Content-Based Publish/Subscribe Communication Model between IoT Devices in Smart City Environment

Fulya Ozturk
Istanbul Kultur Univesity
Computer Engineering Department
Atakoy/Istanbul
fulya4306@gmail.com

Ayse Meliha Ozdemir
Istanbul Kultur Univesity
Computer Engineering Department
Atakoy/Istanbul
ayseozdemir95@hotmail.com

***Abstract*— In recent years the population of the cities has been increasing the getting over half of the whole world population. These peoples are facing with some security and infrastructural needs. Some of these needs can be met with the use of some smart technologies such as Internet of Things Devices (IoT) with different types of sensors. Coordination and communication of these devices are very critical for enabling the digital solutions in a secure and comfortable city life. However, due to the restricted ability of these devices, setting up a flexible communication platform is a very challenging issue. In this paper for setting up the physical security of smart homes/buildings, a content-based publish-subscribe model is proposed by the use of wireless sensor network nodes in the critical environment. Due to the energy constraint of the devices, the broadcasting feature of the wireless communication is not used, instead of this a peer to a communication according to the content of the message is preferred. Experimental results showed that the proposed communication model can be applicable for smart city environments.**

Keywords— Smart City; Smart Home; Publish/ Subscribe; Content-Based Notification

I. Introduction

Smart City is a relatively new however an active research area not only in the academics but also in the city management domain. In smart city domain, there are lots of sensors, and the Internet of Things (IoTs) devices, to collect data from the city and this information is used for increasing the quality of residents in the city. These devices can be used either in Smart Infrastructure, Buildings, Water Managements, Waste Management, Transportation, Healthcare, Education, Traffic Management, Energy, etc. (as depicted in Figure 1). Setting up an acceptable, secure and applicable communication model between all these IoT devices is a trivial issue. In some recent works, authors focused on the use of these IoT devices with different communication models [1, 2]. However, in a smart city/building/home concepts, this is inevitable and must be accomplished. Standard computerized model all computing agents are connected to the Internet and have IP addresses, and therefore they can set up a point to point communication models. However, in the IoT things concept; sometimes, there are not unique addresses and devices are not connected to the Internet directly. Therefore, some alternative communication models need to be set between these devices.

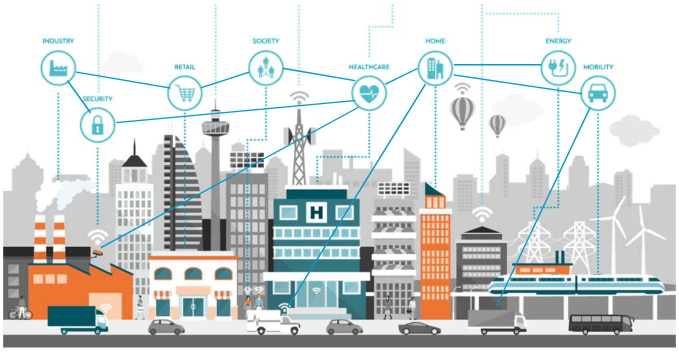


Fig. 1. Smart City and Its Components

Publish-Subscribe is one of the most important methods used for messaging. The name has been chosen to externalize the most prominent feature of this communication model. In this model there two types of devices as subscribers and publishers. Subscribers can subscribe to some types of messages which can be produced by some sensors or computing agents. However, the publisher can either generates some type of data or forward the incoming data according to the requests on them as depicted in Figure 2.

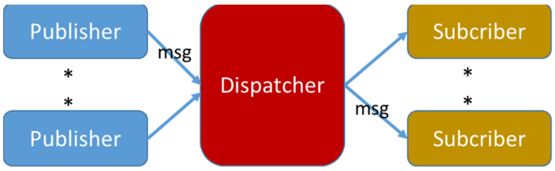


Fig. 2. Publish-Subscribe Communication Model

For example, in a smart city environment, a subscriber wants to get all meteorological data that are produced. Or other all types of traffic data that are produced and register to the publish subscriber service accordingly. However, this type of dispatching need to disseminate lots of IoT sensed data, for decreasing the number of data an additional content based mechanism can be used as mentioned in [3, 4] Content-based

978-1-7281-1315-9/19/$31.00 ©2019 IEEE

publishing / subscriber distributes systems specification a selective, measurement, and decentralized approach to data distribution. Specifically designed to improve resource usage efficiency, it provides the necessary compliance for Content-Based Publishing / Subscription. Content-based relation is a communication service in which the flow of messages sent from the senders to the recipients is directed by the content of the messages determined by the senders and without the explicit addresses attached to the messages. In the content-based service model, the message content is configured as an attribute pair.

