J Netw Syst Manage (2017) 25:416–456
DOI 10.1007/s10922-016-9398-4

Supporting Unified Distributed Management and Autonomic Decisions: Design, Implementation and Deployment

Carlos Ferreira[1] · Lucas Guardalben[1] · Tomé Gomes[1] · Susana Sargento[1] · Paulo Salvador[1] · Daniel Robalo[2] · Fernando J. Velez[2]

Received: 29 July 2015 / Revised: 22 October 2016 / Accepted: 27 October 2016 /
Published online: 31 October 2016

Abstract Nowadays, the prevailing use of networks based on traditional centralized management systems reflects on a fast increase of the management costs. The growth in the number of network equipments and services reinforces the need to distribute the management responsibilities throughout the network devices. In this approach, each device executes common network management functionalities, being part of the overall network management platform. In this paper, we present a Unified Distributed Network Management (UDNM) framework that provides a unified (wired and wireless) management network solution, where further different network services can take part of this infrastructure, e.g., flow monitoring, accurate routing decisions, distributed policies dissemination, etc. This framework is divided in two main components: (A) Situation awareness, which sets up initial information through bootstrapping, discovery, fault-management process and exchange of

✉ Lucas Guardalben
guardalben@ua.pt

Carlos Ferreira
cmf@av.it.pt

Tomé Gomes
tomegomes@ua.pt

Susana Sargento
susana@ua.pt

Paulo Salvador
salvador@ua.pt

Daniel Robalo
robalodaniel@hotmail.com

Fernando J. Velez
fjv@ubi.pt

[1] Instituto de Telecomunicações, University of Aveiro, Aveiro, Portugal

[2] Instituto de Telecomunicações-DEM, Universidade da Beira Interior, Covilhã, Portugal

management information; (B) Autonomic Decision System (ADS) that performs distributed decisions in the network with incomplete information. We deploy the UDNM framework in a testbed which involves two cities (≈250 km between), different standards (IEEE 802.3, IEEE 802.11 and IEEE 802.16e) and network technologies, such as, wired virtual grid, wireless ad-hoc gateways, ad-hoc mobile access devices. The UDNM framework integrates management functionalities into the managed devices, proving to be a lightweight and easy-respond framework. The performance analysis shows that the UDNM framework is feasible to unify devices management functionalities and to take accurate decisions on top of a real network.

Keywords Autonomous management · Mesh networks · Cooperative decisions · Distributed decisions · Network management · Decentralized management

1 Introduction

The multiple network technologies have grown on diversity mainly to support communications for multi-tenancy and stringent network services and applications (e.g. voice and video). A unified distributed network management is one that integrates, for example, wired and wireless components that share common distributed network management functionalities whenever possible. This unification may be distributed and results in increasingly complex networks composed by: (1) different technologies (wired or wireless); (2) nodes that can be fixed/static or mobile; (3) nodes that can be exposed to different load requirements; (4) nodes that may enter or leave the network; (5) multiple vendors, standards, technologies and protocols, etc. Unified distributed network management approaches need more automation and integration of the management functionalities, as well as efficient decision tasks for supporting the myriad of different network technologies.

The current trend of distributed management demands the embedding of management functionalities within the network nodes [1, 2]. However, this embedding requires the network nodes to have sufficient information to take management decisions in an efficient and cooperative way. This information needs to be exchanged, which may increase the network overhead to unbearable limits. Therefore, new management approaches—reducing the use of network resources to a minimum—need to be developed in order to control the network nodes and determine the interactions between them, enabling the optimization of the network information exchange to take accurate decisions.

Transferring the management entities from the central system to the network equipments can distribute the management throughout the network. However, the workload distribution creates new requirements, such as network bootstrapping configuration, situation awareness and decision quality. To address such issues, the network equipments may implement autonomous self-properties [3, 4]. So, in distributed autonomic capabilities, intelligence is pushed from the central management system to the network equipments.

Nodes will be able to learn and cooperate with their neighbours, improving decisions and thus, reducing the need for human intervention, allowing faster

reaction times to network changes. This requires that all network elements have the required information to take efficient management decisions. Since it is important to reduce the management traffic in the network, it is assumed that the network elements only have information about their neighbourhood a few hops. We then have to empower the network with mechanisms that are able to take management decisions, approaching the optimal ones, but resorting to incomplete network information.

The evolution of multimedia entertainment worldwide has brought an accumulating complexity in management tasks [5–10]. Through the diversity of smart devices, interactive applications and services, management approaches need to have the ability to optimize the devices communication without any extra costs. To address those challenges, new lightweight approaches need to be developed in order to organize the network devices and determine the interactions between them, to enable optimized network information exchange and autonomic decisions. Driven by the trend of distributed network management, efforts to unify the management functionalities on top of multiple network technologies are supported by companies such as Cisco [11], Microsoft [12] and HP [13]. The main problem of these approaches, specially [11, 13] is that they use conventional mechanisms for routing and exchange of management information, such as the case of OSPF and CDP routing protocols. OSPF and CDP have a high overhead and convergence time to discover/update their management information. This fact encouraged us to propose a new lightweight protocol which optimizes the way in how management information will be collected and exchanged, by ensuring that each device has the possibility to distribute common network management functionalities with low delay and network load.

The solutions are either proprietary, or based on variations of Simple Network Management Protocol (SNMP) protocol [14], maintaining an aggregate high cost as well as impacting in the scalability of the network. Thus, developing distributed solutions to unify the management functionalities on top of multiple network technologies remains a relevant research challenge.

In this article we present a Unified Distributed Network Management (UDNM) framework which is divided in two main components:

A. Situation awareness, which sets up initial information through bootstrapping, discovery, fault-management process and exchange of management information through the advantages of the use of eyesight perspective [2, 15]. The eyesight perspective aims to narrow the directions through which neighbour devices (acting as end-points or relays) are chosen to continue the dissemination of management information. Topology management and link failure make part of the situation awareness process, which is the process of being aware of what is happening to entity surroundings, such as a new node arrival in the network, link failure and network topology changing events. Therefore, in the work published in [15], we assessed this eyesight perspective through the Neighbours Eyesight Direction (NED) dissemination approach to disseminate management information in simulated scenarios. This work proposes an architecture for autonomic and distributed decisions, and integrates the dissemination

framework with the decision framework in a unified one. Moreover, this is the result of a joint work with an operator and its respective use case, to show in practice how these distributed control and management environments perform in real networks. Thus, in our use case network nodes are always exposed to real changing load requirements and QoS requests. The discovery, dissemination and autonomic decision processes are integrated in a real prototype and they are tested in a distributed approach, involving highly cooperation with the neighboring nodes.

B. Autonomic Decision System (ADS) with Learning capabilities based on Artificial Intelligence concepts, resorting to reinforcement learning mechanisms and exploration to provide distributed network management decisions with incomplete network information. The autonomic decision system receives the topological management information from the situation awareness and management exchange modules, and acts to perform accurate decisions with specific requirements.

The novelty of our approach is to consider a lightweight basis for data management collection and dissemination through the situation awareness process as well as further integrate this lightweight basis with an autonomic decision system in a real prototype. This integration resulted in a lightweight framework for data management dissemination and autonomic decisions.

We assess the UDNM framework on top of a multi-network technologies testbed in two cities (approx. 250 km far from each other), supporting IEEE 802.3 and IEEE 802.11 standards and network technologies, such as, wired virtual grid, wireless ad-hoc gateways, ad-hoc mobile access devices, where the network devices are exposed to the most diverse conditions and events, such as device and link errors, different traffic loads and network requirements. The performance results show that the UDNM framework is feasible to unify devices management functionalities and to take accurate decisions on top of a real network.

The remaining of this article is organized as follows. The related work is described in the Sect. 2. Subsequently, Sect. 3 addresses the requirements to perform unified distributed management and autonomic decisions, and it also shows the proposed UDNM framework interactions and components dynamics. Section 4 describes the implementation of the UDNM framework. Section 5 demonstrates the multi-network technologies testbed and the experimentation results. Finally, Sect. 6 concludes the article.

2 Related Work

The Internet has doubled in size in the last decade. This tremendous growth has brought an accumulating complexity in management tasks [6, 16].

Autonomic networking provides the capability to enable autonomic network management by empowering the network components to automatically adjust to the changing network environment. The autonomic network concepts aim to bring autonomic computing principles to communication networks, mainly on

management policies which can be disseminated through centralized [17] or decentralized [18] approaches, depending on how complex the managed scenarios are [19].

In Nobre and Granville [18] and Nobre et al. [19] the authors propose a distributed, scalable and robust mechanism to maintain the consistency of policy states through decentralized autonomic network management mechanism based on peer-to-peer interactions. To keep the integrity of knowledge bases, the authors propose to use multi-agents. This approach delivers good results in the performance evaluation, but it is still necessary to evaluate more complex scenarios considering both wired and wireless technologies. UDNM uses NED function to keep the integrity of knowledge bases. The NED shows to be lightweight and responsiveness in multi-technology scenarios, including mobile wireless nodes.

In [17], the authors proposed CSAMP which uses a centralized control point, taking control on both routing state and the traffic matrix of the network. The central coordinator is in charge of periodically computing the subset of flows. CSAMP needs a detailed information about the network to calculate the traffic matrix, which is a clearly disadvantage in large-scale scenarios. UDNM monitors the topology and link state information of the network in a fully distributed way. The topology information is generated according to the nodes partial view, in number of hops dealing with this large scale requirement.

The authors in [20] propose a PBM-based mechanism to facilitate scalable management, combining design and theory with testbed implementation and simulation studies. The availability of policies has been increased with the design and implementation of the Distributed Policy Repository (DPR). However, multihop support decisions is the main problem of this approach. UDNM implements a multihop decision mechanism, where the nodes have the possibility to require partial information of the network, without compromising the autonomic network decisions.

In Andrea et al. [21] the authors present a decentralized and scalable coordination system aimed at solving the monitoring of flows problem, DECON. However, DECON architecture is assessed by simulation, and the authors considered only Chord Ring topology for their assessments. UDNM is implemented as a real prototype and it is assessed both in wired and wireless scenarios, considering as well mesh and ad-hoc topologies in its performance assessments.

The work proposed in [22] explores how to repair a failed storage node, so as to minimize the system repair cost. Despite the good results in terms of minimizing repair cost, the authors assessed their solution only in simulated environments. UDMN implements a topology repair mechanism through the NED function according to real failures injected on purpose in the network.

Network management has become extremely challenging with the relentless growth in network size, traffic volume, and service diversity. With the emerging of new online services and increased demand by the clients, communication networks need to be agile and intelligent enough to cope with the future network challenges. Even with the newest developments and concepts in network management, such as Software Defined Networks (SDN) and Network Functions Virtualization (NFV) which have a deeply embedded centralized management concept, they will need to

provide the necessary agility and performance to answer the future network challenges.

The SDN concepts aim to provide a cost reduction of the network operation. By removing core functionalities from the network equipments, the cost of manufacturing is reduced. But, by removing such core functionalities, the reaction time of the network towards nondeterministic events, such as link loss, is increased above the 50 ms which is beyond the commonly accepted. The delay increases because it is necessary to detect the failure, send the event to the central controller, wait for a decision and reconfigure the network to implement a solution [23]. UDNM implements a fully distributed autonomic decision system, which remains below 50 ms response time without resorting to centralized controllers. Nevertheless, UDNM concepts can be implemented on the SDN architecture.

The work proposed in [24, 25] describes an implementation guide for an emerging standard for autonomic management and control of networks and services, namely the ETSI AFI GANA Reference Model for Autonomic Networking, Cognitive Networking and Self-Management. Network control in emerging SDN frameworks is based on a "centralized control paradigm" that leaves the fundamental end-to-end transport network elements with too little intelligence to self-manage, so as to address problems that rather require the network elements to collaborate in a distributed fashion via distributed control algorithms (e.g. optimization algorithms). For example, experiments with OpenFlow based controllers have shown that, relying only on centralized control while removing intelligence from the fundamental network elements makes the network less robust when connection to the controllers is lost. UDNM will be able to integrate its concepts in SDN and improve its scalability.

On the other hand, the propagation of information is fundamental to most of the network processes, such as routing, monitoring and management. Besides the wide set of approaches in the literature for dissemination of management information, most of them are based on variants of flooding-based [26]; probability-based [27] to reduce the amount of dissemination messages; MCDS-Based [28] to define a sequence of stable connected dominating sets; location-based [29] to disseminate only to specific locations; epidemic-based [30] to opportunistically disseminate through the overall network; and cluster-based [31] to limit the dissemination inside network clusters. In this paper we will focus on approaches that consider partial view information of the network, thus reducing substantially the amount of available approaches [32].

In [33] the authors propose HyParView: a membership protocol for reliable gossip-based broadcast. HyParView uses gossip-based interaction to collect and maintain partial view of the nodes, ensuring high levels of reliability even in the presence of high rates of node failure. However, each node maintains two distinct views, thus a larger number of passive views will be generated when a failure is detected. UDNM in comparison to HyParView maintains a local nodes partial view which is periodically updated, maintaining the management information (even larger than one hop) updated on both wired and fixed/mobile wireless scenarios.

The work proposed in [34] investigates issues related to network topology inference with partial information. The authors proposed the iTop, an algorithm for

inferring the network topology when only partial information is available. iTop outperforms previous approaches and its inferred topologies are within 5% of the original networks. UDNM framework in comparison to iTop is a prototype implemented and assessed in both wired and wireless scenarios. Therefore, the partial information gathered by the NED function provides necessary information to the autonomic decision system, improving so far its decisions accuracy.

Therefore, some routing protocols integrate also discovery functionalities in order to create a topology list, such as Open Shortest Path First (OSPF) [35] variants: over Multipath [36] and MPR Extension [37], Cisco Discovery Protocol (CDP) [38], Optimized Link State Routing Protocol (OLSR) [39] and Hierarchical OLSR [40], and BATMAN [41], which consume large overhead to keep the network topology updated [2], or were not designed to operate on heterogeneous scenarios. We have identified the following management issues on using traditional routing protocols for hybrid network:

1 High overhead and convergence time to maintain/update routing information tables;
2. The routing information collected is restricted to manufactured products and thus, it is a proprietary solution which lacks the flexibility to implement new routing functionalities;
3. Within the routing process it is important to determine the best neighbors association, especially in wireless environments, where the management information may be used to perform accurate routing decisions.

