International Journal of Engineering and Advanced Technology (IJEAT)
ISSN: 2249-8958 (Online), Volume-8 Issue-4, April 2019

# The Change Data Capture and the Web Application Messaging Protocol on the Real Time Dashboard

**Herri Setiawan, Andi Abdul Adhiem Sumitro, Rendra Gustriansyah**

***Abstract*: The Telkomsel's RAFI/NARU dashboard problem in Sumbagsel Regional that was built with the LAMP web platform model caused a lag when page loading in the web browser, and blinking when the page refreshed and lacks support for the mobile roadmap. Therefore, this paper will design an alternative real time dashboard for monitoring the Telkomsel's RAFI/NARU networks by implementing the Change Data Capture and the Web Application Messaging Protocol, so that it can improve the performance of the dashboard from the request-reply paradigm to push paradigm and from HTTP to the 'publish & subscribe' on the web socket. The test results show that the presentation of data on the RAFI/NARU dashboard can occur in in real time, push paradigm, and event-driven without page refresh or page reloading, without lag when page loading in the web browser, even this dashboard can also be applied in a mobile environment by using the Angular framework javascript.***

***Index Terms*: Change Data Capture, Real Time Dashboard, The Telkomsel's RAFI/NARU, Web Application Messaging Protocol.***

## I. INTRODUCTION

Telkomsel is the largest cellular telecommunications operator in Indonesia, with more than 170 million subscribers [1] who use the RAFI/NARU dashboard (Ramadan and Eid/Christmas and New Year) to summarize information from various Telkomsel cellular network elements. This information is presented in the form of text, numbers, images, and maps used for network monitoring (Fig. 1).

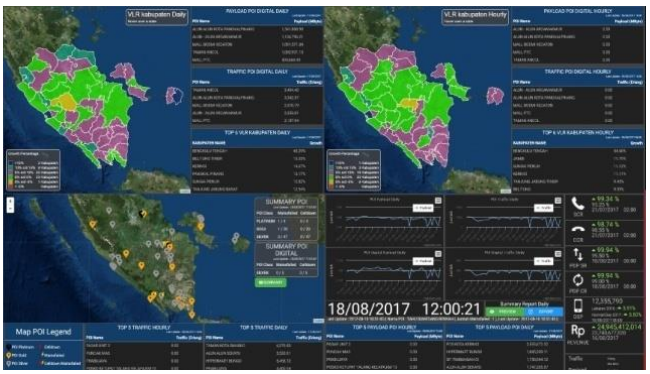


**Fig. 1: The Display of the Telkomsel's RAFI/NARU**

In the period of Ramadhan and Idul Fitri (RAFI) in 2017 alone, the projected traffic of data services (payload) is around 480 terabytes, traffic of voice service is more than 150 million minutes per day, and SMS traffic penetrates more than 40 million SMS per day [1]. However, with the RAFI/NARU dashboard, the officers who served in the 'posko siaga' RAFI/NARU period, including top management felt very helpful in monitoring the current network conditions, so that they could conduct analysis quickly and make the right decisions [2], [3]. The RAFI/NARU dashboard is a software model [4]–[6] based on PHP and MySQL that implements the LAMP web platform stack. Along with technological developments, the demands for providing real time data and developing services in the mobile environment are needed. The current dashboard system, although functioning properly, has several problems because it is built with the LAMP web platform model, namely:

1. Performance disruptions (lag) when page loading on a web browser, due to maximum TCP concurrent connection limitations per host on the same domain [7].
2. Blinking during the web page refresh, because the request-reply communication mode with the HTTP is used on the LAMP web stack platform [8].
   This request-reply communication mode is also not recommended for use in the mobile environment [9].
   Therefore, the RAFI/NARU dashboard of Telkomsel Siaga in the Sumbagsel Regional area needs to update its dashboard system based on real-time for network monitoring and support a sustainable development roadmap in the mobile environment, by designing an alternative dashboard that can implement:
   1. Real time database with push paradigm that uses the Change Data Capture method (CDC).

