University Notification Subscription System using Amazon Web Service

Babur Hayat Malik, Zaheer Mehmood Dar, Sabah Mubarik Kayani, Mahnoor Dar, Muhammad Hassan Shafiq, Imran Kabir, Fatima Masood, Hamna Zakriya, Asad Ali
Department of Computer Science and Information Technology,
University of Lahore
Gujrat Campus, Pakistan

***Abstract*—Publish-Subscribe (Pub-Sub) system is an asynchronous communication service widely used in server-less and micro-services architecture. In a Pub-Sub system, publisher publish message to a topic that is immediately received by all of the subscribers of that topic. Nowadays, students face number of problems regarding admission details, assignments, offered courses, fee schedule, etc. Many a times, they missed the deadlines and it affects their studies. This paper is focused on issues faced by students regarding message delivery, duplication of data and heavy traffic, etc. It should be overcome by using amazon web services to make optimize product and to make it flexible for university Pub-Sub system. Implement the cloud services by using hybrid technique, i.e., content based and topic based architecture. It also explained the multitude use-case of university notification system which leads to make it more adaptable as subscriptions are identified with particular data content.**

Keywords*—Publish-Subscribe system; content and topic-based; university notification; Amazon web services

I. Introduction

A Publisher-Subscriber system defined as Pub-Sub is in a correspondence worldview generally utilized to give occasion scattering between inexactly publishers (distributers) and subscribers (follower) [1]. Distributers must distribute the message which are coordinated and conveyed by Pub-Sub operators (called brokers) to subscribers in light of their enrolled subscriptions. The main focus is to put on coordinating and conveying a high throughput of occasions to these steady subscribers [13]. The representative speaks with various substances (e.g., distributers and follower), coordinates the reasonable client prerequisite, and transmits clients' information [14]. Pub-Sub system implements on application layer of OSI model.

In Pub-Sub system, the distributed applications provide instant event notifications. This model allows event-driven architecture and asynchronous processing for performance improvement, reliability, scalability and versatility [2]. Gregor Hohpe and Bobby Woolf, defines the Competing Consumers design as "Competing Consumers are numerous clients that are altogether to get messages from a Point-to-Point Channel. At the point when the channel conveys a message, any of the buyers could possibly get it. The information system figures out which customer really gets the message, yet as a result the buyers rival each other to be the collector. Once a customer gets a message, it can delegate to whatever is left in its application to help process the message."

There are different types of Pub-Sub system like, content based, type based and topic based. A more adaptable yet additionally complex worldview in the Pub-Sub conspires is content-based membership. It gives greater adaptability to the supporter by giving more control in buying in an occasion in view of the genuine substance of the occasion. It enables endorser of force set of limitations as condition in shaping an inquiry on an occasion warning (otherwise called channel). Making a notice utilizing a channel gives supporters a more refined route for buying in occasions.

The features of Pub-Sub system are Push delivery by using multiple delivery protocols, fan-out, filtering, durability and security. The main purpose of using Pub-Sub system is push delivery and message durability. To develop Pub-Sub system, need messages from authentic sources. Students can receive the reliable data. This paper will discuss the categories of Pub-Sub system, problem statement; propose system and its working and discussion and conclusion.

II. Categories of Pub-Sub System

There are three main categories of Pub-Sub system used to implement any system, i.e, named as: content based, type based and topic based.

A. Content-Based

In content based Pub-Sub system, publisher can publish the content and subscriber follows the content as per need. In this system, publisher publishes the content on the message system. After this, message broker broke the message to know message content. Then send this message to the particular subscribers who want to know about this content. Message brokers use filtering pattern that applied on consumers subscription to elect events by using a subscription language (constraints < >) [3].

In the below Fig. 1, publisher publishes the message over Pub-Sub system [25]. Then message delivers to the concerned subscriber [22].

