WDE using a simple point-and-click interface to automatically enable the process of extracting specific text and images from any website into text files, spreadsheet, or a wide variety of formats, e.g. Mozenda https://www.mozenda.com, Winautomation http://www.winautomation.com, import.io https://import.io and Microsoft Excel…etc.

In this paper, Microsoft Excel's get external data feature (choose Data > From Web) is used to focus on WDE from semi-structured documents. Figure 4 shows a semi-structured web page containing table data to be extracted into the format of individual partners, and the extracted data is then loaded into our designed HBase table - VO Production_Resource (VOPR) HBase Table.



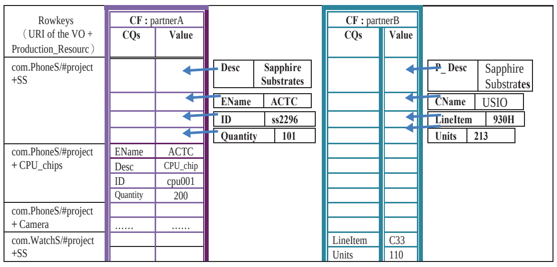
Qty	ID	Diam	Type	Dopant	Orien	Res (Ohm-cm)	Thick (um)	Polish	Grade	Lead Time	Quantity	1 Unit Price	5 Unit Price	10 Unit Price	25 Unit Price	50 Unit Price
100	1251	50.8mm					430um	DSP		In Stock	1410	$65.00	$26.90	$21.90	$16.90	$16.50
1	1268	50.8mm					330um	SSP		In Stock	1	$184.00	$136.00	$100.00	$75.00	
100	1306	50.8mm					430	SSP		In Stock	536	$50.00	$23.90	$18.90	$13.90	$13.90
101	2296	50.8mm			ANY		ANY	SSP	MECH	In Stock	101				$5.00	$4.90
							560um	SSP		In Stock	46	$100.00	$73.90	$68.90	$63.90	
							350um	SSP	MECH	In Stock	9	$75.90	$59.90	$		

Figure 4: A Semi-structured page containing table data to be extracted from web page

3.2. Load the extracted data into the HBase tables

It is well known that the key characteristic of HBase is called "no schema - on-write", which means such systems do not require to the data schema to be pre-defined before loading data into HBase [15, 16]. HBase may offer a better solution for web data extraction and web mining analysis. According to features proposed by the application, a VOPR table was created with multiple Column Families (CFs) – partnerA and partnerB …etc. Each partner has been designed to have his own CF that is physically stored together on the same HDFSs. Moreover, CF can be incrementally created while new numbers of partners join the project. In HBase, each CF has one or more non-predefined Column Qualifiers (CQs). Therefore, the individual project partners are able to use their own preferred terminology in the newly created CQs.

Table 1: The Logical VO Production_Resource (VOPR) HBase Table



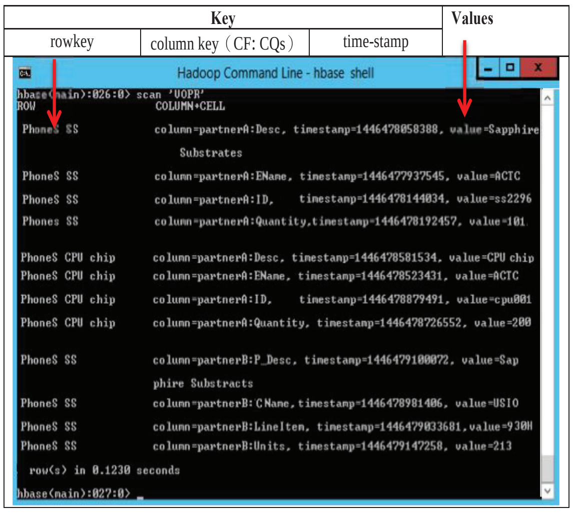
Rowkeys (URI of the VO + Production_Resourc)	CF : partnerA CQs	CF : partnerA Value	CF : partnerB CQs	CF : partnerB Value
com.PhoneS/#project +SS				
com.PhoneS/#project + CPU_chips	EName	ACTC		
	Desc	CPU_chip		
	ID	cpu001		
	Quantity	200		
com.PhoneS/#project + Camera				
	……	……		
com.WatchS/#project +SS			LineItem	C33
			Units	110

