BIONETS

WP 1.2 – INFRASTRUCTURE AND DESIGN

D1.2.2 Disappearing Network Autonomic Operation and Evolution

Reference:	BIONETS/CN/wp1.2/1.0
Category:	Deliverable
Editor:	Danny Raz (TECH) and Francesco De Pellegrini (CN)
Authors:	Francesco De Pellegrini, Daniele Miorandi, Iacopo Carreras (CN), Danny Raz, Reuven Cohen (TECH), Juhani Latvakoski, Tomi Hautakoski (VTT), Lidia Yamamoto (UBASEL), Giovanni Neglia, Sara Alouf (INRIA), Daniel Schreckling (HITEC) Luciana Pelusi, Eleonora Borgia, Marinella Petrocchi, Fabio Martinelli (CNR), Corrado Moiso and Antonio Manzalini (TI).
Verification:	David Linner (TUB) and Paolo Dini (LSE)
Date:	August 6, 2007
Status:	Final
Availability:	Public

SUMMARY

This document presents the results of studying several aspects of current networking technology (disappearing networks) that are expected to play a major role in the deployment of the bio-inspired service centric architecture envisioned by the BIONETS project. It is important to understand that the new architecture will use networking and link layers element in order to establish the services, and thus the characteristics of these layers, their ability to support the proposed architecture and their security aspects are very important aspects of the new networking paradigm.

We start by describing important elements of the link layer with respect to the communication and information dissemination in the network. This is followed by an analysis of the support needed by the service from the lower layers, and the required security aspects. An important part of the document deals with applying the new bio-inspired technique directly into the networking layer, thus demonstrating the multilevel benefit of the concepts developed in the project.

Contents

DOCUMENT HISTORY

Version History

Version	Status	Date	Author(s)
0.1	Draft	2007-05-16	Francesco De Pellegrini (CN)
0.2	Draft	2007-06-22	Francesco De Pellegrini (CN) and Danny Raz (TECH)
0.3	Draft	2007-07-22	Francesco De Pellegrini (CN) and Danny Raz (TECH)
0.4	Draft	2007-07-26	Francesco De Pellegrini (CN) and Danny Raz (TECH)
1.0	Final	2007-08-02	Francesco De Pellegrini (CN) and Danny Raz (TECH)

Summary of Changes

Version	Section(s)	Synopsis of Change
0.1	Backbone structure of D1.2.2	
0.2	First Draft	
0.3	Second Draft	Revisions included
0.4	Third Draft	Typos fixes
1.0	Final version	Typos fixes

1 Introduction

The bio-inspired service centric architecture envisioned by the BIONETS project will be deployed in the forthcoming years over the existing network infrastructure. As explained in great details in D1.1.1 and D1.1.2 the basic elements are the very simplistic T-nodes and the more complex and mobile U-nodes. The communication between the different elements is done via current technology. It is thus very important to study several aspects of these disappearing networks in order to gain a better understanding of the important factors that will affect the novel bio-inspired networks and their expected behavior.

One can partition this study into three major parts. First we need to understand several aspects of the behavior of the underlying networking infrastructure. This is a fundamental aspect, since in BIONETS this is the support for all novel bio-inspired service oriented aspects that leverage the disappearing network. Sections 2 to 5 of the deliverable, in particular, are dedicated to these fundamental issues. In detail, the first part includes optimization issues for the communication between the T-nodes and the U-nodes, with special care given to the correlation among T-nodes; such correlation is exploited in order to achieve better utilization of the information diffusion in the U-node plane. It also includes the potential use of different Erasure-codes in the underlying node to node communication.

The second part of this study deals directly with biology inspired aspects of the networking layer. In this part, i.e., sections 6 and 7 respectively, we include bio-inspired evolution of protocols and of simple forwarding mechanisms. We present initial results that show how bio-inspired concepts (such as evolution) can be used to create self-adaptive services and networks protocols.

The third part of the deliverable deals with the services that are implemented as part of the novel BIONETS vision. We concentrate on the efficient networking support needed by these services in order to provide the complex end user behavior. In particular we concentrate on studying the use of middleware as a basic building block for the envisioned services, and on studying the different primitives needed by these services. Sections 8 to 10 collect all the mentioned aspects.

Providing infrastructure for the next generation services must be accomplished with a close investigation of the security risks associated with the new services and their automated evaluation. For this reason we also study interesting and important security aspects of the proposed architecture.

This deliverable resumes the research on the different aspects at lower layers of the BIONETS architecture, whose aim is to provide the infrastructure for the new services, to explore the potential applicability of biology-inspired techniques even in these layers, and the secure and efficient support needed by the services, and, overall, to present a whole and coherent picture of the work toward a better design of tomorrow's bio-inspired networking.

2 Situated adaptive infrastructure self-optimization

2.1 Introduction

Due to energy and bandwidth considerations, the broadcast of information by the T-nodes to passing-by U-node must be effective and efficient. One way to achieve this target is via collaboration between the T-nodes. For example, a T-node does not have to broadcast information if it is already broadcasted by another T-node in its vicinity. However, this collaboration requires coordination between the T-nodes. Following BIONETS ' assumption where direct communication between T-nodes is not possible, the coordination can be achieved using the assistance of the U-nodes. That is, the U-nodes relay control information between the T-nodes in order to let every T-node decide what data information it should broadcast to the passing-by U-node.

The main purpose of of our research is analyzing the performance gain from the inter T-node collaboration. We would like to decide when it is really necessary to coordinate the broadcasts of the T-nodes, and when such coordination is not really necessary.

2.2 The Model

We now present our formal model for information dissemination from T-nodes to U-nodes. We represent the network as a directed graph $G = (V, E, F)$. Every vertex $v \in V$ is an intersection of roads, and every edge $e \in E$ is a road between two intersections. A *path* is a sequence of edges and a traffic flow $f \in F$ is a set of cars driving along the same path.

Let m be a message that can be sent by some T-nodes. We use $size(m)$ to denote the size of m, and $F(m) \subseteq F$ to denote the set of flows for which this message is relevant. Let $s \in S$ be a T-node, where S is the set of all T-nodes and $size(s)$ be the capacity of s, namely the amount of data the device can send to nearby vehicles while they are within its transmission range. $B(f, m, s)$ is the benefit flow f obtains from receiving message m from T-node s. Thus, $f \in F(m)$ if and only if there exists an s such that $B(f, m, s) > 0$. In a similar way, $F(s)$ is the set of flows that can benefit from assigning a message to s, because each of these flows can view this message, and $B(m, s)$ is the benefit of assigning a message m to T-node s. Thus,

$$B(m, s) = \sum_{f \in F(s)} B(f, m, s). \tag{1}$$

Note that we use a similar notation for $F(s)$ and $F(m)$, since in both cases we refer to the set of relevant flows; $F(s)$ are the flows that are relevant to a specific T-node, and $F(m)$ are the flows that can benefit from a specific message. Let T be an assignment of messages to T-nodes. We say that $(m, s) \in T$ if T contains an assignment of message m to T-node s. Of course, the same message can be assigned to multiple T-nodes. Denote by $T(s)$ the subset of messages that are assigned to s by T. We say that assignment T is legal if and only if for every s, $\sum_{m \in T(s)} size(m) \leq size(s)$ holds.

Figure 1 shows a simple example with two T-nodes (s_1 and s_2), four flows (f_1, f_2, f_3 and f_4) and two possible messages (m_1 and m_2) that are of importance to U-nodes passing through

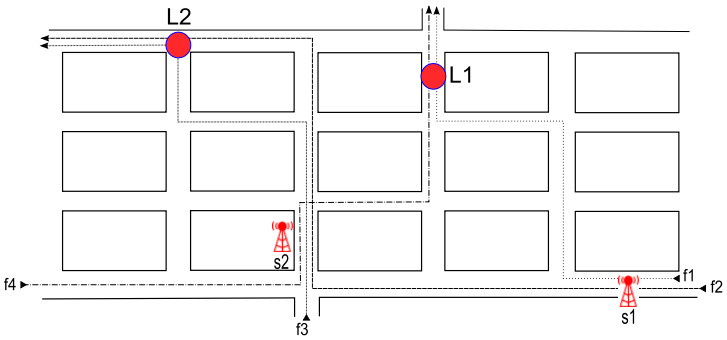

Figure 1: A simple scenario: m_1 and m_2 are messages irrelevant to U-nodes passing through L_1 and L_2 respectively.

L_1 and L_2 respectively. For this example, suppose that $B(f_1, m_1, s_2) = 1$, $B(f_2, m_2, s_1) = 4$, $B(f_2, m_2, s_2) = 6$, $B(f_3, m_2, s_1) = 2$ and $B(f_4, m_1, s_1) = 5$. For the rest of the assignments, the benefit is 0. The benefit of a particular message for each flow is determined by the importance of the message content to the U-nodes comprising the flow and by the size of the flow (in terms of U-node per second). For instance, in the example above, f_2 obtains no benefit from m_1 because it is not affected an event in L_1. In addition, as shown above, the same flow may obtain different benefits from receiving the same message in different places.

2.3 The Maximum Benefit Message Assignment Problem (MBMAP)

We assume that if a message m is assigned to two T-nodes, s and s', and a flow f passes in the transmission range of s and s', then the benefit obtained by f from these assignments is the greater of the two. Hence,

$$B(\{(m,s)\} \cup \{(m,s')\}) = \sum_{f \in F(s) \setminus F(s')} B(f,m,s) + \sum_{f \in F(s') \setminus F(s)} B(f,m,s') + \sum_{f \in F(s) \cap F(s')} \max\{B(f,m,s), B(f,m,s')\}.$$

Under this model, the benefit obtained by a flow f from a message m in a legal assignment T is:

$$B(T,f,m) = \max_{(m,s) \in T} \{B(f,m,s)\},$$

and the total benefit of assignment T is:

$$B(T) = \sum_{f \in F} \sum_{m} B(T,f,m).$$

The optimization problem in this case is:

$$\text{maximize} \quad B(T),$$

$$\text{subject to:} \quad \sum_{m \in T(s)} size(m) \leq size(s) \quad \text{for every s.}$$

We refer to this problem as the maximum benefit message assignment problem or (MBMAP).

2.4 No Interaction Between T-nodes

We start our discussion with the case where there is no collaboration between T-nodes. In such a case, maximizing $B(T)$ is equivalent to maximizing the local benefit obtained for each T-node. We can now distinguish between two cases: the case where all messages have the same size, and the one where different messages might have different sizes. In the former case, an optimal solution can be found in polynomial time using a simple greedy algorithm that assigns the most profitable messages to each device. In the latter case, the problem is equivalent to the well-known NP-Complete Knapsack problem [1, 2]. The greedy algorithm is only a 2-approximation solution, but there exist algorithms that find the optimal solution in pseudo-polynomial time [3]. Moreover, this problem is known to have an FPTAS[1] [4].

2.5 Interaction Between T-nodes

We now show that when T-nodes can interact with each other in order to optimize the profit of their broadcasts then MBMAP is NP-Hard, and that it cannot be approximated within a factor better than $\frac{e}{e-1}$. We show this by a reduction from the well-know maximum coverage problem (MCP) [5], defined as follows: Given a collection of subsets $L = \{S_1 ... S_m\}$ of the universal set $U = \{1, ..., n\}$, and a positive integer p, find a subset $H \subseteq L$ such that $|H| = p$ and the number of covered elements $|\cup_{h \in H} h|$ is maximum.

Theorem 1 *MBMAP cannot be approximated within a factor better than $\frac{e}{e-1}$, even if all messages are of fixed-size, unless $NP \subseteq DTIME(n^{\log \log n})$.*

Proof: [Proof:] It is shown in [6] that MCP cannot be approximated within a factor better than $\frac{e}{e-1}$ unless $NP \subseteq DTIME(n^{\log \log n})$. We now show how to convert an instance of MCP into an instance of MBMAP. Given an instance of MCP, we define the set of flows in MBMAP to be $U = \{1 \dots n\}$ and the set of T-nodes to be $\{s_j\}_{j=1}^{p}$. For every s_j, we set $size(s_j) = 1$ and $F(s_j) = U$. This implies that each device can deliver exactly one message, which will be visible to all flows. In addition, for every subset $S_i \in L$ we define a message m_i such that $F(m_i) = S_i$. Finally, we set $B(f, m, s) = 1$ for every flow f, message m and device s. Observe that under this setting $B(T) = |\bigcup_{(m,s) \in T} F(m))|$. It is easy to see that an optimal solution for the MCP instance is translated into an optimal solution for the MBMAP instance and vice versa. Since the transformation is size preserving and can be performed in linear time, no approximation algorithm for MBMAP can do better than the bound for MCP. ■

At first glance, it seems that MBMAP is very similar to the well-known Generalized Assignment Problem (GAP) [7, 8]. The input for GAP is a set of bins (knapsacks) and a set of items. Each bin has a limited capacity, and for each item i and bin j, $s(i, j)$ and $p(i, j)$ indicate the size and benefit of item i in bin j. The objective is to find the subset of items to be assigned to each bin such that the overall benefit is maximized. However, there are two important differences between MBMAP and GAP:

[1] An FPTAS is an algorithm which given a Knapsack instance and any ϵ, returns in polynomial time a solution that is within $(1 - \epsilon)$ of the optimum.

- In GAP every item can be selected only for one bin, whereas in MBMAP an item (a message) can be selected for multiple bins (SADs).
- In GAP the benefit associated with the selection of an item for a bin is independent of the selection made for other bins. In contrast, in MBMAP there is a strong correlation between assignments. As explained before, if a message is selected for multiple SADs and the same flow passes through some of them, the benefit this flow obtains from this message is not equal to the sum of the benefits, but to the greatest of which.

In light of this similarity, we are working on a solution for MBMAP using a technique similar to the one presented in [9] for GAP.

3 Erasure-coding-based solutions for BIONETS networks

After analysing the two major communication patterns of the BIONETS framework, i.e., T-nodes to U-nodes and U-nodes to U-nodes communications, in this document we study solutions aimed at improving reliability of transmissions and robustness against both packet loss and selfish nodes in BIONETS networks. These solutions rely on erasure codes basically. However, after investigation of pros and cons deriving from application of classical erasure coding based communication protocols (as they are), we now focus, in this working phase, on how to best fine-tune erasure coding to the specific requirements of the BIONETS framework.

3.1 Increasing reliability in communications from T-nodes to U-nodes

The T-nodes to U-nodes communication environment comprises *static* T-nodes located in fixed positions thoughout the environment and *mobile* U-nodes which go from place to place as a consequence of the activities their users carry out over time for job, leisure, etc. Hence, the mobility pattern of U-nodes is quite complex in directions, distances travelled, and also in the speeds assumed. Encounters between U-nodes and T-nodes (which happen when a Unode passes nearby a Tnode can have very different characteristics relative to dynamics factors like the particular trajectory followed by the Unode, the minimum distance to the Tnode which is reached, and the instantaneous speed of the Unode, but also to other factors like the relative antenna orientation of both the Tnode and the Unode encountering, or the presence of obstacles in between the Tnode and the Unode. All these characteristics together determine the *duration* of the time interval in which the Unode and the Tnode are in the communication range of each other (and are therefore able to communicate) and also the *packet loss* profile which is experienced by transmissions attempted in this interval. This time interval is known as *contact time* between the Tnode and the Unode and it can be very limited. Hereafter we will consider contact times as limited *resources*, as we usually consider memory or energy for example. This because contact times can be very short in case U-nodes are extremely fast, but also because during a contact time communications can be very disturbed such that only a small portion of it can really be exploited for transmissions [10]. This leads to the reduction of the *actual* duration of the contact time even in the case U-nodes travel slowly. Another factor which affects the actual duration of a contact time is the presence

of multiple transmissions among different entities. This happens in fact when the contact time is *shared* among multiple nodes, like for example when multiple U-nodes expect to gather data from the same Tnode nearby, or viceversa when a Unode asks for data to multiple T-nodes at the same time because they are located in proximity of each other. We have analysed some of the BIONETS scenarios where communications among T-nodes and U-nodes during contact times are critical due to the occurrence of the above situations. In all these scenarios guaranteeing fast and effective data exchange (transitions) between T-nodes and U-nodes experiencing a contact time is a major requirement. For further discussion on the BIONETS application scenarios analysed please refer to [11].

To allow better exploitation of the limited contact times between T-nodes and U-nodes in the BIONETS framework, we have investigated the possibility to use erasure coding based communication protocols in substitution of classical retransmission-based communication protocols. As is highlighted in the DTN specs [12], *chatty* communication paradigms are not suitable to environments where contact durations are not predictable or are expected to be short. Instead, communication protocols which minimize synchronization between the communication peers and that provide all the data together to destination in a single *one-way* transaction are rather preferable. If we add to this the opportunity to send *interchangeable* data to destination, which can be obtained by erasure coding the original data, a certan degree of loss tolerance is also guaranteed. It results that erasure coding based communication paradigms are quite suitable to the T-nodes to U-nodes communication environment and have the potential to guarantee reliable transactions even in short time interval.

3.1.1 Testing the real behaviour of communication paradigms during contact times

We conducted a preliminary analysis on the behaviour of classical retransmission-based communication paradigms (ARQ-based) and compared this to the behaviour of a *naive* erasure coding based communication paradigm. For our analysis we exploited a real testbed composed of small tiny devices i.e., MICA2 motes, as a case study for T-nodes. In the next subsections we describe the communication paradigms which we compared, the experimental testbed, and the lessons learnt respectively.

ARQ-based vs. Erasure coding based protocols

As retransmission-based communication paradigms we used *stop and wait* and *selective repeat*. On the other hand, as erasure coding based communication protocol we used a naive protocol based on *Reed Solomon* codes [13] [14]. This latter protocol only provides encoding of a burst of original packets into a larger burst of codes and then transmits all the codes generated back to back, in a single transaction. The number of codes generated per single burst of original data packets is the *sketch factor*. When encoding Reed Solomon (RS) codes, the sketch factor is constant. In fact, there exists a limit to the number of distinct codes that can be generated per single burst of original packets, and this limit is fixed. Since the encoding process of RS codes is a matrix-by-vector product, where the matrix is an $(n \times k)$-matrix and the vector is a $(k \times 1)$-vector including all the packets of the burst to be encoded, the resulting $(n \times 1)$-vector which includes all the

codes produced has the same number of rows as the encoding matrix i.e., n. Hence, the maximum number of codes that can be generated corresponds to the maximum number of rows that the encoding matrix can have and, since the encoding matrix of RS codes is a *Vandermonde* matrix, the maximum number of codes that can be generated corresponds to the maximum number of rows that a Vandermonde matrix can have. This directly depends on the cardinality of the *Galois Field* of reference i.e., the field from which all the elements of the Vandermonde matrix, as well as all the elements of the original $(k \times 1)$ data-vector and of the resulting $(n \times 1)$ code-vector are picked up. All the operations involved in the matrix-by-vector product are also executed according to the arithmetics of the Galois Fields. In particular, in the protocol tested the Galois Field (GF) of reference is $GF(2^8) = GF(256)$. Given a set of k data-packets, the number of codes that can be generated for that packets over $GF(256)$ is 255 which corresponds to the number of non-zero elements of $GF(256)$. Furthermore, in the special case of *systematic* codes, the number of codes that can be generated is much higher because increased of k due to the addition of the identity matrix I_k to the Vandermonde matrix to form the overall encoding matrix.

There are many reasons why when encoding RS codes the number of codes generated is not (always) the maximum. Firstly, the computational burden of the matrix-by-vector product increases with the dimension of the encoding matrix. Second, the memory needed to store the encoding matrix permanently at both the sender and receiver nodes is also directly proportional to the dimension of the encoding matrix[2]. Furthermore, the higher is the number of codes generated, the higher is the bandwidth comsumption caused. Since the packet loss experience of codes cannot generally be predicted, the number of codes sent could be much higher than really required thus leading to bandwidth wastage. This would also lead to the wastage of contact time. In the end, the higher the number of codes generated, the longer the delay between transmission of consecutive bursts of packets that a node could have available in its memory. In the *naive* protocol that we tested for RS codes, the sketch factor fixed is 5 corresponding to the ratio between the number of codes generated per burst i.e., 35, and the size of a single burst of packets i.e., 7.

We conducted our tests to compare the efficacy of the three aforementioned communication paradigms on a real-world testbed based on MICA2 sensor motes. This was aimed at verifying the *real* behaviour of classical retransmission-based communication protocols when used for transmissions in a dynamic environment among nodes experiencing a contact time, as well as the *real* behaviour of the encoding-based communication approach.

Experimental Testbed

In our experimentation we considered transmissions of a burst of packets from a static MICA2 mote to a mobile MICA2 mote. We repeated different experiments by beginning transmitting the burst of packets at different points of the contact time, both at the edges (beginning and end) and in the middle. By doing so, it was possible to test both the best and worst case of transmission with all the protocols under study. Obviously, the best case corresponded to transmitting in the middle of the contact time when the distance to destination was minimum, while the worst case

[2]Both the source and destination nodes must know the encoding matrix in advance so as to be able to do the encoding and the decoding respectively.

corresponded to transmitting at the edges of the contact time. To take into account the dynamics of the contact time, we imposed to transmissions the same packet loss previously experienced in a real experimentation campaign and sampled over time. Specifically, the packet loss was measured during transmissions between a static node and a mobile node during a contact time: the mobile node moved along a straight-line trajectory approaching the static node at the beginning, and going away from it afterwards. The mobile node moved with different speeds (corresponding to different scenarios): 1-2m/s, and 20-30-40km/h. Depending on both the instantaneous distance between the mobile node and the static node and the speed of the mobile node, the packet loss gradually decreased first, and increased then. The packet loss profile however, showed great irregularity.
Exploiting packet loss traces in different static experiments rather than repeating the mobile experiment for each protocol to be compared, allowed us to reproduce exactly the same experiment and test each protocol exactly under the same conditions. This fair comparison wouldn't have been possible if we had conducted a sequence of real experiments between a mobile node and a static node because they are always different from each other. Furthermore, the fact that we performed real transmissions between MICA2 motes allowed us to take into account the real timing introduced by a real technology.

Lessons learnt

As was expected, the experimentation highlighted that the RS coding based approach allows better exploitation of the limited resource contact time. In fact, when contrasting the total transfer time needed for a burst of packets to go from the sender to the receiver, this was much shorter when applying RS coding with respect to when applying retransmission-based approaches. When the packet loss was low (in the middle of the contact time), both retransmission based and erasure coding based communication paradigms showed good performance. However, the RS coding based approach caused shorter burst transfer times. When the packet loss was high instead (at the edges of the contact time), we had to discriminate a few cases. With very severe packet loss neither retransmission-based communication protocols nor the encoding-based communication protocol succeeded in transferring the entire burst of packets. When the packet loss gradually decreased, both protocols improved, however retransmission-based communication protocols improved faster than the encoding-based protocol. This means that the RS coding based approach continued to fail while retransmission-based approaches started being successful. This lasted until the packet loss decreased down to a certain threshold corresponding to the redundancy added to transmissions with RS encoding. This was because RS codes can tolerate packet loss only up to that certain degree since the protocol provides transmission of the only codes generated during the encoding phase and then simply stops transmitting. On the other hand, the other protocols attempt retransmissions whenever possible. Hence, the number of failures which were experienced with RS coding was higher than the number of failures experienced with retransmissions. It should finally be noted that when the RS coding approach failed all the codes arrived to destination were useless because they could not be decoded since they were too few. This led to a lot of bandwidth wastage. In case of unsuccessful retransmissions instead, the packets arrived could always be exploited. An advantage of the RS coding based approach is instead that, whenever it is successful, its performance in terms

of total transfer delay of the burst is always better than the other retransmission-based protocols.

Given the lessons learnt from the experimentation described, we have studied a more flexible protocol based on RS codes which combines the advantages of RS codes to the adaptability to the changing packet loss of ARQ-based paradigms. It is discussed in section 3.1.2 below.

3.1.2 A novel adaptive communication paradigm based on Reed-Solomon codes

We have developed a novel *adaptive* communication protocol, based on RS codes, which adapts encoding to the actual BIONETS communication environment. The main idea of this protocol is to manage a flexible generation of codes which takes into account the actual packet loss which is experienced over time. In fact, when the packet loss experience is low it is useless to produce many codes because only a small subset of them will be utilized at destination whereas the others only cause computational overhead during encoding and waste bandwidth during transmission. Hence, when the packet loss is low, only small redundancy is required. This also contributes to preserve energy (that is wasted in both computation and transmission of useless codes) and to delay less the burst transmission (the encoding phase is concentrated at the beginning of transmission). When the packet loss is high more redundancy is required. However, the redundancy should be produced on demand and no limits should constrain the total amount of redundancy introduced. Whenever new codes were required there should be the possibility to generate them.

Our protocol for communications from T-nodes to U-nodes is based on the *pull* model [14]. U-nodes moving around in the network area periodically broadcast advertisements (ADVs) to announce their presence. T-nodes that are able to catch the ADV of a Unode interpret it as a request for context data. Moreover, ADVs are used to keep track of the U-nodes with which a contact time is being experienced. Whenever a Tnode receives an ADV from a Unode, it updates a *contact list* which contains the identity of all the U-nodes that are currently in proximity and have sent an ADV to the Tnode. T-nodes are able to communicate with all the U-nodes in the contact list. After inserting a Unode into the contact list, the Tnode encodes the context data it stores and starts to send out the codes produced to the Unode[3]. The first point here is: how many codes should be generated at the beginning? We have decided to produce a small amount of codes at the beginning, including only 10 to 20% redundancy because there is no knowledge a priori of the packet loss which will be experienced. This produces multiple advantages: i) the computational burden of encoding at the beginning, which is proportional to the number of codes generated, is not that high and causes small energy wastage, ii) the less the codes produced at the beginning, the shorter the initial delay which is introduced before starting the transmission of the burst of codes, and iii) in case the packet loss eventually experienced over the channel is low, the number of codes generated is sufficient to allow the Unode to receive as many codes as needed to do the decoding and access the entire burst of original data.

After sending out the first bunch of codes the Tnode waits for acknowledgement (ACK) from the Unode. A Unode sends an ACK back to the Tnode when it has decoded the burst of data, meaning that it does not need to receive extra codes. The Tnode keeps track in the contact list of

[3]Actually, transmissions are in broadcast.

the U-nodes which have already sent an ACK. If all the U-nodes in the contact list have already sent an ACK the Tnode does not have to produce and send new codes.