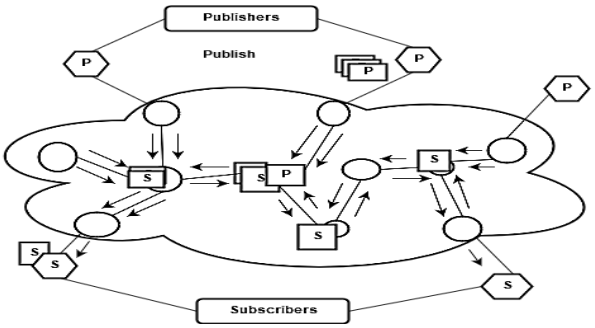

Fig. 1. Content based Pub-Sub system [25].

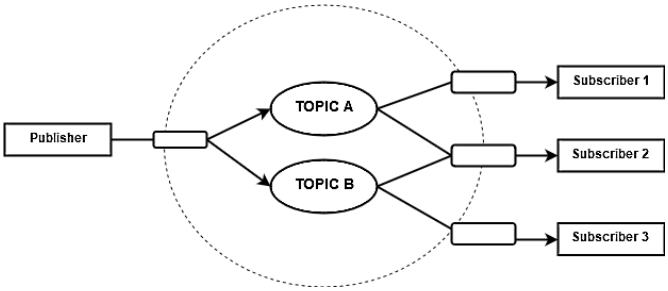

Fig. 2. Topic based Pub-Sub system [12].

B. Topic-Based

Topic based Pub-Sub design; sender sends messages with the name of a topic. The message is send to the message system and then delivered to all the applications that want to receive messages on that topic [6].

In Fig. 2, publisher publishes topic A and topic B and subscribers subscribe the topic according to their interest. One subscriber can subscribe many topics as per need [12].

C. Type-Based

In type based Pub-Sub defined events in its interface [12]. It is based on static and dynamic schemas as shown in Fig. 3. To implement this Pub-Sub system, need languages which support structural reflection. For this, no need for specific events (e.g. Java introspection). In other languages, event can subtype an introspective event type.

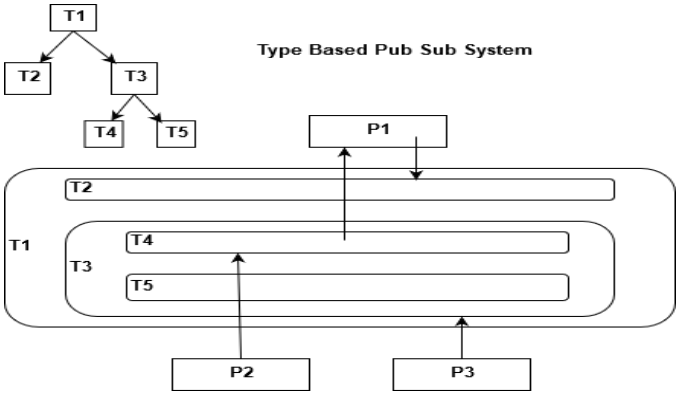

Fig. 3. Type based Pub-Sub system [12].

III. Problem Statement

University Pub-Sub system is particularly dealing with all major entities of system e.g. Admin, Teacher and students. The propose application supports entire range of notification data to facilitate our subscribers on other side they also face some technical issues regarding the implementation of Pub-Sub architecture on university subscription system. Some of them are mentioned below.

A. Delivery Notification Unguaranteed

Perfect knowledge is obvious for publisher but in this system publisher message status is not guaranteed. As publisher is unaware by the system delivery service, that may cause loss of some important notification for both subscribers and publishers [24].

B. Decreased Performance

Public systems accessibility cause high risk of unattended attacks, broker can be vulnerable to attack this system easily. In propose system use case there is a large amount of register subscribers (HR, Teachers and Students) that makes system overloaded. Highly communicative, but on runtime it requires complex protocols to implement subscription functionality [2].