Desc	Sapphire Substrates
EName	ACTC
ID	ss2296
Quantity	101

P_ Desc	Sapphire Substrates
CName	USIO
LineItem	930H
Units	213

For example, partnerA can load its own model and original terminology into the VOPR by adding four new CQs: Desc, EName, ID and Quantity with value of " Sapphire Substrates", "ACTC", "ss2296" and "101", respectively. Similarly, partnerB can load its original terminology into the VOPR by adding four new CQs: P_Desc, CName, LineItem and Units with value of "Sapphire Substrates", "USIO", "930H" and "213", respectively. Table 1 illustrates how the individual partners can each load their original model into the HBase table.

Currently, there are two ways to create and access HBase data: HBase shell and HBase API. In this paper, the HBase physical table is created and manipulated from the HBase shell in Microsoft Azure HDInsight Service. HBase is a database that stores data in tables were created first, so the VOPR table and column family; adding new column qualifiers and data by the shell command. HBase command for data manipulation consists of three primary methods: Get, Put, and Scan. Gets and Puts are specific to particular rows and need the row key to be provided. Scans are done over a range of rows. In the figure 5, the "scan" command lists the VOPR table content that illustrates all the row keys, organized by column key, with the value associated at each timestamp.



```
hbase(main):026:0> scan 'VOPR'
ROW                    COLUMN+CELL
 PhoneS SS             column=partnerA:Desc, timestamp=1446478058388, value=Sapphire
                           Substrates
 PhoneS SS             column=partnerA:EName, timestamp=1446477937545, value=ACTC
 PhoneS SS             column=partnerA:ID,    timestamp=1446478144034, value=ss2296
 Phones SS             column=partnerA:Quantity,timestamp=1446478192457, value=101

 PhoneS CPU chip       column=partnerA:Desc, timestamp=1446478581534, value=CPU chip
 PhoneS CPU chip       column=partnerA:EName, timestamp=1446478523431, value=ACTC
 PhoneS CPU chip       column=partnerA:ID,    timestamp=1446478879491, value=cpu001
 PhoneS CPU chip       column=partnerA:Quantity, timestamp=1446478726552, value=200

 PhoneS SS             column=partnerB:P_Desc, timestamp=1446479100072, value=Sap
                       phire Substracts
 PhoneS SS             column=partnerB:CName, timestamp=1446478981406, value=USIO
 PhoneS SS             column=partnerB:LineItem, timestamp=1446479033681,value=930H
 PhoneS SS             column=partnerB:Units, timestamp=1446479147258, value=213
  row(s) in 0.1230 seconds
hbase(main):027:0> _
```

Figure 5: Using HBase "scan" command to read data from VOPR table

As shows in figure 5, records in HBase are stored in the HDFS as key-value pairs (rowley: value) --> (PhoneS SS: Sapphire Substrates), (PhoneS SS: ACTC), (PhoneS SS: ss2296), (PhoneS SS: 101)…etc. The HDFS itself is a binary file and is not human-readable. HDInsight Service uses Azure Blob storage as the big data store for HDFS. In this case, the HDFS file for the VOPR HBase table store resides in a default file system with Azure storage account for the cluster looking something like table 2: vopr.txt.

