Design and Implementation of Real Time and History multi-view IoT trend Display and Control System

Chuyang Gao, Shuai Zhao, Bo Cheng
State Key Laboratory of Networking and Switch Technology, Beijing University of Posts and Telecommunications,
Beijing, China
email: gaochuyang@126.com

***Abstract*—The Internet of Things technology has brought a lot of convenience to regional situation monitoring. Facing complex and interrelated monitoring methods, IoT systems need to access a variety of sensor devices. It is necessary to comprehensively display a large number of multi-source heterogeneous sensing data. The existing work can access multi-source heterogeneous sensor devices and display the sensing data of different devices on different visual interfaces. However, the information displayed by the system lacks relevance, and it is difficult to clarify the relationship between different sensor data, and it is impossible to display comprehensive information after merging multi-sensor data. Moreover, there is few work that meet the needs of query and playback of historical data.This paper designs and implements a real-time and historical multi-view IoT trend system to deal with the mentioned problems. Using the target information graph, it displays the relationship between different sensor data, and using the method combined real time and history, it proposes a buffered batch data loading algorithm to form a dual-display mode system. An IoT trend simulation platform based on B/S-MVC-MVVM architecture was built. Based on this platform, multi-source heterogeneous sensor data is accessed. The experimental results show that the system can display the relevance of the information and realize the historical data playback effect.**

***Keywords*—integrated visualization, internet of things, knowledge graph, historical data playback**

I. Introduction

The Internet of things is a network of goods, objects, people, and information exchange and communication based on the Internet of things infrastructure. Nowadays, with the rapid development of Internet of things technology, the Internet of things system, whose main functions are monitoring, alarm, target identification and tracking, has been widely used in production and life [1] [2] [3].

Facing of today's complex and interconnected monitoring methods, IoT systems need to access a variety of sensor devices, and it is necessary to comprehensively display a large number of multi-source heterogeneous sensing data [7]. The existing IoT monitoring system can access multi-source heterogeneous sensor devices and display the sensing data of different devices on different visual interfaces. However, the information displayed by the system lacks relevance . The data of different devices are completely separated on the visual interface. The system only provides single information of multiple data sources. It is difficult to clarify the relationship between different device data, and cannot display the fusion of multi-sensor data. And the technology used in the existing system can only meet the needs of real-time monitoring of a display mode. This mode emphasizes the display of situational information in real time, ignoring the query and playback of historical data. Therefore, historical data playback is another challenge [4] [5] [6].

This paper makes the following contributions to deal with those above challenges:

1) We use the target information graph and the multidimensional view to display the relationship between the different device data of the same monitoring target , and present comprehensive data combining various means.
2) We propose a buffered batch data loading algorithm to play back historical data,and based on the method combining real-time and history and GIS technology, we design a dual-display mode system. It not only displays the real-time data transmitted from the sensing layer to the application layer, but also queries and plays back historical data according to patterns such as panorama, region, and event.
3) An Internet of Things trend simulation platform based on B/S-MVC-MVVM architecture is established. Based on this system, the platform accesses multi-source heterogeneous sensing data, and combines real-time and historical integration to display the target information graph and multi-angle view. The results of the system simulation experiment can show the relevance of the information well, and show that the buffered batch loading algorithm can achieve a low-latency, high-efficiency historical playback effect, and can display the situational data in real time and history combined with the dual mode.

II. System Architecture

IoT trend display and control system studied in this paper is implemented based on B / S (browser / server) mode. System architecture is shown in Figure 1.

The server of the system refers to the web-based background system of the target cumulative recognition system. It is built using Spring Boot and organized by the Model View Controller (MVC) pattern using Java Web Spring MVC framework. [8] The Modal layer is used to encapsulate data related to business logic and uses the Object Relational Mapping

978-1-5386-8178-7/19/$31.00 ©2019 IEEE

framework (Hibernate) and the Object Graph Mapping framework (Spring Data Neo4j) to operate the relational database and the graph database to obtain the corresponding business data.The control layer defines the interface to interact with the browser and notify the model layer.

