Technical Report

Horst Hellbrück* and Martin Lipphardt and Axel Wegener

Organic Computing Concepts in Protocol Design for Ad-Hoc Networks

Abstract: With the change from wired to wireless technologies the complexity of computer networks increases significantly. Distributed applications built on top of these wireless networks must be well designed to work in this difficult and changing environment. Especially the ad-hoc wireless communication between devices with wireless transceivers based on ad-hoc structures is still an open research field. Protocols designed in the past based on simulations solely using simple transmission models and graph theoretic assumptions fail to work reliably in real field tests. Therefore, organic computing systems that have "life-like" properties including self-adaptation and self-organization are a promising approach for new protocols in wireless Ad-hoc networks. In this paper we introduce the idea of organic computing and demonstrate how research in this field can inspire protocol design. For use in wireless ad-hoc networks we extract general organic computing principles that guide designing robust, efficient but simple protocols. Finally we will present three exemplary protocols that have been built with these principles including a summarized evaluation.

1 Introduction

During the last years a remarkable improvement in computing performance drove new applications and miniaturization of embedded systems including wireless interfaces. For instance operating systems evolve and provide auto configuration features for attaching new peripheral devices, e.g., finding and installing new wireless LANs even in an unknown environment. But this can be seen only as a first step towards a world of connected devices where even computer systems embedded into daily life objects communicate wirelessly with each other.

One example that illustrates these ideas is a completely decentralized traffic information system where vehicles build a wireless ad-hoc network and traffic data is generated by local sensors and cooperatively between vehicles. Traffic data like a traffic jam on the road is then forwarded to a subset of vehicles approaching the traffic jam. Such a system requires operation out of the box including auto configuration, self-adaptation, and self-organization.

As complexity increases with computers' growing number of interfaces and peripherals, automatic error handling and correction becomes vital for user acceptance. These systems assist users and act more like a living being than a machine. Systems which flexibly adapt to their environment in order to exhibit robust behavior are being studied in a number of research programs, among them the Autonomic Networking Initiative **?** and Organic Computing Initiative **?**. Organic computing systems are expected to exhibit "life-like" properties in the above sense of being adaptive to changing conditions. In other words, the systems have to be able to act like every living being: they have to survive and operate without being helped by a higher-level instance, such as a system administrator.

In this work we follow the definition of Wolf **?** for self-organization and emergence: "*Self-organization* is a dynamical and adaptive process where systems acquire and maintain structure themselves, without external control." "A system exhibits *emergence* when there are coherent emergents at the macro-level that dynamically arise from the interactions between the parts at the micro-level. Such emergents are novel w.r.t. the individual parts of the system."

Emergence in self-organizing systems is a basic property as the individual parts of a system are not aware of the effects at macro level but are accountable for it. We will see this emergent behavior in the example protocols towards the end of the paper.

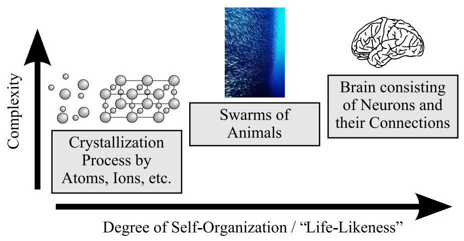
Complexity
Crystallization Process by Atoms, Ions, etc.
Swarms of Animals
Brain consisting of Neurons and their Connections
Degree of Self-Organization / "Life-Likeness"

Figure 1: Examples of Self-Organization in Nature

In this work, we focus on systems built by a huge number of devices with wireless interfaces so called wireless ad-hoc networks. Consequently, our organic phenomena that we are interested in include self-organization of molecules, cells and swarm behavior. Systems under investigation differ very

***Corresponding Author: Horst Hellbrück:** Department of Electrical Engineering and Computer Science - CoSA Center of Excellence, Lübeck University of Applied Sciences, Universität zu Lübeck, Germany
Martin Lipphardt and Axel Wegener: Institute of Telematics, University of Lübeck, Germany, Email: {lipphardt,wegener}@itm.uni-luebeck.de

much concerning their life-likeness property or what we call “degree of self-organization” in the rest of this paper. We illustrate these important characteristics in Figure 1. In physics and chemistry we find very simple kinds of self-organization where atoms, ions, or molecules are driven by strong forces to organize themselves in a grid. Moreover, swarms of animals react on environmental conditions in a more complex way, where a fish swarm moves in the sea like a single but very flexible living being. Swarms can change their shape, split into several parts and unify again. The brain itself as an organ has huge self-organization capabilities that allow hundreds of billions neurons to control the human body, steering of muscles and on-going learning with reorganization at night to allow the humans to adapt perfectly to their environment.

In the following, we will concentrate on phenomena at the lower degree of self-organization, without learning components. We will see that even designing such systems is challenging and reveals interesting emergent results. The distributed systems we build are completely without administrative central component that manages the emergent behavior of the structure, like there is no leading fish controlling its fellows throughout the swarm. The contribution of this article in the ad-hoc networking research area is as follows:

- Presentation of five major organic computing principles for ad-hoc networking to reach scalable protocols.
- Guidelines for design of ad-hoc networking protocols with these principles.
- Robust alternative protocols as examples for application of these guidelines in two major problem domains (routing and data/service dissemination).

We start with a short introduction of the principles to illustrate the idea then explain the principles in-depth following illustrating examples in the two problem domains. The goal is that readers are able to transfer the presented principles towards their own problem domain in order to enable them to design their own protocols with the approach presented in this paper.