As an overview, (1) most of dissemination approaches use non-controlled flooding to perform discovery and dissemination of management information, which is less efficient with increasing number of network nodes; (2) most of the presented works oversight the importance of bootstrapping as the initial warm-up of the network or specific new entity; (3) unified approaches use heavy-weight basis for exchanging management information; (3) the related frameworks presented limitations on performing a complete distributed network management with autonomic decision support in heterogeneous networks; (4) there is lack of autonomic decision approaches that take multihop decision with partial management information in wired and wireless scenarios. These drawbacks motivate this work direction proposed and evaluated in this article.

3 Unified Distributed Management: Requirements and Deployed Framework

Figure 1 depicts a comparison between network management approaches. In the traditional network management, the administrator of the network has the central control of management decision, interacting with the network management through a single command interface. In self-management approaches, the control and decisions are subject to the control-loop in an automatic way. So, most of the self-management approaches use centralized servers to control, act and disseminate the

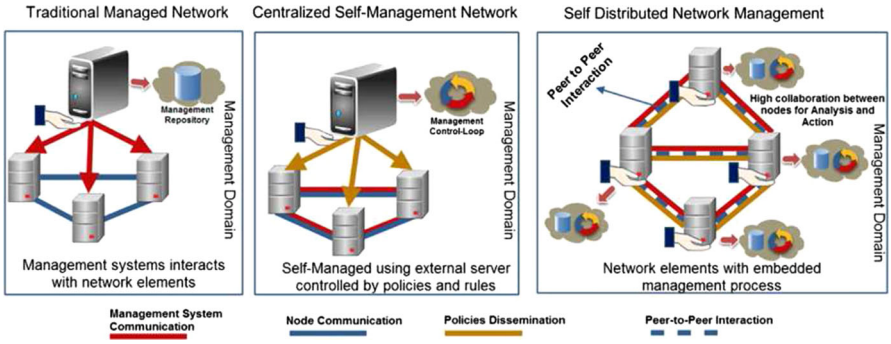

Fig. 1 Traditional, centralized self-management and fully distributed management

policies and rules. The external server approaches turned out to be inadequate in terms of scalability.

As opposed to the traditional management and external self-management dedicated control, in distributed management, the goal is to achieve scalable management for dynamic network environments. By doing so, a trade-off between scalability and low complexity shows to be the main challenge of the current distributed management approaches. The guiding principles for achieving this goal are the decentralization and self-organization. In order to achieve these goals, a number of functional requirements have been proposed [42], and we identified the most important in a distributed management communication infrastructure:

- *Situation awareness:* Suitable mechanisms for real-time monitoring of network-wide metrics, group size estimation, bootstrapping, nodes and topology discovery, data search and anomaly detection. In this paper, we define situation awareness as the process of being aware of what is happening to entity surroundings, such as a new node arrival in the network, link failure, network topology changing events, etc.
- *Scalability:* Support scalability in terms of network size, e.g. the number of network components to be managed. It must provide mechanisms to aggregate the network in domains or in federated multi-domains.
- *Functional Comprehensiveness:* Provide functional richness to support a variety of essential management tasks.
- *Extensibility:* Assure that capabilities of nodes can be extended with new functionalities.
- *Small Footprint:* With respect to storage space, bandwidth consumption, network consumption, and other devices and network resources.

The role of a unified distributed network management is to integrate functionalities i.e., bootstrapping, discovery, exchange of management information and autonomic path decisions into the managed devices in a distributed network for wired and wireless scenarios.

A common challenge for both wired and wireless environments is to maintain the distributed management information always updated until reaching the last-hop

devices without compromising the network performance. Regarding to the autonomic decisions, the challenge is to maintain the consistency on the decisions only taking into account incomplete information about the network (information only at few hops). Network nodes are always susceptible to changes in load requirements and requests for QoS: a node or link can fail due to hardware problems, power loss, damaged cables, frequency interference, misconfiguration or a malicious attack, etc. The network is expected to be collectively aware of the changes in traffic load and entity failures (node, link or both).

Distributed management approaches are characterized by eliminating the centralized coordination device, being crucial to assess how the devices dynamically behave under a diversity of unexpected events and network faults on both wired and wireless networks.

In the proposed distributed management approach each device interacts with its peers, taking decisions based on the gathered knowledge from the other devices, while performing a network of collaboration and cooperation. The management functionalities are placed inside the network, and not in dedicated central servers. Each network management process interacts with the neighbourhood through an overlay association. This communication can be done through peer-to-peer interaction and relying on different propagation schemes to enforce management processes that will allow a level of real-time awareness notification and management reconfiguration. This approach requires continuous interactivity between entities in order to exchange information about each entity (and therefore the network). This information will allow the network to make autonomic decisions, through collaboration between the network nodes, reacting to network changes (such as link failures, load variations, etc), and continuously optimizing the network resources.

The UDNM framework is organized in two main components, according to the Fig. 2: (A) Situation awareness, which sets up initial information through bootstrapping, discovery, fault-management process and exchange of management information; (B) Autonomic Decision System (ADS) that performs distributed decisions of paths in the network with incomplete information. The A and B components provide a basis for devices communication and network management decisions: each device needs to have those components to start the management process by itself. We consider management information as all type of data, gathered from devices that compose the network, such as: identifiers, network topology, devices resources status, bandwidth available, delays, network density, network addresses and domains. In terms of network decisions, it is essential to disseminate the local decisions in order to provide global cooperative decisions between the equipments. It is also expected to define which devices need to receive the management information in order to provide the required action, as well as how to identify them while optimizing the dissemination of autonomic decision process.

The role of the UDNM framework is to unify the distributed management process, making no distinction on equipments involved in the network; both wired and wireless devices may contain an instance of the UDNM framework which collaborate and cooperate seamlessly with each other. At the level of the communication between the nodes (Fig. 2), the direct interaction is formed between

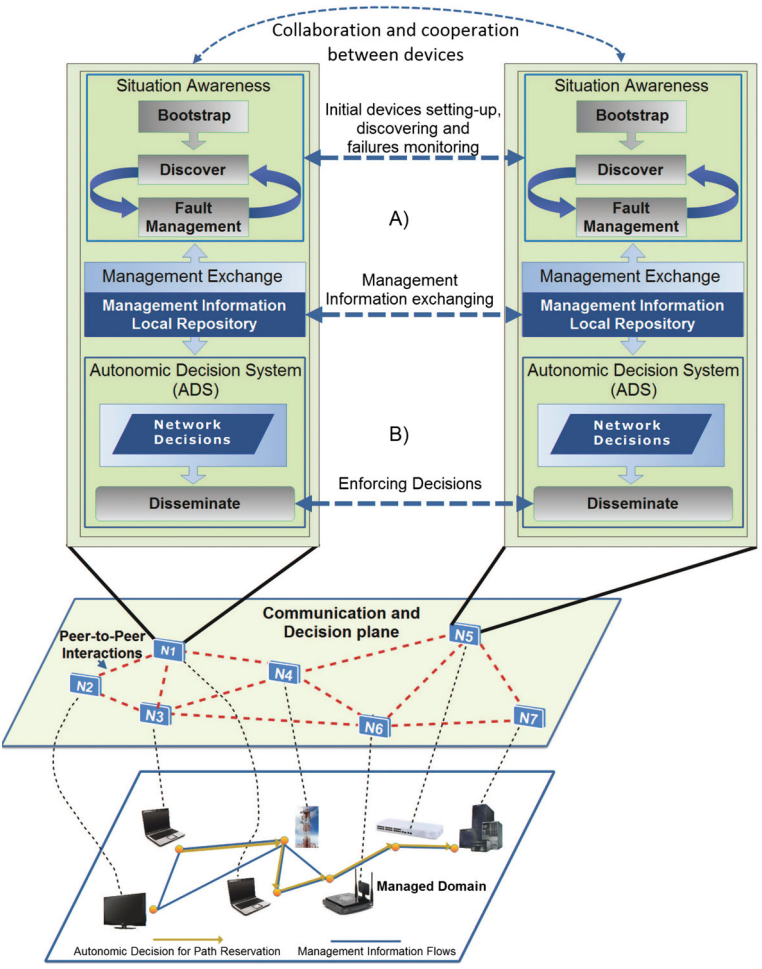

Fig. 2 Overview of the unified distributed network management framework (UDNM)

those residing in the corresponding coverage area, without centralized network elements. In fact, the nodes mobility translates into a more dynamic scenario, implying higher complexity at the embedded node level due to constant changes in the environment. The situation awareness mechanisms are embedded in each network device and each process interacts with the neighbourhood through peer-to-peer interaction, relying on different dissemination schemes to enforce management processes, and allow real-time notification and reconfiguration in response to node addition or failure. Considering the management exchange information, due to the dynamic nature of the nodes, to maintain the complete information of the network is always a complex task, and often impractical depending on the environment or the scenario. This requires considerable effort from all devices to maintain and to share the management information. Therefore, keeping partial network information brings benefits to the communication process, enabling mechanisms to control it without

adding extra costs. To achieve correct decisions with incomplete information, the ADS uses artificial intelligence techniques to learn the network status with only a limited set of information in the neighborhood.

3.1 Situation Awareness and Exchange of Management Information (Component A)

From a single node viewpoint, the bootstrap sets up the initial information, e.g. devices identifiers, initial management policies strategies, assignment of unique identifier, MAC address, BSSID, hardware capabilities and timers. The output information generated from UDNM local repository of each device can be used as the input for many applications, e.g., management/monitoring tools, autonomic path decisions, routing, flow and policies management.

After the bootstrapping stage, the node is ready to start cooperating in the discovery process in order to identify the surrounding neighbours. We consider surrounding devices the ones that can communicate with each other in the transmission range, or directly connected by a cable or physical link. The discovery protocol builds and periodically updates the information about neighbours. This process is based on continuous search for new devices, or on the update of the information of known devices. The signalling process considers direct and indirect contact, which contains information about directly connected devices and also through other devices in a specified range or depth.

3.1.1 NED Function for Discovery

For the discovery process, we propose the Hide and Seek (H&S) mechanism [1, 2]. In order to create and gather information of surrounding neighbours, the devices send HELLO contact messages to its neighbourhood using a pre-defined depth. The role of depth is to ensure the possibility to define the degree of knowledge (number of hops) of the network.

All gathered information is recorded in a local repository denominated by partial view of the device. The devices at first contact exchange HELLO messages using depth equal 1, for example, to avoid long cycle messages. Notice that we consider that each device does not need to know the entire network. Therefore, the managed devices must have cooperative communication among the other devices beyond the limit of directly connected devices, in order to exchange the management information. This cooperation is provided by a Neighbours Eyesight Direction (NED) function. The NED function, firstly presented in [2], is a dissemination function that aims to narrow the directions through which neighbour devices (acting as end-points or relays) are chosen to continue the dissemination of management information. The partial information is controlled by the NED function which defines the degree of knowledge (number of hops) of the network is required, e.g., to make more accurate decisions. This process depends on relay devices to forward dissemination messages until the depth limit is reached. The depth is the number of hops to disseminate the information, which ensures the possibility to define how long a device wants to disseminate the management information in the network.

The NED decision criteria used to choose the next hop neighbour to continue the dissemination process is based on the calculated ranking from the gathered information from neighbour devices. Details about the NED rank calculus and equations are presented in [2].

The NED decision is based on the parameters, such as link bandwidth available, network and device's resources available (RTT, CPU and Memory), and by the neighbours devices density. The link bandwidth available parameter encourages the NED to find better link conditions to disseminate the information, e.g., link congestion or lower link available bandwidth are not suitable to flow the dissemination process.

In case of local and network resources, the NED encourages the choice of neighbouring devices that have appropriate resources and network conditions to relay the management information, e.g., if a device has low RTT, free CPU or memory, it is a good candidate for relaying information device. Otherwise, they are dropped from the best neighbours' decision list.

At last, the network devices density parameter encourages the NED to choose neighbours that have more density of nodes, e.g., if a neighbour has fewer nodes in its surrounding area, it is not probably a good choice to disseminate the management information to a large number of nodes.

In order to demonstrate the operating principles of the NED, in Fig. 3, $N1$ is considered a dissemination device, where it is started the dissemination process to the neighbour devices $n2, n3 \ldots n11$. All devices directly connected to $N1$ send and receive HELLO and *DeviceInfo* messages containing management information about the neighbours (more details about HELLO and *DeviceInfo* packet structure can be seen in the Sects. 4.1.2 and 4.1.3). We consider management information as all type of data gathered from devices that compose the network, such as: identifiers, network topology, devices' resources status, bandwidth available, delays, network density, network addresses and domains. So, the best direction chosen to continue the dissemination is the device $N11$, and information about $N1$ and its neighbours is

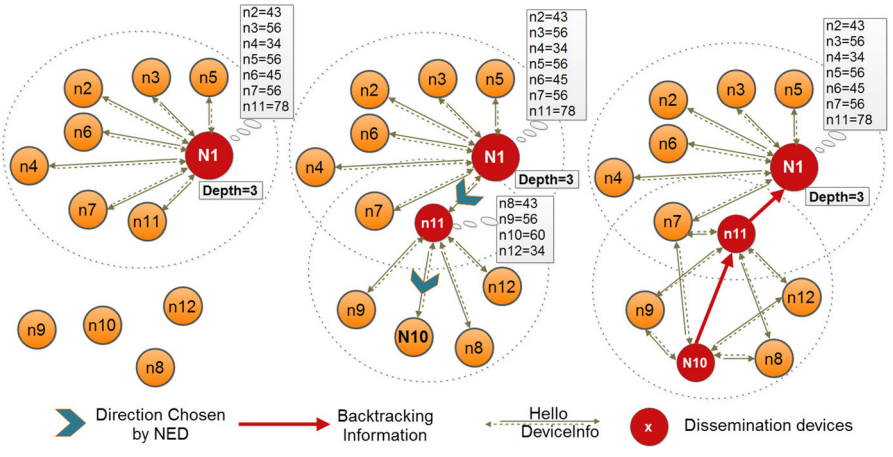

Fig. 3 NED decision and management information exchange

disseminated to $n11$. Note that, the density of the $n11$ and $n7$ are the same, but $n11$ probably has better bandwidth available and local/network resources than $n7$, thus, the NED opted to choose $n11$ as the relay node due to the max NED rank value calculated (e.g. 78).