Browser side refers to the user interface displayed through the browser. It is built with the React framework and implemented by the MVVM (Model View ViewModel) pattern [9].The browser side is composed of Modal layer, View layer and ViewModal layer [10].The Modal layer is responsible for data interaction with the background.The View layer is responsible for the visualization of graphics and user interaction,The ViewModal layer is the information transfer and processing center of Modal layer and View layer [11].

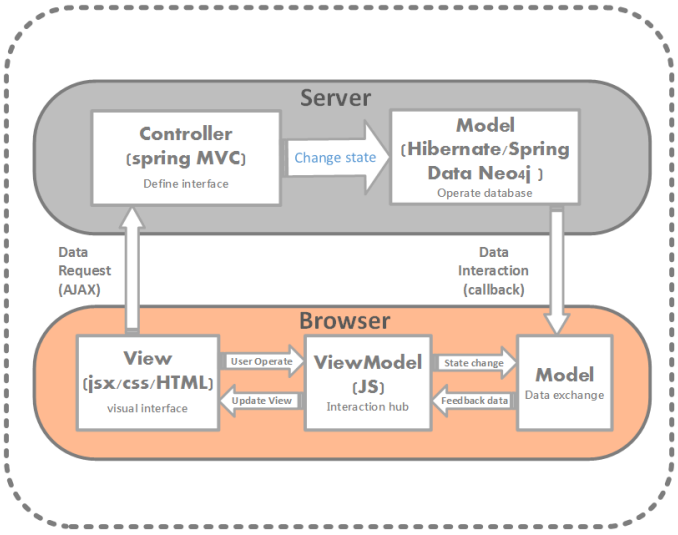

Fig. 1. System B/S-MVC-MVVM Architecture

III. Real Time and History multi-view IoT Trend System

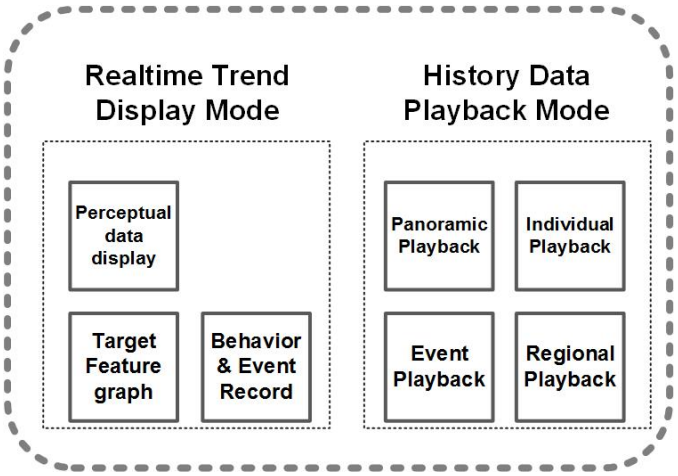

Fig. 2. Real Time and History multi-view IoT Trend System

The function modules of the IoT trend display and control system divided by business logic are listed in Figure 2.The system contains the following two modes: realtime trend display mode,history data playback mode.The features and the process of implementing the function modules and their sub-modules will be specified in the following sections.

A. Realtime Trend Display Mode

The realtime trend display module involves perceptual data display, behavior and event record, target feature graph.