When the Tnode starts transmitting codes, it sets up a timer to a retransmission timeout (RTO) value. This value is an upper bound limit for the time interval within which an ACK is expected for the codes just sent out. When the timer goes off, if there are still U-nodes in the contact list which have not sent their ACK yet, the Tnode starts producing new codes and to transmit them. A new timer is then set up with an (slightly) increased RTO value. The number of codes which is generated this time corresponds to the length of a transmission window (WND). This can be equal to the number of codes generated the first time or a small larger to adapt faster to a higher packet loss. By encoding a WND of packets per time, the computational burden of encoding is distributed over time and useless encoding is avoided. Once collected the vector of original packets to be encoded, one single code at a time can be generated by multiplying a single row of the encoding matrix by the original data vector. There is no need to store permanently the Vandermonde matrix neither at the sender Tnode nor at the receiver Unode since one single row of a Vandermonde matrix is uniquely identified by an element of the Galois Field of reference. Hence, whenever a new code is needed for transmission, a new element (GEN) is randomly extracted from $GF(256)$ and the corresponding row of the Vandermonde matrix is built up. Then, the new code is generated by multiplying this row by the original data vector. By selecting elements of $GF(256)$ at random, the probability to generate the same code multiple times is low. In the end, by including the value GEN together with the code produced into the packet to send out, the receiver is able to reconstruct the row of the Vandermonde matrix that it needs to do the decoding. This causes a little packet overhead but it is minimal and definitely negligible.

Unfortunately, while the computational burden of the encoding process can be distributed in different encoding stages, the computational burden of the decoding phase cannot be distributed in any way and the original data vector can only be reconstructed at once after inverting the submatrix of the encoding matrix which corresponds to the codes arrived at destination. Since the dimension of the matrix to be inverted in the decoding process is $(k \times k)$, the size of the burst of packets which are encoded together should remain as low as possible.

When applying the *pull* model, multiple T-nodes may receive the same ADV sent out by the same Unode. This means that multiple T-nodes will encode their data and send the codes produced at the same time. We assume that a MAC layer protocol is present to solve problems of contention in accessing the channel. Furthermore we assume that T-nodes in proximity of each other store the same context data and hence have the potential to produce the same codes. However, by generating codes relying on a random choice of the GEN value, the probability to generate multiple copies of the same code is very low[4]. When multiple T-nodes send out (distinct with high probability) codes to the same Unode at the same time, the overall effect is that of a larger redundancy generated. This is particularly beneficial in case the packet loss is high, whereas it is slightly inefficient when the packet loss is low because it just leads to bandwidth wastage. However, this effect can partially be attenuated by keeping low the initial amount of redundancy generated. In a similar scenario,

[4]Verbatim copies of the original data packets, which are considered codes in *systematic* codes, should not be sent all at once at the beginning of transmission but rather distributed over time.

it is expected that the Unode is able to decode soon and to send back an ACK consequently. All the T-nodes will stop producing extra codes then.

Let focus now on what happens in case multiple U-nodes approach the same Tnode at the same time. In fact, all these U-nodes share the same contact time, or better, there is overlap among the contact times experienced by each Unode with the same Tnode. Actually, what the Tnode is aware of is a single *extended* contact time over which it sends out (broadcasts) codes because there is *at least* one Unode nearby expecting to receive them. The *contact list* keeps track of the U-nodes nearby. A Unode is inserted in the contact list when the Tnode receives an ADV from it. The contact time contains the identity of the Unode, the time of the last ADV received and a flag stating whether or not an ACK has been received by that Unode. Each entry of the contact list remains active for the entire contact time between the Tnode and the Unode the entry is titled to. Whenever a new ADV arrives from a Unode already present in the contact list, its timestamp is stored. This has the effect to refresh the relative entry of the contact list and to witness that the Unode is still present nearby and that it can receive data from the Tnode. ADVs should be transmitted *frequently* because an ADV can go lost due to packet loss, but this shouldn't be interpreted as the Unode no longer being nearby. Instead, a Unode should be considered far away after having not received a certain number of ADVs from it, say 10 ADVs[5]. A timer can be set to manage periodic updates of the contact list. It should be set up with and initial value corresponding to the expected duration of the contact time. When the timer expires if no recent ADVs have arrived to refresh an entry then the entry should be deleted meaning that the Unode it is devoted to is no longer in proximity. Actually, it does not need to introduce a new extra timer to update the contact list. Instead, the same timer which is used to set the retransmission timeout RTO can be exploited. When the timer goes off the contact list is updated such that inactive entries are eliminated. Then, if there are active entries left, relative to U-nodes which have not sent an ACK, encoding and transmission of a new burst of codes are triggered. Transmissions end when either the contact list is empty or all the U-nodes in the contact list have already sent their ACK.

Fine-tuning of the parameters of this protocol is required. Among the most important parameters to investigate there are: i) the initial value of the transmission window i.e., how many codes should be generated at the beginning, ii) subsequent updates of the transmission window i.e., how many codes should be generated after the first time, iii) which frequency ADVs should be sent out with?, and iv) after how much time an entry of the contact list should be considered obsolete and thus deleted? An accurate selection of these parameters can significantly increase the protocol performance.

3.2 Boosting performance of data dissemination protocols at the Unode plane

In the Unode plane application of erasure coding can give added value to the data dissemination forwarding protocol which is used. Erasure coding has the potential to increase reliability

[5]This causes a *slow* detection of a Unode non-presence, however a timely detection would not add value to this protocol and thus it is not considered an issue.

of communications (against packet loss) and to give to the system much more robustness against uncooperative or selfish nodes (which cause path failures). However many issues arise which needs dealing with: i) how much redundancy should be added to the original data?, and ii) how to distribute codes to the relays met so as to maximize the delivery success rate? The major difference between the U-nodes to U-nodes communication pattern and the T-nodes to U-nodes communication pattern is the number of hops involved in transmissions from a sender to a destination. While in T-nodes to U-nodes communications only single-hop transmissions are concerned, in U-nodes to U-nodes communications multi-hop transmissions are mostly concerned. This basically increases the order of magnitude of the problem.

In the Unode plane packets are spread throughout the network so as to exploit multiuser diversity for data delivery. In fact, if a set of packets with a common destination flow in parallel towards the destination by following different paths, the total delay experienced for the entire set of packets is reduced. Furthermore, if packets are duplicated and spread throughout the network, also multipath diversity is exploited. The delivery success rate is optimised because the system is robust to the loss of some packets and to the loss of some paths too. This thanks to the presence of packet replicas thoughout the network.

The major issue regarding the Unode plane is how to find a route to the destination practically *on-the-fly*. Multiple approaches have been proposed for the data forwarding problem, however this is still an open issue since the right tradeoff between resource consumption (e.g., bandwidth consumption and memory occupancy) and performance achieved (e.g., delivery success rate and transfer delay) has not been met yet. Erasure coding is surely a valid instrument in this larger context, however it should be applied with caution because it could just increase the overall protocol overhead without leading to valuable performance increase.

To investigate adoption of erasure codes at the Unode plane, it is necessary to discriminate between the case in which erasure codes are used combined with dissemination-based data forwarding protocols and the case in which they are used with context-based data-forwarding protocols. The major difference between these two approaches is in the number of intermediate relay nodes exploited for data forwarding, which is higher in dissemination-based protocols, and in the strategy followed to choose relay nodes, which is a *non-strategy* in case of dissemination-based protocols and maximization of some kind of utility function in case of context-aware protocols.

By erasure coding messages at the source Unode, some redundancy is introduced in the network by the source itself. It is defitely better having redundancy in form of codes rather than replicas because codes are interchangeable while replicas should simply be discarded at destination.

In disssemination-based data forwarding protocols messages are disseminated throughout the network in epidemic-like fashion. Whenever a couple of U-nodes meet up with each other, they exchange the messages they carry in their buffers. After data exchange the two U-nodes have their buffers synchronised i.e., with identical content (this actually happens in the extreme case of epidemic protocol). The degree of data replication is very high for this kind of protocols. When exchanging codes rather than plain messages, things should not change that much actually. Multiple U-nodes will store in their buffers the same codes and the probability to collect replicas of the same code at destination is quite high. Even though with dissemination-based data forwarding

protocols, the delivery success rate is very high and the delay experienced of packets is very low instead, they are also the most expensive data forwarding protocols in terms of both bandwith consumption and memory occupancy at the intermediate nodes buffers. Hence, when adding the computational burden of encoding things could get worse.

We expect better performance from application of erasure coding to context-aware data forwarding protocols. According to these protocols, maximization of some utility function is targeted while choosing the next hop node for transmission. A similar approach could be adopted to decide how many codes to pass to the encountered node. Given that in context-aware protocols each node typically keeps statistics about the delivery success rate associated to each potential destination Unode, these could be exploited to generate larger or smaller redundancy when producing codes destinated to that particular Unode. Similarly, the delivery success rate that an encountered relay node shows for a given destination node could be exploited to decide how many codes for that destination to pass to the relay node. By taking *conscious* decisions about generation of redundancy as well as *who to forward what*, and tuning these decisions to the network characteristics will allow erasure coding to boost performance of data forwarding protocols.

However, all these issues will be further investigated in future deliverables.

4 Interoperability with traditional networks

The use of the BIONETS system architecture can be coupled to existing traditional IP networks. In particular, as described in D1.1.2, the existing IP infrastructure can be leveraged in a transparent manner through the use of dedicated nodes, i.e., the Access Points. One fundamental remark, about the use of BIONETS when interfaced with the legacy network infrastructure, is that the operations sustained though APs are by no means intended to inject and propagate uncontrolled *environmental information*, i.e. sensed data coming from the T-nodes, through the traditional network interface. This in fact could cause the injection of a massive amount of raw data into IP networks. Back to the main intuition, in fact, sensed data coming from T-nodes, should leverage just local communications for the sake of scalability. Obeying to the above constraint, the use of Access Points, in particular, will be cautionary limited to the three types of operations resumed in the following.

4.1 Retrieval of Remote Data in the IP Network (B2IP)

In this type of operations, a service in the BIONETS network, hosted on a U-node, can issue a query to the traditional network, where some information can be retrieved. Anyway, such operation is completely transparent to the U-nodes, since the task of the translation of the query from the BIONETS format to a format suitable to the IP domain is demanded to the AP. To this respect, the AP acts as a proxy [14].

One typical application where such operation may prove relevant, is the retrieval of data from a remote server with respect to data which cannot tolerate any significant delay. Such an example can be given by a service which requires the value of a currency exchange; such a value has to be

evaluated locally at a U-node, in the case that an advertisement was found but the actual value of the currency in use has to be evaluated in order to have an exact comparison with another offer. Clearly, in the currency exchange example, the data need to be the most up-to-date, and for such reason only data available at a central server (e.g. a central authority), can provide the required accuracy. Of course, such daily data can be later on made available locally. But, as a first step, the query should be issued locally and propagated through the diffusion mechanism to an AP, in order to retrieve such remote information.

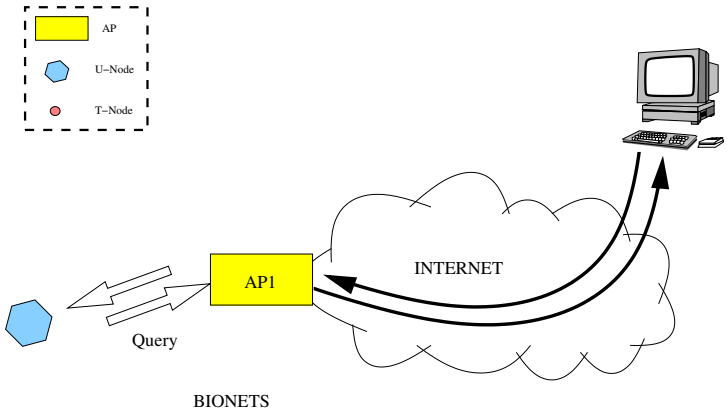

Figure 2: B2IP data exchange: the remote user located in the network retrieves remote information leveraging the interface thorough the AP.

The operations involved in this type of communications are, in order:

1. In the proximity of an AP, the service at a U-node performs a query about a certain data. In the currency example, the daily data for the yen/euro currency exchange is queried;

2. The interface of the AP towards the U-node processes such information and retrieves the query; the AP recognizes that the data should be queried at a remote host, and translates the query to the P2P unit [15]. The way the AP knows that the data is not local is related to the format of the query message issued by service, where the query can be specified.

3. The AP P2P unit issues the query over the IP overlay P2P network and retrieves the address of a server where the data can be found;

4. The query is then delivered by the AP and processed at the remote server over the IP network;

5. The server response is pushed back to the AP P2P unit;

6. Finally, the data matching the U-node query is delivered to the U-node;

In this particular data flow, as depicted in Fig 2, the AP is seen as another U-node to which a query can be performed, and all the data transactions are performed *as if* the AP was actually a U-node which collected the information either from T-nodes readings or leveraging the information diffusion mechanism.

4.2 Retrieval of Environmental Data from the IP Network (IP2B)

Local environmental information can be accessed from nodes in the IP network, as depicted in Fig. 3. In fact, at the boundary of the IP network, an AP will be able to translate the queries about the local environmental status into a BIONETS request of information. In particular, one

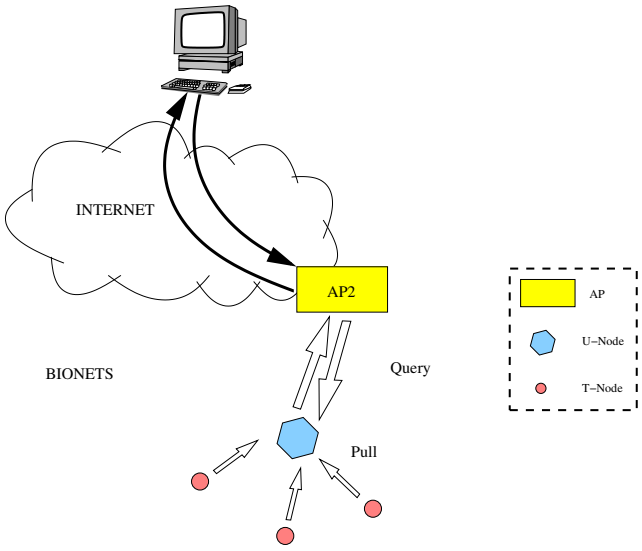

Figure 3: IP2B data exchange.

such example can be given by a user who wants to know if a certain mall is crowded or not at some point of the day, in order to do some shopping. Instead of guessing, the user can perform from her remote location a specific query, and retrieve the answer using the BIONETS communication mechanisms. The operations involved in this type of data retrieval are listed in order in the following:

1. The query can be performed on a p2p system on the user’s device on the IP network: the query, in particular, will possess two attributes, i.e. the data type to be retrieved and the location where the data should be retrieved; in the specific mall example, a certain mall would be targeted by the user;

2. The query is resolved on a p2p client on the user terminal;

3. We assume that a remote p2p host is part of one or more APs where the query can be issued; in the mall example, they are part of some APs installed in the proximity of the mall;

4. The query for the “crowdedness” within the mall is then performed from the AP to U-nodes passing by;

5. The information diffusion on the specific query is performed;

6. The information is possibly retrieved at local T-nodes or measured by U-nodes based on the number of intermeetings during a certain interval;

7. The answer is then recovered at the AP and sent back to the remote user of the IP network.

In the case of a IP2B information exchange, the AP is perceived as a U-node performing a request to another U-node passing by. In the example, the query about the crowdedness of the mall can be carried by a single U-node, which performs local measurements on the number of users in the surroundings and replies such data to the AP. The final delivery over the Internet to the source of the query is performed by the AP.

4.3 Tunneling of queries (B2IP2B)

The third communication pattern which requires the interoperability with IP networks consists of a method which leverages the presence of the IP network nearby a U-node in order to fasten the diffusion of queries. A pictorial representation of this mechanism is reported in Fig. 4. This can be convenient when the local mobility is not sufficient to permit fast enough diffusion of data.

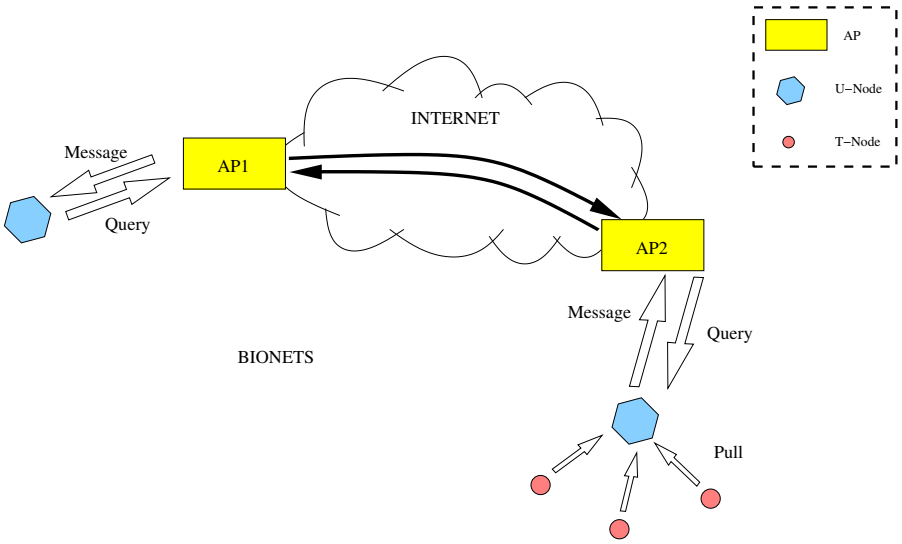

Figure 4: Tunneling of queries through the APs.

Then, queries can anyway diffused if several APs are deployed over a given area and the lack of mobility can be overcome simply tunneling the query from one AP towards another AP located at a different location. The operations involved in this type of data retrieval are as follows:

1. The service at a U-node in the proximity of an AP performs a query about a certain data;
2. The AP interface to the U-node processes such information and retrieves the query; the AP recognizes a tunneling is possible.
3. Within the AP, the P2P unit retrieves the APs in the surroundings where the query can be propagated; in particular, the AP will require a mandatory field with the scope of the query, in order to limit the propagation of the query outside the area of interest[6];

[6]It is recommended, anyhow, that a maximum possible range for the propagation of the queries is hardwired in the AP.

4. The query is cached at such APs waiting for delivery to U-nodes passing nearby;

5. The reply message from a U-node passing nearby is sent back to the peer AP and delivered to the U-node.

In this case, the APs behave exactly as U-nodes propagating queries, but this technique can be used to perform remote sensing.

4.4 Interoperability with legacy Delay Tolerant Networks

Within the BIONETS architecture, there are numerous T-nodes that generate data needed to support a variety of services; this data becomes partially, locally and temporarily available at the U-nodes layer when U- and T-nodes interact (the U-nodes have the ability to read the data available at the T-nodes and apply a service-oriented dissemination of the information acquired). No matter how "smart" and efficient a dissemination mechanism might be, the lack of connectivity (which is a typical characteristic of the BIONETS architecture) makes mobility the primary means for the efficient communication among the nodes; in the extreme case of a totally static network, partitions are an obstacle that cannot be overcome.

The importance of mobility with respect to information dissemination motivates the investigation of whether it could be possible to take advantage of the mobility of other mobile nodes (besides the typical "BIONETS-compatible" U-nodes) wandering around the same area the U-nodes do and constituting other networks besides that of BIONETS . More specifically, the case where a number of U-nodes are located on an area where legacy Delay Tolerant Network (DTN) nodes are also present is considered. The term legacy DTN nodes refers loosely to nodes that form a delay tolerant network, e.g. a network where the store-carry-and-forward paradigm is applied for communication due to lack of node connectivity, but the nodes are not BIONETS-compatible, i.e. they cannot read the data available at the T-nodes and are unaware of the data dissemination rules and the communication protocols used among the U-nodes.

Since the U-nodes are assumed to be equipped with a typical wireless interface, the communication between the U-nodes and the legacy DTN ones is feasible.[7] In this case, the data copies that are shared among the U-nodes could be also relayed through the legacy DTN network; to this end, the U-nodes should encapsulate their BIONETS-specific messages into a legacy DTN-compatible frame that would be spread by the legacy DTN nodes transparently (however, the legacy DTN nodes would not process the message, e.g. by data filtering, as the U-nodes would) leading to a communication tunnel between the U-nodes through the legacy DTN[8].

A challenging and interesting issue to be addressed is how the characteristics of the dissemination algorithms used by the legacy DTN nodes interact with those for data dissemination among the U-nodes. Notice, that such a mechanism would speed up the spreading of the messages in the U-nodes layer but might violate some typical characteristics of information dissemination in

[7] The only additional requirement is that the U-nodes support the corresponding communication protocols.

[8] We may assume that the (legacy DTN) message would be marked with a specific destination address indicating that all BIONETS-compatible nodes, i.e. all the U-nodes, are potential destinations of the specific message, so that the interaction between a U-node and a legacy DTN one leads to the successful transaction of the BIONETS message.

BIONETS . This is because a BIONETS message copy that would be encapsulated in a legacy DTN message would disappear from the U-nodes layer and reappear when delivered to some other U-node (or even U-nodes). These temporal and spatial data displacements as well as the potential existence of several replicas of a single encapsulated BIONETS message are parameters that affect the basic functionality of the BIONETS information dissemination algorithms. For example, a timestamp remains valid, even if the message is encapsulated in some other message; however, a Time-To-Live field, expressed in number of hops, would only account for the hops that include only U-nodes, while it could not take the BIONETS-non-compatible nodes into account. Moreover, in a limited copy spreading policy, the maximum number of copies defined by the algorithm might be violated if several copies of the encapsulated message (as defined by the algorithm applied in the BIONETS-non-compatible network) are delivered to the U-nodes layer.

4.5 Services and interoperability

From the service perspective, the requirement for interoperation with traditional networks makes the BIONETS communication a bit more challenging than the pure infrastructureless BIONETS network operation. When some information needs to be collected from BIONETS to some specific server in the Internet, the services are required to act such as no discontinuity was present. In this case, for example, a BIONETS source node would deliver a message to some specific destination, e.g. to some content storage/server in the legacy Internet, as in the case of B2IP communications. A similar situation may arise if a remote user over the Internet needs to interact with BIONETS services. Since such interoperation may be an essential feature for BIONETS services, here we draw some design considerations related to services and interoperability in presence of opportunistic routing.

The generic BIONETS service is assumed to include a BIONETS service framework for the BIONETS applications, see Fig. 5. The BIONETS communication platform is assumed to consist of several parallel means for communication, from which the BIONETS applications can select the most suitable for communication according to the specific communication needs of the application. Situated opportunistic communication can be one of them and enabling it requires a specific module for T-nodes , U-nodes and APs as seen in Fig. 5.

The delivery of a message from the source node to the access point (AP, gateway) is assumed to happen opportunistically. To this extent, the AP functions towards the BIONETS should be like those of a U-node. However, the AP needs a proxy, which is able to make the mapping of situated opportunistic communication B2IP over the traditional Internet means in such a way that the message eventually reaches its destination.

The opportunistic delivery of messages from a BIONETS node to an AP includes potential application of other BIONETS nodes for carrying the message within BIONETS. Such nodes may be specialized as data carriers, which are nodes having the capability to carry smaller or bigger amount of message data between isolated network islands/nodes. They can be more or less specialized U-nodes for carrying the messages or otherwise capable of working as a data carrier.

The destinations of BIONETS messages in the Internet can vary depending on the use case of

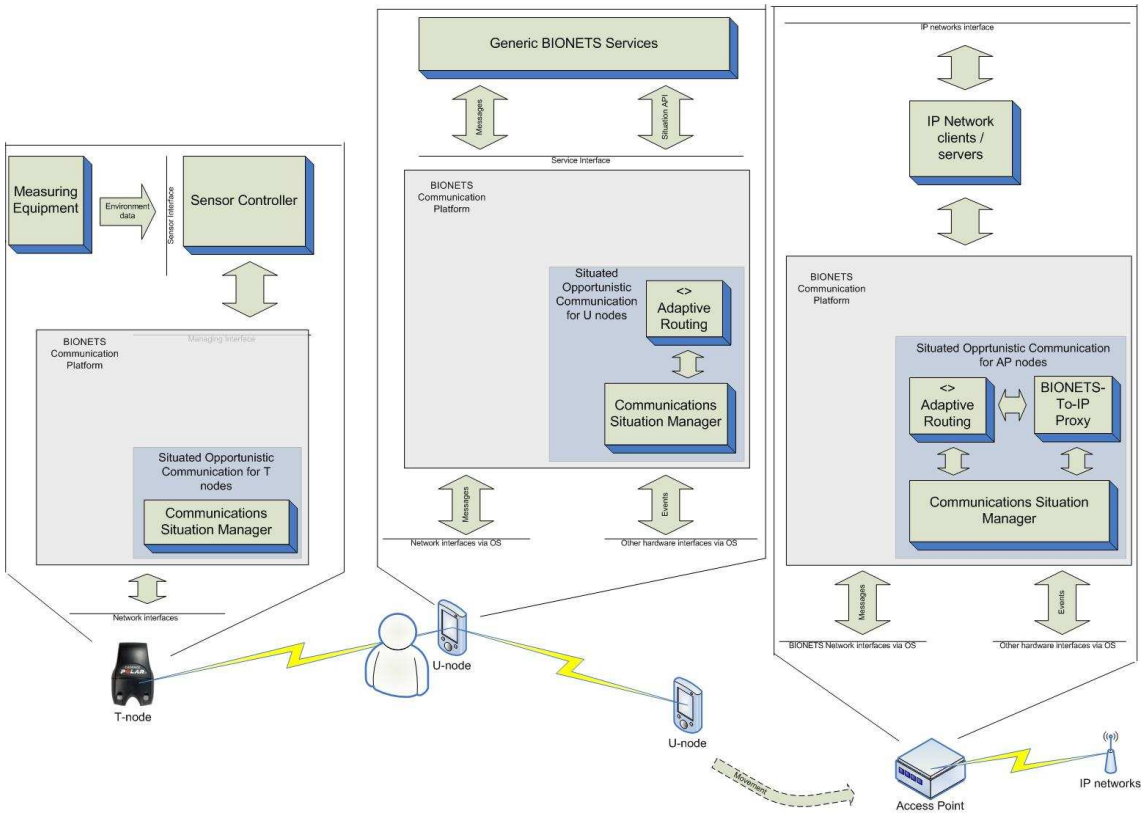

Figure 5: Interoperation of Situated Opportunistic Communication.

the APs and the service. If the Internet is used to enable BIONETS network traffic to overcome lack of communication possibilities between different BIONETS clusters or to possibly shorten the path between clusters, a tunneling solution is needed which would hide and store the BIONETS messages by encapsulating them into IP packets. This is the case of a B2IP2B communication. A different AP having access to some other cluster would then decapsulate the packets and insert them back to the BIONETS network. From the service perspective, information diffusion from BIONETS to the Internet could be enabled using already existing protocols available in the Internet. In this case, the AP would use the proxy functionality to convert the BIONETS formatted message into a traditional protocol used in the Internet. An existing client and server software could be utilized in the sending and receiving process of the proxy operation.