**Manuscript published on 30 June 2019.**
***** Correspondence Author (s)
**Herri Setiawan***, Faculty of Computer Science, Universitas Indo Global Mandiri, Palembang, Indonesia.
**Andi Abdul Adhiem Sumitro**, Faculty of Computer Science, Universitas Indo Global Mandiri, Palembang, Indonesia.
**Rendra Gustriansyah**, Faculty of Computer Science, Universitas Indo Global Mandiri, Palembang, Indonesia.

2. 'Publish & subscribe' mode communication with the Web Application Messaging Protocol (WAMP).
3. A modern client java script library that supports the concept of real time push-servers with Angular framework.

## II. RELATED WORK

### A. HyperText Transfer Protocol (HTTP)

HTTP is an application level protocol that applies the request-reply method for distributed, collaborative, and hypermedia information systems [10].
Request-reply (request-response) is a pattern of exchanging messages where the computer (client) sends a message requesting data to the data provider (server), then the server receives, processes, and answers the data request with a reply message (see Fig. 2).

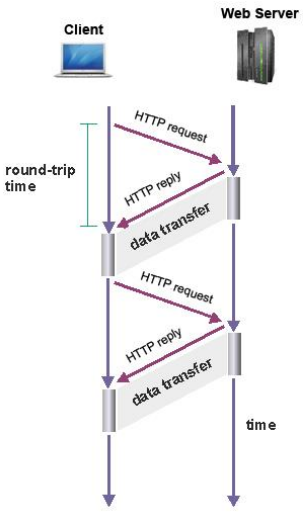


**Fig. 2: the Request-Reply method on HTTP** [10]

This message exchange pattern allows two applications to communicate with each other through a previously built connection, such as when a user is browsing on web pages that generally use HTTP.
The HTTP rule states that the number of parallel client connections to a server is limited. This rule is intended to maintain HTTP response time and prevent congestion [11]. The maximum number of parallel connections per host/server depends on the web browser used (see Table I). The more modern these web browsers, the more parallel connections to one host/server that can be done concurrently.

**Table 1: Maximum Parallel Connections per Host** [11]

| Browser | Max Parallel Connections Per Host |
|---|---|
| IE 6 and 7 | 2 |
| IE 8 and 9 | 6 |
| IE 10 | 8 |
| Firefox 2 | 2 |
| Firefox 3 to 17 | 6 |
| Opera 9.63 | 4 |
| Opera 10 | 8 |
| Opera 11 and 12 | 6 |
| Chrome 1 and 2 | 6 |
| Chrome 3 | 4 |
| Chrome 4 to 23 | 6 |
| Safari 3 and 4 | 4 |

### B. Web Application Messaging Protocol (WAMP)

WAMP is defined as a standard web socket sub-protocol that provides two application messaging patterns namely Publish & Subscribe (PubSub) and Remote Procedure Calls (RPC) for procedures implemented on the WAMP router [12].
PubSub is a messaging pattern where the subscriber component informs the router that they want to receive information about a topic (subscribe), then the publisher component does a publishing message on this topic, so the router can distribute the message to all subscribers as seen in Fig. 3 [13].

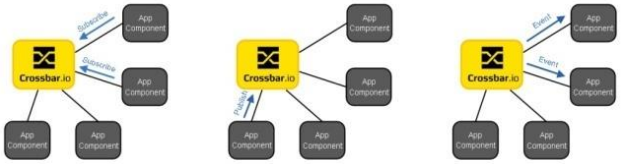


**Fig. 3: PubSub Messaging Patterns on WAMP** [13]

By using WAMP, application developers can build distributed systems from application components that are available and able to communicate in almost real time.

