Combining Data Naming and Context Awareness for Pervasive Networks

Paulo Mendes

{*paulo.mendes*}@*ulusofona.pt*
(+351)925982488
COPELABS, University Lusofona,
Building U, first floor, Campo Grande 376 1749-024 Lisbon
Portugal

Abstract

In recent years large attention is being devoted to a new networking trend focused on data centricity. The major motivation has been the development of a framework able to mitigate the mismatch between the current Internet architecture - host based - and the way most users use the network - agnostic of content location. The first proposals were focus on fix infrastructures in an attempt to devise a data-centric solution for the global Internet. On the other hand, the number of available wireless Internet access points and wireless mobile devices is growing significantly. Hence, it is very important to have in mind scenarios where networked pervasive systems need to communicate independently of the networking conditions. In such pervasive networks, it is important to support the development of communication services aware of users' context: the goal would be to increase data usefulness and delivery probability, as well as to reduce cost and latency. This article presents an information and context oriented networking framework (ICON) able to support the deployment of pervasive networks by combining two networking paradigms that are highly correlated to the efficiency of data sharing: data-centric networking and opportunistic networking.

1 Introduction

Nowadays, most of us use the Internet for some kind of activity on a daily basis. To support an increasing user demand, Internet evolved to include not only the bulky desktop computer, but also more pervasive systems, like smartphones and embedded devices found in intelligent buildings, vehicles, and public places. The popularity of networked pervasive systems exacerbate problems related with network and data management due to their mobility, intermittent availability,

as well as software and hardware heterogeneity. These issues are more relevant when users start to have an active role as a networking entity, and not just as a consumer and producer of data [32, 33].

Besides the growth in the number of networked pervasive systems, the way people make use of the Internet to access and share data is not aligned with the way the Internet works. People care about obtaining content, while being agnostic of its location and the mean needed to get it (e.g., access via a search engine). However, from the Internet perspective data location is essential to the networking process: when someone asks for content by name, that name has to be resolved to a topologic location that the network can identify. Moreover, the current mobile Internet architecture is mostly agnostic of users' context. This property makes the current Internet architecture far from ideal in pervasive networking scenarios, where hosts are mobile, have multiple network interfaces, and may have intermittent connectivity. The latter may happen in extreme cases due to wireless fading, and the absence of Internet infrastructure.

When networked pervasive systems are deployed in urban scenarios, users may have access to wireless Internet access with high probability. In such scenario, data sharing may occur via the Internet. However, Internet access may be avoided, leading to intermittent connectivity, due to two major reasons: expensive wireless Internet access (e.g., 3G systems); participatory wireless Internet access (e.g., Wi-Fi systems). The latter requires frequent user intervention (e.g., authentication), which is not compatible with the opportunistic behaviour of some pervasive systems, such as sensing.

Hence, in a networking scenario where an increasing number of pervasive systems may be used to share data anytime and anywhere, there are two networking paradigms that are highly correlated to the efficiency of pervasive data sharing: data-centric networking and opportunistic networking. In the latter case, opportunistic networking means the capability to exploit any potential communication, which can be provided by mobile devices or by Internet access points.

On the one hand, although some data-centric networking architectures consider mobile access networks [11], where end devices are mobile, such proposals assume the existence of a wired access network and, in some cases, of a centralized control infrastructure (e.g. rendezvous points [16]). Such networking assumptions are not applicable to networked pervasive systems, which can operate as end-devices or forwarders, can be mobile, and may have intermittent connectivity. This is, current data-centric architectures are not suitable for the operation of pervasive systems in opportunistic networking scenarios. On the other hand, most opportunistic networking solutions [10, 27, 34, 13] are host-centric: data is forwarded to specific destinations. However, data-centric networking may bring advantages to the deployment of pervasive systems in opportunistic networking scenarios, by reducing buffer occupation, since transmitted data can be shared by different receivers.

To the best of the author's knowledge, this is the first framework that aligns the data-centric networking and opportunistic networking paradigms to support the deployment of pervasive systems, by combining networking, data, and con-

text management. The proposed information and context oriented networking framework (ICON) extends CCN [15] alike data-centric architectures to allow operation over opportunistic networking scenarios, based on the following major contributions: dynamic configuration; sensing abstraction; social and motion inference; opportunistic forwarding, and message encoder based on protobuffers.

The remainder of this paper is organized as follows. In Section 2 an analysis of prior art related to data-centric networking and opportunistic networking is provided, with special attention given to approaches that aim to combine these two networking paradigms. Section 3 briefly describes a general scenario that should be considered in the design of pervasive networks. In Section 4 the ICON framework is described, including the building blocks related to data, context, and networking management. In Section 5 a study of ICON usefulness for different applications is provided, followed by an experimental analysis of its efficiency to exchange data in pervasive networking scenarios. The article is concluded in Section 6.

2 Related work

The deployment of networked pervasive systems requires a networking framework able to allow pervasive systems to share data anytime and anywhere. Such framework should be devised by combining two networking paradigms that are highly correlated to the efficiency of pervasive systems: data-centric networking and opportunistic networking. In this context, the goal of this section is twofold: first to analyze the limitations that current data-centric architectures and opportunistic networking solutions have to support the deployment of pervasive systems; Second, to analyze the properties of frameworks that may be used to combine data-centric and opportunistic networking.

In the last couple of years several data-centric networking architectures have been proposed to address the problems raised by the Internet dependency upon the location of content, namely mobility and multihoming management. Among them, the Content Centric Networking (CCN) [15], the Publish-Subscribe Internet Routing Paradigm (PSIRP) [16], the Network of Information (NetInf) [11] and the Data Oriented Network Architecture (DONA) [17]. These architectures represent a major shift in what concerns IP networking architectures, by being developed with the same receiver-oriented strategy presented in data-centric architectures. Such strategy aimed to allow multiple consumers to use publish-subscribe services to get data under specific usefulness expectations [8, 7].

Despite their differences regarding issues such as naming convention and resolution, as well as security, all these data-centric approaches consider wired access to the Internet. In the case of NetInf, mobile access networks are considered to investigate the potential instability caused by mobility of sources and consumers of data. Nevertheless, NetInf still distinguishes between static data forwarders and mobile data sources and consumers. In the case of CCN, it presents some properties useful in pervasive scenarios: it is decentralized and supports customization of operational aspects, by means of the strategy layer.

However, the CCN concept is not aware of users' behavior and does not support forwarding in opportunistic networking scenarios.

This article is looking at more dynamic networking scenarios, where all nodes (sources, consumers and forwarders) are or may be mobile and may be intermittently available or connected. In such dynamic scenarios, some effort has been done to show that data-centric architectures can also handle real-time data [14] and work with dynamic networks like MANETs [21, 20, 30]. Although MANETs operate in scenarios more dynamic than the wired Internet, they assume that it is possible to find a path among any pair of nodes, in any moment in time. However, such connectivity scenario may not occur in pervasive networks. In pervasive networks, devices should be able to exploit any wireless contact opportunity to forward messages based on some criteria (e.g., always forward; analyze the likelihood of finding the destination node; only forward if the node is the destination) in a deterministic or probabilistic way: this is, pervasive deployment requires an opportunistic networking approach.

Moreover, it has been shown the advantages of using context-awareness to improve networking operations in mobile systems, such as handover [37]. However, existing data-centric networking proposals are in general agnostic of the context of sources, consumers and forwarders of data. To support networked pervasive systems that may forward data opportunistically, it is important to make use of the sensing capabilities of such systems to extract context information, which can be used to improve the efficiency of data exchange. Among pervasive systems, smartphones are of particular importance, since they exist in a large number, and are used in a ubiquitous way. Smartphones are the right pervasive system to extract information about people daily habits, as well as their surroundings (e.g., smartphones can be plugged in to other sources of context, such as vehicles).

Hence, combining content knowledge, as did by data-centric approaches, with context information, such as behavior and social proximity, shall bring benefits (faster, better content reachability) to networked pervasive systems [26]. This statement is supported by prior analysis that shows that an increase of the performance of opportunistic forwarding can be achieved by: i) focusing on content rather then on hosts [10, 6]; ii) and being aware of social communities [13] or users' habits [27, 25, 29, 28].

In what concerns context awareness in data-centric networks, some prior work aims to exploit users' contextualization in data-centric networks, being context limited to the service type requested by a potential data receiver in fix networks, such as home network [5]. In what concerns more dynamic scenarios, ICAN [38] aims to enable a context-aware ad-hoc network, where application context is used to select the most suitable transport operation (push or pull based).