The function of perceptual data display is divided into three aspects: real-time position data display, alarm flicker and detailed perceptual information card . Real-time position data display uses GIS technology to draw the mark points containing the latitude and longitude information of the target directly on the mark layer of the GIS map [12], and the latest longitude and latitude data of the target are requested from the server side by timer. Then the mark layer of the map is rendered again to refresh the real time position data of the target and the 2D dynamic moving effect of the target is presented. The alarm flicker function means that the mark points of the target will flicker at different frequencies according to the current alarm level of the target, and based on the alarm levels the marking points will be taken on different colors. The function of alarm flicker draws the mark points into different marking layers according to the classification of alarm levels, and refreshes the corresponding layers with different frequencies according to the level of alarm levels, so as to achieve the effect of flicker of different frequencies. The detail perceptual information card shows the details of each dimension of the target perceived by the system. The card is based on the ant design specification , is built by the card and table component of React framework [13], and once the mouse clicks a target marker point, A detailed perceptual information card will pop up and the data in the card will be dynamically updated.

The behavior and event record function shows the historical behavior and alarm event information of the same target associated with the current target in the target information database, and can select a target historical event or behavior for a one-click scene playback.

The function of the target feature graph shows the target information graph obtained by associating the current target information with the data of the target information database, which reflects the cognitive level of the system towards the current target. It includes the information of the target dimension type and the track information of the target, the photoelectric video information and so on.

B. History Data Playback Mode

In order to achieve historical data playback, data loading and speed control issues must be addressed [14] .

In the actual historical playback environment, we need to periodically load the frame data that needs historical playback, but if each playback moment requests the target playback frame data at that moment in the background, the system

will perform I/O-intensive data query [15]. It will increase the system burden, bring a lot of redundant network traffic, reduce the fluency of the entire system historical data playback, and greatly reduce the playback efficiency [16]. In order to solve this problem, we use the method of buffering batch data [17]. During the buffering process, the system will buffer multi-frame data at one time, and each frame includes the fusion sensor data of all monitoring targets at the current playback time, thus the method reduce the system burden and redundant network traffic. The fluency of historical data playback greatly increase [18].

In the historical data playback process, the time of buffer the batch data is very important for the playback effect. It is assumed that T_i is the time required to buffer the ith frame data during buffering data, T_b is the total time of buffering batch data, and w is the number of frames to be buffered. , then the time of buffering batch data once is shown in (1).

$$T_b = \sum_{1}^{w} T_i \tag{1}$$

In the actual network environment, it is assumed that T_{net} is the theoretical time required to transmit w frame data under the current stable network bandwidth condition, U_i is the frame delay between transmitting adjacent two frames of data, and $\overline{T_b}$ is the average time of buffering batch data once, then the average time of the buffering process is shown in (2).

$$\overline{T_b} = T_{net} + \sum_{1}^{w} U_i \tag{2}$$

In the actual network environment, network delay and some other factors are variable and unpredictable. In order to approximate the average buffer time, a factor can be added to smooth the frame delay and delay fluctuation. It is possible to estimate the time delay and delay fluctuation by using a method in which the historical value accounts for a large proportion and the current value accounts for a small proportion. It is assumed that U_f is the historical value of the frame delay at time f, U_l is the current value of the frame delay at time l, and l is greater than u, $|U_f - U_{f-1}|$ is the historical value of the delay fluctuation. , $|U_l - U_{l-1}|$ is the current value of the delay fluctuation, γ is the smoothing factor , then the average buffer time is shown in (3).

$$\begin{cases} \overline{T_b} = T_{net} + \gamma U_f + (1-\gamma) U_l \\ +\gamma \left| U_f - U_{f-1} \right| + (1-\gamma) \left| U_l - U_{l-1} \right| \\ \left| \gamma - \frac{3}{4} \right| < \frac{1}{4} \end{cases} \tag{3}$$

Buffering time is very important for the effect of playback. If this value is set small, it will hardly be fluent. If this value is set large, the delay is too big, and the real-time performance is worse. Therefore, the buffer time needs to be adjusted according to network delay and delay fluctuation.