### C. Change Data Capture (CDC)

CDC is a technique for tracking data changes in real time. The easiest and most efficient way to track data changes in a MySQL database is to use binary logs (MySQL binlogs). CDC that uses binlogs can independently track changes made by source applications. The application of CDC that uses binlogs is also the best approach for tasks, such as audit logs, copying data to other database systems or processing the results of reading events that occur in the database, and can also be applied to database systems that have very high transaction volumes [14].
The use of CDC with MySQL binlogs is more recommended than the use of MySQL triggers, because the use of MySQL triggers can slow down applications [15].
One application that uses the MySQL based CDC binlogs technique is Maxwell's daemon released by Zendesk [16]. Zendesk Maxwell's daemon is an application that has the ability to read MySQL binlogs and write tracking results in an update, lines per line, in JSON (JavaScript Object Notation) format [15]. JSON is a text format for sharing data between servers and clients.

### D. Related Research

Adithama used the CDC-pull technique to build business intelligence in real time for the subject of academic activities at the university. The results of his research show that the ETL process (Extract, Transform, Load) using the CDC-pull approach can be done in real time, with a one-minute execution time series to appear in the client's web browser [17]. Meanwhile, Martinez-Pabon et al. examine real time connections between smart phones and multi screen interactive for public displays using WAMP. The results of them research show that the performance of prototype connections is good for four concurrent users [18].


Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication (BEIESP)

Unlike the research of Rubin and Adithama, this research will apply the CDC-push techniques based on binary logs that use WAMP with only one-way communication mode, namely from the server to the client to obtain real time information.

The binary logs approach is used for real time systems because it is more effective, efficient and superior compared to database trigger methods [15] and CDC-pull [17], especially in the speed of capture process, dynamic data source, host to host, and accuracy.

## III. ARCHITECTURAL DESIGN

The real time dashboard architecture design in this research will use the concept of asynchronous, event-driven communication modes, and push paradigm from the database server as content provider to the user's web browser as content viewer, which can be seen in Fig. 4.

On the Telkomsel's RAFI/NARU dashboard system that is running now still uses a web system with pull paradigm, which means the user action triggers the content request so that the server sends a reply. Whereas in this study, data update activities that occur in the database will be pushed to the client, without going through the request/reply process (push paradigm).

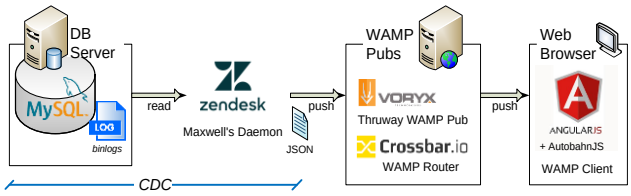


**Fig. 4: The Real-Time Dashboard Architecture**

The real time dashboard architecture design in this research is as follows:

1. The CDC technique will be applied through the activation of the binlogs feature on the MySQL database server;
2. Zendesk Maxwell's Daemon will be used for parsing and relaying data from MySQL binlogs, so that updating data on the MySQL-database server (service event-producer) will be pushed to crossbar.io WAMP Router (event-transport service) using WAMP, with output format in the form of JSON;
3. Use of Voryx Thruway WAMP Pub (JSON reader) with the 'publish & subscribe' communication mode that ensures data will be distributed from Zendesk Maxwell's Daemon to the client with push paradigm.
4. Next, the client web browser will subscribe to the data available on the WAMP Router by using the java script library (AutobahnJS and AngularJS framework) to build a WAMP client (event-consumer service).

The use of AngularJS framework as a WAMP client with the main features of 2-way data binding and promises will ensure communication is done asynchronously based on push paradigm. Because AngularJS framework has development tools that are able to produce applications that can work in asynchronous concepts with push paradigm in the mobile environment, the use of this angular java script framework will also pioneer service providers in the mobile environment.

## IV. RESULTS AND ANALYSIS

The computer network architecture used in the real time dashboard test consists of a host computer, a virtual machine for the real time dashboard, and the dashboard users as shown in Fig. 5.

