On Proximity Application Server

Dmitry Namiot
Lomonosov Moscow State University
Moscow, Russia
dnamiot@gmail.com

Manfred Sneps-Sneppe
Ventspils University of Applied Sciences
Ventspils, Latvia
manfreds.sneps@gmail.com

***Abstract*—In this article, we consider application programming issues for proximity-based services. The work uses the abbreviation ProSe (Proximity Services), introduced in the 3GPP specifications. According to the 3GPP specifications, proximity-based services are designed to find devices suitable for organizing direct interaction between devices (D2D - Device to Device). In our case, we consider another approach in which the determination of closely spaced devices is the ultimate goal of the process. The direct connection of devices is not considered at all, and the main reason for this is security-related problems. The paper considers an approach in which the determination of proximity to a device or devices is the basis for the presentation of data or services without the direct connections between devices.**

I. INTRODUCTION

The term ‘proximity services’ appeared in the 3GPP specification version 12 in 2015 [1]. In this implementation, an LTE device connection with another device was added (a so-called device to device communication or D2D). In this regard, the term “proximity service” (ProSe) appeared. According to the specification, ProSe is part of D2D technology. This is an important point for our further consideration, which captures the fact that in general, it is part of the process to ensure direct interaction between devices. This is operator technology (the process is controlled by the operator), which is based on a number of changes in LTE standards, among which the main thing for D2D is the appearance of the so-called ‘sidelink’ - a radio interface for direct interaction between devices (Fig. 1). ProSe is designed to help one device find nearby devices. It is for such devices the possibility of direct connection (sidelink) has been introduced.

The purpose of this direct interaction can be to reduce the load on the network, increase the bandwidth of the available networks. Another option that was initially considered was the provision of communication in areas where there is no network coverage. In a more general form, the aforementioned goals can be represented (described) as a delegation of their functions by a telecommunication provider [2, 3].

Classically, D2D is defined as the interaction between two devices without the participation of a core network (base stations) in routing [4]. D2D interaction is a P2P network. In accordance with this, it is assumed that this technology will allow interacting devices to share content, view streaming video, etc. [5]. The main advantage for potential D2D services in LTE and 5G is usually claimed to be a high data rate with low latency.

At the application level, the literature usually discusses tasks such as games, multimedia services, social networks, advertising [6,7,8]. All studies that are in the literature focus on delivering content using D2D connections. A typical model (architecture) of the system is shown in Fig. 2.

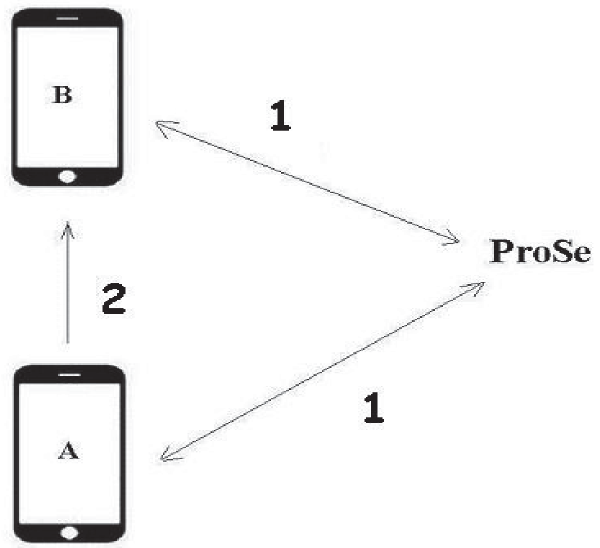

Fig. 1. D2D model

In this picture:

1 - control channels

2 - sidelink (the direct channel between devices A and B)

ProSe - operator-based proximity detection service

Technically, the architecture of the system for determining closely located devices is illustrated in Fig. 3, which shows the main components and the interfaces between them. This figure shows:

- Mobile devices (UE-A, UE-B)
- Radio access network (Evolved Universal Terrestrial Radio Access Network, E-UTRAN) and advanced packet core (Evolved Packet Core, EPC)
- Radio interfaces PC1 - PC5 (PC2 is not necessarily radio interface as there might be cases with Fiber-to-the-site solutions)
- ProseApp application server

