# Sociocast: Design, Implementation and Experimentation of a New Communication Method for the Internet of Things

L. Atzori[1,4], C. Campolo[2,4], A. Iera[2,4], G. M. Milotta[2,4], G. Morabito[3,4], and S. Quattropani[4]

[1]DIEE, University of Cagliari, Italy
[2]University Mediterranea of Reggio Calabria, Italy
[3]DIEEI, University of Catania, Italy
[4]CNIT - National Inter-University Consortium for Telecommunications, Italy

***Abstract*—Today, Internet can support the following data delivery schemes: unicast, multicast, broadcast, and anycast, according to the way in which the endpoints of the information exchanges are identified. However, several reasons exist discouraging network operators to actually offer all such data delivery schemes to end users. As a result, application developers can rely on unicast communications only and more complex group-based data dissemination policies are implemented as part of specific applications and services and through additional patches to the basic Internet implementation. Group-based communications are crucial in several Internet of Things (IoT) application scenarios. Therefore, in this paper a new data delivery scheme called *sociocast* is proposed, which can be safely offered to end users. In sociocast communication, endpoints are dynamically determined based on their mutual positions in a social network built by IoT nodes according to the Social Internet of Things paradigm. In this paper, it will be shown how sociocast can be utilized to address several networking needs.**

***Index Terms*—IoT, sociocast, SDN, Social Internet of Things**

## I. Introduction

The Internet is called to support a flurry of heterogeneous and evolving applications. In this transformation, obviously, the Internet of Things (IoT) [1] will play a significant role as it will strongly contribute to expanding the range of devices capable of exchanging data mutually. Predictably, this process will have a major impact on the methods of communication currently provided by the Internet. The ability of the latter to efficiently support the changed scenario, characterized by a massive presence of *group communications* (many-to-many and one-to-many) in addition to the traditional ones (one-to-one), will be put to the test.

At a close observation, it appears that the current group communication methods and relevant data delivery schemes supported by the Internet Protocol (IP), namely *broadcast*, *anycast*, and *multicast*, already show limits that inevitably will be emphasized in IoT scenarios. In fact, broadcast might represent a critical vulnerability of the Internet due to the numerous scalability and security issues [2]. Obviously, the literature suggests multiple solutions that operators can put in place to tackle the problems mentioned, but with the increase in the number of users interested in the service and with the increase in the size of broadcast groups, each measure adopted is destined to lose effectiveness. Anycast, for its part, is still in its infancy and is utilized by some network services only (such as the Domain Name System, DNS) [3]. For what concerns multicast, instead, it can be observed that several one-to-many applications and services are daily utilized by large user communities and growingly required by IoT (e.g., software updates). Practically, multicast data distribution is performed either at the application layer or through clumsy patches on top of the existing Internet architecture [4]. The main reason for this choice lies in the objective difficulties encountered in managing concurrent multicast services on a global scale and in the consequent vulnerability to which the operator would expose the network [5]. The aforementioned big hurdles and the consequent large costs in supporting the aforementioned group communication modalities in IP networks clearly emerge. This is the principal reason why most Internet application developers today can rely on unicast communications only in real deployments.

This paper aims to provide a new perspective to address the stated research issues, by proposing a novel communication method and data delivery scheme that is called *sociocast*. It is conceived to allow group-communications to be established between end-points according to their relationships over a social network of devices, in particular we refer to the Social Internet of Things (SIoT) [6]. The solution we propose is disruptive because it is based on exploiting the principles of the SIoT paradigm, hitherto conceived only at higher protocol levels, within the control plane of a future Internet network infrastructure. Accordingly, the main contributions of this paper are:

- the definition of the novel sociocast communication method which exploits the SIoT concept to support group communications in an efficient and effective manner;
- the design of all the functional elements required to support the proposed sociocast communication method in a Software-Defined Network (SDN) infrastructure;
- the experimental demonstration of the viability of our proposal, through the implementation of the aforementioned modules and the deployment of the conceived sociocast network application on top of the Open Network Operating System (ONOS) SDN controller [7];
- a validation of the proposed solution, leveraging the

978-1-5386-4980-0/19/$31.00 © 2019 IEEE

widely used mininet network emulator [8], which showcases the benefits of the proposal in some representative use cases.