C. Redundant Data

Publish-Subscribe system offers multiple instances of loggers that can run concurrently looks identical. However, in system designs it allows for a high level of redundancy. Such replications in data make it persistent [17].

D. Inflexible Data

The propose system then firstly make its paper prototype then after analysis, propose the structure then it is become difficult to change when system architecture already established. With a specific end goal to change the structure of the messages, the greater part of the system must be adjusted to acknowledge the changed arrangement.

IV. Proposed System

In this section, propose a Pub-Sub system for educational institutes. At first, describe simple working of Pub-Sub system then explains the architecture how it would be implemented in any use case or helps users to follow proper university subscription system.

A. Pub-Sub system

Publish-Subscribe is a software design pattern that describes the relationship between the flow of users, services or services of all publisher messages as shown in Fig. 4. The so-called Pub sub usually works this way: the publisher (i.e. any data source) pushes the user in the message (i.e. the data recipient) by streaming interest in a real-time feed called a channel (or topic). When a new message is posted on this channel, the subscribers of all the specific publisher channels are notified immediately and the message data (or payload) is received along with the notification. In daily use, especially in the Internet of Things, automation, network operations or distributed cloud environments, an intermediate layer called a message broker is usually required to handle the distribution

and filtering of messages, and also provides a low latency messaging private network infrastructure [18].

In proposed Pub-Sub system used topic based architecture. In topic based architecture, publisher publish message on any topic using Pub-Sub system. Then message broken works and sends it to the concerned subscriber.

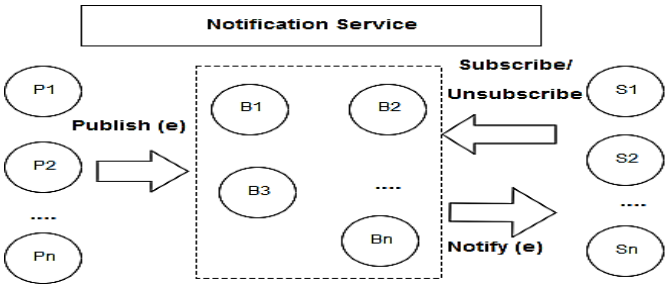

Fig. 4. Subscription model [5].

Centralized event breaking system is a key factor of current Pub-Sub systems framework that relies on a single event broker. If it working well and its working is down then the event propagation will be negotiated within the current framework whereas system vulnerability of whole system would be enhanced if depending on a single event broker [4].

In order to decrease the liability additional architecture can be made, such as received messages receipts receiving. With that component included, feedback can be given to the distributer with regards to the status of the subscriber. It is more adaptable as subscriptions are identified with particular data content and, thus, every packet of information can really be viewed as a solitary dynamic consistent channel. This exponential enlargement of Potential logical channels enlarges exponentially has changed the implementation level working of Pub-Sub system. The pub- sub operational models description is given in Fig. 5.

Event notification Pub-Sub communication system and the compatible web service technology giving a combination of emerging technology named as web service based notification system [7]. Event driven and service oriented architecture both are emerging technology giving attention to web service based notification system [11]. It helps in integrating applications either within or outside the organization. Web service Event [20] and Web service-Notification [21] are two major specifications for such systems.

Create Channel	POST/channel
Subscribe Channel	Subscribe
Publish Events	Publish
Read Events	Get Event Messages
Unsubscribe Channel	Delete or Unsubscribe

Fig. 5. Pub-Sub operational model [5].

B. University Pub-Sub Architecture

The proposed system is implemented by the Amazon Simple Notification Service (Amazon SSN). This is a web service that enables end quote or managing or sending messages to subscribe to customers In Amazon SMS, two types of customers - publishers and consumers - also known as producers and consumers [15]. Publishers send message asynchronously with subscriptions to create messages and send messages on logical access points and topics of communication channels. Subscriber (i.e. web servers, email addresses, Amazon SQS queues, AWS functions) one of the support protocols for message or notification (such as MMS, SMS, HTTP/S, Email). Subscribers to a topic Publish and receive messages on a specific topic through a message service provider. Extensibility is achieved by distributing topic sets in a number of message providers or through a cluster of providers. Different ways of communicating are topic-based university Pub-Sub middleware, such as Amazon SNS server [16]. Among these, the filter group that specifies the event subject by the subscriber connected to the publisher through the broker network.