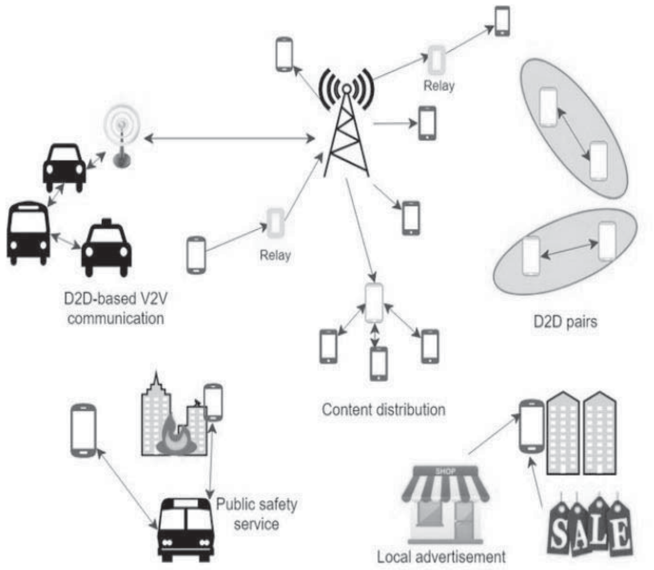

Fig. 2. On D2D network [9]

Technically, the specification defines the following operation algorithm. A mobile device (for example, UE-A in Fig. 3) notifies surrounding devices of its presence by sending some code (*ProSe Application Code*). This code is allocated to the device as part of the call to some function (in fact - the described service) - *ProSe Application*.

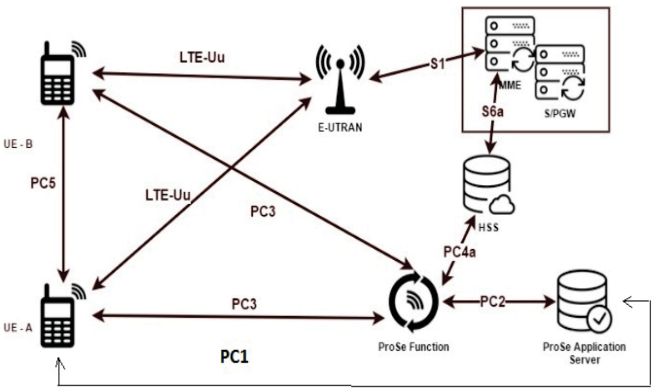

Fig. 3. On ProSe Architecture [10]

All such functions have their own identifier - *ProSe Application ID*. Accordingly, the *ProSe Application Code* is the code associated with the application identifier and used in the *ProSe Discovery* process (service search/submission process).

The received code is sent by the mobile device (the specification speaks of the area up to 500 m). In this case, the distributed code can be available to everyone (everyone can determine the availability of the service) or only to some limited group of devices. From here comes the *ProSe Restricted Code* - a code that makes information about the service (data) available only to some limited group of users. This is due to the search for services nearby with restrictions imposed - only a certain selected group of users should have access to the data.

Discovery Filter: it is a combination of Application Code / Restricted code and a mask (can be considered a regular expression) to track the receipt of data for a given condition

ProSe Function: it is an element that is responsible for providing devices (UE) access to ProSe services, as well as for supporting work with the *ProSe Application ID* and *ProSe Application Code*. The specification speaks of a logical function. Technically, this is a web server (ProSe function will process HTTP requests from devices).

ProSe application server - an entity for storing information about users, function identifiers, metadata, etc.

Services based on the proximity principle include, as per specification:

- *ProSe Direct Discovery*: the process by which a device (UE) detects and identifies other device(s) nearby
- *ProSe Direct Communication*: leveraging LTE resources from a cellular network for messaging
- *Prose Discovery* at the network level (EPC level ProSe Discovery) and wireless network level (WLAN)

Obviously, for the tasks considered in this paper, it is the (and only) ProSe Direct Discovery element that is interesting. The network proximity, in this case, will be determined relative to some allocated device and be provided by receiving some ProSe Application Code from this device.

The specification defines two modes for *Prose Direct Discovery* to work: open and limited. The mode determines who can detect the proximity of a particular device. As part of Direct Discovery, the device sends out some identification information that other devices nearby can receive. The fact of receiving such identification information is confirmation that the sender and the recipient are nearby.