The remainder of this paper is organized as follows. In Section II we motivate our work. In Section III we introduce the major sociocast concepts and discuss the design guidelines we have considered. In Section IV we describe the current implementation of the components needed to support sociocast over the Internet in details. Then, in Section V we show how sociocast can be exploited in representative use cases, before concluding in Section VI.

## II. Motivations

To address the difficulties encountered in enabling data delivery methods different from unicast in today's Internet, a wide variety of application- and network-based solutions are rapidly emerging that entail the definition of novel group-based communication methods.

Disruptive clean-slate architecture designs pursued by the research community to build the future Internet rethink the way data delivery services can be implemented. For instance, the Named Data Networking (NDN) paradigm [9] serves *natively the anycast model*, because the data exchange is not based on any specific content source to retrieve content from. In the MobilityFirst architecture [10], the *context-aware delivery primitive* is proposed which generalizes multicast to groups based on attribute-based descriptors.

Differently, groups of recipients can be set according to people's social relationships, e.g., people sharing common hobbies, social functions, and occupations, similar traffic conditions and environmental factors, the same mobility pattern. However, so far, social ties have been mostly exploited to improve routing performance in opportunistic and delay-tolerant networks (DTNs). For instance, to improve the successful message delivery in opportunistic networks, devices coming into contact should exchange only those messages that have a higher probability of being delivered to destinations, according to their social contact history (e.g., node centrality, in-betweenness) [11]. Most of the approaches either target unicast or multicast data delivery [12], with the exception of a few works. The issue of data broadcasting in a Mobile Social Network, where mobile social users physically interact with each other, is analyzed in [13]. The "small world" properties, typical of social networks, have also been extensively studied to address the routing problem in DTN [14].

Unlike the aforementioned literature, in our paper we intend to take a step forward in that:

- we consider the *social*-like features of devices rather than of their users only;
- we exploit social relationships to define a new communication method that facilitates data delivery among members of a social network of devices;
- we specifically consider the possibility to target as recipients of the data specific nodes of a social network, according to properly defined filters and policies;
- the scope of our solution does not span a single standalone network domain (e.g., delay tolerant and opportunistic networks), but the whole Internet.

## III. Sociocast overview and requirements

*Sociocast* is a novel solution that allows group-based communications in the IoT enhanced with the notion of social ties. In particular, it has been conceived to *(i)* identify in a dynamic manner the end-points of data flows based on their distance in a social network of devices and *(ii)* improve data exchange procedures among them.

To this aim, according to the SIoT paradigm [6], by augmenting devices with the capabilities to create and manage social links, as the humans do in their social life, we expect them to be able to identify the trusted communication endpoints. Thus, the devices will mimic the human behaviour in analysing the profile and trust of each community member by also leveraging on the crowd view.

In our solution, a *Sociocast Relationship Service*, defined at the network control plane, interacts with the SIoT with the purpose of identifying the members of the sociocast group and, then, it enables the efficient delivery of data packets among them at the data plane. Accordingly, a sociocast device, $A$, can generate a sociocast packet, which will be delivered to all devices that have specific positions in the SIoT in relation to the source device $A$. Such positions are defined by the source device $A$ by specifying the maximum distance and the type of social relationships which must be considered.

When a sociocast packet arrives at the first network element, say $n_0$, supporting sociocast, the Sociocast Relationship Service will be queried. This will, in turn, provide $n_0$ with the addresses of devices that are in the positions specified by device $A$. Therefore, delivering the packet to the above devices will be responsibility of the network elements supporting sociocast.

More specifically, in the design and implementation phase of sociocast our major goal has been to provide a solution that can be immediately exploited by network operators and managers as well as application/service developers and experimenters.

Accordingly, the requirements we have considered are:

- **No need for changes in the core functions of end host operating system**. Users are reluctant to update or even configure the operating systems in their devices whereas they are willing to install new applications and apps. Therefore, at this stage sociocast should only involve application layer functions in the end hosts.
- **Incremental deployment**. Support of sociocast should be possible even if based on a very small number of network elements. In fact, support of sociocast as a communication configuration by network providers will happen only if there is a sufficiently large number of applications using it. However, applications which exploit sociocast will be developed only if sociocast is already supported.
- **Compatibility with IPv4**. While we believe that specific *sociocast* packet types should be defined, we are well

aware of the difficulties and duration of the needed standardization process. Therefore, sociocast should be conceived in such a way that, in the initial stage of its introduction, it can work by exploiting existing packet types.