Then, the Pub-Sub middleware is responsible for forwarding the publisher's events to relevant users throughout the network. Event filters distribution, matching process to achieve high scalability, among a large number of brokers [8]. A diagrammatic view of university pub-sub architecture is shown in Fig. 6.

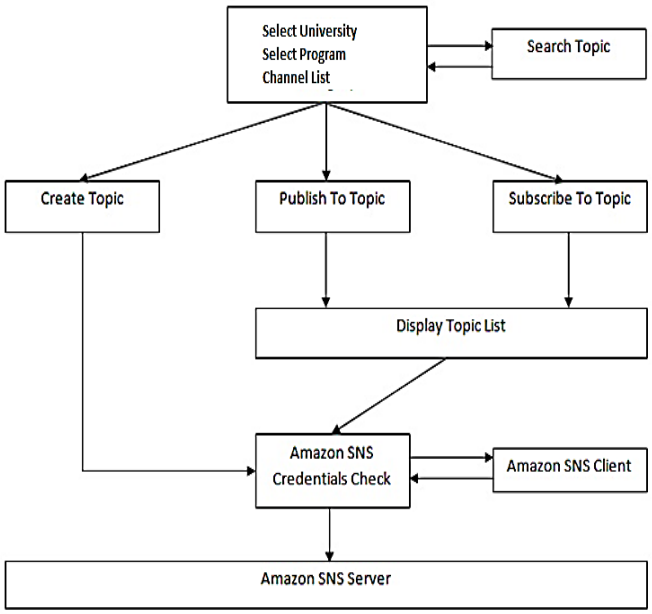

Fig. 6. University Pub-Sub architecture [19].

V. University Pub-Sub Proposed Design

Here is the detailed design of system working as shown in Fig. 7 It is actually a hybrid (content and topic based) publisher-subscriber system.

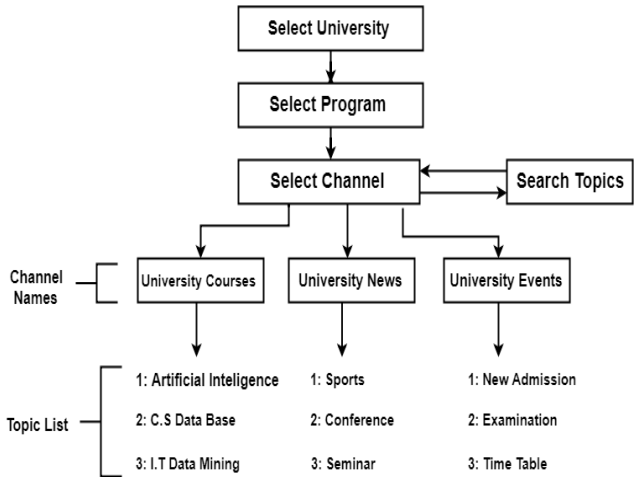

Fig. 7. University Publish-Subscribe proposed design [19].

VI. University Pub-Sub System Use Case

It depicts overall university notification system working. Defining proposed system scope where two main actor's publisher and subscriber play an important role. System is divided into two main modules, i.e. publisher as an admin module or subscriber as a user module. Fig. 8 depicts the use case for university pub-sub system use case.