For the playback speed control problem, we control the speed adjustment factor to control the time difference of the fusion sensor data records contained in two adjacent frames, assuming that T_u is the frame data time difference, speed is the playback speed factor, and c is the time constant we define, then the frame data time difference is shown in (4).

$$T_u = speed \times c \tag{4}$$

The historical playback speed is controlled by the acceleration and deceleration buttons. When the acceleration or deceleration button is clicked, the system will modify the playback magnification. The larger the speed factor, the larger the time difference between the combined sensing data recorded in the two frames, and the faster the actual playback speed. Therefore, we realize the control of historical playback speed.

In order to buffer batch data during historical data playback, we propose a buffered batch data loading algorithm that coordinates frame data rendering and batch frame data loading using a single buffer.It stats with the state of current playback S, the start and end frame time of batch data loading,O and E, play speed factor r, the difference of adjacent frame time D to generate the matrix batch data M.

Algorithm 1 Buffer Batch Data Loading Algorithm

Input:
The state of current playback S.
The start frame time of batch data loading O.
The end frame time of batch data loading E.
Play speed factor r.
The difference of adjacent frame time D.

Output:
The matrix Batch data M.

```
Create a buffer queue Q;
while (S == True) do
    get current time C := getCurrentTime();
    if (getBufferQueueElement(Q, C) == null) then
        while (getBufferQueueElement(Q, E) == null) do
            Q ← bufferBatchData(O, E, D, r);
        end while
        renderFrameData(getBufferQueueElement(Q, C));
        M := getBatchDataMatrix(Q, O, E);
        Return M;
    else
        renderFrameData(getBufferQueueElement(Q, C));
    end if
    C := C + D;
end while
```

Firstly, the algorithm create a buffer queue Q to store the playback target information data corresponding to each time. The index of the queue is the offset of each frame data time relative to the start frame time of batch data loading O (converted into seconds). When the user clicks the start, acceleration, deceleration or drag progress bar, the click event is triggered and the timer is opened,then it will set the state of current playback S true. Next, the current playback time C

is recorded, and the offset between the current time and the playback start time is calculated, according to which to access the Playback data (i.e.$getBufferQueueElement(Q, C)$) . If the playback data exists, system will playback the target information data at the current time (i.e.$renderFrameData(getBufferQueueElement(Q, C))$). If the data does not exist, system will start batch buffering all data within the next buffer time window (i.e. $bufferBatchData(O, E, D, r)$). Finally, after the buffering is complete, It will playback the data of the current time and adds the difference of adjacent frame time D, and wait until the timer's next loop time to re-determine whether the state of current playback S is true.

We can select different playback modes according to the query needs, the system supports four different playback mode:panoramic, individual, event and region.

IV. Evaluation and Result

A. Experimental Environment

In this experiment, the simulation platform topology is shown in Fig. 3:

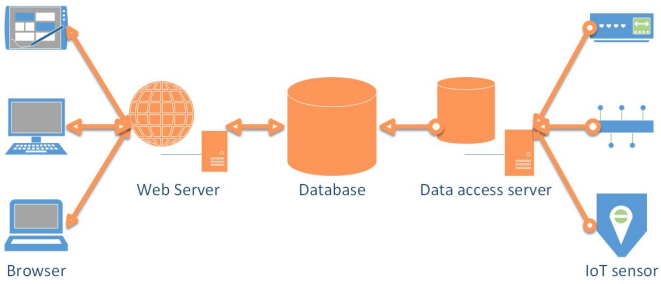

Fig. 3. Simulation Platform Topology

- Browser: Each vendor's browser. Responsible for direct visual interaction and operational interaction with the user.
- Web Server: The tomcat-6.0.45 server installed on the PC. Mainly responsible for providing various web services, background data interfaces and command delivery functions.
- Database: The mysql and neo4j database installed on the PC. Store all types of data.
- Data access platform: processing the data collected by the IoT device, obtaining some fusion results, and providing the original data together with the processed data to the database.
- IoT sensor: all kinds of low-level sensors are responsible for detecting areas and collecting relevant data information.

B. Test Case

This experiment shows the effect of the target information graph of the real-time monitoring mode, and the effect of the historical data playback mode.

