BIONETS Architecture: from Networks to SerWorks

Francesco De Pellegrini,
Daniele Miorandi
CREATE-NET
via alla Cascata, 56/C
38100 — Povo, Trento (Italy)
name.surname@create-net.org

David Linner
Technische Universität Berlin
Franklinstr. 28/29
10587 — Berlin (Germany)
david.linner@tu-berlin.de

Laszlo Bacsardi
Budapest University of Technology and Economics
Department of Telecommunications
Magyar tudosok krt 2., 1111
Budapest (Hungary)
bacsardi@hit.bme.hu

Corrado Moiso
Telecom Italia
via Reiss Romoli, 274
10148 — Torino (Italy)
corrado.moiso@telecomitalia.it

ABSTRACT

This paper presents the networking framework as conceived within the European project BIONETS. The case for such a framework comes from the idea of a "Disappearing Network" paradigm. In the disappearing network, the network ceases to exist as an independent entity and becomes an appendix of the distributed services running on user devices. The overall BIONETS system exploits the presence of embedded devices to provide context-awareness and leverages peer-to-peer interactions among mobile devices in order to ensure system-wide dissemination of data and services. Some of the current networking solutions developed within the project are also presented, highlighting the use of bio-inspired techniques and tools. The paper presents then a first version of the SerWorks architecture, which takes a unifying view on networks and services. In SerWorks, the network becomes a set of particular services, aimed at general connectivity purposes, which can be created and modified at runtime in order to accommodate for specific system conditions and higher-level constraints.

Keywords

networking framework, opportunistic networking, service-oriented networking, serworks

1. INTRODUCTION

The European project BIONETS (Biologically-inspired Networks and Services, `www.bionets.eu`) targets the introduction of nature-inspired solutions for enabling pervasive computing and communication environments.

Such kind of environments, characterized by the presence of a sheer number of devices, building an invisible electronic halo surrounding the user and supporting her/him in all her/his activities, presents a set of features that represent as many challenges to conventional approaches in networking as well as service management. In general, we can identify four main challenges to be faced. The first one is scalability, related to the possibly extremely large number of devices present in the system. The second one concerns the wide heterogeneity expected in terms of devices taking part in the system (from RFIDs to laptops etc.). The third one relates to the high expected level of dynamism in these systems, where quickly varying environmental and system's operating conditions will demand continuous adaptation capabilities. The fourth and last one concerns the complexity issues related to the management of such large-scale heterogeneous and highly dynamic system.

The BIONETS project stems from the observation that nature shows a long successful record in dealing with such problems. There are plenty of examples of large-scale ecosystems which are able to self-organize and co-evolve in such a way to reach efficient equilibria while being able to adapt to varying environmental conditions. In BIONETS, heterogeneity and scalability are mainly tackled at the network architectural level. Indeed, we rely on an architecture which is inherently scalable and at the same time is designed in such a way to natively accommodate devices heterogeneity. Complexity and dynamicity issues are addressed through the introduction of self-evolving autonomic services, built around concepts and tools inspired by the functioning of biological systems. At the same time, one of the most innovative aspects of the BIONETS project is given by the notion of joining networks and services in what we call *SerWorks*. In the SerWorks paradigm, network protocols can be generated at run-time according to the current environmental conditions as well as to the requirements of the higher-level running services. In

Bionetics'07 December 10-13, 2007, Budapest, Hungary

such a way, service-tailored network protocols can be built, achieving higher performance and providing an additional degree of flexibility to the resulting system.
In this paper, we present the networking components developed within the BIONETS framework, together with a first architecture for enabling SerWorks. The paper is organized as follows. In Sec. 2 we present the overall system architecture, focusing on service aspects. In Sec. 3 we introduce the basic BIONETS network architecture. In Sec. 4 we describe some of the networking and algorithmic solutions developed so far. In Sec. 5 we present a first SerWorks architecture. Sec. 6 concludes the paper describing some of the current on-going activities.

2. BIONETS SYSTEM ARCHITECTURE

The vision of BIONETS is to describe an architecture for autonomic computing systems that reflects all points of view on networked services, as illustrated in Fig. 1. Bio-inspired paradigms are checked for applicability on different levels, ranging from data dissemination over device interaction to service interworking. In parallel the socio-economic impacts of deploying self-organising and self-evolving systems to tomorrow's markets is investigated. The BIONETS system architecture is built on two basic classes of devices, T-nodes intended for simple reactive tasks like sensing, which could represent sensors, tags, etc., and U-nodes, an abstraction of user terminal nodes, e.g., notebooks or mobile phones. T-nodes and U-nodes communicate wirelessly, while forming transient islands of connectivity. In this regard, the BIONETS system architecture was designed to address the following requirements:

- Enabling, in a disappearing and mobile network context, the execution of "cooperative" services, involving services and data deployed/available on different U-nodes and T-nodes;
- Supporting dynamic evolution and adaptation of services in order to address changes in the execution context (e.g., network disconnections, changes in resource availability, lack of data/information);
- Dynamic aggregation/collaboration of services and data.