Table 2: the vopr.txt HDFS file

rowkey : value

rowkey	value			
PhoneS SS	Sapphire Substrates	ACTC	ss2296	101
PhoneS CPU_chip	CPU_chip	ACTC	cpu001	200
PhoneS SS	Sapphire Substrates	USIO	9300H	213
………………………………………………………………………				

3.3. Transform and off-load data using Hadoop Hive

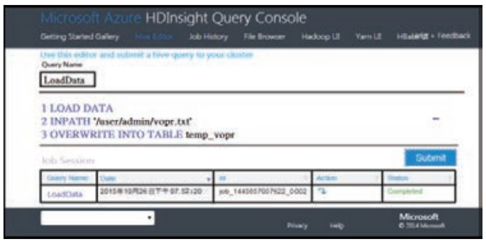
Many traditional data systems use the so called "schema-on-write" model where users need to decide on the schema of their data before loading data into their system. This model would start out by understanding how the data needed to be used, then design appropriate schemas, and manipulate the data to fit those schemas. In contrast, "Schema-on-read" models do not need to know how the data will be stored. This means that "Schema-on-read" models can load / write data first and then apply the structure of a schema to the data "on read". One of the most-cited advantages of the Hadoop Ecosystem is that it enables a "schema-on-read" data analysis strategy. As shown in table 7, HDFS files allow for the storing all the data without understanding what the data elements are.(No schema has been defined yet). A schema to fit the data needs is built later.

The Apache Hive ™ facilitates querying, managing and analysis of large datasets and provides an SQL-like query language called HiveQL https://cwiki.apache.org/confluence/display/Hive/GettingStarted. In addition, the Apache Hive ™ enables users to build a custom schema and directly query self-describing data. Grover and Majumdar [17] called this process "Descriptive Data Modeling". This approach enables data to flow into the system in its original form / native format, and then the schema is parsed at read time, allowing for extreme agility while dealing with complex evolving data structures.

In this paper, Hive's schema-on-read capability is demonstrated to show how data is loaded into HBase tables, stored in HDFS and managed in the Hive environment, using the Microsoft Azure cloud platform. In particular, Azure HDInsight deploys Hadoop clusters, which include HBase and Hive as well as other technologies under the Hadoop Ecosystem. We created an HDInsight cluster on Azure and ran HiveQL jobs using the HDInsight query console. This case study describes the creation of :

1. Loading data from HDFS to external Hive table
2. Creating a new internal Hive table with a custom schema
3. Extracting the data from an external Hive table and copying it into the new schema in an internal Hive table

The first task is to load data from HDFS to Hive. An external *temp_vopr table* is created and loaded with the native format data from vopr.txt HDFS files （see table 2） into the *temp_vopr* table. The completed query is shown in figure 6, which shows the Hive's command in the HDInsight Hive query console.



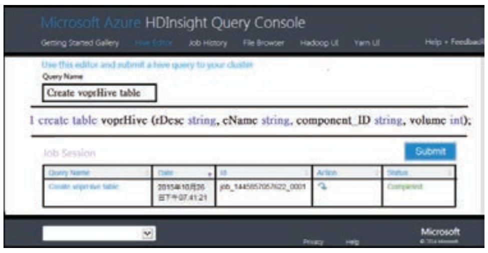
Query Name: LoadData

```
LOAD DATA
INPATH '/user/admin/vopr.txt'
OVERWRITE INTO TABLE temp_vopr
```

Query Name	Date	Id	Action	Status
LoadData	2015年10月26日下午07:52:20	job_1445857057622_0002		Completed

Figure 6: Load the vopr.txt data into the external temp_vopr Hive table

The second task is to create a new internal Hive table called "*voprHive*" and add "Custom Schema" to fit the data in the *temp_vopr* table. The new "*voprHive*" table has four columns, namely, Resource Description (rDesc), Enterprise Name (eName), (component_ID) and the currently available volume (volume), shown in figure 7.



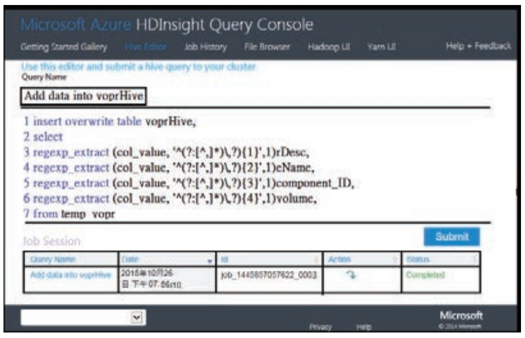
Query Name: Create voprHive table

```
create table voprHive (rDesc string, eName string, component_ID string, volume int);
```

Query Name	Date	Id	Action	Status
Create voprHive table	2015年10月26日下午07:41:21	job_1445857057622_0001		Completed

Figure 7: Create voprHive table with new custom schema

The third task is to extract the data from temp_vopr and copy it into the new schema in voprHive table. To do this we build up a multi-line query. The four regexp_extract calls are to extract the rDesc, eName, component_ID and volume fields from temp_vopr, as shown in figure 8.