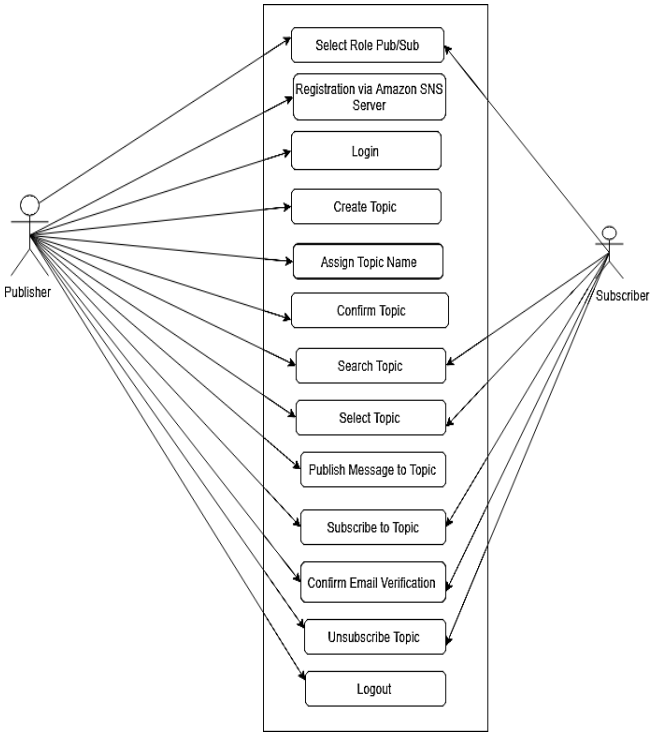

Fig. 8. University Pub-Sub use case diagram [9].

A. University Pub-Sub Actor's Use Cases

There are number of actors in proposed system which plays different roles as a publisher or subscriber.

1) New publisher use case diagram

Fig. 9 shows the pictorial representation of new publisher use case.

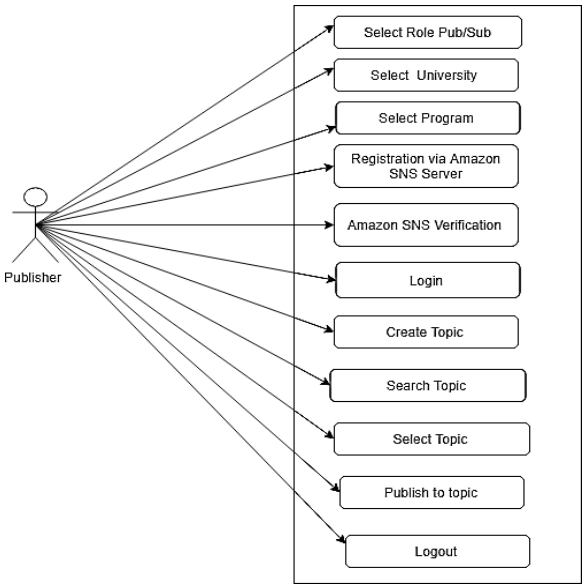

Fig. 9. New publisher use case diagram [10].

2) Teacher Use Case

In publisher module user has a right to publish any content to relevant topic or also subscribe channels within same frame. All use case activities described in the following Fig. 10.

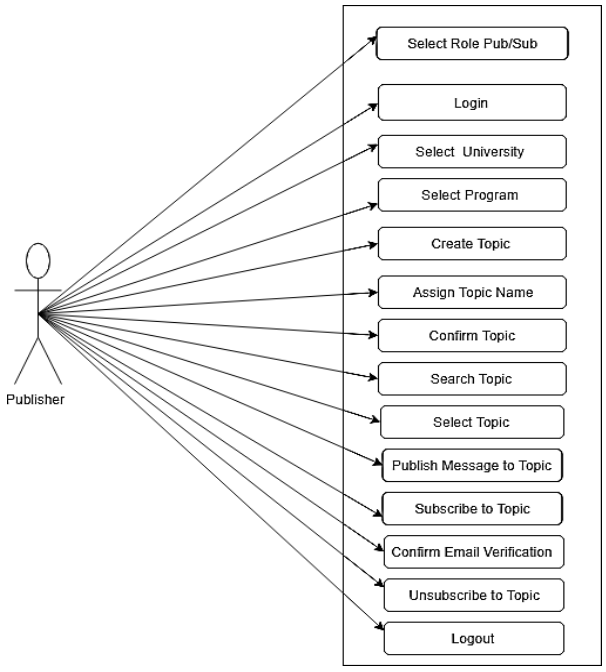

Fig. 10. Teacher as publisher role use case diagram [9].