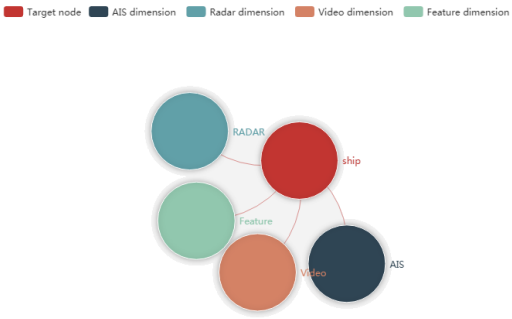

Fig. 4. Target Information Graph

The Fig. 4 shows that in the real-time monitoring mode for the ship target, we can currently associate the data of the four devices to this target, showing the topological relationship between the data of each dimension of the target.

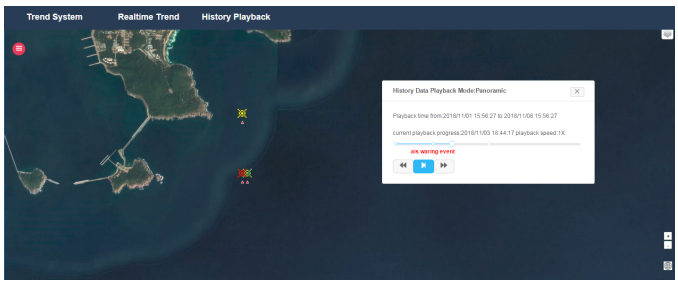

Fig. 5. History Data Playback

The effect of playing back the historical data of the three targets in the panoramic playback mode is shown in Fig. 5. On the progress control panel, the playback can be accelerated, decelerated, started, paused, dragged, and the like. The current playback progress, start and end time can be seen.

Fig. 6 and Fig. 7 show the operation effect of the buffered batch data loading algorithm during historical playback, in which 30 frames of perceptual data are buffered, and each frame contains sensor fusion information of 20 targets.

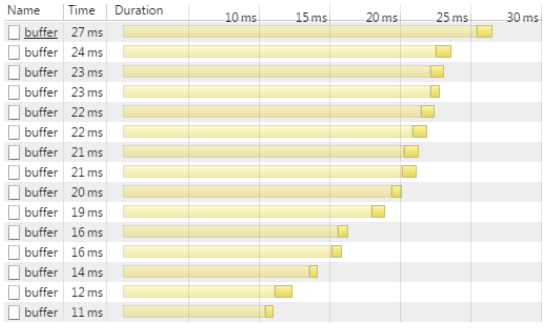

Fig. 6. Duration of Buffer Batch Data Loading Algorithm

Fig. 6 shows the buffer wait time for 15 times buffering. It buffered 30 frames of data at a time, and the average buffer time is 19ms. Fig. 7 shows the timeline for buffering batch

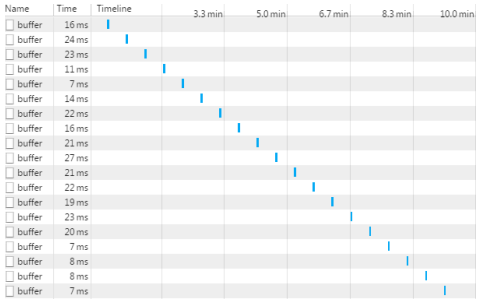

Fig. 7. Timeline of buffering batch data

data within 10 minutes, where the time difference between adjacent frame data is 1s. Each time 30 frames of data are buffered, that is, it is performed every 30 seconds according to the buffered batch data algorithm, and the actual effect is buffered 19 times in 10 minutes, which is in accordance with the algorithm expectation.

The test results show that for the update frequency in the second level, the data back buffered by the buffered batch data loading algorithm is timely, and no obvious block occurs. The buffered data does not interfere the user experience, and the low latency requirement is achieved.