The rest of the paper is organized as follows: The following section discusses related work and existing solutions for designing protocols and illustrates the need for new solutions. In Section 3 we derive and motivate general organic computing principles. Section 4 introduces to the problem domain. The design rules for protocols in ad hoc networks are presented in Section 5. Section 6 and Section 7 show examples designed by these guidelines. We finally address future challenges after a short summary in Section 8.

2 Related Work

This section discusses important work in organic computing systems. Related work in protocol design with focus on wireless ad-hoc networks is presented in Section 4 later.

Scientific research in organic computing can be divided into mathematical or theoretical section, the field of artificial intelligence, and the practical approaches. We discuss them in that order in the rest of this section.

The theoretical and mathematical aspects lay the foundations of organic computing. They answer questions like: What is the definition of self-organization and emergence and what is a suitable classification scheme? Can we predict the properties of a system when we know the properties of the individual parts? Can we identify properties of a system by observation of the state of the parts? Yaneer Bar-Yam has proven **?** that not all properties of a system can be identified by observation of the state of the parts. A descriptive example that he presents is a bit string with parity as a simple system. One cannot identify the parity constrained through observation of the single bits. With that constraint in mind the analysis of self-organizing systems in nature is a challenge in itself.

In the field of artificial intelligence the goal of research is to build self-learning systems that are able to self-adapt to changes of the environment. These self-learning systems like neuronal networks have emergent properties intrinsically. Especially in the field of automation, the challenge is to predict the properties of a system and implement supervised learning as the concepts of artificial intelligence fail to reliably forecast the correct behavior of a system in real applications.

These approaches of controlled learning are part of the practical solutions in organic computing **?**. A system architecture as introduced in **?** called “observer/controller architecture” provides feedback that lead to emergent properties but also control the subsystems. The observer/controller architecture can be implemented on different layers of a system and helps to develop well defined hierarchies in a complex system. It is a valuable extension of the control loop well known from control theory but with more flexibility and prospective integration of self-learning.

In our work we examine practical solutions but are motivated by the notion of emergence by Wolf **?** and by phenomena like the crystallization process, where control is decentralized and local mechanisms influence the global behavior. Especially we do not introduce hierarchy or structures like topologies or clusters or observer/controller architecture at a macroscopic level.

To the best of our knowledge designing protocols for wireless ad-hoc networks following this approach is novel. Furthermore, the application of the presented organic principles is not limited to this application domain.

3 Principles

As already mentioned above, we firmly believe that a design of protocols according to organic computing principles is advantageous for use in unstable environments like mobile ad-hoc networks. All components and the system itself have to be robust against failures and able to self-organize themselves efficiently. Learned from protocol design in the past we were able to derive the following principles as illustrated in Figure 2:

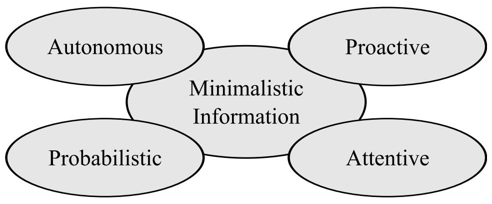

Figure 2: Organic Computing Principles

Autonomous
Nodes in the system are sovereign and operate independently without principles of delegation and roles. Nodes cannot command other nodes for some actions or service. They also cannot claim a specific role in the system like being a cluster head. However, there is some form of cooperation that we illustrate in the following principles.

Proactive
Nodes in organic computing systems operate energetic when data is available. In that sense nodes act like entrepreneurs in a market – they do not wait for others to request data but push new data in the system. E.g., when nodes are switched on in a system, they immediately start to participate actively.

Attentive
On the other side when nodes receive messages even if they are not destined to them, they are not passive but observe continuously their environment to extract relevant information and coordinate their actions. The well-known proverb "Listen before Talk" is a characteristic example of this principle. The goal is to extract a maximum of information just by listening before taking actions.

Probabilistic
As natural systems are not deterministic in the sense that the behavior of individual parts is predictable we apply the principle of randomness to our protocols. Randomization together with *attention* ensures scalability within systems even when nodes operate *autonomously* and *proactively*.

Minimalistic Information
The glue for all the before mentioned principles is the goal to reduce the information exchange between nodes to a minimum. Therefore, nodes do not depend on exact global knowledge, but often work on approximations. That implies that, e.g., the global topology of a network is not known to single nodes.

When building a distributed system we always design protocols as simple as possible to enable their usage on very resource-constrained devices. Now with beginning of the research in the field this simplicity helps to better understand and control the emergent behavior of the system. All these principles can be summarized by the following rule:
A perfect protocol is a protocol where you cannot omit a single part without affecting its basic functionality.

This approach is completely different to most research in wireless ad-hoc networks, where for basic protocols like the routing protocol AODV numerous extensions are proposed to cover special cases by adding new rules, new messages and thereby increase the complexity of the protocol. We argue in this paper that protocols need to reflect characteristics of the systems environment in their basic design.

Although we apply these principles by using simple and often static rules, the emerging system shows life-like properties, where on the one hand at the microscopic level of nodes the behavior is unpredictable and on the other hand the protocol provides macroscopic functionality reliably.

4 Problem Domain

In this section we will introduce the problem domain in wireless ad-hoc networking where we apply the above principles. We focus on transfer of data and services and distinguish between routing, data dissemination and service dissemination.

4.1 Routing

In wireless ad-hoc networks the routing problem is not solved yet. Many suggested solutions model the network as a graph where devices represent the nodes and edges are the physical links between the devices.