3) Student Use Case

Student would always be in the subscriber module unless university management authorize him any rights as shown in Fig. 11.

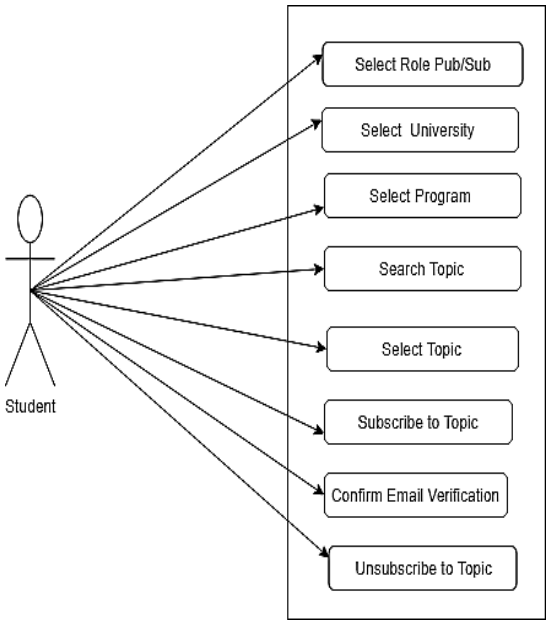

Fig. 11. Student as subscriber use case diagram [10].

4) Examiner Use Case

Here examiner is a person who has a right to publish any examination related news to relevant channel. If he is new registered user then he will first create the topic then publication should be done. The roles of examiner as publisher are shown in Fig. 12.

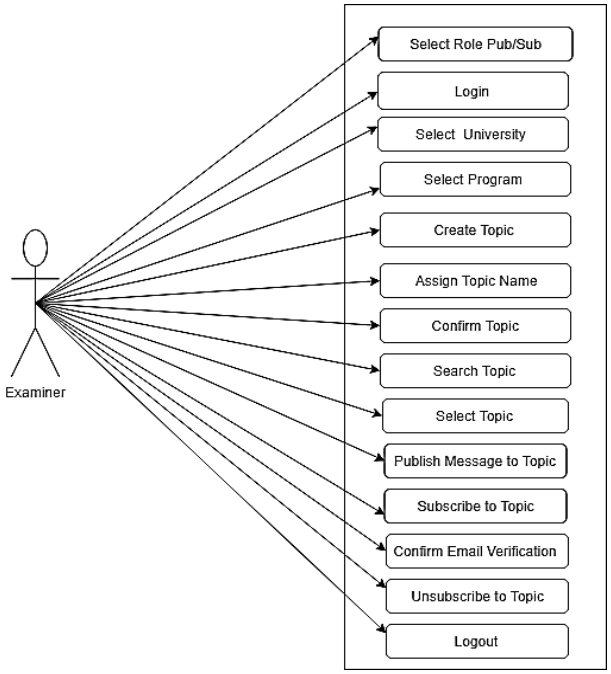

Fig. 12. Examiner as publisher use case diagram [9].

5) Teacher as a Subscriber Use Case

As mentioned earlier in the use case of teacher role. He can also subscribe to channel number of activities mentioned in the use case in Fig. 13.