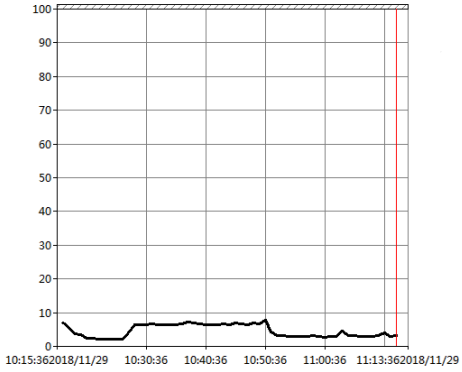

Fig. 8. The CPU usage rate

The system needs to run for a long time to monitor the situation of the target area, so the system has certain requirements for stability. The system stability test method is to perform the stability test of the system for more than one week after the system is started for several times within half a year after the deployment is completed. The CPU usage rate in one hour after the last long-running operation is described as an example, as shown in Fig. 8. The server CPU occupies normal operating system resources, and the overall test results indicate that the server has no memory leaks, disk read and write congestion, and so on. After long-term operation, the system's resource occupancy rate and response speed did not change much. Stability meets the requirements.

C. Result and Analysis

In this experiment, different parameters have an impact on the real-time performance and fluency of historical playback. Take the number of frames buffered once as an example, Fig. 9 shows the effect on history data playback fluency and real-time performance by different numbers of frames buffered once.

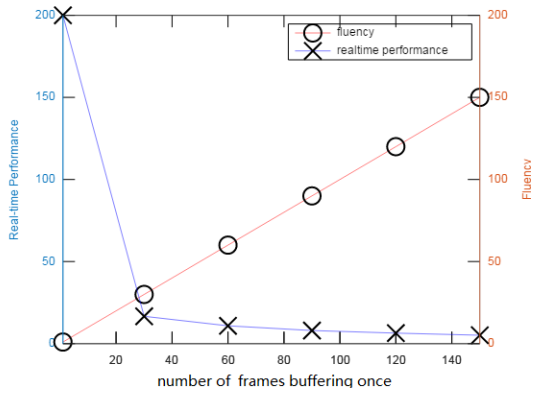

Fig. 9. The effect on history data playback fluency and real-time performance by number of frames buffered once

Obviously, as the parameter increase, the fluency will increase, but the real-time performance will decrease, because the increase in the amount of buffered data leads to a decrease in the playback block,and an increase in the buffer delay, which reduces the playback real-time performance.

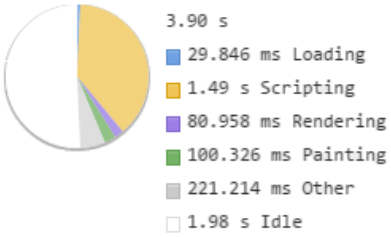

Fig. 10. The time required for each add-on

We use the Chrome browser's Timeline Developer Tools to analyze the loading time. And the time required for each add-on is recorded by means of page refresh. Fig. 10 shows one of the analysis screenshots. According to the 2/5/10 principle, the response within 2 seconds is considered by the user to be a "very attractive" user experience. The response is a "good" user experience in 5 seconds, while 10 seconds is considered a "pass" user experience. If there is no response for more than 10 seconds, the user will consider the request to fail. The test result shows that the average response time of the system page is about 4 seconds, and the difference between each test result and the average value is small. According to this principle, the system is good.

We also use the Timeline tool to count the memory footprint of each component of the page. The statistics mainly include JavaScript heap storage, documents, nodes, and listeners. Since the system uses the Ajax short polling method to locally update the page without flickering data, the memory usage of each component of the page is also periodic, as shown in the Fig. 11.

