BIONETS

WP 5.4 – PROTOTYPING AND VALIDATION

D5.4 Pervasive Ubiquitous Peer-to-Peer context-aware Application

Reference:	BIONETS/CN/wp5.4/1
Category:	Deliverable
Editor:	Iacopo Carreras (CN)
Authors:	Iacopo Carreras (CN)
Verification:	Francesco De Pellegrini (CN) and David Linner (TUB)
Date:	August 1, 2008
Status:	
Availability:	Public

Executive Summary

The design and implementation of a Pervasive Ubiquitous P2P context-aware application corresponds to task $T5.3.4$ of Worckpackage 5 (Prototyping and Validation) of the BIONETS project. $T5.3.4$ is concerned with the development of a middleware specifically tailored to the diffusion of user-centric information, such as contextual and entertainment data, in opportunistic communication environments. The main objectives of this task can be summarized as follows. First, we want to implement and experiment a subset of the BIONETS networking framework components. Now that the project is in a more mature stage, this prototyping activity will provide a valuable feedback to the many algorithms and techniques developed and evaluated at the networking level.
Second, we want to investigate which are the constraints coming from real-world applications. BIONETS-like systems are expected to support pervasive application scenarios, where users access services and information through their mobile handset while moving around during their daily activities. Implementing and experimenting realistic (although simplified) future pervasive application scenarios will allow us to better understand which are the additional constraints that need to be taken into account when designing this type of systems.
Third, we want to evaluate how far we are, in terms of state-of-the art technologies, from BIONETS-like application scenarios.

Contents

DOCUMENT HISTORY

Version History

Version	Status	Date	Author(s)
First			

Summary of Changes

Version	Section(s)	Synopsis of Change

1 Introduction

Opportunistic networking [1] refers to the possibility of delivering data applying an epidemic-like forwarding mechanism, without the need for any dedicated infrastructure. Such communication paradigm received great attention in the last few years as an emerging technology for disseminating data in challenged environments, where due to environmental constraints it is not possible to build an alternative communication infrastructure, or in pervasive environments, where data exchanges are driven by the "social interactions" of mobile users [2]. In particular, the latter case is a direct consequence of the fact that mobile devices (e.g., smartphones or PDAs) are nowadays largely available among people and of the constantly increasing computing, communication and storage power of such devices. Several mobile phones are in fact equipped with Bluetooth, Wi-Fi and Wibree (in the near future), technologies that are directly accessible for programmers through freely available and easy-to-use APIs. Further, mobile phones are now capable of intensive processing operations and of storing large amounts of data in their internal memory.

Starting from this considerations, the BIONETS project developed the "disappearing networking" concept, which is a networking framework addressing the problems of scale (in terms of number of devices) and heterogeneity (in terms of different features supported by the different nodes). In particular, it provides a novel network architecture, based on localized exchanges of information upon opportunistic contacts. A formal specification of its networking architecture can be found in [3].

During the first phase of the project, the concept of opportunistic networking has been deeply investigated from a theoretical point of view, analyzing the existing trade offs among the various system components, and evaluating the performance of different algorithms. This initial study provided the BIONETS consortium with a good theoretical understanding of how the various system parameters impact the ultimate networking performance of the system, and how the system design space is shaped.

However, now that the project is in a more mature stage, there is the need to start evaluating, in a real setting, the proposed system networking framework architecture, together with its algorithms and functional components. Further, when developing the fundamental concepts of the disappearing networking infrastructure, we concentrated mostly on performance aspects such as e.g., end-to-end data diffusion or redundancy, that, although extremely important do not focus on constraints possibly coming from real-world application scenarios or utilized technologies.

Following these considerations, we have developed a *User-centric Heterogeneous Opportunistic Middleware* (U-Hopper), which enables the transparent diffusion of data in BIONETS-like pervasive environments, and implements a subset of the networking framework features. The middleware is running on any java-enabled smartphone and leverages Bluetooth connectivity for exchanging data. Such platform combines the user preferences with the requirements imposed by the pervasive services hosted on users' portable device in order to gather and disseminate data. In line with the BIONETS vision, such process is fully distributed and self-organized, as it does not require any human supervision.
This deliverable provides a brief description of the U-Hopper platform, of its functional blocks and of how the various components implement the BIONETS networking framework functionalities. The reminder of this deliverable organized as follows. In Sec. 2 we will present the middleware system architecture, starting from the constraints that we considered in the system design, and providing then some implementation details. Sec. 3 provides an example of how the developed middleware can be utilized in a real setting, and how it has been demonstrated during the demo sessions of two conferences. In Sec. 4 we will draw the main conclusions of this prototyping activities, pointing out the "Lessons learned". Finally, Sec. 5 concludes this work discussing the main achievements.