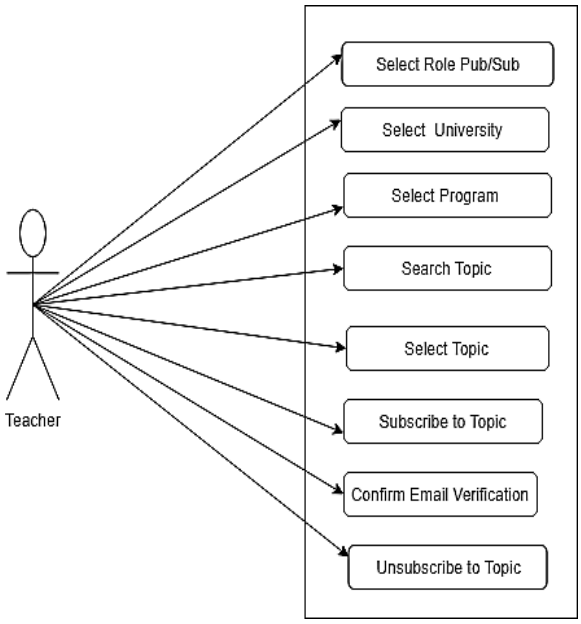

Fig. 13. Teacher as subscriber use case diagram [10].

VII. DISCUSSION

After extensive analysis, some findings that loosely coupled modules, architecture flexibility and reliability of system should lead to a model Pub-Sub system to smoothly run university affairs that provide quality time delivery of event notification.

A. Flexibility in Architecture

You can definitely include more endorsers and add group extension if message creation supplants message utilization without code change. That's why in Pub-Sub model there should be low coupling between each module as message passed to the subscribers by the publishers without any trouble because it makes system highly independent.

B. Configuration Ease

Pub-Sub system is easy to configured and also support different models of subscriptions. It helps publishers and subscribers to distribute and receive the messages using fewer resources by providing offline notification [23].

C. System Service Availability

The communication framework transports the distributed messages just to the applications that are bought in to the relating subject. Since various physical endorsers can have a similar membership, there's ensured bolster for high accessibility, for the subject itself as well as for the theme supporters.

VIII. CONCLUSION

In order to decrease the liability additional architecture can be made, such as received messages receipts receiving. With that component included, feedback can be given to the distributer with regards to the status of the subscriber. Because subscriptions with specific data content are identified, they are more adaptable which helps to make system more flexible, highly configured and proper distributed system for reliable

communication thus, every packet of information can really be viewed as a solitary dynamic consistent channel. This exponential enlargement of Potential logical channels enlarges exponentially has changed the implementation level working of Pub-Sub system.

REFERENCES

[1] Joao Paulo de Araujo, Luciana Arantes, Elias P. Duarte Jr., Luiz A. Rodrigues, Pierre Sens, "A Publish/Subscribe System Using Causal Broadcast Over Dynamically Built Spanning Trees", 29th International Symposium on Computer Architecture and High Performance Computing, 2017.

[2] Tarek R. Sheltami, Anas A. Al-Roubaiey, Ashraf S. Hasan Mahmoud, "A survey on developing publish/subscribe middleware over wireless sensor/actuator networks", Springer Science+Business Media New York 2015

[3] César Canas, et. Al, "Self-Evolving Subscriptions for Content-Based Publish/Subscribe Systems", IEEE 37th International Conference on Distributed Computing Systems, 2017.

[4] Hiroki Nakayama, Dilawaer Duolikun, Tomoya Enokido, Makoto Takizawa,"CausallyOrderedDeliveryofEventMessageswithKeywordVectorsinP2P Publish/SubscribeSystems", IEEE 29th International Conference on Advanced Information Networking and Applications, 2015.

[5] Muhammad Agus Triawan, Hilwadi Hindersah, Desta Yolanda, Febrian Hadiatna, "Internet of Things using Publish and Subscribe Method Cloud-based Application to NFT-based Hydroponic System", IEEE 6th International Conference on System Engineering and Technology(ICSET), Oktober 3-4, 2016 Bandung – Indonesia

[6] Ryohei Banno, Susumu Takeuchi, Michiharu Takemoto, Tetsuo Kawano, Takashi Kambayashi, Masato Matsuo, "A Distributed Topic-based Pub-Sub Method for Exhaust Data Streams Towards Scalable Event-driven Systems", IEEE 38th Annual International Computers, Software and Applications Conference, 2014.