Query Name: Add data into voprHive

```
insert overwrite table voprHive,
select
regexp_extract (col_value, '^(?:[^,]*)\,?){1}',1)rDesc,
regexp_extract (col_value, '^(?:[^,]*)\,?){2}',1)eName,
regexp_extract (col_value, '^(?:[^,]*)\,?){3}',1)component_ID,
regexp_extract (col_value, '^(?:[^,]*)\,?){4}',1)volume,
from temp_vopr
```

Query Name	Date	Id	Action	Status
Add data into voprHive	2015年10月26日下午07:56:10	job_1445857057622_0003		Completed

Figure 8: Add data into voprHive from temp_vopr

As mentioned earlier and shown in figure 6 and 7, records in the HDFS are key-value pairs (rowkey: value) --> (PhoneS SS: Sapphire Substrates), (PhoneS SS: ACTC), (PhoneS SS:

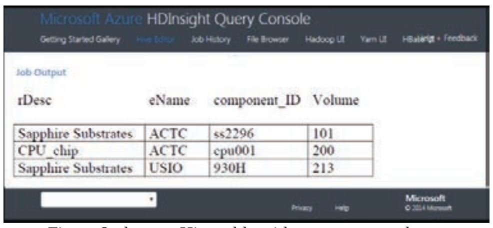
ss2296), (PhoneS SS: 101)…etc. The regexp_extract function in Hive is to used to extract the "col_value" key-value pair and store the value into the newly created custom columns. The "col_value" key is PhoneS SS and returns first pattern value: Sapphire Substrates, and then stores it into the rDesc field. The Hive built-in String function - regexp_extract (string subject, string pattern, int index) is used to read data and returns the string extracted using the pattern. e.g. The piece of code shown below.

```
                          subject            pattern           index
                             ↓                  ↓                 ↓
regexp_extract   ( col_value,   '^(?:([^,]*)\,?){1}',      1)      rDesc
```

The final task is to execute the query and look at the new "voprHive" table with the new custom schema in Hive:

```
Select rDesc, eName, component_ID, volume
from temp_vopr,
```

Once the status shows that the job is completed, click the query name on the screen to see, the voprHive table with the new custom schema, as shown in figure 9.



Job Output

rDesc	eName	component_ID	Volume
Sapphire Substrates	ACTC	ss2296	101
CPU_chip	ACTC	cpu001	200
Sapphire Substrates	USIO	930H	213

Figure 9: the voprHive table with new custom schema

4. Conclusion

This paper explores the use of Hadoop tools for the purpose of information integration across a hyperconnected manufacturing collaboration system. The Hadoop Ecosystem running in the cloud platform is generating new opportunities and presenting new challenges for businesses across every industry sector. The challenges of data integration incorporating data from the web and other unstructured data from multiple sources, is one of the most urgent issues facing the successful implementation of hyperconnected manufacturing collaboration system. Based on our evaluation, using the Microsoft Azure cloud platform and Hadoop tools for enterprise ELT processes to achieve collaborative production in an example of a manufacturing resource e-planning task, the greatest benefits derived are in the areas of improved performance and functionality.

References

[1]. Popplewell, Keith, Nenad Stojanovic, Andreas Abecker, Dimitris Apostolou, Gregoris Mentzas, and Jenny Harding, *Supporting Adaptive Enterprise Collaboration through Semantic Knowledge Services*, in *Enterprise Interoperability III*, K. Mertins, R. Ruggaber, K. Popplewell, and X. Xu, Editors. 2008, Springer London. p. 381-393.