If this kind of proxy method would be inserted into APs, the U-nodes located in the BIONETS network do not need to know the protocols of the IP network as the AP proxy would handle the translation process. This would enable a service of the BIONETS to e.g. send an information content which then would be converted by the AP into SMTP protocol and eventually would be delivered as an email to selected recipients. A third use case for the AP could be enabling the proxy functionality to convert a BIONETS type of message into some new protocol running on top of IP. This in turn would require such a novel protocol to be implemented in the proxy functionality of the AP.

5 U-nodes readings of T-nodes

In this section we focus on the problem of a U-node retrieving local data from T-nodes in radio range. We assume that the U-node will typically obtain the value of some function $f(\cdot)$ of environmental data. This can be seen as a problem of *in-network* computing, which has been addressed recently in literature. In the following, we make first a brief overview of the current state of the art in the field, and we propose a possible solution that applies to T-node to U-node communications in the case of type-sensitive functions.

5.1 Related works

The problem of computing and estimation in wireless sensor networks is a central issue and it has been tackled from several perspectives in the recent networking literature. The rate at which estimates are generated by a sensor network are constrained to the *rate* at which we can sample the measured field *and* compute the target function of the measured values. In a distributed estimation scheme, moreover, it is possible that the values have to be made available at several sink nodes or perhaps at all nodes at once; in the limit case all nodes might posses such estimate. Traditionally, the typical requirement (and also the concern) for WSNs is to scale to a large number of sensor nodes. This is due to the fact that research borrowed a traditional thread of ad-hoc networks: the throughput that can be offered to each node is going to zero with the increase of the number of terminals. This way of looking at the problem restricts to the communication requirements, typically because it is assumed that all data are conveyed to the sink node and then processed there. Using this perspective, in fact, any scheme computing a function of the data will inherently have a maximum estimation rate which is inversely proportional to the many-to-one capacity of the underlying network.

In order to increase scalability, following the indications on the work of Gupta and Kumar [16], several schemes for distributed estimation are based on local communications, i.e., communications involving the neighborhood of a node only. For example, authors of [17] proved that the best linear unbiased estimation of a deterministic parameter can be computed at every sensor with a distributed algorithm. Similarly, the scheme from [18] produces an estimate of the average value of a random field at each sensor; average field measurement is performed by the distributed self-clocking scheme described in [19].

Due to architectural constraints [15,20], in the BIONETS network architecture we cannot resort to gossip propagation since sensors can overhear the channel but cannot transmit to each other[9]. A different approach, proposed in [21], uses a combination of a binary split-tree algorithm coupled to a binary hypothesis testing procedure. Joint MAC/PHY design is proposed in [22], proving an asymptotically optimal MAC for type based estimation.

In literature, the seminal paper in the field is indeed Gallagher's scheme [23], where, under the assumption of perfectly scheduled communication, the proposed solution would permit the parity check on the binary status of a set of nodes with required communication complexity $O(\log \log n)$.

[9] This of course does not exclude that a channel exist due to overhearing.

The later work in [24], proved that, in the case of type-threshold functions, such as AND, OR and MAJORITY, computing requires $O(n)$ broadcasts.

Recently, a radical approach has been addressed by the works in [25, 26]. There, authors provided fundamental scaling laws in the case of colocated and multi-hop packet networks. Also, the authors prove that *type-sensitive* functions are maximally difficult to be computed, since they require a communication complexity which is of the same order of magnitude of the whole data collection. In particular, it turns out that communication complexity in order to compute a type in a colocated network takes on the order of $\Theta(\frac{1}{n})$. In the case of BIONETS, this also indicates that any estimate produced by a U-node, which implies the computation of a type-sensitive function of the data, will require on the order of $\Theta(n)$ transmissions. In particular, the whole histogram of the data is a type-sensitive function, but also the *mean*. Conversely, the *max* and *min* are threshold-sensitive functions, for which a better bound exists, i.e. the communication complexity is $\Theta(\frac{1}{\log n})$. Thus, the work in [25, 26] proves that there exist a strong dependence on the scaling law of the number of messages exchanged and the computed function.

In what follows we then restrict to the the more costly case, i.e. the case of type-sensitive functions: in such case, in fact, the results in [25,26] indicate that, as far the scalability is concerned, the problem is actually equivalent to perform the complete data download. Thus the parameter of interest, in such cases, is the *makespan*, i.e. the time taken from the time 0, when data start to be collected to the time when the last packet is collected. We then describe two algorithms which scale as $\Theta(n)$ in the size of the batch of T-nodes.

We remark that, in the case of batch arrivals, at a first glance, one could think of employing a random backoff strategy, similar to what is done in current technologies such as IEEE802.11 or Zigbee. But, for batched arrivals, in which all n packets arrive at time 0, authors of [27] showed that randomized binary exponential backoff has makespan $\Theta(n \log n)$, and more generally, for any constant r, r-exponential backoff has makespan $\Theta(n \log \log rn)$. Quadratic backoff has makespan $((n/\log n)3/2)$, and more generally, for $r > 1$, r-polynomial backoff has makespan $\Theta((n/\log n)1 + 1/r)$. In general, thus, such algorithms are suboptimal in a logarithmic factor.

5.2 Linear Time Data Retrieval Schemes

The main problem of the data download is the fact that the number of nodes is unknown. Hence there are two possibilities. Either the U-node performs a preliminary stage to estimate the number of potential contending T-nodes, or a scheme independent of such estimate is employed. We indicate schemes for each case, which implement very simple algorithms which should be supported with minimum hardware requirement. In what follows, we consider a set of n T-nodes within the radio range of the sink U-node. In order to retrieve the values of each bean, the sink can perform a sequence of queries asking for a set of measurements in a given interval. When multiple sensors respond to the query, the sink will suffer a collision; we assume that when multiple packets are transmitted concurrently, the sink will not be able to decode them.

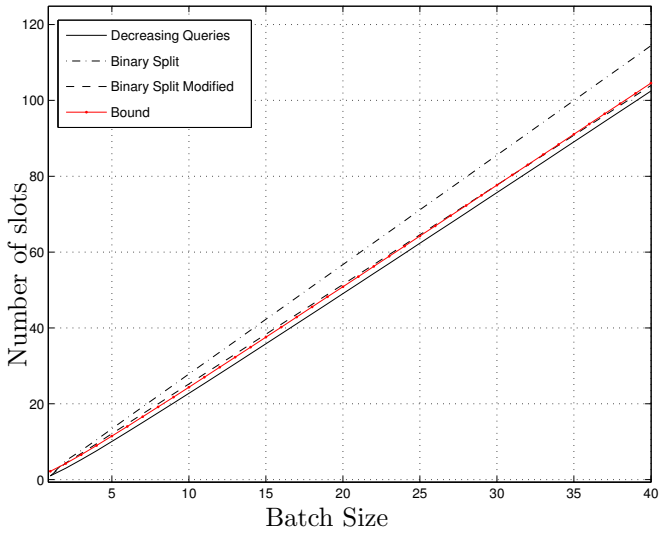

Figure 6: Number of slots for data download.

Data downloading without batch size estimation

A first approach for batch data download in linear time is provided by the binary split algorithm [21, 28], which provides a technique to retrieve all data in expected linear time. In addition to being asymptotically very efficient, the pure tree based splitting algorithms do not require apriori knowledge of the number of nodes. The modified version of the tree due to Tsybackov and Mosey [29, 30] provides an improved bound on the number of required steps in order for all terminals to transmit their data.

Data downloading with batch size estimation

With a different approach, the sink node can preliminarily perform an estimation on the number of the T-nodes in radio range, and then operate accordingly. In order to obtain such estimate, the sink can resolve a fraction of the sensor values as done in [21, 30]. One possibility is then to perform the hybrid technique proposed in [30]: using such a technique, the scaling in the number of nodes is asymptotically bounded by $2.05 \cdot n$, which, so far, is the best known bound for any collision resolution algorithm with homogeneous nodes; such a scheme is nevertheless computationally intensive at the T-node side. As in the case of the binary split tree algorithms, in fact, every node has to reconstruct the collision resolution process. In the limit case, some estimation is known apriori at the U-node side: this can be the case for example of persistent operations of a U-node in a given area. Hence, a much simpler scheme can be used, and for small values of the number of T-nodes, it still performs in a satisfactory manner. In practice, after estimating of the number of nodes n, the sink will broadcast such an estimate. Each of them will then transmit with probability $1/n$ and, in case of success, the sink U-node will ask to transmit with probability $1/(n-1)$, and so on, until all data are collected. The expected time required to retrieve each T-nodes data is then the sum expectations of independent geometric random variables

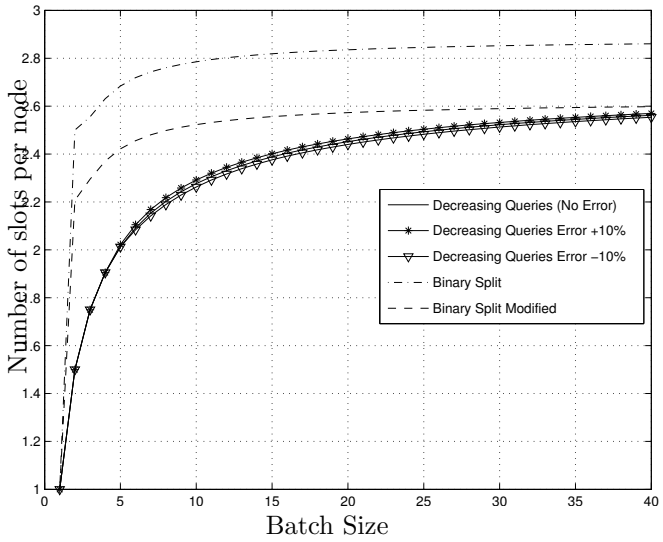

Figure 7: Slots per node in presence of errors in the estimate of the batch size.

in the form

$$L = \sum_{h=1}^{n} \left(1 - \frac{1}{h}\right)^{-(h-1)} . \tag{2}$$

where we neglected the cost due to the batch size estimation, since in case several estimations are performed in the same area, this term would be averaged over several intervals. The expression of (2) shows the linear scaling property of the sequential query strategy described before. In fact, the cost of the data download performed by such a policy is $\Theta(n)$ and for colocated networks [25, 26], it achieves linearity in the number of nodes.

5.3 Comparison

It is known that using Bernoulli trials is sub-optimal compared to the binary split tree [28]. Also, from (2), $L \leq e \cdot n - \log n - H$, where $H = 0.511$, whereas for the modified binary split tree a much better bound is known since it achieves a scaling law of the form $L = \alpha n - 1$ for large n, with $\alpha \leq 2.3175$ [31]. The advantage of the sequential query strategy for small batch sizes, anyhow, is expected since it is known that for small size of the batches the pure split tree is not efficient.

As shown in Fig 6, for a small size of the number of T-nodes, the sequential query provides a satisfactory behavior, performing even better of the modified binary split tree. Of course, this is indeed due to the fact that the sequential query procedure assumes perfect estimation. Thus, there exist an issue of robustness, since when the estimation in the number of contending T-nodes will be affected by an estimation error. In Fig 7 we can see that, nevertheless, up to a 10% error in the estimation of the batch size, the scheme still posess a satisfactory behavior.

6 Evolution of Forwarding Protocols

6.1 Introduction

In this section, we introduce a framework which allows forwarding schemes to evolve in order to adapt to changing and a priori unknown environments. The framework is inspired by genetic algorithms: at each node a genotype describes the forwarding scheme used, a selection process fosters the diffusion of the fittest genotypes in the system and new genotypes are created by combining existing ones or applying random changes. More details are presented in [32].

6.2 Background and Motivation

BIONETS relies on epidemic-style forwarding for disseminating information on the U-Nodes plane [33, 34]. An unconstrained epidemic forwarding scheme (in which an infected node spreads the epidemic to all nodes it encounters) is able to achieve minimum delivery delay at the expense of an increased use of resources such as buffer space, bandwidth, and transmission power. Variations of epidemic forwarding have been recently proposed in order to exploit the trade-off between delivery delay and resource consumption. This family includes, among the others, K-hop schemes [35], K-copy techniques [36], probabilistic forwarding [37, 38], and spray-and-wait [39, 40]. These schemes differ in their "infection process", i.e., the spreading of a message in network. They need to be combined with a "recovery process" that deletes copies of a message at infected nodes, following the successful delivery of the message to the destination. Various recovery schemes have been proposed: some are simply based on timers, others actively spread in the network the information that a copy has been delivered to the destination, using so-called antipackets [38].

Depending on the specific application scenario, different performance metrics could be envisaged. These include, e.g., the probability to successfully deliver a message to the destination, the delivery time, the total energy consumption in the system or a combination of these. For a given optimization goal the choice of a specific forwarding scheme and the configuration of its parameters depend in general on the number of nodes in the system, on their mobility patterns and on the traffic generated in the networks [41]. In many scenarios, these characteristics cannot be known at system design and deployment time and can also significantly change across time and space. For example, let us consider a Personal Digital Assistant carried by a user in its daily activities. During the day the node can travel at different speeds (e.g. from zero up to car speed), moving from highly crowded areas (supermarkets, classrooms,...) to less crowded ones, with very different trajectories (straight along a highway or following a random walk from shop to shop) and different levels of power availability.

In order to deal with these issues, various adaptive techniques for message forwarding can be envisaged. This approach is limited in that it requires an *a priori* definition of the actions to be taken to optimize the mechanism for some specific situation. The approach we propose is different. We want to embed the ability to evolve *autonomously* in the forwarding service itself. This is achieved by using concepts and tools from the Genetic Algorithms (GAs) field [42]. Each node employs a (potentially different) forwarding policy, which prescribes the operations to be

undertaken when receiving a message destined to another node. Such policy is described by an array of parameters called the genotype. Genotypes are associated with a fitness measure which, roughly speaking, indicates the ability of the current set of parameters to achieve good performance in the current environment. Fitness is evaluated using local information and rewards which are sent from the destination backwards within ACK messages, which act also as antipackets. When two nodes meet, they may exchange genotypes (and associated fitness levels), updating the pools they maintain. Each node periodically generates a new genotype judiciously using those in its pool and implements the corresponding policy. The whole system is engineered in such a way as to present a drift towards higher fitness levels.

This is reminiscent of some earlier works carried out by two of the authors [43], presenting a framework for the application of GA-like techniques to BIONETS, from which it differs for two main reasons. First, in [43], the generation of new offsprings was tightly coupled with the opportunistic contact process, in that new service representations were generated upon meetings with other nodes. Here we exploit contacts among nodes only to spread knowledge about other genotypes and their performance; each node maintains a pool of possible genotypes and an associated estimated fitness value which is used for periodical generation of new genotypes. The generation of new offsprings is therefore not coupled with the contact process. Second, in this case message forwarding is coupled with the evolutionary process, in the sense that message forwarding is done according to the policy specified by the genotype which is subject to evolution. This implies some form of bias in the system: for example, a genotype specifying a low forwarding probability will take part in a limited set of message delivery processes, and will therefore be slow in collecting rewards and getting a fitness estimate.

6.3 A Framework for Evolution of Forwarding Services

We first observe that multiple forwarding schemes can co-exist at the same time in the network. In fact this form of information delivery, based on the presence of multiple copies in the network, does not need nodes to be compliant with a specific behavior, enhancing system robustness with respect to conventional schemes. This flexibility comes from the completely distributed nature of the forwarding process in epidemic-style relaying, which allows nodes to use different policies in an uncoordinated fashion.

In our framework, each node can apply a distinct forwarding policy to relay messages to other nodes. We want nodes to *learn* online what is the best forwarding policy (or what are good policies) in the current scenario and change consequently the one they employ. We note that a node cannot evaluate by itself whether its current policy fits the current scenario, because it is in general not aware of the consequences of its actions. For example a given node can never know by itself whether its decisions – according to its forwarding policy – to relay or not to relay a message were the right ones or not. Thus, a node may be relaying a message when the latter has already been delivered to its destination, hence wasting resources. On the other hand, a node may refrain from relaying a message when it happens to be the key node in the message delivery process, e.g., if it is the only node traveling between two disconnected clusters of nodes in the network. It should be

clear therefore that only other nodes in the system can evaluate the fitness of a node's forwarding policy. We also observe that the goodness (or fitness) of a node's policy depends on the policies implemented by the other nodes as well. Message delivery is in fact a collaborative process, whose performance depends on the behaviors of all nodes, so that a specific policy can be beneficial or detrimental depending on other nodes' actions.

The above considerations imply the need of an online distributed fitness evaluation process and raises an issue about the use of the fitness in the evolution process. Once a node get marks for its own policy, how should these be used in order to change the policy? We opt for a blind evolutionary approach, which relies on a homogeneity assumption among nodes in the system.

A first step towards an evolutionary forwarding service consists in a formal representation of a generic forwarding scheme. Such descriptions represent the *genotypes* of the implemented forwarding schemes.

A forwarding policy consists of a set of actions to undertake upon message reception. It defines what nodes do when they come within mutual transmission range. The actions can be specified using parameters and may rely on information contained in message headers (like message generation time or count of hops the message has traversed). For instance, a node can transmit a message with probability P as long as the message has not been forwarded more than H times. The values of the two parameters uniquely specify one forwarding policy. A simple binary string can be used to represent the policy as illustrated in Fig. 8, where the genotype spans 12 bits. The 6 leftmost bits represent the numerator a of the fraction $a/63$, which is the forwarding probability value P, while the 6 rightmost bits represent the maximum number of hops H. Hence, out of such representation it is possible to reconstruct the corresponding forwarding policy.

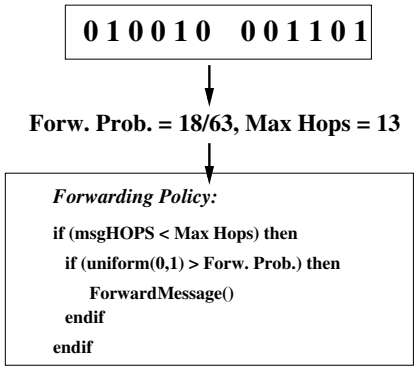

Figure 8: Example of forwarding policy mapping: from the binary representation to the forwarding algorithm.

The behavior of a generic forwarding scheme can thus be changed by tuning the values of the parameters that specify the actions to be undertaken.

The natural selection process promotes the diffusion of organisms presenting a high fitness level. In much the same way, we want to engineer mechanisms for promoting the diffusion of "good" genotypes already present in the population. Those would be the genotypes yielding good performance for the optimization goal. At the same time, new genotypes can be generated in order

to explore new possible solutions. In this section, we focus on the first issue letting the next section discuss how new genotypes can be generated from existing ones.

In traditional GAs, *reproduction* is a process which selects existing genotypes to create new offsprings. At discrete time intervals, genotypes are randomly selected from the population to generate offsprings: each genotype is selected with probability proportional to its fitness, namely. This procedure tends by itself to increase the average fitness of the population. However, in the considered mobile network scenario the different genotypes are distributed over all the nodes of the network. In order to overcome this limitation, we assume that upon meeting nodes exchange information about their respective genotypes and the corresponding fitness indexes. As depicted in Fig. 9, in our system each node maintains a pool of available genotypes (including the one currently in use) and their corresponding fitness values. At regular time intervals, each node goes into a reproduction phase, running the selection process on the genotypes currently stored in its own pool.

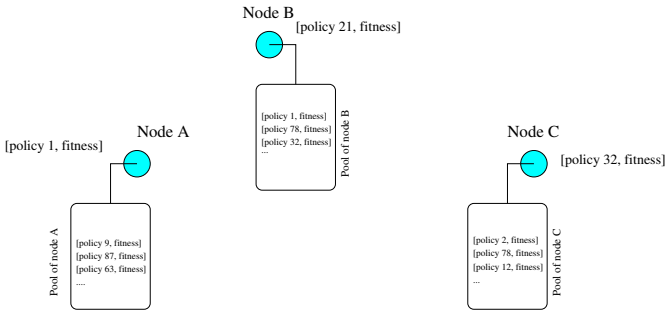

Figure 9: Considered system architecture: each node employs a policy, characterized by a genotype with associated fitness value. Each node maintains a pool of candidate solutions used in the reproduction phase to generate new genotypes.

The fitness of a given node is in this case an estimation of its contribution to successful spreading of information [32]. The process of evaluating such fitness can be triggered by the destination node of a message as it is the best entitled to evaluate the outcome of the infection process. The destination node propagates then the evaluation of the performance of the spreading process or at least information needed for this evaluation to all nodes involved in the infection process. This feedback can be seen as a "reward" to those nodes. The communication cost can become significant so that a communication-cost versus information-accuracy trade-off arises.

New forwarding policies are generated applying GA-like operators to the genotypes maintained by a node in its pool. The two following operators can be used to create new genotypes from existing ones. *Crossing-over* consists in breaking two genotypes at a randomly chosen position and exchanging the tails of the genotypes. Two offsprings, called *crossovers*, are produced, and one is selected at random. *The fitness indices of the crossovers are a priori unknown. Another option that favors genotypes with high fitness consists in selecting the crossover that is closest to the fitter parent. Mutation* consists in a random change occurring in the genotypes. As an example, mutation can be implemented by randomly swapping, with some probability, the bits of a binary representation of the genotype. And, to ensure stability, mutation should occur with

small probability.

6.4 Mechanism Specifications

We consider a simple fixed-length genotype comprising one parameter, which is the probability P to copy a message upon encountering a new node. Let W denote the set of nodes which contributed to the delivery of the first copy to reach the destination. As optimization goal we consider the minimization of the expectation of the weighted sum of the delivery time, T_D, and the number of message copies made by the nodes in W. The cost function is then

$$F = \mathbb{E}\left[T_D + \gamma \sum_{i \in W} C_i\right] \quad , \tag{3}$$

where $\mathbb{E}[X]$ represents the expectation of a random variable X and γ is a parameter which can be understood as the time-equivalent cost of a copy.

Should the optimization goal be to minimize solely the expected delivery time, then the evolutionary forwarding scheme will trivially converge, in a underloaded network – i.e., when traffic is small in comparison to the available capacity – to standard epidemic routing, where messages are flooded in the entire network. Conversely, the presence of the number of copies in the cost function makes also an underloaded network (a realistic case and faster to simulate) an interesting scenario to study. The evolutionary forwarding scheme will limit the number of copies depending on the value of the parameter γ. Taking into account the number of copies is also meaningful because they are strongly related to bandwidth usage and power consumption. A more natural choice would be to consider the total number of copies done in the network (i.e. $\sum_{i=1}^{N} C_i$), but a heavier communication overhead would be required. In our case-study we opt for light signaling.

We assume that all nodes are synchronized and that the message header contains a field specifying the time at which the payload was generated at the source. In such a way the destination can evaluate the delivery time as soon as it receives the first copy of a message. On the other hand, each node knows the number of times it has copied a message, but not the delivery time. We assume that each node, before forwarding a copy of a message, adds its own identifier to the message header (this is analogous to what is done in source routing in mobile ad hoc networks). The set W is nothing but the set node IDs present in the header of the first copy reaching the destination. The destination node sends to nodes in W a new acknowledgment (ACK) message. This message specifies the delivery delay and the number of hops ($h = |W|$) traveled by the message before reaching the intended destination. In this way each node i along the path of the first copy delivered has the following information available: the delivery time, the number of hops in the path and the number of copies it did C_i, the latter value being computed locally at each node. We assume that the node evaluates its reward as a decreasing function of $T_D/h + \gamma C_i$. The intuition behind this choice is that in this way the system is minimizing:

$$\mathbb{E}\left[\sum_{i \in W} (T_D/h + \gamma C_i)\right] = \mathbb{E}\left[T_D + \gamma \sum_{i \in W} C_i\right],$$

which is exactly the cost function given in (3). In fact, by selecting genotypes with higher fitness level, nodes are implicitly selecting genotypes yielding smaller $T_D/h + \gamma C_i$. In a homogeneous

setting we can expect that the minimization of each of these terms leads also to the minimization of their sum. We note that considering the total number of copies would have required each node to be informed about the number of copies done by every other node in the system.

We consider the following expression for the reward at node i:

$$r_i = \max\left\{1 - \frac{T_D/h + \gamma C_i}{R}, 0\right\} \tag{4}$$

where R is set to a high enough value, for instance $2\mathbb{E}[T_D] + \gamma N$.

Upon receiving the n-th ACK message and computing the reward $r_i(n)$ according to Eq. 4, node i updates its estimation of the genotype fitness as follows:

$$\hat{\phi}_i(n) = \frac{n-1}{n}\hat{\phi}_i(n-1) + \frac{1}{n}r_i(n), \tag{5}$$

which corresponds to averaging all rewards received.