Figure 5 show that the database server (MySQL) is connected to an intranet, where all clients (users' dashboards) are located, so that changes that occur in the database server can be directly pushed to the user in real-time.

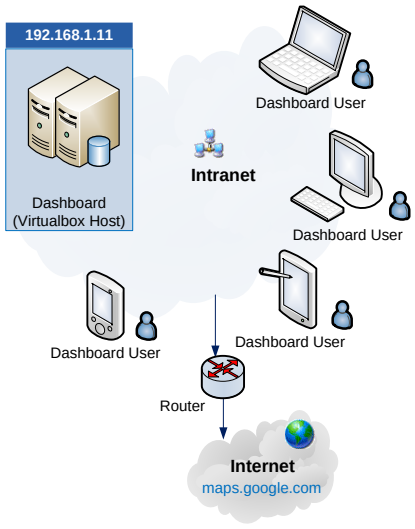


**Fig. 5: The Real-Time Dashboard Architecture on a Computer Network**

The testing process will use Chrome Developer Tools (F12) to ensure that communication occurs between the server and client browser using push paradigm, not request-reply. When a web socket (ws) is activated, the activity/transaction that occurs can be observed in the Tab Frames (Fig. 6).

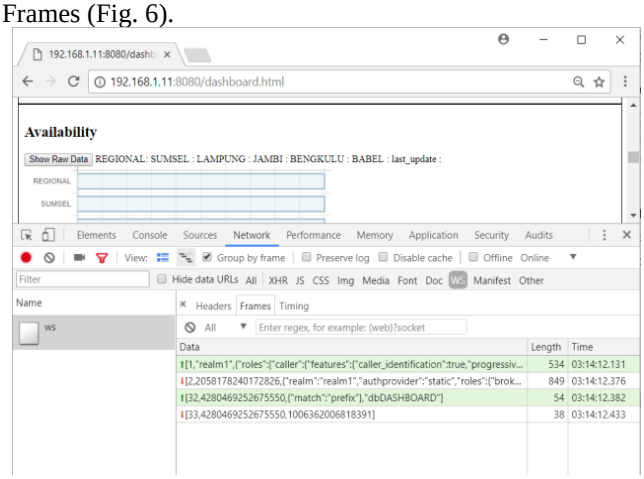


**Fig. 6: Traffic Web Socket When Loading on the Dashboard**

This testing is initiated by two transactions, namely the connection and subscribe request from the user's web browser to the WAMP Router as seen in the Frames Tab in Fig. 6. Furthermore, from the user's web browser, the data is updated in eight tables from the database dB DASHBOARD.


Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication (BEIESP)

The test results show that the presentation of data on the dashboard can be updated in real time with push paradigm (see the 'Availability' section in Fig. 7), and without page reload or page refresh (see the 'Frames' tab in Fig. 7).

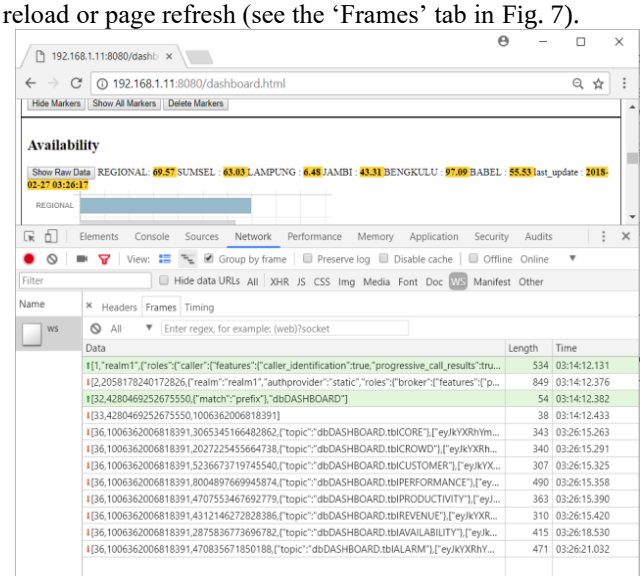


**Fig. 7: Presentation of Data on the Dashboard and Web-Socket-Push Traffic When Updating Data in MySQL**

This proves that a web dashboard that uses the Data Change Capture and the Web Application Messaging Protocol methods with push paradigm can operate in real time without requiring client requests to the server to display data on the dashboard.