Therefore, the architecture is divided coarsely into three parts, like depicted in Fig. 2. The upper layer of the architecture is the Service Framework, which includes the application/services and the functions supporting their distributed and pervasive execution and management. The middle layer includes the Interaction Framework, whose purpose is to provide multiple concurrent and evolvable interaction models supporting the interactions among the distributed services and the realization of a shared data space. The lower layer is the Networking Framework, which provides the basic communication capabilities in the disappearing network context. The Interaction Framework includes a set of primitives built on top of the communication primitives, aiming at offering a common layer to simplify the implementation of the interaction models.
In BIONETS, services are defined as entities, which may provide knowledge, content, or functionality to other services and users. Hence, user applications, application services, and protocol services are summarized behind one architectural concept. Services are described as compositions of other services and service cells, which are atomic logic elements. The Service Framework provides on the one hand a runtime environment for the execution of service logic running on U-nodes and T-nodes. On the other hand, the service framework includes capabilities to react in an autonomic way to changes in the environment of services. Parts of these capabilities are different evolution and adaptation strategies, service life-cycle handling, and mobility coverage. This kind of autonomicity is supported on two different levels, at node level and at service level. Hence, the respective logic can be assigned once with each node or once with each service, while each node hosts multiple services. We can show that node level autonomicity performs better with regard to complexity and scalability, although service level autonomicity can benefit from higher flexibility of the system. To implement node level autonomicity in a flexible way, service mediators were introduced as architecture elements complementary to the services. Service mediators behave like agents for the services sharing the same node with them. However, in contrast to service mediators, services can easily move between different nodes. Services and service mediators interact through the Interaction Framework, no matter if they are hosted on different nodes or share the same one. The interaction framework contains a variety of interaction models. The interaction models have a middleware character and simplify, e.g., the coordination of service mediators. Interaction models currently investigated in the BIONETS context are the Semantic Data Space (SDS) and Distributed Hash Tables (DHT). The Interaction Framework includes several interaction models, basically realized over a common and finite set of primitives. Such primitives, in turn, are implemented on top of the Networking Framework which is in charge of communication services. Hence, the Interaction Framework decouples the Service Framework from the underlying communication protocols, naming/addressing schemata and network characteristics, as in the reference case of nodes disappearing from a connected island.
Network interfaces provide communication primitives, implemented in order to cope with the disappearing network context. The main objective of the networking framework, as detailed in the following, is to provide appropriate means for fostering the evolution at the service level in the presence of large-scale, heterogeneous and often partitioned networks. The division of the BIONETS system architecture into three frameworks allows us to reduce networking functionality and application-level functionality to the same concept, the service. Thus, we can directly apply the same biologically inspired principles that enable autonomic system behaviour on different layers of the architecture. The BIONETS principles assuring autonomy distinguish between adaptation and evolution. Likewise in biology, adaptation refers to the capacity of a given organism to sense, respond and adapt to its environment, while evolution refers to emergence of new, better adapted species in the long run. In terms of services, adaptation can be thought of as a mechanism, based on some hardwired closed-loop algorithms, that observes the environment and acts accordingly. The adaptation is the modification of already existing functionalities in order to adapt to rapid environment changes. Accordingly, the service evolution implies a long term adaptation to changes in the environment and especially the ability to acquire new functionality into the system [18, 10]. In particular, BIONETS targets the introduction of services with self-evolutionary features and

able to react in an autonomic way to the changes of state of the environment and to the (change of the) users needs in order to develop new, unplanned functionalities.
The requirements for the BIONETS services point out the necessity for a service to be able to dynamically change during its lifetime. Hence, a service must adapt its behaviour to the changing environment and conditions. Services must have the capability to evolve. In other terms, evolution is a mean to achieve adaptation. In order to fulfil these requirements, we think that the way the service is defined and implemented, and the service framework that hosts services must be sufficiently flexible. More concretely, the service architecture need to be able to support the dynamic changes of the service. The strategy used by the service to reconfigure and restructure on environmental changes may be described as an evolution process. The BIONETS Service Life-Cycle targets autonomous adaptation of services, based on a formalized model of user goals, to continuously changing environmental conditions. For this purpose, we are investigating distributed solutions to create and evolve services [13].