2 Pervasive Ubiquitous Peer-to-Peer context-aware Application

The BIONETS system architecture consists of two classes of nodes: *User-Nodes* (U-Nodes), which are resource-rich mobile devices (e.g., smartphones, PDAs) carried around by users during their daily activities, and *Tiny-Nodes* (T-Nodes), which are resource-constrained devices embedded in the environment and providing localized information [3]. A part from their technological differences, the 2 classes of devices play a different role in the network. T-Nodes act as *providers of information*, constantly broadcasting localized information such as advertisements, or snapshots of the surrounding environment. Conversely, U-Nodes act as *consumers of information*, reading T-Nodes in their communication range, and augmenting pervasive services with such data. The data generated by T-Nodes is first stored in the U-Nodes internal memory, and then diffused by means of opportunistic peer-to-peer data exchanges. Users' mobility is therefore exploited in order to achieve system-wide communications.

U-Hopper is a User-centric Heterogeneous Opportunistic Middleware [4] residing on U-Nodes and exploiting any proximity communication interface (i.e., Bluetooth, Wi-Fi) in order to (i) gather localized information originating from T-Nodes embedded in the environment (ii) opportunistically disseminate the stored data to other U-Nodes. The information diffused includes data received from T-Nodes as well as any other information shared by the user (e.g., music, videos, etc.).

2.1 BIONETS Networking framework requirements

Tab.1 provides a short summary of the main features that any BIONETS-like computing and communication environment must present (please refer to [3] for a more detailed description of such requirements). As a first step, we have analyzed such features and mapped them into specific requirements that the u-hopper middleware must fulfill. These requirements represented the basic guidelines for the entire system design.

2.2 System Design

U-Hopper is implemented as a middleware, which means that is supposed to provide the necessary programming abstractions for facilitating the implementation and deployment of services based on opportunistic communications.

The u-hopper middleware, in terms of functional blocks, is depicted in Fig. 1, and is composed by 6 distinct modules. The first three are necessary to access the u-hopper middleware, and to configure it depending on the specific service/usage. More in detail:

- the *User Interface Manager* (UIM) handles any human-to-machine interaction such as data insertion and visualization. Through this component users can access any service running on top of the u-hoppe middleware, and process any data opportunistically retrieved while moving;
- the *Service Manager* (SM) is the execution environment where pervasive services are running. This component allows services deployment, deprecation and update. In a system implementing the entire BIONETS architecture, the SM would be replaced by the BIONETS service framework;
- the *Profile Manager* (PM) stores any personal information related to the user (e.g., name, surname, affiliation, etc.). This component, while not being relevant from a functional point of view, results relevant for an application perspective, since most of the envisioned BIONETS services will be "user-situated". This means that very often this information will be necessary to personalize services, and to transparently provide user preferences to the underlining middleware.

Differently, the u-hopper core implements a subset of the networking framework functionalities and is in charge of the opportunistic diffusion of information. Referring to the BIOENETS networking framework architecture [3], the main functional building blocks are the *Basic Communication Unit*, which deals with data communications, and the *Data Processing Unit*, which handles all the necessary data processing tasks (*Information Filtering*, *Naming System*, *Data Dissemination*. See [5] for more details.) that are necessary to ensure some of the main features that BIONETS systems must present. More in detail, the u-hopper middleware implements the Data Processing Unit through the following modules:

- the *Interest Manager* takes into account (i) the user preferences and (ii) the requirements deriving from the pervasive services hosted by the SM, and produces a list of "interests", which are a high-level

Table 1: Summary of the BIONETS features that will be prominently observed in any BIONETS-like environment, and need to be supported by the envisioned "Disappearing Network" architecture. Each feature is then mapped to a specific u-hopper requirement.

Feature	Description	System Requirements
Device Heterogeneity	The BIONETS system is characterized by a large heterogeneity in the kind of devices taking part in the system, presenting very different computational and communication capabilities.	U-Hopper is expected to run on a wide range of heterogeneous devices, characterized by different, e.g., operating system, communication technologies, etc.
Scalability	T-nodes and U-Nodes are expected to be in a very large number. This requires the use of local interactions only for communications and of information filtering mechanisms for tackling the scalability problem.	U-Hopper is expected to use localized algorithms only, and to scale to a (potentially) unlimited number of nodes.
Store-and-forward type of data dissemination	U-nodes are mobile and can carry around data they have read from T-nodes or received from other U-nodes to be utilized elsewhere or spread further. This paradigm, generally referred as store–carry–and–forward, will represent the basic data diffusion mechanism of the disappearing network infrastructure.	U-Hopper will implement a store–carry–and–forward mechanism, with the possibility of utilizing and evaluating different forwarding mechanisms.
Location Specific information distribution	Information disseminated by T-Nodes has only relevance to nodes near the source in terms of time and space domain.	Any content handled through the u-hopper system must be tagged with its spatio/temporal coordinates.
Nodes self-organization	Nodes must have the ability to automatically organize themselves in a way which ensures that a node will react sanely to all kinds of inputs coming from both the users of nodes and from the network.	U-Hopper is expected to run without the need for any direct human supervision.