Currently there is no published research that combines data-centric networking and opportunistic networking by providing a solution that coordinates networking (opportunistic forwarding), data (in-network caching) and context (social inference, motion inference) management, aiming to deploy an efficient data-centric pervasive network, by exploiting the sensing properties of networked

personal devices.

Following this challenging of combining data-centric and opportunistic networking, G. Tyson et al. [35] explores the potential of combining the information-centric networking and delay-tolerant networking concepts, with focus on pocket switched networks. However, the proposed ICDTN model is just focus on forwarding, lacking any information about data management and context management, which are required to allow pervasive systems to adjust to different scenarios. Moreover, the opportunistic forwarding of ICDTN is based on flooding. Contrary to the ICDTN proposal, the ICON framework provides a specification of the data, context, and networking management. In the latter case ICON incorporates a social-aware forwarding mechanism in a data-centric approach, based on the capability to sense users' daily routines. Results show that the proposed opportunistic forwarding scheme presents better performance than other opportunistic forwarding approaches.

Another contribution aiming to combine data-centric and opportunistic networking is provided by I. Psaras et al. [31]. Authors aim to propose a mobile name-based replication system (NREP), where message replication is limited by time and space: that is, data sharing occurs within a certain geographic area and with specific life expectancy. As occurs with ICON, NREP focus on data dissemination and not on point-to-point communications. To achieve the proposed goal NREP analyzes the usefulness of exploring parameters such as priority, time-to-live and geographical constraints in the name. This correlation between name design and forwarding is a major different towards ICON. With ICON, name design is application driven and may include context information, such as geolocation (in the case of sensing applications), but name design does not have an impact on forwarding. Forwarding is based on context related to social proximity and data-interest similarity and not on context related to data, such as priority and geographical visibility. This uncorrelation allows ICON to support different applications (c.f., section 5.1), while NREP focus towards a specific type of application.

Finally, C. Anastasiades et al. [1] introduces and motivates the usage of ICN in mobile and opportunistic networks, and reviews several ICN approaches. The authors follow a tutorial approach, aiming to point out the benefits of using ICN in two distinct scenarios: mobile and opportunistic networking. In the latter case, the authors focus on content discovery and transfer. Content discovery is performed using multicast to quickly detect nearby available sources. This means that the proposed solution requires the dissemination of interest messages, which may lead to an increase of network traffic. In cases where a data requester never meets a valid content source, the proposed solution is to use an agent-based content retrieval: requesters can delegate content retrieval to agents, which retrieve content on behalf of the requesters. However, delegation of data retrieval may lead to a high computational and network burden. ICON follows a different approach: interest messages are not disseminated in the network. ICON proves that a good performance can be achieved by devising a social-aware forwarding mechanism, without the need to propagate interest messages in the network or to create agents.

Figure 1: Heterogeneous pervasive networking scenario

3 The Case of Pervasive Networks

A pervasive networking use-case is generally characterized by having a high diversity of mobile and intermittently available devices, which communicate in a network where wireless connectivity may be also intermittent. Such use-case can be related with networking in challenged scenarios (e.g., critical-mission networks), as well as in urban scenarios. In the latter case, pervasive networking can be associated with scenarios where direct wireless communications are analyzed from the perspective of mobile operators to reduce operational costs (e.g., D2D technology), as well as from the perspective of end-users to reduce communication costs and disruption (e.g., wireless direct, adhoc networking).

Figure 1 illustrates an heterogeneous scenario that can be considered to evaluate the applicability of data-centric solutions in pervasive networking environments. This scenario is composed by different networking situations: communication over a fix global infrastructure to access cloud systems; communication over a wireless local network (WLAN) used mainly for wireless Internet access; direct communication between devices.

The networking scenario based on Internet communications and WLANs is today operational based on an host-to-host IP network able to sustain access to data, as well as real-time communications. A more pervasive networking scenario involving direct communication between embedded and personal devices (e.g., Internet-of-things or content driven delay-tolerant networking) is not fully deployed in the real world, mainly due to the limitations posed by the host-to-host communication model for pervasive operations over more dynamic networking scenarios.

Therefore a data-centric networking framework needs to be able to support a wide set of applications over dynamic networking scenarios. This provides the

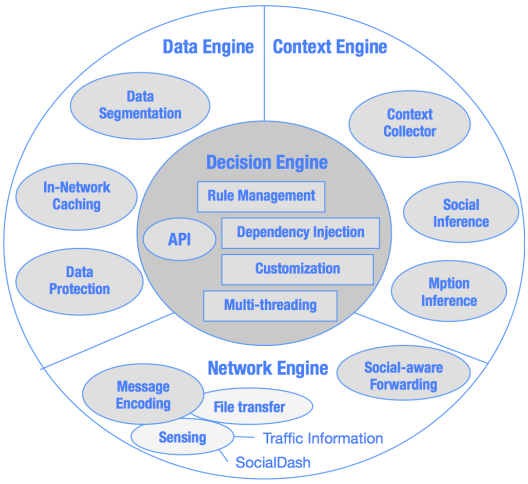

Figure 2: ICON node design

strategic motivation for the development of ICON.

The deployment of data-centric pervasive networks requires a framework that is: i) easily implemented in different devices; ii) easily extended to support novel functionality (e.g., naming schemes); iii) able to exchange data in networking disruptive situations; iv) able to insure data usefulness for the user (mapping data exchange and users' behavior). These are the major requirements considered in the design of ICON.

4 ICON Framework

The Information and Context Oriented Networking framework (ICON) aims to support data-centric networking virtually in any networking scenario. This goal is pursuit by combining the data resilience properties of data-centric networking, with the connectivity resilience provided by opportunistic networking. With ICON, networked pervasive systems make use of any available communication opportunity, which can be provided by another mobile device or by an Internet access point (e.g., a ICON aware Wi-Fi access point). To achieve an efficient operation in pervasive network scenarios, ICON incorporates four components, as illustrated in Figure 2: Decision Engine, Data Engine, Context Engine and Network Engine.

An efficient data sharing over opportunistic networks is ensured by the Network Engine, which encompasses a social-aware opportunistic algorithm that does not require the usage of bread crumbs reverse paths, as done by CCN. The proposed Network Engine also differs from the networking set up of CCN in the following aspects: ICON forwarding mechanism does not require a Forwarding Information Base (FIB); does not use 'bread crumbs'; data packets do not

consume interests; and data and interests have explicit Time-To-Live (TTL).

The implementation of social-aware forwarding schemes require the capability to infer about the social and mobility context of pervasive devices, in order to allow local decisions about potential next hops. For instance from a set of neighbours with similar probability to meet nodes with a certain set of data interests, the ones that present a fast motion profile may be the best forwarders. Such context awareness is provided by the Context Engine, which encompasses: i) a sensing abstraction that allows sensors to be shared among different devices; ii) a social inference algorithm that allows nodes to gather information about social proximity based on the time and duration of wireless contacts; iii) a motion inference algorithm, which is implemented based on Support Vector Machines. The motion inference algorithm allows ICON to use mobility behavior patterns to control the way data interests are present in the network, as well as to influence data forwarding decisions.

Besides the support for reliable data exchange over opportunistic networks, ICON also fulfills other major requirement of pervasive systems: adaptation without disruption of operation. This goal is achieved by the Decision Engine, which allows modules to be replaced at runtime, by using dependency injection, and to be configured by a set of management rules.

The Data Engine allows an ICON node to control the storage of data elements within networked devices, by means of an in-network caching system. Such data elements can be encrypted when they are added to messages. Data messages are then segmented, and exchanged based on the social-aware opportunistic forwarding algorithm implemented by the networking engine. The latter is also responsible for encoding data messages before transmission: ICON uses protobuffers as message encoder, instead of the binary scheme used by CCN.

The in-network caching used in the Data Engine is based on a simple LRU replacement policy, in order to show that a good network performance can be achieved with a simple caching implementation. As future work, the Data Engine will allow cache entries to be removed based on the inference about their importance to neighbour devices, as mentioned in section 6.

The remainder of this section provides detailed information about the four components of the ICON framework: decision, data, networking and context engines.

4.1 Decision Engine

This section describes the software engineering aspects that aim to implement the dynamic nature of ICON. First of all, sensors and communication interfaces are bootstrapped at runtime, through type bindings previously defined on an existing xml settings file. Such interfaces are then seen to the application as plugins for creating device profiles. Since the active interfaces can change at runtime, it is necessary to reduce the number of dependencies among all the software modules. This is achieved by implementing component containers, which are part of a common software engineering pattern called Dependency Injection, also known as Inversion of Control. Dependency Injection enables

registration of concrete interface implementations on a container and injecting them when appropriate, resolving all the dependencies needed, whether they are constructor parameters or properties. All active sensing and communication interfaces can be used simultaneously by making use of the multi-threading property of the .NET framework.

