ISSN 2395-1621

Conflicts analysis of Smart City Services using Live Data

[#1]Shubham Wagdarkar, [#2]Vrushali Phaphle, [#3]Palash Zanzane, [#4]Rani Shinde
[#5]Prof. T.R. Patil

[1]shubhamw0gmail.com
[2]rishaphapale101@gmail.com,
[3]palzanzane07@gmail.com
[4]ranishinde0904@gmail.com
[5]tulsidas.patil@sinhgad.edu

[#12345]Information Technology Department,

NBN Sinhgad School of Engineering.

ABSTRACT

The populations of large cities around the world are growing rapidly. Cities are beginning to address this problem by implementing significant sensing and actuation infrastructure and building services on this infrastructure. However, as the density of sensing and actuation increases and as the complexities of services grow there is an increasing potential for conflicts across Smart City services. These conflicts can cause unsafe situations and disrupt the benefits that the services were originally intended to provide. This project detect and analyze the runtime conflict in smart city transportation services using real-time data feed. We will prepare an architecture, which will focus on conflict that arise on these services. 1. Train 2. Tube 3. Bus Our architecture solution identifies the conflict and provide additional inputs to decision making aspect of services.

ARTICLE INFO

Article History

Received: 14th March 2017

Received in revised form :

14th March 2017

Accepted: 16th March 2017

Published online :

3rd April 2017

I. INTRODUCTION

Smart city is not a new phenomena, but has never before proceeded so rapidly. Over the last century human journey and desire have concentrated to its cities. The consistent availability of employment, public services, and housing is strong motivation that drives this change in human habitation. Further, cities no longer serve their population with a set of discrete services.

With the arrival of technological tools such as Internet of Things (IoT), Big Data, Cloud computing products, and Crowd Sourcing platforms, cities are becoming increasingly able to monitor the state of their infrastructure, services, and populace, cost effectively and at scale. A city that employs such technologies to mitigate the strains of urbanization, to improve the quality of life for it inhabitants, and the competitiveness of it's economy is commonly referred to as a Smart City. There are number of cities which have already done implementation of the Smart City services, such as the city of Santander in Spain.

The main problem is that many services operating simultaneously, conflicts will arise. Conflicts have both an direct effects on human life.

II. LITERATURE SURVEY

The number of sensors deployed around the world is increasing at a rapid pace. These sensors continuously generate enormous amounts of data. However, collecting data from all the available sensors does not create additional value unless they are capable of providing valuable insights that will ultimately help to address the challenges we face every day (e.g. environmental pollution management and traffic congestion management).

Realising the vision of the IoT, therefore, requires an agreed architectural reference model, based on open protocol solutions and key enabling services that enable interoperability of deployed IoT resources across different application domains and contribute to horizontal re-use of the deployed infrastructure.

The smart city vision should be realized through public-controlled integrated urban operating systems to evade vendor monopolies and offer unrestricted data to all citizens versus carving out virtual gated communities and corporate enclaves. Urban performance currently depends not only on the city's endowment of smart capital, but also, on the availability and quality of human and social capital.

© 2017, IERJ All Rights Reserved

Effective and Feasible Ways to Coordinate Urban Technologies. Rapid advances in building information technologies into the very fabric of the city while at the same time using these technologies to integrate and add value to the provision of urban services provide the mandate for the sustained development of new methods. This will involve integrating data, software and organizational forms that best improve the efficiency and competitiveness of the environment in which cities operate. The major intellectual challenge that we and the rest of society face, is embracing the idea that as we develop new digital technologies, we use those same technologies to study the processes of their application, implementation and impact on society.