- **No need for an entity which manages the service**. One major component envisioned in sociocast is the Sociocast Relationship Service, implemented in a distributed and peer-to-peer fashion to avoid the presence of a single management entity with full control on the sociocast operations which would be a crucial limitation.
- **Independence from the specific network layer deployment.** Without loss of generality, at this stage of research, we assume the network infrastructure to be deployed through the SDN paradigm, neatly decoupling the control and data plane and leveraging the centralized intelligence of a controller entity to decide packet forwarding rules. However, it is worth to remark that the proposed sociocast communication method can be leveraged on top of whatever network infrastructure, provided that the Sociocast Relationship Service is designed with the proper Application Programming Interfaces (APIs) to interact with network elements so to let them forward sociocast packets to the intended social destinations.

## IV. Sociocast design and implementation

In this section we present a full solution supporting sociocast in current networks. More specifically, in Section IV-A we will first describe the components needed to support sociocast and their interactions. Then, in Section IV-B we analyze how the sociocast data delivery protocol works.

### *A. Architecture and deployment*

The reference architecture for our system encompasses the following elements: devices, SDN gateways, SDN nodes, the SDN controller and the Sociocast Relationship Service, also depicted in Figure 1.

The **devices** can be the starting or terminating points of a communication (e.g. PCs, smartphones, IoT devices, etc.). They are enabled to create, send and/or receive sociocast packets, filter the incoming traffic based on social relationships with other devices, create and/or join a multicast group that is based on a specific social relationship. This is possible thanks to the Sociocast Support Layer (ScSL), that is responsible of the correct creation and reception of the sociocast packets and sociocast tags. It exposes the sociocast APIs to the applications that want to use the sociocast communication configuration for data delivery.

The **SDN gateways** are the ingress/egress nodes of the SDN network. End-points are directly connected to them, through links that can be also overlay. SDN gateways can communicate with the SDN controller and are responsible for the correct forwarding of sociocast packets.

The **SDN nodes** are SDN-enabled network nodes which are connected to each other and to the gateways and interact with the SDN controller. It is easy to guess that SDN gateways are a special type of SDN nodes.

The **SDN controller** acts as a logically centralized element orchestrating the SDN network. It runs the *sociocast network application* (SNA) which defines the forwarding policies towards the sociocast destinations, as retrieved by interacting with the Sociocast Relationship Service.

The **Sociocast Relationship Service** has to handle digital counterparts of the physical devices. They keep track of meta-data providing information about the nature of the device, the list of friends, feedback about the trust level associated with them, etc. Information about the type(s) of friendship(s), defined according to the SIoT paradigm, are kept for each friend [6]: co-ownership object relationship (OOR), co-location object relationship (CLOR), parental object relationship (POR), co-work object relationship (CWOR), social object relationship (SOR).

Each device's digital counterpart and all the information about the social relations associated with it are contained in a purpose-built distributed data structure. This latter can be configured according to policies set by the owner of the SDN infrastructure.

All the policies and mechanisms are provided by the Sociocast Relationship Service, to create digital counterparts of the devices, as well as to interact with the SIoT at the application layer to receive information and updates relevant to the establishment of social relationships between devices. In particular, the following functionalities are included:

- the *Relationship Manager* (RM), which is responsible for the relationships' lifecyle management, i.e., detecting, creating, updating and deleting relationships;
- the *Sociocast Handler* (SH), which handles the sociocast data delivery service, by interacting with the SDN controller, whenever queried to provide the members of the sociocast group.
- the *Relationship Browser* (RB), which, when triggered by the SH, navigates the social network to find further potential recipients of a sociocast packet, according to their position in the social network.

### *B. Protocol Description*

In the following, we detail the main steps for the creation of the sociocast group and the sociocast data delivery. Such procedures are triggered when the application of a device, say A, acting as a source device, wants to use the services offered by the sociocast framework (e.g., to create a sociocast group).