At runtime, running modules can be customized by the rule management module, which allows the introduction, at any moment, of rules that can alter the internal or external behavior of a node. These rules can be static or dynamic: in the former case, the user establishes a set of rules that can only be changed by him/herself; in the latter case rules can be changed by the Decision Engine itself, based on the information that it gets from other modules, such as the social and motion inference modules. Since ICON is built to take advantage of social similarities, and to be aware of users' behavior (motion), it is configured with two default rules:

Rule 1: forwarding decisions must consider the social strength among users. This rule takes advantage of the principle that says that people with strong social similarities have high probability to meet.

Rule 2: interests that applications have on data (video, audio, text, sensing) should be mapped to the user's motion state. This rule insures that data has a good usefulness level at any moment in time.

As occurred with the definition of interfaces, operational rules are configured through type bindings previously defined on an existing xml configuration file. Rules are used to change the operation of existing module at runtime or to force the replacement of modules, making use of the Dependency Injection property. For instance, the forwarding algorithm is bootstrapped based on the defined rules: the proposed social-aware forwarding algorithm is used in combination with the social inference model by default, due to rule number one. The motion inference module is activated due to the rule number two.

ICON provides an open API for customization. The main user interface has two main functionalities: one activate/deactivate sensor readings, configure sensor settings (e.g., broadcast interval), and add virtual sensors (encapsulate sensors from other devices on the network). The other interface functionality is used to configure general settings (e.g., set memory maxsize, select internal database, insert rules).

4.2 Data Engine

The data engine is composed by three modules responsible for in-network data caching, segmentation and protection. The caching module is responsible for managing how and where the node stores data, gathered through networking and sensing interfaces. Currently, the cache keeps all data in memory to improve the performance in what concerns data search and delivery. When a new interest arrives to a node, the first place (unless there is a rule that says otherwise) to look for the matching data is on the local cache. The default cache replacement police

used in ICON is Least Recently Used (LRU): the items that are less requested are the ones that are removed first from the cache. Essentially a cache hit immediately returns a content message from the cache, while a cache miss forces the execution pipeline of the interest message to proceed (c.f. Section 4.4.1).

The data engine is also responsible for segmenting the data that needs to be made available to the Network Engine, as well as for rebuilding fragmented data received in any interface. The segment size is configurable and is inline with the communication protocol to avoid double segmentation.

In what concerns security, applications using the ICON framework have the opportunity to encrypt their data when it is added to messages, as happens in the CCN framework [15]. Such security procedure does not raise problems for the Networking Engine, since the networking operation is completely agnostic of the carried data.

An important ICON property, worth to be mentioned, is that locally stored data does not hold any information related to its origin or destination. This information is actually not needed by the proposed forwarding mechanism.

4.3 Context Engine

The recent trend of incorporating sensors into mobile phones may have an increasing impact in many aspects of our everyday life. Namely, the impact of data sharing models depends on the method used to map users' context (e.g., social affinities) and users' interests on data. Existing data-centric models are not able to exploit users' social affinities, for instance, to augment the performance of data sharing. Second, reception of data does not consider the users' physical context: for instance, it is not useful for a user to receive video data when moving fast, since the content cannot be consumed at once, meaning that in such situation priority may be given to the reception of other sort of data, such as audio.

To mitigate these issues, ICON provides a software suite able to map context information (social similarities and motion patterns) to data tagging provided via the ICON application interface. The goal of combining context awareness and data-centric networking is two fold: i) data interests injected in the network should reflect users' motion context; ii) data forwarding may be done with high probability, low cost and low latency when based on users' social similarities.

In order to augment the device awareness about users' context, the ICON Context Engine includes three modules: context collector; social inference; motion inference.

4.3.1 Context Collector

The context collector is responsible for controlling sensing probes able to extract data about several aspects of users' context. In the current version, the context collector is configured with probes to capture social (relationships and communication patterns) and motion (users' moving patterns) information.

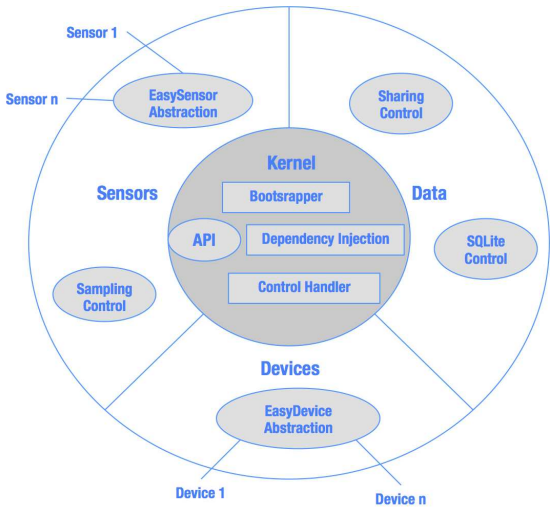

Figure 3: Context collector design

The goal of the context collector is two fold: i) to enable data-centric networking based on a real-time knowledge about users' context; ii) to allow sharing of information about users' context.

The context collector is based on a sensing abstraction called Maestroo [4, 3, 22] that makes use of opportunistic sensing methods [18] to control sampling from any sensor configured in a smartphone. The context collector is able to establish a level of sensing priority big enough to ensure a good sensing performance, and small enough to avoid sluggishness of other applications. Moreover, it introduces the concept of virtual sensors, which mitigates the difficulty in assigning application requests to suitable sensors. Virtual sensing is a novel concept that allows a device to borrow external sensors, presenting them as local sensors to the application. In the context of pervasive networks, the usage of virtual sensors is done by sending query messages to any ICON device in the vicinity (Bluetooth range).

As illustrated in Figure 3, the context collector encompasses three modules: Kernel, Sensors, and Data. This modular approach allows a proper compartmentalization of each functionality, making it easier to maintain and extend.

The kernel module is the one where all the remaining assemblies get referenced. The ControlHandler instantiates and controls sensing operations by: i) configuring sampling intervals and virtual sensors; ii) configuring general device settings (e.g., identifier and type), the memory size and export types; iii) managing the SQLite internal database, and the interval for making sensing data available for sharing.

In terms of data handling, the context collector ensures the persistence required to develop large-scale sensing systems, by providing a set of wrappers for interaction with local databases in a SQLite manner (e.g., Windows phones do

not work natively with SQLite).

In the sensor module, the EasySensor abstraction builds a common interface for multiple sensors available on different devices. The EasySensor abstraction also serves as the basis for the creation of the necessary connectors to both real and virtual sensors. When entering a sensing phase all configured sensors are sampled based on the defined sampling interval, and sensing data is shared based on the configured sharing interval. The sampling interval for each sensor can be statically configured by the user, by an application, or can be configured on-demand by a continuous sensing algorithm [19]. In its current version, the context collector is configured with:

- Wi-Fi sensor: used to collect information related to the ID of neighbor node, as well as time and duration of wireless contacts.
- 3-axes accelerometer sensor: used to collect information related to instantaneous acceleration of the phone.
- GPS sensor: used to collect information related to geoposition (latitude, longitude and range).

In the device module, the EasyDevices abstraction supports different devices (e.g., Windows mobile phones **[3]**) and their specific capabilities in terms of sensors and computational resources. It also enables the creation of different profiles based on sensor and communication interfaces discovered during boot time.

4.3.2 Social Inference

The social inference module allows devices to use wireless context information to infer social affinities. Social affinities are used by the forwarding mechanism (c.f. Section **4.4.1**) to create rules that make data dissemination aware of social ties, aiming to improve the way ICON handles data exchange in pervasive networks.

The social inference module makes use of information collected from the Wi-Fi sensor abstraction: neighbor ID; time and duration of wireless contact. The sampling is done for each contact and information is collected when the contact ends.

The collected information is used to infer about the social strength of a node towards other nodes in relation to a defined set of data interests. Such inference is done in specific time intervals, which defines daily samples. ICON is configured by default to work with daily samples of one hour [27].