description of the information the user is interested in. Such interests regulate the way according to which information is exchanged among U-Nodes. With respect to the Serworks architecture, this component implements the *Information Filtering* and *Data Dissemination* functionalities. The IM runs appropriate data aging algorithms that are needed in order to discard outdated information and preserve the available resources. Such techniques implement information filtering rules that trade off data locality (both in the time and space domains) for available resources (i.e., storage, communication, etc.) [6]. As an example, we can think at a special sale offer ending at 5 pm of the current day. Clearly, as soon as the offer is no longer valid, it is useless to store the corresponding information. The IM is in charge of detecting such situations and of determining when to remove data from users' device permanent storage;

- the *Content Manager* (CM) manages the persistent storage available on mobile devices. In particular, it provides context data insertion/deletion, update and search functionalities. It implements the Serworks *Naming System* functionality [5]. In fact, u-hopper is designed around a fully data-centric architecture, where information is not delivered from a source to a destination, but simply diffused according to its content and relevance to the users. The CM implements most of the complexity related to the data-centric nature of the middleware.

Finally, the *Basic Communication Unit* monitors the availability of data sources in the surrounding environment, and seamlessly performs any networking operation needed for gathering the discovered data.

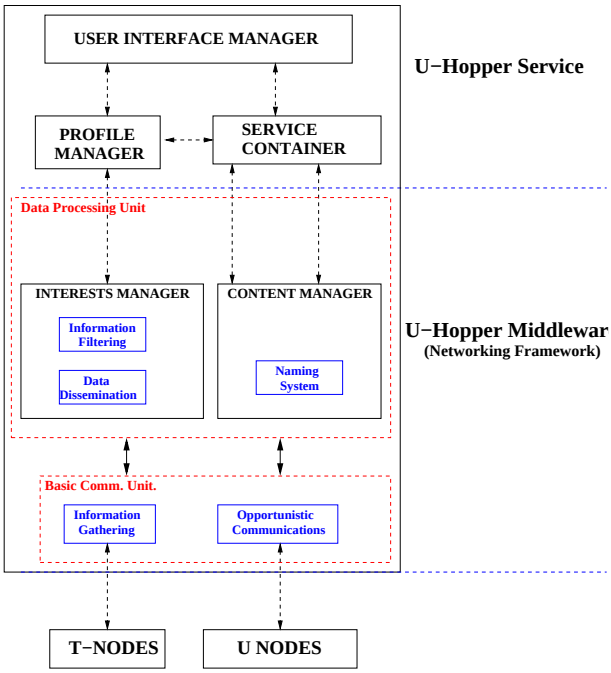

Figure 1: Block diagram and interconnections of the U'Hopper platform. In blue are reported the components of the BIONETS Serworks architecture, and their positioning inside the U'Hopper components.

This includes data originating from T-Nodes (*Information Gathering* of the Serworks architecture) as well as from other U-Nodes (*Opportunistic Communication* of the Serworks architecture).
Fig. 2 depicts the handshake regulating the data exchange between any 2 nodes meeting. The data exchange is triggered by Node 1 receiving a beacon message used for discovering neighboring peers. In response to the beacon message, Node 1 sends its own interests (Interests 1 MSG), which are a description of the information Node 1 is interested in. Node 2 responds with the data stored in its own internal memory matching Node 1 interests (DATA 2 MSG), and subsequently piggy backs its own interests (Interests 2 MSG). Finally, the data exchange is terminated with Node 1 sending any data matching Node 2 interests. The corresponding system components interaction diagram, when generating the user interests, is presented

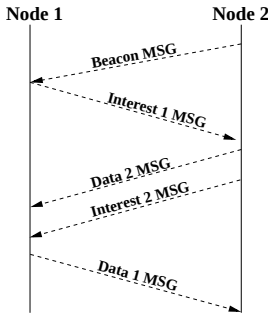

Figure 2: Data handshake between any 2 nodes meeting.

in Fig. 3. When a beacon message is received by the BCU, a request for the user interests is invoked. Such request is then captured by the IM, which gathers the user profile, the service constraints and returns the user interests. It is worth remarking that the described actions are performed by the U-Hopper platform transparently to the user, thus increasing the system usability of the system, since it does not require any human intervention.

On the counterpart, when the interests from an encountered node are received, the data stored in the

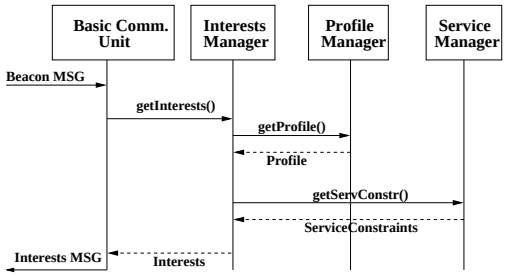

Figure 3: System components interaction flow, when generating the user interests.

internal memory is searched accordingly, and information matching the received interests is send back. The corresponding system components interaction flow is depicted in Fig. 4.