The mode, as indicated above, is defined for the device that sends information (that is, the device that is used as a reference to determine proximity). In the case of an open mode, its distribution can be received by any other device and, accordingly, any device can participate in the proximity determination procedure. On the contrary, in limited mode, the distribution source will determine which other devices can receive its distribution (will be able to participate in the proximity determination procedure). Public services must comply with open mode. Technically, based on the model proposed below, the type of mode will not play any role and the proposed model will be applicable in both modes.

It should be noted that the operating model and any description of software interfaces are currently missing. The scenarios described in the literature are limited to the assertion that a user can find other users nearby to share data with them. The general opinion (statement) is that D2D will become the basis of many services based on the "local" (at the current location) provision of services. But these services, if briefly described, are reduced, in fact, to the transfer of data between devices under the control of a telecom operator. A typical example is the review [11], where, in fact, several variations of device connections between themselves are described. Recognizing the local (here is means operating in a spatially limited area) nature of services, we would like to dwell on this work on what they can provide and what the software architecture for their implementation can be.

The rest of the article is structured as follows. Section II addresses network proximity issues. Section III deals with our proposed model for Proximity Application Server (PAS). Section IV discusses the use of the PAS model for Wi-Fi and Bluetooth wireless networks.

II. On proximity services

In our view, public services will not and cannot, in modern conditions, be based on a direct connection of devices. There is one reason - security. D2D services are described as a scalable solution for a direct connection between devices. Scalability here is provided by the telecom operator, which provides centralized management of the occupation and release of frequency resources. But this control does not cancel the fact that the device will establish a connection with another unknown device. A device for direct connection from the point of view of safety should be treated precisely as an 'unknown' device. Accordingly, a direct connection is an action of the same order as establishing a Bluetooth connection with an unknown device to download a media file, installing Android applications from an arbitrary source, etc. These are actions that most mobile users are avoiding right now, and there is no reason to believe that increasing the speed of data exchange will change the attitude towards such operations. All direct connections carry obvious security risks.

We can note, for example, that the various D2D security solutions described in [32] deal exclusively with network aspects. They cannot in any way guarantee the security of content downloaded from a third-party device, even if the device (client) is correctly authenticated and validated by the network. This cannot be solved by the network technology.

It seems to us that direct connections (D2D) will have limited use. Firstly, here we can name some sort of dedicated environment (corporate subsystem). We can assume that all subscribers in such a network enjoy mutual trust. Another argument in the favor of using direct connections in such networks is the lack of personal (private) data and the presence of some kind of corporate service (IT support), which will solve possible problems with devices. Another option is a service that implicitly uses direct exchanges in its algorithms. In this case, we are talking about trust in the service itself, the mobile subscriber may not be aware of such exchanges. An example is the services that use P2P exchange in their algorithms. A classic example is the Skype P2P protocol [12]. The use by the operator of client devices to perform their tasks (offloading) remains transparent to the subscriber and also falls into this category [13].

It seems to us that proximity-based services should be considered as context-aware ubiquitous services. Proximity information is a part of the context definition. Accordingly, the main thing that the determination (detection) of proximity should do is determine the conditions for the presentation or processing of data.

The public services themselves should work, in our opinion, without establishing a connection (or, at least, without a mandatory requirement to establish such a connection). In fact, it's safe for users today to establish an external connection using web protocols with trusted hosts. From the point of view of cyber-security and the analysis of possible related problems, this is viewing (without any problems) without the obligation of additional data loading and an explicit decision by the user to download data (this is an analogue of a direct connection) to your device.

With this approach, from the D2D service, in fact, there is only one component left - Direct Discovery (DD). Returning to the model shown in Fig. 3 and described above, the receipt of the *ProSe Application Code* by the device is the end of the process of determining proximity (searching and revealing nearby nodes). A context-aware application (and, in fact, all mobile applications should be context-aware), having received such a code, can use this information (the fact of proximity to the sender) when processing data. Each such resulting code is now part of the context [14], [15].

As types of possible actions (operations) with content, we can indicate the following:

- A device (actually – mobile user) entering into the accessibility (distribution) zone of a specific code (codes) or exiting from such a zone and this event causes a change in status (state) in the application.
- A device entering into the accessibility (distribution) zone of a specific code (codes) or leaving this zone and such an event causes a request for information (some kind of access to the data warehouse) for subsequent processing.
- A device stay in the distribution zone of the code (codes) causes a change in status or a request for information upon the occurrence of any other conditions (for example, if the time spent is exceeded).
- Recording (logging) events (entry / exit from the distribution area and stay in the distribution area) for use in subsequent processing.