[7] A. Carzaniga, D. S. Rosenblum, et al. Design and Evaluation of a WideArea Event Notification Service. ACM Transactions on Computer, 2004, pp.343-356

[8] G. Cugola, E. D. Nitto. The JEDI event-based infrastructure and its application to the development of the OPSS WFMS. IEEE Transactions on Software Engineering(TSE), 2001, pp.827-850

[9] S. Shukla, P. Saikia, M. Cheung, J. She and S. Park, "Effectiveness of Mobile Notification Delivery", Mobile Data Management (MDM)", 18th IEEE International Conference, pp. 21-29, 2017.

[10] Carzaniga, A., D.S. Rosenblum, and A.L. Wolf, Achieving scalability and expressiveness in an internet-scale event notification service. Proceeding of Nineteenth ACM Symposium on Principles of Distributed Computing (PODC 2000), 2000.

[11] Banavar, G., et al., An efficient multicast protocol for content-based Publish-Subscribe systems. Proceedings of the 19th International Conference on Distributed Computing Systems (ICDCS'99), 1999. 11. Pietzuch, P. and J. Bacon. Hermes: A Distributed Event-Based Middleware Architecture. in Workshop on Distributed Event-Based Systems (DEBS). 2002.

[12] G. Hohpe and B. Woolf, Enterprise integration patterns. Boston: Addison-Wesley, 2004.

[13] Altinel, M. and M.J. Franklin., Efficient Filtering of XML Documents for Selective Dissemination of Information. Proc. of VLDB 2000, 2000.

[14] Diao, Y. and M.J. Franklin, High-Performance XML Filtering: An Overview of YFilter. IEEE Data Engineering Bulletin, 2003(March, 2003).

[15] Barton, C.M., et al., Streaming XPath Processing with Forward and Backward Axes. Proc. of ICDE, 2003.

[16] Feng Peng, S.S.C., XPath Queries on Streaming Data In Proc. of SIGMOD, 2003.

[17] E.Onica,P.Felber,H.Mercier,andE.Riviére,''Confidentiality- preserving publish/subscribe: A survey,'' ACM Comput. Surv., vol. 49, no. 2, p. 27, 2016

[18] Baldoni, R. and Virgillito, A. 2005. Distributed event routing in publish/subscribe communication systems: a survey. Technical Report TR-1/06. The Computer Journal, vol.50 (2), pp.444 -459

[19] Klein, A., Mannweiler, C., Schneider, J., Schotten, H.D.: Access Schemes for Mobile Cloud Computing. In: Proceedings of the 2010 Eleventh International Conference on Mobile Data Management, pp. 387–392, 2010.

[20] P. Eugster, "Type-based publish/subscribe", ACM Transactions on Programming Languages and Systems, vol. 29, no. 1, p. 6-es, 2007.

[21] S. Lakhotia, "Why we Publish, Where we publish and What we Publish?", Proceedings of the Indian National Science Academy, vol. 80, no. 3, p. 511, 2014.

[22] Y. Zhao and J. Wu, "Building a reliable and high-performance content-based publish/subscribe system", Journal of Parallel and Distributed Computing, vol. 73, no. 4, pp. 371-382, 2013.

[23] S. Oh, J. Kim and G. Fox, "Real-time performance analysis for publish/subscribe systems", Future Generation Computer Systems, vol. 26, no. 3, pp. 318-323, 2010.

[24] L. H., R. S. and N. R., "Review for Event Delivering Techniques in Publish/Subscribe Scheme", International Journal of Computer Applications, vol. 132, no. 16, pp. 1-5, 2015.

[25] P. Narasimhan and P. Triantafillou, Middleware 2012. Heidelberg: Springer, 2012.