Lets us assume that in a daily sample ΔT_i, a node A has n contacts with another nodes having an interest x, where each contact k has a certain duration (*Contact Duration* - $CD\,(a,x)_k$). At the end of ΔT_i, the social inference module starts by computing the *Total Connected Time to Interest* x ($TCTI\,(a,x)_i$) of node A as given by Eq. 1.

$$TCTI\,(a,x)_i = \sum_{k=1}^{n} CD\,(a,x)_k \tag{1}$$

The *Total Connected Time to Interest x* in the same daily sample over consecutive days is used to predict the average duration of contacts towards the data interest x for that specific daily sample. Thus, from the perspective of node A, the *Average Total Connected Time to Interest x* ($ATCTI$) during a daily sample ΔT_i in a day j is given by a cumulative moving average of $TCTI$ in that daily sample ($TCTI(a,x)_{ji}$), and the $ATCTI$ in the same daily sample ΔT_i of the previous day ($ATCTI(a,x)_{(j-1)i}$), as illustrated in Eq. 2.

$$ATCTI\,(a,x)_{ji} = \frac{TCTI\,(a,x)_{ji} + (j-1)ATCTI\,(a,x)_{(j-1)i}}{j} \tag{2}$$

Based on ATCTI, a node A can infer about the *Time-Evolving Contact to Interest x (TECI)* (cf. Eq. 3) to determine its social strength ($w(a,x)_i$) towards nodes tagged with interest x in a daily sample ΔT_i. Such inference is done based on the $ATCTI$ computed in that daily sample and consecutive $t-1$ samples, where t is the total number of samples. The usage of $t-1$ consecutive samples defines a time transitive property [27], represented by the function $\frac{t}{t+\text{k-i}}$ in Eq. 3. The transitive property refers to the probability of two neighbor nodes in a daily sample ΔT_i to continue in contact in consecutive ΔT_{i+x} daily samples.

$$TECI = w\,(a,x)_i = \sum_{k=i}^{i+t-1} \frac{t}{t+k-i} ATCTI\,(a,x)_k \tag{3}$$

4.3.3 Motion Inference

The motion inference module allows the usage of behaviour patterns to control the way data interests are presented to the network, as well as to influence data forwarding decisions. In the former case, motion inference can be used to allow an automatically alignment between reception of data and users' context, leading to an increase in the usefulness that data has to the user. Such alignment is based on data tagging: users map data usefulness to types of motion (e.g., video has highest priority when nodes are stationary, sound when in fast motion, and text when moving slowly).

Motion inference can also be used to influence forwarding decisions, since: a device that is stationary most of the time may not be a good carrier; within devices with similar probability to meet nodes with a certain set of data interests, the ones that present a fast motion profile may be better forwarders; devices with wide movement range may be good elements to spread data.

The initial assumption to infer about users' motion is that the type of human movement is influenced by the place where the user is. For instance, in general people move faster outside (by walking, running or driving) than when moving inside a location. In the latter case, people can be stationary for long periods of time in restaurants, move slowly while within a grocery store, or faster if they are in a shopping mall, for instance. However, it is know that users' type of movement is subject to variations within the same location, depending on people behavior [2]. Therefore, to identify specific behavior, the type of location

is not enough to infer about the type of user's movement. In this case, GPS and accelerometers can be used as a filtering mechanism.

Therefore, the design of the motion inference module is done based on three states of motion: stationary; slow pace; fast moving. If GPS values are valid, identifying an outside location, they are used to estimate the type of user's movement. In this case, GPS readings must come from a local sensor and not from virtual ones, since readings must reflect the user's behavior. On contrary, an inside location is assumed if GPS values are not valid, in which case the outcome of the accelerometer is used.

When in an outside location, GPS sensor reports are used to infer about the user profile in terms of speed and range. From the format of the GPS report the following variables are analyzed: latitude and longitude (in decimal degree); timestamp (seconds); current ground speed (meters per second). Based on these values, the moving profile is estimated based on an exponential moving average of the ground speed, where a value higher than 5 Km/h is used to tag the motion as fast: in the absence of significant external factors, humans tend to walk at about 1.4 m/s (5.0 km/h) [24]. The device range is computed based on the Euclidean distance between the average values of the latitude and longitude: average values are estimated based on an exponential moving average.

When in an inside location, the classification of the three motion states is done based on Support Vector Machines (SVM) [9]. In this case, the training period of SVM can be done first by using readings from a statically phone, and later from readings collected when the user is moving (slow, fast) with the phone [2]. Once the training is accomplished, samples from the 3-axes accelerometer can be fed to the SVM, which then classifies each of these samples to be either in a stationary or moving state. The accelerometer is sampled n times per second, being n defined by configuration. Each sample records the instantaneous acceleration of the phone, and a moving average over a window of 10 recent samples is done to reduce the impact of noise floor, which makes accurate measurements difficult.

To distinguish the three states of users' movement indoors, it is necessary first of all to differentiate between sitting and moving, which can then be further divided into slow pace and fast moving. For this propose, a large number of accelerometer traces can be used to compute the ratio $R = \frac{tmoving}{tstatic}$, where tmoving and tstatic are total durations during which the SVM classified the user as moving or static. Plotting values of R on a real line reveals three groups of accelerometer fingerprints [2]: values of R between 0 and 2 denote a stationary state; between 0.2 and 2 denote slow pace; and higher than 2 denote while fast moving. Based on these thresholds the accelerometer moving average is used to identify the state of the user when moving indoors.

4.4 Networking Engine

The network engine is responsible for supporting data sharing among pervasive devices. Optimization is achieved by a data sharing system that uses data usage patterns and social interaction patterns (provided by the Context Engine) to

decide about the most suitable places in the network to store data in order to allow fast search and retrieval by the end user.

Due to the dynamic nature of pervasive networks, data forwarding is done in a different way than in CCN. In the latter case a consumer asks for content by forwarding interest packets. The Forwarding Information Base (FIB) is used to control forwarding of interest packets toward sources that might match the requested data. It allows multiple sources of data and can query them all in parallel (including broadcast), based on the strategy layer. Any node hearing an interest packet, and having data that satisfies it, can respond with a data packet. When a data packet arrives on an interface, a longest-match look-up is done on its name, based on the Pending Interest Table (PIT). The PIT keeps track of interfaces over which interests packets were received so that returned data can be sent downstream to its(their) requester(s).

In CCN, only interest packets are forwarded and, as they propagate upstream toward potential data sources, they leave a trail of 'bread crumbs' allowing data packets to follow back to the original requester(s). Each PIT entry is a bread crumb. PIT entries are erased as soon as they are used to forward a matching data packet (the data packets 'consumes' the interest). PIT entries for interests that never find a matching data are eventually timed out.

A forwarding scheme based on bread crumbs is not efficient in a pervasive networking scenario, where there is no insurance about the existence of an end-to-end path between any pair of nodes for the duration of a communication section. Moreover, a forwarding scheme in which data packets consume interests may lead to higher latencies in disruptive networks, since devices may not be able to take advantage of sporadic contacts to forward data, if an interest is not present in a neighbor node. Hence, in pervasive systems it is desirable to associate an interest to any packet of a certain data flow (under the same name). This allows the same interest to be used to opportunistically forward any data packet of that flow while the interest is present in a carrier node. In addition, in a pervasive network the forwarding engine can exploit the mobility of nodes to carry data towards interested parties, avoiding the forward of interest packets and the creation of 'bread crumbs'.

Being created to support pervasive networks, ICON encompasses a forwarding mechanism that, as in CCN, is based on data and interest messages, but: does not require FIB; does not use 'bread crumbs'; data packets do not consume interests; data and interests have explicit time to live (TTL). Contrary to CCN, the forwarding mechanism used in ICON is based on the forward of data packets and not of interest packets, which are carried by mobile devices. Any encounter node having data that satisfies carried interests can share data packets with the node carrying such interests, if the latter has higher probability to met nodes with similar interests, than the node that currently carries the data.

Figure 4 illustrates the forwarding engine model used by ICON based on a name syntax following the CCN convention (c.f. Section 5.1 for further information about the naming schemes supported by ICON). The proposed forwarding mechanism makes use of two type of structure: persistent and temporal. Persistent structures are used to store data and interests that are carried by the node:

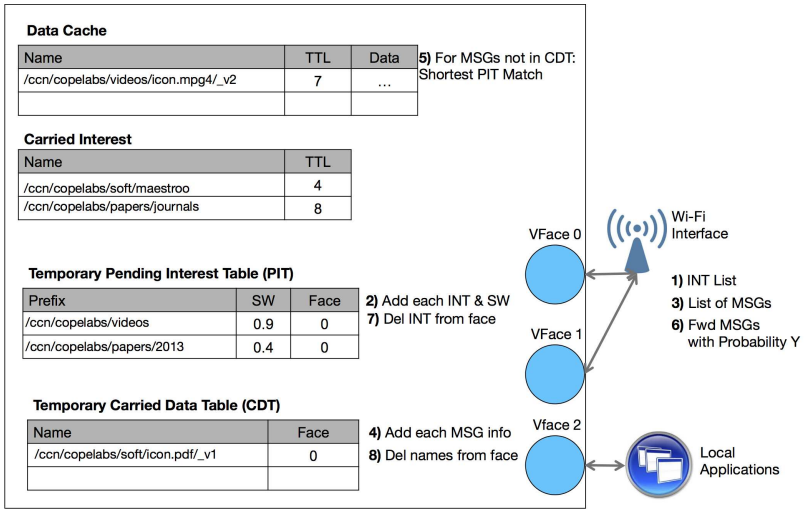

Figure 4: ICON forwarding engine model

data and interest may have been generated by local applications or received via a networking interface. In any of the two persistent caches, an entry has a defined TTL: data TTL defines the time usefulness that the source tagged data with; interest TTL defines the time period over which some node is interested in a specific type of data. By erasing carried data and interests based on TTL, ICON ensures that the pervasive network only transports useful data.