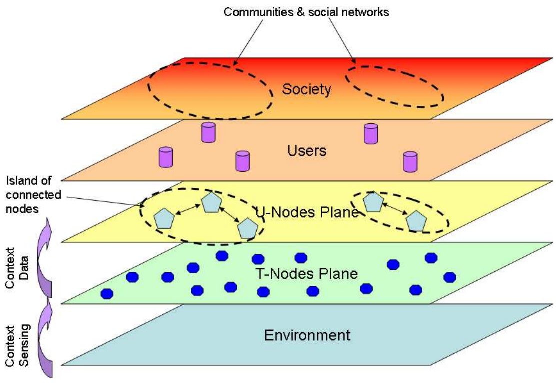

Figure 1: The BIONETS vision: contextual information is gathered through T-Nodes and used by U-Nodes — which form disconnected islands — to provide context-awareness to end-user services, which in turn reflect the social network users belongs to.

3. BIONETS NETWORKING FRAMEWORK

In this section, we present the BIONETS network architecture and the organization of the seven functional components of the networking framework, as represented in Fig. 3.

3.1 Functional System Architecture

BIONETS is built around a two-tier network architecture. As in Fig. 1, devices in the lower tier (T-Nodes) are used to interface with the environment and gather contextual information, while devices in the upper tier (U-Nodes) are used to interact with the user. In terms of role played in the network architecture, T-Nodes act as source of data, whereas U-Nodes act as source/relays/consumers of data. A major difference with, e.g., conventional approaches in wireless sensor networks, is that T-Nodes do not perform store-and-forward operations but act as a distributed interface, through which U-Nodes in proximity can interact with the local environment. The resulting two-tier network architecture is graphically depicted in Fig. 4. A third, optionally

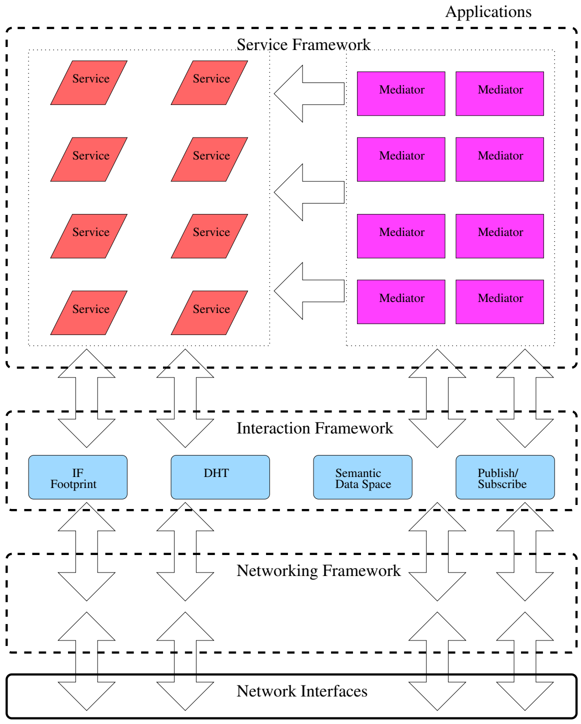

Figure 2: The overall BIONETS architectures: positioning and organization of networking, interaction and service frameworks.

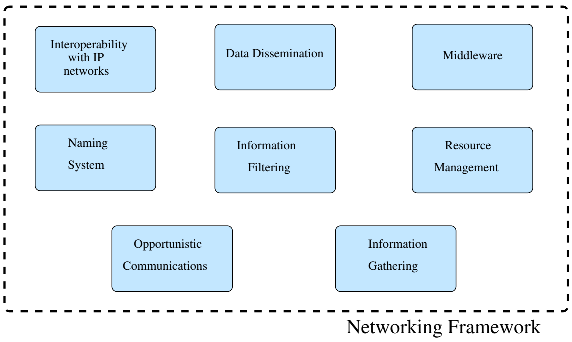

Figure 3: The BIONETS network framework and its seven functional components.

present, type of devices is encompassed, to ensure connectivity with the IP world. In details:
T-Nodes are simple, inexpensive devices with sensing/identifying and basic communications capabilities. T-Nodes act as an interface with the environment and are used to gather contextual information which is utilized by the U-Nodes to provide context-awareness. T-Nodes do not communicate among themselves but are just read by U-Nodes in proximity. They present minimal requirements in terms of processing/storage/communications.
U-Nodes are complex, powerful electronic devices with computing and communication capabilities. PDAs, laptops and smartphones are examples of a U-Node. U-Nodes are typically carried around by users and therefore are inherently mobile. U-nodes mobility is exploited, in BIONETS, to pro-

vide system-wide diffusion of messages. U-Nodes host services and interact with the environment through T-Nodes. From the environment they gather contextual information necessary to provide users with services enhanced by context-aware features. U-Nodes may communicate among themselves to exchange information, such as environmental data or service-specific code (in order to enable service evolution).
Access Points are complex powerful devices that may be used for (i) accessing IP-based services by the BIONETS networks (ii) collecting environmental data (through BIONETS system) from a remote IP service (iii) providing IP shortcuts among disconnected BIONETS islands. APs are envisioned to act as *proxies* between BIONETS networks and IP networks.