Therefore, in this paper, it is aimed to propose a content-based messaging paradigm which is set on a publish-subscribe communication model in the smart city environment which has lots of sensor nodes (as IoT devices) that need to communicate each other. The experimental results showed that the proposed system could be applied not only in a smart city environment but also for the use of smart buildings, smart homes which a narrower area that has many IoT devices in it.

II. RELATED WORK

There is a lot of research in distributed pub-sub systems. Content-based publishing / subscriber based on peer-to-peer systems is currently being addressed. Many of these systems provide scalability and debugging and address central issues in most publishing / subscriber system. First, it provides Content-based publishing subscriber systems through Distributed Mixed Table (DHT) based P2P networks. To improve the efficiency of the overall stream, the activity maps events and the subscription or activity as a combination of the field schema defining and features in the subscription or activities. For DHT-based P2P networks for event deployment, the built-in multicast tree is used. This facilitates mapping and routing [5]. Mainly in the simplest cases, routing can be done with a rule-based mechanism [7]. In this manner, the related forwarding mechanisms are triggered according to the rules stored on router nodes. If a notification reached to the router node, then forwarding mechanisms are activated according to these rules.

In a structured P2P network, it forms a public-scale, centrally-based content-based pub subsystem. Because of a combination of schema identifiers and properties of subscriptions or events, subscriptions and events are mapped to the assignment nodes in the system. All nodes have no obligation to publish events. Only the number of asset properties in registered subscriptions is transferred to nodes that are slightly less than published activity. This reduces the number of change messages for transmission [6]. In this type of systems, fault tolerance mechanism is important and need to be solved according to some algorithmic solutions [9]. Subscriptions are managed according to the first feature in the ordered filters and so only the events that have been published, the activity matches must increase efficiency, and match the subscriptions that have the same first attribute in the appointment nodes. In this system, DHT uses the implicit multicast tree in the routing sub-layer to deliver events, so that the extra layer is applied to the substrate [12].

Another approach is content-based routing (CBR), which is becoming increasingly popular as a building block for distributed applications. CBR is used because messages are redirected according to content in a distributed architecture. Thus, many of the present systems perform the CBR based on a network of overlapping trees. Therefore, it is applied to a wide range of networks. In this approach, named as HyperCBR1, this topological feature is applied in many dimensions' areas with different sections and directing different messages and subscriptions. The Hyper CBR analysis model is used in peer-to-peer networks for content-based searches and content-based publishing subscribers. This is added to messages in the network that reveal the orientation of the nodes relative to their interests. They are defined according to the contents of the respective message classes. The results offer interesting opportunities to adjust the setup time of our protocol, even in very large scale environments, with effective CBR access and expected transmission disturbance processes. [8]

The content network was discussed with a different approach. The content network is a new technology that is managed by content routers that examine content descriptors such as URLs and cookies. In existing distributions of the content network, content routing is mostly about choosing the most suitable back-end server in virtualized web server clusters. Compares the performance of the proposed plan to a DNS-based content access scheme and Routers conduct content-based routing by examining the content of the traffic. In this proposed architecture, it creates a new content derivative tagging scheme based on the label specified in the input router of the routing process, where the content is reviewed only once. It is used to compare architecture to a DNS-based content access scheme, which is the most preferred content access scheme on the Internet. A different approach for routing was to utilize Network layer performance, content-based publishing / subscribing was performed using Software Defined Network (SDN) capabilities. SDN permits the transmission of content events to which the keys are loaded directly to the keys, while negatively affecting the effectiveness of this hardware, causing hardware restrictions and unnecessary network traffic. Realizing a balance between purely application-based and completely network layer-based publishing / subscriber applications. Moreover, it provides different selection algorithms with different degrees of complexity in order to determine the events to be folded in each layer, so that redundant network traffic can be minimized, and mixed software can be minimized. Hybrid middleware offers complete flexibility in accordance with system performance requirements. Implement and thoroughly evaluate the performance of a hybrid content-based pub/sub middleware. [10]. In Data dissemination scheme Fault Tolerance should be taken into consideration as depicted in [6].