Then, to test whether there is a request queue to the server, netstat tools are used. The test results show that the eight topic channel (Fig. 7) only opens two connections to the server on port 8080 as shown in Fig. 8.

```
C:\Users\adhie>netstat -anq | grep 8080
  TCP    192.168.1.12:52451     192.168.1.11:8080      TIME_WAIT
  TCP    192.168.1.12:52461     192.168.1.11:8080      ESTABLISHED
```

**Fig. 8: Test Results Using Netstat**

This means that the HTTP concurrent connection limit per host/domain on the web that uses HTTP is not found on the dashboard in this research, because the dashboard uses a web socket protocol. The absence of concurrent connection restrictions on the web that uses web sockets ensures that data activity is pushed to the user's browser in parallel without the need to queue as is the case with webs that use HTTP.

Testing is also carried out on the mobile environment by entering data on the MySQL database, then observing the results on the user's mobile browser (Google Chrome). The test results are as follows:

1. In the case of a mobile browser active as a foreground process, the mobile browser displays the presentation of data in the dashboard in real time without the need for page refresh or page reloads.
2. In the case of a mobile browser being minimized/background process, the mobile browser still receives data-stream from the MySQL database, then when the mobile browser is activated as a foreground process, the presentation of the data is immediately updated on the dashboard without the need for page refresh or page reload, as shown in Fig. 9.

The results of this test show that the technology and methods used in the design of a real time dashboard can also be applied in a mobile environment without sacrificing user experience (just like in a web browser environment on a computer).

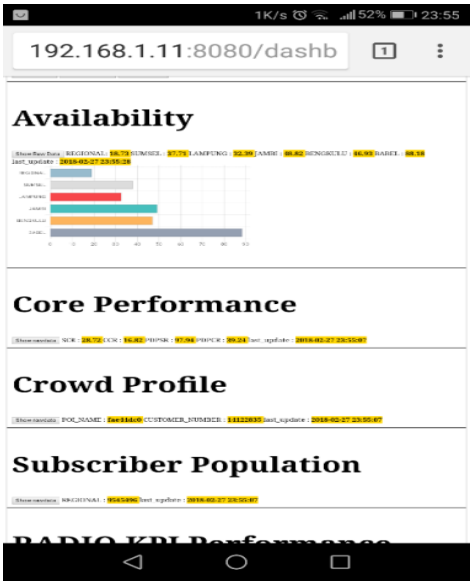


**Fig. 9: The Real Time Dashboard on the User's Mobile Browser**

## V. CONCLUSION

The test results show that by implementing the Capture Change Data and the Web Application Messaging Protocol methods on the dashboard, the presentation of data can occur in real time, push paradigm, and event-driven without page refresh or page reloading, without lag when page loading in the web browser, even this dashboard can also be applied in a mobile environment.

Future research can apply similar techniques for research that prioritizes real time transmission and presentation, such as: online gaming, Internet of Things (IoT), remote processing, notification tools, and even can involve security aspects [19], authorization and authentication that can improve performance prototype.