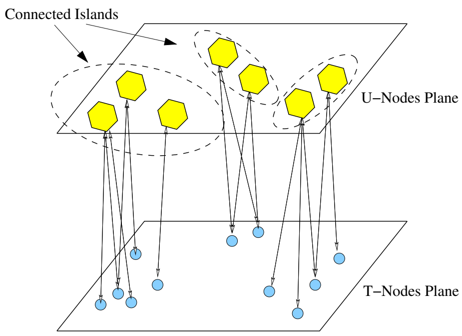

Figure 4: The two-tier BIONETS network architecture: U-Nodes communicate with both U-Nodes (for exchanging data/services) and with T-Nodes (for gathering contextual information).

3.2 Messages

Communications in BIONETS are based on the exchange of messages. Messages are service data unit, i.e., encapsulation of data items meaningful to a service. In general, messages will be much larger than standard IP packets. (This is because single IP packets usually do not expose meaningful data to the service layer.)
Messages will consist of a payload (or content) and metadata (expressed as a set of $< attribute, value >$ pairs) carrying the necessary information for the node to decide which operations should be undertaken.
Communications in BIONETS are asynchronous and connectionless. Messages are treated as datagrams, and the whole system can be thought as a message-switching engine.

3.3 Naming and Addressing

BIONETS relies on *names* for identifying communicating devices. Names in BIONETS are intentional [1] and are defined as a set of pairs $< attribute, value >$. Names are location-independent identifiers, i.e., they have global spatial scope and do not change as the node moves in the system. Names have limited temporal scope, i.e., they might change over time. All nodes in BIONETS have a name. Names are not unique. Names can be used for taking decisions concerning information/data forwarding. Their use, which complies with similar approaches in data-centric wireless sensor networks [8], enables the construction of a content-based architecture (as opposed to conventional IP address-based architectures). A special attribute field value, *tag*, can be used to enable keyword-based queries support.

Nodes which are subject to trust and reputation systems (i.e., all U-Nodes and some classes of T-Nodes) possess a *unique static* identifier called *identity*. The identity of a node has global scope in space and time. It represents a fingerprint of the node, and it is expressed as a numerical value. Node identities are assumed hardwired in the nodes by the manufacturer. Identities are not used for taking forwarding decisions, and are not exposed to the network framework. Identities are exposed to trust and reputation services only.
BIONETS encompasses also the use of identifiers with local scope in both space and time, that are termed *addresses*. An address is constituted by a numerical value, which can be associated to U-Nodes and to some classes of T-Nodes. Addresses are unique within a two-hop neighborhood. Addresses are generated locally according to a random procedure, coupled with mechanisms for resolving collisions [6]. The use of addresses is optional and is meant to provide bandwidth savings in one-hop communications by using short numeric identifiers instead of long, expressive, names.

3.4 Opportunistic Communications

BIONETS overcomes device heterogeneity and achieves scalability via an autonomic and localized peer-to-peer communication paradigm. The basic way BIONETS deals with scalability issues is by giving up stringent connectivity requirements. No conventional client-server paradigm is present: all nodes are equal (i.e., peers) and information exchanges are based on localized interactions in that local connectivity only is, in principle, present. All communication take place inside an island: for such reason, the forwarding schemes employed in BIONETS are opportunistic in nature. Also, since the engine to spread information in the network is the mobility of the devices, the right definition for BIONETS forwarding strategies is not the traditional store-and-forward, but more correctly, as in DTN networks, a store-carry-and-forward technique [7].

3.5 Data Dissemination

The data dissemination in BIONETS relies on the movement of U-nodes and on a peer-to-peer data exchange. The U-nodes, after having read the information provided by the T-nodes, undertake the task of information dissemination. Information dissemination refers to sharing the whole or part of the information with other U-nodes in the area, which may be interested in the information that is spread. Considering the distributed nature of BIONETS, a system based on local policy seems to be more suitable and effcient. Obviously, the reputation indexes together with the cooperation enforcing policy should be redefined. Information collection can be obtained simply exploiting local communications between U-Nodes during the data dissemination process.