And then following, the on-demand scaling feature then provides Content-based Publish Subscribe (CBPS) systems with a content / message deployment option for Content Subscribers to Content Subscribers through Content Brokers. A number of existing systems are known to be reliable for subscriptions to Content Brokers, the privacy of data published by Content Publishers, and subscribers' requests. However, with the increasing use of service-oriented architectures and cloud-like technologies, broker functionality cannot maintain the trust relationship with third-party providers as outsourcing. The problem of providing privacy in CBPS systems is a bit troublesome. Because the solution to this problem should allow some reference decisions about the content without content depending on the content of the content brokers. Previous attempts to address this issue failed to achieve significant success. Content Brokers are also used to make content routing decisions based on content to protect the privacy of subscriptions published by Content Publishers. Content Brokers distribute content to minimize the burden on Content Agents [11]. This type of system use can be better for mapping of the smart buildings especially for selection of best location in an unknown environment as mentioned in [13].

III. METHODOLOGY

This section explains the details of the proposed system. The IoT devices are located in different places in the Smart City environment. These devices can be identified as different computing agents such as wireless sensor nodes or some sink positions as cellular phones or tablets, which are directly connected to the Internet. There are three different roles in the system: publishers, subscribers, and router. Each node should be described at least one of them. In this project, data routing with publisher and subscriber nodes will be performed. The data will be transmitted by the node generated by the node as a hop-by-hop manner. The system is implemented in Visual Studio environment because it is easy to implement and use in a graphical interface. Visual Studio made it easier for us to develop projects with the tools it contains. It also provides us with a variety of possibilities for different applications and applications, such as web-based and network-based transmissions. Use case diagrams are schematic modeling of system behavior for the user of the system. Our diagrams are designed using the Visual Paradigm program. The main purpose of the design is to show the class and transmission operation of the project in a clearer way. The use case diagram is given in Figure 3.

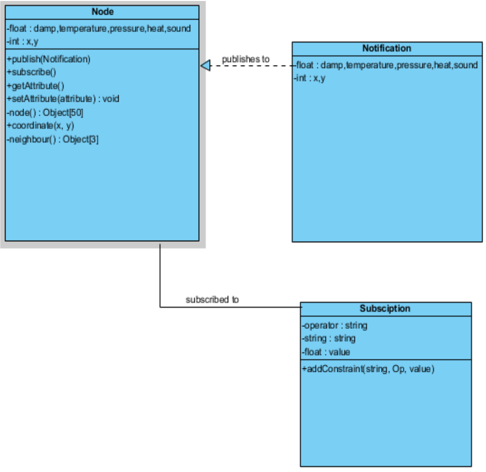


Fig. 3. Class Diagram

The activity scheme offers a different set of events or control streams with a flowchart or a data flow chart. Activity diagrams are generally used for business process models. They can also explain the steps in the usage status diagram. Modeled activities should be sequential and concurrent. In both cases, it should be a beginning and an end of an activity scheme. It is used to describe combined and time situations with activities, flows, decisions, and more. They are used during data transmission with the activity diagram. We use the message to show its usage in different areas, such as publish and subscribe. It shows how the data is transmitted to the Node. In our activity diagram, we see how the notification and subscription message is published and subscribe. The activity diagram for the implemented application is given in Figure 4.