When two nodes meet, they transmit to each other their own genotype and its current fitness level estimation. Each node maintains a pool of available genotypes (including the one currently in use) and their fitness. We use G_j to denote the pool at node j and $\hat{\phi}_{i,j}$ to denote the value of node i fitness known by node j ($\hat{\phi}_{i,j}$ is the value of $\hat{\phi}_i$ at the time of the last meeting between node i and node j, so at a given time instant these two values can be different). We consider a synchronized reproduction phase. Every T_g seconds, the *generation lifetime*, nodes synchronously create a new offspring each, i.e., they update their own genotype. This synchronism allows to clearly identify different generations during the evolution. Crossing-over is performed with probability p_c and requires to select two genotypes from the pool, otherwise only one genotype has to be selected. At each node, e.g., at node j, genotypes are selected with a probability proportional to their own fitness, namely, $p_{i,j} = \hat{\phi}_{i,j}/(\sum_l \hat{\phi}_{l,j})$ where the sum is over all genotypes contained in the pool G_i.

Finally each bit of the current genotype, directly selected from the pool or obtained through a crossing-over of two genotypes, can mutate with probability p_m. The genotype pool is emptied after every generation, and the fitness value of the genotype that will be used is set to zero. If the network is large, the node's pool may not be large enough to keep all genotypes discovered in a generation. Should this be the case, a node may keep only the fittest genotypes, or alternatively select the genotypes according to a fitness-biased distribution.

Two nodes are able to exchange messages when they get within mutual communication range. Once it happens, they perform the following steps:

1. exchange node IDs;
2. exchange header information of data messages;
3. each node decides which messages should be forwarded to the other node;
4. messages are exchanged;
5. each node can drop some messages from its buffer (if full) in order to free space for new messages.

Algorithm 1 Algorithm performed by a node upon reception of an ACK message.

```
1: Add the received ACK message to the internal ACK messages list
2: if msgID ∈ {msgID 1, ..., msgID L} then
3:    Remove the corresponding DATA message from the internal structure.
4:    if Node ID ∈ W then
5:       Update node's fitness value (REWARDING).
6:    end if
7: end if
```

The evolving protocol makes use of two types of messages to be exchanged over the network: DATA messages and ACK messages. DATA messages are those carrying the payload transmitted by any mobile node to a specific destination, whereas ACK messages are used for the following purposes:

- to acknowledge the successful delivery of the message at its intended destination;
- to feed back the reward to the nodes along the successful path from source to destination (rewarding);
- to serve as anti-DATA, by blocking the diffusion of already delivered messages and removing them from nodes buffer.

The fields common to all message headers are (i) [message ID], which is the (unique) identifier (ID) of each message and also specifies whether it is a DATA or an ACK message (ii) [GenTime], which describes the time at which the message has been generated.

Further, data messages include a hop-count field [hops] and the labels of all nodes which forwarded the message along the path from the source to the actual node. ACK messages, on the other hand, include the complete set of nodes involved in the forwarding path and a field, called [Feedback], containing the value of the rewarding metric T_d/h, where T_d is the DATA message delivery time, and h is the value of the field [hops] when the DATA message is received (i.e., the length of the source-to-destination path).

Each mobile node maintains two internal data structures dedicated to the storage of DATA and ACK messages respectively. In the structure storing DATA messages, each item additionally stores a counter of the number of copies of that message already disseminated in the network.

For any DATA message to be relayed, a node first adds its own node ID to the header and then increments by one the [hops] field of the message. The message is kept in the node's internal memory until the corresponding ACK message is received.

In addition to DATA messages, mobile nodes diffuse also ACK messages. Unlike the case of DATA messages, no limiting policy is applied to the forwarding of these messages (ACK messages are simply *flooded* into the network according to the VACCINE recovery scheme [44]).

Whenever it receives an ACK message, a node first adds the received ACK message to the internal message list. It then checks whether the corresponding DATA message is present in its internal memory and, in case, removes it. If the corresponding DATA message was present and the node has contributed to the successful path to the destination (Node ID $\in W = \{$Node ID

$1, \ldots,$Node ID $M\}$), it applies the proper rewarding scheme to update its own fitness, as described in the previous section.

The overall procedure is summarized in Algorithm 1.

7 Computing perspective on protocol evolution

In the context of BIONETS evolution must happen as an online activity, helping in the self-optimization of the system during its operation, in order to increase its autonomicity.

With this motivation in mind, we are working on a framework for the resilient online evolution of protocols and services. The outline of the framework was presented in Deliverable D2.2.2 [45]. It includes a chemical computing model of evolution which aims at achieving resilience via the control of the concentrations of instructions or programs in the system.

We adopt the fraglets language as a starting point. This language was designed to evolve communication protocols automatically, however actual evolution results have only recently been shown. We refer to Section 5.2.4 of Deliverable D2.2.2 [45] for an introduction to the fraglets programming language and instruction set, together with some examples of protocol implementations.

In this section we summarize the current state of our research in what concerns network protocol evolution with fraglets. Currently most of the work has focused on understanding and implementing evolution mechanisms in fraglets, and in extending the language to support more generic computations required by the higher-level services envisaged in SP3. Actually very little has been done on protocol evolution, since many obstacles and research challenges were identified when extending fraglets for doing complete program synthesis from scratch. We have thus proposed to backtrack from specific protocol issues, in order to focus on the language and genetic operation issues that must be solved with priority, before being able to tackle protocols.

7.1 Code Regulation Experiments

In Section 5.2.5 of D2.2.2 [45] we introduce the notion of "autocatalytic software", in which programs are modeled as molecules which regulate their own production and consumption, leading to a system where instructions as well as whole programs are measured in terms of their "concentration" in a multiset structure. The model is inspired by gene expression in biology: genes encode for proteins that actually perform various cell functions; some genes encode proteins that will later act as activators or inhibitors for other genes. Environment conditions may act on the concentration of proteins, determining which genes are expressed at which concentrations. Such gene regulatory networks contain feedback loops that result in the observed phenotype for an individual.

The proposed model for autocatalytic software execution includes a code repository where "genes" are stored, and expressed upon reception of expression signals targeted at a given gene. The expressed gene is injected into an execution environment where it becomes the program that is executed, and whose byproducts may include activation or inhibition signals for the same or other genes in the repository.

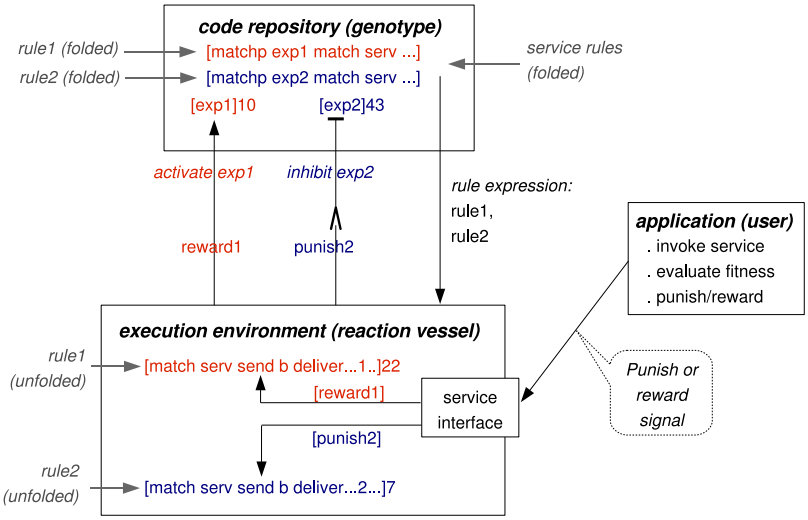

Figure 10: Code Regulation Scheme

Some first experiments with such code regulation model were presented in [46]. There the program is executed and evaluated by an application. The resulting fitness value is used to generate activation (reward) or inhibition (punish) signals that are then used to re-express the gene (if high fitness) or to inhibit it (if low fitness). We show how the concentration of high-fitness programs increases while that of low-fitness programs decreases sharply. This provides a simple method to promote the good programs and eliminate the harmful ones from the system.

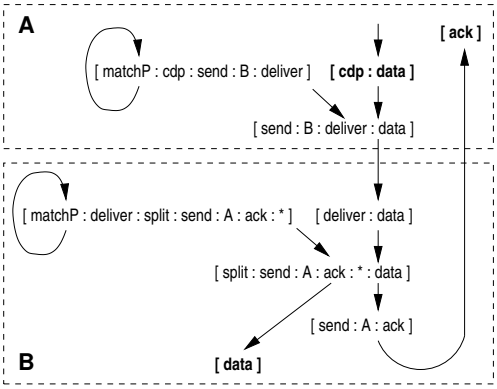

Figure 11: CDP (Confirmed Delivery Protocol) runtime diagram (from [47])

source:
$_A[matchp : cdp : send : B : deliver]$

sink:
$_B[matchp : ack : send : A : deliver : ack]$

Figure 12: CDP implementation in fraglets

The experiments have used the CDP program (see Section 5.2.4 of D2.2.2) which implements a confirmed delivery protocol. The application rewards each instance of CDP for each correctly delivered packet, and punishes it for lost, corrupted or duplicated packets, for false acks and other incorrect behaviour.

The code regulation scheme is depicted in Fig. 10, and a sample of the outcome of the experiments is shown in Fig. 14.

The experiments used the original CDP implementation from [47]. The code regulation mechanism was applied to select CDP variants.

Fig. 12 shows the fraglet code for CDP, both sender and receiver sides. When presented with an application payload of the form ${}_A[cdp : payload]$, the first *matchp* rule at the source is activated, producing a rule that sends to B a fraglet carrying the payload. The sink side then injects a $[ack]$ fraglet that travels back to the source. This is depicted in the runtime diagram of Fig. 11.

$${}_A[matchp : cdp : send : B : deliver]$$
$${}_A[matchp : cdp : nul]$$

Figure 13: Half-bad CDP variant

Figures 11-12 show a well-behaving variant. Other not so well-behaving variants could easily appear by evolution, or be injected into the system from the outside. Figure 13 shows an example of malicious variant that transmits half of the packets correctly, and discards the other half. The system should be able to detect and eliminate such cases, while mitigating their disruptive effect in the system.

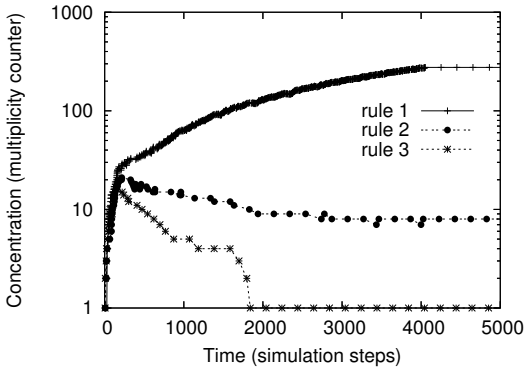

Figure 14: Concentration of rules: rule1=good; rule2=neutral; rule3=bad

Fig. 14 shows the concentrations of three types of programs initially present in the repository: Rule 1 corresponds to a good program as in Fig. 12. Rule 2 shows a "half-bad" program which behaves well only 50% of the time, equivalent to the example in 13. Rule 3 represents a program that always discards packets, therefore is always bad.

These results show that well-performing services are rewarded and can increase their relative code expression rate to the detriment of the others.

7.2 Lessons Learned and Next Steps

The experiments showed that it is possible to control code expression using the proposed regulation mechanism, but many issues remain to be solved. First of all, one must show evolution in this context, with genetic operators that produce viable individuals with high probability. Even if the

system is able to eliminate unsuitable code, it should not spend most of its time doing so. Therefore the fraction of harmful mutations and other code disruptions should be kept to a minimum. We are looking at indirect encodings in Genetic Programming such as Cartesian GP [48], Grammatical Evolution [49], and Artificial Embryogenies [50] as potential directions towards a solution.

Ultimately, we would like to observe the emergence of resilient individuals from the evolution process, instead of pre-programming resilience into them. This is in theory possible due to the chemical programming model employed, which supports parallel or alternative execution paths. We are now implementing an experimental set-up to test this hypothesis.

Research on solving these problems is taking place in SP2 and SP3, and will be mapped to SP1 in a later stage, when it is sufficiently mature.

8 Disappearing network middleware

One of the main characteristics of BIONETS environment is the lack of global connectivity among the nodes. In fact, as described in [20], the communication among U-Nodes is based on opportunistic localized peer-to-peer interactions, as opposed to the end-to-end semantics of standard Internet protocols. U-Nodes communicate among them (by means of single-hop broadcast messages) just when they are in mutual communication range, and when the service running on the user device requires the interactions. U-Nodes within mutual communication range will form an island of nodes. U-nodes in the same island do not have a stable end-to-end connectivity, and, due to their mobility, they could freely leave an island and join another one: the mobility of U-nodes contributes to establish a loose communication among islands, based on the propagation of the information stored in the U-nodes (according to a store-carry-and-forward approach). Several protocols/algorithms were investigated for the propagation of information among the U-nodes [14,51] such as flooding through broadcast strategies, IOBIO (with/without addressing), two-hop relaying algorithm. Such protocols are the ones which provide a basic connectivity among U-nodes. The same protocols could be used to implement message exchanges between a source and a destination U-nodes (or better, between services/service cells executed by them). This section investigates the introduction of a network middleware to create a structured virtual overlay network to decouple the service framework and the characteristics of the underlying communication infrastructure for U-nodes interactions and the corresponding BIONETS communication protocols at network layer. In particular, the network middleware would provide a set of features to support the interaction framework. Moreover, such a middleware would provide to the service framework a set of features which enriches the characteristics/properties of the communication protocols. In the first phase of the activity, whose results are reported in this deliverable, some preliminary assumptions were considered in order to focus the investigation:

- the features and characteristics of the network communication protocols are abstracted as a set of primitives (e.g., a set of API);
- the (distributed) execution of services (supported by the capabilities of the middleware) involving (service components deployed on) multiple co-operating U-nodes would require an availability of the communications (based on BIONETS communication protocols) among the involved U-nodes (even if they can dynamically joint/leave the service).

8.1 Architectural vision and requirements

The NW middleware consists of a set of functions positioned among the NW Communication Primitives provided by the BIONETS networks and the service framework, in order to enable the service interaction framework identified in [52]. According to Fig.15, such middleware functions would extend the BIONETS network with features aiming at simplifying the implementation of the models envisaged for the interaction framework. Two types of interfaces should be envisioned:

- southbound interfaces, towards the network: they would provide an abstraction of the network protocol primitives, e.g., for point-to-point or point-to-multipoint communications;

these interfaces would hide the details of the algorithms/protocols used the network level to distribute information and messages (e.g., IOBIO, two-hop relaying algorithm, flooding);

- northbound interfaces, towards the service framework: they would provide to the service framework the functions realized by the middleware, that could be used to support the service framework interaction models.

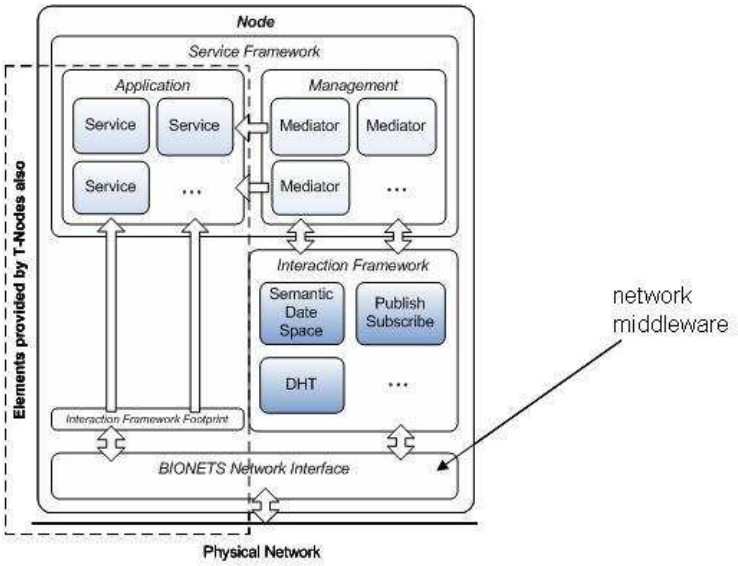

Figure 15: Network Middleware positioning in the BIONETS Service Architecture

The main motivations to introduce such functions are related to the need to:

- decouple the service and interaction frameworks from the underlying communication protocols for U-nodes interactions;
- improve the communications among the U-nodes (and the elements of the service framework deployed/running on them);
- introduce higher level features to the communication, such as enabling of new naming/addressing models for entities in the service framework (e.g., identification of services by properties, handling of information in the semantic data space), support in service self-aggregation, handling of data replications for reducing the effects of U-nodes disappearing from an island, etc.

The network middleware would be implemented as a peer-to-peer overlay, which implements a virtual network of peers and links; the functions of overlay peers are "hosted" on the U-nodes, while the virtual links are based on the underlying communication primitives. Disconnected islands of U-nodes generate separated overlays. The peers are containers for retrieving (in a broad sense) the entities of service framework, such as services, service descriptions, data/information, etc.

In particular, peers are in charge to provide capabilities for:

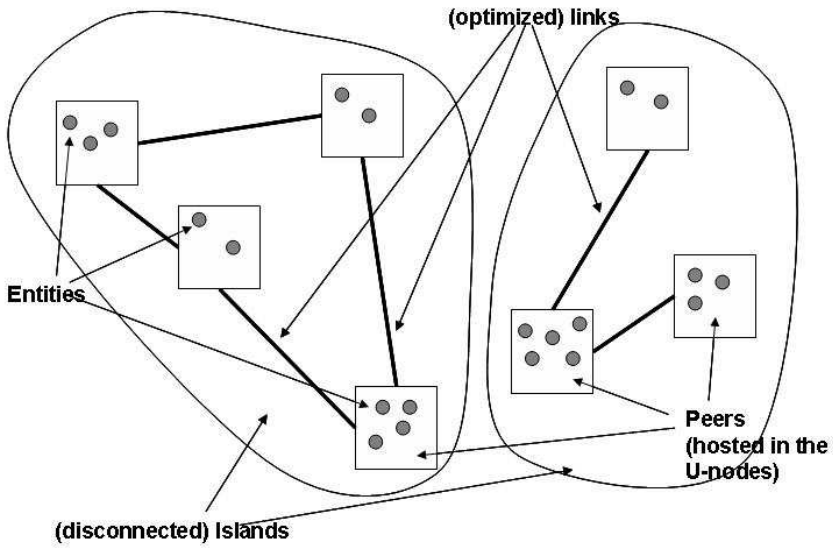

Figure 16: Components of the overlay network

- managing entities of the service framework, by providing support to store/get/put/filter/ address them;
- routing (through the virtual links) messages among the peers (by possibly applying some optimized strategy), and the entities handled by them;
- creating, optimizing and maintaining the overlay (e.g., the links).

The main requirements to consider in designing the overlay are:

- the overlay be able to handle peers and clients extremely transients;
- the overlay must be able to deal with the joining/leaving of peers with an high churn rate;
- the overlay must be able to deal with the joining of peers, which are already peers of another overlay (e.g., joining of islands);
- the overlay must be able to deal with its split into two (or more) parts among which there is not connectivity (e.g., creation of islands);
- the overlay must support the retrieval of service framework entities (e.g., data, documents, services/service descriptions):
  - publishing/storing of entities,
  - search/discovery of published entities;
- the overlay must support the distributed execution of services:
  - routing of messages for self-aggregation,
  - routing of messages to/from published entities,

  - providing addressing models;

- the overlay has to provide mechanism to recover from failure (e.g., due to lack of connectivity, peer churn);
- the overlay has to provide mechanism to optimize the routing of messages among entities which have to be aggregated;
- the overlay must identify/search entity through filters and queries (e.g., keyword search, and not only through keys);
- the overlay must be independent from requirements of specific application scenarios;
- the overlay must provide features to ease models of the interaction framework, including:
  - query/response for data/information retrieval,
  - publish-subscribe,
  - self-*capabilities at the service (cell) level (e.g., self-aggregation, self-adaptation),
  - feeding/maintaining of Data Space/Blackboard, possibly supporting efficient semantic capabilities;
- if AP-node are available, the overlay should be able to interwork with overlays deployed/running on the Internet.

Due to characteristics of connectivity in BIONETS, the overlay could fail in retrieving entities published in the overlay or in routing messages to them. Moreover, there could be limitations in handling modification/deletion of the entities stored in the overlay (in particular, in the case of replications to optimization reasons). Such possibility should be taken into account by the services. In particular, the overlay should not be intended as a distributed implementation of a traditional data base, which has to fulfill consistency requirements, e.g., exact query resolution, ACID transactions, etc., but it should be considered as a mechanism for distributing and sharing information in a highly dynamic context.

8.2 Description of the overlay functions

The overlay consists of a set of functions for efficiently retrieving (e.g., search, discover, address) entities distributed on the U-nodes (e.g., data, information, services). The entities stored in a U-node should be accessible from all the U-nodes belonging to the same island. The overlay implements features to optimize the interactions among the peers (e.g., to improve the search or the message routing). The overlay provides to the elements of the service framework the following set of (abstract) operations:

- publish(property, data): a request to store an entity in (a peer of) the overlay:
  - property: it is a set of information through which it is possible to select an entity published in the overlay (e.g., a key, a set of keywords, a property list, a semantic description, a service description),

  - data: it is the information associated the stored entity (e.g., a file, a document, the reference to a service/object or the service/object itself),
  - the overlay should organize itself in order to improve the retrieval of the published entity;

- deprecate(property): a request to deprecate an entity published in (a peer of) the overlay;
  - property: it is a set of information through which identify the entity published in the overlay to be deprecated,
  - the overlay should organize itself in order to delete from the overlay the deprecated entity;

- search(query): a request to get the data associated to an entity published in the overlay;
  - query: it is a "filter" on the properties associated to the entities stored in the overlay; it is used to identify the entities to retrieve,
  - the overlay returns (a copy of) the data associated entities which fulfill the query; it should performs optimized interactions among the peers;

- send(query, message): a request to send a message to an entity (e.g., an object, a service, an object reference) stored in the overlay:
  - query: it is a "filter" on the properties associated to the entities stored in the overlay; it is used to identify the entities which are the destination of the message,
  - message: it is the message to be provided to the selected entities (e.g., a method to be executed, a self-* message),
  - the overlay has to provide an optimized routing to find the entities and to delivery the message; it may returns a result to the requestor (e.g., an ack on message delivery, the results of the elaboration of the message);

- aggregate(query): a request to optimize the routing to reach (the peer storing) an entity (e.g., an object, a service, an object reference) in the overlay;
  - query: it is a "filter" on the properties associated to the entities in the overlay; it is used to identify the entities for which it is required to optimize the routing,
  - the overlay has to optimize the routing of the messages from the requesting peer to the selected entities, in order to improve the following interactions (e.g., send operations); this request would optimize the overlay in order to improve the interactions in the contexts of an aggregation of entities;

The check to verify whether the property associated to a published entity matches with a query is performed by the peers storing the published entities. Such a match could be performed by considering either syntactic or semantics criteria. Additional operations could be related to modify/update and/or delete published entities. Due to the characteristics of connectivity in BIONETS

these operations should be performed carefully, in order to avoid inconsistency in the entities handled by the overlay. A possibility could be to introduce some time validity for the published entities. These operations would support the interactions of the service framework. For instance, advertising of service entities (i.e., the advertisement of the features/functions/goals of a service) can be performed by publishing in the overlay an entity with the following information:

- properties: they may include, according to the different models, the description of the service interfaces (e.g., a formal description of the provided and/or required functions), its non-functional characteristics, identification labels, etc.;
- data: the service itself, or a reference to interact with the service.

Moreover, the overlay includes a set of functions for creating, using, optimizing and maintaining the links for achieving the interconnection and information exchange among the peers, i.e., the U-nodes hosting the peers. The overlay establishes the links, by relying on the BIONETS communication primitives. The algorithms that govern the interconnection links in the overlay must be able to cope with a high churn rate of the peers involved in the overlay, and possibly in the creation of islands and the merging of two islands. Such algorithms, executed by the peers, must perform the following actions:

- request to join an overlay, by contacting the peers hosted by the near U-nodes;
  - an initial set of link will be established, and information on the published entities will be exchanged,
  - in order to improve the stability of the overlay, a passive join approach could be adopted (see [53] for an example), where initially the new node will act only as client of the overlay and not a member of it;
- routing of the messages (for executing the overlay operations) on the (active) links;
- optimization of the links internal to the overlay (and the rules for routing the messages), e.g., by considering the aggregation (i.e., logical and/or communication relationships) established among the entities of the service framework,
- monitoring of the links, in order to maintain the overlay, and to reconfigure it in case of link failures (e.g., due to the disappearing of the connectivity with the U-node hosting the peer).

Analogous set of actions should be provided by peers in order to react to the requests of such actions performed by other peers.

8.3 Possible approaches for realizing the overlay

This section describes the approaches currently available (from a technology viewpoint) for implementing the overlay enriched with Ant-based control algorithm. Moreover, it provides some preliminary indication on the possible solutions to be adopted in BIONETS. The literature offers a wide variety of algorithms and solutions for handling with overlay networks. Within the scope

of a general networked infrastructure, the proposed overlay networks provide an abstract view on the networked environment, tailored to specific needs, without the necessity to know or care about underlying real network infrastructures and communication protocols. Therefore an overlay network can be seen to act as a specialized middle layer between an application and an underlying network topology, detaching applications from the specific characteristics of the underlying network infrastructure. Such overlay architectures aim at enhancing some specific characteristics of the underlying networked environment or adding new features. Examples are efficient search of data items, reliable and performing huge data storage, new naming/addressing models, improved routing algorithms among the nodes, distributed control of shared resources, reduction of point of failures, etc. Message routing over the overlay can also be used to add extra functionality like resilience, security or multicasting support. P2P overlay network models offer a wide range of the communication framework, which specifies a fully-distributed, cooperative network design with peers building a self-organizing system These approaches aim at guarantee high degrees of scalability, and fault-tolerance, and new proposed solutions have the objective to improve such characteristics. While providing powerful mechanisms to handle decentralized and self-organizing networks of services, current proposals of P2P networks are very often designed tightly-coupled with the applications they should support (e.g., file sharing, streaming, knowledge sharing, communication session control), even if most of them share the implementation of certain capabilities, such as: self-organization in setting-up and maintaining the overlay topology, searching of data entries according to some query format, transferring file or data transfer. In some cases, the overlay topology and the routing algorithms are optimized according to criteria depending on the specific application context. These characteristics are not meeting the requirements of an overlay network decoupled from the above service framework and from the underlying communication protocol. The most popular P2P overlay proposals are targeted to search and retrieve information among huge amount of data (e.g., Gnutella, GIA, Edutella, DHT algorithms [54–56]); another group of proposals is focused on routing of messages to entities (e.g., JXTA [57], P2PSIP [53], or proprietary protocols for VoIP communications) according to some addressing/routing models overcoming the drawbacks/limitations of Internet, by introducing support to resilience, security, multicasting, mobility, and NAT traversal, or avoiding (logically) centralized servers. P2P overlay network solutions can be differentiated by the degree to which the overlay networks contain some level of structure (i.e. the way in which the contents of the network is located with respect to the network topology. In structured solutions, the overlay network topology is tightly controlled and files (or pointers to them) are placed at precisely specified locations. On the other hand the main characteristic of unstructured P2P systems is that the placement of data is completely unrelated to the overlay topology. According to these considerations, current solutions of P2P overlay are in general classified in unstructured and structured approaches [54]:

- Unstructured P2P networks (e.g., Gnutella, Freenet, GIA, Edutella) are composed of peers joining the network with some loose rules, without any prior knowledge of the structure of the overlay. They primarily use flooding as the mechanism to send queries across the overlay: improvements in order to reduce the adoption of pure flooding were introduced by several solutions (e.g., topology adaptation, introduction of super-peers, routing based on indexes,

or on semantics, etc.).