3.6 Information Filtering

One of the key issues for enabling BIONETS-like systems is to devise efficient mechanisms for coping with the scalability issues related to the traffic carrying contextual information. In BIONETS, such information is gathered by T-Nodes and passed to U-Nodes in proximity, either in a proactive fashion (“push” mechanisms) or in a reactive one (“pull” mechanisms). Due to the expected high density of T-Nodes devices embedded in the environment, the system may generate extremely large amounts of contextual information, which may disrupt the network if not adequately managed. Mechanisms are needed to limit the ability of contextual information to spread in the system. In [4], such mechanism has been termed *Information Filtering*. The basic concept is that contextual data looses its significance and usefulness (i.e., its information content) when moving away (in space and time) from the area where it has been generated. In turn, the information content of a message can be related, through standard information-theory tools, to the number of bits necessary to encode it [5]. Ideally, we would therefore have a mechanism which shrinks the size of the messages carrying contextual data as they travel within the network. The mechanisms supporting information filtering should be distributed, i.e., implemented by each U-Node. The local decisions on the level of resolution at which data needs to be kept should be done only based on the *metadata* describing the data type and attributes. Messages will be filtered based on their metadata and stored in an internal database, present at each U-Node. The filters may be used, e.g., for blocking spam messages or for enabling communications only with trusted peers. Filters are based on a set of matching criteria on the $< attribute, value >$ pairs associated to the message. A U-Node can also access and modify a subset of the message metadata. This includes, e.g., the number of hops traversed by a message or the number of its copy already disseminated in the system.

3.7 Middleware

The BIONETS networking framework includes a middleware to create a virtual overlay network, able to decouple the interaction/service framework and the characteristics of the underlying communication infrastructure for U-nodes interactions and the corresponding BIONETS communication protocols at network layer. The design of the middleware functions is meant to extend the BIONETS network with features aiming at simplifying the implementation of the models envisaged for the interaction framework. The network middleware becomes 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 Tab. 1 we reported a set of primitives (and their description) that shall be exposed to the interaction framework. This can be regarded as the set of network framework APIs.

Primitive	Description
`publish(property, data)`	a request to store an entity in (a peer of) the overlay. Property is a set of information through which it is possible to select an entity (e.g., a key, a set of keywords, a property list, a semantic description, a service description). Data is the information associated the stored entity (e.g., a file, a document, the reference to a service/object or the service/object itself).
`deprecate (property)`	a request to deprecate an entity published in the overlay. Property is a set of information identifying the entity to be deprecated.
`search(query)`	a request to get the data associated to an entity published. Query 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.
`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 is a "filter" on the properties associated to the entities stored in the overlay, used to identify the entities which are the destination of the message. Message is the message to be provided to the selected entities (e.g., a method to be executed, a self-* message).
`aggregate(query)`	a request to optimize the data dissemination to reach (the peer storing) an entity (e.g., an object, a service, an object reference) in the overlay. Query is a "filter" on the properties associated to the entities in the overlay; it is used to identify the entities for which the operation is required.

Table 1: Network Framework Primitives

3.8 Interoperability with Legacy IP Networks

Though BIONETS systems do not rely on any infrastructure for functioning and performing the expected tasks, they can *opportunistically* exploit the presence of infrastructure-based IP networks for enhancing the quality and range of offered services. At the same time, legacy services can leverage BIONETS networks for collecting environmental data.
The internetworking capabilities with infrastructured IP networks and services are provided by BIONETS Access Points. APs operations will rely on the presence of a proxy server able to decouple the operations on BIONETS networks (which will be handled through an interface which allows communications with U-Nodes).
APs may advertise their presence by broadcasting beacon messages. A U-Node passing in proximity of a BIONETS AP can decide to register with it, generating an address which will be bound to its name. A registered U-Node will be able to communicate with the IP world only when within communication range of the AP.[1] If the U-Node moves out of the AP range, communications will be broken. Conversely, IP-based servers can access local data by sending a query to the AP, which will then translate the query and send an analogous one to the U-Nodes passing by. The IP infrastructure can also be used for building shortcuts (i.e., BIONETS tunnels) for connecting different islands of nodes. In order to achieve the latter functionality, a service for location management of the APs is needed. In particular, we may assume that all BIONETS APs shall run a service, supporting query resolution for location management and providing the necessary means for establishing a connection among two BIONETS APs. Such a service will build an overlay network, useful to route messages among BIONETS islands.

4. BIONETS NETWORKING SOLUTIONS

Mechanisms and components of the architecture described in the previous section have been resumed in [12, 14] and are detailed in the papers refereed therein. In particular, *opportunistic communication techniques* have been investigated, studying several data dissemination mechanisms. Under *epidemic spreading*, each node maintains a buffer of unsent mes-

[1]For the moment, we decided to avoid the possibility of exploiting spreading of AP-generated or AP-destined messages on the U-Nodes plane. This adheres to the BIONETS philosophy, which encompasses APs as optional elements, which provide "virtual contacts" among disconnected away islands.