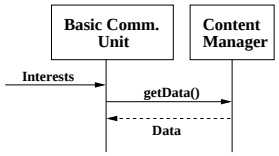

Figure 4: System components interaction flow when retrieving data, starting from user's interests.

2.3 Implementation Details

The main goal of this task is to show a prototype implementation of the described system over widely diffused software/hardware platforms. In order to embrace the largest number of "potentially available" mobile devices, we selected smartphones as the target platform over which we developed U-Hopper. In fact, smartphones are nowadays typically carried around by users during their daily activities and at the same time, they have reached a sufficiently large computing and communication power to perform very complex operations. In particular, smartphones represented the minimum software/hardware requirements that we assumed when developing the middleware. This does not exclude that the platform can run on more powerful devices such as, e.g., laptops or PDAs.
In order to leverage on a widely diffused and standardized computing environment, we choose to develop U-Hopper as a java Midlet running over J2ME (MIDP profile 2.0) [7], which is currently available on most of the smartphones shipped today. By implementing U-Hopper as a java component, we are guaranteed that the software will be portable over a large set of devices. In particular, in addition to smartphones, it will be possible to easily port the platform to any java J2SE device.
The fist technological issue to be solved in developing a middleware for opportunistic environment is given by the selection of the proper network interface. In fact, as detailed in Sec. 2.2, opportunistic communications are the primary mean by which information is diffused in the described environment. Currently, Bluetooth is the largest available network interface on mobile phones and can be easily accessed through J2ME dedicated APIs, such as the well-known JSR 82 ([8]). Hence, we choose to rely on this technology for achieving localized peer-to-peer data exchanges among mobile nodes. Obviously, Bluetooth is not properly designed for opportunistic communications, given the amount of time typically required for establishing a connection between two devices. To shorten up this connection time, we leveraged on some assumptions and on few properties of the Bluetooth technology. At first, we have assumed that all the devices running U-Hopper are assigned with a friendly name starting with a given prefix (for instance they all start with 'uhopper'): this simple assumption allows U-Hopper clients to recognize immediately a candidate that runs the same service, avoiding useless attempts of connection with other devices. We assume also that all devices keep a list of recently visited devices in order to avoid too frequent connections with the same peers. A peer is periodically removed from this list, when a given timer is elapsed.
In U-Hopper, each device is always working in server mode, i.e. it is waiting for incoming connections from U-Hopper clients. Alternately, a device can work in client mode, periodically inquiring close-by devices, and consequently inhibiting its server mode until the client operations are not completed. Whenever another device is found during an inquiry, if its friendly name starts with the given prefix and it was not

recently visited, the inquiry process is stopped and the service discovery operation is performed on that device. This peer is now inserted in the list of recently visited devices. If the U-Hopper service is found and it is available, a connection is established and the communication handshake previously described can start. The node working in server mode updates its list of recently visited devices, in order to avoid a new exchange of information with the same peer in a brief time. Following this procedure, two peers can establish a connection in a relatively short time (depending on the number of Bluetooth devices in radio range), a time suitable for having opportunistic communication on Bluetooth enabled devices.
The second issue we had to face was to design a proper persistent storage on each device. Typically, in smartphones data are saved locally using the Record Management Store (RMS), where information can be easily stored as an array of bytes and retrieved using easy-to-use matching methods, similar to common data base queries. We design then a RMS storage for each device, allowing U-Hopper to maintain persistent data, interests and profiles through different execution of a service running on top of the middleware.

Finally, in line with the heterogeneity of the developed platform, we also ported u-hopper over laptops, using the Avetana Bluetooth library, an open source Bluetooth stack that allows applications designed for JSR-82 to run on Linux devices. Although we had to adapt some of the components to a not-embedded environment, as for instance the local storage and the user interfaces, most of the modules of U-Hopper were easily installed and run on several laptops. Laptops were used as T-Nodes generating advertisement information or as collection points, for showing application statistics, running the same communication module described before.
Tab. 2 presents a concise summary of the technologies and devices used in the U-Hopper prototype.

HW platform	Nokia *E*65, Nokia N80, Dell600
OS	Symbian OS 9.1, Ubuntu 6.11
SW platform	J2ME (MIDP 2.0), J2SE 1.5
BT version	Bluetooth 1.2

Table 2: The U-Hopper prototype in a nutshell.

Finally, in Appendix the uhopper middleware package structure, as well as the class diagrams of the most important modules are presented.

3 U-Hopper Use Case: an Opportunistic Content Distribution Application

To better clarify how the uhopper middleware can be used in a realistic setting, let us consider an opportunistic content distribution application scenario, which has been demonstrated at the demo sessions of IEEE MASS 2007 [4] and of the SAC Workshop [9]. The application gives the possibility to either sense information from sensors embedded in the environment, or to push contents into the network. In both cases, data is then diffused according to an epidemic-like diffusion process, where each couple nodes meeting change information not yet received.
The considered scenario is depicted in Fig. 5, and assumes (i) few smartphones and laptops acting as data sources (T-Nodes), (ii) 4 smartphones running U-Hopper (U-Nodes), (iii) 1 laptop acting as a mobile node (U-Node) for collecting application's statistics.