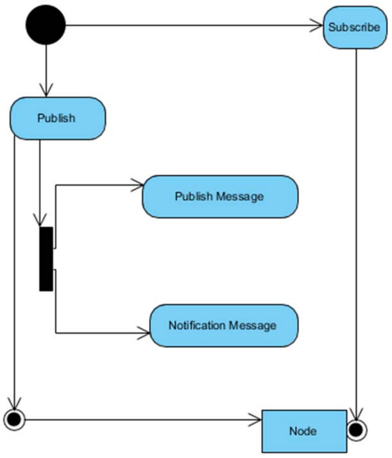
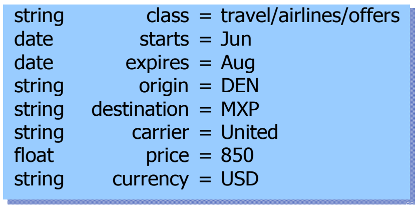


Fig. 4. Activity Diagram

A notification content can be defined as integer, float, double, string as depicted in Figure 5 [5]. And variables represent a value depending on their type. For example, the start date can be assigned as June, and price value can be assigned as 850, and the currency type is USD and so on. In the proposed publish / subscribe system, we used certain types of sensors in the node's communication. These values are values such as damp, heat, temperature, sound. We have taken these values randomly in our own project.



string	class	=	travel/airlines/offers
date	starts	=	Jun
date	expires	=	Aug
string	origin	=	DEN
string	destination	=	MXP
string	carrier	=	United
float	price	=	850
string	currency	=	USD

Fig. 5. Define Type Example (Notification)

All attributes are identical to their names. Attribute types have a limited set of types. For these types, a fixed set of operators is also defined. As the content types of the message described here; such as char, integer, float, string, data array, and byte. Each feature is filtered which like an Equation (=), sorting relationship (>), postfix operator (<) not equal (! =), large equal (> =), small equal (<=) All attributes are filtered. Publishing /subscribing systems enable many parties to communicate simultaneously with a large number of messages exchanged. The Publish Subscribe system provides an inter-layer communication that sends messages to one or more recipients using requests defined by a sender by a sender instead of an open destination address determined by the sender. In particular, a sender transmits messages, while recipients subscribe to messages of interest; The system is responsible for distributing the broadcast messages to the matching subscribers.

Internet applications that can use the broadcasting / subscriber system are in fact utilized as message messaging, personal information management, data sharing, and online news deployment. We offer a routing protocol specifically designed for content-based. In this sense, it is the first protocol used for a content-based communication service over a general point-to-point network. A growing event was shown with the pub / sub communication model as a means to disseminate events with distributed systems in wide area networks. This section provides a detailed introduction of pubs / subsystems. It examined all aspects of pub / subsystems, including goals, features. Routing and forwarding based on subscription and notification content. Subscription Forwarding includes a filter as we are referring to the value it wants for comparison and sends the content as value. In this way, all node-to-node transmission with its neighboring nodes continues through the same filter. and the actual node is reached by passing through all the appropriate nodes. For example, a = the desired value, s1 is filtered according to the figure. (s1:"sound<30") It's shown in Figure 6. In this figure, it is shown that the IoT device “a” send a subscription message its nearest IoT device as “1” which is connected to some other devices. Its subscription is forwarding to the neighbors according to the broadcasting mechanism.

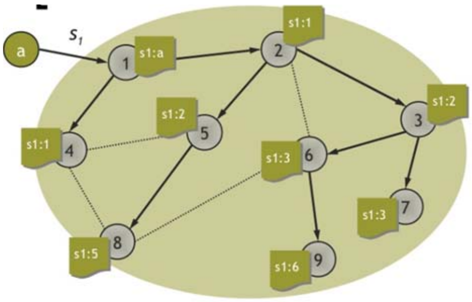


Fig. 6. Subscription Forwarding [5]

Routing and forwarding generalize the idea of subscription merging. Subscription merging which also groups subscriptions based on their similarity and creates a new subscription containing the set and it is used in the matching procedure. It is also used to filter the nodes that provide the subscription message with subscription merging. Values that do not fit within the figure are indicated by a cross. So notification will be transmitted using other ways according to the dispatching mechanism as depicted in Figure 7.

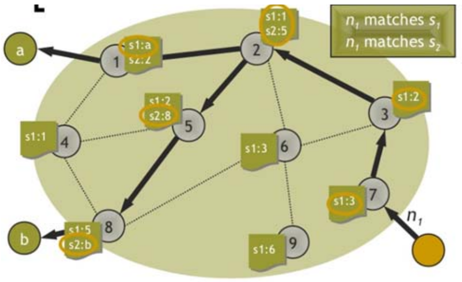


Fig. 7. Notification delivery[5]

With the notification delivery scheme, we see the ways in which the notification message passes by using the subscription such as n1 matches s1, value n1 matches s2 values. The message shows the way forwarded by filters. An event notification is a set of attributes in which each attribute is a type: attribute = (name; type; value). As most other publish/subscribe systems, we assume that an event is a set of attribute-value pairs. Each attribute has a name and a numeric or string value. A subscription is a conjunction of predicates over the attribute values, an operator (<,>, ≤, ≥, =, != , etc.) For example, a subscription expressed on the attributes x and y may be of the form.