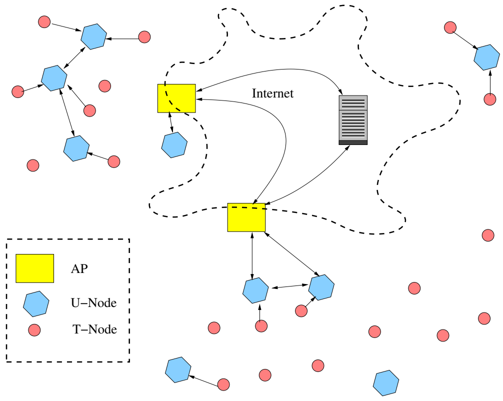

Figure 5: Graphical representation of the BIONETS network architecture, encompassing AP for leveraging IP infrastructure.

sages originated or copied from other nodes. Messages can be forwarded to nodes encountered on the move, with different policies depending on the number, source and type of messages stored in memory. With the *two-hop relay*, in particular, the source node relays a copy of a message to at most one intermediate node; the intermediate nodes that have been forwarded a copy can relay it only to the destination. Also, under the *K-relaying protocol*, each message can be replicated into a maximum number of copies K.

Under the *IOBIO* and *MIOBIO*, namely the (Modified) Information Dissemination Protocol for BIONETS, nodes use three different types of messages for information-exchange (ADV, REQ, DATA) [16]. Such control messages let IOBIO protocol limit unnecessary message replication: to this aim IOBIO uses simple periodic broadcast of ADV packets. MIOBIO, instead, adopts a controlled flood protocol that reduces the number of duplicated messages with no need for control messages, while preserving low delays and robustness of plain flooding.

Issues for U-nodes *information gathering* from T-nodes are described in detail in [14, 16]. In [16], *resources allocation* issues have been investigated in order for a U-node to be able to receive information from all T-nodes in the cluster at the same time using CDMA spreading codes. Game-theoretic tools were used in order to determine the optimal power allocation when several T-nodes transmit to a single U-node, knowing only their own channel (while perfect channel state information is assumed at the U-node). In [14] the effect of uncoordinated/coordinated dissemination of information from T-nodes to U-nodes is investigated, whereas strategies involving binary split protocols are envisioned for data recovery when U-nodes are in radio range of several concurrent T-nodes.

Information filtering [12] has been implemented by means of several distinct techniques, among which *timeouts* and *spatial constraints* in order to achieve bounded message diffusion, thus restricting the message propagation to the local scope of interest. More sophisticated techniques included the use of *wavelets* in order to obtain more efficient data representation and induce tunable granularity and memory occupancy by means of a multiscale data representation.

Furthermore, on top of the BIONETS networking functionalities, certain directions pursued biology inspired techniques within the networking context; some aspects of the communication protocols do resemble bio-inspired techniques, and this is the case of epidemic data dissemination. Nevertheless, the most promising research line for the BIONETS system concerns the adoption of *evolution* to regulate networking functionalities. In particular, so far BIONETS could devise three levels of abstraction at which evolution plays a role, which we conventionally denote order-0 evolution, order-1 evolution and order-2 evolution, as described in the following.

Order-0 Evolution

In general, we refer to order-0 evolution when the evolution acts on a predefined set of parameters that determine the system behavior. To some extent such a technique, introduced in [2, 14], allows networking schemes, i.e. in particular *forwarding schemes*, to evolve in order to adapt to changing and a priori unknown environments. The framework is inspired by genetic algorithms (GA): 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. The advantage of the approach compared to other adaptive techniques for message forwarding lies in that it does not require an *a priori* definition of the actions to be taken to optimize the mechanism for some specific situation. In brief, each node employs a (potentially different) forwarding policy, which prescribes the operations to be undertaken when receiving a message destined to another node. Such a policy is described by an array of parameters called the genotype. Genotypes are associated with a fitness measure which 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. 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.

Order-1 Evolution

What is called order-1 evolution approach has to do with the adaptive composition of functional blocks that concur to create a given networking functionality. In particular, as depicted in Fig. 3, several different functional blocks participate to the overall networking functionalities. Order-1 evolution builds on the idea of loosely coupled service components, which may be composed and orchestrated at runtime, according to the current context, in order to provide optimal performance. Such process builds on a tree structure describing the current composition model; the resulting tree is regarded as the genotype describing the current service. Tree representations undergo an evolutionary process, which will be likely based on tools and techniques developed within the genetic programming field.

Order-2 evolution

Order-2 network evolution is the most challenging form of evolution foreseen in the BIONETS domain. The aim is to support self-generating services and network protocols.

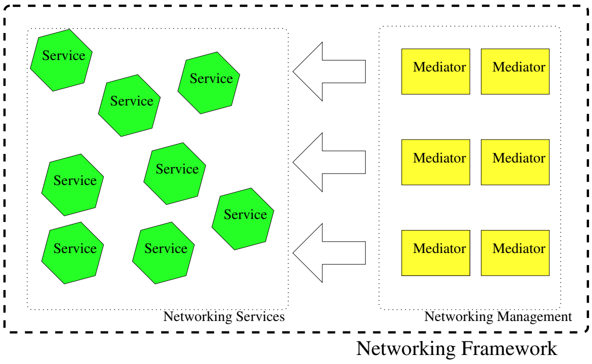

Figure 6: A first approach to networking in the BIONETS SerWorks: network services get composed at run-time to provide a service-tailored and context-aware socket.