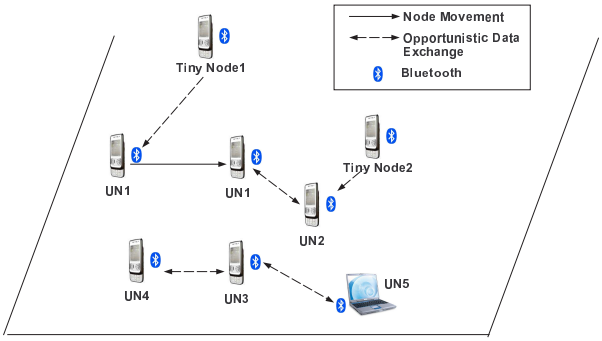

Figure 5: Demonstration of an opportunistic content distribution application scenario.

A few Bluetooth enabled smartphones acted as environmental data sources (T-Nodes). Such smartphones were regularly broadcasting physical data, such as measured temperatures or snapshots of the conference site. Further few laptops were broadcasting context data such as conference social and technical activities. Context data was then collected by other smartphones (U-Nodes) passing by, as for instance *UN1* gathering data from *Tiny Node 1* in Fig. 5. Information was then epidemically diffused by means of opportunistic P2P data exchanges among U-Nodes.
After installing and starting the application, users were asked to select their preferences. Such preferences influenced the data gathering and data exchange process. Users could then visualize the collected information, such as snapshots and advertisement information, at any time on their smartphone's graphical interface. Furthermore, U-Hopper users will have the opportunity to share some information in a totally P2P fashion, directly acting as source of data. For instance they could share some personal midi files (e.g. ring tones) with other peers, if interested in entertainment information, and then listen to the received files through a simple UI. In Fig. 6 two users are shown while using U-Hopper.

Figure 6: U-Hopper on Nokia E65 smartphones.

In order to show the outcomes of the demo (such as collected information and statistics on users P2P communications) in a more user-friendly way, users can visualize at any time such information through an

application running on the laptop acting as a user node shown in Fig. 7), available for everyone interested in our application .

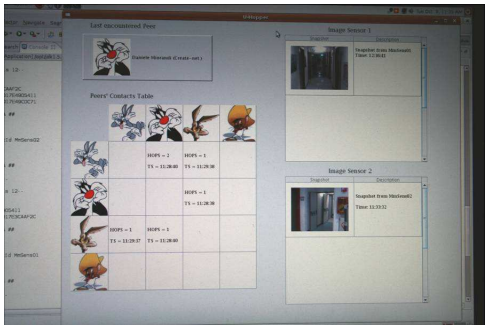

Figure 7: The laptop interface, showing contacts among 4 users and data gathered from T-nodes taking snapshots of the site.

4 Lessons Learned

The main objective of this prototyping activity is to obtain some hands-on experience in developing opportunistic communication systems, and to provide a valuable feedback to the project partners working on the networking aspects of BIONETS.
As the first step, we concentrated on the networking performance of commercially available wireless technologies. In particular, we considered bluetooth as the proximity communication medium and analyzed its performance, as perceived from the application layer. What emerged from our experiments is that connectivity represents a critical bottleneck of existing off-the-shelf wireless proximity communication technologies. More in detail, despite the nominal data rate of the Bluetooth 1.0, it is not possible to exceed 60 Kb/s when exchanging data among mobile nodes. Even worst, the discovery time – the time needed for any two mobile nodes to discover each other and to establish a communication link, can be very large. Such discovery time is composed by (i) the peer discovery time (ii) the service discovery time. Also in this case, we conducted several experiments and the conclusion is that in the average it takes approximatively 95 sec. to complete both discovery phases. From this first analysis of the commercially diffused short-range wireless technologies, we can conclude that is of paramount importance to design forwarding strategies able to maximally exploit any data exchange among nodes. In particular, it is not possible to use a too high redundancy, and it is important to define efficient data management techniques able to limit the amount of data being exchanged. As an example, active queuing techniques can be of help in discriminating between more or less (to the user) relevant content. Further, traditional end-to-end metrics are not sufficient to analyze the performance of opportunistic networks. Instead, data-centric ones (i.e., number of interested users reached) reflect better the nature of the applications that are supposed to run on top of the disappearing network infrastructure.