The device $n11$ repeats the process and chooses a new direction (e.g $n10$) according to the maximum NED rank value calculated (e.g. 60). After this process, the device $n11$ also disseminates the management information to the next hop chosen, and finally all information from $N1$ is disseminated through the neighbour devices. All information collected from $n10$ to $n11$ is back tracked to all devices that belong to the NED paths (e.g. $n11$) up to the device that originated the request (e.g. $N1$).

The NED function process ends when: (1) the depth limit previously defined is achieved; or (2) all possible paths between the devices are explored by the NED. Note that it is possible to configure different depth requests for each dissemination device. The NED can also reuse the depth from a device, without making any further demands [2].

3.1.2 Failures and Fault-Management

The management information is received from the node within reach and can also be updated according to topology changes. To monitor and detect failures, a fault management functionality is introduced, maintaining the consistency of management information. The fault management functionality has support for both wired and fixed/mobile wireless network scenarios. For fixed wired/wireless scenarios, even if the network is stable, the fault management process will detect any event related to the link or node failure, a specific change of traffic and users services. For mobile scenarios, the fault management process will also detect events, even if the nodes move away from the cover area of other nodes. The fault handling is divided into critical and warning levels. The critical failures that affect devices and links shall be handled reactively. Warning level is just for feedback about the devices status (e.g. CPU and memory available).

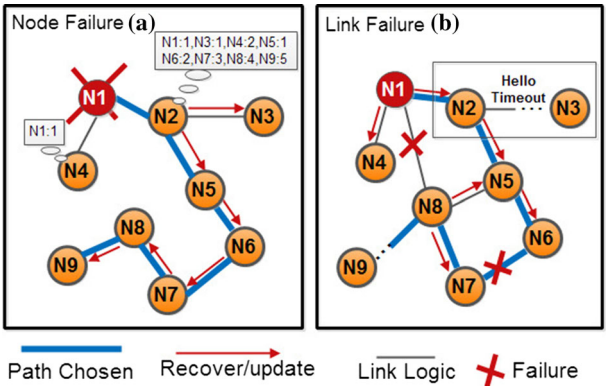

Fig. 4 Example of fault-management functionality for nodes (**a**) and links (**b**) failure

Figure 4 presents an example of the dynamics of the management approach when node (a) and link (b) failure is detected. For example, the device N1 (Fig. 4a) fails losing all the connectivity between the neighbours N2 and N4, whereas if just one link fails (Fig. 4b), N1 is still operational and can advise other devices (N2, N4) about its link failure to the device N8.

In Fig. 4a, the device N1 fails, and devices N4 and N2 detect that N1 stops to advertise its presence for a while. Both N1 and N4 check their local repository information gathered from the discovery process, if there is information larger that 1-hop. In case of N4, it has only information about the N1, which removes it from its local repository. On the other hand, the device N2 has information gathered from other devices in the network, (e.g., N1 at 1-hop up to N9 at 5-hops). Then, the device N2 sends recover/update messages up to the last hop device (e.g. N9 at 5-hops). The path followed by the recover/update messages takes into account the chosen path already made by the discovery through NED function, avoiding extra messages propagation in the process. In fact, only the device N2 has next hop devices to relay the recover/update message. Each node that relays the recover/update messages (e.g. N5, N6, N7, N8) updates their local repositories (about N1 failed) until reaching the last hop device (e.g. N9). Otherwise, if the N1 recovers from failure, N2 advises the same way the existence of the N1 in the network. The failure can occur at any time and at any node, so the neighbours directly connected to the node take all appropriate actions whenever possible. Moreover, if a failure happens on the path already chosen by the NED function, the node that detected the fault (e.g. N6 and N7) has the possibility to call the discovery process again with the goal to find more alternative ways in which management information can be exchanged and recovered in case of failures.

In case of link failures, Fig. 4b, if the node has more than 1 logic link between the neighbours, e.g. N1 is directly connected to N2, N4 and N8, then N1 and N8 detect their direct link fails, thus, both N1 and N8 propagate recover/update messages to remaining links connected until reaching the last hop devices, similarly to Fig. 4a. Devices N3 and N9 are moving out of the coverage area of the devices N2 and N8; thus, N2 and N8 detect also a link failure according to the HELLO time-out limit configured at bootstrapping device stage (e.g. limit detection time of 5% more than the HELLO time interval). Thus, devices are warned without compromising the control performance of the network in comparison to the uncontrolled flooding techniques. Each recover/update message received by the devices updates the local repositories to the new information received, and makes the necessary updates automatically according to the acquired knowledge of the network. This ensures that the management information is always updated whenever any critical failure in the network occurs. In addition, the information required for the management decisions is based on parameters, such as link bandwidth available, network and device's resources available (RTT, CPU and Memory) and by the neighbours devices density. The link bandwidth available parameter induces the decision to find better link conditions to disseminate the information, e.g., congested links or lower link available bandwidth are not suitable to flow the dissemination process.

3.2 Autonomic Decision System (Component B)

The objective of the autonomic decision system is to perform network control and management decisions in a distributed approach, without resorting to the overall network information, as performed by centralized decision systems. We will consider the specific case of a path establishment with bandwidth requirements to describe and analyse the proposed approach.

Network Management is resource-usage oriented, where nodes are aware of the required resource consumption to achieve their objectives. In order to systematically generate better and improved decisions, nodes share, within their neighbourhood range, the state of their own resources. The decision mechanism uses the link state information (availability and link available bandwidth) for the decision. Also, to improve future decisions, the quality of the decision itself is evaluated, where the length of the path is taken into account and compared with the length of the previous decisions. To provide the decision mechanism with a minimum set of accurate information, the neighbourhood topology is maintained up-to-date, by synchronizing the neighbourhood information status. Such synchronization is only performed with the peers within the local neighbourhood. The information set being synchronized is composed by the network links status with the respective node identifications.

In theory, it is possible to infer the status of the network links. By using artificial concepts such as Neural Networks and Deep Learning, and by continuously training these networks, it is possible to create a system, capable of generating predictions with a low prediction error. However, in practice, such systems are known to be resource-hungry, consuming processing time and requiring large amounts of historic data to be transferred between peers, and used to train the neural system [43, 44]. Such resources are usually unavailable in wireless network devices, due to the power consumption or to the costs of implementation.

Our approach will explore the established communication paths between network agents by exploring network paths and continuously learning which path is the best to a destination. The system is itself autonomous and decentralized, establishing paths for bandwidth reservation, effectively creating communication paths between pairs of agents and improve those same decisions, taking into account link bandwidth usage and the path size obtained by the system decisions. The bandwidth estimation is provided by the NED input value from Equation presented in [2]; the ADS has the capability to adapt and improve its decisions, based on the bandwidth consumption from the network links used to establish communication paths.

The autonomic decision system uses a limited subset of knowledge about the network state, which is seen as the local neighbourhood of the node (Fig. 5), the partial view. The local neighbourhood defines three types of node views: Local Node, the node itself that manages his neighbourhood; Neighbourhood Nodes, the nodes within the neighbourhood boundary but not at the boundary itself Gateway Nodes, the nodes at the neighbourhood boundary. By cooperating with the network nodes inside the local neighbourhood, a node is able to discover which ones are the gateway nodes (nodes at the boundary of the neighbourhood). Nodes will contact the others in order to connect with the nodes outside the known neighbourhood.

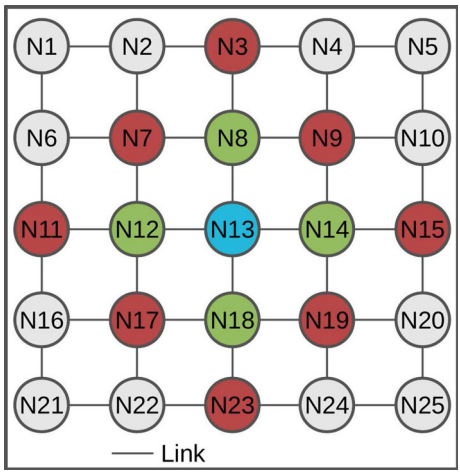

Fig. 5 Communication scheme (1 Hop)

The limitation of the knowledge set creates the partial view ot the node. Each node within the network will have its own view, provided by the component (A) situation awareness through the *partial view* table of the node, in which the size will depend on the depth limit value. The size of the knowledge set may differ from node to node and the network complexity.

Since the autonomic agent in the node has only a partial view of the network state, any service belonging to a node that is required but not present within neighbourhood boundaries will need to be searched across the known border. Since the node can have several gateways, it needs to choose the best one. There will be an active competition for network resources, and therefore, the nodes maintain an experience value for each path that they explore.

In the context of this article, the exploration mechanism finds and explores several possible paths (if alternatives are available), and for each explored path or failure to find one, a reward is calculated, according to the mechanism of reinforcement learning, which will reinforce the existent experience along the path. The learning algorithm is structured in two phases, Exploration and Learning.

In the exploration phase, the agent will first determine which links within its known neighbourhood are capable of supporting the path with the requested bandwidth. Taking this factor into account, the local agent in the node will now determine if the destination node is within contact reach or not. By discarding known neighbourhood links that cannot support the path reservation, the node creates a temporary vision of its own neighbourhood, which might temporarily exclude the destination node, forcing the exploration algorithm to search for path alternatives through the gateway nodes. This exclusion is needed, because the links that would be normally used to establish a Dijkstra [45] shortest path to the destination node may not have enough bandwidth to support the path requirements.

The learning system is based on reinforcement learning concepts, more specifically, the SARSA algorithm [46] in Eq. 1. After the exploration phase, the learning system will use the obtained results from each path. If the exploration

phase has failed in returning a path through the selected gateway node, the experience will be updated with a reward equal to zero; otherwise, it will be updated with a calculated reward.

In the beginning of the network node life (when it boots up), all experience values start at 0, where the objective is to improve future decisions to get the best experience value. During the backtracking of the experience value, each gateway learns and calculates the reward and a new experience value ($Q(G_t)$), updating the forward gateway node experience value. This new experience value is then *rolled-back* to the previous gateway node in the path, and the reward is again calculated for that gateway node along with a new experience value. This process ends when it reaches the origin agent.

The $Q(G_t)$ represents the current gateway agent experience value, and the $Q(G_{t+1})$ is the experience value returned from the chosen gateway agent. P is the locally discovered path, which can be the path from the origin node to the gateway node, from the gateway node to the gateway node, or from the gateway node to the destination node, depending on the node that is calculating the reward. $|P|$ is the length of the local path from the local node to the gateway node. If a possible path solution is not discovered, then $reward = 0$. If the destination node is in the neighbourhood of the gateway node and a local path is found, then $Q(G_{t+1}) = 0$.

In the reinforcement learning equation, it is necessary to choose the values for the learning rate (α) and discount factor (γ). Since the system needs to quickly learn new valid paths, the learning rate needs to take a value which would prefer the most recent actions results. It is also required that the system gives more importance to the shortest paths, rather than long paths. Since the discount factor represents how much of the experience returning from the gateway nodes is actually used to calculate the experience of the chosen action, it is necessary to maintain it with a low value.

The reward (Eq. 2) is composed by the *Explored Path Distance* (E_{pd}), which is the length of the path from the origin node to the destination node, and the *Known Shortest Path Distance* (K_{spd}), which is the minimum known distance to the destiny node by the node determining the reward. The *Remaining Bandwidth* (R_e) value represents the available bandwidth in percentage (multiplied by 100) at the used edge (e) to establish a path.

$$Q(G_t) = Q(G_t) + \alpha(reward + \gamma Q(G_{t+1}) - Q(G_t)) \tag{1}$$

$$reward = \frac{\frac{100.0}{|P|} \times \sum_{e}^{e \in P} R_e}{E_{pd} \times K_{spd}} \tag{2}$$

These explored paths are used by local agents in the nodes to communicate with unknown network nodes outside the range of their respective neighbourhood topologies, using the appropriate neighbour gateway. Figure 6 shows how the autonomic decision system interconnects with the component (A) by receiving its pre-processed network events. This way, the decision system is kept updated regarding the local network state, allowing it to respond to critical network events

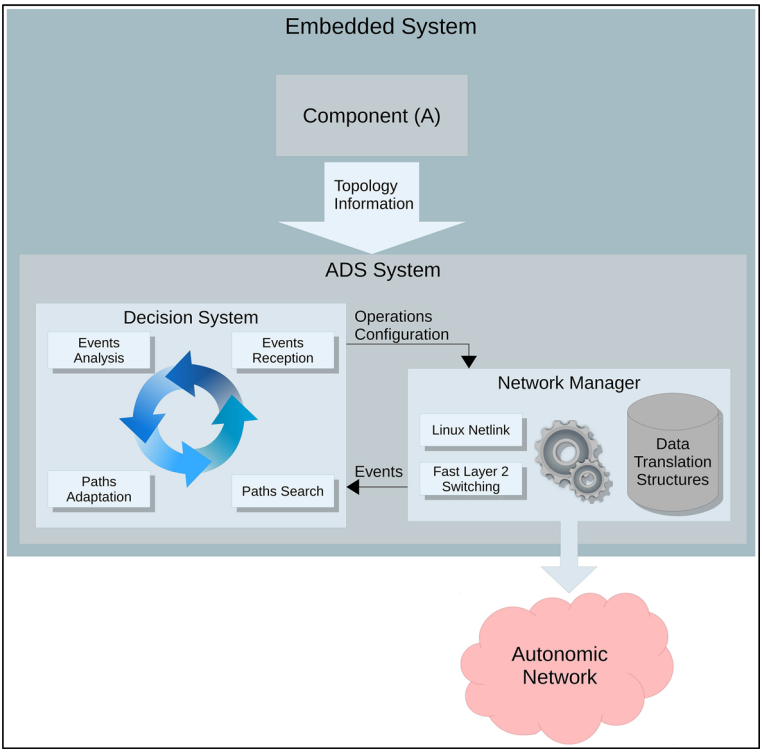

Fig. 6 Decision and processed events

such as the loss of links or nodes. The decision system also interconnects with a network management module, which is responsible to encapsulate the packets with the path ID information, so that it follows the specific path, selected by the reinforcement learning algorithm. Also, the network manager allows the communication with neighbour nodes.