Thereby, the routing task is converted to a well defined mathematical problem, as already shown in Figure 3, that can be efficiently solved with graph theory. However, many assumptions, that are fundamental for the suggested protocols, do not hold for radio networks: Link quality varies with the time. Additionally, links are not always bidirectional. On the contrary, evaluation in **?** shows that unidirectional links are more a rule than an exception. Furthermore, node movement results

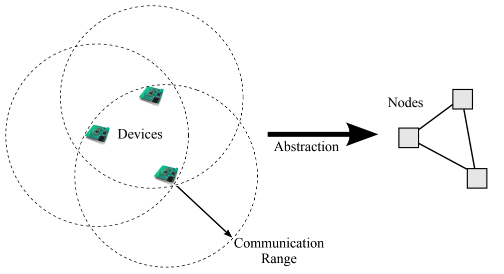

Figure 3: Mathematical Abstraction from wireless Network to a Graph

in drastic changes of the network topology and additionally in changes in location of data source and destination.

In this section we will consider unicast routing on the base of a datagram network or packet switched network. Individual datagrams or preferable packets consist of a source and destination address plus user data and are routed based on the addresses within the network. Protocols like routing protocols have their own packet format with routing information mostly called routing messages. We will use the term message for all packets that contain more than user data. The term message is more general and comprises the term packet if we do not distinguish between these two formats for the rest of this paper.

In recent years different approaches for routing have been proposed. The simplest version is flooding data packets through the whole network. This is a robust approach to deliver the data to its destination. However, this approach is very inefficient since all nodes, even those distant from source and destination, participate in the process of packet forwarding. Additionally, flooding leads to broadcast storms **?** especially in dense networks resulting in congestion of the medium and thus in low throughput.

In order to increase efficiency, approaches have been suggested to form routes consisting of a subset of available links to destination nodes and maintain these routes for future use. These algorithms can be divided into two categories *proactive* (or *table driven*) and *reactive* (or *on-demand* protocols).

Proactive protocols build and maintain routing tables independent of data traffic. Therefore, these protocols continuously generate network overhead in form of routing messages in the network. As the graph topology changes with mobility, the rate of routing messages must increase with mobility to avoid outdated entries that can lead to stale routes and thus reduces the delivery ratio. Please remember that the notion of the organic computing principle "proactive" that we introduced before has a completely different meaning.

Reactive protocols create routes only on demand, when a node sends a packet to a destination. This reduces overhead especially when only a small subset of nodes sends data packets. In order to find an unknown route to a destination node, flooding the network with so called route request messages is needed. This results in latencies in packet delivery and high peaks in network traffic, which again can lead to data loss and thus affects the delivery ratio.

More advanced protocols combine proactive and reactive approaches (*hybrid* protocols) or build and maintain hierarchical structures for routing purpose. As links break and the topology can change unpredictable continuously update and repair of routes is required that introduces additional delays during ongoing data transfer. Other protocols like AODV **?** have complex extensions in order to cope with topology changes and unidirectional links **?**, leading to heavyweight routing schemes. E.g., the official Request for Comments of AODV has about 40 pages and implementations are prone to programming errors.

Alternative classes of routing protocols exploit location information to deliver data packets. In this so called geographic routing nodes forward packets based on location information of source, destination and their own location. The idea seems to be simple and robust but in real world scenarios complex extensions are necessary to overcome dead ends and to avoid circles. If nodes transfer data based on source and destination addresses in a flat addressing scheme like in an IP subnet a mapping from IP addresses to location is mandatory that creates additional overhead. This is one of the main drawbacks since the performance of the routing depends on the quality of the location information.

To achieve robustness, gradient based routing approaches as **?** maintain a gradient over a given metric from each node towards the destination and forward the packets via multiple routes. Packets are sent together with a current cost estimate towards the destination. Direct neighbors receiving packets calculate the gradient between the cost in the packet and their own costs. For a negative gradient the packet will be forwarded by the receiving nodes. The packet follows the direction of descending costs towards the destination. Since the receiving nodes decide whether to forward the packet or not the sender does not need to maintain routes or link information. The elimination of explicit paths removes the overhead for route repair. Still the protocol suggested in **?** is dependent on advertisement messages and does not consider the problems of mobility and congestions along the gradient.

4.2 Data and Service Dissemination

Many mobile applications need to communicate in a many-to-many way without a need to set up long-lasting connections between single network nodes. The basic functionality is dissemination of data or services within a certain region or vicinity. In contrast to unicast routing of packets this task requests a dissimilar solution than for routing. First in this section we will describe data dissemination and secondly we will briefly discuss service dissemination.

The application for data dissemination that we have in mind here is a decentralized traffic information system, which forwards selected traffic information to a subset of vehicles. Thereby, traffic data can be generated by each vehicle that observes – potentially in consensus with its neighbors – critical traffic conditions and each vehicle in the dissemination area helps in further forwarding of the data, even if the content is not relevant for a particular vehicle.

In general, data dissemination in mobile ad-hoc networks can be achieved in two ways: The most obvious way is ad-hoc forwarding via ad-hoc communication between nodes. Unfortunately, a simple rebroadcasting mechanism as flooding fails for partitioned networks since communication is bounded by the nodes' communication range. An additional downside of flooding is the occurrence of broadcast storms **?** in dense networks, and a low delivery ratio in sparse networks, since messages are never sent twice by a node. The second approach is based on moving network nodes carrying data with them as long as they are in a situation to pass it on via the network **?**. The benefit of this idea is that carrying data does not consume any bandwidth. Unfortunately, delay constraints of many applications can only be fulfilled with an appropriate speed of the nodes.