A second conclusion comes from a set of experiments that we conducted inside create-net premises.We have monitored people's encounters by tracing their proximity for a 4 weeks period. During the experiment, 21 workers - with different roles within the organization and working on different floors of the same building - were equipped with a mobile phone running a java application discovering and tracing neighboring peers approximately every 60 seconds. Whenever the proximity of another device was detected, its bluetooth address, together with the meeting timestamp [1], was saved in the permanent storage of the device for a later processing. In order not to overload the devices memory, a bluetooth enabled laptop acted as a gateway toward a centralized database, gathering the stored information from any smartphone in proximity and transmitting such information to a remote repository. The result of this experiment is a trace which includes a series of contacts, with each contact fully characterized by a timestamp, the IDs of the met nodes and the duration of the meeting. The result of this experiment is that the connectivity pattern of nodes is far from being regular. As an example, we have analyzed the network's *Contact Graph* (CG) [**?**], which is a graph-based representation of the network of contacts, as obtained from the experimentation. In such graph, vertexes represent the nodes of the network, and edges a contact (or a series of contacts) between a couple of nodes. The presence of each edge is regulated by some metric such as, e.g., the cumulative contact duration between nodes couples, or periodicity of such contact. In our work, the edges of the CG are regulated by the cumulative contacts duration, which measures the overall time that 2 nodes i and j have been in contact during the entire duration of the experimentation. In other words, an edge exists between any 2 nodes if they have been close to each other, over the entire duration of the experimentation, for a period of time longer than a predefined threshold d_{thr}. As such, the CG shows the stronger relations existing among the people in the office environment.

Fig. 8 presents the CG in the case of a cumulative duration threshold d_{thr} equals to 40000 s., which corresponds to nodes being in proximity for approximately 30 minutes per day. Given this threshold, Fig. 8 shows strong relations among nodes such as, i.e., people working in the same group or going regularly at lunch together. This is confirmed by the different shapes of nodes [2] correspond to different groups within the office environment. As expected, people working together tend to have stronger relations and to be somehow isolated with respect to other colleagues. A few nodes (e.g. nodes 6 and 18) guarantees the connectedness of the network, and represent those people working in collaboration with different groups.

From this experiments it is evident that there is a "social network" residing behind the connectivity patterns of nodes, and highly influencing the ultimate diffusion of messages. Further, this structure, at least in this simple experiment, is far from bein uniform.

[1] Nodes are using the phone's internal clock for determining the timestamp. Since SIM cards are inserted in the phones, it possible to synchronize such clock to the GSM network. This ensures a sufficient level of precision, especially when compared with the granularity of the peer discoveries.

[2] In Fig. **??** the different shapes are explained in correspondence of the office environment organizational structure.

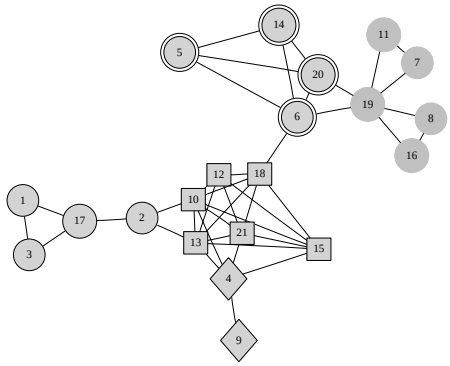

Figure 8: Graph-based representation of the network of contacts, as obtained from the real-world experimentation. An edge exists between any 2 vertexes if the corresponding persons have been in proximity for approximately 30 minutes per day.

In conclusion, the following lessons can be learned from the U-Hopper prototyping activities and be of interest for further development within BIONETS SP1:

1. any forwarding algorithm should explicitly account for the limited duration of contacts. In a real setting, and with real contents being transferred over the wireless link, it is not possible to assume that two nodes meeting are able to exchange all the data stored in their internal memory. This, on the one hand supports the use of forwarding algorithms, as opposed to broadcasting ones, but on the other put a special emphasis in the way these algorithms are designed;
2. given the pervasive nature of the services that will be running on top of the disappearing network infrastructure, it is important to apply data-centric systems designs. The traditional end-to-end paradigm is not well suited for for the mobile network under consideration, since we are not aiming at sending a message from a source to a destination but, rather, to diffuse content to interested users. It is therefore important to define metrics able to better evaluate this performance shift;
3. non-homogeneous scenarios should be taken into consideration when designing any forwarding mechanism. It is not possible to transparently apply results for homogeneous scenario, to heterogeneous ones.

5 Conclusions

In this task, we developed a Pervasive Ubiquitous Peer-to-Peer context-aware Application, specifically tailored to the diffusion of contents in BIONETS-like environments. The prototype implements a subset of the BIOLNETS networking framework functionalities. It has been developed as a java midlet, able to run on a wide set of portable mobile devices. This experience allowed us to better understands (i) which are the constraints and the requirements imposed by real world application scenarios (ii) where state-of-the-art technologies are today, with respect to the BIONETS vision (iii) to run some simple experiments in order to evaluate some of the main characteristics of opportunistic networking.

These conclusions are important for further developing the networking framework, and better understand which are the most promising algorithms.