As per application examples:

- Notification of the intersection (at the entrance or exit) of a certain virtual perimeter (it is an analogue of a geo-fence)
- Sending notification with a coupon / special offer in case of repeated presence in a certain area
- Turn off the call on your mobile phone when you fall into a certain area
- Notification when changes in the set of received (available) codes are detected, etc.

Limited area of distribution (broadcasting) *ProSe Application Code* will always determine a spatially limited area for possible reactions (to represent data, for example). Up to the size of this area, the *ProSe Application Code* can replace geo-positioning. The determination of geo-coordinates will be replaced by the determination of proximity to some sender. If the distributed code (function) is tied to some location, then receiving the code is a confirmation of proximity to the specified place.

Note that the concept of proximity itself is a natural application for location-aware services. Ontological analysis of location-related services is carried out in many works. As examples, we can cite the works [16, 17]. It follows from this that only topological relationships and directions in geo-information systems cannot be described (represented, simulated) by a proximity predicate. A typical example of topological relationships is the intersection of regions. Directions here are geographical directions such as north-west, etc. Obviously, this cannot be described as “being close to ...”. The set measurement accuracy of 500 m, on the one hand, seems quite large. For example, as shown below, using Bluetooth (Bluetooth Low Energy) it can be 1 meter or less. On the other hand, everything is determined by the tasks being solved. For example, today for digital urban applications, the data of mobile operators on the movements of subscribers in Moscow region are available just in squares with a side of 500 m [18].

The use of the *ProSe Application Code* in context-aware applications can be explained as follows. Context (as understood by mobile computing) is arbitrary measurable characteristics that can be added to a location. The resulting *ProSe Application Code* is the (semantically) data of a certain sensor (proximity sensor) that must be supplied to the application for use in its algorithms. Let us see a simple example of a web application. A snippet of JavaScript code that gets the user's current geo-coordinates:

```
<script>
  if (navigator.geolocation) {

navigator.geolocation.getCurrentPosition
(showPosition);
  } else {
    x.innerHTML = "Geolocation is not supported by this browser.";
  }
function showPosition(position) {
//Latitude: position.coords.latitude
//Longitude:position.coords.longitude
// отображение контента в зависимости от координат
…
}
</script>
```

After receiving the coordinates, the callback function *showPosition (position)* is called to which the received coordinates are transmitted. Here you can organize the display of data on the web page for the given coordinates. For example, controlling the visibility of blocks of text using CSS, dynamically creating such blocks, etc. This is how it works on thousands of web pages, customizing their content according to the user's location.

This is a typical use of context (location is always part of the context). This use is based on the fact that the application (in this case, the web application) has access to the context (can read this data). In this case, this access is provided by browser support for the *navigator.geolocation* object. ProSe support will have to mean the presence of a semantically equivalent object (for example, *navigator.prose*), which will also allow us to get the value of the *ProSe Application Code* in the given callback function, and, depending on this, dynamically modify the web-page.

```
<script>
  if (navigator.prose) {
    navigator.prose.getProSeCode
   (showCode);
  } else {
    x.innerHTML = "ProSe is not supported by this browser.";
  }
function showCode(ProSeApplicationCode) {
// отображение контента в зависимости от
// ProSeApplicationCode
…
}
</script>
```

For example, let's assume we have a large mall with a website and there is some code (*ProSe Application Code*) available inside the mall. Then, using the above-described method, we can add a fragment to the web-page that recognizes whether the site visitor is currently physically inside our mall (in this case, the corresponding code will be available on the web-page). Accordingly, the web page can be modified taking into account that the visitor is currently inside (show special offers, promotions with a limited life-time, etc.).

In this case, we do not dwell on the possible technical details of such an implementation for web browsers. This can be a new standard similar to the W3C Geo standard, a custom version for mobile web-browser, a plug-in for Chrome, a proxy application for web pages that adds the necessary objects, etc. Implementation methods may be the subject of a special paper.