The temporary structures are used to store information that the forwarding mechanism (c.f. Section 4.4.1) collects from neighbor nodes: pending interests and carried data. As for the former, when a $Node_i$ meets a neighbor $Node_j$ it creates temporary PIT entries for the list of all interests (INT in Figure 4) carried by $Node_j$ (c.f. steps 1, 2 and 7 in Figure 4). Each PIT entry encompasses the social weights (SW in Figure 4) towards the nodes having such interests. Additionally $Node_i$ creates temporary CDT (Carried Data Table) entries, each one for a message (MSG in Figure 4) that $Node_j$ already carries (c.f. steps 3, 4 and 8 in Figure 4). These entries are temporary since they are erased as soon as the contact between neighbor nodes is broken.

Data that is stored in the persistent cache is forwarded to a neighbor node with a certain probability (c.f. steps 5 and 6 in Figure 4) if two conditions are met: i) such data is not present in the CDT; ii) it satisfy a shortest-match look-up on PIT. The next section provides a detailed description of the forwarding mechanism.

4.4.1 Forwarding

With ICON, data is shared taking into account users' social affinities, and not the capability of devices. In this sense, ICON incorporates a data-centric for-

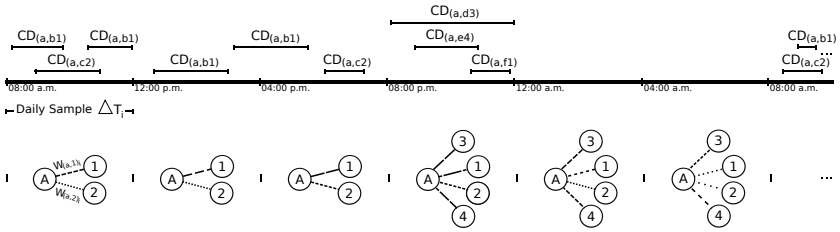

Figure 5: Illustration of daily wireless contacts of a node A

warding algorithm (SCORP) [28][1] that takes into account social weights (provided by the social inference feature of the Context Engine) and content knowledge (received from neighbor nodes) to take forwarding decisions.

SCORP is based on a utility function that reflects the *probability of encountering nodes with a certain interest among the ones that have similar daily social habits*. The reason to use social proximity with content knowledge is two-fold: first, nodes with similar daily habits have higher probability of having similar (content) interests [10]; second, social proximity metrics allow a faster dissemination of data, by taking advantage of more frequent and longer contacts between socially closer nodes.

Fig. 5 illustrates the forwarding assumptions done by ICON, which are related to the mobility and social behavior of a node A, namely its interaction with nodes having data interests x in different daily samples, ΔT_i, throughout its daily routine: Eagle and Pentland [12] show that users have routines that can be used to identify future behavior and interaction with others with whom they share similar behavior.

In the example illustrated in Figure 5, each node encountered by node$A$ only has one type of data interest (nodes B and F have interest on data 1, and nodes C, D and E have interest on data 2, 3, and 4, respectively). Data is carried by node A while the data lifetime, stipulated by the source application, does not expire.

The operation of *SCORP* is based on the information about the social strength indexed to data interests that those nodes have during daily samples (cf. $CD(a, b1)$ in Fig. 5). This way, forwarding can be done based on the different weights (intermittency of lines in Fig. 5) of social interaction that they have with nodes having different data interests ($w(a, 1)$), during specific time periods of their daily activities.

The operation of *SCORP* is very simple: when a $CurrentNode$ meets a $Node_i$ in a daily sample ΔT_k, it gets acquainted with: i) the interests that $Node_i$ has during that daily sample; ii) the social weights that $Node_i$ has towards its neighbours having such interests ($Node_i$.weightsToAllinterests). Additionally, $Node_i$ provides the $CurrentNode$ with a list of data that it carries ($Node_i$.carriedMessages). It is worth to mention that none to the information that $Node_i$ provides $CurrentNode$ with is stored and carried by the latter. Such information is only used by the $CurrentNode$ to make local decision, being discarded afterwards.

[1] Simulation code at http://copelabs.ulusofona.pt/scicommons/index.php/attachments/single/602

After getting all the mentioned information, $CurrentNode$ replicates each $Message_j$ in its buffer to $Node_i$ if:

- $Node_i$ has interest ($Node_i$.getInterests) in the data carried in the message ($Message_j$.getContentType); or
- *The social weight of* $Node_i$ towards a node having that interest (i.e., $Message_j$.getContentType) is higher than the social weight that the $CurrentNode$ has towards any node with the same interest.

With this, *SCORP* is expected to forward messages only to nodes that indeed have interest in the data carried by a message, or that have a strong social relationship with nodes that have that specific interest.

Simulation results (c.f. section 5.2) shown that SCORP has better performance than previous social-aware content-oblivious forwarding proposals, such as *Bubble Rap [13]*, and *dLife [27]*. Moreover, results about cost show that SCORP is able to avoid network flooding, which might happen since an interest packet may potentially match the data stored in a number of different locations.

Although flooding is avoided, SCORP still keeps a suitable level of packet replication to allow data accuracy. This property is importance since a certain percentage of the received data can be inaccurate or even contain false value due to malfunctioning of data sources or highly dynamic environments. Hence, by allowing nodes to get more than one copy of the same data, SCORP is increasing data redundancy in order to support accuracy methods to be implemented by applications.

4.4.2 Message Encoding

Different pervasive applications may use different naming syntaxes. This is illustrated in Section 5.1 by looking at data files, as referred in CCN, and sensing data. The major difference between these two type of data is three fold: size; usefulness; updates. Data files may be very large, meaning that they may need to be segmented, while sensing data normally has small formats; sensing data has a clear usefulness period, which does not happen with data files; sensing data is updated more frequently than data files.

Independently of the enumerated differences, the naming schemes used by ICON follow a hierarchical structure with a variable number of components, as in CCN. However, while CCN uses a binary scheme for message encoding and sequencing, ICON uses a message encoder that has proven to be reliable, extensible, easy to use and freely available: Protocol Buffers (protobuffers).

Protobuffers is a data interchange format that is being used by Google for almost all of its internal RPC protocols and file formats. Moreover, there are several implementations available for different languages and platforms. Any implementation of protobuffers can be used in ICON, as what effectively matters is what is encoded. However, since ICON targets the .Net platform, the .NET serialized is currently used, since it is already based on protobuffers, and this way

consistency with the remaining .NET serializers is ensured and extra complexity avoided.

Currently, the protobuffer encoder is used to codify the following two hierarchical name syntaxes:

- File transfer (following the CCN proposal):
  $/ccn/comp_1/comp_n/\_v < timestamp > /\_sn$
- Sensing (general syntax for sensing data):
  $/sensing/geolocation/timestamp/tag_1/tag_n$

More information about the naming schemes used by ICON can be found in Section 5.1. The inclusion of other naming schemes, to support new applications, requires the update of the message encoder, which can be done without rebooting the ICON device, reducing the disruption to running applications (c.f., Section 4).

As mentioned before, name syntax and forwarding are uncorrelated in what concerns the used context: forwarding uses social proximity and data-interest similarity, while name syntax may include application specific context, such as geolocation (in the case of sensing applications). This uncorrelation allows the proposed message encoding to be used by other data-centric architectures.

5 Applicability and Feasibility Study

In a data-centric network, data transmission starts by having a consumer sending an interest packet that includes a name to identify the desired data. However, different applications have different requirements about the naming syntax: for instance, file transfer applications need to differentiate different versions of the same data, as well as different segments, since data files may be large; sensing data may need to be differentiated by they location, and time usefulness.

Retrieving data by name is suitable for an environment of networked pervasive systems, when compared to retrieving data from a specific location, as happens with the Internet Protocol. However, the disruptive properties of pervasive networks (e.g., devices on/off; absence of Internet access; wireless shadowing; costs of mobile communications) require a data exchange system able to take advantage of any communication opportunity, even if that means replicating data among wireless peers that are met in a probabilistic way. This type of opportunistic forwarding should be considered as a basic property for the deployment of networked pervasive systems, in order to avoid disruption on communications.