In particular, in [10] the notion of "autocatalytic software" was adopted, 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. A promising language able to implement such autocatalytic software is given by the fraglets paradigm [17]. 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. Environmental conditions may act on the concentration of proteins, determining which genes are expressed at which concentrations. 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. The ultimate, and most challenging target is then to obtain network protocol evolution with fraglets [18].
Some preliminary experiments showed that it is possible to control code expression using regulation mechanism, but many challenging issues are pending. 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. Some directions for indirect encodings in Genetic Programming are Cartesian GP [9], Grammatical Evolution [11], and Artificial Embryogenies [15] as potential directions towards a solution.

5. SERWORKS ARCHITECTURE

As introduced before, the BIONETS network architecture supports services running on U-nodes under a disappearing network paradigm. Traditionally, network architectures have been built according to the layered OSI model, based on a stack structure. Each layer of the stack can be implemented in a separated fashion and independently of the other ones. Typically, applications leverage the communi-

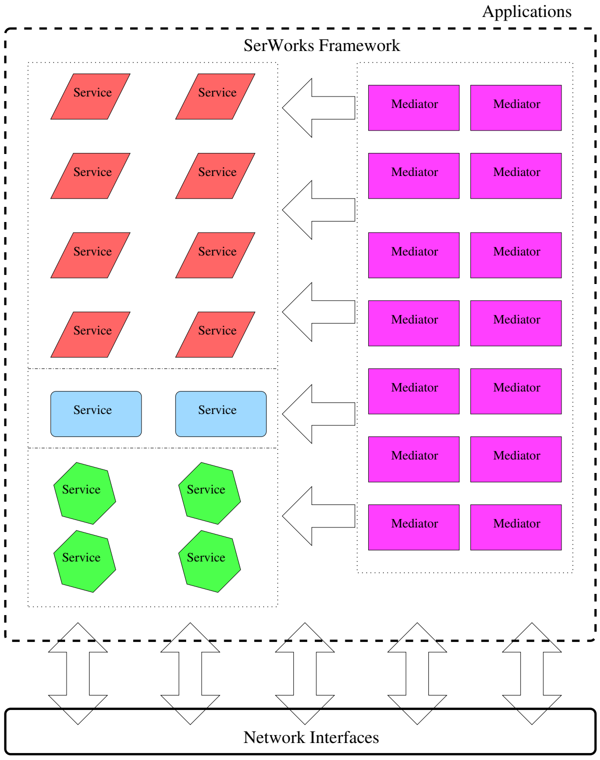

Figure 7: The SerWorks architecture: network-level and application-level services are composed at run-time according to the end-user goals.

cation pipe trough a set of executable APIs, whereas the communication/networking functionalities are part of the operating system. The layered approach has a clear advantage in that it enables (at least in principle) progressive deployment of new solutions. On the other hand, the limit is that, in current systems, the stack is hardwired in the OS and is not really modular.
We would like to stress the two main features that differentiate communication protocols compared, for example, to threads occurring among processes of a same computation or high-level services composed by instances of different programs or modules possibly running on different devices. First, communication protocols have a top-down sequential architecture where the thread occurring at the upper layer (e.g., the application layer) has to be supported by the underlying threads down to the physical bit transmission [19]: packets are passed down to the layer in charge of the "next" step until electromagnetic propagation occurs at the bit level. Thus, the overall flow is loop-free and sequential. Second, current architectures usually use at most one block at each layer at the time: for example, once established, a socket leveraging UDP cannot switch to TCP since the peer UDP module would fail in treating packets of the new session according to TCP.
Nevertheless, non-layered and dynamic architectures are possible: the advantages of such amorphous network architectures are clear to the research community [3]. But, so far, the OSI-like stack paradigm, is *the* way communication protocols are conceived and implemented. There seems to be two main factors limiting the use of dynamic architectures,

where basic building blocks get composed at run-time. The first one is computing power: communication protocols require to carry out operations at the speed at which transmissions on the physical links are possible, which can easily be of the order of hundred Mb/s. The second one is the lack of a suitable distributed run-time execution environment able to support such dynamic reconfigurable architecture.
The SerWorks vision builds on a service-oriented approach to networking protocols. Protocols can indeed be understood as the composition of various networking services. (The latter ones may be represented, e.g., by the components of the networking framework as depicted in Fig. 3.) This makes it possible to build a flexible architecture, where network-level services require input from several other services, and provide different outputs to several services. In a first approach, we may retain the functional separation between the service and the network framework. In the latter one, network services may get dynamically composed at run-time, according to the (functional and non-functional) requirements as determined by the service mediator. Such information will be used by a networking mediator to decide on the best composition of available network services. The resulting architecture is depicted in Fig. 6. From the service point of view, the advantage is clear: the possibility of exploiting a flexible data bearer; the latter can be though as an "ad hoc" socket, offering socket-like functionalities to the interaction framework but tailored to specific service needs.