1) The application in device A makes a request to the ScSL. Via the available APIs, it provides the following information:
   - the kind of sociocast feature needed;
   - the social relationship (e.g., OOR, CLOR) according to which the sociocast group has to be formed;
   - the social distance (number of hops over the social network), which represents the scope of the sociocast group.

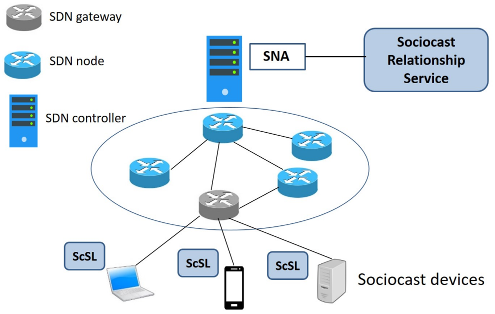


Fig. 1. Main entities.

2) The ScSL reacts to the incoming request by creating an IP packet containing the following information in the header:
SOURCE IP ADDRESS: the source device public IP address.
DESTINATION IP ADDRESS: a fixed public IP address, identified in this paper as $IP_{SC}$, assigned to sociocast that allows SDN gateways to identify sociocast packets. The value of $IP_{SC}$ is 151.97.13.77.
SOCIOCAST TAG: a 2-byte field that is carried inside the destination port and is used to uniquely identify the type of sociocast relationship and other appropriate filters (e.g., number of hops, possible application of sociocast, etc.). The encoding is as follows:
   - Metadata (bit 0-3): device metadata available for future applications.
   - Relationship (bit 4-7): type of relationship (e.g. OOR, SOR, C-LOR, etc.).
   - Feature (bit 8-11): type of sociocast feature needed by the application.
   - Hop (bit 12-15): Maximum distance in hops from the source.

   The source device sends the created packet.
3) The sociocast packet reaches the SDN gateway, which the source device is connected to. Since, initially, a rule is not set in the flow table of the SDN gateway, the GOTOCONTROLLER rule applies for it.
4) Upon receiving the header of the sociocast packet, the controller realizes that a sociocast group must be created. Thus, it issues a request to the Sociocast Relationship Service, to retrieve the set of the devices that satisfy the request from the application.
5) The SH of the Sociocast Relationship Service triggers the browsing of the social network, as specified before, and returns the set of devices of the sociocast group to the controller which will assign an unused multicast address and create the corresponding group.
6) According to the reply from the SH, the controller creates a multicast IP-based routing path to reach all the SDN gateways to which devices belonging to the sociocast group are connected. Details on how this step is performed are provided in the following sections. The SDN gateways that are involved in the sociocast group communication, will be instructed by the controller with a rule that: *(i)* matches the destination IP address and the destination port of the incoming sociocast packet and *(ii)* foresees to forward the packet to the correct port after changing the destination sociocast IP address with the IP destination address as action. This is to ensure that all devices attached to the SDN gateways and belonging to the sociocast group receive the sociocast packet.

Once the sociocast group is created, subsequent sociocast packets transmitted by the source device are handled by the SDN gateway with no need to contact the controller, but rather forwarded according to rules already available in the flow table. The creation and management of the flow table rule by the controller will be analyzed in details in the Section V-A by referring to a practical exemplary use case.

## V. SOCIOCAST IN ACTION

Sociocast can be exploited to address some key issues regarding the control of packet distribution in the network. Today, solutions to such issues already exist but are handled in an heterogeneous often application-dependent manner. More specifically, we will start by demonstrating how sociocast can be used to realize a *push service* which allows a source device to send a certain message to all devices that are within a given distance in the SIoT. Then, we will discuss how the same mechanism can be used to support *publish-subscribe* services.

### A. Sociocast-based push service

By exploiting sociocast it is possible for a given device $A$ to send packets to all the devices within a certain distance in the SIoT. In other terms, it is possible to realize a *push* service in which the destinations are not known *a priori*. Such a service can be effectively utilized in several applications scenarios, such as:

- *Software updates:* by exploiting links of POR type, it is possible to deliver a given software patch to all the devices of the same brand, model, batch.
- *Service advertisement/discovery:* by exploiting links of CLOR type, it is possible to advertise (or search) a given service to all the devices that are currently in the same area.
- *Personal bubbles:* by exploiting OOR links, it is possible for a device to send messages to all other devices belonging to the same owner.