Since communication is much faster than carrying data piggybacked on moving nodes, it is preferred in most scenarios. However, nodes' movement can effectively support communication when networks are partitioned, e.g., in road traffic scenarios by using opposite lane traffic to bridge gaps between cars. Thus, a hybrid solution is preferable in many applications.

In related work, most approaches target only one of these ways of operation. So, **?** summarizes different kinds of selective flooding, whereby several aspects of store-and-forward are studied, e.g., by **?**

Only a few approaches cover the two ways as also targeted by our protocol, that we will describe in Section 7:

In *adaptive flooding* **?** each node switches between pure flooding, selective flooding and a mode where it rebroadcasts all cached packets to every new node that comes into its range. Thereby, the switching between modes is based on nodes' velocity and network load. Unfortunately, the uncorrelated rebroadcasts result in a high protocol overhead in sparse networks.

In *Hypergossiping* **?** nodes perform selective flooding based on their local neighborhood size. Since a flooding process is restarted only if nodes with a very different set of last received packets get together, the rebroadcast probability needs to be kept at a relative high level to ensure that a flooding process reaches all nodes inside a network partition. In contrast, detects different status between nodes reliably and automatically restarts disseminating data if necessary.

Parameterless Broadcasting from Static to Mobile (PBSM) **?** uses a connected dominating set (CDS) that acts as network backbone. The CDS is rebuilt periodically to adapt to topology changes and allow for fairness. If new neighbors come into range, only nodes in the CDS forward data to them to avoid broadcast storms. The use of the CDS results in a simple forwarding mechanism for nodes, but at the cost of periodically building and maintaining a CDS. In contrast our approach uses a probabilistic scheme and avoids assignment and preservation of roles.

Besides data dissemination injecting and distributing functionality throughout the network is an important aspect of (re)programming sensor networks in order to achieve self-organizing properties. Service placement is used to add selective functionalities to the network in the field of service centric architectures. **?** summarizes service placement strategies only for mobile ad-hoc networks. The goal of these strategies is to place single services within a network at an optimal location determined by a given metric. In contrast, we aim to achieve a target for global service coverage within the whole network by replicating different services in order to distribute the functionality among the nodes in an organic manner.

5 Organic Computing Protocol Design

In the previous section we introduced the problem domain, discussed existing solutions and motivated the need for new solutions. In this section we will apply the organic computing principles that we introduced in Section 3. Here we describe these organic computing principles more precisely in the chosen problem domain before we discuss them in detail as applied in exemplary protocols in the remaining sections.

5.1 Autonomous Actions

This first organic computing principle defines the general interaction between the wireless nodes in the system. In our approach all nodes are equal in the sense of accomplishing the same role. If we consider the task of routing data packets between source and destination we do not establish paths where nodes send data packets to each other along this path. Roles like member of a path between source and destination node are not assigned neither explicitly nor implicitly. In contrast, the forwarding of data occurs as a consequence of autonomous decisions of intermediate nodes to forward the data. The same approach applies to service or data dissemination. Especially, we will neither introduce groups nor hierarchies between nodes.

The natural principles we are inspired from are molecules in a salt crystal. They arrange in a specific shape but without cluster heads and specific roles.

Nodes cannot command other nodes for some actions or service, which means that delegation of responsibility is not an option in our approach. Nevertheless, by this design decision we achieve robustness in the system as there is no single point of failure, or no stale information in hierarchical structure. As we see later in the sections for the example protocols scalability can be reached easily when each node is only responsible for its own status.

5.2 Proactive Behavior

We do not allow explicit subscriptions or pull messages. This is similar to natural principles: in organisms, single cells are also not able to pull required substances, at least not from a far distance. Instead the substances are brought to the cells by a push mechanism, e.g., diffusion, and cells consume the nearby substances when needed. Requests across the network like "Route Request" in path oriented routing lead to many problems when links fail. In this dynamic environment of frequent topology changes independent actions increase the robustness of the system.

5.3 Attentive Behavior

On the other side nodes are not passive but observe continuously the received messages and their environment by local sensors to extract relevant information to enable self-coordination of their actions. In our approach we exploit the broadcast nature of the wireless medium and are able to apply cross layer techniques. Therefore, we enrich packet headers with additional information to learn about neighbors and topology and observe other parameters like wireless channel usage to adapt to changing channel conditions. For the routing task, forwarding to the destination allows learning for nodes just by observing received packets. By that, nodes learn even if they are not involved in forwarding process. By implementing "Listen before Talk" we extract a maximum of information just by listening before taking appropriate actions.

5.4 Probabilistic Actions

When nodes act independently and without request and reply mechanisms as well as delegation, we need means to achieve coordination between nodes and scalability as well as saving bandwidth. If we consider routing or data dissemination in a dense network, where tens of possible forwarding nodes receive messages, we risk jamming of the wireless channel. Therefore, we suggest e.g. a probabilistic selective forwarding mechanism where randomly only a subset of possible forwarding nodes gets active. In contrast to other approaches the forwarding behavior is not deterministic and follows an organic mode, where for each message a new set of nodes cooperates to accomplish the forwarding task.

Selective forwarding can be implemented in two ways. In the first solution nodes decide on the forwarding immediately when they receive the message. In the second case nodes having received the message calculate a random forwarding delay and listen if the message is forwarded by others. If the timer expires and nobody has forwarded the message in the meantime, they will forward. However, if the message is already successfully forwarded nodes skip their forwarding. The second solution achieves reliability in message forwarding but will introduce a higher forwarding delay compared to the first mechanism. We will apply both ways of selective forwarding in the example protocols later.