- Structured P2P networks have topologies which are tightly controlled and contents are placed at specified locations to make data retrieval more efficient. These systems use the Distributed Hash Table (DHT) as a substrate, in which data object (or value) location information is placed deterministically according to objects unique key; moreover DHT-based algorithm provide support for failure recovery. Structured P2P solutions differ from the adopted algorithm for the management of the DHT; examples are CAN (adopted also in P2PSIP), Chord, Tapestry, Pastry.

Even if structured P2P networks offer a more stable and reliable solution, from the point of view of information retrieval efficiency and failure recovery, they does not match most of the requirements for a network middleware in the BIONETS context:

- churn causes significant overhead for DHTs, both in terms of computation and message exchange; such a characteristic would not fit with the high churn rate of peers in BIONETS, due to the U-nodes mobility and the lack of connectivity;
- searches are retrieved through a key, a unique identifiers of the stored data; retrieving data through queries and/or keywords, supported by unstructured solutions, are more appropriate for supporting interaction models in the BIONETS service framework, than retrieving uniquely identified data;
- DHT-based algorithms are based on rigid topology, which hardly fit the needs of dynamic networks, such as the BIONETS ones.

Another consideration is that all these solutions address the scalability issue with respect to the number of peers in the P2P system, but it is equally important to address the voluminous information content of environment in which they run. The vast repositories of information are simply not favorable to key-based (or keys hashed from names) searches. Some recent proposals enriched the unstructured approaches by introducing adaptive features: the structure of the network is determined at run-time, by considering also the patterns of activity that occur over the network and the state of the peers. Such techniques exploit gossip-based [58, 59] and probabilistic multicast methods [60], for setting up the overlay structure. Even though the lack of any deterministic structure, such as DHT, makes the implementation of search mechanisms more expensive, the low dimension/diameter characterizing these overlays can be fruitfully exploited to realize multicast/broadcast systems with low message diffusion latencies. These adaptive features improve the applicability of the unstructured approaches to the BIONETS context. In particular, they would provide a support, from the interaction point-of-view, to service aggregation, in particular, for distributed execution of service cells deployed on several U-nodes, and to exchange of messages for self-* activities. The following subsections will highlight two possible approaches for a network middleware in the BIONETS architecture. The first one, based on unstructured solutions, is mainly addressing the support of a distributed data space for the sharing of data among the services executed by U-nodes in the same island. The second proposal, based on adaptive features,

is addressing the support to service aggregation for exchanging messages concerning distributed service execution and self-* behavior.

8.3.1 Unstructured P2P Overlay solution for distributed data space

This section provides some initial indication on the adoption of an unstructured overlay for providing a data space shared among the service/service cells executed by the U-nodes belonging to the same island. GIA [61] could be a possible candidate. It improves the Gnutella solution, a decentralized file sharing scheme which uses flooding for lookup, by adopting:

- a dynamic topology adaptation protocol, to ensure that high capacity nodes are the ones with high degree and that low capacity nodes are within short reach of higher capacity ones;
- an active control flow algorithm, to avoid that a node sends a query to an overloaded neighbor;
- an one-hop replication strategy: each GIA node actively maintains an index of the content of each of its neighbors; these indexes are exchanged when neighbors establish connections to each other, and periodically updated with incremental changes (jointly with the dynamic topology adaptation, this would make the selection of super-peers more dynamic and adaptive);
- a search protocol based on biased random walks: rather than forwarding incoming queries to randomly chosen neighbors, a peer selects the highest capacity neighbor for which it has flow-control tokens and sends the query to that neighbor.

Such solution fits several requirements identified in Section 4.1, among which:

- churn rate: the topology adaptation overhead requires limited processing capacity;
- scalability: the search protocol based on biased random walks address scaling problems with flooding;
- load balancing: the active flow control scheme tries to avoid overloading any of the nodes by explicitly accounting for their capacity constraints;
- distributed space: it is a decentralized system, with dynamic selection of super-peers.

Unfortunately, it does not fit the requirements concerning the possibility to handle data: actually, GIA is oriented to deal with files and the queries are just related on keyword matches.

Edutella [62] is a good candidate for addressing the requirements on data management and query handling missing in GIA. Edutella is a schema-based peer-to-peer network. Through metadata schema and attributes, defined in RDF, it is able handle generic data (i.e., not only files), and complex queries. It allows representing data and information in terms of tuples and provides a suitable environment to support service execution. Edutella should be improved in order to address requirements concerning churn rate and scalability. In order to cover them, a subnet of super-peer is added above the existing layer of peers (which are in charge to store the data). Super-peers

can be selected among the U-nodes that may guarantee suitable processing/storage capacity and a certain level of stability in terms of availability, connectivity, compatible with the assumption made at the beginning of this section (the selection of super-peers could also be influenced by the information produced by SNA algorithms [63]. The selection can be performed according to a passive join algorithm, as the one adopted in P2PSIP [64].

In subnet of super-peers, a clustering environment is adopted, so that peers get associated with the super-peers based on certain properties. Super-peers are connected in a loosely way without any topology to provide flexibility to our approach. In order to make data retrieval effective, super-peers are maintaining indices. Super-peers maintain two set of indices super-peer/super-peer routing indices and super-peer/peer routing indices. For other the other types of messages, super-peers use simple broadcast to communicate with all other super-peers. For improve the efficiency of broadcast some book keeping information is adopted.

To have optimized routing at the time of querying and replying, a swarm-intelligence based approach is adopted. The Ant-Based Control (ABC) algorithm is adopted to provide an effective routing: network exploration agents are periodically spread to maintain routing information. The proposed solution for Unstructured P2P Overlay for distributed data space may, therefore, be synthesized in:

- Edutella features [62]+
- Super peers with clustering [65] +
- Routing indices [65] +
- Ant-Based Control Algorithm [66].

8.3.2 Bio-inspired P2P Overlay solution for service aggregation and self-* interactions

This section reports some preliminary consideration on the definition and analysis of bio-inspired P2P Overlay applicable to BIONETS context, meant as more innovative solutions (with respect structured and unstructured) for the virtual communication and interaction of communities of agents/components with self-* (or bio) behaviors.

The objective of this overlay is to provide a support for the interactions, through direct message exchanges, between service individuals executed in the U-nodes.

According to the bio-inspired approach, the creation of P2P logical communities starts from the exploitation of some local rules of peers. As a matter of fact a number of simple and local rules are likely to govern about all biological systems (e.g. this is valid for cells level, for animal communities, etc.); such local rules allow producing emergent global patterns for resource management, task allocation, interactions, etc. in communities. In particular, such innovative P2P overlay network solutions may enable the virtual communication of agents/components running over systems and Users devices; in particular, such P2P overlay network solutions should be built, maintained and kept optimized with a simple set of local (e.g. bio-inspired) rules of peers. Then, thanks to the overlay interactions between agent/components, services are executed and provisioned to Users.

For example *swarm intelligence* and other nature-inspired heuristics have been investigated in the context of distributed computing. As a matter of fact, the dynamics of the groups of co-operating agents/components recall the social insects colonies: each individual only has a small sub-set of all capabilities required for the survival of the colony and so they heavily rely on each other. An agent framework is created at runtime to enable the decomposition of services workflows, instantiation and injection of *Goal Needed* (GN) into an individual agent/component (or an aggregation of them) to perform services (according to certain levels and/or criteria). Lets assume a local rule where each agent/component sends out a *Goal Achievable* (GA) to all of its neighbors. Each agent/component, that receives a GA message, broadcasts such GA message, combined with its own GA, with a certain TTL (Time To Live) to other peers and if it realizes an internal *Goal Needed-Goal Achievable* matching it sends back a join message to its parent. Parent peers receiving joining requests will select the best peer (according to certain selected criteria): a link will be set-up among the parent peer and the best peer. When the systems reached the stability, the best self-aggregation of peers emerges (based on the selected criteria) for executing that service. The algorithm includes a refresh, in order to react to components leaving/entering the overlay. For instance, when a new service is setup a *service* agent/component is deployed (with the GNs necessary for that service) into the agents/components eco-system. When the *service* agent/component is reached by a chain of concatenated GA (as sent by a certain agent/component) that is matching the GNs of the service, then the *service* agent/component sends a join message to the above agent/component determining the self-aggregation of all agents/components being part of that chain (of GAs).

8.4 Future works

This section presented an analysis of the role of a network middleware in BIONETS architecture, as a bridge between the network communication algorithms and the service framework elements, in particular, to provide a support to the interaction models. Moreover, it identified a couple of solutions for assembling such a middleware. The investigation should evolve in several directions. It is necessary to consider a greater integration of the middleware in the architecture: the functions implementing the middleware features should be implemented as specific service running in the service framework. In this way, it would be possible to apply also to them the adaptation and evolution paradigms applicable to the services. Moreover, the services could dynamically select the required overlay and configure it, according to their specific requirements. Another point to be further investigated is the possibility to consider also the other BIONETS nodes, i.e., T-nodes and AP-nodes, for identifying on one hand whether the overlay can improve the interaction among them and, on the other hand, whether they can play a role in the organization of the overlay itself. For instance, one of the aspects to consider is to interwork/interoperate with other overlays (either BIONETS overlays of other islands, or overlays running on the public internet) through AP-nodes. Finally, it is necessary to verify the impacts of the preliminary assumptions reported at the beginning of this section, in particular, the abstraction of the features and characteristics of the network communication protocols through a set of primitives.

9 Self-organization and situation adaptive message handling

9.1 Introduction

The starting point for the self-organization research line within BIONETS is based on the assumption that dynamic life of each BIONETS node (U/T/AP) requires intelligent situated and adaptive functionality. The *self-organization* capability refers to negotiating and reaching an organizational structure in a distributed manner and enabling it to survive even if changes are continuously happening either in the environment or in the node itself. As depicted in Fig. 17, such changes can be e.g. mobility, born of a node and death of a node. One example definition of self-organization is the following: "Self-organization is a process where the organization of system spontaneously increases, without this increase being controlled by the environment or an encompassing or otherwise external system. Self-organizing system not only regulate or adapt its behavior but it also creates its own organization" [67]. The individual entities exchange information locally and there should be no need for any external control entities. The configuration of the nodes should be defined on the fly at run-time because it cannot be known beforehand due to the various possible startup, execution and environmental situations.

Adaptation refers to the systems built-in capability to automatically adapt to the dynamically changing internal and external situation i.e. adaptation is reaction to some stimuli. There is a significant difference between adaptation and evolution because evolution refers to changes in a population of nodes but the adaptation is reaction to some stimuli happening to an individual node in the system typically just after the stimulus. *Situation awareness* (context awareness) refers to the knowledge of an individual node about itself and its environment. This knowledge can be used as a source of information and can help to make decisions, e.g. in the adaptation process. The system can be claimed to be situated when it takes the environment into consideration when the internal functional behavior is decided. Summarizing, the situated adaptive self-organization refers here to changes happening in the system and capabilities required for reacting to them in such a way that both the system and individual nodes stay functional and communication capabilities are maintained.

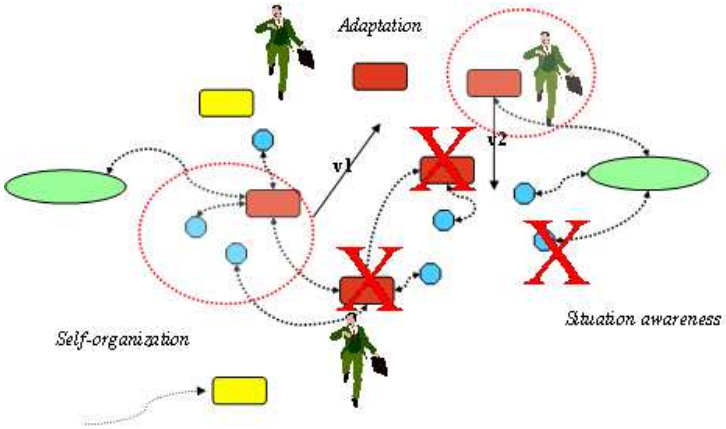

Figure 17: Situated adaptive self-organization.

The communication between any nodes visualized in Fig. 17 is possible using legacy technologies such as ad hoc routing and p2p communication. However, when communication is required to happen between nodes even when they are not necessarily available in the same network at the same time, the problem is more challenging to be solved by such means.

Sine BIONETS is a disconnected opportunistic communication system, mobile nodes are required to communicate even if no assumption can be made with respect to the existence of a complete path between two nodes willing to communicate. The referred opportunistic communication seems to have at least one potential starting point in Delay Tolerant Networks (DTN) [68] which includes the concept of DTN gateways between isolated Internet clusters. However, there is at least one significant difference compared to DTN gateways because in the BIONETS network case each node shall make the decision what to do with the incoming message and in fact each BIONETS node (most likely an U-node) shall act as a dynamic gateway. When a message comes in, the node should decide the next hop(s) according to the specific context to which the node is adapted (situated adaptive).

The opportunistic routing can be categorized to e.g. dissemination based or context based routing [69]. Dissemination based routing techniques aim to deliver messages to the destination by simply diffusing them all over the network. Examples of dissemination based routing techniques are Epidemic routing, Meeting and Visits protocol and Network coding based routing protocol. This approach works quite well when contact opportunities are common. The problem with the dissemination based routing may be the heavy load generated into the network which may cause network congestion and as the total number of messages in the network rises, the use of resources such as battery energy increases. The network traffic can be lowered by limiting the hops a message is allowed to travel or by lowering the number of copies of the messages.

The context based routing schemes apply information about the contextual situation so that the next hop towards the destination can be determined. The trade-off of these methods is the computational overhead caused by reasoning of the suitable next hops. Examples of context based routing techniques are Context-Aware routing (CAR) and MobySpace routing [69]. Because of the situated and adaptive requirements, the context based approach seems to provide a potential approach for BIONETS. Therefore, VTT has focused into situated opportunistic routing to enable communication between the nodes even when they are not necessarily available in the same network simultaneously. In addition, it can be seen that interoperation with legacy Internet infrastructure is essential from the service viewpoint (see section 4 about interoperability). Additionally, as a starting point for the situated opportunistic routing research, it is assumed that there are no strong needs for real-time communication and the required communication needs of BIONETS services are delay tolerant.

The capability to enable situation awareness for communication can be seen as one of the key component for the solution. The term situation awareness is used here as a synonym for context and context-awareness. The *context* and *context-awareness* has been defined and used differently by many researchers and often coupled with the area of the research as the concept can cover a broad range of things depending on the person using it. *"Context is any information that can be used to characterize the situation of an entity. An entity is a person, place, or object that is*

considered relevant to the interaction between a user and an application, including the user and applications themselves" is a definition introduced by Dey and Abowd [67]. When dealing with the BIONETS communication and networking paradigm(s), the definition of context should be bounded especially to the *communication context* from a single nodes point of view. To be more exact, the term communication context should be considered to comprise information which is relevant to communication between different U-nodes, U-/T-nodes and U-/AP-nodes. In addition, the node's internal communication context should be taken into concern together with the interaction between the service and the communication layer.

In [70], Schmidt et al have presented an architecture for context-aware adaptation of mobile devices based on signals received from physical and logical sensors. The sensors give cues on what is happening in the outside environment and inside the device itself and the prototype device's behavior was changed accordingly by a recognition software. This work has some high level concepts which can be related to the self-organization challenges within BIONETS but it does not deal with networking at all and concentrates on a single device's context only. Winograd [71] has explored some models and architectures which can be used when building a context-aware systems. He has described advantages and disadvantages of the presented models and it can be seen that the blackboard based architecture has had an influenced on how the interaction between situated adaptive network layer and the service layer has been initially designed to work in BIONETS. Predicting the future connection possibilities has been proposed by Drugan et al in [72]. Although the study concentrates on extracting neighborhood information from the routing protocol used in Mobile Ad hoc Networks (MANETs) and the idea is also dealt with in [73] by Lindgren et al, this kind of probability knowledge exchange could be integrated into BIONETS networking solutions after some careful thoughts about how much it would introduce overhead to the network traffic.

Bellavista [74] et al provide a middleware package for general wireless networks and discuss important features like mobility and heterogeneity in service provisioning required from a context-aware system when working in wireless network conditions. In consequence, the BIONETS network layer should facilitate the services information about the (changing) capabilities of a node itself and the networking conditions for the services to be able to dynamically tailor the content visible to the user. Korpip et al [75] have presented a framework for processing and providing abstractions of context information received from multiple sources in a mobile environment. The blackboard based framework incorporates some good ideas which have had an influence on the work done in this context. The described use of ontology for sensor data might have use in U-to-T-node information exchange. Also the Bayesian context recognizer could be helpful in the abstraction of lower level contexts into information usable at higher levels as well as making decisions at every protocol level in the stack.

In [76], Huang et al have proposed yet another framework which provides toolkits and services for dissemination and discovery of information in wireless networks. Even if they have targeted their research into more MANET type of networks, it tackles on the same problems of developing services in the dynamic and volatile conditions of these networks and tries to improve the efficiency by taking advantage of spatial, temporal and mobility information available. The local context-, neighbour- and mobility monitoring services are modules which should be present on the architectural level

also in BIONETS U-nodes. Finally, Gold et al in [77] have explored using context-awareness in peer-to-peer communication. They argue that the availability of underlying network conditions at the higher level can resolve into better performance and group organization. In this work, the communication/networking context is used as one (major) part of in recognizing the full situation of the node. However, this is not done at the application level even if some of the information is visible to the service layer via a blackboard type of API.

9.2 Approach

9.2.1 Requirements

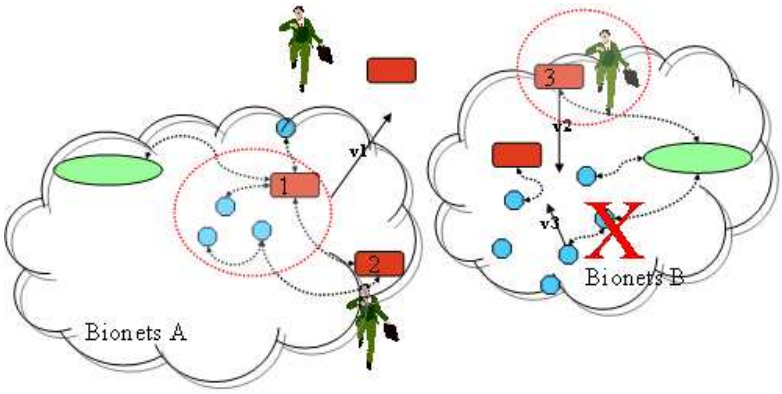

Figure 18: Situated Opportunistic Communication.

An example view to the BIONETS network, consisting of two bionets clusters, is visualized in Fig. 18. As it can be seen, communication between nodes 1 and 2 should be relatively straight forward because a radio link between them can be established. However, in this example, communication between node 1 and node 3 requires novel solutions because there is no communication link available via which a route from node 1 to node 3 could be found using legacy ad hoc networking technologies. This is because network islands, namely Bionets A and Bionets B, do not have any means to communicate with each other due to missing radio link at the time $t1$. It should be noted that the group of nodes around node 1 is moving according to known velocity v1. After some time node number 1, at time $t2$, may have a radio link to Bionets B. This gives opportunity for node 1 to send a message to the Bionets B which may have the required destination node 3. This is only a very simple example of opportunistic routing. There are some different ways to manage opportunistic routing and here two possibilities are discussed: dissemination based or context based. The dissemination based performs the delivery process of a message to a destination by simply diffusing it all over the network. On the other hand, context based routing exploits more information about the context in which the nodes are operating, thus identifying suitable next hops towards the eventual destination [69]. In this case, the term *situated* refers to the context based approach for opportunistic routing.

The technical challenges (i.e. requirements) for the situated opportunistic communication (SOC) are described shortly in the following.

- U-Nodes, T-Nodes and AP may be heterogeneous, and their roles may be different which needs to be taken into account in the SOC solution. Some nodes may have better abilities for forwarding/carrying messages towards the destination(s) than others. If e.g. a node has a very limited battery resources or the owner of the device knows that the battery is running out soon, it might not be possible to use the node for carrying or relaying messages at all;
- Each node should be aware of its internal state at any specific moment of time.
- Each node should be automatically aware in the situation by detecting the environmental conditions to be able to act without instructions received from outside.
- The awareness of the neighbourhood nodes is essential requirement for the situation awareness. This includes discovering the other nodes, identifying them and finding the relevant information content from this context.
- The configuration of the nodes should not need any settings set before the device has been set on.
- The knowledge of the situation should be computed as far as possible locally to lower the network load.
- The nodes may be mobile which causes requirements for situation awareness to be aware of e.g. data carrier/mule possibilities available in the nodes.
- The situation awareness needs to be taken into concern when deciding the next operation in each node. Because the system is in a continuous changing state, the opportunistic routing cannot work efficiently unless the situation is known.
- The adaptation to the situation seems to be a requirement of an essence for the opportunistic routing. First, it is essential to automatically detect the meaningful changes happening in the environment. After a change has been detected, the node's internal component needs to automatically be able to adapt to the new situation and change its internal and external behavior and rest of the system shall be able to adapt to the situation accordingly. In addition, it needs to be proactive in the adaptation in order make the system to be more robust.

The described technical challenges will be refined in the second iteration of the work related to situated opportunistic routing under the self-organization research line within the BIONETS project.

9.2.2 Adaptation to situation

The ability of the nodes to adapt their behavior according to the current situation in which node is in at any specific moment in time is one of the key requirements. The adaptation processs can be seen from several perspectives such as the whole node and its user [78], from the services and

from the communicational point of view. The work will be especially focused on the adaptation from the communication point of view.

The adaptation to the communication situation includes adapting the communication methods of the nodes to match the challenges of changing networking conditions and hardware resources. This includes e.g. monitoring the network interfaces available in the node for connection possibilities, finding information about the network neighborhood and sensing other environmental computing and communication characteristics. Adaptation requires also that the information on nearby T-nodes is available for the U-nodes. The adaptation process differs according to T-node information delivery method (push, pull). If a T-node applies push type information delivery, the detection of sensors nearby would be easier for the U-nodes because the T-nodes would then actively announce their presence and advertise their data.

The adaptation module will have to listen to the announcements by the sensors and update its snapshot of the situation accordingly. If a T-node applies pull type information delivery then the interested U-nodes will send a query to ask in return for the latest data from the sensors which can provide data similar to the asked type. In this case, the availability of sensors can be handled by at least two methods. The first is an active one where the U-node will send a compact query asking for all T-nodes of the given type to respond with a broadcast message. It would enable other U-nodes as well in the vicinity of the sensor to update their comprehension of the situation by listening for the responses. The problem of many U-nodes sending these queries, thus creating a lot of network traffic and consuming the precious energy resources can be avoided by adapting the sending process in a way that a U-node listens for responses sent by the T-nodes and delays its query if someone else has made a query before. The passive method for the U-nodes to update their situation knowledge about T-nodes is to listen for the queries and responses made by other nodes and issuing their own queries only when it is highly necessary. This feature could be more useful to small U-nodes with more limited battery and processing capabilities.

The communication related adaptation should take also some other issues into consideration such as e.g.:

- changes happening at hardware parts of the node which could affect how the traffic should be handled;
- known or concluded mobility patterns of the nodes;
- battery energy level and the rate at it is used if the device is battery operated;
- memory resources available for storing messages; the type is also significant: short lived (RAM) or persistent (disk);
- processing capabilities available;
- currently usable network interfaces and networks;
- nearby available memory storage facilities;

- other (legacy) networks available via APs.

9.2.3 Opportunistic routing

The contribution of opportunistic networking is allowing the nodes to exchange messages even if the parties of the communication may not be connected to the same network at the same time. However, the referred techniques usually require delay tolerance from the applications and also from the users.

Different kinds of information dissemination algorithms for BIONETS network are presented and analysed in deliverables D1.2.2 and D1.3.1 such as the Information Dissemination Protocol for BIOlogically Inspired autonomic Networks and Services (IOBIO) and the two-hop relaying information dissemination algorithms. Usually the dissemination methods work by offering the messages to neighbour nodes when they are within the radio coverage. The offering can consist of sending full message data to the neighbours which then can apply various filtering techniques to lower the network load. Another approach is to send advertisements of the data available at the sender and to the responders can then send a reply to request the wanted data. After this procedure the sender will respond with the actual data message. Both of the dissemination methods include a transmission of either the full data message or an advertisement of it to all nodes. However, it is quite easy to find cases use where there is no need to deliver the message to all the nodes in the network. For example, remote queries for sensor data could be delivered only to nodes which have access to T-nodes or messages delivering an e-mail could be destined to access points only. Some services might communicate by sending encrypted messages between two users located in the same group of mobile nodes. In the last case, the message could be delivered efficiently by attaching the name or the unique identifier of the node into the message and avoiding distributing or advertising the message to other members of the group.