III. On Proximity Application Server

As follows from the above, in the proposed approach from the D2D specification, we focus solely on one of its components - Direct Discovery (DD). Accordingly, the mobile device that sends the code is semantically represented (interpreted) as a tag. The *ProSe Application Code* broadcast is considered, for example, the same way as sending by some iBeacon (it is Bluetooth Low Energy (BLE) tag from Apple) a pair of its identifying values (minor, major) [19]. The differences are in the organization of this process and the presence (absence in the case of BLE) of telecom-based management.

Receiving such a distributed code by a third-party device is a fixation of the fact of proximity with respect to the sender and, at the same time, clarification of information about the context (the recipient device is in the receiving zone of a specific tag or not). Different tags (receiving different codes) simply correspond to different characteristics of the content. In other words, we follow the previously proposed concept of network proximity.

Usually, this term was used with the orientation to the word “network”. This is a metric that estimates the number of intermediate nodes. In other words, how many network nodes need to be overcome by a data packet or a delay in data transmission [20]. In our work, we focused on spatial proximity, which was estimated using network technologies. This assessment was based on a simple restriction of signal propagation in the case of wireless networks [21, 22].

By treating the sender of the *ProSe Application Code* as a “tag”, we can follow the general architectural model for working with tags. The data sent by the tag serves, in general, as the key to search for information associated with this newsletter (or with the tag if the data does not change). This is shown in Fig. 4. In the case of context-oriented systems, the information associated with the broadcasted ID is what describes the additions to the context.

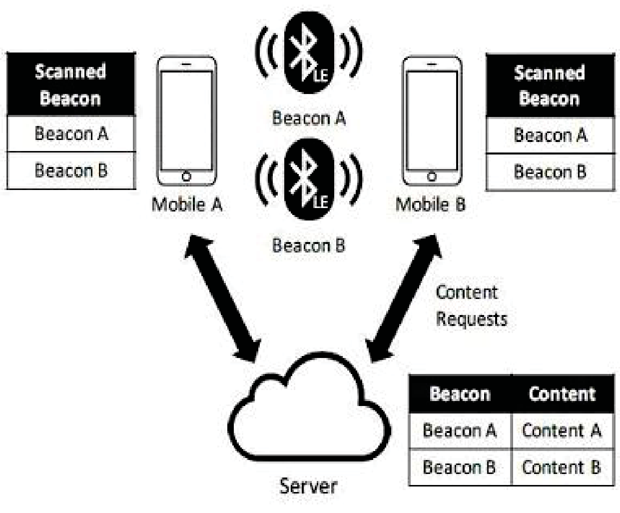

Fig. 4. Content request by tag’s ID [23]

For ProSe, in our understanding, it should be the same, just with the replacement of tags on the broadcast device.

The next point to note is the location of the available content. In Fig. 4, in order to get content, we need to access an external source (server, cloud). It's very easy to explain - the same BLE tags simply do not have the ability to store any content. The issue with connections also disappears for this reason. In general, if we look at services that take location into account, then it is obvious that, by defining closely spaced objects, a service, in most cases, cannot “connect” to them at all. Search, in most cases, is really focused on “local” objects (in relation to either the current location or some imaginary/future location). But this search almost never involves a connection, because objects that are searched simply do not have such functions. The 3GPP specification regarding ProSe speaks only about mobile devices (UE in Fig. 3), but this is precisely because the original goal was to provide a direct connection. Nothing prevents right now to generalize this approach and consider one of the end devices in Fig. 3 as a tag, which, for example, will represent some kind of physical object. In fact, assuming a connection, we, on practice, stop at just one possible type of service, which has natural limitations. In fact, it is the context model that describes the whole set of services.

In the underlying ProSe architecture, Fig. 3 mentions the ProSe Application Server. But all that is being said so far is that it is a structure for storing information about possible functions, i.e. services for which the *ProSe Application Code* is issued. Below we present our vision of how this might look.

The presence of unique identifiers for all types of content leads us to the fact that the basis of the system is a database with a key-value model. The choice of solutions (implementation) is really huge here. In our experiments, we used an open source system Redis [24].

Accordingly, a record in our database is a service code and related content. As such content, we suggest using descriptions on Hypercat [25]. Hypercat is an open specification that is intended to describe resources in IoT projects. Any description on HyperCat is some kind of JSON fragment. The main element of the description is a URI with an accompanying set of RDF-like triplets that define the properties of the object described by this URI. An example of the model is shown in Fig. 5.