References

[1] L. Pelusi, A. Passarella, and M. Conti, "Opportunistic networking: data forwarding in disconnected mobile ad hoc networks," *IEEE Comm. Mag.*, vol. 44, no. 11, Nov. 2006.

[2] M. Nicolai, N. Behrens, and E. Yoneki, "Wireless rope: An experiment in social proximity sensing with bluetooth," in *Proc. of PerCom*, March 2006.

[3] D. Miorandi (editor), "Requirements and Architectural Principles: Architecture, Scenarios and Requirements Refinements," BIONETS (IST-2004-2.3.4 FP6-027748) Deliverable (D1.1.2), Jun 2006.

[4] I. Carreras, D. Tacconi, and D. Miorandi, "Data-centric information dissemination in opportunistic environments," in *In Proc. of MASS*, Pisa, Italy, October 2007.

[5] D. L. L. B. F. De Pellegrini, D. Miorandi and C. Moiso, "Bionets: from networks to serworks," in *Proc. of BIONETICS 2007 - SAC Workshop*, Budapest, Hungary, December 2007.

[6] I. Carreras, I. Chlamtac, F. D. Pellegrini, and D. Miorandi, "Bionets: Bio-inspired networking for pervasive communication environments," *IEEE Trans. on Vehicular Technology*, vol. 56, no. 1, pp. 218–229, Jan. 2007.

[7] Jsr-000118 mobile information device profile 2.0. [Online]. Available: http://jcp.org/aboutJava/communityprocess/final/jsr118/

[8] Jsr-000082 javatm apis for bluetooth. [Online]. Available: http://jcp.org/aboutJava/communityprocess/final/jsr082/

[9] I. Carreras and D. Tacconi, "U-hopper: User-centric heteogeneous opportunistic middleware," in *Proc. of BIONETICS 2007 - SAC Workshop Demo Session*, Budapest, Hungary, December 2007.

Appendix A1

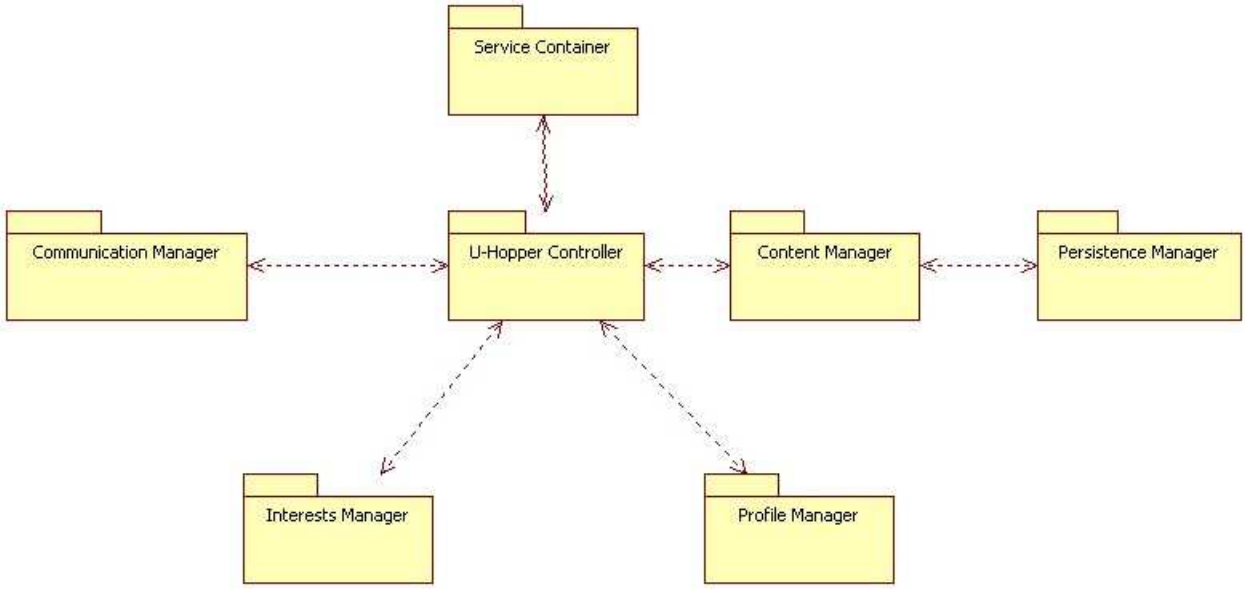

Figure 9: U-Hopper package structure.