The requirements still under discussion within the BIONETS consortium are related to the following questions:

- Is there a need to have a specific address/identifier for the destination nodes or a group of nodes? If so, how destinations are defined and applied in the opportunistic communication?
- Is there a need for actively grouping or clustering BIONETS networks and nodes?
- How necessary it is to have a method for merging and splitting BIONETS network islands?

9.2.4 Architectural considerations

The situated opportunistic communication is a possible approach and building block for the BIONETS communication services, see figure 19. The adaptation of the any BIONETS node to the environmental situation should be managed by the BIONETS frameworks available on the nodes. The applications and services can use generic BIONETS services and communication modules as utilities and APIs in order to reach their objectives. To enable this kind of interaction, well defined abstracted communication interfaces are required. These interfaces can be application data

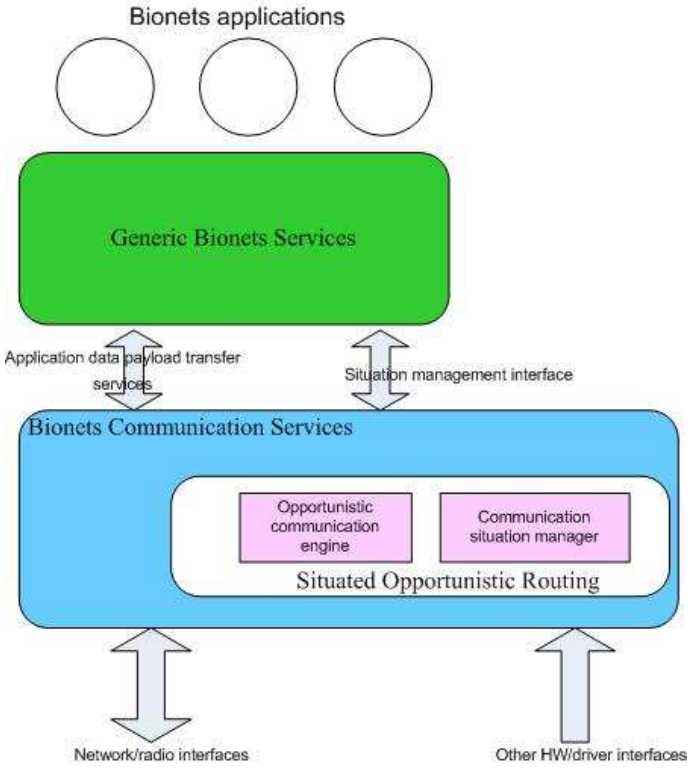

Figure 19: Situated opportunistic communication in the U-node architecture.

payload transfer services (e.g. sending and receiving of data) and situation management interfaces (e.g. publish, subscribe to various events raised from the network layer) which are required for the control and use of the situated opportunistic communication module. Using the situation management interface, a BIONETS service can subscribe for the information about the communication situation which it needs to run successfully. Some information situational data could be of value to be presented for the end user/application. Using the application data payload transfer services, a BIONETS service can transfer its data to anybody, to some group of nodes or to some specific destination. The situated opportunistic communication component will use any resources such as network/radio interfaces, other hardware/driver interfaces and nearby BIONETS nodes to get awareness of the communication situation (communication situation manager) which is then used by opportunistic communication engine as a source of local routing intelligence.

9.3 Description of the Situated Opportunistic Routing

In order to keep the system alive and the individual nodes running, the nodes in BIONETS should have some features which allow them to adapt to changes in their environment as described earlier in section 9.2.1. The referred functions will try to ensure that the nodes will use the networking resources available and has a fall back policy of functionality for the worst case possible scenario. However, because of the heterogeneity of the BIONETS nodes and their operation in variety of roles, the nodes may use that information and the information about the environment to try to optimize the usage of resources over the normal opportunistic way of distributing information in

the network. The following subsection will describe one part of the initial concepts for situated opportunistic routing protocol (SORP).

9.3.1 Destination handling

The BIONETS network may support a large variety of different kinds of services running on top of it. Some of the services may be more focused on providing group based experiences for the users while some involve communications between two parties or users and the transmitted data does not have any value to others. Some part of these information transmissions can be also encrypted or secured by using some other method if the service would handle personal or otherwise discrete information about the users and the goal is to prevent other parties to view or modify the data. A possible example service which might use this kind of data transmission could be a medical status collecting service where private information about the user of the node is regularly gathered for health observation purposes.

If this kind of scenario is looked from the network's point of view, the messages can be disseminated opportunistically with a more restricting relaying parameters, e.g. a small TTL or a near future timestamp. They can be used to indicate that the information inside the message loses its value soon after sending if the sender thinks that the destination can be reached soon. This could happen when the destination could belong to the same group of nodes or could become available relatively soon when a tight timestamp is used. Similarly, if the recipient is known to be far away or the knowledge is not available, the restricting parameters could be set to be less restrictive. Of course, the trade-off of this method is that it may create much more traffic into the network even if the destination node happens to be located near the sender after all.

To save valuable resources in situations described above, the situation awareness of the sender can be used to reduce unneeded transmissions instead of using the same method for information dissemination for all traffic types regardless of the message and node context at the moment. In epidemic routing described in [79], the nodes regularly send summary vectors which describe in a compact way which messages the sender has to offer to other nodes. The recipients compare their vectors to the received one and send a response including a set difference of the vectors. After that the original sender can send the actual data messages to the responder.

For the nodes to be aware of their network neighbourhood, they will need to probe the network to find out their neighbour nodes. To accomplish this, they can broadcast a HELLO message regularly after a defined time period has passed. The messages should be as small as possible in terms of used bytes so the HELLO messages should only include what is necessary for the nodes to identify the sender and to form a snapshot of the current situation about the neighbourhood.

The sender field will include the identifier of the sender node. In this phase it can be assumed that the referred identifier can be a simple random number. The role field of the HELLO message contains the current communicational role of the node in a condensed representation form. The role information can be transferred using bit fields for the mandatory fields but some could be described in text as a attribute/value tuples. Once a node has received a HELLO message, it will respond with a similar message but the type of the message needs to be set to HELLO_RESP.

Also the sender and role fields are set to match the responder. The HELLO_RESP is sent directly to the HELLO message originator in order to save resources in other nodes. This would require a MAC protocol to be available. By using the HELLO process, the nodes can store the received information describing neighbours into a database or to some other suitable data form to be used by the situation module. The neighbourhood status information will be one part of the whole network situation which can be used for example in a way described below.

When a message is received from the upper layers of the protocol stack, e.g. from some service, to be disseminated in the network, algorithm 2 is used. The first check is made if the message is meant to be sent to the localhost, i.e. to the sender itself. In that case, the message will not be sent to the wireless network at all. Next, if the message is not meant for the localhost, the situation module will be queried for the best candidates when the destination of the message is given as a parameter. In the algorithm this has been described as if the message's destination field corresponds to the answer received from the situation module, the message can be unicasted directly to the right receiver. This will prevent the message to be disseminated to possibly wide range of nodes in the network depending on the set spreading restrictions of the message.

If the situation module cannot provide a candidate from the nearby nodes, the sender should issue a SEARCH type of message to its neighbourhood. In addition, a SEARCH_TIMER should be set to run in the background with a configurable timeout. Now as time passes, the sender might get a message from the network whose type is SEARCH_RESP. This will indicate that some of the situation modules of node's nearby can provide a good candidate to the sent SEARCH message. Once this kind of message was received, the original sender should go through its message inventory and see if there are messages waiting for the suitable destination candidate and on success, the service data message can be sent to the node where the SEARCH_RESP came from.

While sending, the data message should be labelled as SEARCH_DATA to make processing and distinguishing of the SEARCH messages possible. Also the SEARCH_TIMER should be cancelled as it has no use anymore because the best candidate was already found. To make the process more fail safe, the received SEARCH_RESP should be dropped if the node does not have matching messages waiting in the message inventory. This can happen if multiple SEARCH_RESP messages are received from different nodes and the first one will trigger the send process, thus making the other ones obsolete. In the case that the SEARCH_TIMER runs out, it marks that no suitable forwarding direction was found. At this point, the message should be added into the message inventory to be disseminated with some protocol which has been selected to be the fail safe protocol. The fail safe protocol could be the same protocol which is used in the network for dissemination of information to wider amount of nodes, e.g. epidemic routing or IOBIO.

When a message is received from the network, it is processed using algorithm 3. In this case, the processing is divided into three distinct segments in accordance with the type of the message. If the received message has the type of SEARCH, the node needs to query its situation module for the given destinations. If a suitable node was found, a SEARCH_RESP message should be sent back with the information received from the situation module. In the case that this node's situation module cannot provide information about the queried node, the SEARCH message should be dropped quietly. This will help to reduce the amount of network traffic but the trade-off is

Algorithm 2 Processing of messages received from the upper layer.

```
1: if msg.dest() == us then
2:    send msg back to upper layer
3: else
4:    if msg.dest() == situation.query_for_best_candidate(msg.dest()) then
5:       send msg to neighbour
6:    else
7:       send a new SEARCH msg to neighbours
8:       start a new SEARCH_TIMER
9:    end if
10: end if
11: ...
12: if SEARCH_RESP msg received then
13:    if queue contains a msg matching SEARCH_RESP then
14:       send DATA msg queued labeled as SEARCH_DATA to sender of SEARCH_RESP
15:       cancel(SEARCH_TIMER)
16:    else
17:       drop msg
18:    end if
19: end if
20: ...
21: if SEARCH_TIMER runs out then
22:    send msg using fail safe protocol
23: end if
```

introducing SEARCH_TIMER amount of delay to the possible message delivery time. However, the timer value used should be relatively low due to the small delays in the used radio links. Also when the overall transmission delay in BIONETS type of disconnected network is considered, the introduced delay can be categorized to be something not to be concerned about. Additionally, the services and applications should be built from the ground to withstand delays and therefore the before mentioned delay should be easily dealt with.

If the received message type is of SEARCH_RESP, the processing should be done in the same way as described in the algorithm 2. Finally, the SEARCH_DATA indicates that the actual service data message is being transferred to the destination. The receiver should first check if the message was destined for it. If true, the message should be delivered up the protocol stack to the service level. In the case of not true, the situation module should be queried if it has knowledge about the destination mentioned in the received message. On success, the message can be forwarded to the right node. If for some reason this is not possible, the message should be saved into the message inventory for possible later delivery attempts if the capacity of the inventory allows this. The messages can have different urgency categories and depending on those and other dissemination limiting factors, the messages should be purged from the memory in a way that the most needed and important messages are stored for a longer period than messages marked with a low priority.

9.4 Conclusions and future work

The challenges, requirements, and preliminary description of one part of the situated opportunistic routing approach and protocol have been described in this section as a result from task 1.2.2.1

Algorithm 3 Processing of received messages received from the lower layer.

```
1: if msg.type() == SEARCH then
2:   if answer = situation.query_for_nearby_nodes(msg.dest()) then
3:     send SEARCH_RESP back to msg sender
4:   else
5:     drop msg
6:   end if
7: end if
8: ...
9: if msg.type() == SEARCH_RESP then
10:  if msg queue has a msg matching SEARCH_RESP then
11:    send DATA queued msg labeled as SEARCH_DATA to sender of SEARCH_RESP
12:    cancel(SEARCH_TIMER)
13:  else
14:    drop msg
15:  end if
16: end if
17: ...
18: if msg.type() == SEARCH_DATA then
19:  if msg.dest() == us then
20:    send msg to upper layer
21:  else
22:    if situation.query_for_nearby_nodes(msg.dest()) then
23:      forward msg to dest
24:    else
25:      save msg into message inventory for later delivery
26:    end if
27:  end if
28: end if
```

situated adaptive self-organization activities. The simulation results of the described version of the protocol have been provided in the deliverable D1.3.2. However, the provided results are preliminary and describe the status of the work done so far and the aim is to continue this area of work in future as described in the following.

The technical challenges (requirements) for situated opportunistic communication will be refined after the BIONETS scenarios have been specified in a more detailed way. In parallel with this process, VTT's aim is to continue the activities to enhance the situated opportunistic communication and related routing algorithms to enable more complicated routing situations, handling of different kinds of arising application scenario requirements on the fly and interoperation with the legacy Internet networks as a multi-tier situated opportunistic communication. The aim is also to investigate into the initial analysis of the approach in a theoretical sense and continue the evaluation of the approach by means of simulations using OMNeT++ (WP1.3).

In addition, the aim is also to make more detailed performance and scalability analysis when the number of nodes and heterogeneity will be increased. Also, it may be valuable to evaluate by means of simulations the possible interoperation of situated adaptive opportunistic communication with some relevant related parts of service framework. This would require a definition of an API between the service framework and the situated communications. Finally, the enhancement of the

realized and planned simulation modules of situated opportunistic routing into smaller and more re-usable modules is planned.

10 On the use of attribute-value pairs

The primitives described in sections 8 9 leverage the BIONETS communications architecture. For example, the `send(query,message)` described in 8 assumes implicitly that services can issue the delivery of the query to the peer neighbors, using the underlying communication mechanism. Thus, it is implicitly assumed that, each time a service triggers communications in radio range, a suitable message formatting and transmission is performed.

The aim of the present section is to provide a description of the syntax for a generic set of primitives leveraging the attribute-value pair syntax described in [15]. The following formulation is one of the possible solutions proposed in SP3, and this examples are meant to fix a reference scenario for Serworks in agreement with the SP3 indications [52].

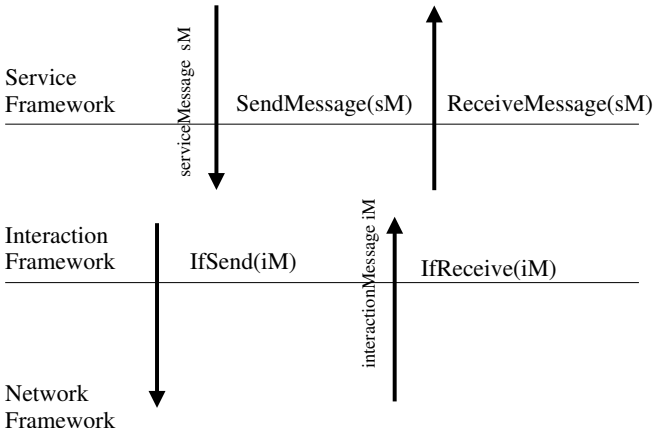

Figure 20: Primitives for the exchange of messages among the different frameworks.

10.1 Service Primitives

At the service level, a set of primitives is used to create a message and then to send it to peer services running on nearby U-nodes. The three fundamental primitives of the U-nodes layer that are provided to the services are a primitive to send a message, a primitive which receives a message and a primitive that issues a query. They operate between the Interaction Framework (IF) and the Service Framework (SF), as described in [15].

At this level, the data unit is the `ServiceMessage`. In order to generate a `ServiceMessage`, a service will concatenate to the main message body which contains the data to be exchanged, with specific attributes that will regulate the transmission and the reception of the message towards peer services [15]; also, in order to maintain the same format, the data field will be a particular pair in the form `<Data; ChunckOfData>` expressed in a format suitable to the service. A `ServiceMessage` contains a mandatory destination field represented in the form of a *set* of pairs

attribute-value, which can be concatenated and composed in order to obtain the addressing of specific services running on U-nodes in range. As an example, if the entry for such field is in the form `<DestinationService; <serviceType; HealthMonitor>;<DataType; Temperature>>`, then the destination for the `ServiceMessage` is any service running on a U-node in radio range which matches all or some of the pairs reported in the field. The precise match, i.e., wheter the correspondence with the entries of the destination field will be complete or partial, will be depending on the service. Hence, one possible destination for such a message will be a service which monitors the health status of the user and leverages temperature measurements. Furthermore, a set of service-specific attributes will be attached to the message in order to complete the message specification: the attributes of the message are modifiers that are specific of the service issuing the transmission. Some modifiers will then refer to the operations that are allowed for a specific message class [20]. In particular, one of such attributes will be the allowed morphing for the message at hand. For example, a pair such as `<MorphingTypeAllowed; singleClass>` will have the meaning that the service is releasing the message under the constraint that any morphing operation will be operated within the same message class. Conversely, a primitive such as `<MorphingType; AggregateData>` will permit to the peer service receiving the message to possibly aggregate all the data of the same kind into a unique message. More complicated modifiers for messages are possible, and the pairs type-attribute can specify a wide range of data handling strategies besides the modifiers related to the morphing strategies.

At the receiver side, a service will check whether it matches the `DestinationService` field inserted by the sender; as said before the set of attributes to match and receive the message are decided in the SF. Thus, for example, if the service is not a health monitoring service, `<ServiceType; HealthMonitor>`, it will simply drop the message. The remaining attributes associated to the message are used by the peer service receiving a message in order to apply the instructions on the operations to be performed over the message, which are released together with the message by the sending node. Thus, an indication in the form `<MorphingType; AggregateData>` will be interpreted as the indication to aggregate the temperature data received. Basically, a message is received together with the instructions from the peer services on the data handling strategies that are expected for such message.

A reserved data unit is the `ServiceQuery`, which will be the concatenation of several pairs attribute-value with no data field. One of such pairs will be the `Scope` field which will optionally include a particular type of attribute that indicated the spatial and time limits of the query, and this will be an input for the underlying filtering mechanisms; of course, information filtering mechanisms are not visible at this level, but this will be of help in order to permit the retrieval of the needed information, and also to override the filtering mechanisms, when for example some particular type of information needs to be searched for with higher priority. As done for the other primitives, a set of attribute pairs will be employed in order to specify the target object of the query, i.e., pairs of the form `<Color; Green>`, `<Fruit; Apple>`, `<Bird; Parrot>` which can be issued to peer services in order to retrieve specific messages. At the same time, a modifier can be attached to the list in order to modify the way the query is performed: for example, specific pairs can be used in order to predicate the previous input arguments of the query. Queries have

a particular properties since they do not carry data, and thus can be propagated and tunneled through alien networks.

The following list resumes the service primitives leveraging `<Attribute; Value>` pairs:

- *Sending a Service message*: the primitive for sending a message to nodes in radio range, operates top-down, i.e. from the SF to the IF and looks as follows:

  `void SFSend(ServiceMessage Message);`

- *Reception of a Service message* : the primitive for receiving a message from nodes in radio range works bottom-up, i.e. from the IF to the SF, and looks as a notification in the following format:

  `void SFReceive(ServiceMessage Message);`

- *Service Query propagation:* When propagating a query, a U-node service will issue a primitive with the dedicated data unit, i.e., the `ServiceQuery`. A query will be propagated with a call of type

  `void QuerySend(ServiceQuery Query).`

10.2 Interaction Framework Primitives

At this level the syntax of two functions are detailed, basically, a general primitive `IFSend(·)` and a primitive `IFReceive(·)`. At the network framework, the U-nodes will resolve the transmissions within radio range. Data units are `InteractionMessages` passed from the IF to the NF. Viceversa, the messages received from neighboring nodes through the NF will be passed to the IF in a dual manner. The `InteractionMessage` is in the usual form of pairs of data type and attribute in order to specify the form they are applied. The `InteractionMessages` are obtained at the IF encapsulating a `ServiceMessage` or a `ServiceQuery` with a set of pairs attribute-value. In particular, the sender node is denoted by the `<senderAddress; A_address>`, in order to provide the address of the sending node, if needed for the current transmission. The field `<sourceType; <typeofNode; U-node>>` is meant to signal that the transmitting node is a U-node, whereas the `<destination; <Address; Broadcast>>` will denote a broadcast transmission. In case when the address for a U-node is available, the sending node might employ a unicast transmission or a multicast transmission with multiple pairs in the form `<destinationAddress; <address; Address1><address; Address2>...>`. A further attribute might specify the type of protocol to be used to access the medium, and one example might be `<MACtype; IEEE802.11>`, where the effect is of selecting the appropriate layer 2 protocol to send the message.

The following list resumes the IF primitives leveraging `<Attribute; Value>` pairs:

- *Sending an IF Message* A primitive for sending a message from a U-node to U-nodes or T-node in radio range will then be

```
void IFsend(InteractionMessages Message);
```

- *Receiving and IF Message:* the dual primitive will be

```
void IFreceive(InteractionMessages Message).
```

10.3 Network Primitives

The primitives at the network layer will be a cover a set of functions in order to:

- Operate the message fragmentation and reassembly: this is due to the requirements of the underlying MAC technology;
- Optionally operate address resolution in radio range and association with a particular service address;
- Map the messages on the correct RF interface, depending on the target transmission, i.e., towards T-nodes, U-nodes or APs.

10.4 Example: HELLO messages for neighbor discovery

The HELLO messages are a typical networking solution adopted to solve practical issues such as, for example, neighbor discovery as already discussed in Sec. 9. The neighbor discovery might be requested by a given service, that needs explicitly the set of U-nodes running the same service in radio range.

In the BIONETS architecture, this can be the case when, for instance, a given U-node, say U-node A, has several MAC/PHY interfaces and needs to find the largest possible set of neighbors that are able to communicate. Hence, it will perform the following operations:

1. At the SF, A generates a `ServiceMessage sM` containing two main fields `<data; Hello>` and `<ServiceID; ID>`, where the field `ID` can be any unique identifier of the service; one way to obtain a temporary identifier is for example `NeighborDiscovery:UserName:DeviceName`. Also, the destination is `<destinationService; <serviceType; NeighborDiscovery>>`, meaning that it is any service instance of the kind `NeighborDiscovery`, and `<networkMode; Exhaustive>` specifies that neighbors should be searched over all possible links;
2. At the IF, A will issue a `MessageSend(sM)` which will trigger i) the generation of a `Interaction Message` and ii) the activation of the `IFsend()`;
3. At the NF, assuming that A has a IEEE802.11 and a Bluetooth MAC, the NF will read `<networkMode; Exhaustive>` and interpret the attribute so that to generate a pair of distinct `InteractionMessage` including the field `<MACtype; IEEE802.11>` in the former case and `<MACtype; Bluetooth>` in the latter case. Also, additional fields `<address; Broadcast>`; `<sender; address_A>` and `<sourceType; <typeofNode; U-node>>` will be specified.
4. At the NF, the message will be possibly fragmented and sent via the IEEE802.11 MAC with a (set of) broadcast frame(s).

A node B in radio range of A, will perform the following sequence of dual operations

1. At the NF, node B, listening on a IEEE802.11 card, will receive a the sequence of broadcast frames, and will reassemble them and pass the message to the IF;
2. Upon reception of the `interaction Message iM` the IF will generate a `IFreceive(iM)`;
3. At the IF, A will extract the `ServiceMessage sM`, and will generates a `MessageReceive(sM)`;
4. At the SF, A will try to match the set of service addresses that could indicate a message destined to services running on B; since the address is a pair of the type `<serviceType; NeighborDiscovery>`, the corresponding instance running on A will be notified;
5. The `NeighborDiscovery` service will extract the fields `<data; Hello>` and `<serviceID; ID>` and will store the ID of the service as a peer service in the neighborhood;

11 Network security

This section proposes a possible security framework for the BIONETS network presented above. A three layer approach is chosen which allows the protection of the network in a multi-layer fashion. We refrain to map the names for these three layers to the traditional ISO OSI reference model (also see [80]) as BIONETS networking is different. Thus a direct mapping is not possible.

Therefore, we start by defining the three layers:

Link Layer Security designates mechanisms which secure wireless point-to-point or point-to-multi-point communication on a channel basis.

Dissemination Layer Security designates mechanisms which run on top of an established link between two nodes and secure messages which belong to the data dissemination process, e.g. ensuring integrity, authenticity, and confidentiality.

Service Layer Security designates to mechanisms securing service related data while being processed in a BIONETS.

The application of all three layers is optional and dependent on the context. Thus we allow for different forms of BIONETS islands which range from completely open (no security mechanism in place) to very restrictive environments (the link layer excludes nodes from injecting messages in the network, nodes are not able to read all data relevant for data dissemination, nor are nodes able to read all service relevant data).

In general the layers should be seen as complementary levels which account for the deficiencies of heterogeneous devices and technologies of communicating entities in BIONETS.

11.1 Requirements

This sections identifies security requirements for the BIONETS network. They are based on refined collections of the requirements which were assembled in deliverable D1.1.1 [20] and D4.2 [81] (which, in turn, contains results from the internal deliverable ID4.1). For this purpose we separately address each of the layers defined above.

11.1.1 Link Layer

- **Access control:** excludes unauthorized nodes from the network. As BIONETS is mainly going to operate using a shared medium this security mechanism will be required to prevent basic eavesdropping and spoofing attacks. To also effectively address denial of service attacks which are based on malicious packet injections access control is required.

- **Authentication:** To finally authorize a node we will need some means to authenticate a node which wants to join the network. Thus, feasible authentication methods for nodes will be required.

Obviously, as we do not assume wired networking, messages which are injected in the network also have to be authenticated such that messages from unauthorized nodes can be detected instantly.

- **Integrity:** Unauthorized nodes should not be able to manipulate messages and re-inject them without these changes to be detected.
- **Confidentiality:** As we are transmitting data over a shared medium eavesdropping is a relatively easy task for an attacker. Thus the traffic between nodes is required to be confidential.
- **Anonymity:** is an important security requirement for nodes in BIONETS. Users of the network may be directly (e.g. a user carrying a node) or indirectly (e.g. cars equipped with a BIONETS node) associated with a node. Thus any information about users which could be derived from the activities of nodes must be protected.
- **Replay resistance:** To prevent replay attacks which use recorded messages and inject them into the network at a later time in order to compromise applied security mechanisms or to manipulate the overall state of the system appropriate mechanisms have to be applied.

  Thus message *freshness* (weak or strong) is a requirement for link layer security in BIONETS.
- **Low overhead:** Application of security mechanisms always implies some cost. The cost in BIONETS will certainly be application dependent. However, as in any other wireless network with nodes which have limited resources power consumption, memory requirements, and computational overhead are prevalent characteristics considered when designing new security mechanisms for such system. Thus, it is required to keep the overhead for the proposed solutions as low as possible.

11.1.2 Dissemination Layer

In BIONETS we foresee the communication between nodes in an island. Although they may be located in direct vicinity they do not have to have established network links between them. For efficiency, however, we assume that this island can form a group of devices whose communication is secured on the link layer. Thus, once a node has managed to become part of this island the traffic between nodes is subject to widely known attacks such as eavesdropping, spoofing, replaying, etc. Given that data dissemination will the backbone of BIONETS communication we can identify the following security requirements for the dissemination layer.