[2]. Harding, J. A. and R. Swarnkar, *Implementing collaboration moderator service to support various phases of virtual organisations.* International Journal of Production Research, 2013. **51**(23-24): p. 7372-7387.

[3]. Dai, Xiaojun, Keith Popplewell, and Muqi Wulan, *Collaboration Knowledge Services Framework*, in *SYNERGY: Seventh Framework Programme ICT-2007-1-1.3*. 2009.

[4]. Hausenblas, Michael, Robert Grossman, Andreas Harth, and Philippe Cudré-Mauroux. *Large-scale Linked Data Processing - Cloud Computing to the Rescue?* in *Proceedings of the 2nd International Conference on Cloud Computing and Services Science*. 2012. Porto, Portugal, .

[5]. Yang, Chao-Tung, Jung-Chun Liu, Wen-Hung Hsu, Hsin-Wen Lu, and William Cheng-Chung Chu. *Implementation of Data Transform Method into NoSQL Database for Healthcare Data*. in *Parallel and Distributed Computing, Applications and Technologies (PDCAT), 2013 International Conference on*. 2013.

[6]. Vassiliadis, Panos, Alkis Simitsis, and Spiros Skiadopoulos. *Conceptual modeling for ETL processes*. in *Proceedings of the 5th ACM international workshop on Data Warehousing and OLAP*. 2002. ACM.

[7]. Simitsis, Alkis, Panos Vassiliadis, and Timos Sellis. *Optimizing ETL processes in data warehouses*. in *Data Engineering, 2005. ICDE 2005. Proceedings. 21st International Conference on*. 2005.

[8]. Skoutas, Dimitrios, Alkis Simitsis, and Timos Sellis, *Ontology-Driven Conceptual Design of ETL Processes Using Graph Transformations*, in *Journal on Data Semantics XIII*, S. Spaccapietra, E. Zimányi, and I.-Y. Song, Editors. 2009, Springer Berlin Heidelberg. p. 120-146.

[9]. Lin, H. K. and J. A. Harding, *A manufacturing system engineering ontology model on the semantic web for inter-enterprise collaboration.* Computers in Industry, 2007. **58**(5): p. 428-437.

[10]. Lin, H. K., J. A. Harding, and W. C. Tsai, *A rule-based knowledge system on semantic web for collaboration moderator services.* International Journal of Production Research, 2012. **50**(3): p. 805-816.

[11]. ETL-Big Data-with-Hadoop. *White Paper: Extract, Transform, and Load Big Data with Apache Hadoop*. 2013 Nov, 2014]; Available from: https://software.intel.com/sites/default/files/article/402274/etl-big-data-with-hadoop.pdf.

[12]. Ferrara, Emilio, Pasquale De Meo, Giacomo Fiumara, and Robert Baumgartner, *Web data extraction, applications and techniques: A survey.* Knowledge-Based Systems, 2014. **70**: p. 301-323.

[13]. Chang, Chia Hui, Mohammed Kayed, M. R. Girgis, and K. F. Shaalan, *A Survey of Web Information Extraction Systems.* Knowledge and Data Engineering, IEEE Transactions on, 2006. **18**(10): p. 1411-1428.

[14]. Tak-Lam, Wong and Lam Wai, *Learning to Adapt Web Information Extraction Knowledge and Discovering New Attributes via a Bayesian Approach.* Knowledge and Data Engineering, IEEE Transactions on, 2010. **22**(4): p. 523-536.

[15]. Bhupathiraju, V. and R. P. Ravuri. *The dawn of Big Data - Hbase*. in *IT in Business, Industry and Government (CSIBIG), 2014 Conference on*. 2014.

[16]. Dimiduk Nick and Amandeep Khurana, *HBase in Action*. 2012: Manning Publications Co.

[17]. Grover, Mark and Prasad Majumdar. *Hive in Enterprises*. 2013 Oct, 2015]; Available from: http://www.slideshare.net/markgrover/hive-in-enterprises?next_slideshow=1.