III. IMPLEMENTATION

a. Big Data Acquisition Unit

Sensing remotely generated data promotes the expansion of our research system as cost-effective parallel data acquisition system to satisfy specific computational requirements. The Science Society originally approved this solution as the standard for parallel processing in this particular context. As this high volume real time data integrated more sophisticated qualifications for improved Big Data acquisition, soon it was recognized that traditional data processing technologies could not provide sufficient power for processing such kind of data. Therefore, the need for parallel processing of the massive volume of data was required, which could efficiently analyze the Big Data. For that reason, the proposed Big Data Acquisition Unit is introduced in the Big Data architecture that gathers the data from various sources around the London Transportation System. It is possible that the received raw data are distorted by scattering and absorption by various Keys and Values. We assume that the Data Provider in our case the TFL and Network rail can correct the erroneous data. For effective data analysis, real time data is preprocesses data under many situations to integrate the data from different sources, which not only decreases storage cost, but also improves analysis accuracy. Some relational data preprocessing techniques are data integration, data cleaning, and redundancy elimination. After preprocessing phase, the collected data are transmitted to a server using data feed API. The data must be corrected in different methods to remove distortions caused due to the motion of the Bus relative to the route, Station Status, Arrival and Departure, Disruption at several levels, etc.

We divided the data processing procedure into two steps, such as real-time Big Data processing and offline Big Data processing. In the case of offline data processing, the Api Data Feed transmits the data to the MongoDB for storage. This data is then used for future analyses. However, in real-time data processing, the data are directly transmitted to the filtration and load balancer server, since storing of incoming real-time data degrades the performance of real-time processing.

b. Data Processing Unit

In data processing unit, the filtration and load balancer server have two basic responsibilities, such as filtration of data and load balancing of processing power. Filtration identifies the useful data for analysis since it only allows useful information, whereas the rest of the data are blocked and are discarded. Hence, it results in enhancing the performance of the whole proposed system. Apparently, the load-balancing part of the server provides the facility of dividing the whole filtered data into parts and assign them to various processing servers but in our case we are using only one server due cost limitations.

The filtration and load-balancing algorithm varies from analysis to analysis; e.g., if there is only a need for analysis of route and Station data, the measurement of these described data is filtered out, and is segmented into parts.

Each processing server has its algorithm implementation for processing incoming segment of data from the load. Each processing Steps makes statistical calculations, any measurements, and performs other mathematical or logical tasks to generate intermediate results against each segment of data.

Since these servers perform tasks independently and in parallel, the performance proposed system is dramatically enhanced, and the results against each segment are generated in real time. The results generated by each steps are then sent to the aggregation step for compilation, organization, and storing for further processing.

c. Data Analysis and Decision Unit

Data Analysis and Decision Unit contains three major portions, such as aggregation and compilation step, results storage step, and decision making step. When the results are ready for compilation, the processing step in Data Processing Unit send the partial results to the aggregation and compilation step, since the aggregated results are not in organized and compiled form. Therefore, there is a need to aggregate the related results and organized them into a proper form for further processing and to store them. In the proposed architecture, aggregation and compilation steps is supported by various algorithms that compile, organize, store, and transmit the results. Again, the algorithm varies from requirement to requirement and depends on the analysis needs as in our case we are proposing the conflicts on different services in terms on Disruption percentage.

Aggregation also stores the compiled and organized results into the result's storage with the intention that any further steps can use it as it can process at any time. The aggregation also sends the same copy of that result to the decision-making mechanism to process that result for making decision. The decision-making mechanism is supported by the decision algorithm, which inquire different things from the result, and then make various decisions (e.g., in our analysis, we analyze Bus, tube, and Network Rail, whereas other finding such as Station, Arrival, Departure, and Disruption can also be found). The decision algorithm must be strong and correct enough that efficiently produce results to discover hidden things and make decisions. The decision part of the architecture is significant since any small error in decision-making can degrade the

© 2017, IERJ All Rights Reserved

efficiency of the whole analysis. Data Analysis and Decision Unit finally displays decisions, so that any application can utilize those decisions at real time to make their development.

The applications can be any business software, general purpose community software, or other social networks that need those findings (i.e., decision-making). The self-explanatory flowchart supporting the working of the proposed architecture.