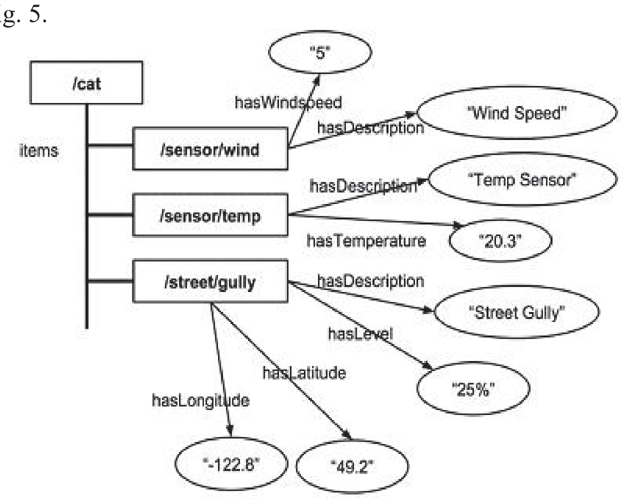

Fig. 5. The resource catalog (a basic unit of a description) in Hypercat [26].

The system is used by British Telecom and was proposed by BSI as a standard for searching in IoT systems [27]. The model used in Hypercat allows you to describe both sets of values and just some kind of web links.

Thus, the base stores a set of pairs:

`<code, directory description in JSON>`

The fact that all directories (this is a unit of description in Hypercat) are stored in one database will allow the use of Hypercat mechanisms (technically, these are HTTP requests) to search for local data. If the system stores links to some web resources that are dedicated to local problems (for example, links to the sites of business-objects located in the distribution zone), then the system, in fact, describes some model of the local Internet. It is a set of Internet resources containing some information that is useful exactly in the local context. For example, web-sites of companies located in some business center, groups or Facebook pages, neighborhood chats in Telegrams, etc. In this case, the notification of the existence of such resources (notification about the content of the local Internet) is organized in the form of sending (broadcasting) some code (*ProSe Application Code*). And only those who are physically located in a certain geographic location can get such a code.

The 3GPP specification suggests that the lifetime of the *ProSe Application Code* is limited. This is done, obviously, so that the code is re-requested, and subscribers who once received such a code, but have already left the distribution area could not use the "eternal" code. This problem is solved by introducing database records of the type

`<temporary code, code>`

for which we can set the lifetime. In Redis, for example, this is the command:

EXPIRE temporary_key N

where *N* is the time in seconds.

It is interesting that supporting such a repository of "local" information may be the main function of some specialized service-provider. As an analogy, we can mention service providers who take over the delivery of PUSH notifications [28]. According to a similar model, aggregators of "local" information can also work. Modern key-value databases are quite scalable, if necessary, for example, the use of Cassandra can be proposed as such global storage [29].

Such storage will allow the operator to also perform the functions of a service provider. It is also possible to offer additional services to application developers. The types of possible applications specified in Section II contain operations that depend on the history of the receipt of the codes. This includes, for example, all tasks that are related to assessing the movement of subscribers, data logging, etc. Such applications need to save this information (log of received codes or, actually – time series) somewhere. The Cassandra system mentioned above is suitable for storing such structured time series. Data will be divided by application (application codes issued to developers), and within applications - by codes (addresses) of mobile devices.

IV. On Application Server model for wireless networks

In this section, we would like to discuss how to simulate the proposed approach right now, without the available APIs for ProSe. Here we will take the advantages of the approach we have been developing for a long time related to the use of arbitrary nodes of wireless networks as tags and the construction of information systems based on the network proximity model on such a base. Wireless node identification is used for transmitting values (codes). For example, it could be the name of a Bluetooth node or Wi-Fi access point. It should be noted that both values can be set programmatically. Also, the values (codes) may be transmitted in the customized presentation (advertisement) of the wireless node. In all these cases, the wireless node broadcasts information, the reception of which does not require the organization of connections (that is, it is safe). As per practical examples, a set of various models and their implementations has been described in our paper [30]. We note here that the Bluetooth Mesh specification released in 2017 works on a similar data transfer model without establishing a connection [31] (so-called connection-less mode).