A closer look of the implemented modules and interactions will be described in the next section.

4 Implemented Modules and Interactions

This section describes the modular design and implementation of the UDNM framework.

4.1 Situation Awareness and Exchange of Management Information

The situation awareness and exchange of management information (component (A) from UDNM framework) process consists of modules with distinct functions as can be observed in Fig. 6. Note also in the figure the integration between

components (A) and (B) from UDNM framework. Component (B) will be better explained in the Sect. 4.2.

Two main blocks are proposed: events handler which addresses the main events occurring in the system, and repository structures where they are stored in the node (Fig. 7).

4.1.1 Functions

LinkDiscover is the main function of the situation awareness process which contains all data structures responsible to trigger the discovery and exchange of management information functionalities. Its execution is launched in each existing device interfaces, including both wired (eth0) or wireless (wlan0). In case of depth larger than 1 hop, the *LinkDiscover* module is responsible to forward the Hello packets through the interface which returns the value calculated by the NED. These packets are periodically sent by the device in order to discover other devices in the network (see Sect. 4.1.2 for more details on the Hello packet structure). Additionally, besides Link Discover function, the UDNM framework integrates the monitoring function in the same process. This function is responsible for monitoring, detecting and reacting according to events or anomalies in the system. For example, the detection time limit of the Hello messages can be configured at the bootstrapping process. The fault-management process can advise other devices through recover/update messages (see Sect. 4.1.4) for more details on the Recover/Update packet structure).

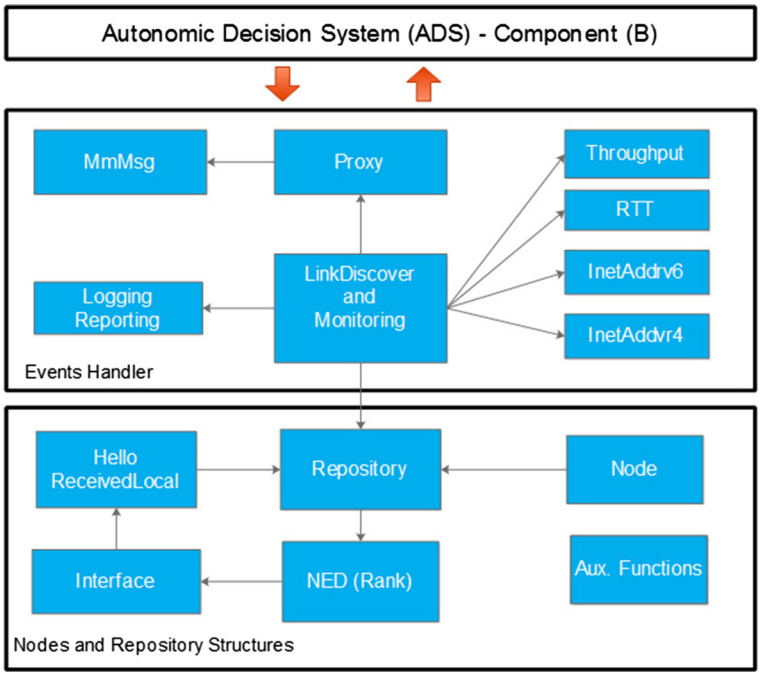

Fig. 7 Functionalities of the situation awareness and exchange of management information

The *NED (Rank)* module stores the ranking of the known devices calculated by the NED function to determine which is the best candidate neighbour to forward the information.

The *Repository* module is local to each device and it is composed by the partial view of the device. The information on each partial view entry is received from the device within reach and can also be updated according to topology changes. Therefore, the partial view is dynamic and its size varies according to the new received information. In addition, if the information is outdated (i.e. the node did not receive any contact in a specific amount of time), it assumes that there are no nodes in range.

The *HelloReceivedLocal* module records the devices identifiers from which Hello packets, which are defined in Sect. 4.1.2, have already been received, in order to reply to them with the appropriate information of the device. The *AuxFunctions* module is defined as a set of helper methods to ease information acquisition tasks. For instance, the methods to obtain the MAC address from the name of the interface, the percentage of free CPU and Memory RAM, a list of all network interfaces of a specific device, a list of IPv4 and IPv6 addresses of an interface, etc. The *Device* module comprises the data structure of the devices, including the unique identifier, network status, devices resources and the rank calculated by the NED. The *MmMsg* module contains the definition of the message structures that are exchanged in the UDNM framework.

When a control packet for discovery and information exchange is received, the *Proxy* module will process it and send a response message back to the device that originated the request. In the case of depth larger than 1, the *Proxy* module is responsible to check if the message is at the last hop and, if so, it responds with the device information that rolls back to the originated device. The *Logging and Reporting* module will return local devices and network feedbacks in case of failures or anomalies e.g., invalid socket, full buffers, out of memory access, interfaces down, crashes on link/device and empty repositories. Modules as *Throughput* and *RTT* collect information about the status of packets and the links in the network, which will be then exchanged between the devices, helping to build the NED (Rank). *InetAddrv4* and *InetAddrv6* modules are responsible to convert automatically all signalling processes, which guarantee the consistence between both technologies (IPv4 and IPv6).

4.1.2 Packet Structures

This section presents the structures of the exchanged packets.

The Hello packet, presented in Fig. 8a, is sent by each device in order to receive the contact of the other devices in the network.

In addition, the time interval to send a Hello packet is adaptive and is given by:

$$H_i = \frac{(L \bmod (U \times P_{i,d}))}{\sqrt{P_{i,d}} \times L} \tag{3}$$

where L and U represent the lower and upper time limit previously adjusted as reference (e.g. [10–20]), and $P_{i,d}$ represents the total number of devices at hop

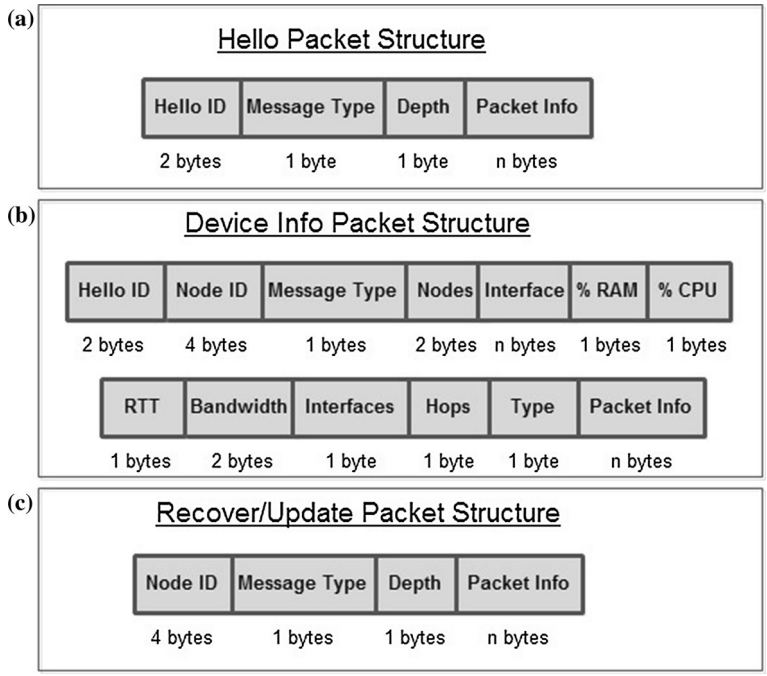

Fig. 8 Packets structure: hello packets, device information and recover/update

distance (d) in the partial view of a device i. Moreover, (H_i) is the time interval which can increase or decrease according to the total number of discovered devices in the network without crossing the limits L and U.

4.1.3 DeviceInfo Packet Structure

The *DeviceInfo* packet is shown in Fig. 8b and it contains the information for the NED decision, which is sent by each device after being contacted, i.e. upon receiving a Hello. In this packet the local resources information of the device is sent, as well as the controlled list of NED decisions, which synchronize the unexplored paths with the Hello Packet Info field. The *Devices* contain the number of known neighbours (more details in [2]). The *Interface* field contains the interface, the device ID and IP from where the message was sent. The *% RAM* contains the amount of memory RAM and free *% CPU* available as well as the value of *RTT* (more details in [2]). *Bandwidth* field contains information about the link capacity of the devices (more details in [2]). *Interfaces* indicates the number of local interfaces of the device. The *Hops* indicates from how many hops the original request came. The *Type* refers to the message subtype of a received DeviceInfo (1: initial exchange of messages by the directly connected devices; 2: reserved for exchange of information with depth larger than 1 hop; 3: updated the directed connected devices with the information gathered after finishing the process of exchange of information with depth larger than 1 hop). This ensures that the entire signalling

process is performed through Hello and DeviceInfo messages, which are handled and forwarded internally by the *Proxy* module between the direct connected devices, or relay devices chosen when the depth is larger than 1 hop. Finally, the *Packet Info* is similar to the Hello messages, i.e. it contains the information of the devices through which the message will be forwarded back (e.g. backtracking) in case of depth larger than 1 hop.

4.1.4 Recover/Update Packet Structure

The Recover/Update packet is shown in Fig. 8c and it contains the information to recover or update the devices when failures are detected. The *Node ID* contains the ID of the device that originates the failure. The *Type* refers to the message subtype of a received Recover/Update (1: propagate error recover; 2: update local repository; 3 device is operational again). *Depth* is the depth in hops to propagate the Recover/Update messages between devices. The *Packet Info* contains the information of the devices through which the recover/update message will be forwarded in the case of depth larger than 1 hop.

4.2 Autonomic Decision System

The autonomic decision system (component (B) from UDNM framework) is depicted in Fig. 6, which contains the integration of the decision system and the network manager.

The decision system implements the algorithms that take all decisions. In order for a node to communicate with another node, a local connection must be established by the network manager. The network manager is responsible for the encapsulation of the communication packets exchanged between pairs of nodes, which makes the stream of packets to follow the path designated by the decision system. The encapsulation protocol works between the OSI Layers 2 and 3, acting like a 2.5 Layer protocol. To function properly without affecting other protocols, the encapsulation protocol uses its own Ethernet Type.

The cooperation between agents is performed in the delegation process. Due to the existent limitations regarding the amount of topological information kept by each node, each node delegates the exploration process to the neighbour nodes located in the peripheral of the neighbourhood. Nodes effectively establish a cooperation chain by continuing the exploration process, sent by the network node. When the exploration terminates, an answer is sent back along the cooperation chain, up to the starting node.

Figure 9 shows the encapsulation header used to encapsulate data exchanged between network nodes. To identify the receiver node and the sender node, the header contains the UUID of the nodes. The header also contains the identification of the source client within the local network controlled by the source node, and the identification of the destination client in the destination node.

The Path ID is used to identify the path in the network, through where the packet should travel. This Path ID is determined by the decision system when a path exploration occurs, being calculated using the hashing algorithm SHA-3 [47, 48].

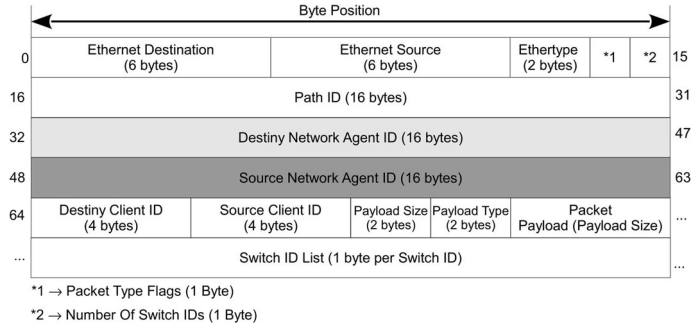

Fig. 9 Encapsulation packet header

At the end of the packet, there is a byte list which is the Switch ID List. These bytes are used for fast packet switching from one network interface to another. This is used to simplify the Internet table lookup overhead, and to reduce the amount of information required to route a packet through a specific network path from the local node to a gateway node. When arriving to a gateway node, it will look at the Path ID, push a new Switch ID list and modify the Path ID value, so that the next gateway node can know where to route the packet. If the destination node is within the known neighbourhood, the PathID will be 0. The remaining fields are for network management only, and are not related to the decision system.

4.2.1 Encapsulation Protocol Overhead

Each packet transmitted between network peers will have an Encapsulation header, which has a deterministic size. The minimum header size is 76 bytes without any switch ID list. The header size will increase 1 byte for each switch ID added to the list. This means, the longer is the local switching path used by the network peer to reach the destination peer or a gateway peer, the bigger will the header be.

4.2.2 Exploration Message

The exploration message is a data structure exchanged between nodes during the path exploration process. It represents a path exploration containing information regarding the path bandwidth requirements, exploration status, path length, the explored PathID and the path accumulated experience. It also identifies who is the **Source Node** that requested the exploration, and who is the **Destination Node** to whom the path under exploration should reach.

The message is exchanged between nodes using a TCP/IP connection, in order to guarantee its delivery and integrity. There is only one message in circulation per path exploration. For reasons of simplicity, the exchanged messages were encoded in a format provided by the Python 3 Language.

4.3 Situation Awareness and Autonomic Decision System Integration

The components of situation awareness, management exchange and autonomic decision system were integrated through intra-process communication via sockets communication (see Fig. 10).

The messages (*InformADSAddAgent* and *InformADSLinkEvent*) contain inputs regarding to MAC address of source/destination devices, interfaces, discovered IP addresses from directly connect devices and larger than 1-hop, link bandwidth available and depth of the gathered information.

In case of failures (e.g. links, interfaces and event applications), they are detected through *fault_management()* process with the *InformADSErrorEvent* messages providing the status of the local devices links and interfaces (e.g, up or down).