## REFERENCES

1. S. Post, "Telkomsel Siap Hadapi Lonjakan Trafik Idul Fitri 2017," *palembang.tribunnews.com*, Palembang, 15-Jun-2017.
2. R. Gustriansyah, D. I. Sensuse, and A. Ramadhan, "Decision support system for inventory management in pharmacy using fuzzy analytic hierarchy process and sequential pattern analysis approach," in *2015 3rd International Conference on New Media (CONMEDIA)*, 2015, pp. 1–6.
3. H. Setiawan, J. Eko, R. Wardoyo, and P. Santoso, "The Group Decision Support System to Evaluate the ICT Project Performance Using the Hybrid Method of AHP, TOPSIS and Copeland Score," *Int. J. Adv. Comput. Sci. Appl.*, vol. 7, no. 4, pp. 334–341, 2016.
4. R. Gustriansyah, D. I. Sensuse, and A. Ramadhan, "A sales prediction model adopted the recency-frequency-monetary concept," *Indones. J. Electr. Eng. Comput. Sci.*, vol. 6, no. 3, 2017, pp. 711–720.
5. A. Sanmorino, R. Gustriansyah, Terttiaavini, and Isabella, "The Toolkit of Success Rate Calculation of Broiler Harvest," *Telkomnika (Telecommunication Comput. Electron. Control.*, vol. 15, no. 4, 2017.

6. H. Setiawan, J. E. Istiyanto, R. Wardoyo, and P. Santoso, "The Use of KPI In Group Decision Support Model of ICT Projects Performance Evaluation," *Int. Conf. Electr. Eng. Comput. Sci. Informatics (EECSI 2015)*, 2015, pp. 19–20, 2015.
7. A. Abudureheman, "Creating a Cost Saving Web Application," 2017.
8. M. A. Jadhav, B. R. Sawant, and A. Deshmukh, "Single Page Application using AngularJS," *Int. J. Comput. Sci. Inf. Technol.*, vol. 6, no. 3, pp. 2876–2879, 2015.
9. A. Developers, "Reducing Network Battery Drain," *developer.android.com*, 2018. [Online]. Available: https://developer.android.com/topic/performance/power/network/index.html. [Accessed: 18-Apr-2018].
10. R. Fielding, "Hypertext Transfer Protocol -- HTTP/1.1." 1999.
11. P. Smith, *Professional Website Performance Optimizing the Front End and the Back End*. Indianapolis: John Wiley & Sons, Inc., 2013.
12. I. Zderadicka, "WAMP Is WebSocket on Steroids," *zderadicka.eu*, 2016. [Online]. Available: http://zderadicka.eu/wamp-is-websocket-on-steroids/. [Accessed: 18-Apr-2018].
13. crossbar.io, "Basic Concepts," *crossbar.io*. [Online]. Available: https://crossbar.io/about/Basic-Concepts. [Accessed: 18-Apr-2018].
14. M. F. R. Amri, I. M. Sukarsa, and I. K. A. Purnawan, "Kajian Pendekatan Binary Log dalam Change Data Capture," *Merpati*, vol. 5, no. 2, pp. 11–22, 2017.
15. A. Rubin, "MySQL CDC, Streaming Binary Logs, and Asynchronous Triggers," *dzone.com*, 2016. [Online]. Available: https://dzone.com/articles/mysql-cdc-streaming-binary-logs-and-asynchronous-t.
16. M. Daemon, "Maxwell's Daemon," *maxwells-daemon.io*. [Online]. Available: http://maxwells-daemon.io/. [Accessed: 18-Apr-2018].
17. S. P. Adithama, "Rancang Bangun Real-Time Business Intelligence untuk Subjek Kegiatan Akademik pada Universitas Menggunakan Change Data Capture," *Buana Inform.*, vol. 5, pp. 63–74, 2014.
18. F. Martinez-Pabon, "Smart TV-Smartphone Multiscreen Interactive Middleware for Public Displays," *Sci. World J.*, 2015.
19. A. Sanmorino and R. Gustriansyah, "An Alternative Solution to Handle DDoS Attacks," *J. Theor. Appl. Inf. Technol.*, vol. 96, no. 3, 2018, pp. 657–667.