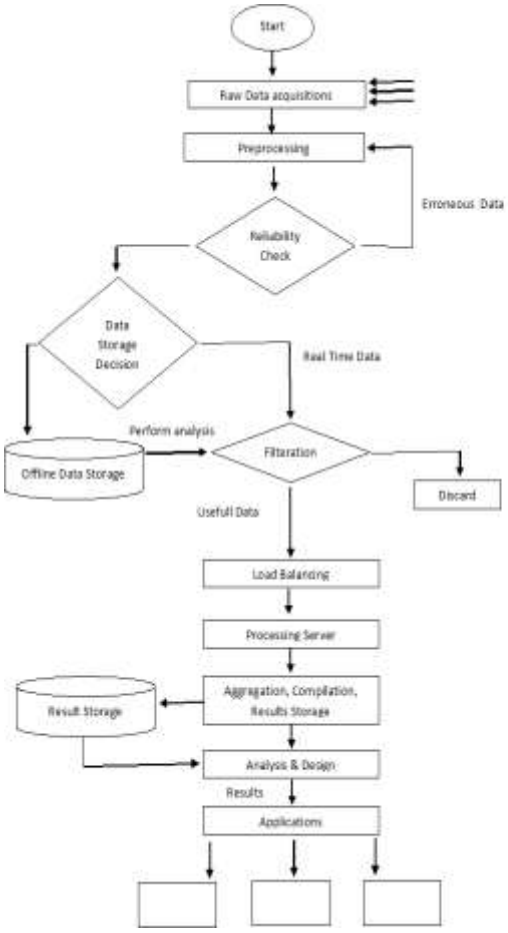

Fig: Architecture

IV. ALGORITHMS

Algorithm I. Filtration and Load Balancing Algorithm

Input: Live Data Feed process data set.
Output: filtered data in fixed size block and send each block to processing Mechanism.
Steps:
1. Filter related data i.e. processed data. All other unnecessary data will be discarded.
2. Divide the Data into Appropriate Key Value Pair.
3. Transmit Unprocessed data directly to aggregation step without processing.
4. Assign and transmit each distinct data block of processed data to various processing steps in Data Processing Unit.

Description: This algorithm takes live data and then filters and divides them into segments and performs load-balancing algorithm.
In step 1, related data is filtered out.
In step 2, filtered data are the association of different key value pairs and each pair is different numbers of sample, which results in forming a data block. In Next steps, these blocks are forwarded to process by Data Processing Unit.

Algorithm II. Processing and Calculation Algorithm

Input: Filtered Data.
Output: Normalized Disruption data.
Steps:
1. For each Network Rail Operator Performance, Categorical Data like G for good, A for average is extracted.
2. Normalize the disruption data for all the three modes.
3. Persist the data into data store and forward it.

Description: The processing algorithm calculates results for different parameters against each incoming filtered data and sends them to the next level.
In step 1, the calculation of Good and Average along with trend Furthermore, in the next step, the results are transmitted to the aggregation mechanism.

Algorithm III. Multi Modal Summarization Algorithm

Input: Normalized Disruption Data.
Output: Final result summary.
1. Gather the data from data store in normalized format.
2. Apply Summarization for Individual modal pie from the total disruption data capture.
3. Persist the final disruption summary into data store.
Description: here the data is collected and the results from each modal is processed against all and then combines, organizes, and stores these results in NoSQL database.

V. TECHNOLOGIES

Cloud

Cloud computing is a type of Internet-based computing that provides shared computer processing resources and data to computers and other devices on demand. It is a model for enabling ubiquitous, on-demand access to a shared pool of configurable computing resources (e.g., computer networks, servers, storage, applications and services), which can be rapidly provisioned and released with minimal management effort. Cloud computing and storage solutions provide users and enterprises with various capabilities to store and process their data in third-party data centers that may be located far from the user–ranging in distance from across a city to across the world. Cloud computing relies on sharing of resources to achieve coherence and economy of scale, similar to a utility (like the electricity grid) over an electricity network.

For Our Project we have created a Small Server for live Data Processing.