5 Multi-network and Multi-technologies Testbed and Experimentation Results

5.1 Testbed

The deployed testbed demonstrates the interactions of the UDNM framework in a scenario considering wired and wireless networks, and networks with different sizes and topologies (fixed, infra-structured and ad-hoc). The involved equipment includes: 5 single-board computers Cambria GW2358-4 running OpenWrt Bleeding Edge (r35830); virtual wired grid running OpenWrt Bleeding Edge (r28129, Guest Xen paravirtualized) as 25 virtual machines on a HP Proliant server (CentOS-5 kernel Xen). Access devices are: 1 EEPC netbook, 1 laptop, 1 Cambria and 1 desktop computer.

For long range technology, 1 WiMAX IEEE 802.16e Alvarion Extreme 5000 (Base Station) and 1 Customer-Premises Equipment (CPE) were used. Note that, the WIMAX BS and CPE do not perform instances of the UDNM framework.

As depicted in the Fig. 11, in the scope of this work three scenarios are considered; (1) Typical operator Network which is represented by the 5 × 5 Grid;

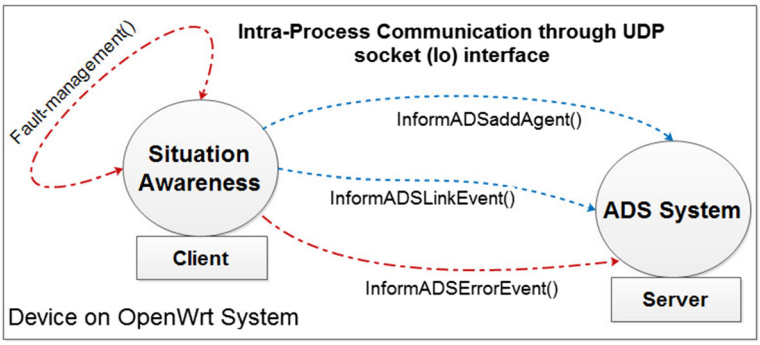

Fig. 10 UDP messages passing through client/server integration model

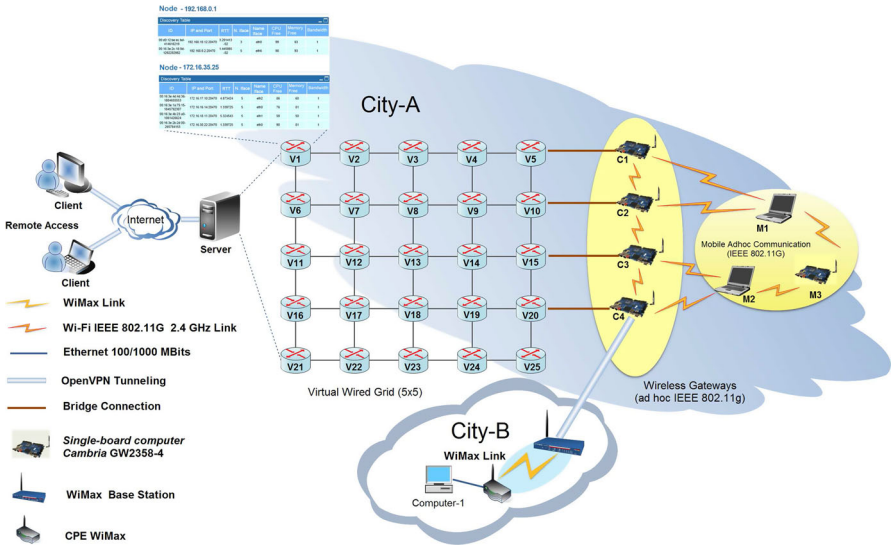

Fig. 11 Overview of the multi-network technologies testbed

(2) Mesh in a city which is represented by the wireless gateway nodes; (3) Mesh hotspots with ad-hoc mobile users represented by the mobile wireless nodes.

The web-interface has modules for interaction within the discovery process and autonomic decisions in order to manage information and path decisions gathered from the network devices. This interaction (see Fig. 12) between the web-interface and UDNM framework is made by Intra Process Communication LIBrary (IPCLIB). The IPCLIB is a middleware which mediates the exchange of management information, making use of local sockets for communication primitives, showing the management information gathered and autonomic decision events of the UDNM framework on the web-interface. Additionally, HTML and PERL scripts are responsible to organize all the gathered information for users perception.

Figure 13 depicts an example of the management information gathered from directly connected nodes and also from other nodes with depth larger that 1-hop, which represents: nodes identifiers, IP addresses and ports, Round-Trip-Time (RTT), name/number of devices interfaces, % of CPU and Memory available and

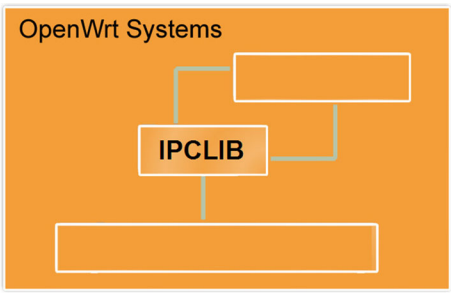

Fig. 12 UDNM framework, IPCLIB and Web-based interactions on top of OpenWrt systems

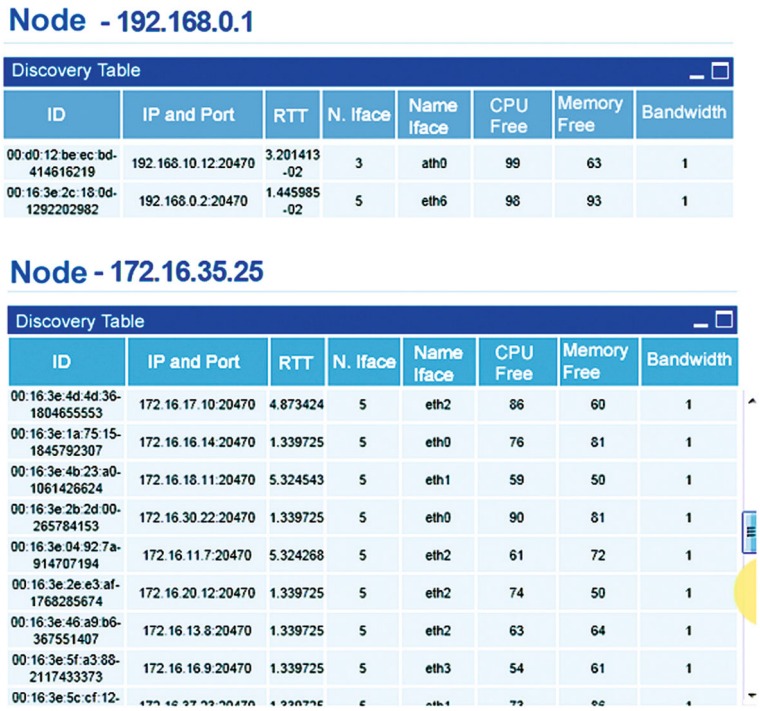

Fig. 13 Web-based discovery interface

capacity of the link. Note that, the basis of UDNM framework is running on top of OpenWrt systems, which is a lightweight Operation System, meeting all software and hardware requirements for devices with low processing power and storage. Figure 14 shows an example of the web-based decision process interface. With the decision interface, it is possible to add, remove and refresh bandwidth paths, also with the ability to select and view the path reservation for any device in the testbed. The table seen in the Fig. 14 contains all reservations known by the node selected, the source node (e.g. Node A) and the destination node (e.g. Node B) identifications, the total bandwidth of this link, available bandwidth for future reservations and used communication path.

In the virtual grid topology, the devices are connected through standard IEEE 802.3 network LAN on different sub-nets, according to the device connected interface (e.g., eth0 172.16.0.1, eth1 172.16.1.1). The grid topology is created through a Python script which automatically generates the virtual bridges and link connections between the virtual machines. The bridging between the virtual border machines (e.g. V5, V10, V15 and V20) to the wireless Cambrias (e.g., C1, C2, C3 and C4) is performed by VLAN connections as well. With respect to the Ad-Hoc wireless gateways through standard IEEE 802.11, the wireless Cambrias (e.g., C1, C2, C3 and C4) are used to extend the virtual grid topology in order to share the wireless connections, also to the mobile Ad-Hoc network devices (e.g. M1, M2 and M3), integrating all different topologies and technologies involved. Additionally,

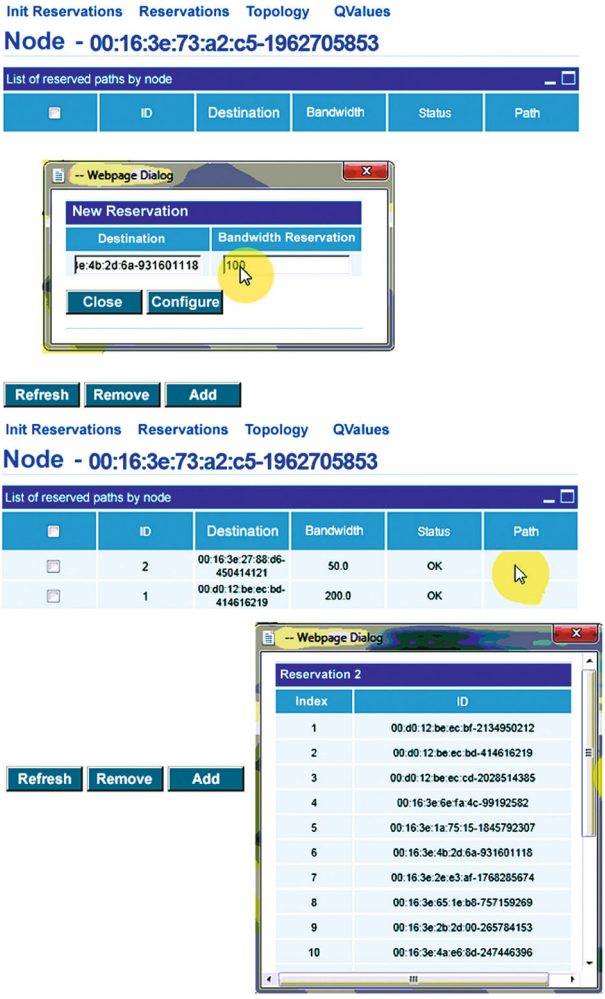

Fig. 14 Web-based decision interface

MAC filters are used to filter the communications devices, in order to provide properly Ad-Hoc connections (e.g., C1 may have to communicate through C2, to reach the device C3).

The server, the virtual grid topology, the wireless Cambrias C1, C2, C3, C4 and the access devices M1, M2 and M3 are situated in the City-A (Aveiro, Portugal), and the WiMAX BS and CPE and the access Computer-1 are located in the City-B (Covilhã, Portugal). Note that the WiMAX link is used to extend the tunnel between cambria C4 and Computer-1.

In order to unify the network access between both cities, tunnelling and virtual private network techniques over a dedicated high speed Intranet network were used. Each city is $\approx$ 250 km far from each other. The WIMAX IEEE 802.11e network

from City-B is assured by a WiMAX point-to-multipoint IEEE 802.16e Base Station (BS) operating at 5.4 GHz, guaranteeing the connectivity between the Computer-1 from City-B and Cambria C4 located in the City-A. The installed BS is an Alvarion Extreme 5000 one, which operates at 4900–5350/5470–5950 MHz licensed free frequency bands on TDD mode. The operating channel bandwidth is 10 MHz and the antenna configuration is 2×2 MIMO single sector. The UDNM framework uses this testbed to unify and perform management functionalities for discovery and exchange of management information process beyond the limit of local area networks.

5.2 Experimentation Results

We present the results of several experiments that evaluate the signalling cost, overhead of link occupancy, time to recover/update network device and link errors, CPU overhead and average time/number devices discovered as well as the accurate path decisions and time, according to the acquired knowledge of the network.

5.2.1 Wired Versus Wireless Networks

The purpose of this evaluation is to measure the signalling cost and overhead link occupancy separately in the virtual wired grid and the wireless devices in comparison to the baselines analysed. We compare the discovery and the exchange of management information of the UDNM framework with well-known discovery baselines, using OpenWrt open-source versions of OSPF [35] version 2, CDP [38] version 2 for wired grid, OLSR [39] version 4 and BATMAN [41] daemon r1439-1 version for wireless. The results are obtained with the objective of measuring the exchange of management information only between directly connected devices. We consider as management information the number of nodes, ID/MAC, IP, RTT, number of interfaces, CPU and memory available, bandwidth capacity and number of hops. The clocks of all devices in the testbed are synchronized through the Precision Time Protocol daemon (PTPd) [49], in order to minimize the offset that occurs among clocks. The values presented in the graphs below are an average of 5 repetitions and 95% of confidence interval.

In the wired grid, OSPF is configured in each interface to be in the same area of its directly connected neighbours. CDP is set up with the information of each virtual machine, and since it can only be used to share information about directly connected equipments, no further configurations are required. The interval between periodic packets (e.g. Hello) for both CDP and OSPF protocols are configured to be 10sec, except for the wireless scenario where it keeps the default values of the protocols (ORIGINATOR interval = 1 second for BATMAN and HELLO interval = 2 seconds for OLSR), following the standard [39, 41] specifications. In the UDNM framework this interval is dynamically adapted with a mean value around 10s for the wired grid and 5s for the wireless scenario. The links capacity of the wired grid is considered to be 10/100 Mb/s, and approximately 54 Mb/s in the wireless scenario, according to the technology standard capacity (e.g. IEEE 802.3 and IEEE 802.11).