This section provides a study about the support that ICON can give to different applications, and about its forwarding efficiency in pervasive networking scenarios in terms of delivery probability, cost and latency.

Since the goal is to have ICON running independently of the hardware and operating system, ICON is programmed in C# over the .NET framework. Currently ICON can be seamlessly executed in Windows mobile, Android (with

MonoDroid), and iPhones (with MonoTouch). With the .NET Micro framework, ICON can also run in embedded devices. The C# language (ISO/IEC 23270:2006) is a multi-paradigm programming language encompassing strong imperative, as well as declarative and component-oriented programming disciplines. This aspect allows computational logic to be expressed without describing its control flow, which this is a major requirement to develop pervasive computational systems.

5.1 Applicability

In order to keep application compatibility with CCN applications, ICON supports the same hierarchical name structure for file transfer. As proposed by Van Jacobson et al. [15] names are hierarchically structured encompassing a number of components, which may be encrypted for privacy. For notation proposes, names are presented as URIs with / characters separating components.

For file transfer, the naming syntax used by ICON follows the structure $/ccn/comp_1/comp_n/\_v < timestamp > /\_sn$. This illustrates a CCN compatible syntax with the identification of the type of application, n string components (names) of variable length, and two fields used to capture the temporal evolution of the content: a version marker, _v, followed by an integer version number; a segment marker, _s, followed by an integer value that may be the frame number of a video file, for instance. The final component of every packet name implicitly includes a SHA256 digest of the packet.

Although the CCN syntax is included in ICON to support data transfer services, it is clear that different pervasive type of applications may need to refer to data in different ways and different granularities. Therefore, the basic version of ICON includes also a name syntax to refer to data used by sensing applications, which are common in pervasive systems, such as: applications related to social sensing and applications related to vehicle-to-vehicle communication. The motivation to support sensing data is two fold: sensing services are becomes common in mobile applications; the data syntax is very different from the data transfer example.

The naming design proposed to describe sensing data also follows an hierarchical syntax structure, to avoid the inclusion of more than one sequencing mechanism in the networking module. Nevertheless, one major difference towards the name used by file transfer applications, is that the syntax used by sensing applications include context information (captured by the context collector module): the inclusion of geolocation information is needed since the usefulness of sensing data has a space constraint.

The syntax $/sensing/geolocation/timestamp/tag_1/tag_n$ is used for sensing data, where**:**

- Sensing: is a string field identifying the type of application (two examples are provided in this section). Different pervasive applications may use different instantiations of the naming syntax.

- Geolocation: is a numeric field composed of three integers in the format $< lat, long, range >$, allowing data to be mapped to a specific geographical area. The location component $< lat, long >$ represents the geographic center of the reported area. This component is implemented based on information collected from a local GPS sensor. The range component (in meters) represents the ratio over which the information is meaningful. This component is implemented based on the standard Wi-Fi range of the device: a typical device using 802.11b or 802.11g might have a range of 35 m indoors and 100 m outdoors, while a device with 802.11n can double that range. The inference about the location (indoor and outdoor) is provided by the availability or not of GPS signal.

- Timestamp: is a numeric field composed of two integers in the format of $< start, end >$, allowing data to be mapped to a specified time period. This is needed, since sensing data is normally useful only for a certain period of time. Each of the two integers follow a UNIX type of format (e.g., 1387457545, meaning Thursday, December 19th 2013, 13:10:05). Two configurations are possible: i) single timestamp (with null end) to collect data about a specific moment; ii) double timestamp, to collect data within a specific time period.

- Tagging: The tag components indicate the meaning of the sensing data itself. Each one of the n integers represent the code of a tag that is meaningful to a specific application. The order of the tags must respect the requirements of each application.

If a source of sensing data has no GPS, a virtual sensor can be configured to make locally available information collected from other GPS-enabled device in the vicinity (c.f. Section 4.3.1). If no GPS-enabled device is in range, the user is prompt to enter a post address.

As an applicability example, the instantiation of the sensing name syntax for social sensing and traffic applications is provided. The former type of applications require social context related to the users' physical and social settings (e.g., VibN [23]). It should include information able to allow inference about people mood in relation to their surroundings and with whom they interact. The instantiation of the sensing name syntax to support social sensing applications, such as a social dashboard, making use of three type of tags, may be */socialdash/geolocation/timestamp/audiotag/videotag/physicaltag*, where:

- Audio tags: reference to short audio clip with a description of the place and the mood of people. May be collected automatically or by demand of the user.

- Visual tags: reference to photo or short video with a brief illustration of the place. Are collected by demand of the user.

- Physical tags: reference to information aiming to allow inference about the social vibe (e.g., running, walking).

Double timestamps are useful for social sensing services, since the mentioned data (audio and video tags, as well as physical activity) may be requested for a specific time frame. For instance to socially characterize places by recent activity or to check the history of a place.

Applications related to vehicular networks can also use the hierarchical sensing name syntax, since vehicle related data is normally useful in a certain geographic area and for a certain time period. The instantiation of the sensing name syntax for the dissemination of traffic information may be as defined by L. Wang et al. [36], based on two tags (datatype and nonce): $/traffic/geolocation/timestamp/datatype/nonce$.

In the case of the sensing syntax used by L. Wang et al., the three tuple identifying the geolocation component does not follow the $< lat, long, range >$ format, but the format $< ID, direction, section >$, where, ID represents the road name; direction represents the traffic direction; and section can be the exit numbers of highways. The data type component indicates the meaning of the data itself, such as closed lane and vehicle speed. The last tag is a nonce, which is a large random number used to distinguish data generated by different producers.

A similar traffic information system based on ICON would not need the tailing component "nonce", since there is no need to explicitly distinguish between data sources generating data with identical values: the forwarding mechanism used by ICON (c.f. Section 4.4.1) is able to avoid the network to be flooded with similar data generated by different devices.

5.2 Feasibility

This section evaluates the ICON capability to share sensing data in a pervasive networking scenario. This feasibility analysis allows the evaluation of five components of the ICON framework: the social-aware forwarding algorithm and message encoding of the Network Engine; the social inference of the Context Engine; and the in-network caching and data segmentation of of the Data Engine.

SCORP is evaluated against *dLife* [27], a social-aware opportunistic forwarding mechanism based on users' daily life routines; *Bubble Rap* [13], a community-aware proposal; and *Spray and Wait* [34], a social-oblivious solution that serves as lower bound reference in what concerns delivery cost.

The full evaluation [28] is done in the Opportunistic Network Environment (ONE) simulator with a 95% confidence interval. Evaluation is done in terms of averaged delivery probability (i.e., ratio between the number of delivered messages and the number of messages that should have been delivered), cost (i.e., number of replicas per delivered message), and latency (i.e., time elapsed between message creation and delivery).

The used scenario is based on human traces (Cambridge traces) and encompasses varying network load, represented by the number of messages/interests (msg/int) per node. Message size ranges from 1 to 100 kB and nodes have only a 2 MB buffer space: we assume that users may not be willing to share all the storage capacity of their devices. All results are taken for a 1-day message TTL:

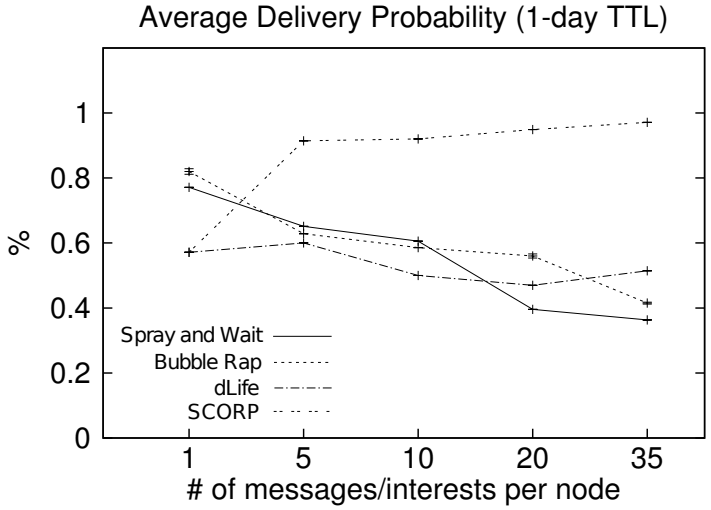

Figure 6: Average delivery probability

the complete performance study [28] shows that such value allows the analyzed forwarding proposals to deliver the most messages in less time and with the least associated cost.