The resulting architecture would encompass mediators at both the service and the network level. In order to optimize the resulting architecture, we envision to join the two entities, resulting in the structure presented in Fig. 7. In this fully integrated SerWorks architecture, three different "layers" are present. They expose well-defined APIs to the upper layer; on the other hand, the definition of the modules implementing the actual functions is left to the mediators which can decide at run-time the optimal joint configuration for the service, interaction and networking frameworks.

6. CONCLUSIONS

In this paper, we have presented the basic building blocks of the BIONETS networking framework, together with a first approach to SerWorks. On-going activities include the introduction of new bio-inspired methods for enhancing the functioning and plasticity of the single building blocks, together with studies aimed at proving experimentally the feasibility of the proposed approaches.

7. ACKNOWLEDGMENTS

This work has been partially supported by the EC within the framework of the BIONETS project IST-FET-SAC-FP6-027748, `www.bionets.eu`. The authors would like to thank all partners of the BIONETS consortium for their contributions.

8. REFERENCES

[1] W. Adjie-Winoto, E. Schwartz, H. Balakrishnan, and J. Lilley. The design and implementation of an intentional naming system. In *Proc. of ACM SOSP*, pages 186–201, Kiawah Island, SC, 1999.

[2] S. Alouf, I. Carreras, D. Miorandi, and G. Neglia. Embedding evolution in epidemic-style forwarding. In *Proc. of IEEE BioNetworks*, Pisa, IT, 2007.

[3] R. Braden, T. Faber, and M. Handley. From protocol stack to protocol heap — role-based architecture, 2002. First Workshop on Hot Topics in Networks (HotNets-I).

[4] I. Carreras, I. Chlamtac, F. De Pellegrini, and D. Miorandi. Bionets: Bio-inspired networking for pervasive communication environments. *IEEE Trans. Veh. Tech.*, 56:218–229, Jan. 2007.

[5] T. M. Cover and J. A. Thomas. *Elements of Information Theory*. John Wiley & Sons, New York, 1991.

[6] J. Elson and D. Estrin. Random, ephemeral transaction identifiers in dynamic sensor networks. In *Proc. of ICDCS*, Phoenix, AZ, 2001.

[7] K. Fall. A delay-tolerant network architecture for challenged Internets. In *Proc. of ACM SIGCOMM*, Karlsruhe, DE, 2003.

[8] J. Heidemann, F. Silva, C. Intanagonwiwat, R. Govindan, D. Estrin, and D. Ganesan. Building efficient wireless sensor networks with low-level naming. *ACM SIGOPS Operating Systems Review*, 35(5), Dec. 2001.

[9] J. F. Miller and P. Thomson. Cartesian Genetic Programming. In R. P. et al., editor, *Genetic Programming, Proceedings of EuroGP'2000*, volume 1802 of *LNCS*, pages 121–132, Edinburgh, Apr. 2000.

[10] D. Miorandi, L. Yamamoto, and P. Dini. Service evolution in bio-inspired communication systems. In *Proc. of SOAS*, 2006.

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

[12] F. D. Pellegrini (editor). Disappearing Network Infrastructure and Design: Functionality and Challenges. BIONETS (IST-2004-2.3.4 FP6-027748) Deliverable (D1.2.1), June 2006.

[13] H. Pfeffer, D. Linner, I. Radusch, and S. Steglich. The bio-inspired service life-cycle: An overview. In *Proc. of ICAS*, Athens, GR, 2007.

[14] D. Raz and F. De Pellegrini (editor). Disappearing Network Autonomic Operation and Evolution. BIONETS (IST-2004-2.3.4 FP6-027748) Deliverable (D1.2.2), June 2007.

[15] K. O. Stanley and R. Miikkulainen. A taxonomy for artificial embryogeny. *Artificial Life*, 9:93–130, 2003.

[16] S. Szabo and V. Simon (Eds.). The Initial Mathematical Models of New BIONETS Network Elements and Algorithms. BIONETS (IST-2004-2.3.4 FP6-027748) Deliverable (D1.3.1), Dec 2006.

[17] C. Tschudin. Fraglets - a metabolistic execution model for communication protocols. In *Proc. AINS*, Menlo Park, USA, 2003.

[18] C. Tschudin and L. Yamamoto. Self-evolving network software. *PIK Magazine*, 28, 2005.

[19] R. van Renesse, K. P. Birman, R. Friedman, M. Hayden, and D. A. Karr. A framework for protocol composition in horus. In *Symposium on Principles of Distributed Computing*, pages 80–89, 1995.