The node generates a notification and sends the declaration it generates with publish. Notification is the value that we send from one node to the other node. Subscription compares our data using constriction and sends it to the other node. The content sent to the neighboring node is notification. An example of a subscription summarization. Node n1, n2 and n3 issued subscriptions {damp < 2}, or {heat < 5} or, {sound < 10}, respectively. Then, nodes n4, n5, and n6 only need to retain subscription {heat< 10} since it covers the other two nodes. Thus, the node performs the value assignment.

IV. EXPERIMENTAL RESULTS

The subscription contains a specific filter for each node. (heat for node1> 50, heat for node 2 = 60). The main program is node creation. Within the scope of this project, we have defined 50 nodes. Node1 sends a subscription message. Node 2 sends a notification message. It is publishing and subscribes using all the appropriate neighbors along 50 nodes. It finds its neighbors with the method of coordinate. The nearest neighbor can find the same coordinate as the nearest one. Such as node1 heat> 20, node2 heat> 20, node3 heat. As we define our classes as object-oriented, all classes work in aggregate.

The test is the process of determining the behavior of a program using static and dynamic methods, using test states to determine whether it meets a set and whether it meets the requirements. In the application interface, the user's steps and the basic principle of the application are described in detail. The test environment used in the project is the Visual Studio on a virtual server. Experiments have been performed on another software program for c#. Because the application is a web application that runs, it has also worked well in Windows, Linux, etc. We have tested our project via postman because our project is web-based. We were able to perform many different operations such as transmission of data with Postman and the occurrence of notification. Subscription message. The publication of the message provides many actions such as filtering neighbors. In this application, test cases performed for error detection are explained step by step in Table 1. As a result of development and testing, the project's performance on other operating systems has been tested. It worked well with the postman for running a web. Experiments have been done on errors that the user will encounter when using the application. The answers to these errors were examined.

The manual work takes even longer in the interfaces used separately in the project. However, the transmission time of the nodes developed with this application will be completed at the end of the transmission which has been completed for the longest period. As you can see, the transmission of the message across all nodes will take time. In addition, this project is the transmission of the data (notification) transmitted to the correct number in accordance with the main purpose of the subscription. Thus, this project has an important role in the postman use of a transmission test. The user has seen both the transmission and the correct way. A total of 30 nodes routing takes place.

TABLE I. ROUTING TIME

Node	Duration
sending subscription	5 seconds
questioning	10 seconds
publishing notification	10 seconds
Find neighbor	15 seconds
Sum	40 seconds

We performed this transmission using by using Notification and Subscription messages. When we tested the system, we are conducting the transmission of the message by locating the coordinates and the neighboring node. Location is important to calculate the transmission of the messages to the neighbors. In response to the transmitted subscription message, node1 generates a notification message. This generated notification message receives a subscription message again from the adjacent nodes and again creates a new notification based on the desired filter. This transmission continues between all nodes. We run our web-based project while we make a notification between nodes. We tested our project with the application 'Postman'. It offers us a very useful interface. Postman usually test APIs.

History queries can be accessed with the 'History' feature. Test and pre-request" scripts can be created. Because we run our project in web-based environment, we can debug queries with Chrome "inspect "tool. The executed tests can also be run and monitored automatically. As a result, nodes generate notification message can be received a subscription message again from the adjacent nodes and again creates a new notification based on the desired filters. This transmission between nodes continues for all nodes the data routing between the nodes and the notification. Notification is transmitted to the neighboring node according to the value of the neighboring node. Except for these tools, the output of the tools will be used for data routing. As a result of this we can set an efficient dissemination system for network based smart city application areas.