Here, we want to emphasize again that if the requirements listed below can already be achieved at the lower layer, the link layer, the security framework has to decide which mechanism should be applied best.

- **Message integrity:** A malicious node may be able to decrease the performance of the network by changing queries and their answers injected in the network. By changing these messages he may also simulate a wrong state of the environment or the network. Thus, the integrity of queries and their corresponding answers is a requirement.

Due to the fact that erasure coding techniques are proposed in BIONETS message integrity at the dissemination layer is a strong requirement.

- **Authenticity.** Not only integrity is required to counter the attacks described above but also authenticity as an attacker may still be able to generate messages which have integrity.
- **Confidentiality:** Due to privacy concerns, especially for U-nodes, it is required that queries and answers to these queries can be kept confidential.
- **Anonymity:** is also a requirement for the dissemination layer. Although isolated messages can not be directly mapped to single nodes, recording traffic over a longer period of time and combining these records with environmental data (e.g. location of nodes, users using the node) or information obtained from the link layer could possibly break the anonymity of users.
- **Replay protection:** Nodes in the network may be listening and recording network traffic and replay this traffic at some later time to surmount some security mechanisms or to manipulate the network status and the observed environment. Hence, freshness of messages disseminated is required.
- **Low overhead:** Although we may deal with powerful devices (U-nodes or APs) capable to perform cost-intensive, in terms of resource consumption, mechanisms we have to account for nodes with very restrictive characteristics. Thus, we need to provide for mechanisms which do not impose large overheads. Of course, different mechanisms for different types of nodes can be used. Consequentially, a BIONETS node has to provide adaptive mechanisms which account for the different communication partners.

11.1.3 Service Layer

Theoretically, the service layer has to have the same requirements as the dissemination and link layer. But, as the security requirements are mainly application dependent the actual requirements for a specific service are determined dynamically by the security policies associated with the service and the node it is running on. In the worst case however, the service layer has to be able to support all the requirements above. Thus, the secure service execution framework (see [81] for more details) will need to be able to provide appropriate security components which support the required security functionalities.

As this layer is service specific we will not elaborate the feasible mechanisms but argue about the particularities for each type of interaction and show how the layer fits in the general infrastructure.

11.2 Implementation

In the last section we listed the requirements for secure networking. This section will identify possible solutions for securing the network proposed in this deliverable. As BIONETS is based on one hop communication we separately look at the pairings of all different node types.

11.2.1 T-node to U-node communication

Deliverable D1.2.1 [14] defines T-node classes defining the communication technology T-nodes are equipped with. Additionally internal deliverable ID4.1 [82] and deliverable D4.2 [81] define the security classes of T-nodes which reflect their capabilities to perform specific security relevant operations.

Under the awareness that costs and technologies may change over time the following table merges both classes and identifies the following T-node classes focusing on communication and security technology present within these T-nodes.

Technology class	A					B					C				D		
Security Class	0	1	2	3	4	0	1	2	3	4	1	2	3	4	2	3	4
Identification	-	+	+	+	+	-	+	+	+	+	+	+	+	+	+	+	+
Authentication	-	-	+	+	+	-	-	+	+	+	-	+	+	+	+	+	+
Integrity	-	-	+	+	+	-	-	+	+	+	-	+	+	+	+	+	+
Bi-Dir. Comm.	-	-	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+
Update Capabilities	-	-	+	+	+	-	-	+	+	+	-	+	+	+	+	+	+
Key Exchange	-	-	-	+	+	-	-	-	+	+	-	-	+	+	-	+	+
Sym. Key Crypto	-	-	-	+	+	-	-	-	+	+	-	-	+	+	-	+	+
Public Key Crypto	-	-	-	-	+	-	-	-	-	+	-	-	-	+	-	-	+

Table 1: Proposed T-node classes (A-D) merged with their security classes (0-4)

In the following, we assemble the class name of T-node by using its technology class followed by its security class. Both classes are separated by a dot. Thus, if we speak of a T-node which uses 802.15.4 as its communication technology and supports symmetric key cryptography we talk about a class B.3 T-node.

As you can see in table 1 the merge of the different classes removed some irrelevant classes. Irrelevant classes were identified by their use of technology and cost. As an example take a class D.0 T-node. This node will never be built as the cost would be too high to not equip it with any security feature. At least, there wouldn't be any reasonable justification. For this reason we left it out. Also note the change in security classes B.x, C.x, and D.x as far as bi-directional communication is concerned. This change is of course based on technology used by these classes.

Based on the classes in 1 and under the assumption that a U-node is going to support all of the above technologies a T-node possesses the following paragraph is going to discuss which existing security mechanisms supported by these technologies can be used to secure T-node to U-node communication.

Link Layer Security

Here, we briefly outline possible link layer security solutions that could be applied according to the different classes of T-nodes.

- **RFID:** Solutions are in the field of RFID security, achieved by protocols developed, e.g., in [83] and [84].

[83] is concerned with privacy issues, in particular, to avoid tracking the behavior of T-nodes; reading, by an adversary, the data stored at the T-node; eavesdropping the data in transit between the T-node and the U-node. [83] addresses both the issues of protecting privacy of contents at RFID tags side (i.e. T-nodes, in our context), and also of assuring that readers (i.e. U-nodes) read data from valid, not manipulated tags. To achieve these goals, each T-node is equipped with a key, and each U-node must search the right key throughout a set of keys it knows, according to algorithms that achieve a good tradeoff between security and efficiency. Having a look to the table, at least A.3 and A.4 T-nodes support privacy protection.

There are also mechanisms which take a more active and inexpensive approach to protect the privacy of a user: so called blocker tags [85]. These tags, in the BIONETS scenario T-nodes, can prevent the identification of a specific tag by blocking the tree walking algorithm. Selective approaches which allow for privacy zones exist.

More intrusive approaches such as the *kill instruction* have been proposed by the non-profit organization EPCGlobal and are applicable in certain environments. Due to the fact that the proposed instruction can disable specific parts of an RFID the instruction has to be designed in such a way that it deletes as much as required to protect the user privacy but as little as possible to keep the system as operational as possible.

Also, T-nodes that are under the complete control of an adversary can be killed ([84]) (at the time of manufacture, they can be programmed in order to receive a particular password that deactivate it). A.1-4 T-nodes can be deactivated.

Each security class of A T-nodes can be isolated by any kind of electromagnetic waves through a Faraday Cage ([84]).

One can also exploit the use of access control schemes based on one-way hash functions [86,87]. The proposed mechanisms assure that only a U-node, possessing a previously selected secret, can access a subgroup of T-nodes. Randomized variants also permit to avoid the tracking of individuals. Appropriate refinements which also account for privacy have been discussed too [88, 89]. To some extent, these schemes can be applied to A.2, A.3, A.4 T-nodes. In the case of A.2 T-nodes, the hash functions must not be based on ciphers.

Finally, some solutions do not use true cryptographic operations, e.g., [90] proposes a set of extremely lightweight challenge response authentication protocols. A.2, A.3, A.4 T-nodes can exploit these.

For public key cryptography two options are currently feasible: RSA and elliptic curve cryptography (ECC) [91]. With *TinyPK* Watro et al. [92] have shown that parts of the RSA cryptosystem are applicable to wireless sensors, which would resemble more powerful T-nodes in BIONETS, A.4 class T-nodes. First implementations of public key cryptography on highly resource constraint nodes using ECC have been shown [93].

Finally, for critical applications we propose the use of class A.x T-nodes which offer physical contact channels known from smart cards [94].

- **IEEE 802.15.4:** When T-nodes are equipped with IEEE 802.15.4 technology, securing communication between T- and U-nodes may rely on the security of that technology. At the link layer, 802.15.4 offers the possibility to employ security mechanisms which secure confidentiality and authentication, access control, replay protection. These mechanisms are based on AES variants and different modes. The basic mechanisms were enhanced in [95], upon an analysis on their possible weaknesses in IV management, key management and integrity protection, based on some optional features of the standard that actually reduced security. B.3 and B.4 T-nodes may implement these security features (apart from access control, a property that is not foreseen to be fulfilled by each security class of T-nodes).

- **IEEE 802.11x:** From the very beginning, several flaws have been found in the part of the 802.11 standard related to security, e.g., the WEP protocol, [96–98]. Flaws mostly regarded the way in which cryptographic algorithms may be used by implementors. Indeed, the key-stream used to encrypt a message guarantees confidentiality only if it is used once; it was recommended to change the key stream, but a problem in the initialization of vector IV brings to frequent collisions. Furthermore, some attacks also succeeded as far as the integrity check is concerned.

  Thus, when T-nodes are enabled with IEEE 802.11x interfaces, attention should be paid to avoid harmful solutions, (e.g., WEP in 802.11) and to adopt more recent, amended proposals, such as IEEE 802.11i, also known as WPA2, [99]. 802.11i makes use of the Advanced Encryption Standard (AES) block cipher, that overcomes the weaknesses of RC4 stream cipher used in WEP.

  In any case, T-nodes must perform key exchange and cryptographic operations to run these protocols, thus only classes C.3 and C.4 can be secured according to these techniques.

- **IEEE 802.16 and 3G:** BIONETS architecture can implement security suites of IEEE 802.16 when T-nodes are in class D.x. In particular, key exchange and cryptography are required for implementing IEEE 802.16 security protocols, thus the interested T-nodes are D.3 and D.4.

  Similarly to the previous item, several flaws have been discovered in this standard. Indeed, designers attempted to reuse a security scheme for wired media while 802.16 technology is wireless. Because of the difference in the threat models, an IEEE 802.16 link must be protected with enhanced solutions compared to the original proposal. [100] outlines changes that could defend the proposal from threats.

  Under 3G (third generation) mobile communication designates the migration of GSM (second generation) via GPRS and EDGE to UMTS. Thus we briefly discuss which mechanisms from UMTS are applicable in BIONETS.

  The 3GPP does not define any authentication algorithm but gives general guidelines for appropriate choices of the algorithm and the key used for authentication. For increased security the authentication algorithm is placed in the smart card contained in the node. UMTS uses the 128-bit block cipher *Kasumi* for encrypting traffic. It is based on the MISTY1

cipher [101] and is also used to provide the integrity of transmitted messages. Despite some flexibility to synchronization errors the design of the UMTS security framework enforces the freshness of sequence numbers and thus prevents common replay attacks. As in GSM networks, UMTS also uses a Temporary Mobile Subscriber ID (TMSI) which protects the subscribers privacy.

TinyOS operating systems may run on T-nodes belonging to technology classes B, C and D. TinySec, [102], is a link-layer encryption mechanism, incorporated in TinyOS, with specific features of confidentiality, replay protection and authentication, with low energy utilization, originally tailored for wireless sensor networks communication. It does not make use of public key cryptographic, but only of symmetric encryption and Message Authentication Codes (MACs), making it suitable for B.3, C.3 and D.3 T-nodes. The same holds for MiniSec, that basically consists of a further improvement with respect to power consumption, [103].

Dissemination Layer Security

Although we may protect the communication using link layer security technologies outlined above we also argue for applying additional security mechanisms on the dissemination layer. Reason for this is the multi-layer approach we are following. With the above link layer security you may exclude specific nodes from participating in the network, prevent them from eavesdropping on the communicating partners or to spoof exchanged messages. We propose to use link layer security for BIONETS on the island level. If nodes are actively participating the network we still want to ensure the privacy, integrity, and authenticity of messages they use to disseminate data. We propose the following mechanisms.

Actually, techniques may be exploited in BIONETS to secure communications over a short range wireless channel. Think to a T-thermometer that makes its readings available to U-nodes over the air. Solutions may be find by exploiting quite general methodologies for ad-hoc networks, aiming at the the settlement of a security association between two nodes communicating through wireless technology.

Key imprinting It is the act of installing cryptographic material in mobile devices by physical contact. By following the "resurrecting duckling security policy model", [104], one can establish a secure, transient association between a sensing device (the T-node) and multiple, serialized owners (i.e., the U-nodes).

Secure side channels Beside physical contact, bootstrapping security associations between "strangers" may be achieved by exchanging a limited amount of public information over a privileged side channel, (e.g., infrared) which will then allow them to complete an authenticated key exchange protocol over the wireless, [105]. This approach has been shown to be secure against passive attacks on the privileged side channel and all attacks on the wireless link.

In the area of data diffusion protocols, the idea is to have a set of sensor nodes (T-nodes) and a sink (U-node). The sink floods the network with queries to establish gradients at each sensor.

As the answer, the data start to be sent to the sink. Depending on the delivery quality, the sink may select some specific sensors. From a security point of view, an attacker may potentially compromise a number of T-nodes, may inject forged queries and forged answers. To fight against this, a promising proposal is in [106], defining a protocol that authenticates the received sensing data by using *location-binding* keys, use encryption to establish secure gradients, and achieve a local containment of malicious traffic.

In-network aggregation is a data-processing primitive according to which sensor nodes forward data towards the sink. As sensor nodes closer to the sink receive data from nodes further away, they *aggregate* the information to significantly save energy. Security issues are in avoiding that a malicious aggregator compromises the data. [107] proposes a scheme to achieve data aggregation that is resilient against node capture, in a way that possible corrupted data have a minor effect on the whole aggregated value. [108] develops algorithms that can improve the integrity guarantees of the result returned by a sensor network where malicious nodes may be present. These solutions can be applied in BIONETS when, e.g., a U-node receiving data from a set of T-nodes must perform aggregation before passing them to some other U-nodes. Note that this discussion includes also U-node to U-node communication (see subsection 11.2.2).

SPINS, a security protocol suite proposed in [109] is also feasible to be employed at the dissemination layer in BIONETS. It provides data confidentiality, two-party authentication and time-stamping (data freshness). SPINS consists of two security building blocks: SNEP and /*micro*TESLA. SNEP provides data confidentiality, two-party data authentication, integrity, and freshness. /*micro*TESLA provides authentication for data broadcast. Apart from the above mentioned functionalities, SNEP provides replay protection with low communication overhead.

Instead of protecting the privacy and anonymity of nodes by using more or less demanding encryption algorithms we also propose to think towards new approaches based on information slicing as proposed in [110]. This would nicely complement the ongoing network coding research in BIONETS.

Service Layer Security

Due to major resource restrictions, at least for A.y and B.y classes, and due to their logical role in BIONETS, T-nodes will not provide support for complicated mechanisms such as service evolution or combination. Thus, the service(s) running on them will most likely be installed during deployment. Dependent on their application, the node characteristics they run on and its supported security functionalities, and dependent on the data they deliver, the security mechanisms protecting the service data can be chosen a priori. U-nodes communicating with T-nodes will have to adapt to the security mechanisms applied at the service layer. Appropriate hand-shake mechanisms will be needed for this purpose.

11.2.2 U-node to U-node communication

Link Layer Security

In this section we work under the same assumption that U-nodes support all the different technologies available for T-nodes. As a consequence the same choice of mechanisms and the according argumentation holds for this subsection.

Data Dissemination Security

The data collected by the U-nodes are envisioned to be disseminated to other U-nodes that are interested. Thus, security has to be achieved regarding queries for collecting certain data and answers through which they are delivered, in order to authenticate originators of these messages and encrypt sensitive information.

Actually, "mobility helps security" in dynamic, mobile, ad hoc networks, [111]. The higher mobility of U-nodes with respect to T-nodes may suggest to set up a distributed public key infrastructure among them.

The infrastructure will consent to authenticate pair of U-nodes, allowing checks on the integrity of the exchanged data, and the establishment of a secret key to encrypt data, by means of standard public key protocols to set up a secret key.

[112] proposes a technique towards that direction, within ad hoc networking, in which security associations between nodes are established, due to the fact that devices move. When they are in the vicinity of each other, they exchange appropriate cryptographic material.

Furthermore, information dissemination is also subject to cooperation between U-nodes. Indeed, storage and forwarding of data requires a cooperative behavior, while a node may want to maintain its resource for other purposes. The BUTE security group involved in BIONETS has started to investigate selfishness of U-nodes thwarting secure data dissemination, [81, 113]. The game-theoretic model presented shows that the approach indeed stimulates cooperation between nodes.

Finally, BIONETS will also employ network coding techniques. New coding mechanisms such as low complexity random linear codes [114] which provide security facing different adversary models should be adopted.

Service Security

Dependent on their knowledge and evaluation of the environment, the network, and the communicating parties U-node are more flexible in their decision what security mechanisms should be applied to protect messages relevant for service execution.

Dependent on the service, node, and data policies, the node functionalities which can provide security at the lower layers the, and the current status of a U-node (e.g. status of resources) the internal logic of a U-node (also see [15, 81]) can decide how to enforce security. For this purpose the U-node provides a security enforcement unit (SEU) which supports dynamic as well as static mechanisms for service security.

11.2.3 U-node to AP communication

Here, the same technologies as in T-node to U-node and U-node to U-node communication can be employed to protect ordinary BIONETS traffic. Indeed, it is also feasible to apply traditional, more demanding, mechanisms such as public key systems. This is due to the fact that APs can setup connectivity to central certification authorities and they may even act as a certification authority itself able to manage and securely deliver credentials. Additionally, APs may support the trust and reputation systems defined in [115] and provide storage for trust and reputation information and record more long-term information about T- or U-nodes it was connected to.

11.2.4 AP to AP communication

Within this type of communication, no link layer security is needed. Also, dissemination layer security wont be needed as we assume that the nodes in the BIONETS network take care of this.

However, to avoid that an attacker, which is not part of a BIONETS, injects numerous queries, and thus drains resources of BIONETS nodes, APs can be connected by a traditional virtual private network, VPN, to communicate confidentially. As usual, the VPN traffic can be carried over a public networking infrastructure, on top of standard communication protocols.

11.3 Exploiting AP connectivity

11.3.1 Authentication

When connectivity of a BIONETS and an access point is available, there is the possibility to use traditional PKI infrastructure to verify signatures and thus ensuring authenticity of U-nodes. Indeed, AP may act as a Certification Authority for securely managing public keys and safely deliver digital certificates.

In such a way, U-nodes may authenticate each others through AP, by, e.g., sending queries to AP about the status of a digital certificate. On the other hand, there could be a possible overhead in BIONETS when nodes are *routing* such queries over multiple hops. At a first analysis, one option to avoid this overhead may be to only allow this verification if the U-node is "directly" connected to the AP.

By using this type of interaction the sensitive issue of code update could be solved. Especially security primitives may be subject to updates. However, updating code may introduce possible vulnerabilities. If the code has not been issued by trusted authorities it may contain code which allows an attacker to access unauthorized resources. Thus, using a AP backbone may enable a U-node to verify the code origin using a PKI and enable or remove it accordingly.

11.3.2 Trust and Reputation

A trust and reputation management system is necessary in BIONETS to provide soft security management of resources. This works not as a substitute for hard security mechanisms, that are used to protect BIONETS from external and internal threats but is a complementary technique that encourages the right behavior by BIONETS components.

Deliverable 4.1, [115], shows in details the state of the art, the reputation system definition for the BIONETS architecture, reflecting the heterogeneity of the nodes, the possible applications in BIONETS and a threat analysis. The resulting system is flexible enough for dealing with a highly disconnected network, like the one under consideration is.

11.4 Threat analysis

Broadly speaking, there are three kinds of attackers menacing the network security architecture. The *outsiders*, entities not authorized to participate in the network. Given the peculiarity of the wireless links, eavesdropping, recording of any communication, spoofing, jamming attacks are possible. These are kind of threats that are faced by means of the various network security techniques considered in this document.

The *insiders* may have control overs nodes, and they mainly consist of services running those nodes, that, given their evolutionary behavior, may have changed from good natured services to malicious ones.

Finally, the inner nature of BIONETS, that is, its autonomous, self organizing behavior, may lead to the so called *destructive behavior*, a new type of passive attack. Since the network self-reacts to situations that it considers dangerous, it could happen that, while securing part of the system, the reaction results harmful for other parts.

The development of adaptive and evolutionary security solutions, along with the modeling and analysis, will be treated in WP4. Nevertheless, given the wide nature of contexts and problems, our research could not be exhaustive and what will remain to do could be tackled by other future projects.

A more thorough threat analysis of the security mechanisms foreseen to be employed in BIONETS will be provided in [81].

11.5 Conclusion

This section gave a short overview on and motivation for the various security requirements for the network outlined for BIONETS. A three layer approach was chosen for the security framework. Appropriate definitions for link, dissemination, and service security layer have been given. Based on the layer-specific requirements we analyzed the existing T-node classes according to their proposed radio technologies and security capabilities. This lead to a new two-dimensional classification of T-nodes. Additionally, this section showed how the existing security mechanisms for the proposed radio technologies could be adopted for their use in BIONETS. This section assumes that this analysis also holds for U-nodes as they should be able to support at least the functionalities a T-node can provide. We concluded by giving a brief threat analysis of the proposed high-level network security architecture.

12 Discussion and Future Work

In this deliverable, the target is threefold. First, at the level of raw network infrastructure, there exists the need to devise the set of constraints imposed to communications in the BIONETS framework. For any service running on top of the BIONETS architecture, in fact, there exist fundamental performance figures related to connectivity and scalability of the disappearing network. We addressed such issues with the aim to complement what was done before, see [14], i.e., the stress was moved to two components not yet addressed before, at least from a system-wide perspective, namely the AP and the T-node. As concerns the T-nodes, we dealt with the optimization in T-nodes data diffusion, and with issues of data retrieval by nearby U-nodes. As concerns APs, we devised integration schemes of BIONETS with existing networks.

The second target of the deliverable is to collect the results of the application of bio-inspired techniques to the network infrastructure, with particular care for the support of evolution in the networking infrastructure. We described the research performed on a general framework for online protocol evolution, in the context of data forwarding, proving that the method is feasible. Here, due to the generality of the method, room remains for investigating such a technique when it is applied to networking aspects other than forwarding. General protocol evolution was also investigated with respect to code regulation, but research is still developing, since the core focus there should be first dedicated to solve deep formal issues related to genetic operators and languages. Nevertheless, some evidence of the applicability of the concept to the networking scenario was derived with respect to the fraglets paradigm, showing an interesting research line.

The third target of the deliverable is the middleware interface towards services, reporting on one hand on the applicability to the BIONETS case of techniques from the p2p domain, and reporting, on the other hand, self-organization at the interfaces with services, analyzed with respect to the use of opportunistic forwarding. Also, an example of usage of the syntax based on `<attribute; value>`, according to specifications in [15], was reported.

Security issues were detailed in a dedicated section, spanning the whole architecure in a cross-cutting fashion.

Overall this deliverable refined on networking issues and, furthermore, provided contributions on the application of bio-inspired concepts applied even to the networking infrastructure. Some directions for future work have been identified, and in our view further work should address the following points:

- evolution-based approaches so far were applied in a "best effort" manner, meaning that so far we could not identify a clear case when evolution-based approaches do represent a practical advantage: it would be interesting to prove such a reference benchmark case when the above holds;
- the overall network architecture should be integrated: it would be interesting to infer some general properties of the joint T-nodes/U-nodes/AP architecture;
- the occasional interrogation of T-nodes data by U-nodes raised issues of in-network computation; some solutions were proposed, more advanced solutions for data download from

T-nodes are under investigation;

- the interaction of SP1 and SP3 have been initiated in this deliverable, with large room devoted to service related aspects; this direction will be pursued in a dedicated task-force activity.

13 Glossary

- **U-Nodes:** complex powerful electronic devices with computing/communication capabilities, carried around by users (hence inherently mobile) and used for running services. U-nodes interact with the environment through T-nodes, from which they gather information to run context-aware services. Environmental data or service-specific code (in order to enable service evolution) can be communicated among U-nodes when getting in proximity. PDAs, laptops, smartphones, GPS car navigation systems, car-to-car platforms represent examples of a U-Node.

- **T-Nodes:** simple, inexpensive embedded devices with sensing/identifying capabilities. They act as an interface with the environment and are needed to provide context-awareness to BIONETS services. They do not communicate among themselves but are just read by U-nodes in proximity. They present minimal requirements in terms of processing, storage, and communications.

- **Access Point (AP):** device presenting an interface for communicating with U-Nodes and an interface supporting the TCP/IP protocol suite. In BIONETS, they are used to provide internetworking capabilities between BIONETS islands and legacy IP networks/services.

- **Opportunistic Forwarding:** mechanism for diffusing information in a highly partitioned network, based on the exploitation of "contact opportunities" between nodes in the system. Opportunistic forwarding is based on localized interactions only and exploits mobility of the nodes to ensure network-wide diffusion of messages.

- **Information Filtering:** mechanism for limiting the diffusion of data messages with low information content. This is related to the fact that, in most context-aware applications, context-related data looses its usefulness when being far (in both space and time dimensions) from the originating context.

- **Name:** location-independent identifier of a logical BIONETS entity (in this deliverable: node). A name in BIONETS is constituted by a set of pairs `<attribute; value>`. Names in BIONETS are intentional and dynamic, i.e., they may change over time. As an example, a PDA belonging to, let's say, John Doe from Santa Barbara, can present as name {`<name;John>`, `<surname;Doe>`, `<residence;Santa Barbara>`}. Names can therefore be thought as profiles.

- **Address:** location-dependent identifier of a logical BIONETS entity (in this deliverable: node). Addresses can be bound to names dynamically whenever needed, and, in general, are not intentional, i.e., they do not carry any semantic content.

- **Identity:** globally unique identifier associated to each U-Node (and to *some* T-Nodes, see BIONETS D4.2). Being location-independent, an identity is technically a name (even if we will use the two terms separately to avoid confusion). As an example, we can think of

the WiFi MAC address as the identity of the node which presents it as the unique wireless interface.

- **Proxy server:** application program which services the requests of its clients by making requests to other servers.

References

[1] V. Chvatal, "Hard knapsack problems," *Operations Research*, vol. 28, pp. 1402–1411, 1980.

[2] B. Gavish and H. Pirkul, "Efficient algorithms for solving multiconstraint zero-one knapsack problems to optimality," *Mathematical Programming*, vol. 31, pp. 78–105, 1985.

[3] N. Fisher and S. Baruah, "A fully polynomial-time approximation scheme for feasibility analysis in static-priority systems with arbitrary relative deadlines," in *ECRTS '05: Proceedings of the 17th Euromicro Conference on Real-Time Systems (ECRTS'05)*, 2005, pp. 117–126.