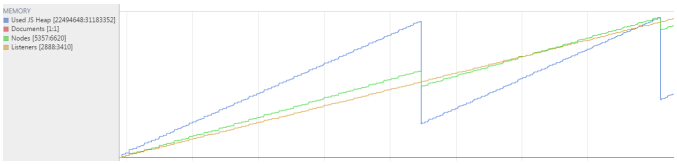

Fig. 11. The memory footprint of each component of the page

As can be seen from this analysis, JavaScript memory usage is strictly floating in the cycle, there is no hidden danger of memory leaks.

V. Conclusion

This paper proposes a technical solution for an IoT trend system combining real-time and history. Experiments show that the system can meet the needs of displaying the relevance of sensor data and history data playback. The parameters of history data playback must be determined according to the actual situation, in order to ensure the balance of real-time performence and fluency.

Acknowledgment

This work is supported by National Natural Science Foundation of China under Grant No.61501048, and by the Fundamental Research Funds for the Central Universities under Grant No. 2017RC12.

References

[1] B. Cheng, D. Zhu, S. Zhao, and J. Chen, Situation-aware iot service coordination using the event-driven soa paradigm, IEEE Transactions on Network and Service Management, vol. 13, no. 2, pp. 349361, June 2

[2] G. W. Liu, Y. X. Liu, Y. G. Ji, and C. Wang, Track association for high-frequency surface wave radar and AIS based on fuzzy double threshold theory, Systems Engineering and Electronics, vol.38, no. 3, pp. 558-

[3] Y. L. Zhang, and J. Y. Wang, Overview of the Multi-sensor Data Fusion Technology, Ship Electronic Engineering, vol.33, no. 2, p. 41, 2013.

[4] J. Kang, The key technology research of Multi-sensors Information Fusion, Harbin Engineering University, 2013.

[5] J. Dong, Research and implementation on support batch computing and streaming computing Big Data System, Northwest University, 2015.

[6] X. Y. Zhu, and P. P. Pang, Review of Big Data Stream Computing System, Group Technology & Production Modernization, no.4, pp. 50-53, 2016.

[7] Duan Y , Shao L , Hu G . Specifying Knowledge Graph with Data Graph, Information Graph, Knowledge Graph, and Wisdom Graph[C]// IEEE International Conference on Software Engineering Research. IEEE, 2017.

[8] Information on https://en.wikipedia.org/wiki/Model-view-controller

[9] Zhang D , Wei Z , Yang Y . Research on Lightweight MVC Framework Based on Spring MVC and Mybatis[C]// Sixth International Symposium on Computational Intelligence and Design. IEEE, 2014.

[10] Li X , Chang D L , Peng H , et al. Application of MVVM design pattern in MES[J]. 2015.

[11] Information on https://github.com/facebook/react/

[12] Jia PingLiu Juhai,Wang Yuan. GIS Based on Cloud Computing and Internet of Things [J]. Information Forum,2012,6

[13] Information on https://ant.design/docs/react/introduce-cn

[14] Su Y Y , Liu P C , Kao C C . Efficient Buffer Output Control for Multimedia Stream Playback[C]// IEEE International Symposium on Multimedia. IEEE Computer Society, 2013.

[15] Y. Peng, Y. Xu, and H. B. Jin, An analysis of time registration in multi-sensor data fusion system, Radar & Ecm, no. 2, p. 17, 2005.

[16] M. Mancioppi, O. Danylevych, D. Karastoyanova, and F. Leymann, Towards classification criteria for process fragmentation techniques, in Business Process Management Workshops. Springer, 2012, pp. 1 1

[17] Yi-Yu Su, Pin-Chuan Liu, and Ching-Chun Kao, "Efficient startup segment selection and scheduling for mesh-pull peer-to-peer live streaming system," IEEE 1st Global Conference on Consumer Electronics(GCCE), pp.450-451, 2-5 Oct., 2012.

[18] Yan Li; A. Markopoulou, J. Apostolopoulos, N. Bambos, "Content-Aware Playout and Packet Scheduling for Video Streaming Over Wireless Links," IEEE Transactions on Multimedia, vol.10, no.5, pp.885-895, Aug. 2008.