In the following of this section we show how sociocast supports a push service in an experimental testbed.

The experimental setting consists of the network depicted in Figure 2, which resembles a fat-tree topology. It has been realized by exploiting mininet, with its built-in support for SDN, and ONOS has been considered as a reference controller in the context of this work, due to its scalability properties and its highly modular architecture [7]. There are seven OpenFlow (OF) switches, i.e., $Sw1$, $Sw2$, ..., $Sw7$, and eight devices, i.e., $H1$, $H2$, ..., $H8$, which are tied by the social links.

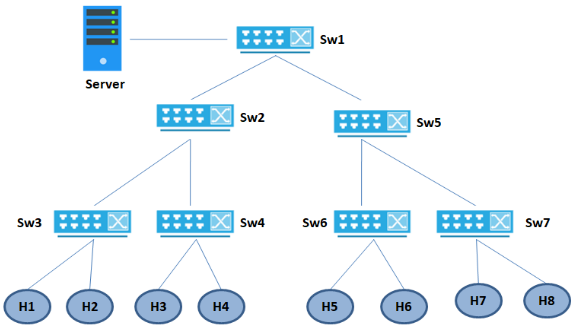


Fig. 2. Network topology for the experimental campaigns.

At a given time, device $H8$ with IP address $IP_{H8}$ wants to send a given packet to all devices that are linked to it with a relationship of type OOR. Accordingly, it generates a UDP packet with source IP address set to $IP_{H8}$, the source port is given by the application and destination IP address set to the IP address identifying sociocast packets, i.e., $IP_{SC}$. The destination port is utilized to carry the sociocast tag described in the previous Section IV. In this case, the sociocast tag is

Metadata = 0000
Relation filter = OOR = 0001
Feature = Group Creation = 0000
Radius = 1 hop = 0001
*0000000100010010* → 257 (UDP Port)

The packet generated by $H8$ does not fit the *rules* specified in any of the entries of the flow table in $Sw7$, therefore, upon arriving to $Sw7$, the header of the packet is forwarded to the controller. The latter one analyzes the header and extracts the information necessary to perform the query to the Sociocast Relationship Service. Therefore, it issues a POST containing the following information:

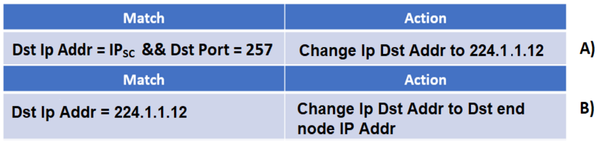
– Device ID (known from ONOS);
– Sociocast-Tag.RelationType, in this case "OOR".

The RESPONSE will contain the list of all IP addresses that meet the search requirements. In our case, devices related to $H8$ by OOR relationships are $H3$, $H5$, and $H6$. The ONOS controller can, therefore, create the sociocast group to which it assigns an IP multicast address. In this phase the SDN controller maps the multicast routes which connect the selected devices and bounds these devices with the chosen IP multicast address:

- 224.1.1.12 : $[IP_{H3}, IP_{H5}, IP_{H6}]$.

After creating the group, the SDN controller will send the translation rule for the flow table to the gateways belonging to the group. The fixed IP address used to identify sociocast packets is translated into the IP multicast address chosen by the controller. Each gateway of each sociocast group participant e.g., gateway of devices *H3*, *H5*, *H6*, *H8* will be implemented with the flow rules in Figure 3.

The flow rule A shown in the Figure 3, allows to send a sociocast packet and applies at the gateway of device $H8$, while the rule B shown in Figure 3, allows to translate the multicast IP address into the public unicast IP address of the receiving device and is injected into SDN gateways of the intended sociocast destinations. Thanks to this, the end point does not have to join the group in order to receive packets, as typically done in IP-based multicast groups [15]. All the following sociocast packets will be delivered without further interactions between switches and controller.



| Match | Action | |
|---|---|---|
| Dst Ip Addr = $IP_{SC}$ && Dst Port = 257 | Change Ip Dst Addr to 224.1.1.12 | A) |
| **Match** | **Action** | |
| Dst Ip Addr = 224.1.1.12 | Change Ip Dst Addr to Dst end node IP Addr | B) |

Fig. 3. SDN flow table rules.