5.5 Minimalistic Information

One key characteristic of our protocol design is the exchange of minimum information. For routing just the distance to the destination is used as a metric. Compared to distance vector routing packets are not forwarded along a path to the next hop but broadcasted by nodes if these nodes are nearer to the destination. Thereby, nodes forward packets probabilistic towards the destination. Moreover, the protocols we design in this work do not need accurate information but operate on fuzzy information. This is beneficial to wireless ad-hoc networks as accurate topology information is not available due to continuously changing conditions. You may argue that nodes decide on incomplete information, but information is always inconsistent in reality in wireless ad-hoc networks. Protocols we present here are designed to work with incorrect fuzzy information, compared to other protocols that depend on correct information but fail to operate when routing information is wrong or a stable path cannot be established. In particular our protocols do not rely on complete topology information but work on a simple distance estimate between nodes as routing metric or on estimates of neighbors.

Another advantage is a consequent lightweight implementation with simple message types and also simple but robust rules. This increases efficiency as no extra messages or even flooding of the network is needed.

Our design is driven by the idea to reduce the number of message types and amount of data to a minimum and achieve a reliable operation under the harsh conditions of a wireless ad-hoc network where certainty is not given. We will achieve required reliability by introducing redundancy for example in forwarding messages but make the forwarding process itself probabilistic as exact information is unachievable in wireless ad-hoc networks. In the following sections we will apply these five organic computing protocol design principles in three exemplary protocols for routing, data and service dissemination.

6 Example I - Routing

In this section we address the problem domain of Routing. We introduce **?** as the first organic routing protocol that is independent of routing messages and robust to link failures and topology changes.

As shown in the Section 4 building and maintaining structures like routes in a wireless ad-hoc network, where topology varies over the time due to node movement and nodes joining or leaving the network is complex and communication intensive. In nodes are proactive in the sense of an organic principle, learn about the network topology implicitly, and do not build routes or link states. There are no additional message types like *HELLO-Beacons*, *RouteRequests*, *LinkFailures* etc, but just data packets, which simplifies the implementation. Whenever a data packet is sent by a node, listening nodes implicitly receive information about source and destination from the packet header due to the broadcast nature of the radio medium. With increasing amount of data traffic more and more information about the current network can be gathered by the nodes. Based on this not necessarily exact information each node independently reaches its own forwarding decisions. These autonomous actions based on simple local rules are easy to implement and results in a simple robust routing scheme as we will show in the following.

A. Autonomous Actions: In packets travel along a gradient of a given metric (here we use hop-count as a simplification) from the source to the destination. To build this gradient we add small extensions to the datagram header as we will show later. exploits the broadcast nature of the radio medium, namely to reach all neighbors of the sending node. The forwarding decision is left to all receiving nodes based on a predicted hop count to the destination that is included in the packet header. Nodes compare the predicted hop count to their estimated hop count and compute a gradient to the destination. A negative result for the gradient is a strong indicator for a node to forward the data packet. In this manner packets follow the gradient from the source node towards the destination node (cf. Figure 4). As a consequence, is independent of routes or link states and forwards the packets over multiple paths.

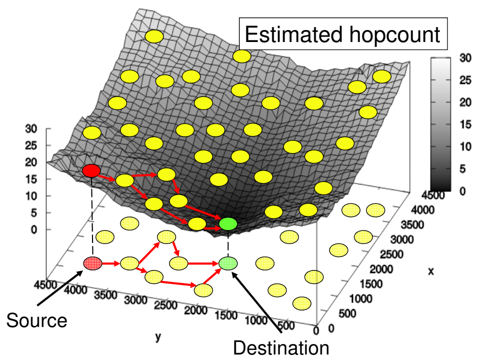

Figure 4: Gradient within an Ad-Hoc Network towards Destination

B. Proactive Beavior: To optimize the learning of the gradient, adds header fields as given in Table 1 to datagrams sent through the network and thus proactively distributes topology information among the nodes; we will use the term packet header if we refer to this data set in the rest of this section. Additionally, each node keeps a simple forwarding table with the fields *destination node address*, *estimated hop count* and a *timestamp* of the entry. When sending a packet to a destination node, the source node initializes the estimated hop count to the destination in the packet header to the value given in its forwarding table or the global constant UNKNOWN_HOP_COUNT, if it does not have an entry for the destination yet. This information is used by the receiving nodes to learn and make a forwarding decision.

Table 1: Packet Header

field	description
id	**unique packet identifier**
src	**the node address of the origin of the packet**
dst	**the node address of the destination of the packet**
fwd	**the node address of the forwarding node**
dst_hops	**the estimated hop count to the destination**
src_hops	**the current hop count of the data packet**

C. Attentive Behavior: By listening to all packets and processing the information in the packet header described in Table 1, each receiving node will update the entries in its forwarding table with newly arriving packets and modify the packet header before forwarding the data packet. The packet header contains three tuples relevant for the forwarding table:

– source address, hop count to the source (more precisely the hop count from the source to this node);
– destination address, expected hop count (or infinity if not available)

– the address of the forwarding node, implicit hop count 1 (as it is a neighbor)

An entry of the forwarding table will be updated, if the hop count in the packet header is less than or equal to the value in the entry. If the hop count to the source in the forwarding table is smaller than the estimated hop count in the received data packet, the routing layer will update the packet header and forward the data packet. By its attentive behavior forwarding nodes smoothly take over from nodes that move out of range in minimal time. In contrast AODV permanently causes peak delays, when paths break as shown in the evaluation of **?**.