© 2017, IERJ All Rights Reserved

Python

Python is a widely used high-level, general-purpose, interpreted, dynamic programming language. Its design philosophy emphasizes code readability, and its syntax allows programmers to express concepts in fewer lines of code than possible in languages such as C++ or Java. The language provides constructs intended to enable clear programs on both a small and large scale.

Python supports multiple programming paradigms, including object-oriented, imperative and functional programming or procedural styles. It features a dynamic type system and automatic memory management and has a large and comprehensive standard library.

Python interpreters are available for many operating systems, allowing Python code to run on a wide variety of systems. Python code can be packaged into stand-alone executable programs for some of the most popular operating systems, so Python-based software can be distributed to, and used on, those environments with no need to install a Python interpreter.

PyMongo

The PyMongo distribution contains tools for interacting with MongoDB database from Python. The bson package is an implementation of the BSON format for Python. The pymongo package is a native Python driver for MongoDB. The gridfs package is a gridfs implementation on top of pymongo.
PyMongo 3.3 supports MongoDB 2.4, 2.6, 3.0, and 3.2.

VI. VERIFICATION AND VALIDATION

In this system, we are taking different data as input (Bus, Train, Metro). We are using LIVE FEED for tracking transport, CLOUD SERVER for storing data, and we are using WATCHDOG ARCHITECTURE to alert the Authorities.

Our system has following advantages:
1. It improves efficiency of Travelling.
2. Reducing the overhead of the passenger.
3. Reduces Disruption.

VII. CONCLUSION

Conflicting services pose serious safety threats and operational failure in a Smart City environment. This project focuses on formulating the problem of conflicts. Specifically, it (i) outlines several characteristics of services that contribute towards conflicts, (ii) proposes a conflict taxonomy in terms of origin of conflict.

In addition, a watchdog architecture is designed for intercepting actions from all services and detecting and resolving conflicts.

REFERENCES

[1] M. Ma; S. Masud Preum; W. Tarneberg; M. Ahmed; M. Ruiters; J. Stankovic., *"Detection of Runtime Conflicts among Services in Smart Cities"* DOI: 10.1109/SMARTCOMP.2016.7501688, 2016.

[2] L.Sanchez, L.Mu˜noz, J. A.Galache, P.Sotres, J. R.Santana, V.Gutierrez,R.Ramdhany, A. Gluhak, S. Krco, E. Theodoridis *et al.*, "Smartsantander:Iot experimentation over a smart city testbed," *Computer Networks*, vol. 61, pp. 217–238, 2014.

[3] C. Perera, A. Zaslavsky, P. Christen, and D. Georgakopoulos, "Sensing as a service model for smart cities supported by internet of things, "*Transactions on Emerging Telecommunications Technologies*, vol. 25,no. 1, pp. 81–93,2014.

[4] A. Bask, K. Spens, G. Stefansson, and K. Lumsden, "Performance issues of smart transportation management systems," *International Journal of Productivity and Performance Management*, vol. 58, no. 1, pp. 55–70,2008.

[5] S. Zygiaris, "Smart city reference model: Assisting planners to conceptualize the building of smart city innovation ecosystems," *Journal of the Knowledge Economy*, vol. 4, no. 2, pp. 217–231, 2013.

[6] J. Chinrungrueng, U. Sunantachaikul, and S. Triamlumlerd, "Smart parking: An application of optical wireless sensor network," in *SAINT Workshops 2007. International Symposium on Applications and the Internet Workshops, 2007.* IEEE, 2007, pp. 66–66.

[7] M. I. Ali, F. Gao, and A. Mileo, "Citybench: A configurable benchmarkto evaluate rsp engines using smart city datasets," in *In proceedings ofISWC 2015 - 14th International Semantic Web Conference*. Bethlehem,PA, USA: W3C, 2015, pp. 374–389.

[8] C. Harrison and I. A. Donnelly, "A theory of smart cities," in *Proceedings of the 55th Annual Meeting of the ISSS-2011, Hull, UK*, vol. 55, no. 1, 2011.

© 2017, IERJ All Rights Reserved