In the rest of this section we compare the behavior of sociocast, explained above, with what would happen by supporting a similar service at the application layer. We therefore, assume that one of the hosts, say host $H9$ is a server that interacts with the SIoT and is responsible for distributing the sociocast packets received by $H8$ to all intended destinations, i.e., $H3$, $H5$, and $H6$.

In Figure 4 we show the number of incoming and outgoing packets at each switch in the following cases: *(i)* when a legacy unicast approach is leveraged; *(ii)* when using a server/middlebox which is responsible for replicating packets towards the intended destinations; *(iii)* in the sociocast scheme. A legacy multicast approach has not been considered in this scenario because, in terms of incoming and outgoing packets from the switches, it produces the same results as the sociocast.

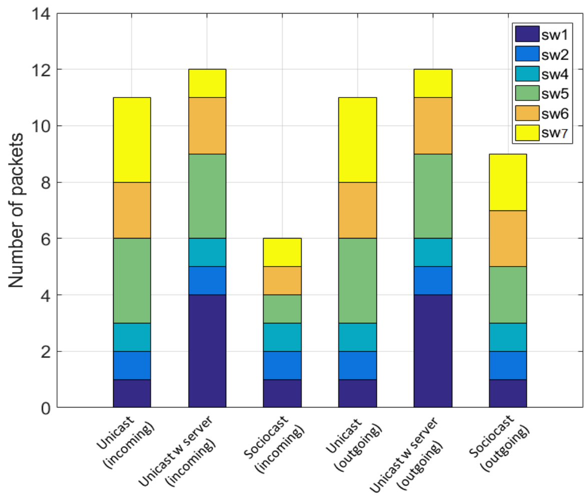


Fig. 4. Ingoing and outgoing packets at each switch for the compared schemes.

### B. Publish-Subscribe

Sociocast can be exploited to support publish-subscribe interaction model as well. In fact, a device can *subscribe* to receive packets *published* by devices identified by their position in the SIoT. For example, assume that device $H8$ in Figure 2 wants to subscribe to receive packets generated by its *friends* of type OOR. If this is the case, it will generate a packet in which the source IP and port addresses are set as described in the previous section, the destination IP address is $IP_{SC}$, whereas the destination port is:

Metadata = 0000
Relation filter = OOR = 0001
Feature = Publishe-Subscribe = 0000
Hop = 1 hop = 0001
*0000000100010001* → 273 (UDP Port)

Such an information will reach the controller which will perform the following operations:

1) Send a query with this information to the Sociocast Relationship Service and receive the identities of the devices with position in the social network consistent with the request by device $H8$. As discussed previously, in this case such devices are $H3$, $H5$, and $H6$.
2) Insert this information in a pending interest table which stores information about all subscriptions received by devices. When a device begins to publish data as described in Section IV, the controller checks whether there are devices that have subscribed to their updates and in case, it will act as described in item 3.
3) Check if one of the above devices is an active publisher. If this is the case, the controller will send the appropriate flow entries to the OF switches. In our case we assume that $H3$ is active and therefore appropriate entries will be sent to Sw1, Sw2, Sw4, Sw5, and Sw7.

Let us refer to the following example. At time $t = 29$s, device $H8$ generates a subscription as described above and therefore, the packets published by $H3$ begin to reach $H8$.

This is reflected in Figure 5 where we show the packet reception rate for device $H8$ over time.

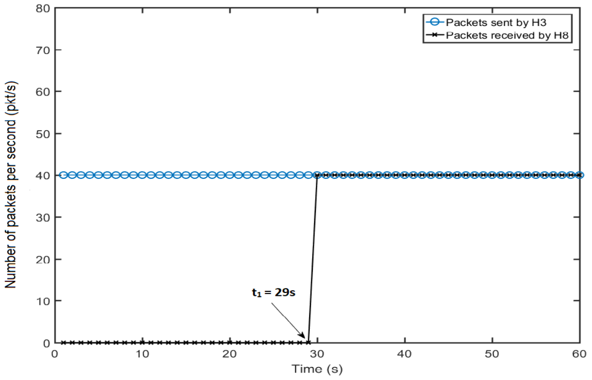


Fig. 5. Traffic transmitted by $H3$ and received by $H8$ Vs. time.