D. Probabilistic Actions: A so called *Thinning* approach prevents nodes, who decided to forward a packet from unnecessary forwarding, avoiding congestions, and saving bandwidth and energy. By listening on the medium for a randomly chosen backoff time, the nodes can hear, if direct neighbors in the same distance to the destination forward the packet. If the packet is heard, these nodes choose a new random backoff time plus the maximal backoff time. By this, nodes wait for potential nodes closer to the destination to forward packets. Based on information given in the packet header each node can identify if a packet was forwarded by neighbors in the same distance to the destination or if this forwarding was successful and nodes closer to the destination transmit the packet towards the destination. In this case they can drop the packet since it was forwarded successfully. This combination of probabilistic actions and attentive behavior reduces the bandwidth usage and forces packets to travel along disjoint gradients towards the destination.

Algorithm 1 The Routing Engine

```
1: while TRUE do                                         // runs periodically
2:     receive packet $p$ and examine packetheader $ph$
3:                               // update entries in forwarding table $ft[]$
4:     $ft[ph.fwd].hops = 1$                             // forwarding node
5:     if $ft[ph.src].hops >= ph.src\_hops$ then
6:                                          // update entry from source in $ph$
7:         $ft[ph.src].hops = ph.src\_hops$
8:     end if
9:     if $ft[ph.dst].hops >= ph.dst\_hops + 1$ then
10:                                    // update entry from destination in $ph$
11:        $ft[ph.dst].hops = ph.dst\_hops + 1$
12:    end if
13:        // forward, if smaller hop count or packet is in flooding mode
14:    if $ph.dst\_hops = UNKNOWN\_HOP\_COUNT$ or
   $ft[ph.dst].hops < ph.dst\_hops$ then
15:        $ph.src\_hops{+}{+}$
16:        $ph.fwd = this$
17:        $ph.dst\_hops = ft[ph.dst].hops$
18:        forward $p$
19:    end if
20: end while
```

E. Minimalistic Information: An *Aging* mechanism for information enables the protocol in a simple manner to cope with mobility. Due to the fact that does not rely on absolute hop count, but only on the gradient (relative change) to the destination node, increasing the hop count over time will not change the protocols behaviour. This process called *Aging* preserves the gradient, but allows the nodes to reach destinations, even if they move within the network, since nodes assume that the destination is further away. By this scheme can work even in highly mobile scenarios. The evaluation shows, that for high mobility it outperforms even established routing protocols in terms of delivery ratio.

F. Design Summary: With simple local rules, the basic algorithm can be implemented in a few lines of code as shown in Algorithm 1. The processing of a packet header *ph* and thus the implicit retrieval of routing information for the forwarding table ft is described in line 4 to 12. The forwarding decision is made in line 14 and the modification of the new packet header is performed in line 15 to 18. This basic routing algorithm does not include the components *Thinning*, *Aging* for space restrictions in this paper. The complete protocol description and evaluation is published in **?**.

7 Example II - Data and Service Dissemination

In this section we introduce two organic protocols for data and service dissemination. The problem of service dissemination is addressed by the protocol, introduced in **?**. aims to achieve a uniform service distribution within a wireless multi-hop network. For a certain service the user can demand a global coverage of a service on wireless nodes, e.g., 10 % of the nodes should run a specific service. will accomplish this service distribution and preserve this value based on above introduced organic design principles. Each node uses local knowledge about the service coverage in its direct neighborhood and autonomously decides whether to provide a specific service or not. A detailed description exceeds space restrictions of this paper. Further information including evaluation in a real life test bed is available in **?**.

In order to disseminate data within an ad-hoc wireless network we recently developed the organic computing protocol **??**. In the following we focus on the organic aspects of approach and refer to the aforementioned articles for further details.

As the protocol is designed independent of data types used in the application we use the term data unit for the rest of this section if we refer to data that needs to be disseminated in the network. For a simple protocol design, we restrict ourselves to a single message format that contains *protocol information*:

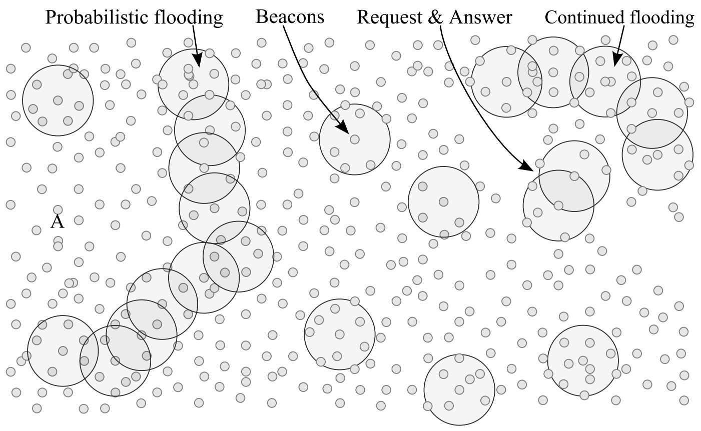

Figure 5: Illustration of AutoCast Actions.

neighborhood information, *hashes of data units*, and *data units* as described in the following. To reflect a node's state of information, hash values for data units are computed and a list of hash values is added to each transmitted message. Those hash values uniquely identify data units of the sender by adding only a fractional amount of overhead. The strategy for forwarding in is based on the locally observed neighborhood.