This approach works without any operator's control. Moreover, this will work even completely without a telecommunications operator. Using Bluetooth Low Energy allows you to make the distribution area small enough (1 meter or less), data broadcasting can be created dynamically via program-based creating nodes of wireless networks. The price for this flexibility is the lack of scalability that the 3GPP specification promises.

The approach described above (in Section III) with the proximity application server can be fully reproduced in such a model. The identification of the wireless network host will play the role of the *ProSe Application Code*. All types of data operations specified in Section II are fully implementable in such a model.

Due to the fact that the distribution areas of such code can be small (for example, when using Bluetooth, this distribution will be in the line of sight), there is a simple interpretation for the content in such an application server. The system can be described as the network equivalent of a QR-code. The user takes a QR-code using the phone's camera and receives the content to which this code refers. In the network version, the phone scans (automatically or after an explicit indication) the available network codes and receives the content to which these codes refer. All forms of content that are in the QR-code standard can be described using Hypercat. You can add also any other content for recognition. For example, a Twitter feed or a Telegram channel that is related to local issues. All of this is easily described using URIs in Hypercat.

It should be noted one more point where such a server for wireless networks will differ from the same for 3GPP ProSe. For wireless networks, we can easily use a custom-generated broadcast identifier (custom generated *ProSe Application Code*). In particular, the sent identifier may contain some information from the sender (the organizer of the broadcast). This means that in addition to fixing the fact of proximity, receiving such an identifier (code) means also receiving some piece of data from the sender. Which, in turn, in some applications, may make it completely unnecessary to access the server (cloud service) for getting content. All the necessary information will be transmitted in the code itself. Using, for example, advertising for wireless nodes, we can transfer significant pieces of data. Accordingly, for wireless networks, one more type is added to possible types of services - relaying of received data. The code (data) received by one node can be relayed by it to its own neighbors. This allows, for example, to organize a simple mesh network. In 3GPP, at least for today, the structure of the *ProSe Application Code* is fixed.

V. CONCLUSION

This article deals with the development of public services based on 3GPP D2D. We propose a model that uses only part of this specification - DD (Direct discovery). The other component (direct device interaction) is left for consideration because of security problems. In this paper, we consider an architectural model, in which the definition of proximity to some device(s) is the basis (trigger) for the presentation of data (services). In such consideration, device proximity definition can replace (as well as extend or supplement) work with geo-coordinates and also be an element of context definition in context-aware systems. In this work, we propose an operating model, which includes a database with a key-value model and JSON fragments from the open source system Hypercat as the main tool for content description. The types of services that can be implemented in such a model are considered. Also, the paper describes the modeling of such a system based on Bluetooth and Wi-Fi networks.

REFERENCES

[1] 3GPP TS 24.334 V12.1.1, December 2014; Technical Specification Group Core Network and Terminals, Proximity-services (ProSe) User Equipment (UE) to ProSe function protocols; Stage 3; Release 12.

[2] Aijaz, Adnan, Hamid Aghvami, and Mojdeh Amani. "A survey on mobile data offloading: technical and business perspectives." IEEE Wireless Communications 20.2 (2013): 104-112.

[3] Malandrino, Francesco, Claudio Casetti, and Carla-Fabiana Chiasserini. "LTE offloading: When 3GPP policies are just enough." In 11th Annual Conference on Wireless On-demand Network Systems and Services (WONS). IEEE, 2014.

[4] Militano, Leonardo, et al. "Device-to-device communications for 5G internet of things." EAI Endorsed Trans. Internet Things 1.1 (2015): 1-15.

[5] Sneps-Sneppe, Manfred, and Dmitry Namiot. "On 5G Projects for Urban Railways." In 22nd Conference of Open Innovations Association (FRUCT). IEEE, 2018.

[6] Vo, Nguyen-Son, et al. "Optimal video streaming in dense 5G networks with D2D communications." IEEE Access 6 (2017): 209-223.

[7] Pu, Lingjun, et al. "sNDN: a social-aware named data framework for cooperative content retrieval via D2D communications." In Proceedings of the 10th International Workshop on Mobility in the Evolving Internet Architecture. ACM, 2015.

[8] Mumtaz, Shahid, Kazi Mohammed Saidul Huq, and Jonathan Rodriguez. "Direct mobile-to-mobile communication: Paradigm for 5G." IEEE Wireless Communications 21.5 (2014): 14-23.