## VI. Conclusions

In this paper we introduced *sociocast*, a novel communication method that identifies the end-points of data exchanges based on their position in the SIoT. Sociocast candidates itself as a prominent solution to flexibly support a wide range of applications that entails *group-based* communications *dynamically established among devices tied by social relationships* (e.g., software updates, service discovery/advertisement, personal bubbles). The design of all the functional components and their behaviours needed to support *sociocast* has been discussed, when referring to a *software-defined network infrastructure*. In addition, the interfaces that can be used by application developers to exploit sociocast, as well as those required by the network control plane to enforce sociocast-based data delivery have been described. An experimental playground, based on the mininet network emulator and the ONOS SDN controller, has been used to practically showcase the viability of the proposal and its benefits, when compared to legacy alternative solutions. Evaluation under realistic data traffic patterns and social relationships dynamics will be a subject matter of future work.

## Acknowledgement

This work was partially supported by the European Union's Horizon 2020 research and innovation program under the COG-LO project (grant agreement no. 769141).

## References

[1] L. Atzori, A. Iera, and G. Morabito, "The internet of things: A survey," *Computer networks*, vol. 54, no. 15, pp. 2787–2805, 2010.
[2] J. Granjal, E. Monteiro, and J. S. Silva, "Security for the internet of things: A survey of existing protocols and open research issues," *IEEE Communications Surveys & Tutorials*, vol. 17, no. 3, 2015.
[3] S. Weber and L. Cheng, "A survey of anycast in IPv6 networks," *IEEE Communications Magazine*, vol. 42, no. 1, pp. 127–132, 2004.
[4] M. Hosseini, D. T. Ahmed, S. Shirmohammadi, and N. D. Georganas, "A survey of application-layer multicast protocols," *IEEE Communications Surveys and Tutorials*, vol. 9, no. 1-4, pp. 58–74, 2007.
[5] C. Diot, B. N. Levine, B. Lyles, H. Kassem, and D. Balensiefen, "Deployment issues for the IP multicast service and architecture," *IEEE network*, vol. 14, no. 1, pp. 78–88, 2000.
[6] L. Atzori, A. Iera, G. Morabito, and M. Nitti, "The social internet of things (SIoT)–when social networks meet the internet of things: Concept, architecture and network characterization," *Computer networks*, vol. 56, no. 16, pp. 3594–3608, 2012.
[7] "ON.LAB, "introducing ONOS - a SDN network operating system for service providers," 2014.
[8] B. Lantz, B. Heller, and N. McKeown, "A network in a laptop: rapid prototyping for software-defined networks," in *Proc. of the 9th ACM SIGCOMM Workshop on Hot Topics in Networks*. ACM, 2010, p. 19.
[9] L. Zhang, A. Afanasyev, J. Burke, V. Jacobson, P. Crowley, C. Papadopoulos, L. Wang, B. Zhang *et al.*, "Named data networking," *ACM SIGCOMM Computer Comm. Review*, vol. 44, no. 3, pp. 66–73, 2014.
[10] t. Venkataramani, "MobilityFirst: a mobility-centric and trustworthy internet architecture," *ACM SIGCOMM Computer Communication Review*, vol. 44, no. 3, pp. 74–80, 2014.
[11] X. Hu *et al.*, "A survey on mobile social networks: Applications, platforms, system architectures, and future research directions," *IEEE Communications Surveys & Tutorials*, vol. 17, no. 3, pp. 1557–1581, 2015.
[12] W. Gao, Q. Li, B. Zhao, and G. Cao, "Multicasting in delay tolerant networks: a social network perspective," in *Proc. of ACM MobiHoc*, 2009, pp. 299–308.
[13] J. Fan *et al.*, "Geocommunity-based broadcasting for data dissemination in mobile social networks," *IEEE Trans. on Parallel and Distributed Systems*, vol. 24, no. 4, pp. 734–743, 2013.
[14] K. W. *et al.*, "Exploiting small world properties for message forwarding in delay tolerant networks," *IEEE Trans. on Computers*, vol. 64, no. 10, pp. 2809–2818, 2015.
[15] B. Cain *et al.*, "RFC 3376, Internet Group Management Protocol, version 3," Tech. Rep., August 2006.