[4] O. Ibarra and C. Kim, "Fast approximation algorithms for the knapsack and sum of subset problems," *J. ACM*, vol. 22, no. 4, 1975.

[5] D. Hochbaum, Ed., *Approximation Algorithms for NP-hard Problems*. PWS Publishing Company, 1997.

[6] U. Feige and J. Kilian, "Making games short (extended abstract)," in *Annual ACM Symposium on Theory of Computing*, 1997, pp. 506–516.

[7] C. Chekuri and S. Khanna, "A PTAS for the multiple knapsack problem," in *Symposium on Discrete Algorithms*, 2000, pp. 213–222.

[8] D. B. Shymoys and E. Tardos, "An approximation algorithm for the generalized assignment problem," *Mathematical Programming*, vol. 62, no. 1-3, pp. 461–474, 1993.

[9] R. Cohen, L. Katzir, and D. Raz, "An efficient approximation for the generalized assignment problem," *Information Processing Letters*, vol. 100, pp. 162–166, 2006.

[10] G. Anastasi, M. Conti, E. Gregori, C. Spagoni, and G. Valente, "Motes sensor networks in dynamic scenarios: an experimental study for pervasive applications in urban environments," *International Journal of Ubiquitous Computing and Intelligence*, vol. 1 (1), January 2006.

[11] Create-Net, "D1.3.2 BIONETS Simulation Framework and Initial Performance Analysis," 2007, BIONETS, IST-2004-2.3.4 - FP6-027748. [Online]. Available: http://www.bionets.org/

[12] V. Cerf, S. Burleigh, A. Hooke, L. Torgerson, R. Durst, K. Scott, K. Fall, and H. Weiss, "Delay-tolerant Network Architecture," 2005. [Online]. Available: http://www.dtnrg.org/docs/specs/draft-irtf-dtnrg-arch-03.txt/

[13] I. S. Reed and G. Solomon, "Polynomial codes over certain finite fields," *Journal of the Society for Industrial and Applied Mathematics*, vol. 8, pp. 300–304, June 1960.

[14] F. D. Pellegrini, D. Miorandi, I. Carreras, G. Alfano, D. Raz, R. Cohen, E. Borgia, J. Latvakoski, D. Schreckling, A. Panagakis, and A. Vaios, "Disappearing network infrastructure and design: Functionality and challenges," Bionets Deliverable (D1.2.1), September 2006. [Online]. Available: http://www.bionets.eu/docs/BIONETS_D1_2_1.pdf

[15] M. Tahkokorpi, J. Latvakoski, E. Borgia, A. Panagakis, V. Simon, F. D. Pellegrini, and T. Hautakoski, "D1.1.2 - architecture, scenarios and requirements refinements," Bionets Deliverable (D1.1.2), June 2006. [Online]. Available: http://www.bionets.eu/docs/BIONETS_D1_1_2.pdf

[16] P. Gupta and P. R. Kumar, "The capacity of wireless networks," *IEEE Trans. on Inf. Th.*, vol. 46, no. 2, pp. 388–404, Mar. 2000.

[17] I. Schizas, A. Ribeiro and G. B. Giannakis, "Distributed estimation with ad hoc wireless sensor networks," in *EURASIP EUSIPCO*, Florence, 3-7 September 2006.

[18] D. S. Scherber and H. Papadopoulos, "Distributed Computing of Averages over Ad-hoc Networks," *IEEE JSAC*, vol. 23, no. 4, pp. 755–764, April 2005.

[19] S. Barbarossa, G. Scutari, and A. Swami, "Distributed detection and estimation in decentralized sensor networks: and overview," in *EURASIP EUSIPCO*, Florence, 3-7 September 2006.

[20] D. Miorandi, F. D. Pellegrini, I. Carreras, G. Alfano, M. Tahkokorpi, S. Szabo, E. Borgia, J. Latvakoski, D. Schreckling, A. Panagakis, and A. Vaios, "Requirements and architectural principles: Application scenario analysis, network architecture requirements and high-level specification," Bionets Deliverable (D1.1.1), June 2006. [Online]. Available: http://www.bionets.eu/docs/BIONETS_D1_1_1.pdf

[21] E. Bottega, P. Popowsky, M. Zorzi H. Yomo and R. Prasad, "Hypothesis testing over a random access channel in wireless sensor networks," in *EURASIP EUSIPCO*, Florence, 3-7 September 2006.

[22] G. Mergen and L. Tong, "Type based estimation over multiaccess channels," *IEEE Transactions on Signal Processing*, vol. submitted.

[23] R. Gallagher, "Finding parity in a simple broadcast network," *IEEE Transations on Information Theory*, vol. 34, no. 2, pp. 176–179, March 1988.

[24] E. Kushilevitz and Y. Mansour, "Computation in noisy radio networks," in *Proc. of ACM-SIAM Symposium on Discrete Algorithms*, San Francisco, USA, 25-27 January 1998.

[25] A. Giridhar and P. R. Kumar, "Computing and Communicating Functions over Sensor Networks," *IEEE JSAC*, vol. 23, no. 4, pp. 755–764, April 2005.

[26] ——, "Towards a Theory of In-Network Ccomputation in Wireless Sensor Networks," *IEEE Communications Magazine*, vol. 44, no. 4, pp. 98–107, April 2006.

[27] M. A. Bender, M. Farach-Colton, S. He, B. C. Kuszmaul, and C. E. Leiserson, "Adversarial contention resolution for simple channels," in *SPAA '05: Proceedings of the seventeenth annual ACM symposium on Parallelism in algorithms and architectures.* New York, NY, USA: ACM Press, 2005, pp. 325–332.

[28] J. I. Capetanakis, "Tree algorithms for packet broadcast channels," *IEEE Trans. Inform. Theory*, vol. 25, no. 4, pp. 505–515, September 1979.

[29] Tsybakov, "Free synchronous packet access in a broadcast channel with feedback," *Problemy Peredachi Informatsii*, vol. 14, no. 4, pp. 32–59, Oct-Dec 1978.

[30] I. Cidon and M. Sidi, "Conflict multiplicity estimation and batch resolution algorithms," *IEEE Transactions on Information Theory*, vol. 34, no. 1, pp. 101–110, 1988. [Online]. Available: citeseer.comp.nus.edu.sg/cidon88conflict.html

[31] R. Rom and M. Sidi, Eds., *Multiple Access Protocols: Performance and Analysis.* Springer-Verlag, 1990.

[32] S. Alouf, I. Carreras, D. Miorandi, and G. Neglia, "Evolutionary epidemic routing," INRIA RR, Tech. Rep. RR-6140, 2007. [Online]. Available: http://hal.inria.fr/inria-00130803/

[33] D. Miorandi (editor), "Requirements and Architectural Principles: Application scenario analysis, network architecture requirements and high-level specification," BIONETS (IST-2004-2.3.4 FP6-027748) Deliverable (D1.1.1), June 2006.

[34] ——, "Architecture, Scenarios and Requirements Refinements," BIONETS (IST-2004-2.3.4 FP6-027748) Deliverable (D1.1.2), June 2007.

[35] R. Groenevelt, P. Nain, and G. Koole, "Message delay in mobile ad hoc networks," *Performance Evaluation*, vol. 62, no. 1-4, pp. 210–228, October 5-7 2005, proceedings of Performance 2005, Juan-les-Pins, France.

[36] A. Chaintreau, P. Jui, J. Crowcroft, C. Diot, R. Gass, and J. Scott, "Impact of human mobility on the design of opportunistic forwarding algorithms," in *Proc. of IEEE INFOCOM*, Barcelona, Spain, 2006.

[37] A. Lindgren, A. Doria, and O. Schelen, "Probabilistic routing in intermittently connected networks," in *Proceedings of the SAPIR 2004 Workshop, Fortaleza, Brazil*, ser. Lecture Notes in Computer Science, vol. 3126, August 2004, pp. 239–254.

[38] Z. J. Haas and T. Small, "A new networking model for biological applications of ad hoc sensor networks," *IEEE/ACM Transactions on Networking*, vol. 14, no. 1, pp. 27–40, February 2006.

[39] T. Spyropoulos, K. Psounis, and C. S. Raghavendra, "Spray and wait: an efficient routing scheme for intermittently connected mobile networks," in *ACM Workshop on Delay-tolerant networking*, 2005.

[40] T. Small and Z. J. Haas, "Resource and performance tradeoffs in delay-tolerant wireless networks," in *ACM workshop on Delay Tolerant Networking*, 2005.

[41] G. Neglia and X. Zhang, "Optimal delay-power tradeoff in sparse delay tolerant networks: a preliminary study," in *ACM SIGCOMM workshop on Challenged Networks (CHANTS 2006)*, September 2006.

[42] J. A. Foster, "Evolutionary computation," *Nature*, vol. 2, pp. 428–436, Jun. 2001.

[43] I. Carreras, F. De Pellegrini, D. Miorandi and H. Woesner, "Service evolution in a nomadic wireless environment," in *Proc. of WAC*, Athens, 2005.

[44] X. Zhang, G. Neglia, J. Kurose, and D. Towsley, "Performance modeling of epidemic routing," *Computer Networks*, 2007.

[45] L. Yamamoto, D. Miorandi, P. Dini, E. Altman, and H. Kameda, "D2.2.2 - framework for distributed on-line evolution of protocols and services," Bionets Deliverable (D2.2.2), November 2006. [Online]. Available: http://www.bionets.eu/docs/BIONETS_D2_2_2.pdf

[46] L. Yamamoto, "Code Regulation in Open Ended Evolution," in *Proceedings of the 10th European Conference on Genetic Programming (EuroGP 2007)*, ser. LNCS, Ebner et al., Ed., vol. 4445, Valencia, Spain, Apr. 2007, pp. 271–280, poster presentation.

[47] C. Tschudin, "Fraglets – A Metabolistic Execution Model for Communication Protocols," in *Proc. 2nd Annual Symposium on Autonomous Intelligent Networks and Systems (AINS)*, Menlo Park, USA, July 2003.

[48] J. F. Miller and P. Thomson, "Cartesian Genetic Programming," in *Genetic Programming, Proceedings of EuroGP'2000*, ser. LNCS, R. P. et al., Ed., vol. 1802, Edinburgh, Apr. 2000, pp. 121–132.

[49] M. O'Neill and C. Ryan, *Grammatical Evolution: Evolutionary Automatic Programming in an Arbitrary Language*. Kluwer Academic Publishers, 2003.

[50] K. O. Stanley and R. Miikkulainen, "A taxonomy for artificial embryogeny," *Artificial Life*, vol. 9, pp. 93–130, 2003.

[51] S. Szabo, V. Simon, S. Szabo, E. Varga, T. Csvorics, M. Berces, L. Bacsardi, B. Tujner, F. D. Pellegrini, I. Carreras, D. Miorandi, A. Panagakis, A. Vaios, E. Altman, M. Debbah, F. Baude, and A. A. Hanbali, "D1.3.1 - the initial mathematical models of new bionets network elements and algorithms," Bionets Deliverable (D1.3.1), February 2007. [Online]. Available: http://www.bionets.eu/docs/BIONETS_D1_3_1.pdf

[52] J. Huusko, J. Lahti, F. Baude, L. Ferrari, D. Linner, H. Pfeffer, and S. Steglich, "D3.1.2 - refinement of service architecture and requirement," Bionets Deliverable (D3.1.2), June 2007. [Online]. Available: http://www.bionets.eu/docs/BIONETS_D3_1_2.pdf

[53] [Online]. Available: http://ietf.org/html.charters/p2psip-charter.html

[54] K. Lua, J. Crowcroft, M. Pias, R. Sharma, and S. Lim, "A survey and comparison of peer-to-peer overlay network schemes," *Communications Surveys & Tutorials, IEEE*, pp. 72–93, 2005.

[55] S. Ratnasamy, I. Stoica, and S. Shenker, "Routing algorithms for dhts: Some open questions," in *IPTPS '01: Revised Papers from the First International Workshop on Peer-to-Peer Systems*. London, UK: Springer-Verlag, 2002, pp. 45–52.

[56] I. Stoica, R. Morris, D. Karger, F. Kaashoek, and H. Balakrishnan, "Chord: A scalable peer-to-peer lookup service for internet applications," in *Proceedings of the 2001 ACM SIGCOMM Conference*, 2001, pp. 149–160.

[57] [Online]. Available: http://www.jxta.org

[58] A. Allavena, A. Demers, and J. E. Hopcroft, "Correctness of a gossip based membership protocol," in *In Proc. of ACM Annual Symposium on Principles of Distributed Computing (PODC05)*, where?, 2005.

[59] M. Jelasity, A. Montresor, and O. Babaoglu, "Gossip-based aggregation in large dynamic networks," *ACM Trans. Comput. Syst.*, vol. 23, no. 3, pp. 219–252, 2005.

[60] P. T. Eugster, R. Guerraoui, S. B. Handurukande, P. Kouznetsov, and A.-M. Kermarrec, "Lightweight probabilistic broadcast," *ACM Trans. Comput. Syst.*, vol. 21, no. 4, pp. 341–374, 2003.

[61] Y. Chawathe, S. Ratnasamy, L. Breslau, N. Lanham, and S. Shenker, "Making gnutella-like p2p systems scalable," in *SIGCOMM '03: Proceedings of the 2003 conference on Applications, technologies, architectures, and protocols for computer communications.* New York, NY, USA: ACM Press, 2003, pp. 407–418.

[62] W. Nejdl, B. Wolf, C. Qu, S. Decker, M. Sintek, A. Naeve, M. Nilsson, M. Palmer, and T. Risch, "Edutella: a p2p networking infrastructure based on rdf," in *WWW '02: Proceedings of the 11th international conference on World Wide Web.* New York, NY, USA: ACM Press, 2002, pp. 604–615.

[63] A. Mannella, "D2.2.3 - autonomic control loop based on social modelling," Bionets Deliverable (D3.1.2), June 2007. [Online]. Available: http://www.bionets.eu/docs/BIONETS_D2_2_3.pdf

[64] E. Marocco and E. Ivov, "Xpp extensions for implementing a passive p2psip overlay network based on the can distributed hash table." [Online]. Available: http://tools.ietf.org/id/draft-marocco-p2psip-xpp-pcan-00.txt

[65] A. Loser, W. Siberski, M. Wolpers, and W. Nejdl, "Information integration in schema-based peer-to-peer networks," in *in Advanced Information Systems Engineering Proc. 15th International Conference of Advanced Information Systems Engineering (CAiSE 03).* Springer, June 2003.

[66] P. Arabshahi, A. Gray, and K. Das, "Adaptive routing in wireless communication networks using swarm intelligence," in *Proc. of 19th AIAA Int. Communications Satellite Systems Conf*, April 17-20 2001.

[67] A. Dey and G. Abowd, "Towards a Better Understanding of Context and Context-Awareness," *CHI 2000 Workshop on the What, Who, Where, When, and How of Context-Awareness*, 2000.

[68] V. Cerf, S. Burleigh, A. Hooke, L. Torgerson, R. Durst, K. Scott, K. Fall, and H. Weiss, "Delay tolerant network architecture," 2007, IETF RFC 4838.

[69] L. Pelusi, A. Passarella, and M. Conti, "Opportunistic Networking: Data Forwarding in Disconnected Mobile Ad hoc Networkso," *Communications Magazine, IEEE*, vol. 44, no. 11, pp. 134–141, 2006.

[70] A. Schmidt, K. Aidoo, A. Takaluoma, U. Tuomela, K. Van Laerhoven, and W. Van de Velde, "Advanced Interaction in Context," *Proceedings of the 1st international symposium on Handheld and Ubiquitous Computing*, pp. 89–101, 1999.

[71] T. Winograd, "Architectures for Context," *Human-Computer Interaction*, vol. 16, no. 2, 3 & 4, pp. 401–419, 2001.

[72] O. Drugan, T. Plagemann, and E. Munthe-Kaas, "Building resource aware middleware services over MANET for rescue and emergency applications," *Proc. of 16 thInternational Symposium on Personal Indoor and Mobile Radio Communications (PIMRC05). IEEE Press, Sept*, 1995.

[73] A. Lindgren, A. Doria, and O. Schelen, "Probabilistic routing in intermittently connected networks," *ACM SIGMOBILE Mobile Computing and Communications Review*, vol. 7, no. 3, pp. 19–20, 2003.

[74] P. Bellavista, A. Corradi, R. Montanari, and C. Stefanelli, "Context-aware middleware for resource management in the wireless Internet," *Software Engineering, IEEE Transactions on*, vol. 29, no. 12, pp. 1086–1099, 2003.

[75] P. Korpip, J. Mntyjarvi, J. Kela, H. Kernen, and E. Malm, "Managing context information in mobile devices," *Pervasive Computing, IEEE*, vol. 2, no. 3, pp. 42–51, 2003.

[76] Q. Huang, "Supporting context-aware computing in ad hoc mobile environments," Technical Report WUCS-02-36, Department of Computer Science and Engineering, Washington University, Tech. Rep., 2002.

[77] R. Gold and C. Mascolo, "Use of context-awareness in mobile peer-to-peer networks," *Distributed Computing Systems, 2001. FTDCS 2001. Proceedings. The Eighth IEEE Workshop on Future Trends of*, pp. 142–147, 2001.

[78] R. Popescu-Zeletin, S. Abranowski, I. Fikouras, G. Gasbarrone, M. Gebler, S. Henning, H. van Kranenburg, H. Portschy, E. Postmann, and K. Raatikainen, "Service architectures for the wireless world," *Computer Communications*, vol. 26, no. 1, pp. 19–25, 2003.

[79] A. Vahdat and D. Becker, "Epidemic routing for partially connected ad hoc networks," *Duke University*, 2000.

[80] A. Tanenbaum, *Computer Networks*. Prentice Hall Professional Technical Reference, 2002.

[81] L. Buttyan, R. Cascella, R. Damico, L. Dora, A. Garg, F. Martinella, M. Petrocchi, and D. Schreckling, "*D4.2: Towards Security in BIONETS*," University at Hamburg," BIONETS deliverable, July 2007.

[82] D. Schreckling, "*ID4.1: Security Architecture and Infrastructure Draft*," University at Hamburg," BIONETS internal deliverable, November 2006.

[83] I. Damgård and M. Østergaard, "RFID security: Tradeoffs between security and efficiency," Cryptology ePrint Archive, Report 2006/234, IACR, 2006.

[84] P. Peris-Lopez, J. C. Hernandez-Castro, J. Estevez-Tapiador, and A. Ribagorda, "RFID systems: A survey on security threats and proposed solutions," in *11th IFIP International Conference on Personal Wireless Communications – PWC06*, ser. Lecture Notes in Computer Science, vol. 4217. Springer-Verlag, September 2006, pp. 159–170.

[85] A. Juels, R. L. Rivest, and M. Szydlo, "The blocker tag: selective blocking of rfid tags for consumer privacy," in *CCS '03: Proceedings of the 10th ACM conference on Computer and communications security*. New York, NY, USA: ACM Press, 2003, pp. 103–111.

[86] S. Sarma, S. Weis, and D. Engels, "RFID systems and security and privacy implications," in *Cryptographic Hardware and Embedded Systems – CHES 2002*, ser. Lecture Notes in Computer Science, B. Kaliski, c. Kaya ço, and C. Paar, Eds., vol. 2523. Redwood Shores, CA, USA: Springer-Verlag, August 2002, pp. 454–469.

[87] S. A. Weis, S. E. Sarma, R. L. Rivest, and D. W. Engels, "Security and privacy aspects of low-cost radio frequency identification systems," in *SPC 2003*, ser. Lecture Notes in Computer Science, vol. 2802, 2003, pp. 201–212.

[88] M. Ohkubo, K. Suzuki, and S. Kinoshita, "Cryptographic approach to a privacy friendly tag," 2003. [Online]. Available: citeseer.ist.psu.edu/ohkubo03cryptographic.html

[89] S. A. Weis, S. E. Sarma, R. L. Rivest, and D. W. Engels, "Security and Privacy Aspects of Low-Cost Radio Frequency Identification Systems," in *Security in Pervasive Computing*, ser. Lecture Notes in Computer Science, vol. 2802, 2004, pp. 201–212. [Online]. Available: citeseer.ist.psu.edu/weis03security.html

[90] I. Vajda and L. Buttyan, "Lightweight authentication protocols for low-cost rfid tags," in *Ubiquitious Computing (UBICOMP)*, 2003. [Online]. Available: citeseer.ist.psu.edu/vajda03lightweight.html

[91] B. Schneier, *Applied Cryptography: Protocols, Algorithms, and Source Code in C*. New York, NY, USA: John Wiley & Sons, Inc., 1993.

[92] R. Watro, D. Kong, S.-F. Cuti, C. Gardiner, C. Lynn, and P. Kruus, "Tinypk: securing sensor networks with public key technology," in *SASN '04: Proceedings of the 2nd ACM workshop on Security of ad hoc and sensor networks*. New York, NY, USA: ACM Press, 2004, pp. 59–64. [Online]. Available: http://dx.doi.org/10.1145/1029102.1029113

[93] D. Malan, M. Welsh, and M. Smith, "A public-key infrastructure for key distribution in tinyos based on elliptic curve cryptography," 2004. [Online]. Available: http://citeseer.ist.psu.edu/713889.html

[94] W. Rankl and W. Effing, *Handbuch der Chipkarten*, 4th ed. München, Wien: Carl Hanser, 2002, vol. ISBN 3-446-22036-4.

[95] N. Sastry and D. Wagner, "Security considerations for IEEE 802.15.4 networks," in *WiSe '04: Proceedings of the 2004 ACM workshop on Wireless security*. New York, NY, USA: ACM Press, 2004, pp. 32–42.

[96] S. Fluhrer, I. Mantin, and A. Shamir, "Weaknesses in the key scheduling algorithm of RC4," in *Proc. Workshop on Selected Areas in Cryptography*, vol. LNCS 2259, 2001, pp. 1–24. [Online]. Available: citeseer.nj.nec.com/fluhrer01weaknesses.html

[97] W. Arbaugh, N. Shankar, and Y. J. Wan, "Your 802.11 wireless network has no clothes," in *Proc. Wireless LANs and Home Networks*. World Scientific, 2001.

[98] A. Stubblefield, J. Ioannidis, and A. D. Rubin, "Using the Fluhrer, Mantin and Shamir attack to break WEP," in *Proc. NDSS'02*. The Internet Society, 2002. [Online]. Available: citeseer.nj.nec.com/stubblefield01using.html

[99] "IEEE 802.11.i/D10.0 Medium Access Control (MAC) security enhancement, amendment 6 to IEEE Standard of local and metropolitan area networks part 11: Wireless Medium Access Control (MAC) and Physcal Layer (PHY) specification," April 2004.

[100] D. Johnston and J. Walker, "Overview of IEEE 802.16 security," *IEEE S&P*, vol. 2, no. 3, pp. 40–48, 2004.

[101] M. Matsui, "New block encryption algorithm misty," in *Fast Software Encryption - FSE'97*, ser. Lecture Notes in Computer Science 1267, B. Biham, Ed. Springer-Verlag, 1998, pp. 54–68.

[102] C. Karlof, N. Sastry, , and D. Wagner, "Tinysec: A link layer security architecture for wireless sensor networks," in *Second ACM Conference on Embedded Networked Sensor Systems (SenSys'04)*, 2004.

[103] M. Luk, G. Mezzour, A. Perrig, and V. Gligor, "Minisec: a secure sensor network communication architecture," in *IPSN '07: Proceedings of the 6th international conference on Information processing in sensor networks*. New York, NY, USA: ACM Press, 2007, pp. 479–488.

[104] F. Stajano and R. Anderson, "The resurrecting duckling: security issues for ad hoc wireless networks," in *Security Protocols*, ser. Lecture Notes in Computer Science, vol. 1796, 2000.

[105] D. Balfanz, D. K. Smetters, P. Stewart, and H. C. Wong, "Talking to strangers: authentication in ad-hoc wireless networks," in *NDSS 2002*, 2002.

[106] H. Yang, S. Wong, S. Lu, and L. Zhang, "Secure diffusion for wireless sensor networks," in *Broadnets 2006*, 2006.

[107] L. Buttyn, P. Schaffer, and I. Vajda, "Resilient aggregation with attack detection in sensor networks," in *Second IEEE International Workshop on Sensor Networks and Systems for Pervasive Computing (PerSeNS)*. IEEE Computer Society Press, 2006.

[108] H. Chan, A. Perrig, B. Przydatek, and D. Song, "Sia: Secure information aggregation in sensor networks," *Journal of Computer Security, Special Issue on Adhoc and Sensor Networks*, 2007.

[109] A. Perrig, R. Szewczyk, J. D. Tygar, V. Victor Wen, , and D. Culler, "Spins: Security protocols for sensor networks," *Wireless Networks*, vol. 8, no. 5, pp. 512–534, 2002.

[110] S. Katti, J. Cohen, and D. Katabi, "Information slicing: Anonymity using unreliable overlays," in *Proceedings of the 4th USENIX Symposium on Network Systems Design and Implementation (NSDI)*, April 2007.

[111] S. Capkun, J. P. Hubaux, and L. Buttyan, "Mobility helps peer-to-peer security," *IEEE Transactions on Mobile Computing*, vol. 5, no. 1, 2006.

[112] ——, "Mobility helps peer-to-peer security," *IEEE Transactions on Mobile Computing*, vol. 5, no. 1, 2006.

[113] L. Buttyán, L. Dóra, M. Félegyházi, and I. Vajda, "Barter-based cooperation in delay-tolerant personal wireless networks," in *IEEE Workshop on Autonomic and Opportunistic Communications (AOC 2007)*, 2007.

[114] S. Jaggi, M. Langberg, S. Katti, T. Ho, D. Katabi, and M. Medard, "Resilient network coding in the presence of byzantine adversaries," in *INFOCOM 2007. 26th IEEE International Conference on Computer Communications*. IEEE, May 2007, pp. 616–624.

[115] A. Garg, R. Battiti, R. Cascella, A. Montresor, and M. Brunato, "*D4.1: Trust and Reputation Management System Definition*," University of Trento," BIONETS deliverable, June 2007.