V. CONCLUSIONS AND FUTURE WORK

In recent years, the smart city concept gains great attention due to the extended usage of the Internet of Thing devices in Smart City environments. Especially, for the security of smart buildings of smart homes, the use of these devices are inevitable and enable high-level physical security. As a result of the wireless communication capability of these devices, standard IP based networking utilities are not possible to use in these environments. Therefore, new communication mechanisms and protocols are needed. In this paper, a content-based publish / subscription communication mechanism is proposed in the physical security of smart buildings/homes in a smart city environment. The system based on dynamic peer-to-peer communication model between nodes distributes the load evenly while maintaining the independence of each node. Firstly, the data generators as smart sensors in the environments dispatch their abilities, and according to these features related subscribers, register/subscribe on these nodes. After these sensors generate some important notifications, these data are disseminated according to the content of data with the use of publish/subscribe communication mechanism. We performed some experiments on the transmission using notification and subscription messages. Experimental results showed that the proposed system is applicable in smart city environments.

As a future work, it is aimed to connect lots of devices to the system. By this way, the scalability and extensibility of the system can be increased to a city size to enable in many of the smart city environments as a whole. Additionally, the use of multimedia sensors can also be adapted to the system according to subscribers' requests.

REFERENCES

[1] F. T. El-Hassan and D. Ionescu, "Design and Implementation of a Hardware Versatile Publish-Subscribe Architecture for the Internet of Things," in IEEE Access, vol. 6, pp. 31872-31890, 2018. doi: 10.1109/ACCESS.2018.2842706

[2] I. Made Wirawan, I. Dwi Wahyono, G. Idfi and G. Radityo Kusumo, "IoT Communication System Using Publish-Subscribe," 2018 International Seminar on Application for Technology of Information and Communication, Semarang, 2018, pp. 61-65

[3] Y. Tekin and O. K. Sahingoz, "A Publish/Subscribe messaging system for wireless sensor networks," 2016 Sixth International Conference on Digital Information and Communication Technology and its Applications (DICTAP), Konya, 2016, pp. 171-176.

[4] H. Cam, O.K. Sahingoz, and A.C. Sonmez, "Wireless sensor networks based on publish/subscribe messaging paradigms", International Conference on Grid and Pervasive Computing, 2011, pp. 233-242

[5] A. Carzaniga, D. S. Rosenblum, and A. L. Wolf. Design and Evaluation of a Wide-Area Event Notification Service. ACM Transactions on Computer Systems, 19(3), pages 332-383, 2001.

[6] E. Ertugrul, O.K. Sahingoz, "Fault Tolerance in Real-Time Systems: A Review", Intelligent Systems Design and Applications, Springer, pp: 283--293, 2018.

[7] O.K. Sahingoz and N. Erdogan, "RUBDES: A rule based distributed event system", International Symposium on Computer and Information Sciences, 2003, pp. 284-291

[8] P. Eugster, P. Felber, R. Guerraoui, and A.-M. Kermarrec, "The many faces of publish/subscribe," Computing Surveys, vol. 2, no. 35, 2003. [2] A. Bulut, A. K. Singh, and R. Vitenberg, "Distributed data streams indexing using content-based routing paradigm," in Proc. of 19th IEEE Int. Parallel and Distributed Processing Symposium (IPDPS), 2005.

[9] O. K. Sahingoz and A. C. Sonmez, "Agent-based fault tolerant distributed event system", Computing and Informatics, 26 (5), 2012, pp. 489-506

[10] P. R. Pietzuch and J. Bacon. Hermes: A distributed event-based middleware architecture. In ICDCSW '02: Procs. of the 22nd Int. Conf. on Distributed Computing Systems, 2002.

[11] A. Ouksel, O. Jurca, I. Podnar, and K. Aberer. Efficient probabilistic subsumption checking for content-based publish/subscribe systems. In Proceedings of 7th Middleware, 2006.

[12] Publish/Subscribe Internet Routing Paradigm. Conceptual architecture of pirp including subcomponent descriptions. Deliverable d2.2, PSIRP project , August 2008.

[13] E. Ertugrul, U. Kocaman and O. K. Sahingoz, "Autonomous aerial navigation and mapping for security of smart buildings," 2018 6th International Istanbul Smart Grids and Cities Congress and Fair (ICSG), Istanbul, 2018, pp. 168-172.