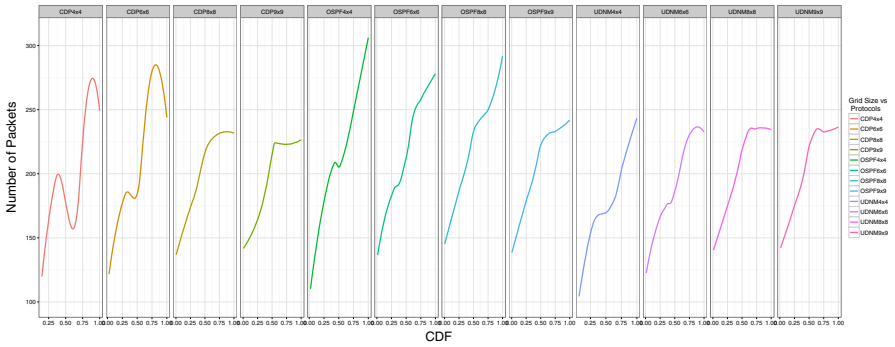

Fig. 15 Signalling cost (number of packets) in function of nodes increase on top of the wired grid

Figure 15 shows the CDF of the signaling cost in terms of the number of packets exchanged for discovery and exchange of management information process. The signaling cost is measured on top of the wired grid topology according to the Fig. 11. Note that the number of packets sent depends largely on the solutions and the number of nodes in the scenario. CDP and OSPF are configured with static refresh controls settled at bootstrapping process, although UDMN implements a dynamic refresh interval, which works according to the number of surrounding neighbors (see Eq. 3).

Table 1 presents the link occupancy for different grids and protocols analyzed. Note that CDP has a link occupancy of 1.54 Mbit/s, taking in consideration the link capacity of 1Mbit/s and the observation time of 300 s. An important observation is that the discovery and exchange of the management information process of the CDP protocol saturated the link capacity in more than 0.5 Mbit/s for all grids sizes. This link saturation explains the accentuated packets oscillation for the CDP protocol as presented in the Figs. 15 and 16. Therefore, OSPF has a link occupancy around 0.27 Mbit/s, while UDNM has the lower link occupancy of 0.02 Mbit/s. This reinforces that UDNM is a lightweight solution for exchanging management information at low overhead cost when compared to the baseline protocols analyzed. Therefore, this does not necessarily mean that UDNM framework has less information than OSPF and CDP, but instead that no redundant data is used, i.e., only the strictly required one is forwarded between devices.

Table 1 Protocols link occupancy

Grid size	CDP (MB) avg. overhead	OSP (MB) avg. overhead	UDNM (MB) avg. overhead	Link occupancy (link capacity 1 Mbit/s)
Grid 4 × 4	57.8	10.2	0.80	CDP ≈ 1.54, OSPF ≈ 0.27, UDNM ≈ 0.021
Grid 6 × 6	60.3	10.2	0.88	CDP ≈ 1.608, OSPF ≈ 0.27, UDNM ≈ 0.023
Grid 8 × 8	59.8	10.3	0.90	CDP ≈ 1.59, OSPF ≈ 0.27, UDNM ≈ 0.024
Grid 9 × 9	59.0	9.7	0.89	CDP ≈ 1.57, OSPF ≈ 0.258, UDNM ≈ 0.023

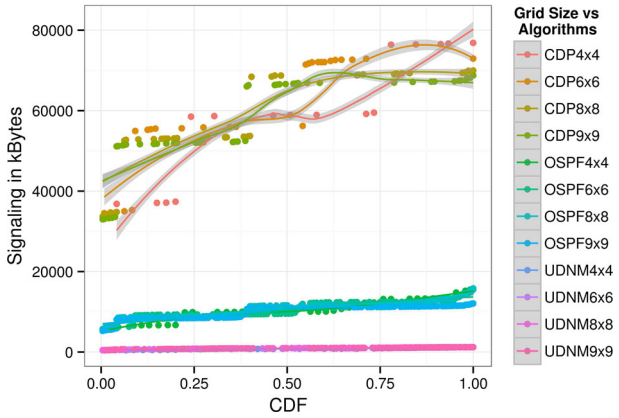

Fig. 16 Signaling cost (KBytes) in function of nodes increase on top of the wired grid

Figure 16 depicts the signaling cost in Kbytes, where the UDNM framework has a reduced cost when compared to CDP and OSPF protocols. CDP sends announcements and the directed connected devices receive and store them in a local table that will be shared to the other CDP devices. The frame size of the messages exchanged in the network increases, impacting in the high signaling cost for all devices where CDP messages are sent. Note that, UDNM signaling cost is smaller when transmitting the same amount of information than the CDP and OSPF.

Additionally, the impact of the overhead in a typical operator link is also measured and it is shown in Table 2. Assuming a link capacity of 1 Gbit/s, the percentage of the control overhead is very low for all protocols. As it can be noticed, the impact of the CDP protocol grows proportionally with the network size, and the OSPF protocol has a similar behaviour to CDP but with smaller impact. UDNM demonstrates to have much lower rate of occupancy than CDP and OSPF.

Table 2 Average occupancy of each protocol on a 1 Gbit/s link, for different network sizes on wired virtual grid topology

Protocol	Network size	Average occupancy (1Gbit/s link)
CDP	4 × 4	0.827 ± 0.0016%
OSPF	4 × 4	0.146 ± 0.0050%
UDNM	4 × 4	0.011 ± 0.0012%
CDP	6 × 6	2.013 ± 0.0072%
OSPF	6 × 6	0.347 ± 0.0304%
UDNM	6 × 6	0.029 ± 0.0017%
CDP	8 × 8	3.597 ± 0.0065%
OSPF	8 × 8	0.623 ± 0.0011%
UDNM	8 × 8	0.054 ± 0.0078%
CDP	9 × 9	4.505 ± 0.0145%
OSPF	9 × 9	0.747 ± 0.0420%
UDNM	9 × 9	0.068 ± 0.0016%

Figure 17 shows the average signalling cost in bytes/number of packets exchanged in the discovery and information management exchanged/updated for each wireless nodes and access network devices.

As observed, UDNM has lower signalling cost and requires less bandwidth to control the management information when compared to BATMAN and OLSR protocols. BATMAN sends excessive Originator Messages (OMG), advertising the existence of devices. OLSR sends several topology control (TC) messages to discover and disseminate link state information. Both have a significant higher cost than UDNM in the network.

Figure 18 shows the average time to discover the wireless devices. UDNM requires a lower time ($\approx$ 40 ms) to discovery the wireless devices in the testbed scenario (total of 7 wireless devices according to the Fig. 11). OLSR spends more time to choose the Multi-Point-Relays (MPRs) nodes, which are the nodes that propagate the discovery messages in the network. In the case of BATMAN, it sends constantly discovery messages with 1 second interval, regardless if the network increases or decreases. In case of UDNM discovery, the initial exchange of discovery messages are configured in the bootstrapping process, and it is initially lower (0.5 seconds) than the adaptive HELLO limit L presented in Eq. 3. After exchanging a few discovery messages, the adaptive HELLO interval functionality is normalized and it sends HELLO messages without crossing the limits $L = 1$ second and $U = 5$ seconds. Thus, UDNM has clear advantages in terms of average time and number of discovered wireless nodes when compared to the baselines analysed.

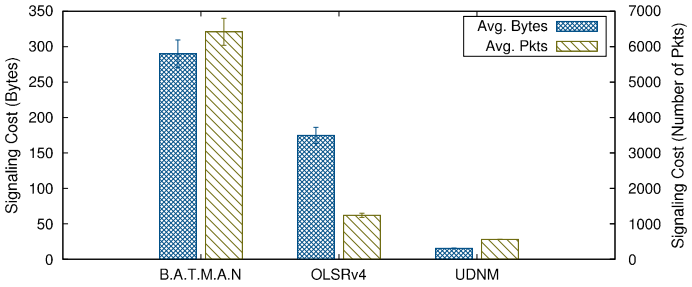

Fig. 17 Average signalling cost (bytes/number of packets) for each wireless device

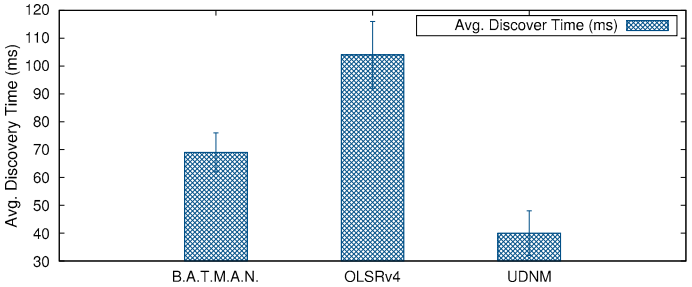

Fig. 18 Average time to discover the wireless devices

5.2.2 Wired and Wireless Networks

This section shows how UDNM behaves with multiple networks, wired and wireless. The performance of the UDNM is measured in terms of time for link error detection, recovery, propagation and update of the management information, average CPU overhead and time/number of discovered devices on the worst case scenario (with high background traffic load). To introduce background traffic, IPERF [50] is used in each device interface, injecting different levels of background traffic (e.g. low = 10%, medium = 50% and High = 80% of the bandwidth capacity of the link). The CPU consumption is measured through the HTOP command line tool. Both traffic injection and CPU load have an observation time of 5 min. The links capacity differs from wired and wireless scenarios, and the software IFTOP [51] is used to measure the real bandwidth capacity on the warm-up stage of the experiments, defining the correct traffic load levels. The wired grid size contains (5×5) 25 virtual devices (see Sect. 5). Notice that only UDNM is evaluated in this section, since it is the only one that is able to work in this environment.

The CDF of the average time for recovery/update management information for any device is shown in Fig. 19, assuming a link error originated from the wired virtual node V1 at the extremity of the grid (see Fig. 11). At a first glance, for both wired and wireless scenarios, UDNM spends less than 1 second to detect, propagate, recover and update the management information for all 32 nodes in the scenario. Therefore, immediately after nodes V2 and V6 detect a failure, they will propagate messages for the closest nodes (which represents 25% of the cases), by spending a maximum of 150 ms to recover/update around 3 hops distance. The other 75% of the cases are due to the increasing hop distance for message propagation where the propagation time will increase proportionally until it reaches the maximum number of hops (we settled the depth to 10 hops). In fact, it depends on where the failure is detected (V1 is chosen because it is an edge node). Therefore, the recovery messages propagation depth is configured at bootstrapping process and depends on the network size. Note that the virtual grid scenario shares the same physical

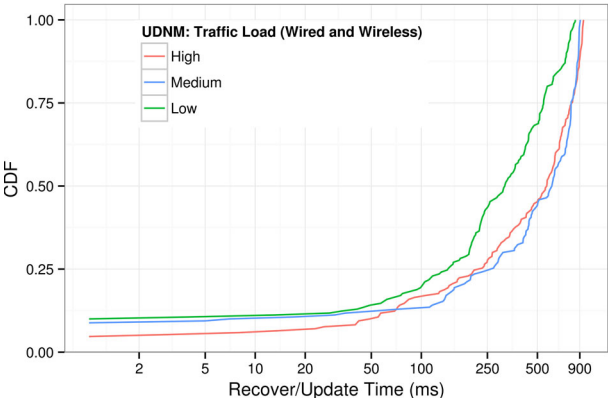

Fig. 19 Link recover/update time as a function of traffic load

resources for each virtual machine. For instance, if this test is replicated using only dedicated devices, it will surely take less time for messages propagation; this fact is proven in Fig. 18, which shows an average time of approximately 40 ms to discover all dedicated wireless nodes.

The impact of the CPU overhead is depicted in Fig. 20. Without UDNM and background traffic processes running, the reference of CPU load is approximately 0.061%. Only with background traffic running, the consumption is nearly 1, 2 and 3% for low traffic load, medium and high. With both UDNM and background traffic processes running, the CPU overhead increases to 3, 5 and 6%. Thus, UDNM shows to consume low CPU for executing the functionalities for discovery and exchange of management information.

The UDNM average time/number of discovered devices from different depths on top of the worst case scenario is quantified in Fig. 21. For depth equals to 2 hops, the UDNM spends 210 (ms) to discover approximately 8 devices, whereas for depths equal to 10 hops, UDNM spends 500 (ms) to discover approximately 30 devices. The factor that most influences the discovery time is explained according to the cooperation between devices to depths reuse functionality. For example, the devices do not search over the whole network until reaching the maximum depth, and just some of them update the management information gathered from the others, without making any further demands. This contributes to an efficient discovery process,

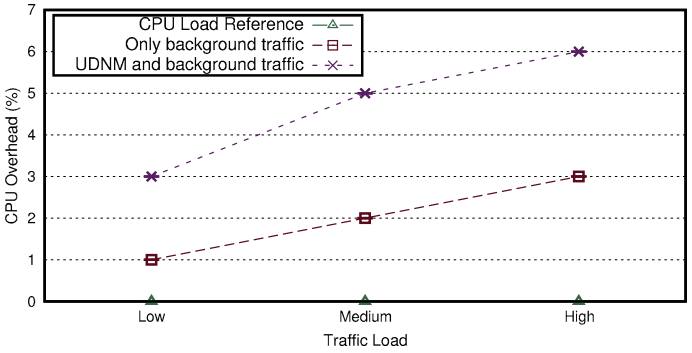

Fig. 20 Average CPU overhead as a function of traffic load

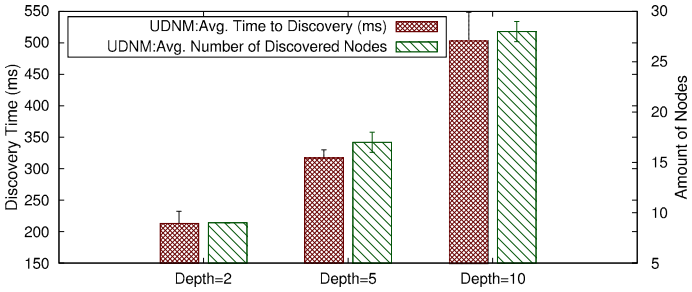

Fig. 21 Average time to discover and number of discovered devices with background traffic

according to the acquired knowledge of the network. Moreover, in order to maintain the information always updated, the partial views of each node consider the most recent information exchanged as well.

5.2.3 Path Decisions Analysis

To evaluate the autonomic decision component of the UDNM framework, the network equipments topology was used (see Figure 11). The metrics evaluated are the amount of data transferred between each peer and the time required to find the optimal path (the optimal path is always found). For each of the evaluation, 10 path requests were sent to the autonomic decision system, requesting a path to be established to nodes outside of the local agent known neighbourhood at different hop distances. For each assessed depth, the known neighbourhood depth value is modified from 1 Hop, which is the minimum known depth, up to 7 hops, which is the maximum known depth.