Appendix A2

A2.1 Content Manager Class Diagram

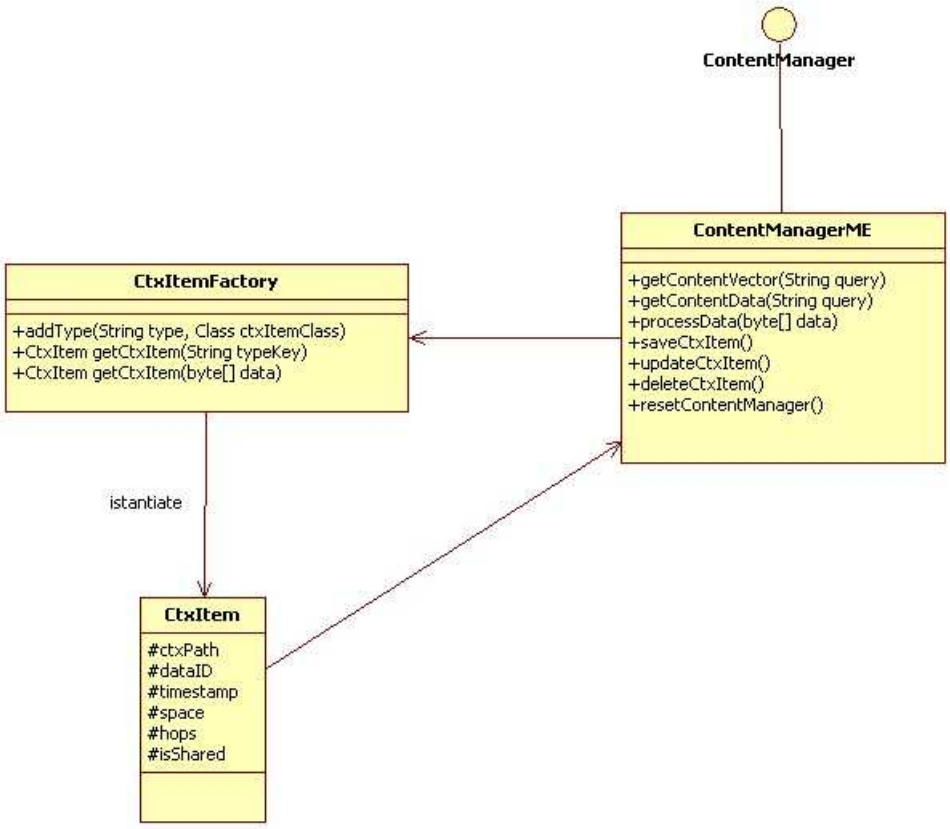

Figure 10: Content Manager class diagram.

A2.2 Interets Manager Class Diagram

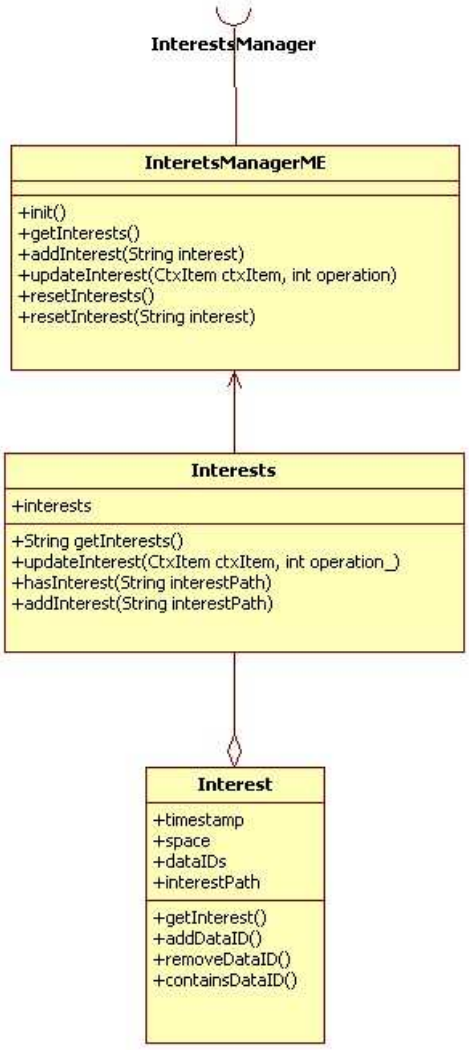

Figure 11: Interets Manager class diagram.

A2.3 Service Container Class Diagram

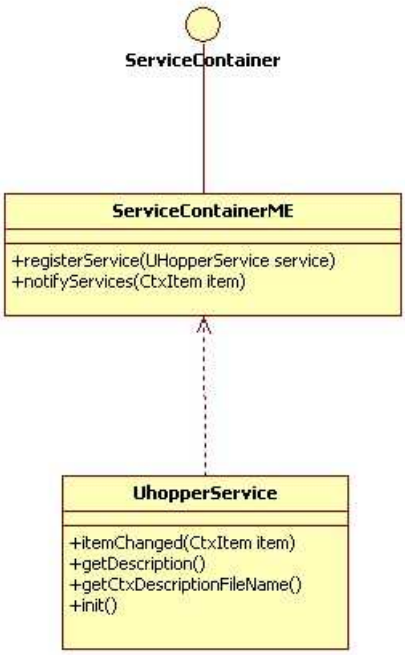

Figure 12: Service Container Class Diagram.