A. Autonomous Actions:

We distinguish between the following actions that are depicted in Figure 5: *Beacons* are sent periodically to track the neighborhood. A data unit, initially sent by node A, is forwarded by *probabilistic flooding* resulting in a wave moving through the network partition. Thereby, each node decides independently whether it participates in the flooding process considering the neighborhood and a randomized criterion so that on the average two nodes will become active. We will describe the criterion in the following. When network partitions merge as is the case in the upper right of Figure 5, *beacons* are treated as (implicit) *requests* for data units by receiving neighbors. As a consequence, receiving neighboring nodes *answer* with missing data units. Thus, the process of *probabilistic flooding* is restarted. All these actions are executed independently by each node.

For robustness we implement different delays for priority of local actions and thus achieve a protocol that reveals a stable performance in all scenarios. The delays are multiples of the time δ, that is the time interval to transmit the largest message. We distinguish the following actions and their delays:

1. Nodes answer to requests for data units after 1δ, so that actually missed data units can be added to subsequent flooding messages.
2. Nodes flood data units with a delay of 2δ, so that requests from other nodes do not interrupt this action too much.
3. Nodes request missing data units after 3δ.

To avoid jamming the wireless channel, which we experienced during design of the protocol, the strategy "Listen before Talk" is used. Therefore, actions get postponed by the aforementioned delays, every time a new message is received.

When the action with the highest priority, e.g., answering to a request is triggered, other actions are included implicitly in the message, e.g., flooding.

B. Proactive Behavior: Especially in highly mobile and sparse scenarios links between nodes break frequently as the distances between nodes increases larger than the communication range and thereby the network splits into several partitions. In order to cope with partitions uses a recovery or *store-and-forward* mechanism. Nodes piggyback data units and broadcast them later, when "new" nodes enter their communication range.

The following situations can occur at a receiving node:

– When a node finds out that the received message contains more hashes of data units than it knows itself, it can *request* the missing ones.
– When a received message does not contain hashes of data units, a node can *answer* to such an (implicit) request message with a message including those missing *requested* data units.

In both cases the node proactively pushes information in the network. The periodic beacon messages that maintain neighborhood act also as implicit request, if one or more receiving nodes stored data units, which are unknown to the sender of the beacon message.

C. Attentive Behavior:

messages are always broadcasted. Every receiving node is a potential forwarding node of the data units and uses aforementioned strategies to decide, if it actually forwards the data. So, each node processes all messages for information and data units as well as hashes of data units.

Every node is especially attentive for beacon messages sent by all nodes. If a beacon message is received by a node, the node is inserted in the local neighborhood table. The next expected beacon from this new neighboring node is scheduled due to the beacon interval given in the beacon. A node is removed again from the neighboring table if its scheduled beacon did not arrive in time.

If a node decides to answer, request or flood data units, the message also includes the beacon information. Thus, a periodic beacon without data units is only required, if the node performs no other action for a duration of the beacon interval.

D. Probabilistic Actions: Since every node is aware of the number of its direct neighbors, it adapts its data dissemination strategy to the number of its neighbors. Thus, it is not important who exactly sends out the data as long as nodes, who receive a new data unit forward it. We balance the protocol, so that on the average two nodes out of this 40 % forward new data units by means described in **?**.

In order to optimize the dissemination process, it is preferable that neighbors with the largest distance to the sending node forward the data unit. Therefore a forwarding node adds a set of at most 20 randomly chosen neighbors' ids to the message. When receiving this neighborhood information each node calculates the fraction $\frac{\textit{ratio of new neighbors}}{\#\,\textit{neighbors}}$. The lower this value the more likely nodes take actions.

E. Minimum Information: exploits local neighborhood information as metric for the data dissemination strategy of a node. Neighborhood changes continuously due to movement of the nodes. By choosing the optimal size for beacon interval as described in **?**, the deviation between the real and observed neighborhood can be bounded to a maximum. Due to this fast and reliable neighborhood detection the dissemination strategy adapts perfectly to local network densities.

F. Design Summary:

The evaluations in **??** show that is a stable protocol that adapts to dense and sparse scenarios seamlessly.

Furthermore, the primary goal of reaching as many nodes as possible is reached; data units are delivered to nearly all nodes in their specific dissemination areas. Thereby it saves bandwidth by using hashes of data units to detect the knowledge of surrounding nodes. The protocol is totally self-organizing without any prior configuration parameters necessary.

8 Conclusion & Future Work

In this paper we showed the need for a new protocol design for wireless ad-hoc networks where we apply organic computing principles. We derived general principles for protocol development following organic computing research that we applied in three exemplary protocols in wireless ad-hoc networks. The guideline for our design is: Creating a perfect protocol by reducing the protocol parts like types of messages and protocol data to a minimum. It is considered perfect if we cannot omit a single part of the protocol without sacrificing its basic functionality. We always question the need of special messages to handle special situations. We exploit the broadcast nature of the medium by attentive observing of the nodes and let nodes autonomously take actions in a probabilistic manner. To cope with the changing conditions in a wireless ad-hoc network we introduce redundancy and recovery mechanisms locally.

We demonstrated the efficiency and the potential of this general purpose approach by presenting three examples , and . In comparison to existing solutions and theoretical optimal reference and flooding approaches they perform very well and stable. Although all protocols run simple algorithms with reduced message types compared to existing solutions they prove to operate efficiently and show organic life-like behavior by self-adapting and self-organizing towards changing conditions.