To choose a combination of values for the learning rate and discount factor, a series of tests were conducted to find the values that could work better for the autonomic system. These tests consist in simulating the decision system algorithm, in grid-shaped networks with different dimensions (NxN), also varying the known neighbourhood depth. Two different values were used for the Learning Rate and Discount Factor, creating four different configurations. Dimensions from 4 to 10 were tested, with neighbourhood depths from 1 to 8. The results are shown in Fig. 22, where the horizontal blue line in each graphic demarks the shortest path between the chosen source node and target node. The near the blue line the result is, the better. The decision of using the combination (LR: 0,9; DF:0,1) was taken based on the fact that it was the combination with the most near-to-the-best results.

Figure 23 presents the results for the average amount of exchanged data between agents, for path requests at different hop distances, with different known depth levels (Hops).

The Fig. 24 presents the results of the time required to find the optimal path with the required bandwidth, for path requests at different hop distances, with different known depth levels (Hops).

Through the observation of the results, it is possible to see a linear increase of both transmitted data and time values when the distance between peers increases. This is as expected since the decision process is being delegated from one agent to the next one, and the amount of total information to be transmitted increases with the increase of the knowledge depth.

However, it is important to notice what happens when the known topology depth is increased. Each time an agent increases the known depth by one hop, this means that the hop distance of the delegation process is increased. Therefore, instead of delegating hop by hop, the agents delegate at a multi-hop distance, which turns in similar results for distances in between the gateway agents. For example, when observing the results for the exchanged data, we can observe very similar results for path search using 2 hop depth (star symbol) for the known topology, for the distances (3,4), (5,6) and (7,8), since the distances 3, 5 and 7 are in between the gateways. The same can be observed when using 3 hop depth (triangle symbol) for

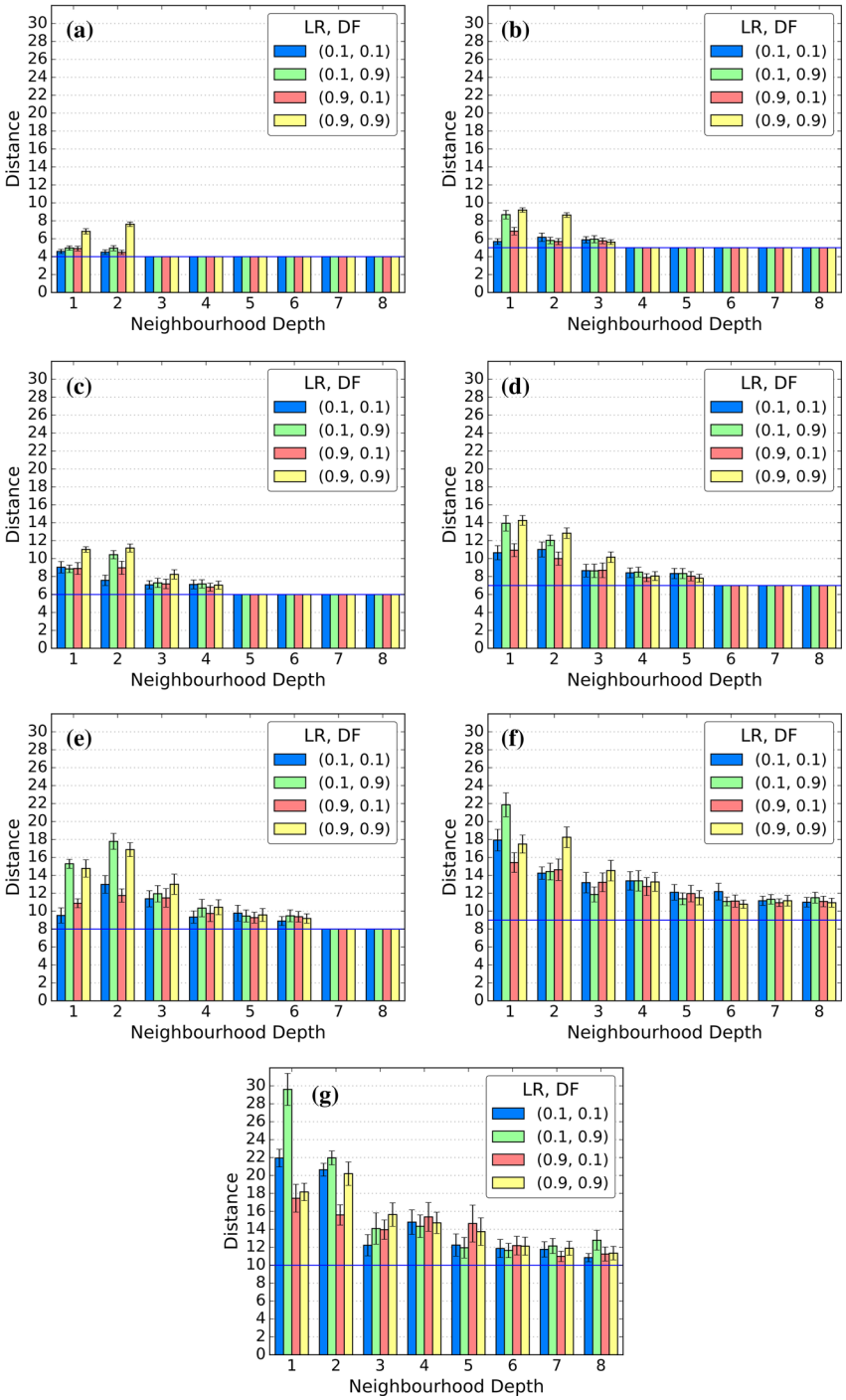

Fig. 22 Four different learning rate and discount factor combos for different network dimensions

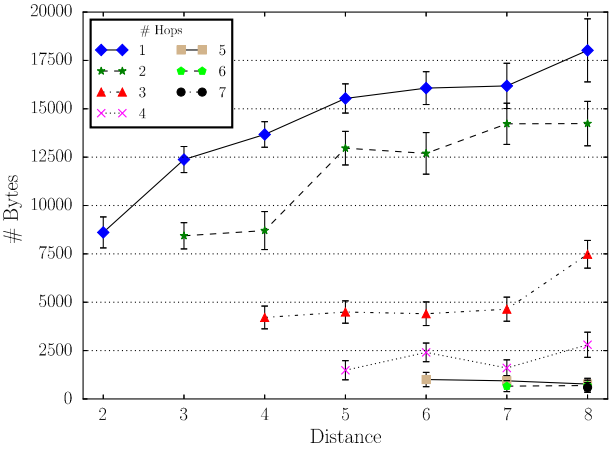

Fig. 23 Exchanged data for path search

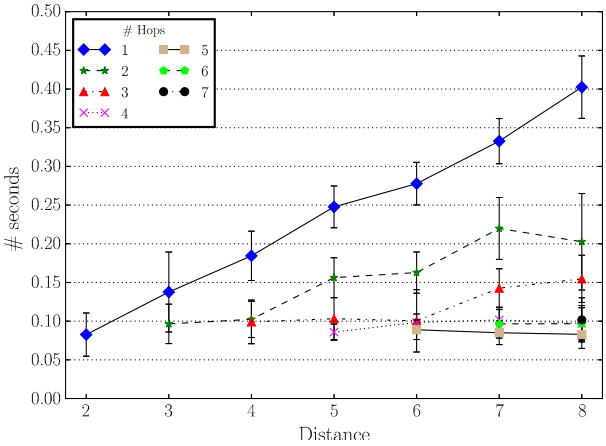

Fig. 24 Path search time

distances 4, 5 and 6. The same pattern can be observed in the results for the time analysis of a path search with the same configurations.

6 Conclusions

This paper proposed a Unified Distributed Network Management (UDNM) framework and its respective deployment on top of multi-network technologies platform. The UDNM framework brings benefits to the distributed management process and decisions, due to the possibility to control the amount of information to be exchanged in the network, by enabling cooperation between multi-network topologies, devices and technologies. The path exploration, learning and

maintenance workload are distributed through the nodes involved in the autonomic decision process, exchanging control messages only when needed, and using only partial information regarding the network state.

The performance analysis shows that UDNM framework has real conditions to unify distributed management and autonomic decisions functionalities, proving to be an easy-respond framework in terms of time to detect/recover and update faults, CPU overhead and average time/number of devices discovered, according to the acquired knowledge of the network. The distributed decisions are similar to the ones of the central traditional approach, while keeping the state information in the neighbourhood and without requiring a central element for the overall management and control.

Future work will research the autonomic approach with incomplete information in Software Defined Network (SDN) environments. Wireless Backhauls possess a natural challenge regarding the wireless links management. Weather conditions vary with little possibility to predicting its behaviour, affecting the backhaul normal operation and thus, creating scenarios where the information possessed by the network controller will not be accurate. SDN based wireless networks need to address such issue in order to maintain a proper operation [52].

Acknowledgements This work was supported by the *Fundação para a Ciência e Tecnologia* (FCT), through the grants SFRH/BD/62511/2009 and SFRH/BPD/108794/2015. The authors also thank the support provided by UBIQUIMESH PTDC/EEA-TEL/105472/2008 and the LoCoSDN projects.

References

1. Guardalben, L., Gomes, T., Pinho, A., Salvador, P., Sargento, S.: Nodes discovery in the in-network management communication framework. In: Mobile Networks and Management. Springer, Berlin, pp. 145–157. (2012). doi:10.1007/978-3-642-30422-4_11
2. Guardalben, L., Gomes, T., Sousa, R., Salvador, P., Sargento, S.: Lightweight discovery and exchange of network information in distributed network management. In: IFIP/IEEE International Symposium on Integrated Network Management (IM 2013), 2013, pp. 740–743. (2013). [Online]. Available: https://www.it.pt/Publications/PaperConference/13973
3. Babaoglu, O., Jelasity, M., Montresor, A., Fetzer, C., Leonardi, S, Van Moorsel, A., Van Steen, M. E.: Self-Star Properties in Complex Information Systems, vol. 3460.. Springer, Berlin (2005). doi:10.1007/b136551
4. Dobson, S., Zambonelli, F., Denazis, S., Fernández, A., Gaïti, D., Gelenbe, E., Massacci, F., Nixon, P., Saffre, F., Schmidt, N.: A survey of autonomic communications. ACM Trans. Auton. Adapt. Syst. **1**(2), 223–259 (2006). doi:10.1145/1186778.1186782
5. Brunner, M., Dudkowski, D., Mingardi, C., Nunzi, G.: Probabilistic decentralized network management. In: 2009 IFIP/IEEE International Symposium on Integrated Network Management. IEEE, June, pp. 25–32. (2009). doi:10.1109/INM.2009.5188783
6. Gazis, V., Kousaridas, A., Polychronopoulos, C., Raptis, T., Alonistioti, N.: Self-Management Capacities in Future Internet Wireless Systems. In: 2009 Computation World: Future Computing, Service Computation, Cognitive, Adaptive, Content, Patterns. IEEE, Nov 2009, pp. 9–14. (2009). doi:10.1109/ComputationWorld.2009.36
7. Raptis, T., Polychronopoulos, C., Kousaridas, A., Spapis, P., Gazis, V., Alonistioti, N., Chochliouros, I.: Technological enablers for self-manageable future internet elements. In: 2009 Computation World: Future Computing, Service Computation, Cognitive, Adaptive, Content, Patterns. IEEE, Nov 2009, pp. 499–504. (2009). doi:10.1109/ComputationWorld.2009.34
8. Niebert, N., Baucke, S., El-Khayat, I., Johnsson, M., Ohlman, B., Abramowicz, H., Wuenstel, K., Woesner, H., Quittek, J., Correia, L.M.: The way 4WARD to the creation of a future internet. In:

2008 IEEE 19th International Symposium on Personal, Indoor and Mobile Radio Communications. IEEE, Sep 2008, pp. 1–5. (2008). doi:10.1109/PIMRC.2008.4699967
9. Dudkowski, D., Brunner, M., Nunzi, G., Mingardi, C., Foley, C., de Leon, M., Meirosu, C., Engberg, S.: Architectural principles and elements of in-network management. In: 2009 IFIP/IEEE International Symposium on Integrated Network Management. IEEE, June 2009, pp. 529–536. (2009). doi:10.1109/INM.2009.5188858
10. Kim, S.-S., Choi, M.-J., Ju, H.-T., Ejiri, M., Hong, J.W.-K.: Towards management requirements of future internet. In: Challenges for Next Generation Network Operations and Service Management. Springer, Berlin, pp. 156–166 (2008). doi:10.1007/978-3-540-88623-5_16
11. Cisco: Building the Mobile Business with a Unified Wireless Network, Cisco, Technical Report, 2006. [Online]. Available: https://www.cisco.com/web/AP/wireless/pdf/-overview.pdf
12. Bahl, P., Chandra, R., Maltz, D., Patel, P., Padhye, J.: Towards Unified Management of Networked Services in Wired and Wireless Enterprise Networks, Microsoft, Technical Report (2008). [Online]. Available: http://research-srv.microsoft.com/pubs/70651/tr-2008-148.pdf
13. Hewlett-Packard: HP Unified Access: Unify Wired and Wireless Access at Your Own Pace. Hewlett-Packard, Technical Report (2014). [Online]. Available: http://www.techrepublic.com/resource-library/whitepapers/hp-unified-access-unify-wired-and-wireless-access-at-your-own-pace/
14. Jung, S.-J., Lee, J.-H., Han, Y.-J., Kim, J.-H., Na, J.-C., Chung, T.-M.: SNMP-Based Integrated Wire/wireless Device Management System. In: The 9th International Conference on Advanced Communication Technology. IEEE, Feb (2007), pp. 995–998. [Online]. doi:10.1109/ICACT.2007.358526
15. Guardalben, L., Condeixa, T., Gomes, T., Salvador, P., Sargento, S.: On the analysis of dissemination management information through an Eyesight perspective. In: 2013 IEEE Globecom Workshops (GC Wkshps). IEEE, Dec (2013), pp. 1001–1006. [Online]. doi:10.1109/GLOCOMW.2013.6825122
16. Prieto, A.G., Gillblad, D., Steinert, R., Miron, A.: Toward decentralized probabilistic management. IEEE Commun. Mag. **49**(7), 80–86 (2011). doi:10.1109/MCOM.2011.5936159
17. Sekar, V., Reiter, M.K., Willinger, W., Zhang, H., Kompella, R.R., Andersen, D.G.: CSAMP: A System for Network-Wide Flow Monitoring. Usenix—The Advanced COmputing Systems Association, (2008). [Online]. Available: https://www.cs.cmu.edu/~dga/papers/csamp-nsdi2008.pdf
18. Nobre, J.C., Granville, L.Z.: Consistency maintenance of policy states in decentralized autonomic network management. In: 2010 IEEE Network Operations and Management Symposium—NOMS 2010. IEEE, (2010), pp. 519–526. doi:10.1109/NOMS.2010.5488469
19. Nobre, J.C., Granville, L.Z., Clemm, A., Prieto, A.G.: Coordination in P2P management overlays to improve decentralized detection of SLA violations. In: 2013 IEEE International Conference on Communications (ICC). IEEE, June (2013), pp. 3456–3460. doi:10.1109/ICC.2013.6655084
20. Hadjiantonis, A., Pavlou, G.: Policy-based self-management of wireless ad hoc networks. In: Proceedings of the 11th IFIP/IEEE International Conference on Symposium on Integrated Network Management, (2009), pp. 796–802. [Online]. Available: http://dl.acm.org/citation.cfm?id=1688933.1689039
21. Andrea, D.P., Felipe, H., Diego, C., Saverio, N.: DECON: decentralized coordination for large-scale flow monitoring. In: 2010 INFOCOM IEEE Conference on Computer Communications Workshops. IEEE, March (2010), pp. 1–5. doi:10.1109/INFCOMW.2010.5466642
22. Yu, Q., Sung, C.W., Chan, T.H.: Repair topology design for distributed storage systems. In: 2012 IEEE International Conference on Communications (ICC). IEEE, June (2012), pp. 7009–7013. doi:10.1109/ICC.2012.6364721
23. Adrichem, N.L.V., Asten, B.J.V., Kuipers, F.A.: Fast recovery in software-defined networks. In: 2014 Third European Workshop on Software Defined Networks. IEEE, Sep (2014), pp. 61–66. doi:10.1109/EWSDN.2014.13
24. Chaparadza, R., Ben Meriem, T., Radier, B., Szott, S., Wodczak, M., Prakash, A., Jianguo Ding, Soulhi, S., Mihailovic, A.: Implementation guide for the ETSI AFI GANA model: a standardized reference model for autonomic networking, cognitive networking and self-management. In: 2013 IEEE Globecom Workshops (GC Wkshps). IEEE, Dec (2013), pp. 935–940. doi:10.1109/GLOCOMW.2013.6825110
25. Chaparadza, R., Ben Meriem, T., Radier, B., Szott, S., Wodczak, M., Prakash, A., Jianguo Ding, Soulhi, S., Mihailovic, A.: SDN enablers in the ETSI AFI GANA reference model for autonomic management & control (emerging standard), and virtualization impact. In: 2013 IEEE Globecom Workshops (GC Wkshps). IEEE, Dec 2013, pp. 818–823. doi:10.1109/GLOCOMW.2013.6825090

26. Chi, C., Huang, D., Lee, D., Sun, X.: Lazy flooding: a new technique for information dissemination in distributed network systems. IEEE/ACM Trans. Netw. **15**(1), 80–92 (2007). doi:10.1109/TNET.2006.890125
27. Drabkin, V., Friedman, R., Kliot, G., Segal, M.: RAPID: reliable probabilistic dissemination in wireless ad-hoc networks. In: 2007 26th IEEE International Symposium on Reliable Distributed Systems (SRDS 2007). IEEE, Oct (2007), pp. 13–22. doi:10.1109/SRDS.2007.9
28. Meghanathan, N.: An algorithm to determine the sequence of stable connected dominating sets in mobile ad hoc networks. In: Advanced Internatioanal Conference on Telecommunications and International Conference on Internet and Web Applications and Services (AICT-ICIW'06). IEEE, (2006), pp. 32–32. doi:10.1109/AICT-ICIW.2006.44
29. Sabbineni, H., Chakrabarty, K.: Location-aided flooding: an energy-efficient data dissemination protocol for wireless-sensor networks. IEEE Trans. Comput. **54**(1), 36–46 (2005). doi:10.1109/TC.2005.8
30. Ozkasap, O., Genc, Z., Atsan, E.: Epidemic-based reliable and adaptive multicast for mobile ad hoc networks. Comput. Netw. **53**(9), 1409–1430 (2009). doi:10.1016/j.comnet.2009.01.007
31. Chen, C., Ma, J., Salomaa, J.: Simulation study of cluster based data dissemination for wireless sensor networks with mobile sinks. In: 2008 10th International Conference on Advanced Communication Technology. IEEE, Feb 2008, pp. 231–236. doi:10.1109/ICACT.2008.4493751
32. Qiao, C., Xu, D.: Distributed partial information management (DPIM) schemes for survivable networks 1. In: Proceedings of the INFOCOM 2002. Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies. IEEE, vol. 1, pp. 302–311 (2002)
33. Leitao, J., Pereira, J., Rodrigues, L.: Hyparview: a membership protocol for reliable gossip-based broadcast. In: 37th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN'07), June 2007, pp. 419–429 (2007)
34. Holbert, B., Tati, S., Silvestri, S., Porta, T. L., Swami, A.: Network topology inference with partial path information. In: 2015 International Conference on Computing, Networking and Communications (ICNC), Feb 2015, pp. 796–802 (2015)
35. Moy, J.: OSPF Version 2. IETF, Technical Report, 1998. [Online]. Available: http://www.rfc-base.org/rfc-2328.html
36. Baccelli, E., Fuertes, C., Juan, A., Jacquet, P.: OSPF over multi-hop ad hoc wireless communications. Int. J. Comput. Netw. Commun. **2**(5), 37–56 (2010)
37. Baccelli, E., Jacquet, P., Nguyen, D., Clausen, T.: OSPF Multipoint Relay (MPR) Extension for Ad Hoc Networks. IETF, Technical Report, Feb (2009). doi:10.17487/RFC5449
38. Rodriguez, S.R.: Topology Discovery Using Cisco Discovery Protocol. Networking and Internet Architecture, Cornell University Library, July (2009). [Online]. Available: http://arxiv.org/abs/0907.2121
39. Clausen, T., Jacquet, P.: Optimized Link State Routing Protocol (OLSR). IETF, Technical Report, Oct (2003). doi:10.17487/RFC3626
40. Villasenor-Gonzalez, L., Ge, Ying, Lament, L.: HOLSR: a hierarchical proactive routing mechanism for mobile ad hoc networks. IEEE Commun. Mag. **43**(7), 118–125 (2005). doi:10.1109/MCOM.2005.1470838
41. Open-mesh. B.A.T.M.A.N.—Better Approach To Mobile Ad-hoc Networking (2012). [Online]. Available: http://tools.ietf.org/html/draft-wunderlich-openmesh-manet-routing-00
42. Prieto, A.G., Dudkowski, D., Meirosu, C., Mingardi, C., Nunzi, G., Brunner, M., Stadler, R.: Decentralized in-network management for the future internet. In: Proceedings of the 2009 IEEE International Conference on Communications Workshops, ICC 2009, pp. 1–5, June (2009). doi:10.1109/ICCW.2009.5207964
43. Bianchini, M., Scarselli, F.: On the complexity of neural network classifiers: a comparison between shallow and deep architectures. IEEE Trans. Neural Netw. Learn. Syst. **25**(8), 1553–1565 (2014). doi:10.1109/TNNLS.2013.2293637
44. Schmidhuber, J.: Deep learning in neural networks: an overview. Neural Netw. **61**, 85–117 (2015). doi:10.1016/j.neunet.2014.09.003
45. Cormen, T.H.: Introduction to Algorithms. Ilustrated MIT Press, Cambridge (2009). [Online]. Available: https://books.google.pt/books?id=aefUBQAAQBAJ
46. Sutton, R .S., Barto, A.G.: Reinforcement learning: an introduction. IEEE Trans. Neural Netw. **9**(5), 1054 (1998). doi:10.1109/TNN.1998.712192. (Publication of the IEEE Neural Networks Council)
47. Bertoni, G., Daemen, J., Bertoni, G., Daemen, G., Peeters, J., Van Assche, M.: The road from Panama to Keccak via RadioGatún. In: Handschuh, P., Helena, L., Stefan, P., Bart, R., (eds.)

Symmetric Cryptography. Leibniz-Zentrum fuer Informatik, Dagstuhl, pp. 1–9 (2009). [Online]. Available: http://drops.dagstuhl.de/opus/volltexte/2009/1958
48. Bertoni, G., Daemen, J., Peeters, M., Van Assche, G.: Keccak. In: DROPS, pp. 313–314. Springer, Berlin (2013). doi:10.1007/978-3-642-38348-9_19
49. Lee, K., Eidson, J.C., Weibel, H., Mohl, D.: IEEE 1588-Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems. IEEE, Technical Report (2005). [Online]. Available: https://standards.ieee.org/findstds/standard/1588-2008.html
50. Dugan, J., Elliott, S., Mah, B., Poskanzer, J., Prabhu, K.: IPerf—The Network Bandwidth Measurement Tool. [Online]. Available: https://iperf.fr/iperf-doc.php
51. Binnie, C.: Real-Time Network Statistics with Iftop. In: Practical Linux Topics. Apress, Berkeley, CA (2016), pp. 1–12. doi:10.1007/978-1-4842-1772-6_1
52. Pakzad, F., Portmann, M., Hayward, J.: Link capacity estimation in wireless software defined networks. In: 2015 International Telecommunication Networks and Applications Conference (ITNAC). IEEE, Nov (2015), pp. 208–213. doi:10.1109/ATNAC.2015.7366814

Carlos Ferreira is a Computer Engineering PhD Student at the University of Aveiro. He received his Master Degree in 2009 in Computers and Telemathics Engineering. Since then, he's a full-time researcher at Instituto de Telecomunicações - Aveiro. His interests include Distributed Management Protocols for both wireless and wired networks. Currently, he is focused in distributed management for Software Defined Networks.

Lucas Guardalben is a postdoctoral researcher at Instituto de Telecomunicações - Aveiro, Portugal. PhD at University of Aveiro, Portugal (April 2014); M.Sc. at Federal University of Santa Catarina, Florianópolis-SC, Brazil (February 2009). Author/co-author more than 30 scientific works with 8 years experience in R&D, and 4 years in projects collaboration and deployment of IT solutions. He has been involved in several national (Portuguese) and international projects (past: 4WARD, UBIQUIMESH, SELF, REMAR, GAPOTT, NOTTS, HANCAD, and current: Celtic-MONALIS, ULTRA-TV, SmartCityMules and OT2Delivery). His current research interests are related to future Internet architectures, ad hoc and mesh networks, social networks, autonomic and self-management, Over-the-Top services and delay tolerant networks.

Tomé Gomes is a researcher at Instituto de Telecomunicações - Aveiro. He conclude in July 2011 his Integrated M.Sc. in Electronics and Telecommunications Engineering from the Department of Electronics, Telecommunications, and Informatics of University of Aveiro. Since September 2011 he has joined Institute of Telecommunications, located in University of Aveiro Campus, as a researcher. He has been involved in several national projects, and his current research interests are related with next generation networks, self-management and cooperative networks.

Susana Sargento is an Associate Professor with "Habillitation" in the University of Aveiro and the Instituto de Telecomunicações - Aveiro, where she is leading the Network Architectures and Protocols (NAP) group. She has more than 15 years of experience in technical leadership in many national and international projects, and worked closely with telecom operators and OEMs. She has been involved in several FP7 projects (4WARD, Euro-NF, C-Cast, WIP, Daidalos, C-Mobile), EU Coordinated Support Action 2012-316296 "FUTURE-CITIES", national projects, and CMU-Portugal projects (S2MovingCity, DRIVE-IN with the Carnegie Melon University). She has been TPC-Chair and organized several international conferences and workshops, such as ACM MobiCom, IEEE Globecom and IEEE ICC. She has also been a reviewer of numerous international conferences and journals, such as IEEE Wireless Communications, IEEE Networks, IEEE Communications. Her main research interests are in the areas of self-organized networks, in ad-hoc and vehicular network mechanisms and protocols, such as routing, mobility, security and delay-tolerant mechanisms, resource management, and content distribution networks. In March 2012, Susana has co-founded a vehicular networking company, Veniam , a spin-off of the Universities of Aveiro and Porto, which builds a seamless low-cost vehicle-based internet infrastructure. Susana is the winner of the 2016 EU Prize for Women Innovators.

Paulo Salvador received in 2005 the PhD degree in Electrical Engineering from University of Aveiro, Portugal. In 1998, he joined Instituto de Telecomunicações - Aveiro, Portugal, as a researcher. In 2003 he joined the Department of Electronics, Telecommunications and Informatics of University of Aveiro, Portugal, being an Assistant Professor since 2006. His research activities have been focused on: (i) network design, management and security, and (ii) traffic, network and user behavior modeling.

Daniel Robalo received the Licenciado, MSc and PhD degrees in Electrical Engineering from University of Beira Interior, Covilhã, Portugal, in 2005, 2008 and 2014, respectively. He was the winner of the "ANACOM URSI Portugal 2014 Award" with the work "Enhanced Multi-Band Scheduling for Carrier Aggregation in LTE-Advanced Scenarios". He is currently a Case Handling engineer at Nokia.

Fernando J. Velez has been with the Department of Electromechanical Engineering of Universidade da Beira Interior, Covilhã, Portugal, where he is Assistant Professor, since 1995. He is also a researcher at Instituto de Telecomunicações - DEM. Fernando has been serving as the European IEEE VTS Chapter coordinator since 2010. His main research areas are cellular planning tools, traffic from mobility, cross-layer design, spectrum management, RF energy harvesting, wearable sensors and WBANS, and cost/revenue performance of advanced mobile communication systems.