[9] Goian, Huda S., et al. "Popularity-based video caching techniques for cache-enabled networks: a survey." IEEE Access 7 (2019): 27699-27719.

[10] LTE Direct http://www.3glteinfo.com/lte-direct/ Retrieved: Jan, 2020

[11] Farzamiyan, Amir Hossein. "A Survey on Device-to-Device Communication in 5G Wireless Networks." In 14th International Conference on Software Technologies. 2019.

[12] Wang, Hao. "Skype VoIP service-architecture and comparison." INFOTECH Seminar Advanced Communication Services (ASC). 2005.

[13] Andreev, Sergey, et al. "Cellular traffic offloading onto network-assisted device-to-device connections." IEEE Communications Magazine 52.4 (2014): 20-31.

[14] Namiot, Dmitry, and Manfred Sneps-Sneppe. "Context-aware data discovery." In 16th International Conference on Intelligence in Next Generation Networks. IEEE, 2012.

[15] Namiot, Dmitry. "Context-Aware Browsing--A Practical Approach." In Sixth International Conference on Next Generation Mobile Applications, Services and Technologies. IEEE, 2012.

[16] Couclelis, Helen. "Ontologies of geographic information." International Journal of Geographical Information Science 24.12 (2010): 1785-1809.

[17] Kolas, Dave, Mike Dean, and John Hebeler. "Geospatial semantic web: Architecture of ontologies." In IEEE Aerospace Conference. IEEE, 2006.

[18] Namiot, Dmitry, Oleg Pokusaev, and Alexander Chekmarev. "Use of telecommunications operators' data in transport planning." International Journal of Open Information Technologies 7.12 (2019): 51-57.

[19] Namiot, Dmitry, and Manfred Sneps-Sneppe. "The physical web in smart cities." In Advances in Wireless and Optical Communications (RTUWO). IEEE, 2015.

[20] Sharma, Puneet, et al. "Estimating network proximity and latency." In ACM SIGCOMM Computer Communication Review 36.3 (2006): 39-50.

[21] Namiot, Dmitry, and Manfred Sneps-Sneppe. "On Proximity-Based Information Delivery." In International Conference on Distributed Computer and Communication Networks. Springer, Cham, 2018.

[22] Namiot, Dmitry, and Manfred Sneps-Sneppe. "Geofence and network proximity." In Internet of Things, Smart Spaces, and Next Generation Networking. Springer, Berlin, Heidelberg, 2013. 117-127.

[23] Jeon, Kang Eun, et al. "Ble beacons for internet of things applications: Survey, challenges, and opportunities." IEEE Internet of Things Journal 5.2 (2018): 811-828.

[24] Carlson, Josiah L. Redis in action. Manning Publications Co., 2013.

[25] Duke, Alistair, and Sandra Stincic Clarke. "Hypercat RDF: Semantic Enrichment for IoT." In Semantic Technology: 6th Joint International Conference, JIST 2016, Singapore, Singapore, November 2-4, 2016, Revised Selected Papers. Vol. 10055. Springer, 2016.

[26] Blackstock, Michael, and Rodger Lea. "IoT interoperability: A hub-based approach." In 2014 international conference on the internet of things (IOT). IEEE, 2014.

[27] Namiot, Dmitry, and Manfred Sneps-Sneppe. "On Search Services for Internet of Things." In International Conference on Distributed Computer and Communication Networks. Springer, Cham, 2017.

[28] Notification provider https://onesignal.com/ Retrieved: Jan, 2020

[29] Chebotko, Artem, Andrey Kashlev, and Shiyong Lu. "A big data modeling methodology for Apache Cassandra." In IEEE International Congress on Big Data. IEEE, 2015.

[30] Namiot, Dmitry, and Manfred Sneps-Sneppe. "On Content Models for Proximity Services." In 24th Conference of Open Innovations Association (FRUCT). IEEE, 2019.

[31] Baert, Mathias, et al. "The Bluetooth mesh standard: An overview and experimental evaluation." Sensors 18.8 (2018): 2409.

[32] Wang, Mingjun, and Zheng Yan. "A survey on security in D2D communications." Mobile Networks and Applications 22.2 (2017): 195-208.