The used human traces including 36 nodes, for two months while Cambridge University students moved throughout their daily routines. As general remark regarding this dataset, the measurements that we did to prepare the configuration of the experiments show that it has an average of 32 contacts per hour among nodes and such contacts happen sporadically. Additionally, the average number of formed community (with *Bubble Rap*) is approx. 6.7, where most of them comprise almost all nodes.

In what concerns traffic load, with the *Spray and Wait, Bubble Rap* and *dLife* proposals, each source creates and sends 1, 5, 10, 20 and 35 different messages towards each of the 35 destinations. In the case of *SCORP,* the source creates 35 messages with different interests once, and each receiver is configured with 1, 5, 10, 20, and 35 different interests. Since node 0 is the source of these messages to the remaining 35 nodes, it means that a total of 35, 175, 350, 700, and 1225 messages reach the destinations in any of the simulations. The goal is to use a generic traffic load setting in order to create a fair evaluation environment among such a different set of forwarding mechanisms: the configurations of messages and interests are done to guarantee the same amount of potential messages being delivered by any proposal.

As for the proposals, *Spray and Wait* runs in binary mode with number of copies L set to 10. *Bubble Rap* uses algorithms for community formation and node centrality computation (K-Clique and cumulative window) [13]. *dLife* and *SCORP* consider 24 daily samples of one hour as mentioned in *dLife*'s paper [27].

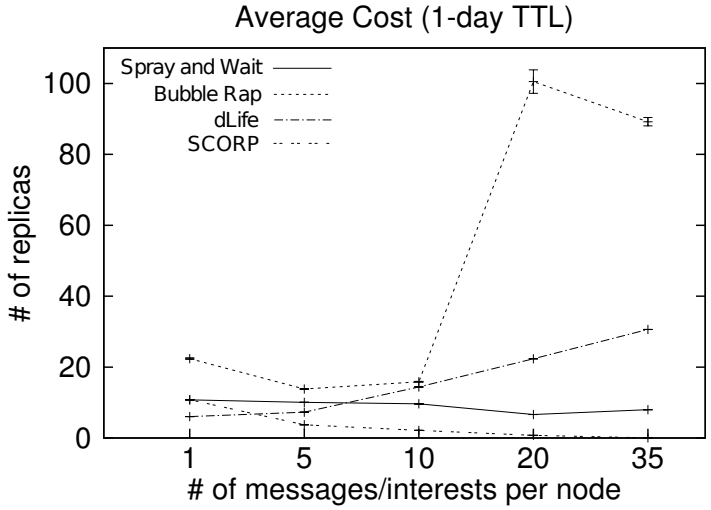

Figure 7: Average cost

Fig. 6 presents the results about the average delivery probability. The msg/int notation denotes the number of different messages sent by *Spray and Wait, Bubble Rap* and *dLife* sources or the number of different interests of each of the *SCORP* receivers.

The forwarding performance of ICON shows the advantage of using content-awareness in pervasive networks: the delivery ratio increases as the ability of nodes to become good message carriers increases: the more interests a node has, the better it is to deliver sensing data to others since they potentially share interests.

Regarding average cost (cf. Fig. 7), it is observable that with a greater list of data interests, an ICON node can act as carrier for a larger number of nodes, which means that unwanted replicas observed in the one msg/int configuration have a positive effect while spreading content. Moreover, the cost is reduced since messages are only replicated to interested nodes or to nodes that have a stronger social weight towards other nodes with higher interest in the data than the current carrier. Moreover, SCORP keeps resource usage (i.e., buffer) at a low usage rate: with content awareness, the estimated maximum buffer occupancy varies between ˜0.03 MB (1 msg/int) and 0.15 MB (35 msg/int).

Fig. 8 shows the average latency that messages experience. By looking at the delivered messages, we observe that *dLife* and *SCORP* perform mostly (90%) direct deliveries as the source node meets destinations within the first two hours of simulation. This surely reduces the overall latency, explaining why they take the same time to perform a delivery.

SCORP experiences up to 93.61%, 90.25% and 89.94% less latency than *Spray and Wait*, *Bubble Rap* and *dLife*, respectively. Based on *SCORP*, an ICON node can receive more data, since it is interested in the data being replicated,

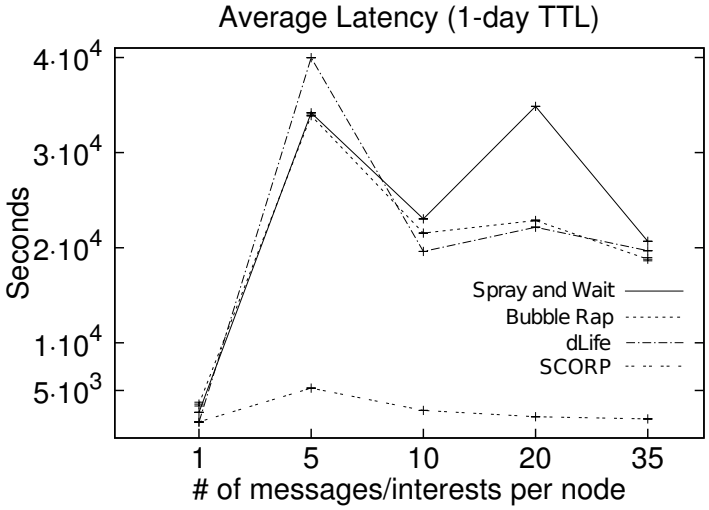

Figure 8: Average latency

and becomes a better forwarder as the probability of meeting nodes sharing the same interest is high.

As conclusion, our findings show that an efficient dissemination of data over pervasive networks is ensured by ICON, since forwarding is designed having content knowledge and social proximity in mind. Its forwarding module, *SCORP,* has better performance than previous social-aware content-oblivious forwarding proposals (e.g,. *Bubble Rap* and *dLife*): *SCORP* delivers up to 97% of its content in an average of 46.9 minutes, against the 335.5 and 343.7 minutes needed by *Bubble Rap* and *dLife*, respectively. Additionally, *SCORP* produces up to approximately 13.9 and 4.7 times less replicas than *Bubble Rap* and *dLife*, respectively.

6 Conclusion

This paper describes a data-centric networking framework for pervasive networks, called ICON. The major motivation for this work is the lack of a data-centric networking solution to support communication in challenge networking scenarios. A suitable data-centric framework for pervasive networks needs to fulfill the requirements related with the exchange of data over dynamic networks (high delivery probability, low cost and latency), and related with software design (fast deployment).

The ICON framework is designed having in mind all the specific requirements of a dynamic network of pervasive devices, namely two: i) the need to have a flexible, extendable, and easy to port software design, in order to motivate people to install and use ICON; ii) the capability to forward data by exploiting any wireless contact opportunity, ensuring a good networking experience.

Simulations done based on real traces show that the social-aware forwarding mechanism proposed for ICON has higher delivery probability, and lower cost and latency than other social-aware and social-oblivious forwarding schemes.

As next steps, the current design of ICON will be improved by further analyzing the correlation between TTLs (for data and interests) and motion inference: when new interests are injected in the network after detecting a change in the user's motion pattern, the TTL associated with those interests should reflect the time over which the user will stay on that motion state, since the importance of data is linked with that situation. Another improvement is the development of a ICON/CCN gateway module, to be included in the networking engine. This gateway is needed since ICON and CCN use different message encoders: protobuffers in the case of ICON and binary encoding in the case of CCN.

In what concerns caching, the current cache replacement policy (Least Recently Used) will be replaced by a system that allows cache entries to be removed based on the inference about their importance to neighbor devices. For instance a data segment that complements data stored in neighbor devices should be removed with low probability.

In what concerns the evaluation of ICON, its performance will be tested in a real testbed. In order to make the evaluation scenario scalable, real equipment will be combined with mobility and traffic models simulated in MatLab, and a large number of ICON nodes will be used based on virtual machines.

Acknowledgement

Acknowledgement go to the CitySense project of COPELABS, for all the support provided during the development of the ICON framework.

References

[1] C. Anastasiades, T. Braun, and V.A. Siris. Information-centric networking in mobile and opportunistic networks. *Springer LNCS on Wireless Networking for Moving Objects: Models, Approaches, Techniques, Protocols, Architectures, Tools, Applications and Services*.

[2] M. Azizyan, I. Constandache, and R. Choudhury. Surroundsense: Mobile phone localization via ambience fingerprinting. In *Proc. of ACM Mobicom*, 2009.

[3] R. Barbosa. Maestroo - an immersive sensing tool. *SITI-SW-12-05 software package*, 2012.

[4] Ricardo Barbosa. Sensing middleware: Collect and share. *University Lusofona Master Thesis*, December 2012.

[5] Trisha Biswasy, Asit Chakrabortiz, Ravishankar Ravindranz, Xinwen Zhangz, and Guoqiang Wang. Contextualized information-centric home network. *ACM Computer Communication Review*, 43(4), October 2013.