We are currently working on the challenges that were not addressed in this paper so far. Those comprise the formal proof of an optimal solution to routing and service/data dissemination in the problem domain. Additionally, we are currently implementing new self-adaptable protocols for different problem domains by applying these organic computing principles presented in this paper.

Acknowledgments

The authors would like to thank the DFG for the funding of the projects AutoNomos and Aesop's Tale.

References

J. O. Kephart and D. M. Chess, "The vision of autonomic computing," *Computer*, vol. 36, no. 1, pp. 41–50, Jan. 2003.

H. Schmeck, "Organic Computing – vision and challenge for system design," in *Proceedings of the International Conference on Parallel Computing in Electrical Engineering*. Los Alamitos, CA, USA: IEEE Computer Society, 2004, p. 3.

T. De Wolf and T. Holvoet, "Emergence versus self-organisation: Different concepts but promising when combined," *Engineering Self-Organising Systems*, vol. 3464, pp. 1–15, 2005.

Y. Bar-Yam, "A mathematical theory of strong emergence using multiscale variety," *Complexity*, vol. 9, no. 6, pp. 15–24, Aug. 2004.

C. Müller-Schloer and B. Sick, "Emergence in organic computing systems: Discussion of a controversial concept," in *Proceedings of the Third International Conference on Autonomic and Trusted Computing*, Wuhan, China, Sep. 2006, pp. 1–16.

G. Zhou, T. He, S. Krishnamurthy, and J. A. Stankovic, "Impact of radio irregularity on wireless sensor networks," in *Proceedings of the 2nd international conference on Mobile systems, applications, and services (MobiSys '04)*. Boston, MA, USA: ACM, 2004, pp. 125–138.

Y.-C. Tseng, S.-Y. Ni, Y.-S. Chen, and J.-P. Sheu, "The broadcast storm problem in a mobile ad hoc network," *Wireless Networks*, vol. 8, no. 2/3, pp. 153–167, Mar. 2002.

C. Perkins, E. Belding-Royer, and S. Das, "Ad hoc On-Demand Distance Vector (AODV) Routing," RFC 3561 (Experimental), Jul. 2003. [Online]. Available: http://www.ietf.org/rfc/rfc3561.txt

V. Ramasubramanian, R. Chandra, and D. Mosse, "Providing a bidirectional abstraction for unidirectional ad hoc networks," in *Proceedings of the 21st Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, vol. 3, 2002, pp. 1258–1267.

F. Ye, G. Zhong, S. Lu, and L. Zhang, "Gradient broadcast: a robust data delivery protocol for large scale sensor networks," *Wireless Networks*, vol. 11, no. 3, pp. 285–298, May 2005.

M. Grossglauser and D. N. C. Tse, "Mobility increases the capacity of ad-hoc wireless networks," *IEEE/ACM Transactions on Networking*, vol. 10, no. 4, pp. 477–486, Aug. 2002.

B. Williams and T. Camp, "Comparison of broadcasting techniques for mobile ad hoc networks," in *Proceedings of the 3rd ACM international symposium on Mobile ad hoc networking & computing (MobiHoc)*. Lausanne, Switzerland: ACM, 2002, pp. 194–205.

K. Viswanath and K. Obrazcka, "An adaptive approach to group communications in multi hop ad hoc networks," in *Proceedings of the 7th International Symposium on Computers and Communications (ISCC)*. Washington, DC, USA: IEEE Computer Society, 2002, pp. 559–566.

A. Khelil, P. J. Marrón, C. Becker, and K. Rothermel, "Hypergossiping: A generalized broadcast strategy for mobile ad hoc networks," *Ad Hoc Networks*, vol. 5, no. 5, pp. 531–546, 2007.

A. A. Khan, I. Stojmenovic, and N. Zaguia, "Parameterless broadcasting in static to highly mobile wireless ad hoc, sensor and actuator networks," in *Proceedings of the 22nd IEEE International Conference on Advanced Information Networking and Applications (AINA)*. Ginowan, Okinawa, Japan: IEEE Computer Society, 2008, pp. 620–627.

G. Wittenburg and J. Schiller, "A survey of current directions in service placement in mobile ad-hoc networks," in *IEEE International Conference on Pervasive Computing and Communications*. Los Alamitos, CA, USA: IEEE Computer Society, 2008, pp. 548–553.

M. Lipphardt, H. Hellbrück, A. Wegener, and S. Fischer, "GRAPE - Gradient based Routing for All PurposE," in *Proceedings of the 4th IEEE International Conference on Mobile Ad-hoc and Sensor Systems (MASS)*, Pisa, Italy, Oct. 2007, pp. 1–6.

M. Lipphardt, J. Neumann, S. Groppe, and C. Werner, "DySSCo - A protocol for Dynamic Self-organizing Service Coverage," in *Proceedings of the 3rd International Conference on Self-Organizing Systems*, 2008, pp. 109–120.

A. Wegener, H. Hellbrück, S. Fischer, C. Schmidt, and S. P. Fekete, "Autocast: An adaptive data dissemination protocol for traffic information systems," in *Proceedings of the 66th IEEE Vehicular Technology Conference Fall 2007 (VTC2007-Fall)*, Baltimore, USA, Oct. 2007, pp. 1947–1951.

H. Hellbrück, A. Wegener, and S. Fischer, "Autocast: A general-purpose data dissemination protocol and its application in vehicular networks," *Ad Hoc & Sensor Wireless Networks journal (AHSWN)*, vol. 6, no. 1–2, pp. 145–122, 2009.