[6] Chiara Boldrini, Marco Conti, and Andrea Passarella. Design and performance evaluation of contentplace, a social-aware data dissemination system for opportunistic networks. *Journal Computer Networks: The International Journal of Computer and Telecommunications Networking*, 54(4), March 2010.

[7] Eduardo Cerqueira, Paulo Mendes, and Edmundo Monteiro. Multi-user Session Control in Next Generation Wireless Systems. In *Proc. of ACM Workshop on Mobility Management and Wireless Access (MobiWac)*, Malaga, Spain, October 2006.

[8] Eduardo Cerqueira, Luis Veloso, Marília Curado, Edmundo Monteiro, and Paulo Mendes. QoS Mapping and Adaptation Control for Multi-user Sessions over Heterogeneous Wireless Networks. In *Proc. of ACM Mobimedia*, Nafpaktos, Greece, 2007.

[9] C. Cortes and V. Vapnik. *Support-vector networks*. Kluwer Academic Publishers, 1995.

[10] P. Costa, C. Mascolo, M. Musolesi, and G. P. Picco. Socially-aware routing for publish-subscribe in delay-tolerant mobile ad hoc networks. *IEEE Journal of Selection Areas in Communication*, 26(5), June 2008.

[11] C. Dannewitz, J. Golic, B. Ohlman, and B. Ahlgren. Secure naming for a network of information. In *Proc. of INFOCOM*, March 2010.

[12] Nathan Eagle and Alex Pentland. Eigenbehaviors: identifying structure in routine. *Behavioral Ecology and Sociobiology*, 63(7):1057–1066, May, 2009.

[13] Pan Hui, Jon Crowcroft, and Eiko Yoneki. Bubble rap: Social-based forwarding in delay-tolerant networks. *IEEE Transactions on Mobile Computing*, 10(11), November 2011.

[14] Van Jacobson, Diana Smetters, Nicholas Briggs, Michael Plass, Paul Stewart, James Thornton, and Rebecca Braynard. Voccn: voice-over content-centric networks. In *Proc. of ACM Rearch workshop*, December 2009.

[15] Van Jacobson, Diana Smetters, James Thornton, Michael Plass, Nicholas Briggs, and Rebecca Braynard. Networking named content. In *Proc. of ACM CoNext*, December 2009.

[16] Petri Jokela, Andrea Zahemszky, Christian Rothenberg, Somaya Arianfar, and Pekka Nikander. Lipsin: line speed publish/subscribe inter-networking. In *Proc. of ACM SIGCOMM*, August 2009.

[17] Teemu Koponen, Mohit Chawla, Byung-Gon Chun, Andrey Ermolinskiy, Kye Hyun Kim, Scott Shenker, and Ion Stoic. A data-oriented (and beyond) network architecture. *ACM CCR*, 37(4):181–192, October 2007.

[18] Nicholas D. Lane, Shane B. Eisenman, Mirco Musolesi, Emiliano Miluzzo, and Andrew T. Campbell. Urban sensing systems: Opportunistic or participatory? In *Proc. of HotMobile*, Napa Valley, USA, February 2008.

[19] Hong Lu, Jun Yang, Zhigang Liu, Nicholas D. Lang, Tanzeem Choudhury, and Andrew T. Campbell. The jigsaw continuous sensing engine for mobile phone applications. In *Proc. of ACM SenSys*, Zurich, Switzerland, November 2010.

[20] Michael Meisel, Vasileios Pappas, and Lixia Zhang. Ad hoc networking via named data. In *Proc. of ACM MobiArch*, 2010.

[21] Michael Meisel, Vasileios Pappas, and Lixia Zhang. Listen first, broadcast later: Topology-agnostic forwarding under high dynamics. In *Annual Conference of International Technology Alliance in Network and Information Science*, September 2010.

[22] P. Mendes. Cooperative mobile sensing framework. Technical report sititr-12-20, SITILabs/University Lusofona, December 2012.

[23] Emiliano Miluzzo, Michela Papandrea, Nicholas D. Lane, Andy M. Sarroff, Silvia Giordano, and Andrew T. Campbell. Tapping into the vibe of the city using vibn. In *Proc. of Symposium on Social and Community Intelligence*, 2011.

[24] B. Mohler, W. Thompson, S. Creem-Regehr, H. Pick, and W. Warren. Visual flow influences gait transition speed and preferred walking speed. *Experimental Brain Research*, 181(2):221–228, 2007.

[25] Waldir Moreira, Manuel de Souza, Paulo Mendes, and Susana Sargento. Study on the Effect of Network Dynamics on Opportunistic Routing. In *Proc. of AdhocNow*, Belgrad, Servia, July 2012.

[26] Waldir Moreira and Paulo Mendes. *Social-aware Opportunistic Routing: The new trend*. Springer, August 2013.

[27] Waldir Moreira, Paulo Mendes, and Susana Sargento. Opportunistic Routing based on daily routines. In *Proc. of IEEE WoWMoM AOC*, San Francisco, USA, June 2012.

[28] Waldir Moreira, Paulo Mendes, and Susana Sargento. Social-aware Opportunistic Routing Protocol based on User's Interactions and Interests. In *Proc. of AdhocNets*, Barcelona, Spain, October 2013.

[29] Hoang Anh Nguyen and Silvia Giordano. Context information prediction for social-based routing in opportunistic networks. *Journal Ad Hoc Networks*, 10(8), November 2012.

[30] Soon Oh, Davide Lau, and Mario Gerla. Content centric networking in tactical and emergency manets. In *Proc of Wireless Days*, October 2010.

[31] I. Psaras, L. Saino, M. Arumaithurai, K.K. Ramakrishnan, and G. Pavlou. Name-based replication priorities in disaster cases. In *Proc of IEEE workshop on Name Oriented Mobility*, April 2014.

[32] Rute Sofia and Paulo Mendes. User-provided networks: Consumer as provider. *IEEE Communication Magazine, Feature Topic on Consumer Communications and Networking - Gaming and Entertainment*, 46(12):86–91, 2008.

[33] Rute Sofia, Paulo Mendes, Manuel Jose Damasio, Sara Henriques, Fabio Giglietto, Erica Giambitto, and Alessandro Bogliolo. Moving towards a socially-driven internet architectural design. *ACM CCR*, 42(3), 2012.

[34] Thrasyvoulos Spyropoulos, Konstantinos Psounis, and Cauligi S. Raghavendra. Spray and wait: an efficient routing scheme for intermittently connected mobile networks. In *Proc. of ACM SIGCOMM workshop on Delay-tolerant networking*, August 2005.

[35] Gareth Tyson, John Bigham, and Eliane Bodanese. Towards an Information-Centric Delay-Tolerant Network. In *Proc. of IEEE INFOCOM Workshop on Emerging Design Choices in Name-Oriented Networking (NOMEN)*, Turin, Italy, May 2013.

[36] Lucas Wang, Ryuji Wakikawa, Romain Kuntz, Rama Vuyyuru, and Lixia Zhang. Data naming in vehicle-to-vehicle communications. In *Proc. of IEEE INFOCOM NOMEN*, March 2012.

[37] Qing Wei, Paulo Mendes, Christian Prehofer, Nima Nafisi, Karoly Farkas, and Bernhard Plattner. Context-aware handover based on active network technology. In *Proc. of IFIP IWAN - Selected to Elsevier Computer Communications Magazine, October 2006*, Kyoto, Japan, December 2003.

[38] Yu-Ting Yu, Chris Tandiono, Xiao Li, You Lu, M. Y. Sanadidi, and Mario Gerla. Ican: Information-centric context-aware ad-hoc network. In *Proc. of ICNC*, February 2014.

Bibliographies

Paulo Mendes is vice-director of the Cognition and People Centric Computing Research Center (COPELABS) at University Lusofona, where he is associated professor, and director of the Ph.D program New Media and Pervasive Systems (NEMPS). His research interests are in the area of self-organizing networked systems, pervasive sensing, network structure and dynamics and data-centric networking. He graduated (93) in Informatics Engineering from the University

of Coimbra, M.Sc. (98) in Electrical and Computer Engineering from the Technical University of Lisbon, and Ph.D. (03) in Informatics Engineering from the University of Coimbra, having performed his Ph.D. thesis work as a visiting scholar at Columbia University. After his Ph.D. he was a senior researcher at NTT DOCOMO Euro-labs, and a research coordinator at the INESCTEC. In 2010 we founded the research center on Informatics and Technology Systems (SITILabs) at University Lusofona, which gave place to COPELABS in October 2013. His publication record includes over 70 scientific articles and 13 international patents.